PY32E407 Reference Manual

32-bit ARM® Cortex® -M4F Microcontroller

Puya Semiconductor (Shanghai) Co., Ltd.

Contents

1. List of abbreviations for registers

Abbreviation	Description
Read/Write (RW)	Software can read and write to this bit.
Read-only (R)	Software can only read this bit.
Write-only (W)	Software can only write to this bit.
Read/ClearWrite0 (RC_W0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on this bit.
Read/ClearWrite1(RC_W1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on this bit.
Read/ClearWrite(RC_W)	The software can read and clear the bit by writing to the register. The value written to this bit is not important.
Read/Clearby read (RC_R)	Software can read this bit. Reading this bit automatically clears it to 0. Writing this bit has no effect on the bit value.
Read/SetbyRead (RS_R)	Software can read this bit. Reading this bit automatically sets it to 1. Writing this bit has no effect on the bit value.
Read/Set (RS)	Software can read as well as set this bit by writing 1. Writing 0 has no effect on this bit.
Toggle (T)	The software can toggle this bit by writing 1. Writing 0 has no effect.
Reserve (Res)	Reserved bit, must be kept at reset value.

2. System architecture block

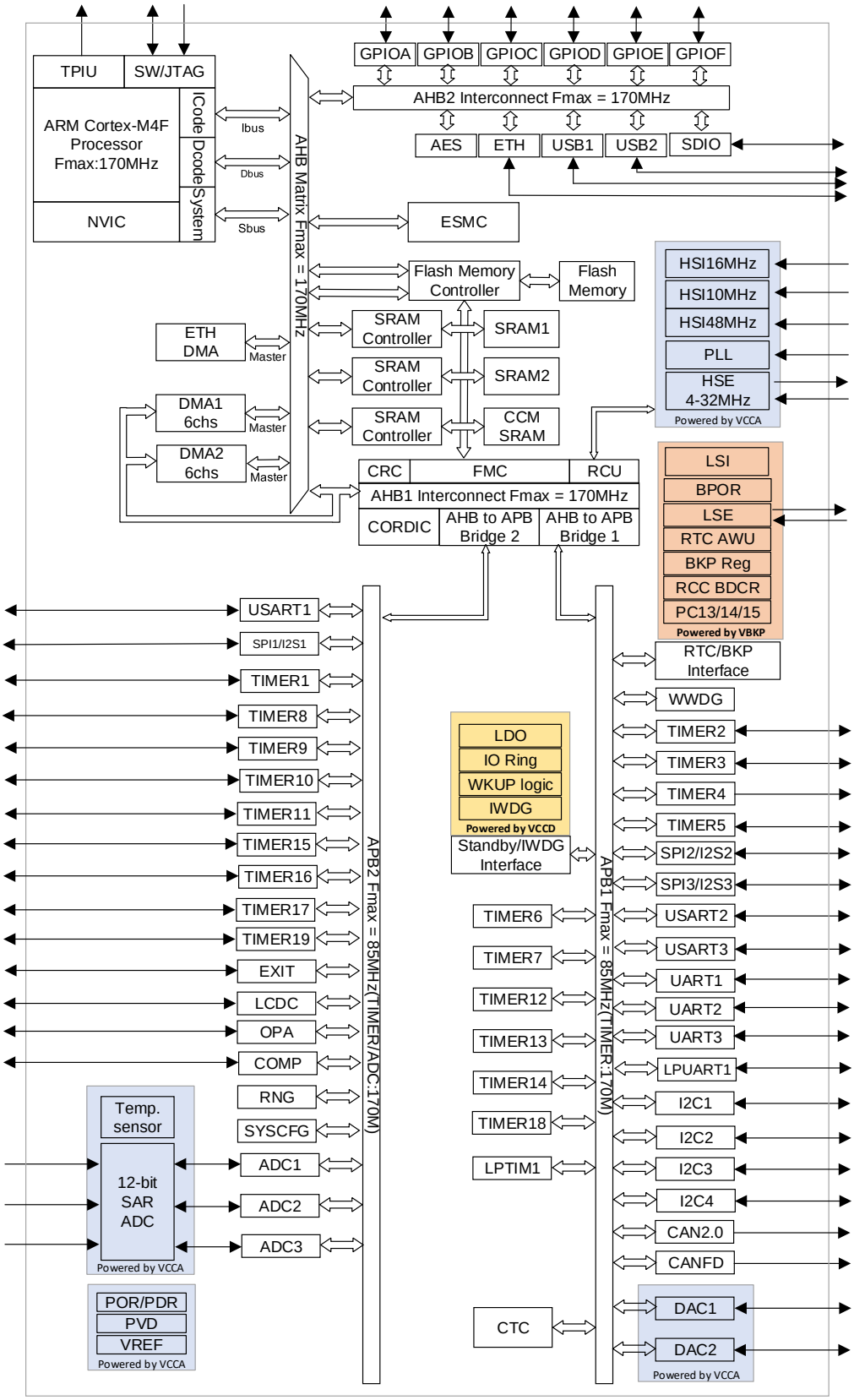


Figure 2-1 System block diagram

3. System and memory architecture

The PY32E407 series devices are based on Arm® Cortex® -32-bit general-purpose microcontroller for the-M4F processor. Arm® Cortex® The-M4F processor includes three AHB buses called I-CODE bus, D-Code bus and system bus. Cortex® All storage accesses of the-M4F processor, depending on different destinations and destination storage spaces, will be performed on these three buses. The organization of memory adopts Harvard architecture, pre-defined memory mapping and up to 4 GB of storage space, which fully guarantees the flexibility and scalability of the system.

3.1 Arm® Cortex® -M4F processor introduction

Cortex® The-M4F processor is a 32-bit processor with low interrupt latency time and low-cost debugging features. High integration and enhanced features make Cortex® The-M4F processor is suitable for those market segments that require high-performance and low-power microcontrollers. Cortex® The-M4F processor is based on the Armv7 architecture and supports a powerful and extensible instruction set, including general data processing I/O control tasks, enhanced data processing bit field operations, digital signal processing (DSP) and floating point arithmetic instructions. Listed below by Cortex® Some of the system peripherals supplied by M4F:

- Internal bus matrix for the interconnection of I-Code bus, D-Code bus, system bus, private bus (PPB) and debug private bus (AHB-AP)
- Nested Vector Interrupt Controller (NVIC)
- Flash address reload and breakpoint unit (FPB)
- Data observation point and tracking unit (DWT)
- Instrumentation trace macrocell (ITM)
- Embedded Tracking Macro Unit (ETM)
- Serial line and JTAG debug interface (SWJ-DP)
- Trace port interface unit (TPIU)
- Memory protection unit (MPU)
- Floating point unit (FPU)

3.2 System architecture

The system adopts a 32-bit multi-layer AHB bus matrix, which can ensure parallel communication between multiple masters and multiple slaves:

- Six main control buses:
  - Cortex® -M4F core I-Code bus, D-Code bus and system bus
  - DMA1 bus
  - DMA2 bus
  - ETH DMA Bus
- Eight controlled buses:
  - Internal Flash I-Code bus
  - Internal Flash D-Code bus
  - Internal SRAM1 bus
  - Internal SRAM2 bus

- Internal CCM SRAM bus
- AHB1 peripheral bus (including AHB to APB bus bridge and APB peripherals)
- AHB2 Peripheral Bus
- ESMC

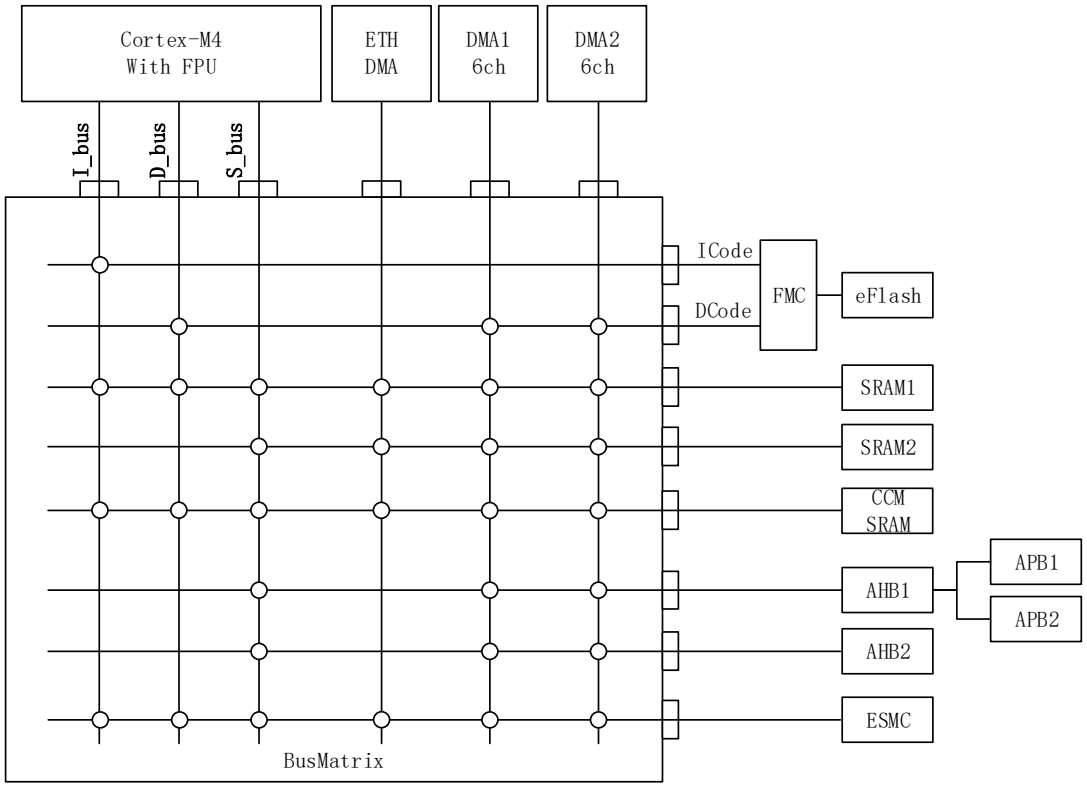


Figure 3-1 System architecture diagram

3.2.1 I_bus

This bus is used to connect the Cortex® The instruction bus of the-M4F core is connected to the bus matrix. The kernel fetches instructions through this bus. The object accessed by this bus is the memory containing the code (internal Flash/SRAM1/CCM SRAM or external memory via ESMC).

3.2.2 D_bus

This bus is used to connect the Cortex® The-M4F data bus is connected to the bus matrix. The kernel makes immediate digital load and debug access through this bus. The object accessed by this bus is the memory containing code or data (internal Flash/SRAM1/CCM SRAM or external memory via ESMC).

3.2.3 S_bus

This bus is used to connect the Cortex® The system bus of the-M4F core is connected to the bus matrix. This bus is used to access data located in peripherals or SRAM. The objects accessed by this bus are 144 KB of internal SRAM1/SRAM2/CCM SRAM, APB1 peripherals including AHB peripherals, APB2 peripherals, and external memory via ESMC.

3.2.4 DMA1/DMA2 BUS

This bus is used to connect the DMA memory bus master interface to the bus matrix. The objects accessed by DMA through this bus are 144 KB of internal SRAM1, SRAM2, CCM SRAM, APB1 peripherals including AHB peripherals, APB2 peripherals, and external memory through ESMC.

3.2.5 ETH DMA BUS

This bus is used to connect the ETH DMA memory bus master interface to the bus matrix. The objects accessed by ETH DMA via this bus are 144 KB of internal SRAM1, SRAM2, CCM SRAM, and external memory via ESMC.

3.2.6 Bus matrix

The bus matrix is used for access arbitration management between master buses. The arbitration uses a Round Robin algorithm. The system includes 6 Master devices, namely ARM® Cortex® I_BUS, D_BUS, S_BUS, DMA1, DMA2, and ETH DMA of-M4F; And eight Slave devices Slave, which are internal Flash memory, internal SRAM1, SRAM2, CCM SRAM, AHB1 peripheral (peripheral corresponding to AHB2APB bridge 1 and peripheral corresponding to AHB2APB bridge 2), AHB2 peripheral, and ESMC, respectively.

3.2.7 AHB/APB bus bridge

With two AHB/APB bus bridges, APB1 and APB2 can achieve a fully synchronized connection between the AHB bus and the two APB buses, allowing flexible peripheral frequency selection.

After each chip reset, all peripheral clocks are turned off (except SRAM and Flash interfaces). Before using a peripheral, its clock must be enabled in the RCC_AHBxENR or RCC_APBxENR register.

For AHB and APB peripheral address mapping information, refer to 3.3.2.

3.3 Memory organization architecture

3.3.1 Introduction

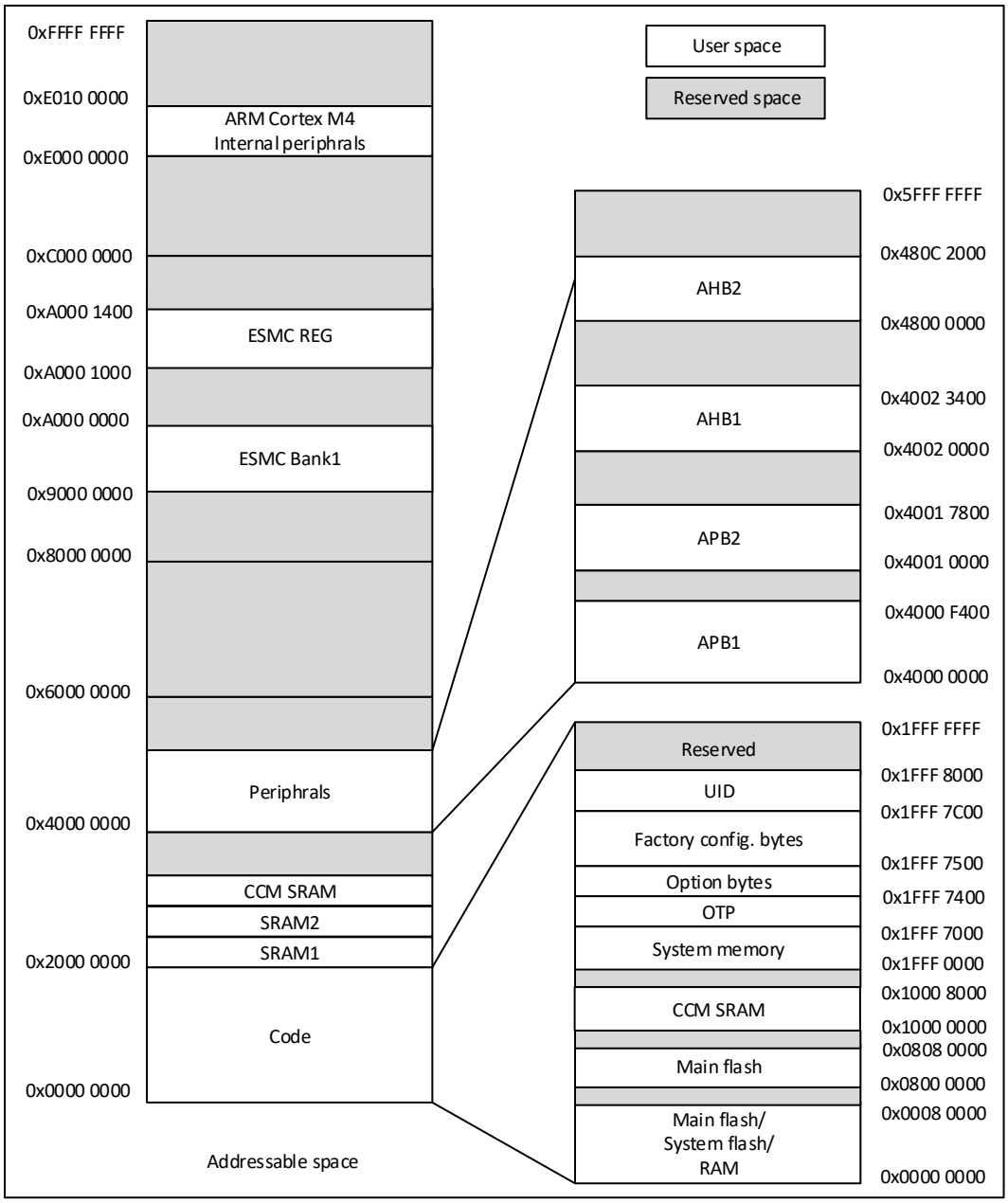
Arm® Cortex® The-M4F processor adopts Harvard architecture and can use independent buses to read instructions and load/store data. The instruction code and data are located in the same memory address space, but in different address ranges. Program memory, data memory, registers, and I/O ports are all within the same linear 4 GB address space.

The bytes are coded in memory in Little Endian format. The lowest numbered byte in a word is considered the word’s least significant byte and the highest numbered byte the most significant.

For details on peripheral register mapping, see the relevant sections.

The addressable storage space is divided into 8 main blocks, each of which is 512 MB.

3.3.2 Memory map



All other memory space not allocated to on-chip memory and peripherals is reserved address space. Refer to the table below for detailed storage and register space mapping.

The peripherals that can be used in the product and the corresponding address boundaries are given in the table below.

Table 3-1 Memory boundary address

Type	Boundary Address	Size	Memory Area	Description
SRAM	0x2002 4000 - 0x3FFF FFFF	511 MB	Reserved	1. When the CPU reads and writes to this space, a Response error is generated, and then a HardFault exception is entered 2. A TEIF status bit is generated when DMA access.
	0x2000 0000 - 0x2002 3FFF	144 KB	SRAM	If the hardware power-on configuration SRAM is 144 KB, the SRAM address space is 0x2000 0000-0x20023FFF
Code	0x1FFF 8000 - 0x1FFF FFFF	32 KB	Reserved	-
	0x1FFF 7E00 - 0x1FFF 7FFF	512 Bytes	UID bytes	Unique ID
	0x1FFF 7600 - 0x1FFF 7DFF	2 KB	Factory config. bytes	-
	0x1FFF 7400 - 0x1FFF 75FF	512 Bytes	Option bytes	Software and hardware option bytes information
	0x1FFF 7000 - 0x1FFF 73FF	1 KB	OTP	-
	0x1FFF 0000 - 0x1FFF 6FFF	28 KB	System memory	Store Boot loader
	0x1008 0000 - 0x1FFE FFFF	256 MB	Reserved	-
	0x1000 0000 - 0x1000 7FFF	32 KB	CCM SRAM	-
	0x0808 0000 - 0x0FFF FFFF	127 MB	Reserved	-
	0x0800 0000 - 0x0807 FFFF	512 KB	Main flash memory	-
	0x0008 0000 - 0x07FF FFFF	127 MB	Reserved	1. When the CPU reads and writes to this space, a Response error is generated, and then a Hard fault exception is entered; 2. A TEIF status bit is generated when DMA access.
	0x0000 0000 - 0x0007 FFFF	512 KB	Depending on the Boot configuration selection: 1) Main flash memory System memory 3) SRAM	-

1. The above space is marked as a **reserved** space and cannot be written. It is read to 0 and a response error is generated.

Table 3-2 Peripheral register address

Bus	Memory boundary address	Peripheral
AHB3	0xA000 1000 - 0xA000 13FF	ESMC
AHB2	0x480C 2000 - 0x5FFF FFFF	Reserved
	0x480C 0000 - 0x400C 1FFF	ETH
	0x4808 0000 - 0x480B FFFF	USB2 OTG FS
	0x4804 0000 - 0x4807 FFFF	USB1 OTG FS
	0x4800 2800 - 0x4803 FFFF	Reserved
	0x4800 2400 - 0x4800 27FF	AES
	0x4800 2000 - 0x4800 23FF	SDIO
	0x4800 1800 - 0x4800 1FFF	Reserved
	0x4800 1400 - 0x4800 17FF	GPIOF
	0x4800 1000 - 0x4800 13FF	GPIOE
	0x4800 0C00 - 0x4800 0FFF	GPIOD

Bus	Memory boundary address	Peripheral
	0x4800 0800 - 0x4800 0BFF	GPIOC
	0x4800 0400 - 0x4800 07FF	GPIOB
	0x4800 0000 - 0x4800 03FF	GPIOA
AHB1	0x4002 3400 - 0x4002 FFFF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2400 - 0x4002 2FFF	Reserved
	0x4002 2000 - 0x4002 23FF	FMC
	0x4002 1400 - 0x4002 1FFF	Reserved
	0x4002 1000 - 0x4002 13FF	RCC
	0x4002 0C00 - 0x4002 0FFF	CORDIC
	0x4002 0800 - 0x4002 0BFF	Reserved
	0x4002 0400 - 0x4002 07FF	DMA2
	0x4002 0000 - 0x4002 03FF	DMA1
APB2	0x4001 7800 - 0x4001 FFFF	Reserved
	0x4001 7400 - 0x4001 77FF	LCDC
	0x4001 7000 - 0x4001 73FF	OPA
	0x4001 6C00 - 0x4001 6FFF	COMP
	0x4001 6800 - 0x4001 6BFF	RNG
	0x4001 6400 - 0x4001 67FF	TIMER19
	0x4001 6000 - 0x4001 63FF	TIMER17
	0x4001 5C00 - 0x4001 5FFF	TIMER16
	0x4001 5800 - 0x4001 5BFF	TIMER15
	0x4001 5400 - 0x4001 57FF	TIMER11
	0x4001 5000 - 0x4001 53FF	TIMER10
	0x4001 4C00 - 0x4001 4FFF	TIMER9
	0x4001 4000 - 0x4001 4BFF	Reserved
	0x4001 3C00 - 0x4001 3FFF	ADC3
	0x4001 3800 - 0x4001 3BFF	USART1
	0x4001 3400 - 0x4001 37FF	TIMER8
	0x4001 3000 - 0x4001 33FF	SPI1/I2S1
	0x4001 2C00 - 0x4001 2FFF	TIMER1
	0x4001 2800 - 0x4001 2BFF	ADC2
	0x4001 2400 - 0x4001 27FF	ADC1
	0x4001 0800 - 0x4001 23FF	Reserved
	0x4001 0400 - 0x4001 07FF	EXTI
	0x4001 0000 - 0x4001 03FF	SYSCFG
APB1	0x4000 F400-0x4000 FFFF	Reserved
	0x4000 E000 - 0x4000 F3FF	CAN2.0
	0x4000 CC00-0x4000 DFFF	CANFD
	0x4000 C800 - 0x4000 CBFF	CTC
	0x4000 B400 - 0x4000 C7FF	Reserved
	0x4000 AC00 - 0x4000 B3FF	CANMEM
	0x4000 8800 - 0x4000 ABFF	Reserved
	0x4000 8400 - 0x4000 87FF	I2C4
	0x4000 8000 - 0x4000 83FF	LPUART1
	0x4000 7C00 - 0x4000 7FFF	LPTIM1
	0x4000 7800 - 0x4000 7BFF	I2C3

Bus	Memory boundary address	Peripheral
	0x4000 7400 - 0x4000 77FF	DAC
	0x4000 7000 - 0x4000 73FF	PWR
	0x4000 6D00 - 0x4000 6FFF	Reserved
	0x4000 6C00 - 0x4000 6CFF	BKP
	0x4000 6000 - 0x4000 6BFF	Reserved
	0x4000 5C00 - 0x4000 5FFF	UART3
	0x4000 5800 - 0x4000 5BFF	I2C2
	0x4000 5400 - 0x4000 57FF	I2C1
	0x4000 5000 - 0x4000 53FF	UART2
	0x4000 4C00 - 0x4000 4FFF	UART1
	0x4000 4800 - 0x4000 4BFF	USART3
	0x4000 4400 - 0x4000 47FF	USART2
	0x4000 4000 - 0x4000 43FF	Reserved
	0x4000 3C00 - 0x4000 3FFF	SPI3/I2S3
	0x4000 3800 - 0x4000 3BFF	SPI2/I2S2
	0x4000 3400 - 0x4000 37FF	Reserved
	0x4000 3000 - 0x4000 33FF	IWDG
	0x4000 2C00 - 0x4000 2FFF	WWDG
	0x4000 2800 - 0x4000 2BFF	RTC
	0x4000 2400 - 0x4000 27FF	TIMER18
	0x4000 2000 - 0x4000 23FF	TIMER14
	0x4000 1C00 - 0x4000 1FFF	TIMER13
	0x4000 1800 - 0x4000 1BFF	TIMER12
	0x4000 1400 - 0x4000 17FF	TIMER7
	0x4000 1000 - 0x4000 13FF	TIMER6
	0x4000 0C00 - 0x4000 0FFF	TIMER5
	0x4000 0800 - 0x4000 0BFF	TIMER4
	0x4000 0400 - 0x4000 07FF	TIMER3
	0x4000 0000 - 0x4000 03FF	TIMER2

1. AHB in the above table is marked as a **reserved** address space, which cannot be written. The readback is 0, and a Hard fault is generated.

3.4 Embedded SRAM

PY32E407 series devices have built-in static SRAM with a maximum of 144 KB bytes. According to different product definitions, the SRAM capacity will vary. Please refer to the data sheet for details. It can be accessed in bytes, half words (16 bits), or full words (32 bits). The start address of the SRAM is 0x2000 0000.

96K SRAM1 (mapped to address 0x2000 0000)

16K SRAM2 (mapped to address 0x2001 8000)

32K CCM SRAM (mapped to address 0x1000 0000 and end 0x2001 C000 of SRAM2)

when boot fromSRAM1 is selected or when physical remap is selected,it is accessed by the CPU through I-Code/D-Code bus for maximum performance.

The CCM SRAM SRAM mapped at address 0x1000 0000.

Due to accessed by I-Code bus for maximum performance from CCM SRAM without any remapping.

The CCM SRAM succeeds the address at the end SRAM, providing contiguous address space with SRAM and SRAM.

3.4.1 Parity check

The first 32 KB of SRAM1 and the entire CCM SRAM comes with parity.

The user can enable parity using the option bit SRAM_PE in the user options byte.

The data bus width is 36 bits because 4 bits are available for parity (1 bit per byte).

Parity bits are automatically calculated and stored when written to SRAM. Then, it is automatically checked when reading. If 1 bit fails, an NMI end interrupt is generated. Errors can also be linked to BRK_IN of TIM1/TIM8/TIM15/TIM16/TIM17.

Note: When SRAM parity is enabled, it is recommended to initialize the SRAM with software to avoid parity errors when reading.

3.4.2 CCM SRAM Write Protection

CCM SRAM can be write-protected with a page granularity of 1 K bytes.

Write protection can be enabled in the SYSCFG CCM SRAM write protection register (SYSCFG_SWPR) in SYSCFG. This is a register with a one-time write "1" mechanism, which means that "write protection" is set for this page of SRAM by a one-time write "1" and this setting can only be cleared by a system reset.

Table 3-3 CCM SRAM configuration

Page number	Start address	End address
Page 0	0x1000 0000 / 0x2001 C000	0x1000 03FF / 0x2001 C3FF
Page 1	0x1000 0400 / 0x2001 C400	0x1000 07FF / 0x2001 C7FF
Page 2	0x1000 0800 / 0x2001 C800	0x1000 0BFF/0x2001 CBFF
Page 3	0x1000 0C00 / 0x2001 CC00	0x1000 0FFF/0x2001 CFFF
Page 4	0x1000 1000 / 0x2001 D000	0x1000 13FF / 0x2001 D3FF
Page 5	0x1000 1400 / 0x2001 D400	0x1000 17FF / 0x2001 D7FF
Page 6	0x1000 1800 / 0x2001 D800	0x1000 1BFF/0x2001 DBFF
Page 7	0x1000 1C00 / 0x2001 DC00	0x1000 1FFF/0x2001 DFFF
Page 8	0x1000 2000 / 0x2001 E000	0x1000 23FF / 0x2001 E3FF
Page 9	0x1000 2400 / 0x2001 E400	0x1000 27FF / 0x2001 E7FF
Page 10	0x1000 2800 / 0x2001 E800	0x1000 2BFF/0x2001 EBFF
Page 11	0x1000 2C00 / 0x2001 EC00	0x1000 2FFF/0x2001 EFFF
Page 12	0x1000 3000 / 0x2001 F000	0x1000 33FF / 0x2001 F3FF
Page 13	0x1000 3400 / 0x2001 F400	0x1000 37FF / 0x2001 F7FF
Page 14	0x1000 3800 / 0x2001 F800	0x1000 3BFF/0x2001 FBFF
Page 15	0x1000 3C00 / 0x2001 FC00	0x1000 3FFF/0x2001 FFFF
Page 16	0x1000 4000/0x2002 0000	0x1000 43FF / 0x2002 03FF
Page 17	0x1000 4400/0x2002 0400	0x1000 47FF / 0x2002 07FF
Page 18	0x1000 4800/0x2002 0800	0x1000 4BFF / 0x2002 0BFF
Page 19	0x1000 4C00 / 0x2002 0C00	0x1000 4FFF / 0x2002 0FFF
Page 20	0x1000 5000/0x2002 1000	0x1000 53FF / 0x2002 13FF

Page 21	0x1000 5400/0x2002 1400	0x1000 57FF / 0x2002 17FF
Page 22	0x1000 5800/0x2002 1800	0x1000 5BFF / 0x2002 1BFF
Page 23	0x1000 5C00 / 0x2002 1C00	0x1000 5FFF / 0x2002 1FFF
Page 24	0x1000 6000/0x2002 2000	0x1000 63FF / 0x2002 23FF
Page 25	0x1000 6400/0x2002 2400	0x1000 67FF / 0x2002 27FF
Page 26	0x1000 6800/0x2002 2800	0x1000 6BFF / 0x2002 2BFF
Page 27	0x1000 6C00 / 0x2002 2C00	0x1000 6FFF / 0x2002 2FFF
Page 28	0x1000 7000/0x2002 3000	0x1000 73FF / 0x2002 33FF
Page 29	0x1000 7400/0x2002 3400	0x1000 77FF / 0x2002 37FF
Page 30	0x1000 7800/0x2002 3800	0x1000 7BFF / 0x2002 3BFF
Page 31	0x1000 7C00 / 0x2002 3C00	0x1000 7FFF / 0x2002 3FFF

3.4.3 CCM SRAM Read protection

The CCM SRAM has read protection (RDP). For more information, see4.10.1: Flash Read Protection (RDP).

3.4.4 CCM SRAM Erasure

The CCM SRAM may use the option configuration bit CCMSRAM_RST toerase the content at the time of reset (refer to 4.8.1the Flash option byte description). The CCMSR control bit implementation can also be set using software methods (refer to10.2.22 the SYSCFG CCM SRAM Control and Status Register (SYSCFG_SCSR)).

3.5 Embedded Flash

Flash memory consists of two different physical areas:

- The Main flash area, 512 KB, consists of two Banks, which contain application and user data. According to different product definitions, Flash capacity will vary. Please refer to the data sheet for details. Used to store user programs and user data. Software access to space outside the set range will produce a Hard fault.
- Information area, 32 KB, consists of the following sections:
  - Factory config. bytes: 2048 Bytes
  - UID: 512 Bytes, used to store the UID of the chip
  - OTP: 1024 Bytes, used to store user OTP data
  - Option bytes: 256 Bytes, used to store configuration values for chip hardware and storage protection
  - System memory: 28 KB for the Boot loader

Flash interface realizes instruction reading and data access based on AHB protocol, and it realizes the basic program/erase operations of Flash through registers.

3.6 Bit segment

Cortex ® The-M4F memory map includes two bit segment regions. These regions map each word in the memory alias region to a corresponding bit in the memory bit segment region. When a word is written in the alias area, it is equivalent to performing a read-modify-write operation on the target bit of the bit segment area.

In the PY32E407 device, both peripheral registers and SRAM are mapped to a bit region, which enables read and write operations of a single bit region. These operations are only available for Cortex® -M4F access, not valid for other bus master interfaces such as DMA.

The correspondence between each word in the alias region and each bit in the bit segment region may be described by a mapping formula. The mapping formula is:

Bit_word_addr = bit_band_base + (byte_offset x 32) + (bit_number x 4)

with:

- Bit_word_addr represents the address of the word that will be mapped to the target bit in the alias area
- Bit_band_base represents the starting address of the alias zone
- Byte_offset represents the byte number in the bit segment area where the target bit is located
- Bit_number represents the bit position of the target bit (0-7)

example

The following example illustrates how to map bit 2 of a byte at SRAM address 0x2000 0300 to an alias region:

0x2200 6008 = 0x2200 0000 + (0x300 * 32) + (2 * 4)

Performing a write operation to address 0x2200 6008 is equivalent to performing a read-modify-write operation at bit 2 of the byte at SRAM address 0x2000 0300.

Performing a read operation to address 0x2200 6008 will return the value of bit 2 of the byte at SRAM address 0x2000 0300 (0x01 for position bit, 0x00 for bit reset).

For more information about bit segments, see Cortex ® -M4F Programming Manual (DUI508B_cortex_m4_ugrm.pdf).

3.7 Start Configuration

By configuring the bits BOOT0 pin, nBOOT0/nBOOT1/nSWBOOT0 (stored in option bytes), you can select three different boot modes, as shown in the following table:

Table 3-4 Boot configuration

Boot mode configuration					Mode
BOOT_Lock	nBOOT1 FLASH_OPTR2[8]	nBOOT0 FLASH_OPTR2[14]	BOOT0 Pin PB8	nSWBOOT0 FLASH_OPTR2[13]	
1	X	X	X	X	Boot from Main flash
0	X	X	0	1	Boot from Main flash
0	X	1	X	0	Boot from Main flash
0	0	X	1	1	Boot from SRAM1
0	0	0	X	0	Boot from SRAM1
0	1	X	1	1	Boot from System flash
0	1	0	X	0	Boot from System flash

The Boot loader program is stored in the System memory and is used to download Flash programs through USART, USB, SPI, and I^2C interfaces.

After system reset, the value of the BOOT0 pin or the user option configuration bit (depending on the value of the nSWBOOT0 bit in the FLASH_OPTR register) and the nBOOT1 bit are latched. When you exit from standby mode, the value of the BOOT pin will be re-latched; Therefore, the BOOT pin should remain in the required BOOT configuration in standby mode.

Depending on the selected boot mode, the main flash memory, the system memory, or the SRAM 1 may be accessed in the following manner:

- Boot from the main flash memory: The main flash memory is mapped to the boot space (0x0000

0000), but it can still be accessed at its original address (0x0800 0000), i.e. the contents of the flash memory can be accessed in two address areas, 0x0000 0000 or 0x0800 0000.

- Boot from system memory: System memory is mapped to boot space (0x0000 0000), but it can still be accessed at its original address (0x1FFF 0000).
- Boot from built-in SRAM1: SRAM1 is mapped to the boot space (0x0000 0000), but it can still be accessed at its original address (0x2000 0000).

After boot, the CPU acquires the address at the top of the stack from the address 0x0000 0000, and starts executing code from the address indicated by 0x0000_0004 of the boot memory. Because of the fixed memory image, the code area always starts at address 0x0000_0000 (accessed via the ICode and DCode buses), while the data area (SRAM) always starts at address 0x2000_0000 (accessed via the system bus). The CPU of the Cortex M4F always gets the reset vector from the ICode bus, i.e. boot is only suitable for starting from the code area (typically boot from Flash). The PY32E407 series microcontrollers implement a special mechanism that can be booted from on-chip SRAM. When booting from on-chip SRAM, in the initialization code of the application, the NVIC's exception table and offset register must be used to remap the vector table into SRAM.

3.7.1 System Storage Startup

The system memory contains an embedded boot loader that can be used to program the flash memory. The boot loader reprograms the flash memory through the USART or USB interface.

3.7.2 Physical remap

When the boot pin is selected, the application software can set some memory to be accessed from the code space (so that code can be executed over the ICode bus instead of the system bus). Such modifications are achieved by programming SYSCFG.MEM_ MODE in the SYSCFG controller. The following memories can thus be remapped:

- Main flash
- System memory
- Embedded SRAM1
- ESMC

Table 3-5 Physical remapping address

Address	Main Flash Start/Remap	Embedded SRAM1 Start/Remap	System memory Start/Remap	ESMC Remapping
0x2000 0000 - 0x2001 7FFF	SRAM1	SRAM1	SRAM1	SRAM1
0x1FFF 0000 - 0x1FFF 7FFF	System memory	System memory	System memory	System memory
0x0808 0000 - 0x0FFF FFFF	Reserved	Reserved	Reserved	Reserved
0x0800 0000 - 0x0807 FFFF	Flash	Flash	Flash	Flash
0x0400 0000 - 0x07FF FFFF	Reserved	Reserved	Reserved	ESMC
0x0000 0000 - 0x03FF FFFF	Flash (using aliases)	SRAM1 (using aliases)	System memory (28 KB) (using aliases)	ESMC

4. Embedded Flash interface

4.1 Introduction

The Flash interface can manage the CPU to access Flash through the AHB I-Code and D-Code buses. The interface can be executed against Flash

Erase and program operations, and implement read and write protection mechanisms.

The Flash memory interface accelerates code execution with a system of instruction prefetch and cache lines.

4.2 Main Features

- Memory organization:
  - Main memory: Up to 512 KB
  - Information memory: 32 KB
  - Page size: 512 Bytes
  - Sector size: 4 KB
  - Block size: 64 KB
  - 137-bit wide data read (128 bits data+9 bits ecc)
  - 137-bit wide data write (128 bits data+9 bits ecc)
- Flash read operation
- Flash programming operations (page programming)
- Flash erase operation (page erase/sector erase/block erase/full erase)
- Read/write protection
- Prefetch operations on I-Code
- Branch caching on I-Code
- Data caching on D-Code
- Option byte loading function

4.3 Flash memory organization

Flash memory consists of a 137-bit wide (128 bits data + 9 bits ecc) cell, has a dual-Bank architecture that supports read-write function (RWW), and can be used as storage for programs and data. Page size is 512 Bytes, Sector size is 4 KB, and Block size is 64 KB.

Functionally, Flash memory is divided into Main flash and Information flash. The former has a maximum capacity of 512 KB, and the latter has a capacity of 32 KB, as follows:

Table 4-1 Flash memory structure and boundary address

Area	Space Name				Address	Size (bytes)
Main memory	Bank0	Block 0	Sector 0	Page 0-7	0x0800 0000 - 0x0800 0FFF	4 °K
			Sector 1	Page 8-15	0x0800 1000 - 0x0800 1FFF	4 °K
			…	…	…	…
			Sector 15	Page 120-127	0x0800 F000-0x0800 FFFF	4 °K
		…	…	…	…	…
		Block 3	Sector 48	Page 384-391	0x0803 0000 - 0x0803 0FFF	4 °K
			Sector 49	Page 392-399	0x0803 1000 - 0x0803 1FFF	4 °K
			…	…	…	…
			Sector 63	Page 504-511	0x0803 F000-0x0803 FFFF	4 °K
	Bank1	Block 4	Sector 64	Page 512-519	0x0804 0000 - 0x0804 0FFF	4 °K
			Sector 65	Page 520-527	0x0804 1000 - 0x0804 1FFF	4 °K

			…	…	…	…
			Sector 79	Page 632-639	0x0804 F000-0x0804 FFFF	4 °K
		…	…	…	…	…
		Block 7	Sector 112	Page 896-903	0x0807 0000 - 0x0807 0FFF	4 °K
			Sector 113	Page 904-911	0x0807 1000 - 0x0807 1FFF	4 °K
			…	…	…	…
			Sector 127	Page 1016-1023	0x0807 F000-0x0807 FFFF	4 °K
Informatin block	Bank0	System memory			0x1FFF 0000 - 0x1FFF 6FFF	28 °K
		OTP			0x1FFF 7000 - 0x1FFF 73FF	1 °K
		Option bytes			0x1FFF 7400 - 0x1FFF 75FF	512
		Factory config. bytes			0x1FFF 7600 - 0x1FFF 7DFF	2 °K
		UID			0x1FFF 7E00 - 0x1FFF 7FFF	512

4.4 Error code correction (ECC)

The data in the flash memory is 137 bits: each 128 bits add 9 bits ECC.

The ECC mechanism supports:

- One-bit error detection and correction
- Two-bit error detection

When a one-bit erroris detected and corrected, the flag ECCC (ECC Correction) is set in the FLASH_ECCR register. If ECCCIE is also set, an interrupt will be generated.

When a two-bit error is detected, the flag ECCD (ECC detection) is set in the FLASH_ECCR register. In this case, an NMI interrupt is generated.

When an ECC error is detected, the failure address and its associated Bank are saved in ADDR_ECC, SYSF_ECC, and BK_ECC in the FLASH_ECCR register.

When ECCC or ECCDis set, ADDR_ECC, SYSF_ECC, and BK_ECC are not updated if a new ECC error occurs. FLASH_ECCR is updated only afterthe ECC flag is cleared.

Note: When accessing the initial data: 0x1FF FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF, a one-bit error is detected and corrected.

4.5 Read access latency

In ordertocorrectly read instructions/data from the Flash memory, it is necessary to set the number of waiting cycles (Latency) of the register Flash_ACR according to the frequency of the CPU clock (HCLK).

After chip reset (including power-on reset, system reset, and exit Standby reset), the HCLK clock frequency is 16 MHz and the number of Flash read cycles is 0.

When changing the CPU clock frequency or range, you must follow the following software steps to adjust the number of waiting cycles required to access Flash memory.

Operation steps when increasing CPU frequency

1. Program a new number of wait cycles to the LATENCY bit in the FLASH_ACR register
2. Check whether the new waiting period is set successfully by reading the FLASH_ACR register
3. Modify the CPU clock source by overwriting the SW bit in the RCC_CFGR register
4. If necessary, the CPU clock prescaler can be modified by rewriting the HPRE bit in the RCC_CFGR
5. By reading the corresponding clock source state (SWS bit) and/or AHB prescalation value in the RCC_CFGR register

(HPRE bit), check whether the new CPU clock source and/or the new CPU clock prescalation value is successfully set

Operation procedures when reducing CPU frequency

1. Modify the CPU clock source by overwriting the SW bit in the RCC_CFGR register
2. If necessary, the CPU clock prescaler can be modified by rewriting the HPRE bit in the RCC_CFGR
3. By reading the corresponding clock source state (SWS bit) and/or AHB prescalation value in the RCC_CFGR register
(HPRE bit), check whether the new CPU clock source and/or the new CPU clock prescalation value is successfully set
4. Program the new number of waiting cycles to the LATENCY bit in FLASH_ACR
5. Check whether the new waiting period is set successfully by reading the FLASH_ACR register

4.6 Adaptive real-time memory accelerator

Proprietary Adaptive Real Time (ART) Memory Accelerator for ARM ® Cortex ® The-M4F processor is optimized. This accelerator well embodies the inherent performance advantages of ARM Cortex M4F, and overcomes the situation that high-speed processors often need to wait for FLASH reading during operation under normal conditions.

In order to give full play to the performance of the processor, the accelerator will implement instruction prefetch queue and branch cache, thus improving the program execution speed of 128-bit Flash. According to the CoreMark benchmark, the performance obtained with the ART accelerator is equivalent to Flash executing programs with 0 wait cycles at CPU frequencies up to 170 MHz.

- **Instruction prefetch**

Each Flash read operation can read 128 bits, which can be 4 lines of 32-bit instructions or 8 lines of 16-bit instructions, depending on the program programmed in Flash. Therefore, for sequentially executed code, it takes at least 4 CPU cycles to execute the previously read 128-bit instruction line. When the CPU requests the current instruction line, the prefetch operation of the I-Code bus can be used to read the next consecutively stored 128-bit instruction line in Flash.

- **Instruction cache memory**

In order to reduce the time loss caused by instruction jumping, 64 lines of 128-bit instructions can be saved to the instruction cache memory. Whenever an instruction is missing (i.e., the requested instruction does not exist on the currently used instruction line, prefetch instruction line, or instruction cache memory), the system copies the newly read line into the instruction cache memory. If the instruction requested by the CPU already exists in the instruction buffer, it can be fetched immediately without any delay. After the instruction cache memory is full, the LRU (Least Recently Used) policy can be used to determine the instruction line to be replaced in the instruction cache memory. This feature is particularly useful in case of code containing loops.

- **Data management**

During the CPU pipeline execution phase, the data buffer in Flash will be accessed through the D-Code bus. The CPU pipeline will only continue to execute after it obtains the requested data. In order to reduce the resulting time loss, access via the AHB data bus D-Code takes precedence over access via the AHB instruction bus I-Code. This feature works similarly to instruction cache memory, but the reserved data size is limited to 8 lines and 128 bits.

4.7 Erasure and program operations

Flash can be programmed through ICP (In-circuit programming) or IAP (In-application programming).

ICP: Used to update the contents of the entire Flash memory, you can use the JTAG/SWD protocol or Boot loader,

Load the user application into the MCU.

IAP: You can use the communication interface supported by the chip to download the data to be programmed into Flash. IAP allows the user to program flash memory again while the application is running.

If a reset occurs during a Flash program or erase operation, the contents of the Flash memory are not protected.

During a program or erase operation on one Bank, read access can be made to another Bank. Hard fault will be generated when reading and writing BANK in the program or erase operation.

HSI must be turned on for program and erase operations.

4.7.1 Unlocking the Flash memory

After reset, the Flash control register (Flash_CR) does not allow write operations to prevent unexpected operations on the Flash due to electrical interference or the like. Each erase and program operation on Flash must be unlocked by writing Flash_KEYR to enable access tothe Flash_CR register.

This is done in the following steps:

Step 1: Write KEY1 = 0x4567 0123 to the FLASH_KEYR register

Step 2: Write KEY2 = 0xCDEF 89AB to the FLASH_KEYR register

Any mistiminglocks the FLASH_CR register until the next reset. At the wrong KEY timing, a bus Hard fault interrupt will be generated. Such errors include a KEY1 mismatch for the first write cycle, or a KEY1 match but a KEY2 mismatch for the second write cycle.

The FLASH_CR register may be lockedagain by software writing the LOCK bit of the FLASH_CR register.

Note: When the BSY bit of the FLASH_SR register is set, the FLASH_CR register cannotbewritten.

4.7.2 Flash memory erase

Flash erase operation supports page, sector, block, bank erase, and does not work on information memory when Bank0 erase is executed.

4.7.2.1 Flash page erasure

The Page erase is used to erase the 512 Bytes Main flash, but it does not work for the Information area.

When a page is protected by WRP, it will not be erase. At this time, the WRPERR bit is set.

The specific steps of the page erase operation are as follows:

1) Check the BSY bit of the flash_SR register to confirm that there is no flash operation in progress

2) Write KEY1 and KEY2 to the FLASH_KEYR register in turn to release the protection of the FLASH_CR register

3)Setthe PER bit and EOPIE bit of the FLASH_CR register

4) Write arbitrary data to this page (must be 32-bit data)

5) Wait for the BSY bit to be cleared

6) Check that the EOP flag bit is set

7) Clear the EOP flag

4.7.2.2 Flash sector erase

Sector erase is used to erase the 4 KB Main flash, but it does not work for the Information area.

When a sector is protected by WRP, it will not be erase. At this time, the WRPERR bit is set.

The specific steps of the sector erase operation are as follows:

1) Check the BSY bit to confirm whether there is no ongoing Flash operation

2) Write KEY1 and KEY2 to that FLASH_KEYR register in turn to release the protection of the FLASH_CR register

3)Setthe SER bit and EOPIE bit of the FLASH_CR register

4) Write arbitrary data to this sector (must be 32-bit data)

5) Wait for the BSY bit to be cleared

6) Check that the EOP flag bit is set

7) Clear the EOP flag

4.7.2.3 Flash block erasure

The Block erase is used to erase the 32 KB Main flash, but does not work for the Information area.

When a Block is protected by WRP, it will not be erase. At this time, the WRPERR bit is set.

The specific steps of block erase operation are as follows:

1) Check the BSY bit to confirm whether there is no ongoing Flash operation

2) Write KEY1 and KEY2 to that FLASH_KEYR register in turn to release the protection of the FLASH_CR register

3)Setthe BER bit and EOPIE bit of the FLASH_CR register

4) Write arbitrary data to this block (32 bits data must be)

5) Wait for the BSY bit to be cleared

6) Check that the EOP flag bit is set

7) Clear the EOP flag

4.7.2.4 Flash Bank Erasure

Bank erase is used to erase the 256 KB Main flash, but it does not work for the Information area.

When WRP is enabled, the Bank erase function is invalid, no Bank erase operation is generated, and the WEPERR bit is set.

The specific steps of Bank erase operation are as follows:

1) Check the BSY bit to confirm whether there is no ongoing Flash operation

2) Write KEY1 and KEY2 to that FLASH_KEYR register in turn to release the protection of the FLASH_CR register

3) Set the MER1 bit or MER0 bit andthe EOPIE bit of the FLASH_CR register

4) Write any data to any Flash space corresponding to Bank Flash (32 bits data must be required)

5) Wait for the BSY bit to be cleared

6) Check that the EOP flag bit is set

7) Clear the EOP flag

4.7.3 Flash memory programming

Flash memory is performed in units of 32 bits (word) each time (a half word or byte operation will generate a Hard fault)

program operation for the entire page. When the PG bit of the Flash_CR register is set and the CPUwrites 32 bitsof data tothe Flash memory address space, the program operation starts.

If the flash address space to be programmed is an area set to protect by the flash_WRPR register, then

The program operation is ignored and the WRPRTERR bit of the FLASH_CR register is set. Knots of Program operations

Bundle, the EOP bit of the FLASH_CR register is set.

The specific operation steps of the Flash program are as follows:

1) Check the BSY bit of the Flash_SR register to determine whether there is currently no continuing Flash operation

2) If there is no ongoing Flash erase or program operation, the software reads out 128 of the Page word (if you need to keep the data already existing on this page, perform this step, otherwise skip this step)

3) Write KEY1 and KEY2 to the FLASH_KEYR register in turn to release the protection of the FLASH_CR register

4)Setthe PG bit and EOPIE bit of the FLASH_CR register

5) Perform the 1st to 127th word program operations to the target address (only 32-bit programs are accepted)

6)Setthe PGSTRT of the FLASH_CR register

7) Write the 128th word

8) Wait for the BSY bit of the FLASH_SR register to be cleared

9) Check the EOP flag bit of the FLASH_SR register (this bit is set when the program operation has been successful), and then

Software clears this bit

10) If there is no more program operation, the software clears the PG bit

When the above step 7) is successfully executed, the program operation is automatically started and the BSY bit is set by the hardware.

4.8 FLASH option bytes

4.8.1 Flash option byte description

Part of the Flash Information area in the chip is used as an option byte, which is used to store the configuration of the chip or the hardware required by the user for the application. As a configuration example, the watchdog may be selected in hardware or software mode.

For data security, option bytes are stored separately in positive code and negative code form.

Table 4-2 Options byte format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Option byte 1 inverse code								Option byte 0 inverse code							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Option byte 1								Option byte 0							

The contents of the option bytes can be read from the memory addresses described in the option byte structure in Table 4-3, or from the registers associated with the following option bytes:

- Flash user options register 1 (Flash_OPTR1)
- Flash user options register 2 (Flash_OPTR2)
- Flash WRP area address register (FLASH_WRPR)
- Flash PCROP0SR address register (FLASH_PCROP0SR)
- Flash PCROP0ER address register (FLASH_PCROP0ER)
- Flash PCROP1SR address register (FLASH_PCROP1SR)
- Flash PCROP1ER address register (FLASH_PCROP1ER)

Table 4-3 Options byte structure

Word Address	Description
0x1FFF 7400	User options byte 1 and inverse code
0x1FFF 7404	User options byte 2 and inverse code
0x1FFF 7408	Reserved
0x1FFF 740C	WRP address option byte and complemented code
0x1FFF 7410	PCROP0SR Address Option Byte and complemented code
0x1FFF 7414	PCROP0ER Address Option Byte and complemented code
0x1FFF 7418	PCROP1SR Address Option Byte and complemented code
0x1FFF 741C	PCROP1ER Address Option Byte and complemented code
…	Reserved
0x1FFF 74FF	Reserved

4.8.1.1 Option bytes for Flash user option 1

Flash memory address: 0x1FFF 7400

Production value: 0x0955 F6AA

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the Flash information memoryand written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ WWDG_SW	~ nRST_STDBY	~ nRST_STOP	~ IWDG_SW	~ BFB	~ IWDG_STDBY	~IWDG_STOP	Res	~RDP[7: 0]							
R	R	R	R	R	R	R		R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WWDG_SW	nRST_STDBY	nRST_STOP	IWDG_SW	BFB	IWDG_STDBY	IWDG_STOP	Res	RDP[7: 0]							
R	R	R	R	R	R	R		R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~ WWDG_SW	R	Complemented code of WWDG_SW
30	~ nRST_STDBY	R	Inverse code of NRST_STDBY
29	~ nRST_STOP	R	Inverse code of NRST_STOP
28	~IWDG_SW	R	Complemented code of IWDG_SW
27	~ BFB	R	Complemented code of BFB
26	~ IWDG_STDBY	R	Inverse code of IWDG_STDBY
25	~IWDG_STOP	R	Complemented code of IWDG_STOP
24	Reserved	-	-
23: 16	~ RDP	R	Complemented code of RDP
15	WWDG_SW	R	0: Hardware window watchdog 1: software window watchdog
14	NRST_STDBY	R	0: Reset generated when entering Standby mode 1: No Reset generation
13	NRST_STOP	R	0: Reset is generated when entering Stop mode 1: No Reset generation

12	IWDG_SW	R	0: Hardware independent watchdog 1: Software independent watchdog
11	BFB	R	0: Start from Bank0 (Bank0 mapped at 0x0800 0000; Bank1 mapped at 0x0804 0000) 1: Start from Bank1 (Bank1 mapped at 0x0800 0000; Bank0 mapped at 0x0804 0000)
10	IWDG_STDBY	R	Set the IWDG timer running status in Standby mode 0: freeze timer 1: Normal operation
9	IWDG_STOP	R	Set the running status of IWDG timer in Stop mode 0: freeze timer 1: Normal operation
8	Reserved	-	-
7: 0	RDP	R	0xAA: level 0, read protection is invalid Non-0xAA: level 1, read protection valid

4.8.1.2 Option bytes for Flash user option 2

Flash memory address: 0x1FFF 7404

Production value: 0x80FF 7F00

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BOOT_LOCK	~ nBOOT0	~ nSW-BOOT0	~ CCMSRAM_RST	~ SRAM_PE	~ NRST_MODE		~ nBOOT1	Res							
R	R	R	R	R	R	R	R	-							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_LOCK	nBOOT0	nSW-BOOT0	CCMSRAM_RST	SRAM_PE	NRST_MODE		nBOOT1	Res							
BOOT_LOCK	R	R	R	R	R	R	R	-							

Bit	Name	R/W	Function
31	~ BOOT_Lock	R	Inverse code of BOOT_LOCK
30	~ nBOOT0	R	Complemented code of nBOOT0
29	~ nSWBOOT0	R	Inverse code of nSWBOOT0
28	~ CCMSRAM_RST	R	Inverse coding of CCMSRAM_RST
27	SRAM_PE	R	Inverse coding of SRAM_PE
26: 25	~ NRST_MODE	R	Complemented code of NRST_MODE
24	~ nBOOT1	R	Complemented code of nBOOT1
23: 16	Reserved	-	-
15	BOOT_LOCK	R	Use to force boot from user Flash zone 0: Start based on pad/option bit configuration 1: Forced boot from master flash
14	nBOOT0	R	nBOOT0 option bit 0: nBOOT0 = 0 1: nBOOT0 = 1
13	nSWBOOT0	R	Software BOOT0 0: BOOT0 is taken from option bit nBOOT0 1: BOOT0 is taken from PB8/BOOT0 pin
12	CCMSRAM_RST	R	CCM SRAM erase on system reset: 0: CCM SRAM erase when system reset occurs 1: CCM SRAM does not erase when system reset occurs
11	SRAM_PE	R	SRAM1 and CCM SRAM parity grant bits 0: license 1: Disabled
10: 9	NRST_MODE	R	PF5 pad mode 00: Reset input/output 01: Reset input 10: GPIO 11: Reset input/output
8	nBOOT1	R	The boot configuration and the BOOT0 pin determine the selection of boot mode from flash main memory, SRAM1, or system memory. See Section 3.7: Startup Configuration.
7: 0	Reserved	-	-

4.8.1.3 Option bytes for Flash WRP addresses

Flash memory address: 0x1FFF 740C

Production value: 0xFF00 00FF

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								~WRP[7: 0]							
-								R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								WRP [7: 0]							
-								R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31: 24	Reserved	-	-
23: 16	~ WRP	R	Complemented code of WRP
15: 8	Reserved	-	-
7: 0	WRP	R	0: block [y] is protected 1: block [y] Unprotected y=0~7

4.8.1.4 Option bytes for Flash PCROP0SR address

Flash memory address: 0x1FFF 7410

Production value: 0xFE00 01FF

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							~ PCROPOSR [8: 0]								
-							R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							PCROPOSR [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31: 25	Reserved	-	-
24: 16	~ PCROP0SR	R	Complemented code of PCROP0SR
15: 9	Reserved	-	-
8: 0	PCROP0SR	R	BANK0 PCROP Zone start address (in Page)

4.8.1.5 Option bytes for Flash PCROP0ER address

Flash memory address: 0x1FFF 7414

Production value: 0xFFFF 0000

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							~ PCROPOER [8: 0]								
-							R	R	R	R	R	R	R	R	R
14	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							PCROPOER [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31: 25	Reserved	-	-
24: 16	~ PCROP0ER	R	Complemented code of PCROP0ER
15: 9	Reserved	-	-
8: 0	PCROP0ER	R	BANK0 PCROP Zone End Address (in Page)

4.8.1.6 Option bytes for Flash PCROP1SR address

Flash memory address: 0x1FFF 7418

Production value: 0xFE00 01FF

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							~ PCROP1SR [8: 0]								
-							R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							PCROP1SR [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31: 25	Reserved	-	-
24: 16	~ PCROP1SR	R	Complemented code of PCROP1SR
15: 9	Reserved	-	-
8: 0	PCROP1SR	R	BANK1 PCROP zone start address (in Page)

4.8.1.7 Option bytes for Flash PCROP1ER address

Flash memory address: 0x1FFF 741C

Production value: 0xFFFF 0000

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							~ PCROP1ER [8: 0]								
-							R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							PCROP1ER [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31: 25	Reserved	-	-
24: 16	~ PCROP1ER	R	Complemented code of PCROP1ER
15: 9	Reserved	-	-
8: 0	PCROP1ER	R	BANK1 PCROP Zone End Address (in Page)

4.8.2 Flash Option Byte Write

After reset, the bits associated with the option bytes in the FLASH_CR register are write-protected. The OPTLOCK bit in the FLASH_CR register must be cleared before relevant operations can be performed on the option byte.

The following steps are used to unlock the register:

1. The write protectionofthe FLASH_CR registeris releasedby the unlocking process
2. Write OPTKEY1 = 0x0819 2A3B to the FLASH_OPTKEYR register
3. Write OPTKEY2 = 0x4C5D 6E7F to the FLASH_OPTKEYR register

Any wrong sequence locks up the FLASH_CR register until the next reset. A bus Hard Fault interrupt is generated at the wrong KEY timing.

The User option (the option byte of information flash) can be protected from unwanted erase/program operations by writing the OPTLOCK bit of the FLASH_CR register by software.

If the software sets the Lock bit, the OPTLOCK bit is also automatically set.

4.8.2.1 Modify option bytes

The program operation of the option byte is different from the operation of Main flash. To modify the option bytes, the following steps are needed:

1. Clear the OPTLOCK bit using the previously described steps
2. Check the BSY bit to confirm that there are no ongoing Flash operations
3. Write the desired value (1 to 7 words) to the option byte register FLASH_OPTR1/FLASH_OPTR2/FLASH_WRPR/FLASH_PCROR0SR/FLASH_PCROR0ER/FLASH_PCROR1SR/FLASH_PCROR1ER
4. Set the OPTSTRT bit and the EOPIE bit
5. Write arbitrary data to 0x1FFF7400 address (32 bits data is required)
6. Wait for the BSY bit to be cleared.
7. Check that EOP flag bit is set
8. Clear EOP flags

For any changes to the option byte, the hardware will first erase the entire page corresponding to the option byte, and then write the values of the FLASH_OPTR1, FLASH_OPTR2, FLASH_WRPR, FLASH_PCROR0SR, FLASH_PCROR0ER, FLASH_PCROR1SR, and FLASH_PCROR1ER register-sintothe option byte. Furthermore, the hardware automatically calculates the corresponding complement and writes the calculated value to the corresponding area of the option byte.

4.8.2.2 Load location bytes

After the BSY bit is cleared, all new option bytes are written to the Flash information memory, but are not applied to the system. A read operation on the option byte register still returns the value in the last loaded option byte. Only when they (new values) are loaded do they work on the system.

The loading of option bytes occurs in the following two cases:

- When the OBL_LAUNCH bit in the FLASH_CR register is set
- After Power On Reset (POR)

The operation performed by loading option byte is to read the option byte in the information memory area, and then store the read data in the internal option register (FLASH_OPTR1, FLASH_OPTR2, FLASH_WRPR, FLASH_PCROROSR, FLASH_PCROR0ER, FLASH_PCROR1SR, FLASH_PCROR1ER).

Each option bit has a corresponding complement at its same double word address (the next half word). During the option byte load, the option bit and its inverse code are verified to ensure that the load is done correctly.

If the positive and negative codes match, the option bytes are copied into the option register.

Ifthe wordand its complement are not matching, the OPTVERR status bit of the FLASH_SR register is set. The mismatched values are written to the option register:

- For OPTR
  - The RDP bit is written as 0xff (i.e. level 1)
  - BFB is written as 0 (i.e. no bank exchange)
  - The rest of the mismatched values are written as 1
- For WRP option, the mismatched value is the default value "Unprotected"
- For the PCROP option, the mismatched value is the default value "All Flash Protection"

After the system reset, the contents of the option byte are copied to the following option register (software readable and writable):

- FLASH_OPTR1
- FLASH_OPTR2
- FLASH_WRPR
- FLASH_PCROP0SR
- FLASH_PCROP0ER
- FLASH_PCROP1SR
- FLASH_PCROP1ER

These registers are also used to modify option bytes.

4.9 Flash user OTP byte

A partial section of the Flash information area in the device is referred to as Flash USER OTP memory Bytes.

There are 2 Pages in total, and each Page is controlled separately.

Table 4-4 USER OTP Structure

Page	Word	Address	Contents
1	0	0x1FFF 7000	Bit [31:16]: Deposit user data Bit [15: 0]: USER OTP MEMORY_LOCK
	1	0x1FFF 7004	Store user data
	…	…	Store user data
	127 COMP block diagram.	0x1FFF 71FC	Store user data
2	0	0x1FFF 7200	Bit [31:16]: Deposit user data Bit [15: 0]: USER OTP MEMORY_LOCK
	1	0x1FFF 7204	Store user data
	…	…	Store user data
	127 COMP block diagram.	0x1FFF 73FC	Store user data

These two Pages are arranged in the information area, and the program and erase of this Page area are processed according to the method of Main flash. In addition, the mass erase of the Main Flash area is not valid for this area.

Set that the content of USER OTP MEMORY_LOCK will not be updated immediately until the power-on reset (POR/PDR), which will play a protection function.

This Page Write has the following protection.

Table 4-5 Flash USER OTP write protection

USER OTP MEMORY_LOCK	Write protection
0xAA55	Read: Yes Program and erase: No
Any value except (0xAA55)	Read, Program and erase: Yes

4.10 Flash protection

The protection of Main flash area includes the following mechanisms:

- Read protection (RDP), which prevents access from outside.
- Wrtie protection (WRP) control to prevent unwanted write operations (due to clutter of program memory pointer PC). The size of write protection is designed to be 32 KB.
- Option byte write protection, has a dedicated unlocking design.

4.10.1 Flash Read Protection (RDP)

The read protection function can be activated by setting the RDP option byte and performing a system reset (POR or OBL reset) to load a new RDP option byte. RDP protects flash main memory, CCM SRAM, and backup registers.

If the read protection is set while the debug through the SWD is still connected, a power-on reset is required instead of a system reset.

Flash memory is protected when the RDP option byte and the inverse code exist correctly in pairs in the option byte.

Table 4-6 Flash read protection status

RDP option byte value	RDP option byte inverse code value	Equal protection level
0xAA	0x55	Level 0
Any value except the combination of [0xAA, 0x55]		Level 1

System memory is readable regardless of the protection level, but it cannot perform program and erase operations.

4.10.1.1 Level 0: no protection

Read, program, and erase operations on the Main flash are possible, as are any operations on option bytes, CCM SRAM, and backup registers.

4.10.1.2 Level 1: Read protection

If the RDP and its inverse code in the option byte contain any combination other than [0xAA, 0x55], Level 1 read protection takes effect, and Level 1 is the default protection level.

- User mode: A program executed in User mode (boot from Main flash) that can do all operations on the Main flash, option bytes, CCM SRAM and backup registers.
- debug, boot from SRAM, boot from system memory modes: In debug mode, or when booting from SRAM or system memory, main flash, CCM SRAM And backup registers are not accessible. In these modes, a read or write access to the main flash results in a bus hard fault interrupt.

The read protection level can be changed:

- From Level 0 to Level 1, change the value of the RDP byte to anything except 0xAA
- Change the value of the RDP byte to 0xAA from Level 1 to Level 0

Level 1 changes to Level 0, which triggers the hardware mass erase Main flash.

Table 4-7 Relationship of access status to protection level and execution mode

Area	Protection level	User execution (BootFromFlash)			Debug/BootFromRam/ BootFromLoader		
		Read	Write	Erase	Read	Write	Erase
Main memory	0	Yes	Yes	Yes	Yes	Yes	Yes
	1	Yes	Yes	Yes	No	No	No
System memory	x	Yes	No	No	Yes	No	No
Option bytes	x	Yes	Yes	N/A	Yes	Yes	N/A
OTP	x	Yes	Yes	Yes	Yes	Yes	Yes
Backup registers	0	Yes	Yes	Yes	Yes	Yes	Yes
	1	Yes	Yes	Yes	No	No	No
CCM SRAM	0	Yes	Yes	Yes	Yes	Yes	Yes
	1	Yes	Yes	Yes	No	No	No

Notes:

(1) The Information zone is read-only, regardless of protection level and execution mode.

(2) A modification of the RDP from Level 1 to Level 0 triggers a mass erase of the hardware to the Main flash.

(3) There are two situations for executing programs from SRAM or system memory:

One is boot from, the other is boot from another memory, and the program jumps to SRAM or system memory for execution.

4.10.2 Flash Write Protection (WRP)

Flash can be set to write-protected for unwanted write operations. Define a write-protected (WRP) area with a control granularity of 64 KB per bit of the WRP register, that is, 1 block size. See the description of the WRP register for details.

When the WRP area is activated, no erase or program operation is allowed. Accordingly, even if only one area is set to write protection, the bank erase function does not function. In addition, if an attempt is made to perform an erase or program operation on a write-protected area, the write-protected error identifier (WRPERR) of the FLASH_SR register is set.

Note: Write protection only works for Main flash, not system memory.

4.10.3 Proprietary Code Readout Protection (PCROP)

In addition to flash memory, it can also prevent reading and writing from third parties. The protected area is execution-only: it can only be accessed by the CPU as instruction code, while all other access (DMA, debugging and CPU data reading, writing and erasing) is strictly prohibited. When the RDP protection is changed from level 1 to level 0, the inclusive PCROP area is also erased (see Changing the Read Protection Level).

Each PCROP area is defined by a start page and an end page address associated with a physical flash address. These addresses are defined in the PCROP address registers FLASH PCROP start address register (FLASH_PCROPOSR, FLASH_PCROP1SR), FLASH PCROP end address register (FLASH_PCROPOER, FLASH_PCROP1ER).

Any read access performed over the D-code bus to the PCROP protected area will trigger an RDERR flag error.

Any PCROP-protected address is also write-protected, and any write access to one of these addresses triggers WRPERR.

Any PCROP areas are also erase protected. Therefore, any erasure of pages in this area is not possible (including pages containing the start and end addresses of this area). Furthermore, software Bank Mass erasure cannot be performed if an area is protected by PCROP.

PCROP can only be disabled if the RDP is changed from level 1 to level 0. If you attempt to clear the PCROP or narrow the PCROP area by modifying the user options, you can start the option programming, but the PCROP area remains unchanged. Instead, the PCROP region can be increased.

Table 4-8 PCROP Protection

PCROPx register value (x = 0, 1)	PCROP Protected Area
PCROPxSR > PCROPxER	No PCROP area.
PCROPxSR < = PCROPxER	The region between PCROPxSR and PCROPxER is protected.

4.10.4 Forced Boot from Main flash

For increased security, the BOOT_LOCK option bit of the FLASH_OPTR2 register allows the system to be forced to boot from the main flash memory,ignoringother boot options. Reset BOOT_ LOCK only if:

- RDP set to Level 0

- RDP Level 1 modified to Level 0

4.11 Flash interrupt

Table 4-9 Flash interrupt request

Interrupt event	Event flag	Time flag/interrupt clearing	Control bit enable
End of operation	EOP	Write EOP=1	EOPIE
Write protection	WRPERR	Write WRPERR=1	ERRIE
Read error	RDERR	Write RDERR=1	RDERRIE
ECC correction	ECCC	Write ECCC=1	ECCCIE

The following event does not have a separate break identity, but produces a Hard fault:

- Flow error unlocking FLASH_CR register for Flash memory
- Write operation flow error to unlock Flash option byte
- Flash program, erase operation fails to align 32-bit data
- Read and write access to Bank in program, erase
- In one Bank program, erase, another Bank program, erase

4.12 TIMx registers

4.12.1 Flashaccesscontrol register (Flash_ACR)

Address offset: 0x00

Reset value: 0x0000 0700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LATENCY [3: 0]			
-					-			-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10: 8	Reserved		3'h7	Write only initial values
7: 4	Reserved	-	-	-
3: 0	LATENCY [3: 0]	RW	4'h0	Waiting state corresponding to Flash read operation: HCLK < = 25 MHz 0: The flash read operation has no waiting state, that is, each flash read requires 1 system clock cycle 25 MHz < HCLK < = 50 MHz 1: The flash read operation has a waiting state, that is, each flash read requires 2 system clock cycles 50 MHz < HCLK < = 75 MHz 3: The flash read operation has 3 waiting states, that is, each flash read requires 4 system clock cycles 75 MHz < HCLK < = 100 MHz 4: The flash read operation has 4 waiting states, that is, each flash read requires 5 system clock cycles 100 MHz < HCLK < = 125 MHz 5: The flash read operation has 5 waiting states, that is, each flash read requires 6 system clock cycles 125 MHz < HCLK < = 150 MHz 6: flash read has 6 waiting states, that is, each flash read requires 7 system clock cycles 150 MHz < HCLK < = 170 MHz 7: The flash read operation has 7 waiting states, that is, each flash read requires 8 system clock cycles Remaining values are not recommended for configuration Note: When the latency bit is configured, no value can be written to other bits of this register

4.12.2 Flashkey register (Flash_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

All register bits are write-only and the read returns 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY [31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY [15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 0	KEY [31: 0]	W	32'h0	The following values must be written continuously to unlock the flash_CR register and enable the program/erase operation of flash KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.12.3 Flashoption key register (Flash_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

All register bits are write-only and the read returns 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPTKEY [31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTKEY [15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 0	OPTKEY [31: 0]	W	32'h0	The following values must be written consecutively to unlock the flash option register and enable the program/erase operation of the option byte KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F

4.12.4 Flashstatus register (Flash_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY1	BSY0
-														R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTV ERR	RD ERR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP ERR	Res.	Res.	Res.	EOP
RC_W1	RC_W1	-									RC_W1	-			RC_W1

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	BSY1	R	0	Busy bit This bit indicates that the operation of flash bank1 is in progress. This bit is set by the hardware at the beginning of the flash operation, and is cleared by the hardware when the operation is completed or an error occurs.
16	BSY0	R	0	Busy bit This bit indicates that the operation of flash bank0 is in progress. This bit is set by the hardware at the beginning of the flash operation, and is cleared by the hardware when the operation is completed or an error occurs.
15	OPTVERR	RC_W1	0	Option byte loading error When option and its inverse do not match, the hardware sets this bit. Load mismatched option bytes, forced to safe values, reference FLASH option bytes. This bit is cleared by writing 1.

14	RDERR	RC_W1	0	PCROP Read Error When the address to be read through the D-code bus belongs to the read protection area of the flash memory, it is set by hardware (PCROP protection). This bit is cleared by writing 1.
13: 5	Reserved	-	-	-
4	WRPERR	RC_W1	0	Write protection error When the address to be programmed/erase is in the write-protected flash area (WRP), the hardware sets this bit. This bit is cleared by writing 1.
3:1	Reserved	-	-	-
0	EOP	RC_W1	0	When the program/erase operation of flash is successfully completed, the hardware is set. This bit is set only if the EOPIE bit of the FLASH_CR register is enabled. This bit is cleared by writing 1.

4.12.5 FlashControl Register (Flash_CR)

Address offset: 0x14

Reset value: 0xC000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPT LOCK	Res.	Res.	OBL LAUNCH	RD ERR IE	ERR IE	EOP IE	Res.	Res.	Res.	Res.	PG STRT	Res.	OPT STRT	Res.
RS	RS	-		RC_W1	RW	RW	RW	-				RW	-	RW	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	BER	SER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MER1	MER0	PER	PG
-			RW	RW	-							RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	FLASH_CR Lock bit. The software can only set this bit. When set, the FLASH_CR register is locked. When the unlock flow is successfully given, the bit is cleared by hardware. [This bit is set by software after the program/erase operation is completed] When an unsuccessful unlock flow is given, the bit remains set until the next system reset.
30	OPTLOCK	RS	1	Option byte Lock bit. The software can only set this bit. After reset, the bits associated with the option bytes in the FLASH_CR register are locked. When the unlock flow is successfully given, the bit is cleared by hardware. [This bit is set by software after the program/erase operation is completed] When an unsuccessful unlock timing is given, the bit remains set until the next system reset.
29: 28	Reserved	-	-	-
27	OBL_LAUNCH	RC_W1	0	Force option byte loading When set, this bit forces the system to reload the option byte. This bit is cleared by hardware only after the option byte load is completed. If the OPTLOCK bit is set, the bit cannot be written. 0: Option byte loading complete 1: Generate an option byte load request, generate a system reset, and reload the option byte.
26	RDERRIE	RW	0	PCROP Read Error Interrupt Enable When the RDERR bit in FLASH_SR is set, if the bit is enabled, an interrupt request is generated. 0: No interrupt occurrence 1: An interrupt occurs
25	ERRIE	RW	0	Error interrupt enable When the WRPERR bit of the FLASH_SR register is set, if the bit is enabled, an interrupt request is generated. 0: No interrupt occurrence 1: An interrupt occurs
24	EOPIE	RW	0	End of operation interrupt enable When the EOP bit of the FLASH_SR register is set, if the bit is enabled, an interrupt request is generated.

				0: EOP interrupt disabled 1: EOP interrupt enabled
23: 18	Reserved	-	-	-
19	PGSTRT	RW	0	The start bit of the program operation for the Flash main memory. This bit starts the program operation of the Flash main memory when set. This bit is cleared by hardware when the BSY bit is cleared in FLASH_SR.
18	Reserved	-	-	-
17	OPTSTRT	RW	0	Flash option byte modification start bit This bit tiggers the modification of the option byte. This bit is set by software and is cleared when the BSY bit is cleared in FLASH_SR. Note: When the flash option byte is modified, the hardware automatically performs an erase operation on the entire 128Bytes page, and then performs a program operation, which also includes automatic write of inverse code.
16: 13	Reserved	-	-	-
12	BER	RW	0	Block erase operation 0: block erase operation for flash not selected 1: Select the block erases operation of flash
11	SER	RW	0	Sector erase action 0: sector erase operation for flash is not selected 1: Select the flash sector erases operation
10: 4	Reserved	-	-	-
3	MER1	RW	0	Bank 1 erase operation 0: bank1 erase operation for flash not selected 1: Select the bank1 erases operation of flash
2	MER0	RW	0	Bank 0 erase operation 0: bank0 erase operation for flash not selected 1: Select the bank0 erases operation of flash
1	PER	RW	0	Page erase operation 0: page erase operation for flash is not selected 1: Page erase operation of flash selected
0	PG	RW	0	Program operation 0: program operation of flash is not selected 1: Select the program operation of flash

4.12.6 Flash ECC Register (FLASH_ECCR)

Address offset: 0x18

Reset value: 0x0000 0000.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	ECCC	Res	Res	Res	Res	Res	ECCCIE	Res	SYSF_ECC	BK_ECC	Res	Res	Res	ADDR_ECC [17: 16]	
RC_W1	RC_W1	-					RW	-	R	R	-			R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR_ECC [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31	ECCD	RC_W1	0	ECC detection flags Set by hardware when an ECC two-bit error is detected, and when this bit is set, an NMI interrupt is generated. Clear by writing 1.
30	ECCC	RC_W1	0	ECC Correction Flag Set by hardware when a bit ECC error is detected and corrected, and generate an interrupt when ECCCIE is set. Clear by writing 1.
29: 25	Reserved	-	-	-
24	ECCCIE	RW	0	ECC Correction Interrupt Enable 0: Disable ECCC interrupt 1: Enable ECCC interrupt.
23	Reserved	-	-	-
22	SYSF_ECC	R	0	ECC failure memory 0: Main memory

				information memory
21	BK_ECC	R	0	ECC Main memory failure bank 0: bank 0 1 bank 1
20: 18	Reserved	-	-	-
17: 0	ADDR_ECC [17: 0]	R	18'h0	ECC fault address Record fault address

4.12.7 Flash Option1 Register (Flash_OPTR1)

Address offset: 0x20

Reset value: 0x0000 XXXX.

After the power-on reset (POR/OBL_LAUNCH) is released,the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WWDG _ SW.	nRST STDBY	nRST STOP	IWDG _SW	BFB	IWDG _ STDBY	IWDG _ STOP	Res.	RDP[7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	WWDG_SW	RW		0: Hardware window watchdog 1: software window watchdog
14	NRST_STDBY	RW		0: Reset generated when entering Standby mode 1: No Reset generation
13	NRST_STOP	RW		0: Reset is generated when entering Stop mode 1: No Reset generation
12	IWDG_SW	RW		0: Hardware watchdog 1: Software watchdog
11	BFB	RW		0: Start from Bank0 (Bank0 mapped at 0x0800 0000; Bank1 mapped at 0x0804 0000) 1: Start from Bank1 (Bank1 mapped at 0x0800 0000; Bank0 mapped at 0x0804 0000)
10	IWDG_STDBY	RW		Set the IWDG timer running status in Standby mode 0: freeze timer 1: Normal operation
9	IWDG_STOP	RW		Set the running status of IWDG timer in Stop mode 0: freeze timer 1: Normal operation
8	Res			
7: 0	RDP	RW		0xAA: level 0, read protection is invalid Non-0xAA: level 1, read protection valid

4.12.8 Flash Option2 Register (Flash_OPTR2)

Address offset: 0x24

Reset value: 0x0000 XXXX.

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOT_LOCK	nBOOT0	nSWBOOT0	CCMSRAM_RST	SRAM_PE	NRST_MODE		nBOOT1	Res							
RW	RW	RW	RW	RW	RW	RW	RW	-							

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	BOOT_LOCK	RW		Use to force boot from user Flash zone 0: Start based on pad/option bit configuration 1: Forced boot from master flash
14	nBOOT0	RW		nBOOT0 option bit 0: nBOOT0 = 0 1: nBOOT0 = 1
13	nSWBOOT0	RW		Software BOOT0 0: BOOT0 is taken from option bit nBOOT0 1: BOOT0 is taken from PB8/BOOT0 pin
12	CCMSRAM_RST	RW		CCM SRAM erase on system reset: 0: CCM SRAM erase when system reset occurs 1: CCM SRAM does not erase when system reset occurs
11	SRAM_PE	RW		SRAM1 and CCM SRAM parity grant bits 0: license 1: Disabled
10: 9	NRST_MODE	RW		PF5 pad mode 00: Reset input/output 01: Reset input 10: GPIO 11: Reset input/output
8	nBOOT1	RW		The boot configuration and the BOOT0 pin determine the selection of boot mode from flash main memory, SRAM1, or system memory. See Section 3.7: Startup Configuration.
7: 0	Reserved	-	-	-

4.12.9 FlashWRP address register (FLASH_WRPR)

Address offset: 0x2C

Reset value: 0x0000 00XX.

After the power-on reset (POR/OBL_LAUNCH) is released, the corresponding value is read from the option byte area of the Flash information memoryand written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	WRP [7: 0]	RW		Write protection, each bit corresponds to a block 0: block N, write protection valid 1: block N, write protection is invalid N = 0-7

4.12.10 FlashPCROP0 start address register (FLASH_PCROP0SR)

Address offset: 0x30

Reset value: 0x0000 0XXX.

After the power-on reset (POR/OBL_LAUNCH) is released,the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROPOSR [8: 0]								
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	PCROP0SR	RW		BANK0 PCROP Zone start address (in Page)

Notes:

1．Without using the PCROP function, this register is set to 0x1FF.

2．Using the PCROP function, this register is set to the start address.

4.12.11 FlashPCROP0 end address register (FLASH_PCROP0ER)

Address offset: 0x34

Reset value: 0x0000 0XXX.

After the power-on reset (POR/OBL_LAUNCH) is released,the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROPOER [8: 0]								
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	PCROP0ER	RW		BANK0 PCROP Zone End Address (in Page)

Notes:

1. Without using the PCROP function, this register is set to 0x1FF.
2. Using the PCROP function, this register is set to the start address

4.12.12 FlashPCROP1 start address register (FLASH_PCROP1SR)

Address offset: 0x38

Reset value: 0x0000 0XXX.

After the power-on reset (POR/OBL_LAUNCH) is released,the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1SR [8: 0]								
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	PCROP1SR	RW		BANK1 PCROP zone start address (in Page)

Notes:

1. This register is set to 0x1FF without using the PCROP function
2. Using the PCROP function:

   For 128K products, this register is set to the start address +0x80;

   For 256K products, this register is set to the start address +0x100;

For 384K products, this register is set to the start address +0x180, and can only protect the first 64KB of BANK1.

For 512K products, this register is set to the start address.

4.12.13 Flash PCROP1 end address register (FLASH_PCROP1ER)

Address offset: 0x3C

Reset value: 0x0000 0XXX.

After the power-on reset (POR/OBL_LAUNCH) is released,the corresponding value is read from the option byte area of the flash information memory and written to the corresponding option bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1ER [8: 0]								
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	PCROP1ER	RW		BANK1 PCROP Zone End Address (in Page)

Notes:

1. This register is set to 0x1FF without using the PCROP function
2. Using the PCROP function:

   For 128K products, this register is set to the start address +0x80;

   For 256K products, this register is set to the start address +0x100;

   For 384K products, this register is set to the start address +0x180, and can only protect the first 64KB of BANK1.

   For 512K products, this register is set to the start address.

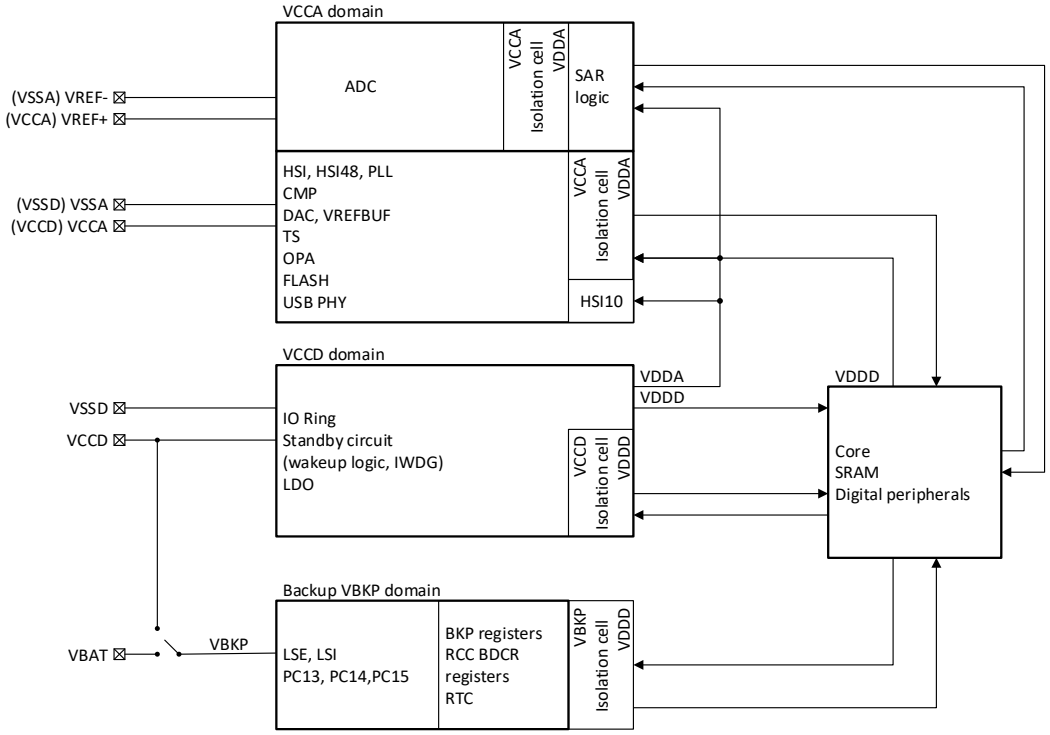
5. Power Control (PWR)

5.1 Introduction

The PWR module performs the following functions:

- Chip power-on and power-off process control
- Analog PMU control signal generation
- Generation of memory low power control signal
- Low Power Mode Entry and Wake Timing Control
- PVD functionality

5.2 Power Structure



- The chip operating voltage is 2.3 to 3.6 V
- All low-power entry and exit logic is implemented in the VCCD power domain.
- The VBKP domain logic has an independent power-on reset BPOR
- The power-on reset of digital logic in the VDDD domain is generated by the VCCD module
- Using the VCC domain clock generated by HSI 16 as the clock for the standby mode entry and wake-up logic, HSI_ON needs to be a VCCD domain signal
- In standby mode, IWDG can be configured to enable operation, and its counting clock is LSI, so the LSI clock needs to be output in standby mode, that is, the LSI clock will also output the VCCD domain clock.

5.2.1 System Main Power Supply

5.2.1.1 Independent A/D converter power supply and reference voltage

To improve conversion accuracy, the ADC is equipped with an independent power supply that can individually filter and mask the noise on the PCB.

- The ADC voltage source is input from a separate VCCA pin.
- The VSSA pin provides a separate power ground connection.

To improve the accuracy of the low-voltage input, users can connect a separate ADC external reference voltage input on the VREF+. The voltage on VREF + ranges from 2.3 V to VCCA.

5.2.1.2 Backup battery domain

To preserve the contents of the RTC backup register and power the RTC after the VCCD is powered down, you can connect the VBAT pin to a battery or other backup power source.

In order for the RTC to operate even after the main digital power supply (VCCD) is turned off, the VBAT pin needs to power each of the following modules:

- RCC BDCR Register
- RTC module (except RTC_CR register)
- LSE Oscillator
- BKP registers
- PC13 to PC15 I/O

The switch of the VBAT power supply is controlled by a power-down reset circuit (BPOR) built into the reset module.

Note:

- After tRSTTEMPO (VCC start time) or PDR is detected, the power switch between VBAT and VCCD is still connected to VBAT.
- During the start-up phase, if the established VCCD is less than tRSTTEMPO (see tRSTTEMPO value in the datasheet) and VCCD > VBAT + 0.6 V, current can be injected into the VBAT by connecting an internal diode between the VCCD and the power switch (VBAT).

If the power supply/battery connected to the VBAT pin cannot support this current injection, it is highly recommended to connect a diode between this power supply and the VBAT pin.

If no external battery is used in the application, it is recommended to connect the VBAT pin to the VCCD, which needs to be connected in parallel with a 100 nF decoupling ceramic capacitor.

When powering the backup domain via the VCCD (analog switch connected to the VCCD):

- PC14 and PC15 can be used as GPIO or LSE pins
- The PC13 can be used as a GPIO or configured for other functions (TAMPER pin, RTC calibration clock, RTC alarm, or second output)

Note: Due to the limited sink capability of this switch (3 mA), the following limitations exist when using GPIO PC13 through PC15 in output mode: the rate must not exceed 2 MHz, the maximum load is 30 pF, and these I/Os cannot be used as current sources (e.g., for driving LEDs).

When powering the backup domain via VBAT (the internal power switch is connected to VBAT because there is no VCCD), the following pins can be implemented:

- PC14 and PC15 can only be used as LSE pins

- PC13 can be used as an RTC additional function pin (TAMPER pin, RTC alarm clock or second output)

Backup domain access

After reset, the backup domains (RTC registers and RTC backup registers) will be protected against unintended write access. To enable access to the backup domain (RTC and RTC backup registers), you need to configure as follows:

- Enable the power interface clock by PWREN position 1 in the RCC_APB1ENR1 register
- Enable access to the backup domain by DBP location 1 in the PWR_CR1 register
- Configure the RTCSEL register in the RCC backup domain control register (RCC_BDCR) to select the RTC clock source
- Configure RTCEN in RCC Backup Domain Control Register (RCC_BDCR) to enable RTC clock

5.2.1.3 VDDD voltage regulator

Embedded linear VR powers all digital circuitry other than the backup domain and standby (wake-up) circuitry.

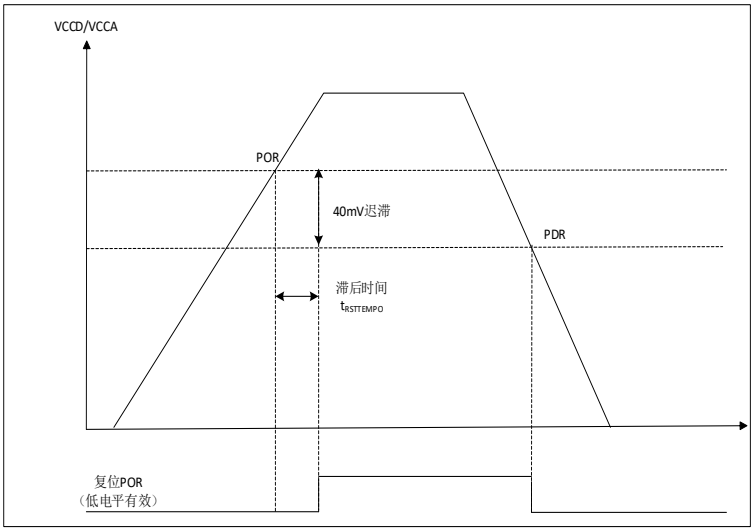
When activated by software, the voltage regulator is always enabled after reset. Depending on the application mode, three different modes can be used to work:

- In Run mode, the voltage regulator provides full power to the core, memory and digital peripherals. In this mode, the regulator output voltage can be adjusted by software to different voltage values (configurable via the VOS [1: 0] bit of the PWR_CR register).
- In stop mode, the main regulator or low power regulator supplies a low power voltage to the VDDD domain, saving the contents of the registers and internal SRAM. The regulator may also be placed in main regulator mode (MR) or low power mode (LPR).
- In standby mode, the voltage regulator powers down. Except for the standby (wake-up) circuit and backup domain, the contents of the registers, SRAM1 and CCM SRAM will be lost, and the contents of SRAM2 can be optionally preserved (controlled by the RRS bit of PWR_CR3).

5.2.2 Power Detection

5.2.2.1 POR/PDR

The chip contains Power-on reset (POR) and Power-off reset (PDR) modules to provide power-related resets. Where POR/PDR works by default in all power modes. POR/PDR output reset low active.



5.2.2.2 PVD

The PVD module detects whether the VCCis lower than the threshold set by PWR_CR2. PLS.

The PVD function is enabled by configuring the PWR_CR2. PVDE register.

The PVDO flag in the Power Control/Status Register (PWR_CSR)is used to indicate whether VCCis above or below the voltage threshold of PVD. This event is internally connected to Line16 of EXTI and generates an interrupt if the interrupt is enabled in the external interrupt register. A PVD interrupt occurs when the VCC falls below the PVD threshold or when the VCC falls above the PVD threshold, according to the rising/falling edge trigger setting of EXTI Line16. In practical applications, this feature can be used to perform emergency shutdown tasks.

5.3 System Low Power Mode

By default, the microcontroller is in Run mode after a system or a power Reset. In the running mode, the CPU provides clock through the HCLK and executes program code. The system provides multiple low-power modes to save power when the CPU is not required to run, such as when waiting for external events. It is up to the user to select the specific low-power mode according to the application, and seek the best balance between low power consumption, short startup time and available wake-up sources.

The device has five low-power modes:

- Sleep mode: The kernel stops running, and the kernel peripherals (NVIC, Sys Tick) can be configured to run;
- Low-power run mode (Low-power run): the system clock is configured to HSI16 and HCLK is divided below 8 (i.e. HCLK is not greater than 2MHz), and VDDD is powered by LPR;
- Low-power Sleep mode (Low-power Sleep): enter the Sleep mode from Low-power run;
- Stop mode: All high-speed clocks in the VDDD domain are turned off, and SRAM and register contents are held. The LSI and LSE can be selected whether to turn on or not according to the wake-up source. The LDO working mode when entering STOP can be configured, and both MR and LPR configurations are supported. The power consumption of these two modes decreases in turn, and the wake-up time becomes longer in turn.
- Standby mode:The VDDD domain is powered off and only the VCCD and VBKP domains work. You can select whether SRAM2 is maintained by configuring (PWR_CR3. RRS); The LSI and LSE can be selected whether to turn on or not according to the wake-up source.

The device has VBAT power supply, so when the VCC is powered down, the device only works in the VBKP domain.

In addition, the power consumption of the operating mode can be reduced by one of the following methods:

- Slowing down system clocks
- When APBx and AHBx peripherals are not in use, the corresponding peripheral clock is turned off.

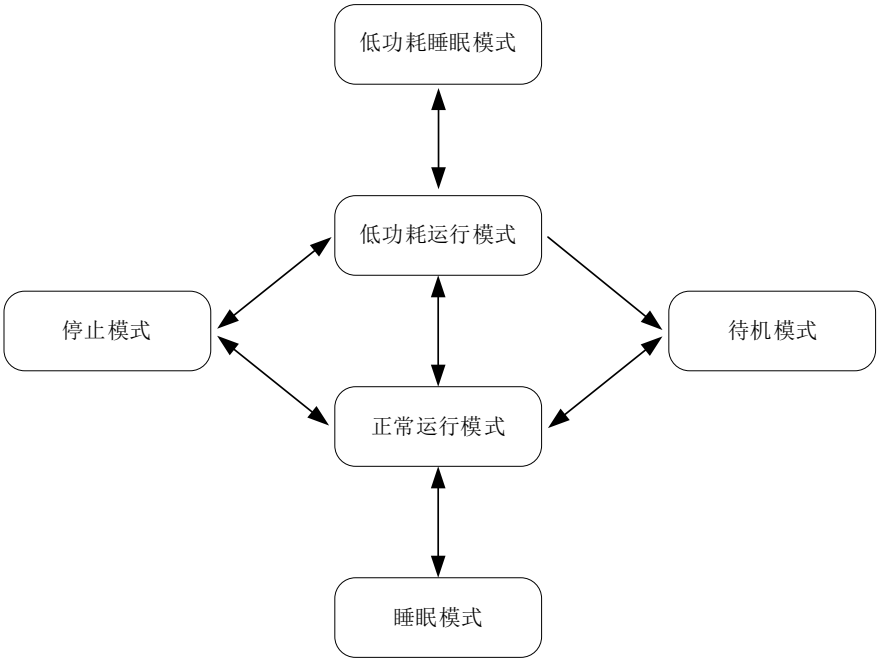


Figure 5-1 Low power mode state transition diagram

Table 5-1 Low power modes

Mode	Clock status	Entry conditions	Exit Conditions	Wake-up delay	System clock after wake up	MR status	LPR status
Sleep	CPU clock OFF	WFI or return from interrupt SLEEPDEEP = 0	Arbitrary interrupt	6 system clocks (1)	Pre-entry clock	ON	OFF
		WFE	Wake up event				
Low-power run	System clock: HSI divided 4	LPR = 1	LPR = 0	2us(2)		OFF	ON
Low-power Sleep	CPU clock OFF	WFI or return from interrupt SLEEPDEEP = 0LPR = 1	Arbitrary interrupt	6 system clocks (1)			
		WFELPR = 1	Wake up event				
Stop0	HSI/LSI/LSE can optionally turn on other clocks off	LPMS = 00SLEEPDEEP = 1 WFI or return from interrupt or WFE	Any EXTI specific peripheral wake -up event	5.5 us (3)		ON	OFF
Stop1	Clocks OFF except LSE, LSI	LPMS = 01 SLEEPDEEP = 1 WFI or return from interrupt or WFE		5.5/7.5 us (4)		OFF	ON
SRAM hold	Clocks OFF except LSE, LSI	LPMS = 11 SLEEPDEEP = 1 RRS = 1 WFI or return from interrupt or WFE	WKUP IORTCTAMP External Reset PIN Reset IWDG Reset	37 us (5)	HSI	OFF	OFF
SRAM does not hold	Clocks OFF except LSE, LSI	LPMS = 11SLEEPDEEP = 1 RRS = 0WFI or return from interrupt or WFE				OFF	OFF

1. The system clock is the sys_clk (reference RCC module) cycle configured before entering SLEEP.
2. The MR stabilization time is about 2 μs;
3. Including HSI16 stabilization time and memory (Flash) low power exit stabilization time;
4. There are two configurations, HSI16 and MR are turned on simultaneously (5.5 μs) and HSI16 waiting for MR to stabilize before turning on (7.5 μs).

5. Including MR start-up to stabilization time of 10μs, HSI16 stabilization time of 2μs, Flash stabilization time of 5.1 μs and power-on internal calibration data loading time of 20μs.

   The above are all simulation values, and the actual value is estimated to be less than the above time (please refer to the data sheet for specific parameters)

The following table shows whether each functional module exists in different modes:

Peripheral	Run	Sleep	LP Run	LP Sleep	Stop		Standby		VBAT
					-	Wake-up	-	Wake-up	
CPU Core	Y	-	Y	-	-	-	-	-	-
Flash memory	Y	Y (2)	Y	Y (2)	-	-	-	-	-
SRAM1	Y	Y (3)	Y	Y (3)	Y	-	-	-	-
SRAM2	Y	Y (3)	Y	Y (3)	Y	-	O(4)	-	-
CCM SRAM	Y	Y (3)	Y	Y (3)	Y	-	-	-	-
ESMC	O	O	O	O	-	-	-	-	-
BKP	Y	Y	Y	Y	Y	-	Y	-	Y
PVD	O	O	O	O	O	O	-	-	-
DMA	O	O	O	O	-	-	-	-	-
HSI16	O	O	O	O	O(5)	-	-	-	-
HSI48	O	O	-	-	-	-	-	-	-
HSE	O	O	O	O	-	-	-	-	-
LSI	O	O	O	O	O	-	O	-	O
LSE	O	O	O	O	O	-	O	-	O
PLL	O	O	-	-	-	-	-	-	-
HSE Clock Security System (CSS)	O	O	O	O	-	-	-	-	-
RTC/Auto Wakeup	O	O	O	O	O	O	O	O	O
Number of RTC TAMP pin	1	1	1	1	1	O	1	O	1
USBx (x = 1, 2)	O	O	-	-	-	O	-	-	-
Ethernet MAC	O	O	-	-	-	O	-	-	-
USARTx (x = 1, 2, 3)	O	O	O	O	-	-	-	-	-
UARTx (x = 1, 2, 3)	O	O	O	O	-	-	-	-	-
LPUART1	O	O	O	O	O(6)	O(6)	-	-	-
I2Cx (x = 1, 2, 3, 4)	O	O	O	O	O(7)	O(7)	-	-	-
SPIx (x = 1, 2, 3)	O	O	O	O	-	-	-	-	-
CANx (x = 1, 2)	O	O	O	O	-	-	-	-	-
ADCx (x = 1, 2, 3)	O	O	O	O	-	-	-	-	-
DACx (x = 1, 2)	O	O	O	O	-	-	-	-	-
VREFBUF	O	O	O	O	-	-	-	-	-
OPAx (x = 1, 2, 3)	O	O	O	O	-	-	-	-	-
COMPx (x = 1, 2, 3, 4)	O	O	O	O	O	O	-	-	-
Temperature sensor	O	O	O	O	-	-	-	-	-
Timers	O	O	O	O	-	-	-	-	-
LPTIM	O	O	O	O	O	O	-	-	-
IWDG	O	O	O	O	O	O	O	O	-

WWDG	O	O	O	O	-	-	-	-	-
SysTick timer	O	O	O	O	-	-	-	-	-
RNG	O	O	O	O	-	-	-	-	-
AES	O	O	O	O	-	-	-	-	-
CORDIC	O	O	O	O	-	-	-	-	-
SDIO	O	O	O	O	-	-	-	-	-
CRC	O	O	O	O	-	-	-	-	-
GPIOs	O	O	O	O	O	O	O(8)	O(9)	-

Notes:

1. Y = YES (ENABLE); O = Optional (Off by default, can be enabled by software); -= not available
2. FlashInSleepand LP _Sleepmodes, the software can select whether the clock isonor not
3. SRAM InSleepand LP _Sleepmodes, the software can select whether the clock is on or not
4. The SRAM2 may be maintained
5. In Stop mode, the HSI 16 can be switched by software
6. LPUART can be received in Stop mode, resulting in START and address matching interrupt wake-up
7. I2C provides address matching and timeout interrupt wake-up in Stop mode
8. In Standby mode, IO can be configured as pull-up, pull-down or float
9. IO with wake-up function in Standby mode: PA0 PA2 PC5 PC13 PE6

5.3.1 Normal Operating Mode

In normal operation mode, power consumption can be reduced by reducing the system clock (SYSCLK, HCLK, PCLK); Before entering SLEEP mode, the peripheral clock frequency can also be reduced by frequency division;

In the normal operation mode, different peripherals and memories can separately turn off the clock through gating to reduce power consumption; Before entering SLEEP, peripheral clocks that are not used for wakeup can be turned off;

5.3.2 Low power operating mode

To further reduce the power consumption in the operating mode, the MR can be turned off, and the VDDD domain is powered by the LPR. At the same time, reduce the operating frequency of the system, which cannot exceed 2 MHz;

IO status in low power operation mode

In the low-power operation mode, all IO states remain consistent with the operating state.

Entry of low power operating mode

1. Optional: The program jumps to SRAM to run, and configures Flash to enter low power mode;
2. Reduce the system clock to less than 2 MHz;
3. Set VDDD to use LPR power and MR to turn off.

Exit of low power operating mode

1. Set VDDD to use MR power;
2. Wait for the MR power-on to complete;
3. Restore the normal system clock.

Low-power run	Description
Mode entry	Reduce system clock by less than 2 MHz LPR = 1
Mode Exit	LPR = 0 Wait for MR power-on to complete Return to normal clock
Wake-up delay	MR on-to-steady time

5.3.3 Low power mode entry and exit

Enter:

Low power mode can be entered by executing a WFI (Wait for Interrupt) or WFE (Wait for Event) instruction, or by SLEEPONEXIT position 1 in the system control register upon return from ISR.

Only when there is no interrupt or event pending, you can enter low power mode through WFI or WFE, that is, the CPU needs to clear the suspended interrupt or event before entering low power.

In the process of entering the low power mode, a wake-up signal appears, and the system will judge how to deal with it according to the current state:

1. Exit into the flow to handle wakeup interrupts or events before the system clock shuts down;
2. After the system clock is turned off, hold the wake-up signal first, and then enter the low-power mode normally. After entering the low-power mode, wake up according to the normal wake-up process.

Exit:

Exit these two low-power modes depending on how you enter sleep and stop modes:

- Any peripheral interrupts acknowledged by NVIC can wake up the device if you enter low power mode using a WFI command or return from the ISR.
- If the WFE instruction is used to enter the low power mode, the MCU will immediately exit the low power mode when an event occurs. The wakeup event can be generated either by:
  - NVIC IRQ Interrupt:

  When SEVONPEND = 0, you can wake up from the WFE by enabling the interrupt enable control bit of the NVIC and the corresponding peripheral. When the system wakes up, the interrupt suspend bits in NVIC and peripherals must be cleared;

  When SEVONPEND = 1, you can wake up from WFE only by enabling the interrupt enable control bit of the peripheral, and the interrupt enable in NVIC can select whether it is enabled or not. When the system wakes up, the interrupt suspend bit in the peripheral must be cleared. If the interrupt in NVIC is enabled, it must also be cleared at the same time;

  All NVIC interrupts can wake up the CPU, even those not enabled in NVIC.

  - Event:

  Configure an external or internal EXTI line to event mode. When the CPU recovers from the WFE, it is not necessary to clear the interrupt suspend bit or the NVIC IRQ channel suspend bit of the EXTI peripheral because the suspend bit of the corresponding event line is not set to 1. Interrupt flags in peripherals may need to be cleared. This mode takes the shortest time to wake up because no time is spent on the entry or exit of the interrupt.

  When an external reset (NRST pin), an IWDG reset occurs, a rising edge appears on the enabled WKUPx pin, or an RTC wake-up event is triggered, the MCU exits standby low power mode.

After waking up from standby mode, the program will be re-executed in the manner after reset (starting boot pin sampling, reset vector acquisition, etc.).

5.3.4 Sleep Mode

Before entering sleep mode, all interrupt pending bits must be cleared.

IO status in sleep mode

In sleep mode, all IO states remain consistent with the running state.

Sleep Mode	Description
Mode entry	WFI or WFE and: --SLEEPDEEP = 0 and --There is no interrupt hanging.
	Returns from the lowest priority ISR, and: --SLEEPDEEP = 0 and --SLEEPONEXIT = 1 --There is no interrupt hanging.
Mode Exit	Enter using WFI or returning from lowest priority ISR: Interrupt (enabled) Entered with WFE and SEVONPEN = 0: Wake event Enter with WFE and SEVONPEN = 1: Peripheral enable interrupt (even if disabled in NVIC)
Wake-up delay	None

5.3.5 Low power sleep mode

IO status in low power sleep mode

In low power sleep mode, all IO states remain consistent with the running state.

Low power sleep	Description
Mode entry	The low power sleep mode can only be accessed from the low power operating mode WFI or WFE and: --SLEEPDEEP = 0 and --There is no interrupt hanging.
	The low power sleep mode can only be accessed from the low power operating mode Returns from the lowest priority ISR, and: --SLEEPDEEP = 0 and --There is no interrupt hanging.
Mode Exit	Enter using WFI or returning from lowest priority ISR: Interrupt (enabled) Entered with WFE and SEVONPEN = 0: Wake event Enter with WFE and SEVONPEN = 1: Peripheral enable interrupt (even if disabled in NVIC) Exit the low power sleep mode and the system enters the low power operation mode.
Wake-up delay	None

5.3.6 Stop mode

The stop mode is based on kernel SleepDeep mode and peripheral clock gating. The VDDD domain LDO can be configured in either normal mode (MR) or low power mode (LPR). In the stop mode, all high-frequency clocks in the VDDD domain will stop, and the stopped clocks include PLL, HSI16, HSI48 and HSE, where PLL, HSI48 and HSE are turned off by software control, and HSI16 is turned off by hardware. The internal SRAM and register contents will be preserved.

IO status in stop mode

In stop mode, all IO states remain consistent with the running state.

Entry of stop mode

If FLASH programming is being executed, the entry of stop mode will be delayed until the end of the memory access (controlled by the HREADY signal of the FLASH controller interface and the operation flag, and the software can execute WFI or WFE instructions after the end).

Before entering the stop mode, if the APB domain is being accessed, the entry of the stop mode will be delayed until the end of the APB access. (via Software Assurance)

In stop mode, the following functions can be selected by programming each control bit:

- Independent Watchdog (IWDG): The IWDG is booted by writing to its key register or using a hardware option. Once started, it cannot be stopped except upon a reset.
- Real Time Clock (RTC): Configured via the RTCEN bit in the RCC backup domain control register (RCC_BDCR).
- Internal RC Oscillator (LSI RC): Configured by RCC clock control and LSION bit in the status register (RCC_CSR).
- External 32.768kHz oscillator(LSE OSC): Configured via the LSEON bit in the RCC backup domain control register (RCC_BDCR).

In Stop mode, some low-power peripherals can also be configured as wake-up sources, such as LPUART and I2C;

Before entering stop mode, some analog modules (ADC, DAC) can also work. In order to further reduce power consumption, please turn off before entering.

Exit of Stop Mode

Exiting stop mode, HSI16 will be selected as the system clock. If you are configured to exit the stop mode and enter the low power operation mode, you need to configure the system clock below 2 MHz before entering the stop mode; At the same time, select the power supply mode MR (STOP0) or LPR (STOP1) of VDDD.

When entering Stop, the required delay after startup is different according to the configured VDDD power supply mode.

Stop mode	Description
Mode entry	WFI or WFE and: --No interrupt (for WFI) or event (for WFE) pending; --SLEEPDEEP = 1; --Configure LPMS in PWR_CR register to 00 (STOP0) or 01 (STOP1)
	Returns from the lowest priority ISR, and: --SLEEPDEEP = 1; --No interrupt hanging; --Configure LPMS in PWR_CR register to 00 (STOP0) or 01 (STOP1)
Mode Exit	Enter using WFI or back from ISR: any EXTI line configured to interrupt mode (while enabling the corresponding EXTI interrupt in the NVIC). The interrupt source is an external interrupt or an interrupt generated by a peripheral device having a wake-up function. Enter with WFE and SEVONPEND = 0: Any EXTI line configured to event mode. Enter using WFE and SEVONPEND = 1: --Any EXTI line configured to interrupt mode (even if the EXTI interrupt corresponding to the NVIC interrupt is disabled). The interrupt source is an external interrupt or an interrupt generated by a peripheral device having a wake-up function. --wakeup event NRST Reset IWDG reset
Wake-up delay	HSI16 wake-up time + (LDO exit time from low power mode (different LDO modes have different times)) + FLASH wake-up time (may coincide according to the configuration time)

5.3.7 Standby mode

In standby mode, VR (including MR and LPR) stops working, and the VDDD domain is powered off. PLL, HSI 16, HSI 48, HSE are all closed, with PLL, HSI 48, and HSE being closed by software control, and HSI 16 being closed by hardware. Except for the registers of the VBKP domain (RTC register and backup register) and standby circuit, other SRAM1 and CCM SRAM and register contents are lost. Depending on the configuration SRAM2, you can choose whether to be powered by the LPR alone.

IO status in standby mode

In standby mode, IO can be configured as pull-up and pull-down, or analog mode;

The output port PC13 of the RTC, the two ports PC14 and PC15 of the LSE, the WKUP IO and the TAMPER IO are all in the working state;

Entry of standby mode

In Standby mode, the following features can be selected by programming individual control bits:

- Independent Watchdog (IWDG): The IWDG is booted by writing to its key register or using a hardware option. Once started, it cannot be stopped except upon a reset.
- Real Time Clock (RTC): Configured via the RTCEN bit in the RCC backup domain control register (RCC_BDCR).
- Internal RC Oscillator (LSI RC): Configured by RCC clock control and LSION bit in the status register (RCC_CSR).
- External 32.768kHz oscillator(LSE OSC): Configured via the LSEON bit in the RCC backup domain control register (RCC_BDCR).

Standby mode	Description
Mode entry	WFI or WFE and: --No interrupt (for WFI) or event (for WFE) pending; --SLEEPDEEP = 1; -configure LPMS in the PWR_CR register to 11; -configure the WUF bit clearing in the PWR_CSR register; -clearing the flag corresponding to the selection of the wake source;
	Returned from ISR, and: --SLEEPDEEP = 1; --SLEEPONEXIT = 1; --No interrupt hanging; -configure LPMS in the PWR_CR register to 11; -clearing the WUF bit in the PWR_CSR register; -clearing the flag corresponding to the selection of the wake source;
Mode Exit	WKUPx pin (x = 1, 2, 3, 4, 5)
	RTC wake event generation
	NRST Pin Reset
	IWDG reset
Wakeup latency	HSI16 wake-up time + LDO power-on time + LDO waiting time (configurable 0us/5us/10us/20us) + FLASH waiting time 3us (may coincide according to the configuration time) + internal calibration value load time

In standby mode, except for the following parts, the remaining IOs are in a high impedance state:

- Reset pin
- TAMPER pin when set to anti-intrusion or calibration output
- 5 WKUP pins (PA0/PC13/PE6/PA2/PC5) if enabled

5.3.8 Debugging of stop mode and standby mode

In stop and standby modes, by default, the debugging function will be interrupted because the CPU core is clockless. However, if the relevant register in DBGMCU_CR are configured, debugging can still be performed even if the CPU enters SLEEPDEEP mode.

5.3.9 Automatic wake-up in low power mode (AWU)

The RTC can wake up the MCU in the low power mode without relying on an external interrupt (automatic wake-up mode). The RTC provides a programmable time base for periodically waking up from stop or standby modes. By programming the RTCSEL [1: 0] bit of the backup area control register (RCC_BDCR), two of the three RTC clock sources can be selected to implement this function:

- Low power 32.768 kHz external crystal oscillator (LSE)

The clock source provides a low-power and accurate time reference. (consumes less than 1 μ A under typical conditions)

- Low power internal RC oscillator (LSI)

Using this clock source, the cost of a 32.768 kHz crystal oscillator is saved. But that RC oscillator will increase power consumption slightly. In order to wake the system from stop mode with an RTC alarm event, you must do the following:

- Configure EXTI17 as rising edge trigger.
- Configure the RTC so that it can generate wake-up events.

If you want to wake up from standby mode, you do not have to configure EXTI17 (wake up standby directly with the RTC alarm wake event).

The peripheral registers can be accessed by half-words (16-bit) or words (32-bit).

5.4 TIMx registers

These peripheral registers can be operated in a half-word (16-bit) or word (32-bit) manner.

5.4.1 Power Control Register 1 (PWR_CR1) (0x00)

Address offset: 0x00

Reset value: 0x0001 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.											STDBY_MRRDY_WAIT		FLS_WUPT		HSION_CTRL
-											RW		RW		RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	LPR	Res	SRAM_RETV		VOS		DBP	Res	Res	Res	Res	Res	Res	LPMS	
-	RW	-	RW		RW		RW	-						RW	

Bit	Name	R/W	Reset Value	Function
31: 21	Reserved	-	-	-
20: 19	STDBY_MRRDY_WAIT	RW	2'h0	After the standby mode wakes up and MR is ready, the time you need to wait. 00: 3 μs; 01: 5us; 10: 10 μs; 11: 20 μs
18: 17	FLS_WUPT	RW	2'b00	In the stop mode wake-up sequence, after the HSI is stabilized, a waiting time is required before the FLASH operation. 00: 3 μs; 01: 5 μs; 10: 2 μs; 11: 0 μs;
16	HSION_CTRL	RW	1	When waking up from stop mode, the HSI turns on time control. 0: After waiting for MR to stabilize, enable HSI; 1: Turn on at the same time as VR, that is, HSI is enabled immediately when waking up.
15	Reserved	-	-	-
14	LPR	RW	0	VR working mode selection; 0: Working in main mode (MR) 1: Working in low power mode (LPR)
13	Reserved	-	-	-
12: 11	SRAM_RETV	RW	2'b00	Used to control the voltage of SRAM2 in STANDBY mode SRAM_RETV = 2b '00, high voltage mode SRAM_RETV = 2b '01, medium voltage mode SRAM_RETV = 2b '10, medium and low voltage mode SRAM_RETV = 2b '11, low voltage mode
10: 9	VOS	RW	2'b00	Select the VR output voltage level. This bit is used to control the output voltage of the internal VR in order to achieve balance between the device and power consumption. When VR is in LPR mode: 00: high voltage mode 01: medium voltage mode

				10: medium and low voltage mode 11 low voltage mode When VR is in MR mode: 0x: high voltage mode; 1x: low voltage mode
8	DBP	RW	0	Backup domain register write protection. 0: RTC and backup registers are not accessible; 1: RTC and backup registers can be accessed;
1: 0	LPMS	RW	0	Low power mode selection 00: Stop0 (MR work) 01: Stop1 (LPR work) 10: Reserve 11: STANDBY (VR OFF)

5.4.2 Power Control Register 2 (PWR_CR2) (0x04)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	FLT_CTRL			FLTDIS	Res.	Res.	Res.	Res.	PLS			PVDE
-				RW			RW	-				RW			RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11: 9	FLT_CTRL	RW	3’b000	PVD filtering time configuration. 111: Reserved; 110: The filtering time is about 1024 filtering clocks (about 30.7 ms for LSI or LSE); 101: The filtering time is about 128 filtering clocks (about 3.8 ms for LSI or LSE); 100: The filtering time is about 64 filtering clocks (about 1.92 ms for LSI or LSE); 011: The filtering time is approximately 16 filtering clocks (approximately 480μs for LSI or LSE); 010: The filtering time is about 4 filtering clocks (about 120μs for LSI or LSE); 001: The filtering time is about 2 filtering clocks (about 60μs for LSI or LSE); 000: The filtering time is about 1 filtering clock (about 30μs for LSI or LSE);
8	FLTEN	RW	0	PVD digital filtering enabled. 0: PVD digital filtering disabled 1: PVD digital filter enabled
7: 4	Reserved	-	-	-
3:1	PLS [2: 0]	RW	0	PVD level selection. Written by the software for selecting the voltage threshold of the PVD. 000: Reserved 001: Reserved 010: Reserved 011: VPVD3 (around 2.4 V) 100: VPVD4 (around 2.6 V) 101: VPVD5 (around 2.8 V) 110: VPVD6 (around 3.0 V) 111 external IO input
0	PVDE	RW	0	PVD enabled 0: PVD prohibited 1: Enable PVD

5.4.3 Power Control Register 3 (PWR_CR3) (0x08)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	RRS	Res.	Res.	Res.	EWUP5	EWUP4	EWUP3	EWUP2	EWUP1
-							RW	-			RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	RRS	RW	0	SRAM2 data retention in STANDBY mode 0: SRAM2 is powered off in STANDBY mode; 1: SRAM2 is powered by LPR in STANDBY mode, and data is held.
7: 5	Reserved	-	-	-
4	EWUP5	RW	0	Enable WKUP5 pin. This bit is set to 1 and cleared by the software. 0: The WKUP5 pin is for general purpose I/O. Events on the WKUP5 pin do not wake up the device from standby mode. 1: The WKUP5 pin is used to wake up from standby mode, the wake edge is determined by WP5.
3	EWUP4	RW	0	Enable WKUP4 pin. This bit is set to 1 and cleared by the software. 0: The WKUP4 pin is for general purpose I/O. Events on the WKUP4 pin do not wake up the device from standby mode. 1: The WKUP4 pin is used to wake up from standby mode, and the wake edge is determined by WP4.
2	EWUP3	RW	0	Enable WKUP3 pin. This bit is set to 1 and cleared by the software. 0: The WKUP3 pin is for general purpose I/O. Events on the WKUP3 pin do not wake up the device from standby mode. 1: The WKUP3 pin is used to wake up from standby mode, and the wake-up edge is determined by WP3.
1	EWUP2	RW	0	Enable the WKUP2 pin. This bit is set to 1 and cleared by the software. 0: The WKUP2 pin is for general purpose I/O. Events on the WKUP2 pin do not wake up the device from standby mode. 1: The WKUP2 pin is used to wake up from standby mode, and the wake-up edge is determined by WP2.
0	EWUP1	RW	0	Enable the WKUP1 pin. This bit is set to 1 and cleared by the software. 0: The WKUP pin is for general purpose I/O. Events on the WKUP pin do not wake up the device from standby mode. 1: The WKUP pin is used to wake up from standby mode, and the wake edge is determined by WP1.

5.4.4 Power Control Register 4 (PWR_CR4) (0x0C)

Address offset: 0x0C

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WP5	WP4	WP3	WP2	WP1
-											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 5	Reserved	-	-	-
4	WP5	RW	0	WKUP5 pin polarity selection, this bit is set to 1 and cleared by the software. 0: WKUP pin rising edge wakes up; 1: Interrupt is inhibited
3	WP4	RW	0	WKUP4 pin polarity selection, this bit is set to 1 and cleared by the software. 0: WKUP pin rising edge wakes up; 1: Interrupt is inhibited
2	WP3	RW	0	WKUP3 pin polarity selection, this bit is set to 1 and cleared by the software.

				0: WKUP pin rising edge wakes up; 1: Interrupt is inhibited
1	WP2	RW	0	WKUP2 pin polarity selection, this bit is set to 1 and cleared by the software. 0: WKUP pin rising edge wakes up; 1: Interrupt is inhibited
0	WP1	RW	0	WKUP1 pin polarity selection, this bit is set to 1 and cleared by the software. 0: WKUP pin rising edge wakes up; 1: Interrupt is inhibited

5.4.5 Power status register (PWR_SR) (0x10)

Address offset: 0x10

Reset value: 0x0000 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDO
-															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	MR_RDY	SBF	Res.	Res.	Res.	WUF5	WUF4	WUF3	WUF2	WUF1
-						R	R	-			R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	PVDO	R	0	PVD output flag. This bit is set to 1 and cleared by hardware. This bit is valid only if the PVD is working. 0: VCC is above the PVD threshold for PLS [2: 0] bit selection. 1: The VCC is below the PVD threshold for PLS [2: 0] bit selection. Note: The PVD is turned off when entering standby mode. Therefore, after standby mode or after reset, this bit is equal to 0. After the PVD is enabled, the corresponding value is output according to the detected voltage threshold.
15: 10	Reserved	-	-	-
9	MR_RDY	R	1	Used to indicate MR operating status 0: MR off; 1: MR work;
8	SBF	R	0	Standby mode flag. This bit is set to 1 by hardware, and clearing can only be achieved by POR/PDR or by placing CSBF position 1 in the PWR_SCR register. 0: Device does not enter standby mode 1: The device has entered standby mode Note: After the standby mode wakes up, the software needs to clear this bit
7: 6	Reserved	-	-	-
4	WUF5	R	0	WKUP5 wakeup flag. This bit is set to 1 by hardware, and it can only be cleared by POR/PDR or by clearing CWUF5 position 1 in the PWR_SCR register. 0: No wake-up event occurred 1: Received WKUP5 wake event
3	WUF4	R	0	WKUP4 wakeup flag. This bit is set to 1 by hardware, and it can only be cleared by POR/PDR or by clearing CWUF4 position 1 in the PWR_SCR register. 0: No wake-up event occurred 1: Received WKUP4 wake event
2	WUF3	R	0	WKUP3 wakeup flag. This bit is set to 1 by hardware, and it can only be cleared by POR/PDR or by clearing CWUF3 position 1 in the PWR_SCR register. 0: No wake-up event occurred 1: Received WKUP3 wake event
1	WUF2	R	0	WKUP2 wakeup flag.

				This bit is set to 1 by hardware, and clearing can only be cleared by POR/PDR or by clearing CWUF2 position 1 in the PWR_SCR register. 0: No wake-up event occurred 1: Received WKUP2 wake event
0	WUF1	R	0	WKUP1 wakeup flag. This bit is set to 1 by hardware, and it can only be cleared by POR/PDR or by clearing CWUF1 position 1 in the PWR_SCR register. 0: No wake-up event occurred 1: Received WKUP1 wake event

5.4.6 Power Status Clear Register (PWR_SCR) (0x14)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res.	Res	Res	Res	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSBF	Res.	Res.	Res.	CWUF5	CWUF4	CWUF3	CWUF2	CWUF1
-							W	-			W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 5	Reserved	-	-	-
8	CSBF	W	0	SBF wakeup flag clear signal. The SBF signal in PWR_SCR is cleared by setting this bit to 1
7: 6	Reserved	-	-	-
4	CWUF5	W	0	WKUP5 wakeup flag clear signal. The WUF5 signal in PWR_SCR is cleared by setting this bit to 1
3	CWUF4	W	0	WKUP4 wakeup flag clear signal. Clear the WUF4 signal in PWR_SCR by setting this bit to 1
2	CWUF3	W	0	WKUP3 wakeup flag clear signal. The WUF3 signal in PWR_SCR is cleared by setting this bit to 1
1	CWUF2	W	0	WKUP2 wakeup flag clear signal. The WUF2 signal in PWR_SCR is cleared by setting this bit to 1
0	CWUF1	W	0	WKUP1 wakeup flag clear signal. Clear the WUF1 signal in PWR_SCR by setting this bit to 1

6. Reset and Clock Control (RCC)

6.1 Introduction

6.1.1 Main Features

This module completes the following functions:

- Generate system clock and system reset
- Generate individual module clock and reset
- Generate a clock source control signal

6.2 System reset

System reset resets all registers to reset values except the reset flag in the clock control register CSR and the registers in the backup domain.

System reset is generated when one of the following events occurs:

- NRST pin low (external reset)
- Window Watchdog Count End (WWDG Reset)
- End of Independent Watchdog Count (IWDG Reset)
- Software Reset (SYSRESETREQ Reset)
- Low Power Management Reset (NRST_STDBY/NRST_STOP)
- Option Byte Load Reset

6.2.1 NRST pin (external reset)

With a specific option bit (NRST_MODE), the NRST pin can be configured to:

- Reset Inputs/Outputs (default when device is delivered)

Any valid reset signal on the reset input/output (default when device is delivered) pin is propagated to device internal logic, and all internal reset sources are driven to this pin via a pulse generator. The GPIO function (PF5) is not available. The pulse generator guarantees a minimum reset pulse duration of 20 μ s for each internal reset source output on the NRST pin. This mode is always active (independent of the option byte setting) during each device power-on reset (until the option byte is loaded).

- Reset input

In this mode, any valid reset signal on the NRST pin is propagated to the internal logic of the device, the reset generated inside the device. In this configuration, the GPIO function (PF5) is not available.

- GPIO

In this mode, the pin can be used as a PF5 standard GPIO. The reset function of the pin is not available. The reset can only be done from the device internal reset source and does not propagate to this pin.

6.2.2 Software reset

This can be determined by looking at the reset flag in the RCC clock control and status register (RCC_CSR). To software reset the device, you must interrupt and reset the SYSRESETREQ position 1 in the CPU control register.

6.2.3 Low Power Management Reset

There are two ways to trigger a low-power management reset:

- Reset occurs when entering standby mode:

This reset is enabled by clearing the nRST_STDBY bit in the user option byte. And the PWR_CR.PDDS register is configured to standby mode. When entering the low power mode,the device will reset instead of entering the standby mode.

- Reset is generated when entering stop mode:

This reset is enabled by clearing the nRST_STOP bit in the user option byte. And the PWR_CR.PDDS register is configured to stop mode. When entering the low power mode,the device will reset instead of entering the stop mode.

6.2.4 Option Byte Load Reset

When the OBL_LAUNCH bit (bit 27) is set in the FLASH_CR register, an option byte load reset is generated. This bit is used to load via the software boot option byte.

6.3 Power reset

A power reset is generated when one of the following events occurs:

- Power On/Power Off Reset (POR/PDR Reset)
- Exiting Standby mode

A power reset resets all registers except the backup domain.

The reset entry vector is fixed at address 0x0000_0004 in the memory map.

6.4 Backup domain reset

A backup domain reset will reset all backup domain registers, including RCC_BDCR, backup registers, and RTC partial registers.

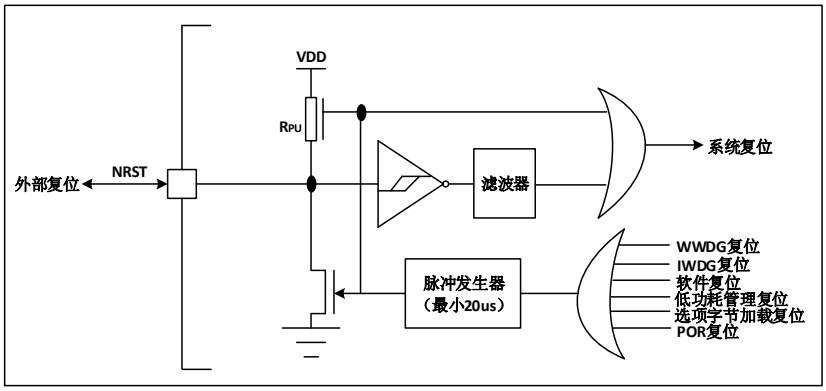
A backup domain reset is generated when one of the following events occurs:

- Software reset, triggered by setting the BDRST bit in the RTC domain control register (RCC_BDCR).
- VCC or VBAT power on, if both supplies have previously been powered off.

6.5 Uniform handling of resets other than backup domain resets

Except for the backup domain reset, in the NRST pin reset input/output mode, other source resets act on the NRST pin, which keeps the clock low during the reset process.

The circuit is shown in the figure below:



When the system is internally reset (high level is active), the pulse generator starts to generate a pulse signal and ensures that the pulse signal lasts for at least 20μs. This pulse signal turns on the N-tube. When the N tube is turned on, the voltage of the NRST pin will be continuously pulled down. When it

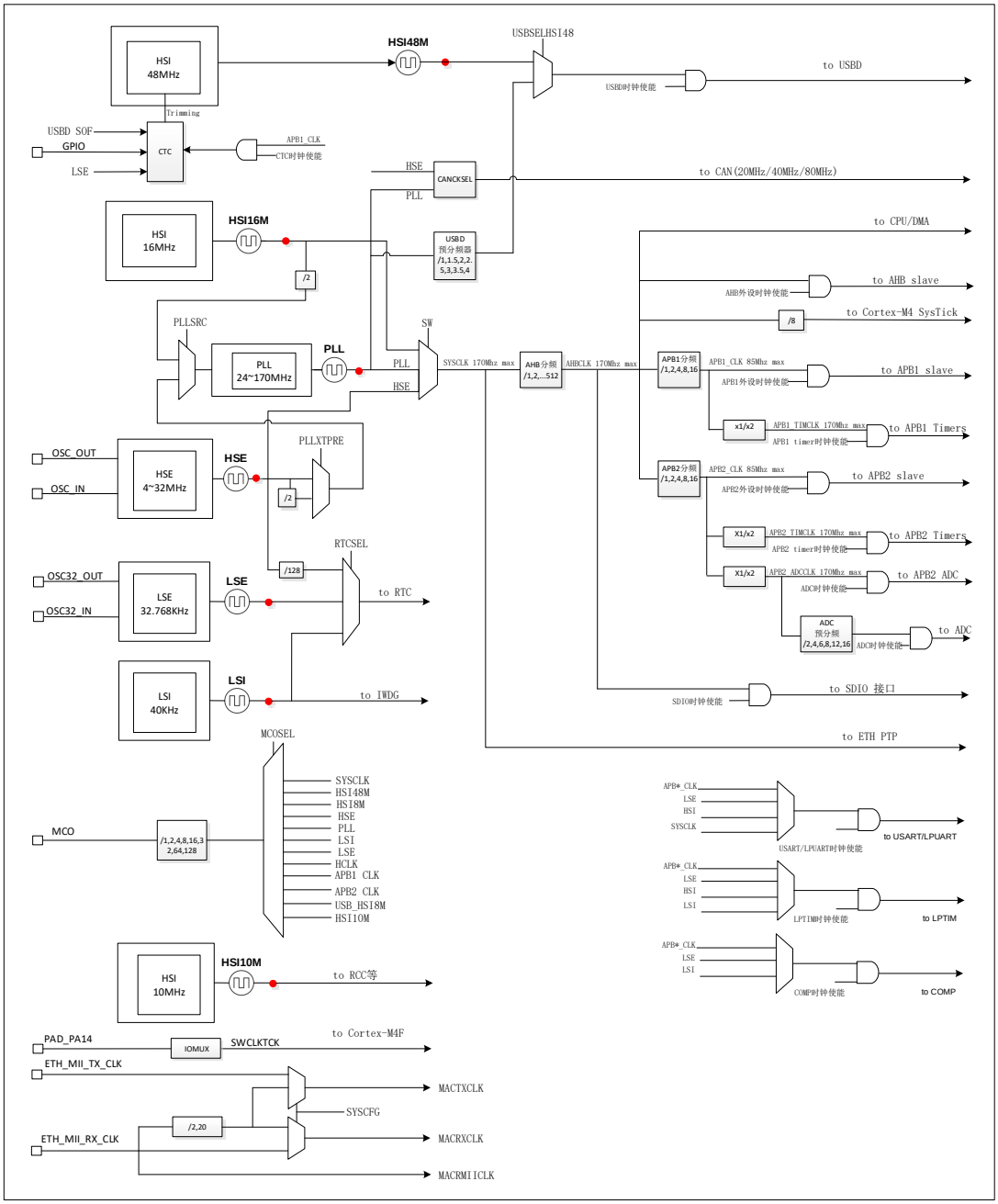
is pulled down to VIL, the NRST pin will generate a low-level signal. After the signal is Schmidt triggered and filtered, a system reset is generated (low active).

Note 1: Reset sources other than the external PIN reset source, that is, the source of the pulse generator, will not produce system reset after 30 us filtering;

Note 2: After the reset source of the pulse generator generates a system reset, in addition to its own reset flag, the external pin reset flag will also be set.

6.6 Clock structure

The clock structure is shown in the following figure:



6.7 Clock sources

The following clock sources exist in the system:

6.7.1 HSE clock

External High Frequency OSC. Frequency Range 4 to 32 MHz.

The settling time of the HSE clock is counted by the analog HSE module, which generates the HSE_RDY signal after starting and settling.

The HSE clock can have two operating modes:

- External XTAL OSC + internal start circuit (HSEBYP = 0)
- External clock input via OSC_IN IO (HSEBYP = 1)

In order to ensure that the first CLOCK output by CLOCK is a stable CLOCK, HSE contains a counter inside, and the rising edge of the first CLOCK is output after the counting is completed. The counting period at HSEBYP = 1 is half that of HSEBYP = 0.

6.7.2 HSI16 clock

Internal 16 MHz RC oscillator. Compared with HSE, it has low power consumption, short stabilization time, but low accuracy. The analog HSI modulegenerates an HSI_ RDY signalafter starting and stabilizing.

After the power-on reset, during the load phase, the calibration value of the HSI is loaded into the RCC_CR.HSICAL register.

After waking up from stop mode, the HSI will act as the system clock source.

6.7.3 HSI48 clock

The HSI48 clock supports self-calibration through the CTC module, and the HSI48 is used as the clock of the USBD module.

6.7.4 PLL clock

The PLL module is used to double the frequency of HSI or HSE clocks. The PLL input clock frequency range is 2 ~ 32 MHz, and the output clock frequency range is 24 ~ 170 MHz.

The PLL lockout time is typically 25 μs and the maximum is 100 μs.

6.7.5 LSE clock

External 32.768 kHz OSC, used as a low power clock.

You can balance settling time and power consumption by configuring LSEDRV. The LSE stabilization time is 0.5 s (typical). The LSE stabilization signal is generated by the LSE analog module.

Similar to the HSE source, the LSE also has two sources:

- 32.768 kHz XTAL + internal start circuit (LSEBYP = 0)
- External clock viaOSC32_ IN input (LSEBYP = 1)

In order to ensure that the first CLOCK output by CLOCK is a stable CLOCK, LSE contains a counter inside, and the rising edge of the first CLOCK is output after the counting is completed. The counting period at LSEBYP = 1 is half that of LSEBYP = 0.

In the case ofLSEBYP = 1, the clock output by the MCO is the original embedded clock and is not controlled by the LSE_ON signal. However, the logic using the LSE clock is controlled by the LSE_ON signal.

6.7.6 LSI clock

Internal low frequency 40 kHz clock. Used as an IWDG, RTC clock.

6.7.7 HSI10M clock

This clock acts as a low precision clock and serves as a filtered count for the NRST pin. Every time a system reset is generated, after passing through the pulse generator, it is returned through the NRST pin, so the HSI10M clock is enabled every time a reset is generated.

6.8 Clock security system (CSS)

The clock security system can be activated by software. In this case, the clock detector is enabled after the HSE oscillator startup delay and disabled when this oscillator is stopped.

If a fault is detected on the HSE clock, the HSE oscillator is automatically disabled and the clock fault event is sent to the interrupt input of the timers (TIM1/TIM8 and TIM15/16/17) and an interrupt is generated to notify the software of the fault (Clock Safe System Interrupt CSSI), allowing the MCU to perform rescue operations. CSSI is linked to Cortex via an FPU NMI (unmaskable interrupt) exception vector® -M4F.

Note: Once CSS is enabled, if the HSE clock fails, a CSS interrupt occurs and an NMI is automatically generated. NMI executes indefinitely unless the CSS interrupt suspend bit is cleared. Therefore, in the NMI ISR, the user must clear the CSS interrupt by setting the CSSC bit in the clock interrupt register (RCC_CIR

If the HSE oscillator is used directly or indirectly as the system clock (indirectly meaning: it is used as the PLL input clock and the PLL clock is used as the system clock), the detected failure causes the system clock to switch to the HSI16 oscillator and disables the HSE oscillator. If the HSE clock is the clock source of the PLL used as the system clock, the PLL is also disabled.

6.9 Peripheral clock enable register

Each peripheral clock may be enabled by the xxxxEN bit of the RCC_AHBxENR, RCC_APBxENRy register.

When the peripheral clock is not active, read or write access to peripheral registers is not supported.

The enable bit has a synchronization mechanism, after the enable bit is set, there is a delay of 2 clock cycles before the clock is activated.

Note: After enabling the clock for the peripheral device, the software must wait some time before accessing the peripheral registers.

6.10 TIMx registers

This module register can be accessed by bytes (8 bits) [except RCC_BDCR], half-words (16 bits), or words (32 bits).

6.10.1 Clock control register (RCC_CR)

Address offset: 0x00

Reset value: 0x0080_2083

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.						PLLRDY	PLLON	HSICAL [10: 8]			Res.	CSSON	HSEBYP	HSERDY	HSEON
-						R	RW	R	R	R	-	RW	RW	R	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSICAL [7: 0]								HSITRIM [4: 0]					HSIKERON	HSIRDY	HSION
R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	R	RW

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	PLLRDY	R	0	PLL clock ready flag.

				Hardware set, indicating that the PLL clock is stable. 0: PLL is not ready; 1: PLL ready;
24	PLLON	RW	0	PLL enabled This bit cannot be cleared when the PLL clock is used (SWS is PLL) or will be used as the system clock (SW selects PLL). 0: PLL OFF; 1: PLL ON
23: 21	HSICAL [10: 8]	R	3'h4	The HSI16 clock calibration value is 3 bits higher. During the loading phase of the power-on process, this register loads the Flash information area value. When the software reads this register, the return value is HSICAL + HSITRIM. When the HSITRIM value changes, the register value will also be updated.
20	Reserved	-	-	-
19	CSSON	RW	0	HSE clock security system enable. When the bit is 1, the hardware enables the clock detection module if the HSE OSC is ready; If the HSE detection fails, the clock detection module is turned off. (This bit can only be written to 1, and the system resets and clears it) 0: Clock safety system off (clock detection off); 1: The clock safety system is turned on (if the HSE clock is stable, the clock detection is turned on, otherwise the clock detection is turned off);
18	HSEBYP	RW	0	HSE shielded crystal oscillator, select pin input clock. This bit can only be written if HSEON = 0. 0: HSE crystal oscillator is not shielded, and external high-speed clock selects crystal oscillator; 1: HSE crystal oscillator shielding, external high-speed clock select external pin input clock source;
17	HSERDY	R	0	HSE crystal oscillator clock ready flag. This bit is set by hardware to indicate that the HSE crystal oscillator is stable. 0: HSE crystal oscillator is not ready; 1: HSE crystal ready
16	HSEON	RW	0	HSE crystal oscillator enabled. When HSE is directly or indirectly the system clock source, this bit cannot be set to 0. 0: HSE crystal OFF 1 HSE crystal oscillator ON
15: 8	HSICAL [7: 0]	R	8'h20	HSI16 clock calibration value. During the loading phase of the power-on process, this register loads the Flash information area value. When the software reads this register, the return value is HSICAL + HSITRIM. When the HSITRIM value changes, the register value will also be updated.
7: 3	HSITRIM [4: 0]	RW	5'h10	HSI16 clock Trimming value. The software writes to this register to adjust the HSI 16 clock, superimposes it on the HSICAL and outputs it to the analog HSI 16. A system reset resets this register.
2	HSIKERON	RW	0	The HSI 16 is provided as a kernel clock configuration to peripherals USART and I2C, etc. Keeping the HSI 16 open in the stop mode avoids slowing down the communication speed due to the HSI 16 startup time. This bit has no effect on the HSION value. 0: No effect on the HSI16 oscillator. 1: The HSI16 oscillator is forcibly enabled even in stop mode.
1	HSIRDY	R	1	HSI16 clock ready flag. Set by hardware to indicate that the HSI oscillator is stable This bit is only valid if HSION = 1. 0: HSI16 OSC not ready; 1: HSI16 OSC ready; When HSION is cleared, HSIRDY will pull down after 6 HSI16 clocks.
0	HSION	RW	1	HSI16 clock enabled. When the hardware enters the stop or standby mode, it will clear the register to stop HSI16 as needed.

				When HSI 16 is directly or indirectly used as the system clock, this register cannot be 0. 0: HSI16 OSC OFF; 1: HSI16 OSC ON; The hardware enables HSI 16 when: 1) After the hardware wakes up from stop or standby mode; 2) HSE direct/indirect as system clock but HSE CSS fail occurs;

6.10.2 Clock configuration register (RCC_CFGR)

Address offset: 0x04

Reset value: 0x0000_0000

When the clock source is switched, there is a waiting period of 1 or 2 clocks to access this register.

When the APB or AHB frequency division value is updated, there may be a waiting period of 0 to 15 clocks to access this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	MCO [3: 0]				SRAM2PRE	USBPRE [2: 0]			Res.	ADCPRE [2: 0]		
-				RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	PPRE2 [2: 0]			PPRE1 [2: 0]			HPRE [3: 0]				SWS [1: 0]		SW [1: 0]	
-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	R	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 24	MCO [3: 0]	RW	4'h0	MCO output clock selection. 0000: MCO does not output 0001: LSE 0010: LSI40K 0011: HSI48 0100: SYSCLK 0101 HSI16 0110: HSE 0111: PLL 1000: HCLK 1001: APB1 clock 1010: APB2 clock 1011: HSI10M others: no output Note 1: When selecting the system clock for MCO output, it is necessary to ensure that the output frequency does not exceed the maximum allowable frequency of IO.
23	SRAM2PRE	RW	0	SRAM2 clock division factor. Frequency division coefficients based on HCLK. 0: no frequency division 1: 2 frequency division
22: 20	USBPRE [2: 0]	RW	3'h0	USB clock crossover. 000: PLL clock divided by 1.5 001: PLL clock 010: PLL clock divided 2.5 011: PLL clock divided by two 100: PLL clock divided into 3 101: PLL clock 3.5 divided 11x: PLL clock divided into four
19	Reserved	-	-	-
18: 16	ADCPRE [2: 0]	RW	3'h0	Configure the frequency division factor of the ADC clock. 000: APB2 (ADC) Division of 2 001: APB2 (ADC) Division 4 010: APB2 (ADC) Division 6 011: APB2 (ADC) Division 8 100: APB2 (ADC) Division 2 101: APB2 (ADC) division of 12 110: APB2 (ADC) division of 8 111: APB2 (ADC) division 16

15: 14	Reserved	-	-	-
13: 11	PPRE2 [2: 0]	RW	3'h0	High-speed APB (APB2) clock division factor. PCLK is based on the division coefficient of HCLK. 0xx: no frequency division 100: divided by 2 101: 4 division 110: 8 division 111: 16 division
10: 8	PPRE1 [2: 0]	RW	3'h0	High-speed APB (APB1) clock division factor. PCLK is based on the division coefficient of HCLK. 0xx: no frequency division 100: divided by 2 101: 4 division 110: 8 division 111: 16 division
7: 4	HPRE [3: 0]	RW	4'h0	The AHB clock HCLK is based on the frequency division coefficient of SYSCLK. 0xxx: no frequency division 1000: Division 2 1001: 4 division 1010: 8 division 1011: 16 division 1100: 64 division 1101: 128 division 1110: 256 division 1111: 512 division To ensure the normal operation of the system, the appropriate frequency needs to be config-ured according to the VR power supply situation. Note: It is recommended to switch the frequency division coefficients gradually.
3: 2	SWS [1: 0]	R	2'h0	System clock selection status. This bit is controlled by hardware and indicates the selection state of the system clock source. 00: HSI16 01: HSE 10: PLL CLK Others: reserve
1: 0	SW [1: 0]	RW	2'h0	System clock source selection. This bit is controlled by hardware and software and indicates the selection of the system clock source. 00: HSI16 01: HSE 10: PLL CLK Others: reserve Scenarios where the hardware is configured as HSI include: ● MCU exits from stop, standby mode ● Software configuration is 01 (HSE) but HSE test fails ● Software configuration is 10 and PLL source selects HSE but HSE detection fails

6.10.3 Clock Interrupt Register (RCC_CIR)

Address offset: 0x08

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSC	Res.	HSI48RDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
-								W	-	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res		HSI48RDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	Res.	HSI48RDYF	PLLRDYF	HSERDYF	HSIRDYIF	LSERDYF	LSIRDYF

-	RW	RW	RW	RW	RW	RW	R	-	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	CSSC	W	0	LSE clock security system (CSS) interrupt flag cleared. 0: No effect; 1: Clear CSSF flag
22	Reserved	-	-	-
21	HSI48RDYC	W	0	HSI48 ready interrupt flag cleared. 0: No effect; 1: Clear HSI48RDYF flag;
20	PLLRDYC	W	0	PLL ready interrupt flag is cleared. 0: No effect; 1: Zero PLLRDYF flag;
19	HSERDYC	W	0	HSE ready interrupt flag is cleared. 0: No effect; 1: Clear HSERDYF flag;
18	HSIRDYC	W	0	The HSI ready interrupt flag is cleared. 0: No effect; 1: Clear HSIRDYF flag;
17	LSERDYC	W	0	The LSE ready interrupt flag is cleared. 0: No effect; 1: Zero LSERDYF flag;
16	LSIRDYC	W	0	LSI ready interrupt flag is cleared. 0: No effect; 1: Zero LSIRDYF flag;
15: 14	Reserved	-	-	-
13	HSI48RDYIE	RW	0	HSI48 clock ready interrupt enabled. 0: Interrupt is disabled 1: Enabled
12	PLLRDYIE	RW	0	PLL ready Interrupt enabled. 0: Interrupt is disabled 1: Enabled
11	HSERDYIE	RW	0	HSE ready interrupt enable 0: Interrupt is disabled 1: Enabled
10	HSIRDYIE	RW	0	HSI16 clock ready interrupt enabled. 0: Interrupt is disabled 1: Enabled
9	LSERDYIE	RW	0	LSE ready interrupt enable 0: Interrupt is disabled 1: Enabled
8	LSIRDYIE	RW	0	LSI ready interrupt enable 0: Interrupt is disabled 1: Enabled
7	CSSF	R	0	Clock safety system interrupt flag. When the HSE clock fails, it is set to 1 by hardware. 0: HSE clock safety system interrupt is not generated; 1: HSE clock safety system interrupt generation; Write CSSC register 1 to clear this bit.
6	Reserved	-	-	-
5	HSI48RDYF	R	0	HSI48 clock ready interrupt flag. When the HSI48 clock is stable and HSI48RDYIE = 1, the hardware sets this register. 0: HSI48 clock ready interrupt not generated; 1: HSI48 clock ready interrupt generation; Write HSI48RDYC register 1 to clear this bit.
4	PLLRDYF	R	0	PLL clock ready interrupt flag. When the PLL clock is stable and PLLRDYDIE = 1, the hardware sets this register. 0: PLL clock ready interrupt not generated; 1: PLL clock ready interrupt generated; Write PLLRDYC register 1 to clear this bit.
3	HSERDYF	R	0	HSE ready interrupt enable

				Set by hardware when the HSE clock becomes stable and HSERDYDIE = 1. 0: HSE clock ready interrupt is not generated; 1: HSE clock ready interrupt generated; Write HSERDYC register 1 to clear this bit.
2	HSIRDYF	R	0	HSI16 clock ready interrupt flag. When the HSI16 clock is stable and HSIRDYIE = 1, the hardware sets this register. 0: HSI16 clock ready interrupt not generated; 1: HSI16 clock ready interrupt generation; Write HSIRDYC register 1 to clear this bit.
1	LSERDYF	R	0	LSERDY ready interrupt enable Set by hardware when the LSE clock becomes stable and LSERDYDIE = 1. 0: LSERDY clock ready interrupt not generated; 1: LSERDY clock ready interrupt generated; Write LSERDYC register 1 to clear this bit.
0	LSIRDYF	R	0	LSIRDY ready interrupt enable Set by hardware when the LSI clock becomes stable and LSIRDYDIE = 1. 0: LSIRDY clock ready interrupt not generated; 1: LSIRDY clock ready interrupt generated; Write LSIRDYC register 1 to clear this bit.

6.10.4 AHB1 peripheral reset register (RCC_AHB1RSTR)

Address offset: 0x28

Reset value: 0x0000_0000

The register is set and cleared by software. After the software is set, the module maintains a reset until the software clears.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res -															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	ESMCRST	Res.	Res.	Res.	Res	Res.	CRCRST	CORDICRST	Res			DMA2RST	DMA1RST
-			RW	-					RW	RW	-			RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	ESMCRST	RW	0	ESMC (QSPI) module reset. 0: No effect; 1: Reset
11: 7	Reserved	-	-	-
6	CRCRST	RW	0	CRC reset 0: No effect; 1: Reset
5	CORDICRST	RW	0	CORDIC reset 0: No effect; 1: Reset
4: 2	Reserved	-	-	-
1	DMA2RST	RW	0	DMA2 reset 0: No effect; 1: Reset
0	DMA1RST	RW	0	DMA1 reset 0: No effect; 1: Reset

6.10.5 AHB2 peripheral reset register (RCC_AHB2RSTR)

Address offset: 0x2C

Reset value: 0x0000_0000

The register is set and cleared by software. After the software is set, the module maintains a reset until the software clears.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETHRST	USB2RST	USB1RST	Res.	Res.	AESRST	SDIORST	Res.	IOPFRST	IOPERST	IOPDRST	IOPCRST	IOPBRST	IOPARST	Res.	Res.
RW	RW	RW	-		RW	RW	-	RW	RW	RW	RW	RW	RW	-	

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	ETHRST	RW	0	ETH reset 0: No effect; 1: Reset
14	USB2RST	RW	0	USB2 reset 0: No effect; 1: Reset
13	USB1RST	RW	0	USB1 reset 0: No effect; 1: Reset
12: 11	Reserved	-	-	-
10	AESRST	RW	0	AES reset 0: No effect; 1: Reset
9	SDIORST	RW	0	SDIO reset 0: No effect; 1: Reset
8	Reserved	-	-	-
7	IOPFRST	RW	0	IOPF reset 0: No effect; 1: Reset
6	IOPERST	RW	0	IOPE reset 0: No effect; 1: Reset
5	IOPDRST	RW	0	IOPD reset 0: No effect; 1: Reset
4	IOPCRST	RW	0	IOPC reset 0: No effect; 1: Reset
3	IOPBRST	RW	0	IOPB reset 0: No effect; 1: Reset
2	IOPARST	RW	0	IOPA reset 0: No effect; 1: Reset
1: 0	Reserved	-	-	-

6.10.6 APB1 peripheral reset register 1 (RCC_APB1RSTR1)

Address offset: 0x38

Reset value: 0x0000_0000

The register is set and cleared by software. After the software is set, the module maintains a reset until the software clears.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTCRST	I2C3RST	DACRST	PWRRST	Res.	CAN2RST	CAN1RST	Res.	UART3RST	I2C2RST	I2C1RST	UART2RST	UART1RST	USART3RST	USART2RST	Res.
RW	RW	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3RST	SPI2RST	Res.	Res.	WWDGRST	Res.	TIM18RST	TIM14RST	TIM13RST	TIM12RST	TIM7RST	TIM6RST	TIM5RST	TIM4RST	TIM3RST	TIM2RST
RW	RW	-		RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	CTCRST	RW	0	CTC reset 0: No effect;

				1: Reset
30	I2C3RST	RW	0	I2C3 reset 0: No effect; 1: Reset
29	DACRST	RW	0	DAC reset 0: No effect; 1: Reset
28	PWRRST	RW	0	Power interface reset 0: No effect; 1: Reset
27	Reserved	-	-	-
26	CAN2RST	RW	0	CANFD reset 0: No effect; 1: Reset
25	CAN1RST	RW	0	CAN2.0 reset 0: No effect; 1: Reset
24	Reserved	-	-	-
23	UART3RST	RW	0	UART3 reset 0: No effect; 1: Reset
22	I2C2RST	RW	0	I2C2 reset 0: No effect; 1: Reset
21	I2C1RST	RW	0	I2C1 reset. 0: No effect; 1: Reset
20	UART2RST	RW	0	UART2 reset 0: No effect; 1: Reset
19	UART1RST	RW	0	UART1 reset 0: No effect; 1: Reset
18	USART3RST	RW	0	USART3 reset 0: No effect; 1: Reset
17	USART2RST	RW	0	USART2 reset 0: No effect; 1: Reset
16	Reserved	-	-	-
15	SPI3RST	RW	0	SPI3 reset 0: No effect; 1: Reset
14	SPI2RST	RW	0	SPI2 reset 0: No effect; 1: Reset
13: 12	Reserved	-	-	-
11	WWDGRST	RW	0	WWDG reset 0: No effect; 1: Reset
10	Reserved	-	-	-
9	TIM18RST	RW	0	TIM18 reset 0: No effect; 1: Reset
8	TIM14RST	RW	0	TIM14 reset. 0: No effect; 1: Reset
7	TIM13RST	RW	0	TIM13 reset 0: No effect; 1: Reset
6	TIM12RST	RW	0	TIM12 reset 0: No effect; 1: Reset
5	TIM7RST	RW	0	TIM7 reset 0: No effect; 1: Reset
4	TIM6RST	RW	0	TIM6 reset 0: No effect; 1: Reset

3	TIM5RST	RW	0	TIM5 reset 0: No effect; 1: Reset
2	TIM4RST	RW	0	TIM4 reset 0: No effect; 1: Reset
1	TIM3RST	RW	0	TIM3 reset 0: No effect; 1: Reset
0	TIM2RST	RW	0	TIM2 reset 0: No effect; 1: Reset

6.10.7 APB1 peripheral reset register 2 (RCC_APB1RSTR2)

Address offset: 0x3C

Reset value: 0x0000_0000

The register is set and cleared by software. After the software is set, the module maintains a reset until the software clears.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	I2C4RST	LPUART1RST	LPTIM1RST
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	I2C4RST	RW	0	I2C4 reset 0: No effect; 1: Reset
1	LPUART1RST	RW	0	LPUART1 reset 0: No effect; 1: Reset
0	LPTIM1RST	RW	0	LPTIM1 reset 0: No effect; 1: Reset

6.10.8 APB2 peripheral reset register (RCC_APB2RSTR)

Address offset: 0x40

Reset value: 0x0000_0000

The register is set and cleared by software. After the software is set, the module maintains a reset until the software clears.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	LCDC RST	OPA RST	COM PRST	RNG RST	TIM1 9RST	TIM1 7RST	TIM1 6RST	TIM1 5RST	TIM1 1RST	TIM1 0RST	TIM9 RST	Res.	Res.	Res.
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC3 RST	USART 1RST	TIM8 RST	SPI1 RST	TIM1 RST	ADC2 RST	ADC1 RST	Res								SYSCF GRST
RW	RW	RW	RW	RW	RW	RW	-								RW

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29	LCDCRST	RW	0	LCDC reset 0: No effect; 1: Reset
28	OPARST	RW	0	OPA reset 0: No effect; 1: Reset

27	COMPRST	RW	0	COMP reset 0: No effect; 1: Reset
26	RNGRST	RW	0	RNG reset 0: No effect; 1: Reset
25	TIM19RST	RW	0	TIM19 reset 0: No effect; 1: Reset
24	TIM17RST	RW	0	TIM17 reset 0: No effect; 1: Reset
23	TIM16RST	RW	0	TIM16 reset 0: No effect; 1: Reset
22	TIM15RST	RW	0	TIM15 reset 0: No effect; 1: Reset
21	TIM11RST	RW	0	TIM11 reset 0: No effect; 1: Reset
20	TIM10RST	RW	0	TIM10 reset 0: No effect; 1: Reset
19	TIM9RST	RW	0	TIM9 reset 0: No effect; 1: Reset
18: 16	Reserved	-	-	-
15	ADC3RST	RW	0	ADC3 reset 0: No effect; 1: Reset
14	USART1RST	RW	0	USART1 reset 0: No effect; 1: Reset
13	TIM8RST	RW	0	TIM8 reset 0: No effect; 1: Reset
12	SPI1RST	RW	0	SPI1 reset 0: No effect; 1: Reset
11	TIM1RST	RW	0	TIM1 reset 0: No effect; 1: Reset
10	ADC2RST	RW	0	ADC2 reset 0: No effect; 1: Reset
9	ADC1RST	RW	0	ADC1 reset 0: No effect; 1: Reset
8: 1	Reserved	-	-	-
0	SYSCFGRST	RW	0	SYSCFG reset 0: No effect; 1: Reset

6.10.9 AHB1 peripheral clock enable register (RCC_AHB1ENR)

Address offset: 0x48

Reset value: 0x0000_0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res -															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	ESMCEN	Res.	Res.	Res.	Res	Res.	CRCEN	CORDICEN	FMCEN	Res.	SRAMEN	DMA2EN	DMA1EN
-			RW	-					RW	RW	RW	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	ESMCEN	RW	0	ESMC (QSPI) module clock enabled. 0: Interrupt is disabled 1: Enabled
11: 7	Reserved	-	-	-
6	CRCEN	RW	0	Enable CRC clock 0: Interrupt is disabled 1: Enabled
5	CORDICEN	RW	0	CORDIC clock enable 0: Interrupt is disabled 1: Enabled
4	FMCEN	RW	1	FLASH interface module clock enabled for Sleep mode. 0: Interrupt is disabled 1: Enabled
3	Reserved	-	-	-
2	SRAMEN	RW	1	SRAM memory area clock enabled for Sleep mode. 0: Interrupt is disabled 1: Enabled
1	DMA2EN	RW	0	DMA2 enable 0: Interrupt is disabled 1: Enabled
0	DMA1EN	RW	0	DMA1 enable 0: Interrupt is disabled 1: Enabled

6.10.10 AHB2 Peripheral Clock Enable Register (RCC_AHB2ENR)

Address offset: 0x4C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETHRXEN	ETHTXEN
-														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETHEN	USB2EN	USB1EN	Res.	Res.	AESEN	SIOEN	Res.	IOPFEN	IOPEEN	IOPDEN	IOPCEN	IOPBEN	IOPAEN	Res.	Res.
RW	RW	RW	-		RW	RW	-	RW	RW	RW	RW	RW	RW	-	

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	ETHRXEN	RW	0	ETH module RX clock enabled. 0: Interrupt is disabled 1: Enabled
16	ETHTXEN	RW	0	ETH module TX clock enabled. 0: Interrupt is disabled 1: Enabled
15	ETHEN	RW	0	ETH clock enable 0: Interrupt is disabled 1: Enabled
14	USB2EN	RW	0	USB2 enable 0: Interrupt is disabled 1: Enabled
13	USB1EN	RW	0	USB1 enable 0: Interrupt is disabled 1: Enabled
12: 11	Reserved	-	-	-
10	AESEN	RW	0	AES clock enable 0: Interrupt is disabled 1: Enabled
9	SIOEN	RW	0	SDIO clock enable 0: Interrupt is disabled 1: Enabled
8	Reserved	-	-	-
7	IOPFEN	RW	0	IOPF clock enable

				0: Interrupt is disabled 1: Enabled
6	IOPEEN	RW	0	IOPE clock enable 0: Interrupt is disabled 1: Enabled
5	IOPDEN	RW	0	IOPD clock enable 0: Interrupt is disabled 1: Enabled
4	IOPCEN	RW	0	IOPC clock enable 0: Interrupt is disabled 1: Enabled
3	IOPBEN	RW	0	IOPB clock enable 0: Interrupt is disabled 1: Enabled
2	IOPAEN	RW	0	IOPA clock enable 0: Interrupt is disabled 1: Enabled
1: 0	Reserved	-	-	-

6.10.11 APB1 peripheral clock enable register 1 (RCC_APB1ENR1)

Address offset: 0x58

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTC EN	I2C3 EN	DAC EN	PW REN	BKPE N	CAN 2EN	CAN 1EN	Res.	UART 3EN	I2C2 EN	I2C1 EN	UART 2EN	UART 1EN	USAR T3EN	USAR T2EN	Res.
RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPI3 EN	SPI2 EN	Res	Res.	WWD GEN	Res.	TIM1 8EN	TIM1 4EN	TIM13 EN	TIM1 2EN	TIM 7EN	TIM6 EN	TIM5 EN	TIM4E N	TIM3E N	TIM 2EN
RW	RW	-		RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	CTCEN	RW	0	CTC clock enable 0: Interrupt is disabled 1: Enabled
30	I2C3EN	RW	0	I2C3 enable 0: Interrupt is disabled 1: Enabled
29	DACEN	RW	0	DAC clock enable 0: Interrupt is disabled 1: Enabled
28	PWREN	RW	0	PWR clock enable 0: Interrupt is disabled 1: Enabled
27	BKPEN	RW	0	BACKUP clock enable 0: Interrupt is disabled 1: Enabled
26	CAN2EN	RW	0	CANFD clock enable 0: Interrupt is disabled 1: Enabled
25	CAN1EN	RW	0	CAN2.0 enable 0: Interrupt is disabled 1: Enabled
24	Reserved	-	-	-
23	UART3EN	RW	0	UART3 enable 0: Interrupt is disabled 1: Enabled
22	I2C2EN	RW	0	I2C2 enable 0: Interrupt is disabled 1: Enabled
21	I2C1EN	RW	0	I2C1 enable 0: Interrupt is disabled 1: Enabled
20	UART2EN	RW	0	UART2 enable 0: Interrupt is disabled 1: Enabled
19	UART1EN	RW	0	UART1 enable 0: Interrupt is disabled

				1: Enabled
18	USART3EN	RW	0	USART3 enable 0: Interrupt is disabled 1: Enabled
17	USART2EN	RW	0	USART2 enable 0: Interrupt is disabled 1: Enabled
16	Reserved	-	-	-
15	SPI3EN	RW	0	SPI3 enable 0: Interrupt is disabled 1: Enabled
14	SPI2EN	RW	0	SPI2 enable 0: Interrupt is disabled 1: Enabled
13: 12	Reserved	-	-	-
11	WWDGEN	RW	0	WWDG clock enable 0: Interrupt is disabled 1: Enabled
10	Reserved	-	-	-
9	TIM18EN	RW	0	TIM18 enable 0: Interrupt is disabled 1: Enabled
8	TIM14EN	RW	0	TIM14 enable 0: Interrupt is disabled 1: Enabled
7	TIM13EN	RW	0	TIM13 enable 0: Interrupt is disabled 1: Enabled
6	TIM12EN	RW	0	TIM12 enable 0: Interrupt is disabled 1: Enabled
5	TIM7EN	RW	0	TIM7 enable 0: Interrupt is disabled 1: Enabled
4	TIM6EN	RW	0	TIM6 enable 0: Interrupt is disabled 1: Enabled
3	TIM5EN	RW	0	TIM5 enable 0: Interrupt is disabled 1: Enabled
2	TIM4EN	RW	0	TIM4 enable 0: Interrupt is disabled 1: Enabled
1	TIM3EN	RW	0	TIM3 enable 0: Interrupt is disabled 1: Enabled
0	TIM2EN	RW	0	TIM2 enable 0: Interrupt is disabled 1: Enabled

6.10.12 APB1 peripheral clock enable register 2 (RCC_APB1ENR2)

Address offset: 0x5C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	I2C4EN	LPUART1EN	LPTIM1EN
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	I2C4 EN	RW	0	I2C4 enable 0: Interrupt is disabled 1: Enabled
1	LPUART1EN	RW	0	LPUART1 enable 0: Interrupt is disabled

				1: Enabled
0	LPTIM1EN	RW	0	LPTIM1 enable 0: Interrupt is disabled 1: Enabled

6.10.13 APB2 peripheral clock enable register (RCC_APB2ENR)

Address offset: 0x60

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	LCDCEN	OPAEN	COMPEN	RNGEN	TIM19EN	TIM17EN	TIM16EN	TIM15EN	TIM11EN	TIM10EN	TIM9EN	Res		
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC3EN	USART1EN	TIM8EN	SPI1EN	TIM1EN	ADC2EN	ADC1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGEN
RW	RW	RW	RW	RW	RW	RW	-								RW

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29	LCDCEN	RW	0	LCDC clock enable 0: Interrupt is disabled 1: Enabled
28	OPAEN	RW	0	OPA clock enable 0: Interrupt is disabled 1: Enabled
27	COMPEN	RW	0	COMP clock enable 0: Interrupt is disabled 1: Enabled
26	RNGEN	RW	0	RNG clock enable 0: Interrupt is disabled 1: Enabled
25	TIM19EN	RW	0	TIM19 enable 0: Interrupt is disabled 1: Enabled
24	TIM17EN	RW	0	TIM17 enable 0: Interrupt is disabled 1: Enabled
23	TIM16EN	RW	0	TIM16 enable 0: Interrupt is disabled 1: Enabled
22	TIM15EN	RW	0	TIM15 enable 0: Interrupt is disabled 1: Enabled
21	TIM11EN	RW	0	TIM11 enable 0: Interrupt is disabled 1: Enabled
20	TIM10EN	RW	0	TIM10 enable 0: Interrupt is disabled 1: Enabled
19	TIM9EN	RW	0	TIM9 enable 0: Interrupt is disabled 1: Enabled
18: 16	Reserved	-	-	-
15	ADC3EN	RW	0	ADC3 enable 0: Interrupt is disabled 1: Enabled
14	USART1EN	RW	0	USART1 enable 0: Interrupt is disabled 1: Enabled
13	TIM8EN	RW	0	TIM8 enable 0: Interrupt is disabled 1: Enabled
12	SPI1EN	RW	0	SPI1 enable 0: Interrupt is disabled 1: Enabled
11	TIM1EN	RW	0	TIM1 enable

				0: Interrupt is disabled 1: Enabled
10	ADC2EN	RW	0	ADC2 enable 0: Interrupt is disabled 1: Enabled
9	ADC1EN	RW	0	ADC1 enable 0: Interrupt is disabled 1: Enabled
8: 1	Reserved	-	-	-
0	SYSCFGEN	RW	0	SYSCFG clock enable 0: Interrupt is disabled 1: Enabled

6.10.14 Peripherals independent clock configuration register (RCC_CCIPR)

Address offset: 0x68

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res	Res	Res	Res	COMP4SEL [1: 0]		COMP3SEL [1: 0]		COMP2SEL [1: 0]		COMP1SEL [1: 0]	
-								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPTIM1SEL [1: 0]		LPUART1SEL [1: 0]		Res.						USART3SEL [1: 0]		USART2SEL [1: 0]		USART1SEL [1: 0]	
RW	RW	RW	RW	-						RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23: 22	COMP4SEL [1: 0]	RW	2'h0	COMP4 clock source selection. 00: PCLK 01: LSI 10: LSE 11: Reserved; Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
21: 20	COMP3SEL [1: 0]	RW	2'h0	COMP3 clock source selection. 00: PCLK 01: LSI 10: LSE 11: Reserved; Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
19: 18	COMP2SEL [1: 0]	RW	2'h0	COMP2 clock source selection. 00: PCLK 01: LSI 10: LSE 11: Reserved; Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
17: 16	COMP1SEL [1: 0]	RW	2'h0	COMP1 clock source selection. 00: PCLK 01: LSI 10: LSE 11: Reserved; Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
15: 14	LPTIM1SEL [1: 0]	RW	2'h0	LPTIM1 clock source selection. 00: PCLK 01: LSI 10: HSI

				11: LSE Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
13: 12	LPUART1SEL [1: 0]	RW	2'h0	LPUART1 Clock source selection. 00: PCLK 01: SYSCLK; 10: HSI 11: LSE Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
11: 6	Reserved	-	-	-
5: 4	USART3SEL [1: 0]	RW	2'h0	USART3 clock source selection. 00: PCLK 01: SYSCLK; 10: HSI 11: LSE Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
3: 2	USART2SEL [1: 0]	RW	2'h0	USART2 clock source selection. 00: PCLK 01: SYSCLK; 10: HSI 11: LSE Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.
1: 0	USART1SEL [1: 0]	RW	2'h0	USART1 clock source selection. 00: PCLK 01: SYSCLK; 10: HSI 11: LSE Note: Before switching the clock source, you need to turn off the clock enable, and then turn on the clock enable after the switching is completed.

6.10.15 RTC domain control register (RCC_BDCR)

Address offset: 0x70

Reset value: 0x0000_0000

The LSEON, LSEBYP, LSEDRV, RTCSEL and RTCEN bits in the register are in the backup domain and can be reset only when the backup domain is reset, and the backup domain reset includes a backup domain power-on reset (BPOR) and a backup domain soft reset (BDRST).

When this register is accessed continuously , 0 ≤ wait state ≤ 3.

This register does not belong to the VDDD domain but belongs to the RTC domain (VBAK domain). After reset, the register needs to be write-protected, that is, the PWR_CR1. DBP bit needs to be set to 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															BDRST
-															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL [1: 0]		Res.	Res.	Res.	LSEDRV [1: 0]		LSEBYP	LSERDY	LSEON
RW	-					RW	RW	-			RW	RW	RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	BDRST	RW	0	Backup domain soft reset. 0: No effect; 1: reset
15	RTCEN	RW	0	RTC and TAMP clock enabled. 0: Interrupt is disabled 1: Enabled
14: 10	Reserved	-	-	-
9: 8	RTCSEL [1: 0]	RW	2'h0	RTC clock source selection. 00: No clock 01: LSE 10: LSI HSE divided by 128. Once the RTC clock source is selected, it cannot be changed unless the backup domain reset is cleared.
7: 5	Reserved	-	-	-
4: 3	LSEDRV [1: 0]	RW	2'h0	LSE drive capability settings 00: gm 2.5 uA/V 01: gm 3.75 uA/V 10: gm 8.5 uA/V 11: gm 13.5 uA/V
2	LSEBYP	RW	0	LSE OSC bypass 0: Low speed external clock selection crystal oscillator; 1: Low speed external clock Select external interface input clock; This bit can be written only when the external 32.768 KHz oscillator is disabled (LSEON=0 and LSERDY=0).
1	LSERDY	R	0	LSE OSC ready. Hardware Configuration This bit of 1 indicates that the LSE clock is stable. After LSEON is cleared, this bit needs 6 LSE clocks before clearing.
0	LSEON	RW	0	LSE oscillator enable 0: Interrupt is disabled 1: Enabled

6.10.16 RCC control/status register (RCC_CSR)

Address offset: 0x74

Reset value: 0x0800_0000

The reset flag bit in this register can only be reset by power reset, and the rest by system reset.

When this register is accessed continuously , 0 ≤ wait state ≤ 3.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PWRRSTF	PINRSTF	OBLRSTF	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R	RW	-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSIRDY	LSION
-														R	RW

Bit	Name	R/W	Reset Value	Function
31	LPWRRSTF	R	0	Low power reset flag. The hardware sets this register when the stop/standby low power mode is entered. This register can only be operated if the nRST_STOP, nRST_STDBY option byte is in the clear (active) state. Cleared by writing 1 to the RMVF bit.
30	WWDGRSTF	R	0	Window WDG reset flag.

				Cleared by writing 1 to the RMVF bit.
29	IWDGRSTF	R	0	Independent watchdog reset flag Cleared by writing 1 to the RMVF bit.
28	SFTRSTF	R	0	Software reset flag Cleared by writing 1 to the RMVF bit.
27	PWRRSTF	R	1	POR/PDR reset flag. Cleared by writing 1 to the RMVF bit.
26	PINRSTF	R	0	PIN reset flag Cleared by writing 1 to the RMVF bit.
25	OBLRSTF	R	0	Option byte loader reset flag. Cleared by writing 1 to the RMVF bit.
24	RMVF	RW	0	The software sets the clear reset flag. The operation of writing this bit to 1 will clear the reset flag starting at bit25. After the software writes 1, the bit will remain at 1, and the hardware will not clear it.
23: 2	Reserved	-	-	-
1	LSIRDY	R	0	LSI oscillator stable flag 0: LSI OSC is not ready; 1: LSI OSC ready;
0	LSION	RW	0	LSI oscillator enable 0: Interrupt is disabled 1: Enabled When the hardware turns on the analog LSI: 1) Hardware IWDG enabled;

6.10.17 Clock configuration register 1 (RCC_CFGR1)

Address offset: 0x78

Reset value: 0x1080_8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USBSELHSI48	PVDSEL [1: 0]		HSI48CAL [12: 0]												
RW	RW	RW	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSI48TRIM [6: 0]							Res.	Res.	Res.	HSI48RDY	HSI48ON	Res.	MCOPRE [2: 0]		
RW							-			R	RW	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	USBSELHSI48	RW	0	USBD module clock selection. 0: USBD clock selection PLL (frequency division); 1: USBD clock selection HSI 48;
30: 29	PVDSEL [1: 0]	RW	2'h0	PVD filtered clock selection. 00: PVD filter clock selection LSI 01: PVD filter clock selection LSE 10: PVD filter clock selection PWR PCLK 11: Reserved; Note: Before configuring PWR_CR.PVDE = 1 to enable the PVD function, you need to configure this register to select the PVD filter clock, and the clock selection will not change during the entire process.
28: 16	HSI48CAL [12: 0]	R	13'h1080	HSI48 clock calibration value. During the loading phase of the power-on process, this register loads the Flash information area value. When the software reads this register, the return value is HSI48CAL+HSI48TRIM. When the HSI48TRIM value changes, the register value will also be updated.
15: 9	HSI48TRIM [6: 0]	RW	7'h40	HSI48 clock Trimming value. The software writes into this register to adjust the HSI48 clock, superimposes it on the HSI48CAL and outputs it to the analog HSI48. This register is reset when the system is reset.
8: 6	Reserved	-	-	-
5	HSI48RDY	R	0	HSI48 clock ready logo. The hardware setting indicates that HSI48 is stable. This bit is valid only if HSI48ON = 1. 0: HSI48 is not ready;

				1: HSI48 ready When HSI48ON is cleared, HSI48RDY will pull down after 6 HSI48.
4	HSI48ON	RW	0	HSI48 clock enabled. 0: HSI48 OFF; 1: HSI48 ON;
3	Reserved	-	-	-
2: 0	MCOPRE [2: 0]	RW	3'h0	MCO (microcontroller clock output) frequency division factor. The software controls these bits and sets the frequency division factor of the MCO output: 000: 1 001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128 These bits are set before the MCO output is enabled.

6.10.18 Clock configuration register 2 (RCC_CFGR2)

Address offset: 0x7C

Reset value: 0x0000_0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	HSE_DRV [1: 0]		Res.	Res.	Res.	Res.	CANCKSEL [3: 0]			
-						RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9: 8	HSE_DRV [1: 0]	RW	2'h0	HSE Drive Capability Configuration 00: 4-8MHz 01: 8-16 MHz 10: 16-32MHz 11: Reserved
7: 4	Reserved	-	-	-
3: 0	CANCKSEL [3: 0]	RW	4'h8	CAN communication clock selection. 0000: PLL clock 0001: PLL clock divided by 2; 0010: PLL clock divided into 3; 0011: PLL clock divided into 4; 0100: PLL clock divided into 5; 0101: PLL clock divided into 6; 0110: PLL clock divided into 7; 0111: PLL clock divided into 8; 1000 HSE clock others: no clock;

6.10.19 Clock configuration register 3 (RCC_CFGR3)

Address offset: 0x80

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL_FBDIV [7: 0]								Res.		PLL_POSTDIV [1: 0]		PLL_PREDIV [1: 0]		PLLXTPRE	PLLSRC
RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 8	PLL_FBDIV [7: 0]	RW	8'h0	Configure the PLL doubling factor. 00000000: x2 00000001: x3 00000010: x4 ….. 11111101: x255 11111110: x256 11111111: x257
7: 6	Reserved	-	-	-
5: 4	PLL_POSTDIV [1: 0]	RW	2'h0	Frequency division ratio setting after phase-locked loop 00: 1 frequency division 01: 2 division 10: 4 division 11: 8 division
3: 2	PLL_PREDIV [1: 0]	RW	2'h0	Frequency division ratio setting before phase-locked loop 00: 1 frequency division 01: 2 division 10: 4 division 11: 8 division
1	PLLXTPRE	RW	0	HSE frequency division serves as PLL source. This bit can only be written when the PLL is closed. 0: HSE clock as PLL source 1: HSE clock divided by 2 as PLL source
0	PLLSRC	RW	0	PLL clock source selection. This bit can only be written when the PLL is closed. 0: HSI16 2-divided as PLL input clock 1: HSE clock configured by PLLXTPRE as PLL input clock

7. Backup register (BKP)

7.1 Introduction

The backup register consists of 32 32-bit registers, which can be used to store 128 bytes of user data. The modules are in the backup domain, and when the VCC power is cut off, they are still maintained by VBAT (Backup Domain Power). They are also not erased when the system is awakened in standby mode, system reset or power reset (POR).

The backup register is reset by a backup domain power-on reset (BPOR) or software reset (BDRST), and when an intrusion event is detected, the contents of the backup register are erased.

The BKP control register is used to manage intrusion detection and RTC calibration functions.

After reset, the backup registers and RTC are not accessible, and the backup domain is protected from possible unexpected write operations. Do the following to enable write access to the backup registers and RTC:

- Turn on the power and back up the domain module clock by setting the PWREN and BKPEN bits of register RCC_APB1ENR1
- The DBP bit of the power supply control register 1 (PWR_CR1) is set to 1 to enable write access to the backup register and the RTC.

7.2 Main features

- Supports 128-byte data backup register
- Status and control registers that manage intrusion detection and have interrupt functions
- Store RTC Check Value

- Output an RTC calibration clock, an RTC alarm pulse or a second pulse on the PC13 pin (when this pin is not used for intrusion detection)
- Backup registers are protected by RDP (Read Protection) of FLASH

7.3 Functional description

7.3.1 Intrusion detection

When the signal on the TAMPER pin changes from 0 to 1 or from 1 to 0 (depending on the TPAL bit of the backup control register BKP_CR), an intrusion detection event occurs. The intrusion detection event clears all data backup register contents. However, in order to avoid loss of intrusion events, the intrusion detection signal is a logical AND of the signal of the edge detection and the intrusion detection permission bit, so that intrusion events that occur before the intrusion detection pin is permitted can also be detected.

- When TPAL = 0: If the pin is already high before starting the intrusion detection TAMPER pin (by setting the TPE bit), once the intrusion detection function is started, an additional intrusion event will be generated (although there is no rising edge after the TPE position '1').
- When TPAL = 1: If the intrusion detection pin TAMPER is already low before the intrusion detection pin is started (by setting the TPE bit), once the intrusion detection function is started, an additional intrusion event will be generated (although there is no falling edge after the TPE position '1').

Set the TPIE bit of the BKP_CSR register to '1' to generate an interrupt when an intrusion event is detected. After an intrusion event is detected and cleared, the intrusion detection pin TAMPER should be disabled. Then, the intrusion detection function is reactivated with the TPE bit before writing to the backup data register again. In this way, the software can be prevented from writing to the backup data register when there is still an intrusion event on the intrusion detection pin. This is equivalent to level detection of the intrusion pin TAMPER.

NOTE: The intrusion detection function is still active when the VCC power is disconnected. To avoid unnecessarily resetting the data backup registers, the TAMPER pin should be connected to the correct level off-chip.

7.3.2 RTC calibration

For the convenience of measurement, the RTC clock can be output to the intrusion detection pin TAMPER by 64 division. This function is turned on by setting the CCO bit of the RTC check register (BKP_RTCCR).

By configuring the CAL [6: 0] bit, the clock can be slowed down by up to 121 ppm.

7.3.2.1 Overview

Real-time clock (RTC) accuracy is a requirement for most embedded applications, but due to changes in external ambient temperature, the crystal frequency of the clock causes the RTC accuracy to be not as accurate as expected. The circuit is calibrated using a digital clock, allowing the application to compensate for crystal and temperature changes.

7.3.2.2 RTC calibration method

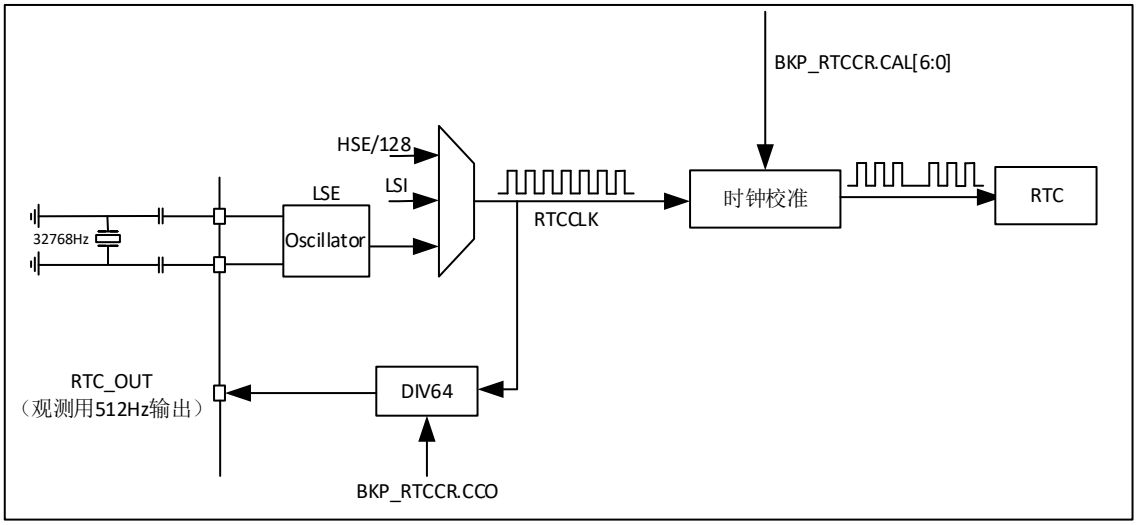
The RTC clock can be driven by a quartz crystal controlled oscillator with a rated frequency of 32.768 kHz. The crystal oscillator is one of the most accurate circuits to provide a fixed frequency. There are two reasons for clock errors:

1. Temperature Change

2. Crystal changes

As mentioned earlier, most clock chips compensate for crystal frequency and temperature changes by using cumbersome trimmer capacitors. The design employs periodic counter correction. The digital calibration circuit eliminates 0 to 127 cycles every 220 clock cycles. The number of pulses blanked depends on the value of the RTC clock calibration register BKP_RTCCR.CAL loaded into the BKP. Since the RTC clock calibration register is in the backup domain, if the battery is connected to the VBAT pin, the calibration value is not lost even if the device is powered off.

The structure block diagram looks like this:



Note: The clock output on the pin is the RTC clock before calibration, so the calibration does not change its value.

The effect of each calibration step is to subtract 1 oscillator cycle for every 1 048 576 (220) actual oscillator cycles. That is, the adjustment amount of each calibration step in the calibration register is 0.954 (1000000/220) ppm. As a result, the oscillator clock can be slowed down from 0 to 121 ppm.

The table below shows the number of ppm and seconds each bit is represented in real time in each month (30 days).

Calibration value	Rounded to near-est ppm	Values are in seconds, rounded to the nearest second per month (30 days)	Calibration value	Rounded to near-est ppm	Values are in seconds, rounded to the nearest second per month (30 days)
0	0	0	64	61	158
1	1	2	65	62	161
2	2	5	66	63	163
3	3	7	67	64	166
4	4	10	68	65	168
5	5	12	69	66	171
6	6	15	70	67	173
7	7	17	71	68	176
8	8	20	72	69	178
9	9	22	73	70	180
10	10	25	74	71	183
11	10	27	75	72	185
12	11	30	76	72	188
13	12	32	77	73	190
14	13	35	78	74	193

15	14	37	79	75	195
16	15	40	80	76	198
17	16	42	81	77	200
18	17	44	82	78	203
19	18	47	83	79	205
20	19	49	84	80	208
21	20	52	85	81	210
22	21	54	86	82	213
23	22	57	87	83	215
24	23	59	88	84	218
25	24	62	89	85	220
26	25	64	90	86	222
27	26	67	91	87	225
28	27	69	92	88	227
29	28	72	93	89	230
30	29	74	94	90	232
31	30	77	95	91	235
32	31	79	96	92	237
33	31	82	97	93	240
34	32	84	98	93	242
35	33	87	99	94	245
36	34	89	100	95	247
37	35	91	101	96	250
38	36	94	102	97	252
39	37	96	103	98	255
40	38	99	104	99	257
41	39	101	105	100	260
42	40	104	106	101	262
43	41	106	107	102	264
44	42	109	108	103	267
45	43	111	109	104	269
46	44	114	110	105	272
47	45	116	111	106	274
48	46	119	112	107	277
49	47	121	113	108	279
50	48	124	114	109	282
51	49	126	115	110	284
52	50	129	116	111	287
53	51	131	117	112	289
54	51	133	118	113	292
55	52	136	119	113	294
56	53	138	120	114	297
57	54	141	121	115	299
58	55	143	122	116	302
59	56	146	123	117	304
60	57	148	124	118	307
61	58	151	125	119	309
62	59	153	126	120	311
63	60	156	127	121	314

As mentioned above, the RTC clock calibration circuit only subtracts cycles from the crystal clock. Based on the RTC prescaler value set to 32768 by default, faster crystal frequencies (> 32.768 kHz) can be calibrated, while slower crystal frequencies (< 32.768 kHz) cannot be compensated (only the faster crystals are calibrated, the slower ones will exacerbate the slower situation). So only the crystal frequency range [32 772, 32 768] can be calibrated.

Since the crystal frequency may vary around 32.768 kHz, the crystal frequency may be adjusted from 32.768 kHz to 32.766 kHz, for example, by setting the RTC prescalation to 32766 (instead of 32768). In this way, the range of crystal frequencies of [32770, 32766] can be compensated for.

7.3.2.3 Calculate the amount of calibration required

In order to determine how much calibration is needed in a given application, it is necessary to use the output function of the RTC clock on a case-by-case basis. The RTC clock output can be used to

measure the accuracy of the crystal oscillator. If the frequency of the crystal oscillator is an exact 32.768 kHz, the frequency of the RTC clock output is exactly 512 Hz.

The method can be divided into the following steps:

1. Enable the Low Speed External Oscillator (LSE), select LSE as the RTC clock source, and then enable the RTC clock.

2. Output the64-point frequency of the RTC clock on the RTC_OUT pin (PC13)for crystal frequency measurement. This is achieved by placing CCO position 1in BKP_RTCCR.

3. Calculate crystal frequency deviation (ppm). Assuming the RTC prescaler value of 32766, the deviation in ppm can be quickly calculated by dividing the measured deviation of 511.968 Hz (32766/64 ≈ 511.968) by 511.968 and multiplying the result by 1 million. This table is a direct look-up table of calibration values based on variation values expressed in ppm.

4. Write calibration values to the RTC calibration register to compensate for crystal deviation.

Note: To set the RTC prescaler to 32766, write 32765 to the RTC prescaler register.

For example, if the frequency measured in the actual measurement is 511.982 Hz, δ is 0.014. Divided by 511.968 and multiplied by 1 million, the result is 27.35 ppm. In this case, the closest compensation value is 28. The error will be reduced from 27.35 ppm (71 seconds per month) to 0.65 ppm (1.7 seconds per month).

Note: Since the principle of RTC calibration is to ignore part of the RTC clock pulse, it does not improve counting in short time, it only improves counting in long time. For example, using an RTC count of 0.01 seconds without calibration would be more accurate than with calibration. Since the calibration cycle may or may not occur within the time frame considered, the resulting values may vary significantly. Therefore, depending on the application, it is preferable not to use calibration.

7.3.3 Erasure of backup registers

The backup register may be erased in the following ways:

① VBAT domain power-on reset

② The intrusion detection circuit successfully captures the intrusion event

③ BDRST (backup domain soft reset) of register RCC_BDCR is written to 1

④ RDP level Changes from 1 to 0

⑤ TEF flag bit is 1

Note: When erasing is being done, the bus will be in an unanswering state, and the bus cannot perform any read and write operations on the BKP.

7.4 TIMx registers

Note: All registers of the BKP module can only be accessed in word mode

7.4.1 Backup Control Register (BKP_CR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.														TPAL	TPE
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	TPAL	RW	0	Intrusion detection TAMPER pin active level. 0: A high level on the intrusion detection TAMPER pin will clear all data backup registers (when TPE = 1); 1: A low level on the intrusion detection TAMPER pin will clear all data backup registers (when TPE = 1); Note: This bit cannot be changed when TPE = 1.
0	TPE	RW	0	Start intrusion detection TAMPER pin. 0: The intrusion detection TAMPER pin is used as a general-purpose IO port; 1: Intrusion detection TAMPER pin is used as intrusion detection;

7.4.2 Backup Control/Status Register (BKP_CSR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.						TIF	TEF	Res.					TPIE	CTI	CTE
-						R	R	-					RW	W	W

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	TIF	R	0	Trespass interrupt flag. When an intrusion event is detected and the TPIE bit is 1, this bit is set to 1 by the hardware. This flag bit is cleared (and the interrupt is also cleared) by writing 1 to the CTI bit. If the TPIE bit is cleared, this bit is also cleared. This flag can be cleared by system reset, but not by backup domain reset. This flag clears when the microcontroller wakes up from standby mode 0: no intrusive interruption; 1: Generate intrusion interrupt;
8	TEF	R	0	Trespass event flag. This bit is set to 1 by the hardware when an intrusion event is detected. This flag bit can be cleared by writing 1 to the CTE bit. 0: No invasive event; 1: Intrusion event detected; Note: The intrusion event resets all BKP_DRx registers. When this bit is set to 1, if a write operation is performed to BKP_DRx, the written value will not be saved.
7: 3	Reserved	-	-	-
2	TPIE	RW	0	Allow intrusion into the TAMPER pin interrupt. 0: Disable intrusion detection interrupt; 1: Allow intrusion detection interrupt (TPE bit of BKP_CR register must also be set to 1) This flag can be cleared by system reset, but not by backup domain reset. When the microcontroller wakes up from standby mode, this flag is cleared
1	CTI	W	0	Clear intrusion detection interrupt. This bit can only write 1 and read 0. 0: invalid; 1: Clear intrusion detection interrupt and TIF intrusion detection interrupt flags.
0	CTE	W	0	Clear intrusion detection events. This bit can only write 1 and read 0. 0: invalid; 1: Clear the TEF intrusion detection event flag (and reset the intrusion detector).

7.4.3 RTC clock calibration register (BKP_RTCCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.						ASOS	ASOE	CCO	CAL [6: 0]						
-						RW	RW	RW	RW						

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	ASOS	RW	0	Alarm clock or second output selection. When the ASOE bit is set, the ASOS bit can be used to select whether the RTC second pulse or the alarm pulse is output on the TAMPER pin. 0: Output RTC alarm pulse; 1: Output second pulse Note: This bit can only be cleared by backup domain reset.
8	ASOE	RW	0	Allows the output of alarm clock or second pulse. Depending on the ASOS bit configuration, this bit allows an RTC second pulse or an alarm pulse to be output to the TAMPER pin. The width of the output pulse is one cycle of the RTC clock. When the ASOE bit is set, the function of TAMPER cannot be turned on. Note: This bit can only be cleared by backup domain reset.
7	CCO	RW	0	Calibrate the clock output. 0: No effect; 1: Output the RTC 64-divided clock at the intrusion detection pin. After CCO position 1, the intrusion detection function must be turned off to avoid detecting invalid intrusion events. Note: This bit is cleared when the VCC power is disconnected.
6: 0	CAL [6: 0]	RW	6'h0	Calibration values. The calibration value indicates how many clock pulses will be skipped every 220 clock pulses. This can be used to calibrate the RTC to slow down the clock at a ratio of 1000000/220 ppm. The RTC clock can be slowed down by 0 ~ 121 ppm.

Note: ① When CCO and ASOE are 1 at the same time, ASOS priority is high.

② The data of this register needs to be synchronized from APB to RTC clock domain to ensure that the written value is confirmed to be correct and can be written again.

7.4.4 Backup Data Register (BKP_DRx) (x = 0 … 31)

Address offset: 0x80 + 0x04 * x, (x = 0 to 31) (0x80 ~ 0xFC)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D [31: 16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 0	D [31: 0]	RW	32'h0	Backup data. These bits may be used to hold user data. Note: The BKP_DRx register will not be reset by system reset, power reset, wake-up reset from standby mode. Can be reset by a backup domain reset or (if the intrusion detection pin TAMPER function is turned on) by an intrusion pin event.

8. CRC Computing Unit (CRC)

8.1 Introduction

The cyclic redundancy check (CRC) unit may calculate the CRC code from the 8-bit, 16-bit, 32-bit generator polynomial using a 7-bit, 8-bit, 16-bit, 32-bit custom generator polynomial.

In other applications, CRC-based techniques are used to verify the integrity of data transmission or storage. Within the field of functional safety standards, CRC-based technologies provide a way to verify the integrity of flash memory. The CRC computing unit may compute the identity of the software at the time of the program running, then compare it with the reference identity generated at the time of connection, and then store it in a designated memory space.

8.2 CRC main features

- Using CRC-32 Ethernet Polynomial: 0x4C11DB7

X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 + X8 + X7 + X5 + X4 + X2 + X+1

- Supports programmable polynomials, and the polynomial bit width is also programmable (supports 7, 8, 16, 32-bit polynomials)
- Supports calculation of 8, 16, 32 bits of data
- Customizable CRC initial value
- General purpose 32-bit register (can be used to store temporary data)
- With input cache function
- One 32-bit data register for input/output
- CRC calculation time: 4 AHB clock cycles (HCLK) required for 32-bit data; For 16-bit data, 2 AHB clock cycles are required; 1 AHB clock cycle (HCLK) required for 8-bit data
- Support input data inversion, output data inversion
- The CRC_DR register can be accessed by the AHB bus at 8 bits right aligned, 16 bits right aligned, 32 bits, and other registers can only be accessed by the AHB bus at 32 bits

8.3 CRC functional description

8.3.1 CRC Structure Block Diagram

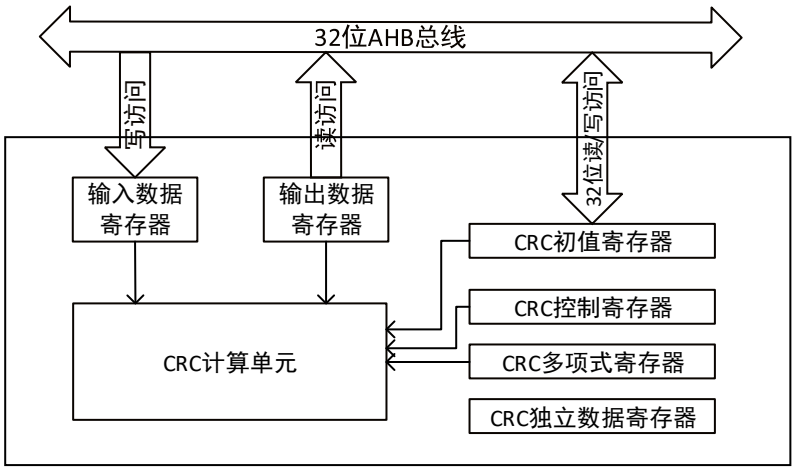


Figure 8-1 Structure of CRC module

8.3.2 CRC Operation

The CRC computing unit contains one 32-bit data register (CRC_DR). For write operations, it is used to save input data; For read operations, it is used to save the previous CRC calculation results. Each time the data register is written, the calculation result is a combination of the previous CRC calculation result and the new calculation result (CRC calculation is performed on the entire 32-bit word, rather than byte by byte), depending on the format of the written data. The CRC_DR register can be accessed by words (32 bits), right-aligned half-words (16 bits), right-aligned bytes (8 bits), while other CRC registers only allow 32 bits.

For data with different bit widths, the CRC calculation module needs different time, as follows:

- Computing 32-bit data requires 4 AHB clock cycles
- Computing 16-bit data requires 2 AHB clock cycles
- It takes 1 AHB clock cycle to calculate 8-bit data

The input cache allows the user to write the second data immediately after the first data is written, without waiting for the previous data to be calculated.

Users can dynamically adjust the data size to reduce the number of accesses to this module when calculating a specific number of bytes of data. For example, to calculate a 5-byte data, one word of data can be written to this module, and then one byte of data can be written to obtain the calculation result.

This module supports input data inversion function to adapt to different size and size ends. According to REV_IN [1: 0] of the CRC_DR register, the input data inversion can be inverted in units of 8 bits, 16 bits, and 32 bits.

For example, for input data 0x1A2B3C4D that requires a CRC operation:

- The data after inversion in bytes (8 bits) is 0x58D43CB2
- The data after inversion in units of half words (16 bits) is 0xD458B23C
- The data after inversion in words (32 bits) is 0xB23CD458

The REV_OUT bit in the CRC_CR register can invert the output data. This operation is done at the bit level, for example: the output data is 0x112233544, which is inverted to become 0x22CC4488

Placing the RESET position 1 of the CRC_CR register enables the initial value of the CRC calculator to be loaded with a programmable value (default value is 0xFFFFFFF)

The CRC initial value can be programmed to the value contained in the CRC_INIT register. When the CRC_INIT register is written, the CRC_DR register is automatically initialized.

The CRC_IDR register may be used to hold temporary values related to the CRC calculation. It is not affected by the RESET bit in the CRC_CR register.

8.3.3 Polynomial programmability

The coefficients of the polynomial can be fully customized by the CRC_POL register, and the bit width of the polynomial can be configured as 7 bits, 8 bits, 16 bits, 32 bits according to POLYSIZE [1: 0] of the CRC_CR register. Even polynomials are not supported, for example: even polynomials are of the form x+x2 + … + xn, while odd polynomials are of the form 1 + x+x2 + … + xn.

If the CRC data is less than 32 bits, the value of CRC will be read from the least significant bit of CRC_DR.

In order to obtain a reliable CRC calculation result, the value or size of the polynomial cannot be changed during the calculation of the CRC. The CRC must be reset before the CRC polynomial can be changed.

The default polynomial is the CRC-32 (Ethernet) polynomial: 0x4C11DB7.

8.4 TIMx registers

8.4.1 CRC data register (CRC_DR)

Address offset: 0x00

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DR [31: 0]	RW	32'hFFFFFFFF	Data register bits. When new data of the CRC counter is written, it is used as an input register. Returns the result of the previous CRC calculation when read. If the CRC data is less than 32 bits, the lower bits of this register are used to read or write the correct data.

8.4.2 Independent Data Register (CRC_IDR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	IDR [31: 0]	RW	32'h0	General purpose 32-bit data register bits. Data can be stored temporarily. This register is not affected by CRC resets generated by the RESET bit in the CRC_CR register. Note: This register does not participate in CRC calculation and can store any data.

8.4.3 CRC control register (CRC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								REV_OUT	REV_IN [1: 0]		POLYSIZE [1: 0]		Res.		RESET
-								RW	RW		RW		-		w

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	REV_OUT	RW	0	Invert output data This bit determines whether the bit order of the output data is inverted 0: The bit order of the output data is not inverted 1: Bit order inversion of output data
6: 5	REV_IN [1: 0]	RW	2'b0	Reverse input data These 2 bits determine whether the order of the input data is reversed 00: The order is not reversed 01: The input data is reversed every byte 10: The input data is reversed every half-word 11: The input data is reversed every word
4: 3	POLYSIZE [1: 0]	RW	2'b0	CRC polynomial bit width These 2 bits determine the bit width of the CRC polynomial 00: 32-bit CRC polynomial 01: 16-bit CRC polynomial 10: 8-bit CRC polynomial 11: 7-bit CRC polynomial
2: 1	Reserved	-	-	-
0	RESET	w	0	When the software is set, the CRC module will be reset, and the data register will be loaded with the value of the CRC_INIT register. The software can only write 1, which is cleared by the hardware.

8.4.4 CRC Initial Register (CRC_INIT)

Address offset: 0x10

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRC_INIT [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRC_INIT [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	CRC_INIT [31: 0]	RW	32'hFFFFFFFF	Programmable CRC initial value This register is used to store the initial CRC value

8.4.5 CRC Polynomial (CRC_POL)

Address offset: 0x14

Reset value: 0x04C1 1DB7

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
POL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	POL [31: 0]	RW	32'h04C11DB7	Programmable CRC polynomial This register is used to store the coefficients of the CRC polynomial If the CRC polynomial is less than 32 bits, it will be stored right-aligned in this register

9. General IO (GPIO)

9.1 Introduction

Each general purpose IO port includes four 32-bit configuration registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, and GPIOx_PUPDR), two 32-bit data registers (GPIOx_IDR and GPIOx_ODR), one 32-bit bit bit/reset register (GPIOx_BSRR),one 32-bit reset register (GPIOx_BRR), one 32-bit lock register(GPIOx_LCKR), and two 32-bit multiplex function select registers (GPIOx_AFRH and GPIOx_AFRL).

9.2 Main Features

- Up to 86 controlled IOs
- Output status: push-pull or open-drain + pull-up/pull-down
- Output from output data register (GPIOx_ODR) or peripheral (multiplex function output) to pin
- Different speeds can be selected for each IO
- Input states: floating, pull-up/down, analog
- Input pins into data registers (GPIOx_IDR) or peripherals (multiplexing function input)
- Set/reset register (GPIOx_BSRR) with bitwise write privileges to GPIOx_ODR
- Reset register (GPIOx_BRR) with bitwise write privileges to GPIOx_ODR
- Lock register (GPIOx_LCKR) to freeze IO configuration
- Multiplexing function input/output select register (one IO can have up to 16 multiplexing functions)
- Fast flip, with as fast as two clock cycles per flip
- Pin multiplexing is very flexible, allowing IO pins to be used as GPIO or one of a variety of peripheral functions

9.3 Function description

Depending on the characteristics of each IO port listed in the datasheet, each port bit of a General Purpose IO (GPIO) port can be configured by software to multiple modes:

- Input floating
- Input pull-up
- Input-pull-down
- Analog function
- Open drain output with pull-up or pull-down function
- Push-pull output with pull-up or pull-down function
- Multiplexing function push-pull with pull-up or pull-down function
- Multiplexing function open drain with pull-up or pull-down function

Each IO port bit is freely programmable, but the IO port registers must be accessed by word, half-word, or byte. The GPIOx_BSRR and GPIOx_BRR registers allow independent access to reads/changes to any GPIO register. In this way, there is no risk of an IRQ occurring between the read and the modify access.

The following diagram shows the basic structure of 5 V compatible and basic IO port bits. Table 9-1 shows possible port bit configuration schemes.

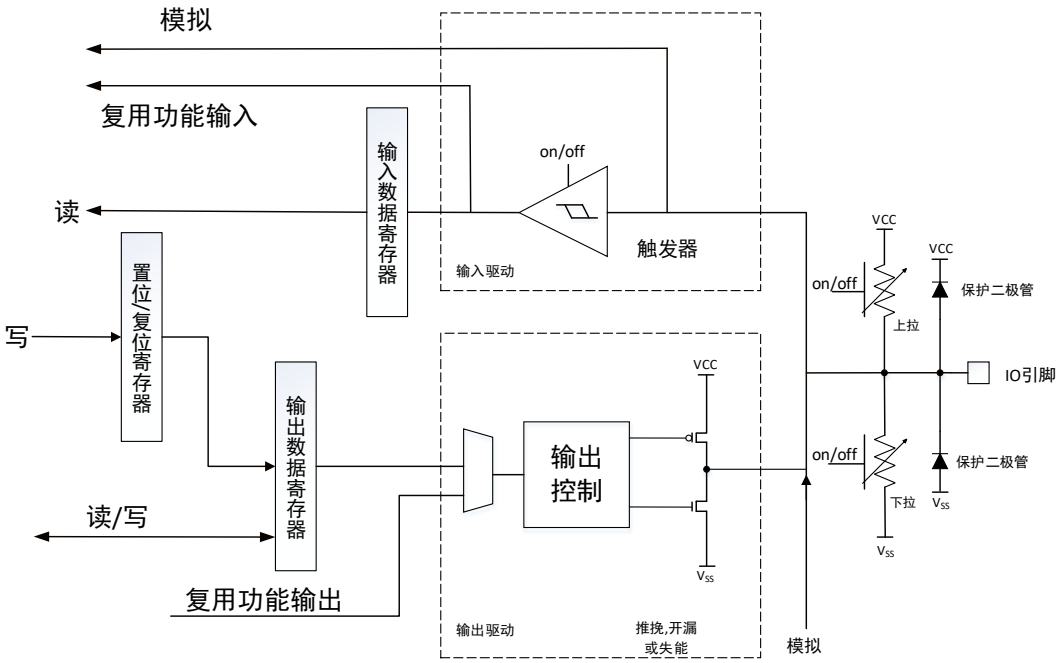


Figure 9-1 IO port basic structure

Table 9-1 Port bit configuration table

MODE (i) [1: 0]	OTYPE	OSPEED (i) [1: 0]		PUPD (i) [1: 0]		IO configuration	
01	0	SPEED [1: 0]		0	0	GP output	PP
	0			0	1	GP output	PP+PU
	0			1	0	GP output	PP+PD
	0			1	1	Reserved	
	1			0	0	GP output	OD
	1			0	1	GP output	OD+PU
	1			1	0	GP output	OD+PD
	1			1	1	Reserved (GP output OD)	
10	0	SPEED [1: 0]		0	0	AF	PP
	0			0	1	AF	PP+PU
	0			1	0	AF	PP+PD
	0			1	1	Reserved	
	1			0	0	AF	OD
	1			0	1	AF	OD+PU
	1			1	0	AF	OD+PD
	1			1	1	Reserved	
00	x	x	x	0	0	Input	Floating
	x	x	x	0	1	Input	PU
	x	x	x	1	0	Input	PD
	x	x	x	1	1	Reserved (input floating)	
11	x	x	x	0	0	Input/Out	Analog
	x	x	x	0	1	Reserved	
	x	x	x	1	0	Input/Out	Analog, PD
	x	x	x	1	1	Reserved	

GP = general-purpose, PP = push-pull, PU = pull-up, PD = pull-down, OD = open-drain, AF = alternate function.

9.3.1 General IO (GPIO)

During and immediately after the reset, the multiplexing function has not been activated, and most IO ports are configured in analog mode. After reset, the debug pin is in the multiplexing function pull-up/pull-down state and the BOOT0 pin is in the input mode pull-down state:

- PA15: JTDI in pull-up state
- PA14: JTCK/SWCLK in pull-down state
- PA13: JTMS/SWDAT in pull-up state

- PB4: NJTRST in pull-up state
- PB8: BOOT0 is in pull-down state

The input data register (GPIOx_IDR) captures data from the IO pin every 1 AHB clock cycle.

All GPIO pins have internal weak pull-up and pull-down resistors that can be turned on/off based on the value in the GPIOx_PUPDR register.

9.3.2 IO pin multiplexer and mapping

The IO pin is connected to the onboard peripheral/module via a multiplexer that allows the multiplexing function (AF) of only one peripheral to be connected to the IO pin at a time. This ensures that there are no collisions between peripherals sharing the same IO pin.

Each IO pin has a multiplexer with 16 multiplexing function inputs (AF0 to AF15) that can be configured through the GPIOx_AFRL (for pins 0 to 7) and GPIOx_AFRH (for pins 8 to 15) registers.

To configure IO to the desired functionality, follow these steps:

Debug assignment

After the system is reset, the debugging-related pins are multiplexed as AF for use by the system debugging interface.

GPIO

Configure the required IO as output or input in the GPIOx_MODER register.

Peripheral multiplexing function

In the GPIOx_AFRL or GPIOx_AFRH register, the multiplexing function of IO is set

-Select output type, pull-up/pull-down, and output speed, respectively, via GPIOx_OTYPER, GPIOx_PUPDR, and GPIOx _OSPEEDR registers

-Configure the required IO as multiplexing function in the GPIOx_MODER register

Additional functions:

-For ADC, DAC, OPA and COMP, it is necessary to configure the corresponding IO into the analog mode through the register GPIOx_MODER, andconfigure the corresponding functions in ADC, DAC, OPA and COMP; Special attention should be paid to the output of additional functions (such as DAC or OPA), the corresponding IO needs to be configured as analog mode first, and thenthe control registers of peripheralsshould be enabled.

-For additional functions TAMP_RTC, WKUPx,PVD_IN, OSCandOSC32, corresponding functionsneed to be configuredin modules such as RTC, PWR and RCC,which take precedence over-GPIO settings.

9.3.3 IO port control register

Each GPIO has four 32-bit control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) and can be configured with up to 16 IOs. The GPIOx_MODER register is used to select IO operating modes(input, output,multiplex, analog). GPIOx_OTYPER is used to select the output type (push-pull or open drain). The GPIOx_OSPEEDR register is used toset the IO speed. The GPIOx_PUPDR register is used to select pull-up/pull-down.

9.3.4 IO port data register

Each GPIO has two 16-bit data registers: input and output data registers (GPIOx_IDR and GPIOx_ODR). GPIOx_ODR stores the data to be output, it is read/write accessible. IO input data is stored in GPIOx_IDR, which is a read-only register.

9.3.5 IO data bit operation

The Set Reset Register (GPIOx_BSRR) is a 32-bit register that allows applications to set and reset individual data bits in the output data register (GPIOx_ODR).

Each data bit in GPIOx_ODR corresponds to two control bits in GPIOx_BSRR: BS (i) and BR (i). When 1 is written, the BS (i) bit sets the corresponding ODR (i) bit. When 1 is written, the BR (i) bit clears the corresponding bit of ODR (i).

Writing any bit to 0 in GPIOx_BSRR does not have any effect on the corresponding bit in GPIOx_ODR. If both set and clear operations are attempted on a bit in GPIOx_BSRR, the set operation takes precedence.

Using the GPIOx_BSRR register to change the values of individual bits in GPIOx_ODR is a “one-shot” effect that does not lock the GPIOx_ODR bits. The GPIOx_ODR bits can always be accessed directly. The GPIOx_BSRR register provides a way of performing atomic bitwise handling.

There is no need for the software to disable interrupts when programming the GPIOx_ODR at bit level: it is possible to modify one or more bits in a single atomic AHB write access.

The reset register (GPIOx_BRR) is a 32-bit register that allows an application to perform a reset operation on individual data bits in the output data register (GPIOx_ODR). The reset register functions similarly to the set reset register, but can only reset one or more bits of the output data register.

9.3.6 GPIO locking mechanism

It is possible to freeze the GPIO control registers by applying a specific write sequence to the GPIOx_LCKR register. The frozen registers include GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL, and GPIOx_AFRH.

To perform a write operation to the GPIOx_LCKR register, a specific write/read sequence must be used. Afterthe softwareoperatesthe LCKR [16] of this register with the correct locking sequence, the value ofLCKR [15: 0] will be used to lock the configuration of IO (the value of LCKR [15: 0] mustremain the sameduring the write sequence). After one or more ports are locked, they cannot be unlocked until a system reset occurs, or GPIO is reset. Each GPIOx_LCKR bit freezes the corresponding bit in the control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL and GPIOx_AFRH.

The lock sequencecan only operate on the GPIOx_LCKR register via word access.

Refer to the GPIOx_LCKR register for specific operations.

9.3.7 IO multiplexing function input/output

There are two registers that can be used to select from the 16 multiplexing function inputs/outputs available for each IO.

Different multiplexed signals have different input/output directions, which are determined by their respective peripheral modules.

9.3.8 External interrupt line/wake-up line

All ports have external interrupt capability. To use an external interrupt line, the port must be configured in input mode, see the Interrupts and Events section.

9.3.9 Input configuration

When the IO port is configured as input:

- The output buffer is disabled
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- Each AHB clock sample pin level and is stored in the input data register
- Read the input data register to know the pin level

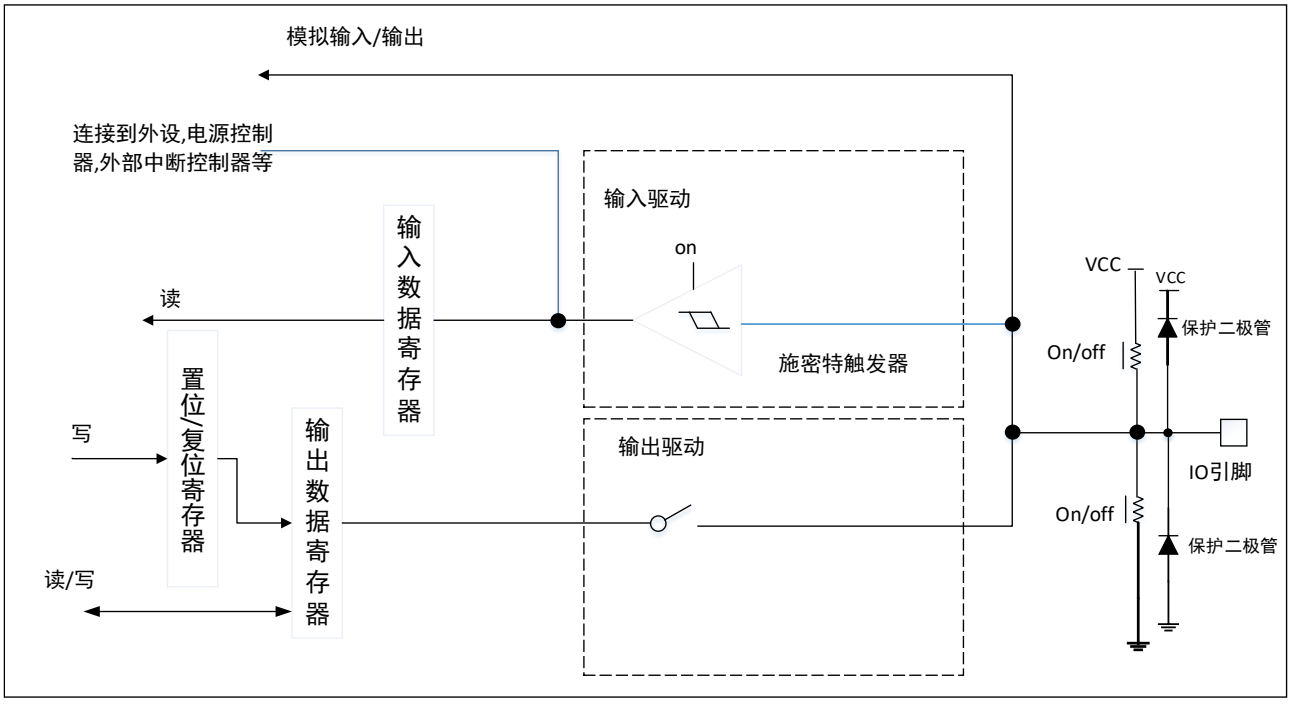


Figure 9-2 Input floating/pull-up/pull-down configuration

9.3.10 Output configuration

When an IO port is configured as an output:

- Output buffer enable

-Open-drain mode: '0' on the output register turns on N-MOS, while '1' on the output register puts the port in a high impedance state (PMOS is not conducting).

─ Push-pull mode: '0' on the output register turns on N-MOS, while '1' on the output register turns on P-MOS.

- Schmidt Trigger Input Enable
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- Each AHB clock sample pin level and is stored in the input data register
- Read the input data register to know the pin level
- Read the output data register to know the value of the last GPIO write

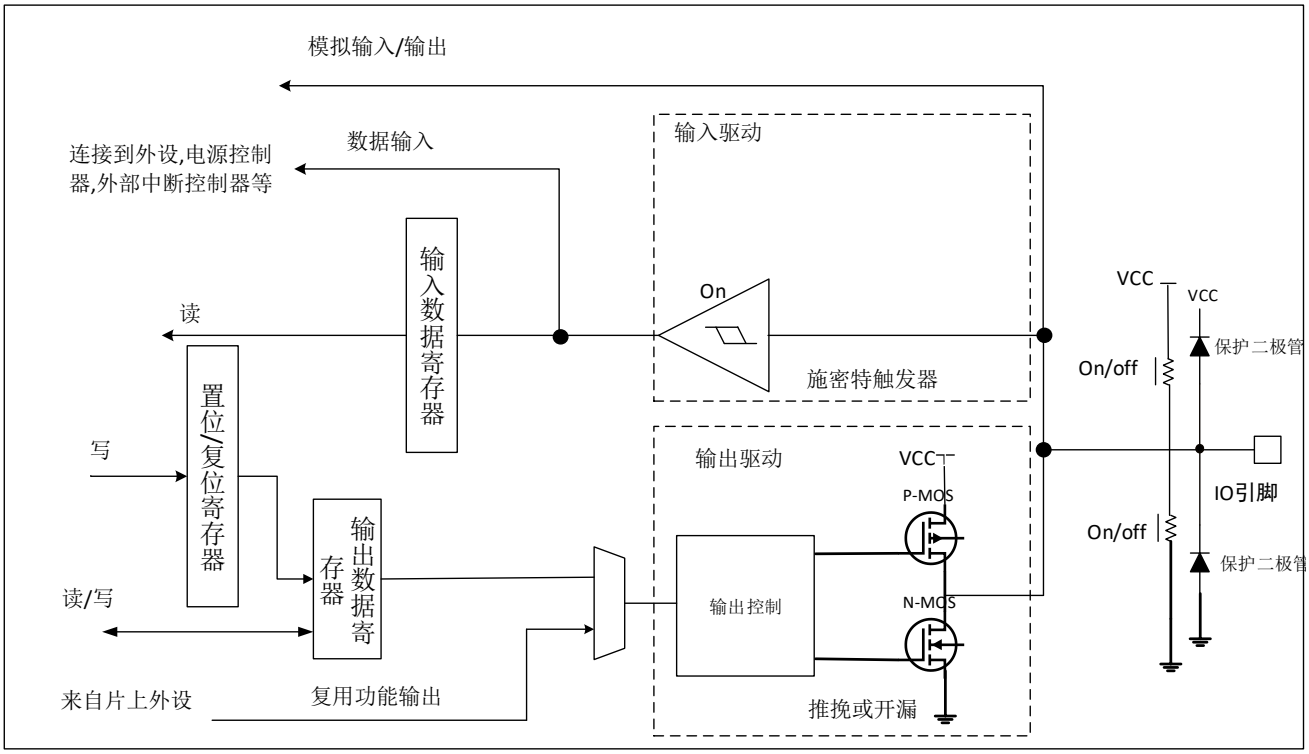


Figure 9-3 Output configuration

9.3.11 Alternate function configuration

When an IO port is configured for multiplexing function:

- Output buffer enable
- The output data and output enable of the output buffer are driven by the peripheral module
- Schmidt Trigger Input Enable
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- Each AHB clock sample pin level and is stored in the input data register
- Read the output data register to know the value of the last GPIO write

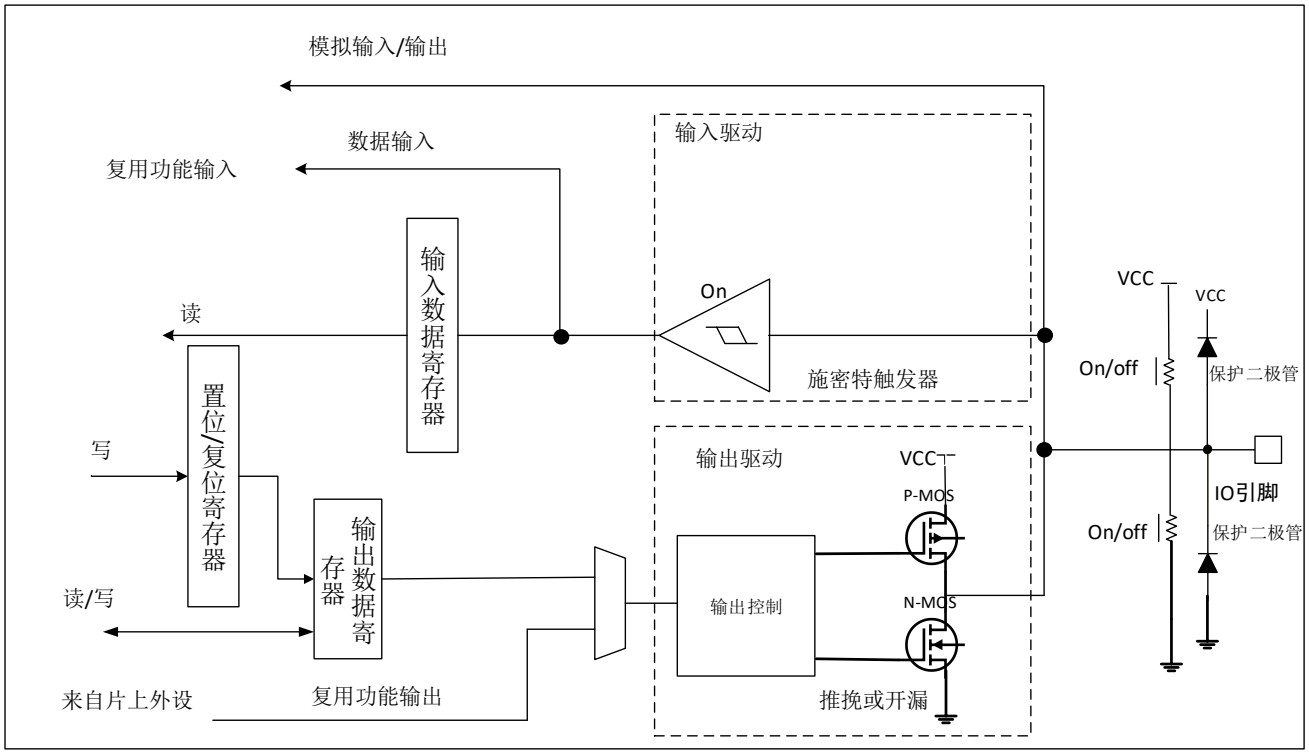


Figure 9-4 Alternate function configuration

9.3.12 Analog configuration

When the IO port is configured in analog mode:

- The output buffer is not enabled;
- The Schmidt trigger input is not enabled, achieving zero power consumption on each analog IO pin. The output of the Schmitt trigger is forced to a constant value (0);
- Weak pull-up and pull-down resistors are disabled (software setting is required);
- Read access to the input data register gets the value “0”.

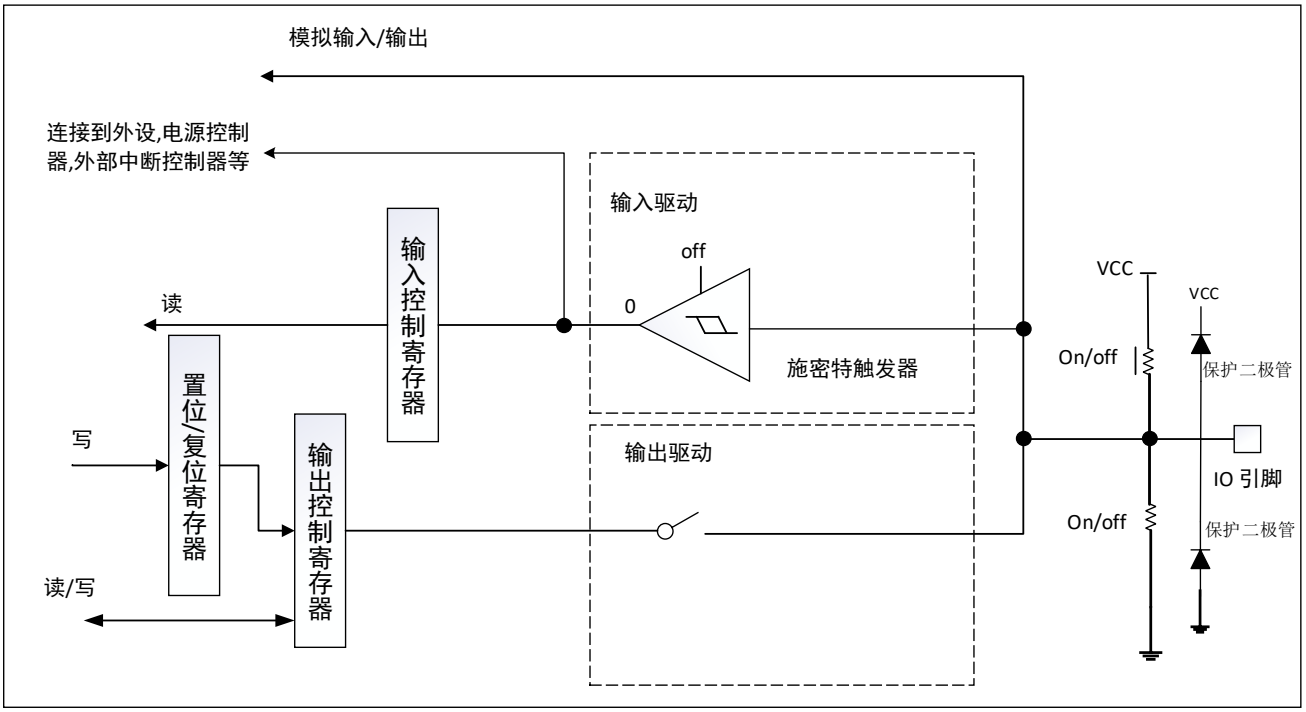


Figure 9-5 High impedance-analog configuration

9.3.13 Use HSE or LSE oscillator pins as GPIO

When the HSE or LSE function is not enabled (the default after reset), the corresponding pin can be regarded as a normal GPIO.

When the HSE or LSE function is enabled (HSEON or LSEON is set in the RCC register), the functions of the relevant pins are controlled by the oscillator and are not affected by the settings of these GPIO registers.

When the crystal oscillator is configured in the user external clock mode, only OSC_IN or OSC32_IN is used to receive the external clock input, while the OSC_OUT or OSC32_OUT pin can still be used as the normal GPIO.

9.3.14 Use the GPIO pins in the RTC power domain

The PC13/PC14/PC15 GPIO function fails when the VDDD is powered down (when the device enters standby mode). If the RTC configuration does not override its GPIO settings, these pins are in an analog input/output state after standby wake-up.

9.3.15 Using PB8 as GPIO

PB8 can be used as a boot pin (BOOT0) or as a GPIO. Select based on the nSWBOOT0 bit in the user options byte:

- nSWBOOT0 = 1, BOOT0 pin in reset.
- nSWBOOT0 = 0, GPIO pin in reset.

Precautions:

PB8 is a GPIO pin regardless of nSWBOOT0 after reset.

9.3.16 Using PF5 as GPIO

PF5 can be used as a reset pin (NRST) or as a GPIO. According to the NRST_MODE bit in the user options byte, it switches to the following mode:

- Reset input/output: NRST_MODE =2 'b11(default)or NRST_MODE =2'b00

- Reset input only: NRST_MODE =2 'b01
- GPIO PF5 mode: NRST_MODE =2'b10

9.4 TIMx registers

This section describes the register functionality of GPIO in detail

For a summary of register bits, register offset addresses, and reset values, see the GPIO register mapping table;

GPIO registers can be accessed by byte (8-bit), half-word (16-bit), or word (32-bit).

GPIOA base address: 0x4800_0000

GPIOB base address: 0x4800_0400

GPIOC base address: 0x4800_0800

GPIOD base address: 0x4800_0C00

GPIOE base address: 0x4800_1000

GPIOF base address: 0x4800_1400

9.4.1 GPIO Port Mode Register (GPIOx_MODER) (x = A.. F)

Address offset: 0x00

Reset value: 0xABFF_FFFF (port A)

Reset value: 0xFFFF_FEBF (port B)

Reset Value: 0xFFFF FFF (Other C.. F)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15 [1: 0]		MODER14 [1: 0]		MODER13 [1: 0]		MODER12 [1: 0]		MODER11 [1: 0]		MODER10 [1: 0]		MODER9 [1: 0]		MODER8 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MODER7 [1: 0]		MODER6 [1: 0]		MODER5 [1: 0]		MODER4 [1: 0]		MODER3 [1: 0]		MODER2 [1: 0]		MODER1 [1: 0]		MODER0 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	MODEy [1: 0]	RW	PA13 to PA15, PB3 to PB4: 2'b10 Others: 2 'b11	y = 15..0 The software uses these bits to configure the corresponding IO mode 00: Input mode 01: General purpose output mode 10: Alternate function mode 11 analog mode (reset state)

9.4.2 GPIO port output type register (GPIOx_OTYPER) (x = A.. F)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	OTy	RW	0	y = 15..0 Software Configuration IO Output Type 0: push-pull output (reset state) 1: Output open-drain

9.4.3 GPIO port output speed register (GPIOx_OSPEEDR) (x = A.. F)

Address offset: 0x08

Reset Value: 0x0C00 0000 (Port A)

Reset value: 0x0000 0000 (other)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15 [1: 0]		OSPEED14 [1: 0]		OSPEED13 [1: 0]		OSPEED12 [1: 0]		OSPEED11 [1: 0]		OSPEED10 [1: 0]		OSPEED9 [1: 0]		OSPEED8 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSPEED7 [1: 0]		OSPEED6 [1: 0]		OSPEED5 [1: 0]		OSPEED4 [1: 0]		OSPEED3 [1: 0]		OSPEED2 [1: 0]		OSPEED1 [1: 0]		OSPEED0 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	OSPEEDy [1: 0]	RW	PA13: 2 'b11 Others: 2 'b00	y = 15..0 These bits are written by software to configure the I/O output speed. 00: Very low 01: Low speed 10: high speed 11: Very high speed

9.4.4 GPIO port pull-up/pull-down register (GPIOx_PUPDR) (x = A.. F)

Address offset: 0x0C

Reset value: 0x6400 0000 (port A)

Reset value: 0x0002 0100 (port B)

Reset value: 0x0000 0000 (other)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPD15 [1: 0]		PUPD14 [1: 0]		PUPD13 [1: 0]		PUPD12 [1: 0]		PUPD11 [1: 0]		PUPD10 [1: 0]		PUPD9 [1: 0]		PUPD8 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PUPD7 [1: 0]		PUPD6 [1: 0]		PUPD5 [1: 0]		PUPD4 [1: 0]		PUPD3 [1: 0]		PUPD2 [1: 0]		PUPD1 [1: 0]		PUPD0 [1: 0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	PUPDy [1: 0]	RW	2 'b01: PA13, PA15, PB4 2 'b10: PA14, PB8 2 'b00: others	y = 15..0 Software configuration IO port pull-up or pull-down 00: No pull-up, pull-down 01: pull-up 10: pull down 11: Reserved

9.4.5 GPIO port input data register (GPIOx_IDR) (x = A.. F)

Address offset: 0x10

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
R															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	IDy	R		y = 15..0 This is read-only, and the read value bit corresponds to the status of the IO port

9.4.6 GPIO port output data register (GPIOx_ODR) (x = A.. F)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ODy	RW	0	y = 15..0 For the GPIOx_BSRR or GPIOx_BRR registers(x = A, B, C, D, E, F), each ODR bit can be set/cleared independently.

9.4.7 GPIO port set/reset register (GPIOx_BSRR) (x = A.. F)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
W															

Bit	Name	R/W	Reset Value	Function
31: 16	BRy	W	0	y = 15..0 Read return value is 0 0: The corresponding ODRy bit is not affected 1: Clear the corresponding ODRy bit Note: If the corresponding bits of BSy and BRy are written at the same time, the BSy bit works
15: 0	BSy	W	0	y = 15..0 Read return value is 0 0: The corresponding ODRy bit is not affected 1: Set the corresponding ODRy bit

9.4.8 GPIO Port Configuration Lock Register (GPIOx_LCKR) (x = A.. F)

This register is used to lock the configuration of the port bits when the correct write sequence set LCKK is performed. LCKR [15: 0] is used to lock the configuration of GPIO ports. The LCKR [15: 0] cannot be changed during a specified write operation. After performing the locking sequence on the corresponding port, the configuration of the port bits can no longer be changed until the next system reset. Note: A specific write sequence is used to write to the GPIOx_LCKR register. Only word access is allowed in lock timing.

Each lock bit freezes a specific configuration register (control and alternate function registers).

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK
-															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	LCKK	RW	0	This bit can be read any time. It can only be modified using the lock key write sequence. 0: Port configuration lock key not active 1: The port configuration lock key is activated, and the GPIOx_LCKR register is locked before the next system reset Lock key value write sequence: Write 1-> Write 0-> Write 1-> Read 0-> Read 1. The last read can be omitted, but can be used to confirm that the lock key has been activated. Note: The value of LCK [15: 0] cannot be changed when operating the write sequence of the lock key. Any error in the lock sequence aborts the lock. After the first key-lock timing of any bit of the port, the read LCKK bit returns 1 until the system is reset or GPIO is reset.
15: 0	LCKy	RW	0	y = 15..0 Can only be written when the LCKK bit is 0. 0: Port configuration not locked 1: Port configuration locked

9.4.9 GPIO multiplexing function low register (GPIOx_AFRL) (x = A.. F)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7 [3: 0]				AFSEL6 [3: 0]				AFSEL5 [3: 0]				AFSEL4 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AFSEL3 [3: 0]				AFSEL2 [3: 0]				AFSEL1 [3: 0]				AFSEL0 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	AFSELy [3: 0]	RW	4'h0	y = 7..0 Configure multiplexing function IO AFSELy selection: 0000: AF01000 : AF8 0001: AF11001 : AF9 0010: AF2 1010 : AF10 0011: AF3 1011 : AF11 0100: AF4 1100 : AF12 0101: AF5 1101 : AF13 0110: AF6 1110 : AF14 0111: AF7 1111 : AF15

9.4.10 GPIO multiplexing function high register (GPIOx_AFRH) (x = A.. F)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL15 [3: 0]				AFSEL14 [3: 0]				AFSEL13 [3: 0]				AFSEL12 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AFSEL11 [3: 0]				AFSEL10 [3: 0]				AFSEL9 [3: 0]				AFSEL8 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	AFSELy [3: 0]	RW	4'h0	y = 15..8 Configure multiplexing function IO AFSELy selection: 0000: AF01000 : AF8 0001: AF11001 : AF9 0010: AF2 1010 : AF10 0011: AF3 1011 : AF11 0100: AF4 1100 : AF12

				0101: AF5 1101 : AF13 0110: AF6 1110 : AF14 0111: AF7 1111 : AF15

9.4.11 GPIO port bit reset register (GPIOx_BRR) (x = A.. F)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	2	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	BRy	W	0	y = 15..0 Read return value is 0 0: The corresponding ODy bit is affected 1: Clear the corresponding ODy bit

10. System configuration controller (SYSCFG)

10.1 Main features

The SYSCFG module provides the following functions:

- I2C IO Noise Filter Control
- I2C Fm + Mode control
- All IO Noise Filter Control
- EXTI IO Selection
- Analog Channel 2 Control
- Mapping the initial program area according to different boot modes
- DMA Peripheral Channel Selection
- Timer brake input and external trigger control
- CCM erase control, CCM busy status flag
- Parity control of SRAM1 and CCM
- CCM page write protection

10.2 TIMx registers

10.2.1 SYSCFG configuration register 1 (SYSCFG_CFGR1)

Address offset: 0x00

Reset value: 0x0000 000x (x depends on the boot mode configuration in the BOOT0 pin and options byte)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
I2C_EIIC [15: 0]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					I2C4 _ FMP	I2C3 _ FMP	I2C2 _ FMP	I2C1 _ FMP	CTC_SOF_SEL	PTP_PPS_REMAP	TIM2ITR1_REMAP [1: 0]		ETH_PHYSEL	MEM MODE [1: 0]	
-					RW	RW	RW	RW	RW	RW	RW		RW	RW	

Bit	Name	R/W	Reset Value	Function
31: 16	I2C_EIIC [15: 0]	RW	16'h0	Analog filtering of I2C-dependent IOs is enabled, each bit controls one IO 0: Analog filter disabled 1: Analog filter enabled

				I2C_EIIC [0]: controls the I2Canalog filter enable of GPIOA8 I2C_EIIC [1]: controls the I2Canalog filtering enable of GPIOA9 I2C_EIIC [2]: controls the I2C analog filtering enable of GPIOA13 I2C_EIIC [3]: controls the I2C analog filtering enable of GPIOA14 I2C_EIIC [4]: controls the I2C analog filtering enable of GPIOA15 I2C_EIIC [5]: I2Canalog filtering enable for controlling GPIOB5 I2C_EIIC [6]: controls the I2C analog filtering enable of GPIOB7 I2C_EIIC [7]: controls the I2Canalog filtering enable of GPIOB8 I2C_EIIC [8]: controls the I2C analog filtering enable of GPIOB9 I2C_EIIC [9]: I2Canalog filtering enable for controlling GPIOC3 I2C_EIIC [10]: I2C analogfiltering enable for controlling GPIOC6 I2C_EIIC [11]: I2Canalog filtering enable for controlling GPIOC7 I2C_EIIC [12]: controls the I2Canalog filtering enable of GPIOC8 I2C_EIIC [13]: I2Canalog filtering enable for controlling GPIOC9 I2C_EIIC [14]: controls the I2Canalog filtering enable of GPIOC11 I2C_EIIC [15]: I2Canalog filtering enable for controlling GPIOF0
15: 11	Reserved	-	-	-
10	I2C4_FMP	RW	0	I2C4 Fast-mode plus driving capability activation 0: Fm + drive mode of the pin selected by the AF channel of I2C4 is not enabled 1: I2C4 is enabled by the Fm + drive mode of the selected pin of the AF channel Configuring this bit to 1 automatically causes the driving capabilities of the following IOs to be configured to a maximum value: PA13, PB7, PC6, PC7
9	I2C3_FMP	RW	0	I2C3 Fast-mode plus driving capability activation 0: Fm + drive mode of the pin selected by the AF channel of I2C3 is not enabled 1: I2C3 is enabled by the Fm + drive mode of the selected pin of the AF channel Configuring this bit to 1 automatically causes the driving capabilities of the following IOs to be configured to a maximum value: PA8, PB5, PC8, PC9, PC11
8	I2C2_FMP	RW	0	I2C2 Fast-mode plus driving capability activation 0: I2C2 Fm + drive mode of pin selected by AF channel is not enabled 1: I2C2 is enabled by the Fm + drive mode of the selected pin of the AF channel Configuring this bit to 1 automatically causes the driving capabilities of the following IOs to be configured to a maximum value: PA8, PA9, PC3, PF0
7	I2C1_FMP	RW	0	I2C1 Fast-mode plus driving capability activation 0: I2C1 Fm + drive mode of pin selected by AF channel is not enabled 1: I2C1 is enabled by the Fm + drive mode of the selected pin of the AF channel Configuring this bit to 1 automatically causes the driving capabilities of the following IOs to be configured to a maximum value: PA13, PA14, PA15, PB7, PB8, PB9

6	CTC_SOF_SEL	RW	0	Select usb sof to output to ctc 0: usb sof from USB1 1: usb sof from USB2
5	PTP_PPS_REMAP	RW	0	Ethernet PTP PPS mapping It controls whether the Ethernet MAC PPS_PTS outputs to the PB5 pin 0: PTP_PPS does not output to PB5 pin 1: PTP_PPS output to PB5 pin
4: 3	TIM2ITR1_REMAP [1: 0]	RW	2'h0	TIM2 Internal Trigger1 Remap It controls the internal remapping of TIM2 ITR1. 00: low level 01: usb1_sof 10: usb2_sof 11: eth_ptp_intr
2	ETH_PHYSEL	RW	0	Ethernet PHY Interface Select Bit 0: PHY receives MII clock, PHY MII interface works 1: PHY receives RMII clock, PHY RMII interface works
1: 0	MEM_MODE [1: 0]	RW	x	Start mode select bit Mapping of 0x0000 0000 addresses of controls memory. 00: Main flash, mapped to 0x0000 0000 01: System flash, mapped to 0x0000 0000 10: ESMC, mapped to 0x0000 0000 11: SRAM, mapped to 0x0000 0000

10.2.2 SYSCFG configuration register 2 (SYSCFG_CFGR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.							SPF	Res.				ECCL	PVDL	SPL	CLL
-							RC_W1	-				RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	SPF	RC_W1	0	Parity error flag bits for SRAM1 and CCM SRAM This bit is set to 1 by the hardware when there is a parity error for SRAM1 or CCM SRAM. The software writes 1 to this bit to clear it 0: No parity error detected 1: Parity error detected
7: 4	Reserved	-	-	-
3	ECCL	RS	0	ECC lock enable bit This bit is set to 1 by software and can only be cleared by system reset. This bit can be used to enable and lock the FLASH ECC error connected to the break input of TIM1/8/15/16/17 Software writing 0 to this bit does not change the value of this bit 0: ECC error not connected to brake input for TIM1/8/15/16/17 1: ECC error connected to brake input for TIM1/8/15/16/17
2	PVDL	RS	0	PVD lock enable bit Software set, system reset and clear. It can be used to enable and lock the PVD connected to the brake input of TIM1/8/15/16/17, and also to lock the PVDE of the PWR_CR register. Software writing 0 to this bit does not change the value of this bit 0: The PVD interrupt is not connected to the brake input of TIM1/8/15/16/17. The PVDE bit can be written by software. 1: PVD interrupts the brake input connection to TIM1/8/15/16/17. The PVDE bit is read only.
1	SPL	RS	0	SRAM1 and CCM SRAM parity lock enable bits This bit is set to 1 by software and can only be cleared by system reset. This bit can be used to enable and lock the

				parity error signal of SRAM1 or CCM SRAM connected to the break input of TIM1/8/15/16/17 Software writing 0 to this bit does not change the value of this bit 0: Parity error signal of CCM SRAM is not connected to brake input of TIM1/8/15/16/17 1: Parity error signal of CCM SRAM is connected to brake input of TIM1/8/15/16/17
0	CLL	RS	0	Cortex-M4F LOCKUP Enable bit Software set, system reset and clear. It can enable and lock the LOCKUP (hardfault) output of the Cortex-M4F to the brake input of TIM1/8/15/16/17. Software writing 0 to this bit does not change the value of this bit 0: The LOCKUP output of the Cortex-M4F is not connected to the brake input of TIM1/8/15/16/17; 1: The LOCKUP output of the Cortex-M4F is connected to the brake input of TIM1/8/15/16/17;

10.2.3 SYSCFG configuration register 3 (SYSCFG_CFGR3)

Address offset: 0x8

Reset value: 0x7F7F 7F7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DMA 1_MAP4 [6: 0]							Res.	DMA 1_MAP3 [6: 0]						
-	RW							-	RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	DMA 1_MAP2 [6: 0]							Res.	DMA1_MAP1 [6: 0]						
-	RW							-	RW						

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 24	DMA 1_MAP4 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details
23	Reserved	-	-	-
22: 16	DMA 1_MAP3 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details
15	Reserved	-	-	-
14: 8	DMA 1_MAP2 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details
7	Reserved	-	-	-
6: 0	DMA1_MAP1 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details

10.2.4 SYSCFG configuration register 4 (SYSCFG_CFGR4)

Address offset: 0XC

Reset value: 0x7F7F 7F7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DMA2_MAP2 [6: 0]							Res.	DMA2_MAP1 [6: 0]						
-	RW							-	RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	DMA 1_MAP6 [6: 0]							Res.	DMA 1_MAP5 [6: 0]						
-	RW							-	RW						

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 24	DMA2_MAP2 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details
23	Reserved	-	-	-
22: 16	DMA2_MAP1 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details
15	Reserved	-	-	-

14: 8	DMA 1_MAP6 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details
7	Reserved	-	-	-
6: 0	DMA 1_MAP5 [6: 0]	RW	7'h7F	DMA1 channel request mapping. See DMA section for details

10.2.5 SYSCFG configuration register 5 (SYSCFG_CFGR5)

Address offset: 0x10

Reset value: 0x7F7F 7F7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DMA2_MAP6 [6: 0]							Res	DMA2_MAP5 [6: 0]						
-	RW							-	RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	DMA2_MAP4 [6: 0]							Res	DMA2_MAP3 [6: 0]						
-	RW							-	RW						

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 24	DMA2_MAP6 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details
23	Reserved	-	-	-
22: 16	DMA2_MAP5 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details
15	Reserved	-	-	-
14: 8	DMA2_MAP4 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details
7	Reserved	-	-	-
6: 0	DMA2_MAP3 [6: 0]	RW	7'h7F	DMA2 channel request mapping. See DMA section for details

10.2.6 External interrupt configuration register 1 (SYS_EXTICR1)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EXTI3 [3: 0]				EXTI2 [3: 0]				EXTI1 [3: 0]				EXTIO [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	EXTIx [3: 0]	RW	4'h0	EXTIx configuration (x = 0 ~ 3) Used to select the input source of the EXTIx external interrupt. See EXTI External Interrupt Event Mapping for details. 0000: PA [x] pin 0001: PB [x] pin 0010: PC [x] pin 0011: PD [x] pin 0100: PE [x] pin 0101: PF [x] pin

10.2.7 External interrupt configuration register 2 (SYS_EXTICR2)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EXTI7 [3: 0]				EXTI6 [3: 0]				EXTI5 [3: 0]				EXTI4 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	EXTIx [3: 0]	RW	4'h0	EXTIx configuration (x = 4 … 7) Used to select the input source of the EXTIx external interrupt. See EXTI External Interrupt Event Mapping for details. 0000: PA [x] pin 0001: PB [x] pin 0010: PC [x] pin 0011: PD [x] pin 0100: PE [x] pin 0101: PF [x] pin (only when x = 4, 5)

10.2.8 External interrupt configuration register 3 (SYS_EXTICR3)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI11 [3: 0]				EXTI10 [3: 0]				EXTI9 [3: 0]				EXTI8 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	EXTIx [3: 0]	RW	4'h0	EXTIx configuration (x = 8 … 11) Used to select the input source of the EXTIx external interrupt. See EXTI External Interrupt Event Mapping for details. 0000: PA [x] pin 0001: PB [x] pin 0010: PC [x] pin 0011: PD [x] pin 0100: PE [x] pin

10.2.9 External interrupt configuration register 4 (SYS_EXTICR4)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI15 [3: 0]				EXTI14 [3: 0]				EXTI13 [3: 0]				EXTI12 [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	EXTIx [3: 0]	RW	4'h0	EXTIx configuration (x = 12 … 15) Used to select the input source of the EXTIx external interrupt. See EXTI External Interrupt Event Mapping for details. 0000: PA [x] pin 0001: PB [x] pin 0010: PC [x] pin 0011: PD [x] pin 0100: PE [x] pin

10.2.10 GPIOA Filter Enable Register (PA_ENS)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

PA_ENS[15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PA_ENS[15: 0]	RW	16'h0	Noise filtering enabled for GPIOA 0 ~ 15 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.11 GPIOB Filter Enable Register (PB_ENS)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB_ENS[15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PB_ENS[15: 0]	RW	16'h0	Noise filtering enabled for GPIOB 0 ~ 15 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.12 GPIOC Filter Enable Register (PC_ENS)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC_ENS[15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PC_ENS[15: 0]	RW	16'h0	Noise filtering enabled for GPIOC 0 ~ 15 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.13 GPIOD Filter Enable Register (PD_ENS)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD_ENS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PD_ENS [15: 0]	RW	16'h0	Noise filtering enabled for GPIOD 0 ~ 15 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.14 GPIOE Filter Enable Register (PE_ENS)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE_ENS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PE_ENS [15: 0]	RW	16'h0	Noise filtering enabled for GPIOE 0 ~ 15 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.15 GPIOF Filter Enable Register (PF_ENS)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										PF_ENS[5: 0]					
-										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	PF_ENS[5: 0]	RW	6'h0	Noise filtering enabled for GPIOF 0 ~ 5 0: Turn off noise filtering function 1: Turn on noise filtering function

10.2.16 GPIOA analog channel 2 enable register (PA_ANA2ENR)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					PA10_ENA2	Res.		PA7_ENA2	Res.	PA5_ENA2	PA4_ENA2	PA3_ENA2	PA2_ENA2	PA1_ENA2	PA0_ENA2
-					RW	-		RW	-	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10	PA10_ENA2	RW	0	PA10 analog channel 2 enabled:USB1_OTG_FS_ID 0: Disabled 1: Enable
9: 8	Reserved	-	-	-
7	PA7_ENA2	RW	0	PA7 Analog Channel 2 Enable: COMP2_INP 0: Disabled 1: Enabled
6	Reserved	-	-	-
5	PA5_ENA2	RW	0	PA5 analog channel 2 enabled: COMP2_INM, 0: Disabled 1: Enabled
4	PA4_ENA2	RW	0	PA4 analog channel 2 enabled: COMP1_INM 0: Disabled 1: Enabled
3	PA3_ENA2	RW	0	PA3 analog channel 2 enabled: COMP2_INP 0: Disabled 1: Enabled
2	PA2_ENA2	RW	0	PA2 Analog Channel 2 Enable: COMP2_INM 0: Disabled 1: Enabled

1	PA1_ENA2	RW	0	PA1 analog channel 2 enabled: COMP1_INP 0: Disabled 1: Enabled
0	PA0_ENA2	RW	0	PA0 analog channel 2 enabled: COMP1_INM, COMP3_INP 0: Disabled 1: Enabled

10.2.17 GPIOB analog channel 2 enable register (PB_ANA2ENR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.													PB2 _ ENA2	PB1 _ ENA2	PB0 _ ENA2
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	PB2_ENA2	RW	0	PB2 Analog Channel 2 Enable: COMP4_INM 0: Disabled 1: Enabled
1	PB1_ENA2	RW	0	PB1 analog channel 2 enabled: COMP1_INP 0: Disabled 1: Enabled
0	PB0_ENA2	RW	0	PB0 analog channel 2 enabled: COMP4_INP 0: Disabled 1: Enabled

10.2.18 GPIOC analog channel 2 enable register (PC_ANA2ENR)

Address offset: 0x48

Reset value: 0x0000 C000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC15 _ ENA2	PC14 _ ENA2	Res.										PC3 _ ENA2	PC2 _ ENA2	PC1 _ ENA2	PC0 _ ENA2
RW	RW	-										RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	PC15_ENA2	RW	1	PC15 Analog Channel 2 Enable: LSE (OSC32) 0: Disabled 1: Enabled
14	PC14_ENA2	RW	1	PC14 Analog Channel 2 Enable: LSE (OSC32) 0: Disabled 1: Enabled
13: 4	Reserved	-	-	-
3	PC3_ENA2	RW	0	GPIOC3 Analog Channel 2 Enable: ADC1_CH9/ADC2_CH9 0: Disabled 1: Enabled
2	PC2_ENA2	RW	0	PC2 Analog Channel 2 Enable: ADC1_CH8/ADC2_CH8/USB2_OTG_FS_ID 0: Disabled 1: Enabled
1	PC1_ENA2	RW	0	PC1 analog channel 2 enabled: COMP3_INP 0: Disabled 1: Enabled
0	PC0_ENA2	RW	0	PC0 analog channel 2 enabled: COMP3_INM

				0: Disabled 1: Enabled

10.2.19 GPIOD analog channel 2 enable register (PD_ANA2ENR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		PD13_ENA2	PD12_ENA2	PD11_ENA2	PD10_ENA2	Res.									
-		RW	RW	RW	RW	-									

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13	PD13_ENA2	RW	0	PD13 Analog Channel 2 Enabled: ADC3_CH10 0: Disabled 1: Enabled
12	PD12_ENA2	RW	0	PD12 Analog Channel 2 Enable: ADC3_CH9 0: Disabled 1: Enabled
11	PD11_ENA2	RW	0	PD11 Analog Channel 2 Enable: ADC3_CH8 0: Disabled 1: Enabled
10	PD10_ENA2	RW	0	PD10 Analog Channel 2 Enable: ADC3_CH7 0: Disabled 1: Enabled
9: 0	Reserved	-	-	-

10.2.20 GPIOE analog channel 2 enable register (PE_ANA2ENR)

Address offset: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			PE12_ENA2	PE11_ENA2	PE10_ENA2	Res.		PE7_ENA2	Res.						
-			RW	RW	RW	-		RW	-						

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	PE12_ENA2	RW	0	PE12 Analog Channel 2 Enable: ADC3_CH15 0: Disabled 1: Enabled
11	PE11_ENA2	RW	0	PE11 Analog Channel 2 Enabled: ADC3_CH14 0: Disabled 1: Enabled
10	PE10_ENA2	RW	0	PE10 Analog Channel 2 Enable: ADC3_CH13 0: Disabled 1: Enabled
9: 8	Reserved	-	-	-
7	PE7_ENA2	RW	0	PE7 Analog Channel 2 Enable: COMP4_INP 0: Disabled 1: Enabled

6: 0	Reserved	-	-	-

10.2.21 PVD_IN analog channel 2 enable register (PVDIN_ANA2ENR)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															PVDIN_ENA2
-															RW

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	PVDIN_ENA2	RW	0	Enable analog channel 2 of the PVD_IN pin (PB7)

10.2.22 SYSCFG CCM SRAM Control and Status Register (SYSCFG_SCSR)

Address offset: 0x70

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.														CCMBSY	CCMER
-														R	W

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	CCMBSY	R	0	CCM SRAM Erase Flag Bit 0: No CCM SRAM erase operation in progress 1: CCM SRAM erase operation in progress
0	CCMER	W	0	CCM SRAM Erasure Setting this bit to 1 will initiate a hardware erase operation of CCM SRAM, and the bit will be cleared by hardware

10.2.23 SYSCFG CCM SRAM write protection register (SYSCFG_SWPR)

Address offset: 0x74

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P31 WP	P30 WP	P29 WP	P28 WP	P27 WP	P26 WP	P25 WP	P24 WP	P23 WP	P22 WP	P21 WP	P20 WP	P19 WP	P18 WP	P17 WP	P16 WP
RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P15 WP	P14 WP	P13 WP	P12 WP	P11 WP	P10 WP	P9WP	P8WP	P7WP	P6WP	P5WP	P4WP	P3WP	P2WP	P1WP	P0WP
RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31: 0	PxWP (x = 0-31)	RS	0	CCM SRAM page x write protection These bits are set to 1 by software and can only be cleared by system reset. 0: CCM SRAM page x write protection off 1: CCM SRAM page x write protection enabled

10.2.24 SYSCFG CCM SRAM Key Register (SYSCFG_SKR)

Address offset: 0x78

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								KEY [7: 0]							

-	W

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
KEY [7: 0]	KEY [7: 0]	W	8’h0	Software erases write protection key values of CCM SRAM The CCMER bit of the SYSCFG_SCSR register cannot be write-protected only by following the following steps: 1. Write "0xCA" to the KEY of this register [7: 0] 2. Write "0x53" to KEY of this register [7: 0] If write sequence is wrong, write protection will be reactivated

11. Peripheral interconnection matrix

11.1 Introduction

Some peripherals have connectivity relationships with each other.

This allows autonomous communication between peripherals, thereby saving CPU resources while also saving power consumption. In addition, these hardware connections eliminate software latency. Depending on the peripheral, these interconnections can operate in run, sleep, and stop modes.

Source	Destination																																		
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM6	TIM7	TIM8	TIM9	TIM10	TIM11	TIM12	TIM13	TIM14	TIM15	TIM16	TIM17	TIM18	TIM19	LPTIM	ADC1	ADC2	ADC3	DAC1	DAC2	COMP1	COMP2	COMP3	COMP4	IRTIM
TIM1	-	1	1	1	1			1	1	1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3	1 4	1 4	1 0	1 0	1 0	1 0	
TIM2	1	-	1	-	1			1	1	1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3	1 4	1 4	1 0	1 0	1 0	1 0	
TIM3	1	1		1	1			1	1	1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3	1 4	1 4	1 0	1 0	1 0	1 0	
TIM4	1	1	1		1			1	1	1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3	1 4	1 4	1 0	1 0	1 0	1 0	
TIM5	1	1	1	1				1	1	1	1	1	1	1	1	1	1	1	1					1 4	1 4					
TIM6																					1 3	1 3	1 3	1 4	1 4					
TIM7																					1 3	1 3	1 3	1 4	1 4					
TIM8	1	1	1	1	1				1	1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3	1 4	1 4	1 0	1 0	1 0	1 0	
TIM9	1	1	1	1	1			1		1	1	1	1	1	1	1	1	1	1		1 3	1 3	1 3							
TIM10	1	1	1	1	1			1	1		1	1	1	1	1	1	1	1	1		1 3	1 3	1 3							
TIM11	1	1	1	1	1			1	1	1		1	1	1	1	1	1	1	1		1 3	1 3								
TIM12	1	1	1	1	1			1	1	1	1		1	1	1	1	1	1	1											
TIM13	1	1	1	1	1			1	1	1	1	1		1	1	1	1	1	1				1 3							
TIM14	1	1	1	1	1			1	1	1	1	1	1		1	1	1	1	1				1 3							
TIM15	1	1	1	1	1			1	1	1	1	1	1	1		1	1	1	1		1 3	1 3		1 4	1 4	1 0	1 0	1 0	1 0	
TIM16	1	1	1	1	1			1	1	1	1	1	1	1	1		1	1	1											1 1
TIM17	1	1	1	1	1			1	1	1	1	1	1	1	1	1		1	1				1 3							1 1
TIM18	1	1	1	1	1			1	1	1	1	1	1	1	1	1	1		1		1 3	1 3		1 4	1 4	1 0	1 0	1 0	1 0	
TIM19	1	1	1	1	1			1	1	1	1	1	1	1	1	1	1	1			1 3	1 3								
LPTIM																														
ADC1	2	-	2	-																										
ADC2								2																						
ADC3								2																						
DAC1																														
DAC2																														
HSE																4	4													
LSE					4											4	4													
HSI16																														
LSI					4											4	4													
MCO																4	4													
EXTI																								1 4	1 4					
COMP 1	4 2 3 8 9	4 2 3 5	4 2 3 5 7	4 2 3 5	4 2 3 5			4 2 3 8 9							3 4 8	3 8	3 8	4 2 3 5												
COMP 2	4 2 3 8 9	4 2 3 5	4 2 3 5 7	4 2 3 5 6	4 2 3 5			4 2 3 8 9							3 4 5 8	3 8	3 8	4 2 3 5												
COMP 3	4 2 3 8 9	4 2 3 5	4 2 3 5 6	4 2 3 5	4 2 3 5			4 2 3 8 9							3 5 8	3 8	3 8	4 2 3 5												

COMP 4	4 2 3 8 9	4 2 3 5 6	4 2 3 5	4 2 3 5	4 2 3 5			4 2 3 8 9							3 5 8	3 8	3 8	4 2 3 5											
SYST ERR	1 2							1 2							1 2	1 2	1 2												

11.2 Connection Details

11.2.1 Input trigger (itrr)

Table 11-1 Interconnect 1

Timer input trigger signal	Timer input triggers source allocation									
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM9	TIM12	TIM15	TIM18
Timx_itr 0	-	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o	Tim1_trg o
Timx_itr 1	Tim2_trg o	-	Tim2_trg o	Tim2_trg o	Tim2_trg o	Tim2_trg o	Tim2_trg o	Tim2_trg o	Tim2_trg o	Tim2_trg o
Timx_itr 2	Tim3_trg o	Tim3_trg o	-	Tim3_trg o	Tim3_trg o	Tim3_trg o	Tim3_trg o	Tim3_trg o	Tim3_trg o	Tim3_trg o
Timx_itr 3	Tim4_trg o	Tim4_trg o	Tim4_trg o	-	Tim4_trg o	Tim4_trg o	Tim4_trg o	Tim4_trg o	Tim4_trg o	Tim4_trg o
Timx_itr 4	Tim5_trg o	Tim5_trg o	Tim5_trg o	Tim5_trg o	-	Tim5_trg o	Tim5_trg o	Tim5_trg o	Tim5_trg o	Tim5_trg o
Timx_itr 5	Tim8_trg o	Tim8_trg o	Tim8_trg o	Tim8_trg o	Tim8_trg o	-	Tim8_trg o	Tim8_trg o	Tim8_trg o	Tim8_trg o
Timx_itr 6	Tim9_oc 1	Tim9_oc 1	Tim9_oc 1	Tim9_oc 1	Tim9_oc 1	Tim9_oc 1	-	Tim9_oc 1	Tim9_oc 1	Tim9_oc 1
Timx_itr 7	Tim9_oc 2	Tim9_oc 2	Tim9_oc 2	Tim9_oc 2	Tim9_oc 2	Tim9_oc 2	-	Tim9_oc 2	Tim9_oc 2	Tim9_oc 2
Timx_itr 8	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c	Tim10_o c
Timx_itr 9	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c	Tim11_o c
Timx_itr 10	Tim12_o c1	Tim12_o c1	Tim12_o c1	Tim12_o c1	Tim12_o c1	Tim12_o c1	Tim12_o c1	-	Tim12_o c1	Tim12_o c1
Timx_itr 11	Tim12_o c2	Tim12_o c2	Tim12_o c2	Tim12_o c2	Tim12_o c2	Tim12_o c2	Tim12_o c2	-	Tim12_o c2	Tim12_o c2
Timx_itr 12	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c	Tim13_o c
Timx_itr 13	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c	Tim14_o c
Timx_itr 14	Tim15_trg o	Tim15_trg o	Tim15_trg o	Tim15_trg o	Tim15_trg o	Tim15_trg o	Tim15_trg o	Tim15_trg o	-	Tim15_trg o
Timx_itr 15	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c	Tim16_o c
Timx_itr 16	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c	Tim17_o c
Timx_itr 17	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	Tim18_trg o	-
Timx_itr 18	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c	Tim19_o c

Some TIMx timers are connected together internally for synchronization or connection of timers. When a timer is in master mode, it can reset, start, stop the slave mode timer.

11.2.2 External trigger (etr)

Table 11-2 Interconnect 2

Timer external trigger input signal	Timer external trigger source allocation						
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM18
Timx_etr0	TIM1_ETR	TIM2_ETR	TIM3_ETR	TIM4_ETR	TIM5_ETR	TIM8_ETR	TIM18_ETR
Timx_etr1	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out
Timx_etr2	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out
Timx_etr3	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out
Timx_etr4	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out
Timx_etr5	adc1_awd1	TIM3_ETR	TIM2_ETR	TIM3_ETR	TIM2_ETR	adc2_awd1	TIM3_ETR
Timx_etr6	adc1_awd2	TIM4_ETR	TIM4_ETR	TIM5_ETR	TIM3_ETR	adc2_awd2	TIM4_ETR
Timx_etr7	adc1_awd3	TIM5_ETR	-	TIM18_ETR	TIM18_ETR	adc2_awd3	TIM5_ETR
Timx_etr8		TIM18_ETR	adc1_awd1			adc3_awd1	

Timx_etr9			adc1_awd2			adc3_awd2	
Timx_etr10			adc1_awd3			adc3_awd3	

The output of the comparator may be connected to a timer (TIMx) input capture or a TIMx_ETR signal or a TIMx_OCREFCLR signal,or a brake input signal of the timer may be generated.

11.2.3 Ocref clear

Table 11-3 Interconnect 3

Timer ocref Clear signal	Timer ocref clear signal allocation							
	TIM1	TIM2	TIM3	TIM8	TIM15	TIM16	TIM17	TIM18
timx_ocref_clr0	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out
timx_ocref_clr1	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out
timx_ocref_clr2	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out
timx_ocref_clr3	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out

11.2.4 TI1

Table 11-4 Interconnect 4

Timer TI1 Input signal	Timer TI1 Signal allocation									
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM15	TIM16	TIM17	TIM18
timx_ti1_in0	TI1	TI1	TI1	TI1	TI1	TI1	TI1	TI1	TI1	TI1
timx_ti1_in1	Comp1_out	Comp1_out	Comp1_out	Comp1_out	LSI	Comp1_out				Comp1_out
timx_ti1_in2	Comp2_out	Comp2_out	Comp2_out	Comp2_out		Comp2_out	Comp1_out	MCO	MCO	Comp2_out
timx_ti1_in3	Comp3_out	Comp3_out	Comp3_out	Comp3_out		Comp3_out	Comp2_out	HSE_Div32	HSE_Div32	Comp3_out
timx_ti1_in4	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp1_out	Comp4_out				Comp4_out
timx_ti1_in5					Comp2_out					
timx_ti1_in6					Comp3_out			LSI	LSI	
timx_ti1_in7					Comp4_out					

11.2.5 TI2

Table 11-5 Interconnect 5

Timer T2 Input signal	Timer TI2 Signal allocation							
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM15	TIM18
timx_ti2_in0	TI2	TI2	TI2	TI2	TI2	TI2	TI2	TI2
timx_ti2_in1		Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp2_out	Comp1_out
timx_ti2_in2		Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp3_out	Comp2_out
timx_ti2_in3		Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out		Comp3_out
timx_ti2_in4		Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out		Comp4_out

11.2.6 TI3

Table 11-6 Interconnect 6

Timer TI3 Input signal	Timer TI3 Signal allocation						
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM18
timx_ti3_in0	TI3	TI3	TI3	TI3	TI3	TI3	TI3
timx_ti3_in1		Comp4_out	Comp3_out	Comp2_out			

11.2.7 TI4

Table 11-7 Interconnect 7

Timer TI4 Input signal	Timer TI4 Signal allocation						
	TIM1	TIM2	TIM3	TIM4	TIM5	TIM8	TIM18
timx_ti4_in0	TI4	TI4	TI4	TI4	TI4	TI4	TI4
timx_ti4_in1		Comp1_out		Comp3_out			
timx_ti4_in1		Comp2_out					

11.2.8 break1

Table 11-8 Interconnect 8

	Timer Brake 1 Signal Distribution				
Timer Brake 1 Source	TIM1_break	TIM8_break	TIM15_break	TIM16_break	TIM17_break
	TIM1_BKINpin	TIM8_BKINpin	TIM15_BKINpin	TIM16_BKINpin	TIM17_BKINpin
	Comp1_out	Comp1_out	Comp1_out	Comp1_out	Comp1_out

	Comp2_out	Comp2_out	Comp2_out	Comp2_out	Comp2_out
	Comp3_out	Comp3_out	Comp3_out	Comp3_out	Comp3_out
	Comp4_out	Comp4_out	Comp4_out	Comp4_out	Comp4_out

The timer's brake input has two channels:

1. break1 channel, which includes application failures and system failures.

2. The break2 channel only includes application failures.

11.2.9 break2

Table 11-9 Interconnect 9

	Timer Brake 2 Signal Distribution	
	TIM1_break2	TIM8_break2
Timer Brake 2 Source	TIM1_BKIN2pin	TIM8_BKIN2pin
	Comp1_out	Comp1_out
	Comp2_out	Comp2_out
	Comp3_out	Comp3_out
	Comp4_out	Comp4_out

11.2.10 Comparator blanking source

Table 11-10 Interconnect 10

	Comparator blanking source allocation			
	COMP1	COMP2	COMP3	COMP4
Comparator blanking source	tim1_oc5	tim1_oc5	tim1_oc5	tim3_oc4
	tim2_oc3	tim2_oc3	tim3_oc3	tim8_oc5
	tim3_oc3	tim3_oc3	tim2_oc4	tim15_oc2
	tim8_oc5	Tim8_oc5	Tim8_oc5	tim1_oc5
	tim18_oc3	tim18_oc3	tim18_oc4	tim18_oc4
	tim15_oc1	tim15_oc1	tim15_oc1	tim15_oc1
	tim4_oc3	tim4_oc3	Tim4_oc3	tim4_oc3

TIMx can be used as a blanking window for COMPx.

The output signal TIMx_OCx of the timer is the blanking source input of COMPx.

11.2.11 IRTIM control signal

Table 11-11 Interconnect 11

IRTIM control signal	IRTIM Control Signal Allocation
Modulated envelope signal	Tim16_oc1
Carrier signal	Tim17_oc1

The general purpose timer (TIM16/TIM17) output channel TIMx_OC1 is used to generate the wave-form output of the infrared signal.

11.2.12 System error

Table 11-12 Interconnect 12

System Error Source	System error signal is assigned to Timer signal				
	TIM1	TIM8	TIM15	TIM16	TIM17
Flash ECC error	TIM1_BK	TIM8_BK	TIM15_BK	TIM16_BK	TIM17_BK
PVD output					
SRAM1/CCM parity					
Cortex-M4F Lockup					
CSS					

12. DMA Controller (DMA)

12.1 Overview

Direct memory access (DMA) is used to provide high-speed data transfer between peripherals and memory, between memory and memory, or between peripherals and peripherals. Data can be quickly moved by DMA without any CPU actions. This keeps the CPU resources free for other operations. The two DMA controllers have a total of 12 channels (6 channels for DMA1 and 6 channels for DMA2), and each channel is dedicated to managing access requests from one or more peripheral devices. The DMA also has an arbiter to coordinate the priorities of individual DMA requests.

12.1.1 Main features

The DMA supports:

- Single AHB Master
- There is programmable channel priority
- Supports peripheral-to-memory, memory-to-peripheral, memory-to-memory and peripheral-to-peripheral data transfers
- Supports on-chip memory devices (such as Flash, SRAM, AHB, and APB peripherals) as source and destination
- Supports programmable source and destination addresses, the addresses can be optionally incremented, decremented or unchanged
- The multi-block transfer function can be realized by automatic reloading of channel registers and consecutive addresses between blocks
- Supports programmable burst transaction sizes
- Each channel generates an interrupt request. Each interrupt request is caused by any of six DMA events: Transfer Complete, Block Transfer Complete, Source Transaction Complete, Destination Transaction Complete, Transfer Error, or Half Block Transfer Complete
- The FIFO depth of channels 0 to 4 is 8 bytes, and the FIFO depth of channel 5 is 32 bytes

12.1.2 Basic definition

- Source Peripheral-A device located on the AHB layer from which the DMA can read data, which then stores in the channel FIFO. The source peripheral forms a channel together with the target peripheral. The source peripheral is an AHB or APB slave. If the source is an APB slave, it is accessed through the AHB-APB bridge.
- Destination peripheral-The device to which the DMA writes data stored in the FIFO (previously read from the source peripheral). The target peripheral is an AHB or APB slave. If the target is an APB slave, it is accessed through the AHB-APB bridge.
- Memory-The source or destination is always ready for a DMA transfer and does not need to have a handshake interface with the DMA.
- Channel-The path through which read/write data occurs via FIFO between a source peripheral on a configured AHB layer and a target peripheral on the same or different AHB layer. If the source peripheral is not memory, the source handshake interface is assigned to the channel. If the target peripheral is not memory, the target handshake interface is assigned to the channel. Source and

destination handshake interfaces can be dynamically allocated by programming channel registers.

- Flow Controller-The device (DMA source or destination peripheral) that determines the DMA block transmission length and terminates it.
- Transport Hierarchy Figure 12-1 shows the hierarchy between DMA transmissions, block transmissions, transactions (single or burst), and AHB transmissions (single or burst) for non-memory peripherals

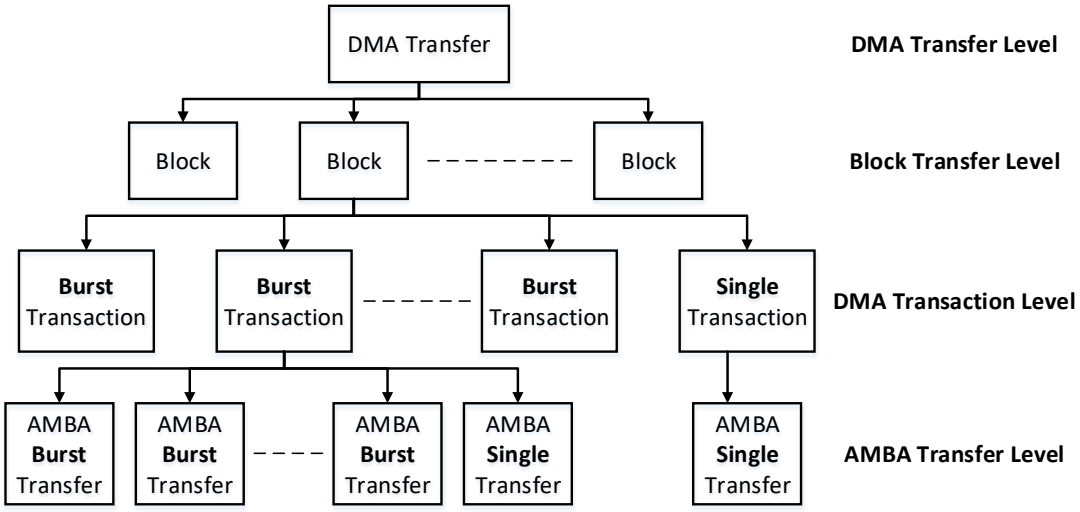


Figure 12-1 DMA transfer hierarchy for non-memory peripherals

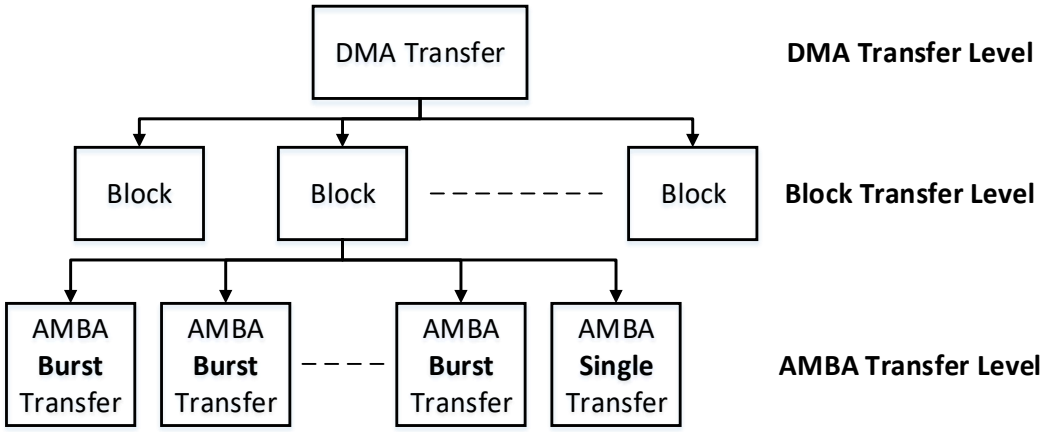


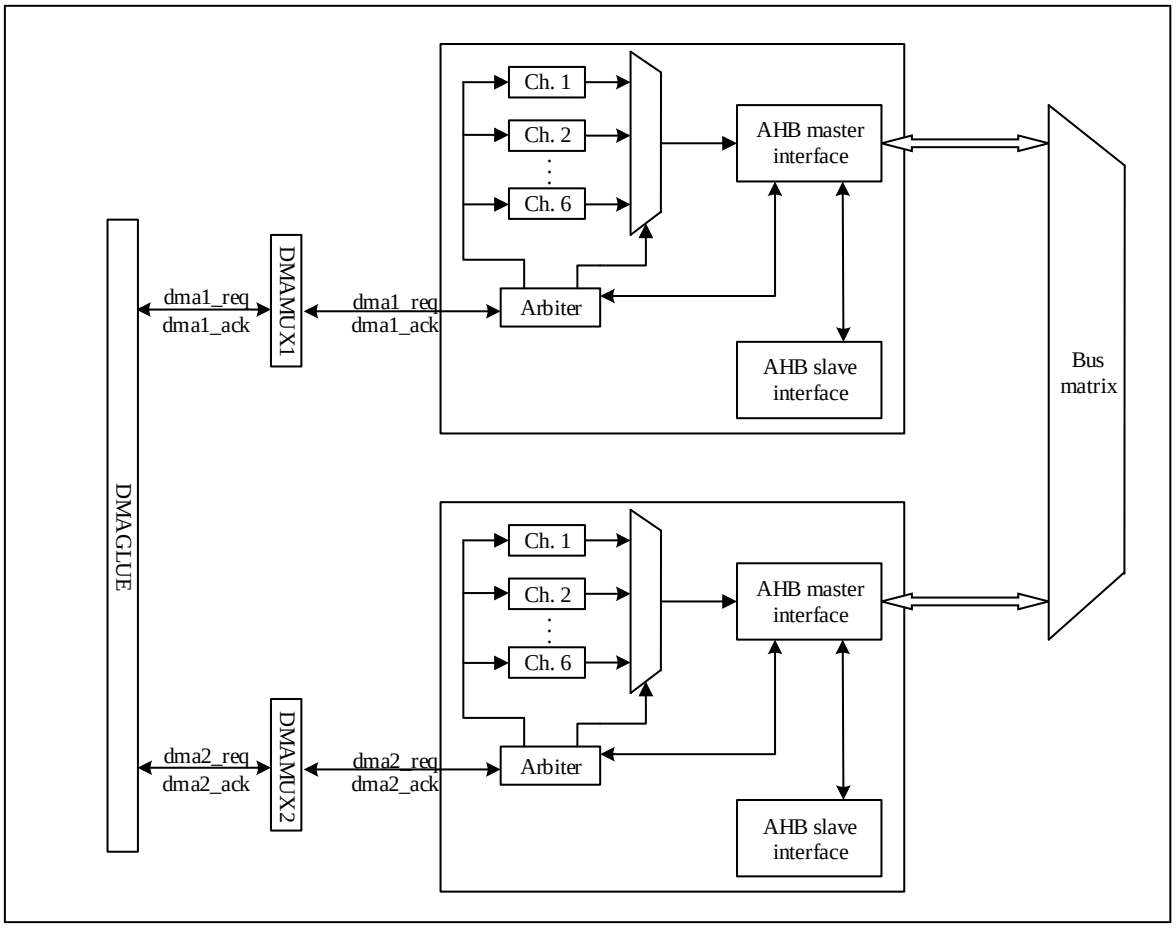
Figure 12-2 DMA Transfer Hierarchy of Graph Memory

- Block-A block of DMA data, the number of which is the block length, determined by the flow controller. For transfers between DMA and memory, a block is directly decomposed into a series of bursts and individual transfers. For transfers between DMA and non-memory peripherals, a block is broken down into sequences of DMA transactions (single and burst).
- Transaction-The basic unit of DMA transmission, determined by a hardware or software handshake interface. If the peripheral is a non-memory device, then the transaction is only related to the transfer between the source or destination peripheral of the DMA.
- There are two kinds of transactions:
  - Single Transaction-The length of a single transaction is always 1 and converted to a single AHB transfer.
  - Burst Transaction-The length of a burst transaction can be configured. A burst transaction is

converted into a series of bursts and an AHB single transmission. The burst transaction length is controlled by the program, and usually has a certain relationship with the FIFO size of DMA, source peripheral and target peripheral.

- DMA Transfer-The software controls the number of blocks for DMA transmission. After the DMA transfer is complete, the hardware in the DMA disables the channel and generates an interrupt to indicate that the DMA transfer is complete. The channel can then be reprogrammed for the new DMA transmission.
  - Monoblock DMA transmission-consists of a single block.
  - Multi-block DMA transmission-A DMA transmission may consist of multiple DMA blocks. Multi-block DMA transfers are supported by automatic reloading of channel registers and contiguous blocks.
    - Automatic Reload-DMA automatically reloads the value when the channel was first enabled into the channel register at the end of each block.
    - Contiguous Blocks-The address between contiguous blocks is selected as a continuation of the end of the previous block.

12.1.3 Top-level structure diagram



The DMA and CPU share the system bus. If the DMA and CPU access the same destination (memory or peripheral), the DMA request will stop the CPU from accessing the system bus for some clock cycles. Bus matrix arbitration uses the round-robin method to ensure that the CPU occupies at least half of the system bus bandwidth.

12.2 Functional description

12.2.1 Interface definition

Source single transaction size in bytes: src_single_size_bytes = DMA_CH_CTRLLx.SRC_TR_WIDTH/8

Source burst transaction size in bytes): src_burst_size_bytes = DMA_CH_CTRLLx.SRC_MSIZE * src_single_size_bytes

Target single transaction size in bytes: dst_single_size_bytes = DMA_CH_CTRLLx.DST_TR_WIDTH/8

Target burst transaction size in bytes): dst_burst_size_bytes = DMA_CH_CTRLLx.DST_MSIZE * dst_single_size_bytes

Block size in bytes): blk_size_bytes_dma = DMA_CH_CTRLHx.BLOCK_TS * src_single_size_bytes

12.2.2 Memory peripheral

DMA_CH_CTRLLx.SRC_MSIZE and DMA_CH_CTRLLx.DST_MSIZE are only valid for peripherals with handshake interfaces, andthey cannot be used to define burst lengths of memory peripherals.

Consequently:

- The length of the burst transfer to the memory is always equal to the number of data items available in the channel FIFO or the number of data items required to complete the block transfer, whichever is smaller.
- The length of a burst transfer from the memory is always equal to the space available in the channel FIFO or the number of data items required to complete the block transfer, whichever is smaller.

12.2.3 DMA transmission

DMA block transfers may be requested from peripherals or triggered by software in the case of memory-to-memory transfers.

After the event, a single DMA transfer step:

- The peripheral sends a DMA request signal to the DMA controller.
- The DMA controller responds to the request according to the priority of the channel associated with this peripheral request.
- Once the DMA controller grants the peripheral, the transmission is initiated, and when the transmission is complete, the DMA controller sends a reply to the peripheral.
- When the peripheral receives an answer from the DMA controller, it releases its request.
- Once the peripheral cancels the request assertion, the DMA controller releases the channel currently occupied by the peripheral.

The request/acknowledgement protocol is used when the peripheral is the source or destination of the transmission. For example, in a memory-to-peripheral transfer, the peripheral initiates the transfer by driving its single request signal to the DMA controller. The DMA controller then writes the data from the FIFO to the peripheral.

For a given channel x, a DMA block transmission consists of the following repetitive sequence:

- Data read from a peripheral data register or a location in memory, the data bit width is configured in the DMA_CH_CTRLLx.SRC_TR_WIDTH register, the data length is configured in the DMA_CH_CTRLLx.SRC_MSIZE register, addressed through the internal current source address register, and the start address for the first transfer is configured in the DMA_SARx register.
- Data is written to a peripheral data register or a location in memory, the data bit width is configured in the DMA_CH_CTRLLx.DST_TR_WIDTH register, the data length is configured in the DMA_CH_CTRLLx.DST_MSIZE register, addressed through the internal current destination address register, and the start address for the first transfer is configured in the DMA_DARx register.
- The DMA_CTLx.BLOCK_TS register contains the number of data items remaining to be transferred, which indicates the total number of data items that have beenreadfrom the source after DMA initiates the transfer.

The above sequence is repeated until the set value of the DMA_CTLx.BLOCK_TS register is reached.

12.2.4 Arbiter

The arbiter initiates access to the peripheral/memory according to the priority of the channel request.

When a request channel x is granted (hardware-requested or software-triggered) access by the arbiter, a single or burst DMA transmission is made. The arbiter then again arbitrates all requested channels and selects the channel with the highest priority.

The priority of each channel can be set in the DMA_CH_CFGx. register, and there are 6 levels:

- ✓ 0
- ✓ 1
- ✓ 2
- ✓ 3
- ✓ 4
- ✓ 5

Priority 5 > 4 > 3 > 2 > 1 > 0.

A channel with a lower channel number at the same priority has a higher priority.

12.2.5 DMA channel

12.2.5.1 Programmable data sizes

Each channel can be DMA transferred between peripheral registers with fixed addresses and memory addresses. The amount of data transferred by DMA is programmable, up to 4095.

12.2.5.2 Pointer incrementation

By setting the SINC and DINC flag bits in the DMA_CH_CTRLLx register, pointers to peripherals and memory can be selectively autoincremented after each transfer.

When set to incremental mode, the next address to be transferred will be the previous address plus an incremental value, which depends on the selected data width of 8/16/32 bits. The first address transferred is the address stored in the DMA_SAx/DMA_DAx register. During transfer, these registers maintain their initial values, and the software cannot change and read out the address currently being transferred (in the internal current peripheral/memory address register).

When the channel is configured to transmit in a single block, DMA operations will no longer occur after the transmission ends (i.e. the transmission count becomes 0). To start a new DMA transfer, the number of transfers needs to be re-written in the DMA_CH_CTRLHx.BLOCK_TS register with the DMA channel closed.

12.2.5.3 Memory-to-memory mode

The operation of the DMA channel can be performed without peripheral requests. This operation is memory-to-memory mode. When the DMA_CH_CTRLLx.TT register is set, the DMA transfer will begin as soon as the software sets the CH_EN bit in the DMA_CH_ENx register to start the DMA channel. When the number of transferred data reaches the set value of the DMA_CH_CTRLHx.BLOCK_TS register, the DMA transfer ends.

12.2.6 Error handling

When an error occurs in an AHB transfer, the following happens:

1. Ongoing DMA transmission stops immediately
2. Related channel is disabled
3. Issue interrupt (if not masked)

If multiple channels are open, only the channels where AHB errors are detected are closed.

The contents of the FIFO are not cleared and become inaccessible and overwritten once the channel is re-enabled to initiate a new transfer.

Automatic resumption of transmission from the point where the error occurred is not supported, and the entire block transmission must be restarted to retransmit.

12.2.7 Interrupts and events

For each channel, there are six types of interrupt sources:

- Block Transfer Completion Interrupt

This interrupt is generated when the DMA block transmission is complete.

- Destination transaction completion interrupt

This interrupt is generated after the last AHB transmission of the requested single/burst transaction is completed from the handshake interface (hardware or software) of the target.

If the target of a channel is memory, then the channel will never generate a target transaction completion interrupt. Therefore, the corresponding enable of this interrupt does not work.

- Error interrupt

This interrupt occurs when an error response is received from an AHB slave on the bus during a DMA transmission. In addition, generating an error interrupt cancels the DMA transmission and disables the channel.

- Source transaction completion interrupt

This interrupt is generated after the last AHB transfer of the requested single/burst transaction is completed from the source's handshake interface (hardware or software).

If the source is memory, then the source transaction completion interrupt should be ignored because memory does not have the concept of "DMA Transaction Level".

- DMA transmission completely interrupted

This interrupt is generated when a DMA transfer to the target peripheral is completed.

- Half-block transmission completion interrupt

This interrupt is generated when half of the transmission of the DMA block is complete to the target peripheral.

12.2.8 DMAMUX mapping

DMA peripheral requests are mapped to each channel of DMA1 (6 channels) or DMA2 (6 channels), which is controlled by the DMAx_MAP register of SYSCFG. Each peripheral request can be mapped to any of the 12 channels through configuration.

The relationship between sequence number and peripheral request is shown in the following table:

DMA request MUX input	Source	DMA request MUX input	Source	DMA request MUX input	Source
0	ADC1	32	TIM1_TRIG	64	TIM5_CH4
1	ADC2	33	TIM1_COM	65	TIM5_UP
2	ADC3	34	TIM8_CH1	66	TIM5_TRIG
3	DAC1	35	TIM8_CH2	67	TIM18_CH1
4	DAC2	36	TIM8_CH3	68	TIM18_CH2
5	SPI1_RX	37	TIM8_CH4	69	TIM18_CH3
6	SPI1_TX	38	TIM8_UP	70	TIM18_CH4
7	SPI2_RX	39	TIM8_TRIG	71	TIM18_UP
8	SPI2_TX	40	TIM8_COM	72	TIM18_TRIG
9	SPI3_RX	41	TIM6_UP	73	TIM15_CH1
10	SPI3_TX	42	TIM7_UP	74	TIM15_CH2
11	USART1_RX	43	TIM2_CH1	75	TIM15_UP
12	USART1_TX	44	TIM2_CH2	76	TIM15_TRIG
13	USART2_RX	45	TIM2_CH3	77	TIM15_COM
14	USART2_TX	46	TIM2_CH4	78	TIM16_CH1
15	USART3_RX	47	TIM2_UP	79	TIM16_UP
16	USART3_TX	48	TIM2_TRIG	80	TIM17_CH1
17	LPUART1_RX	49	TIM3_CH1	81	TIM17_UP
18	LPUART1_TX	50	TIM3_CH2	82	AES_in
19	I2C1_RX	51	TIM3_CH3	83	AES_OUT
20	I2C1_TX	52	TIM3_CH4	84	Cordic_Read
21	I2C2_RX	53	TIM3_UP	85	Cordic_Write
22	I2C2_TX	54	TIM3_TRIG	86	ESMC_RX
23	I2C3_RX	55	TIM4_CH1	87	ESMC_TX
24	I2C3_TX	56	TIM4_CH2	88	SDIO
25	I2C4_RX	57	TIM4_CH3	89	UART1_RX
26	I2C4_TX	58	TIM4_CH4	90	UART1_TX
27	TIM1_CH1	59	TIM4_UP	91	UART2_RX
28	TIM1_CH2	60	TIM4_TRIG	92	UART2_TX
29	TIM1_CH3	61	TIM5_CH1	93	UART3_RX
30	TIM1_CH4	62	TIM5_CH2	94	UART3_TX
31	TIM1_UP	63	TIM5_CH3	95	LCDC_TX

12.2.9 Software Application Description

12.2.9.1 DMA Transmission Type

The DMA transmission may consist of a single or multiple block transmissions. On contiguous blocks of multi-block transfer, the SAx/DAx registers in the DMA are reprogrammed using one of the following methods:

- Automatic reload
- Continuous addresses between blocks

On contiguous blocks of multi-block transfers, the CH_CTRLx register in the DMA is reprogrammed using the following method:

- Automatic reload

12.2.9.2 Multi-block transmission

Multiblock transmission does not support exchanging source and destination during transmission.

Table 12-1 Programming of transfer types and channel register update methods

Transmission Type	RELoAD_SRC (CH_CFGLx)	RELoAD_DST (CH_CFGLx)	CH_CTRLx Update Method	SAx Update Method	DAx update method
1. Last transmission of a single or multiple blocks	0	0	None, reprogrammed	None	None
2. Automatic reload multi-block transmission, SA continuous	0	1	Load initial values	Continuous	Automatic reload
3. Automatic overload multi-block transmission, DA continuous	1	0	Load the initial value.	Automatic reload	Continuous
4. Automatic reload multi-block transmission	1	1	Load the initial value.	Automatic reload	Automatic reload

Automatic reload channel register

During automatic reloading, the channel registers are reloaded at each block completion with their initial values and a new value for the new block. Depending on the number of rows in Table 12-1, some or all of the SAx, DAx, and CH_CTRLx channel registers are reloaded from their initial values at the beginning of the block transfer.

Continuous addresses between blocks

In this case, the address between consecutive blocks is selected as a continuation of the end of the previous block.

Continuing the source or destination address between blocks is a function of the CH_CFGLx.RELOAD_SRC and CH_CFGLx.RELOAD_DST registers (see Table12-1).

You cannot select both SAx and DAx updates as consecutive updates. If you want to use this function, you should increase the size of the block transfer (CH _CTRLHx.BLOCK_TS).

Suspend transmission between blocks

At the end of each block transmission, a block end interrupt is asserted in the following cases:

1. Interrupt enable, CH_CTRLLx.INT_EN = 1
2. The channel block interrupt is unmasked, MASKBLOCK [n] = 1, where n is the channel number

For rows 2, 3, and 4 in Table 12-1 (SAx and/or DAx are automatically reloaded between block transmissions), if the block end interrupt is enabled and unmasked, the DMA transmission automatically stops after the block end interrupt.

The DMA does not proceed to the next block transfer until the hardware detects a write to the CLEARBLOCK [n] block interrupt clearing register (done by software to clear the channel block complete interrupt).

For rows 2, 3, and 4 in Table 12-1 (SAx and/or DAx are automatically reloaded between block transmissions), DMA transmissions do not stop if:

- Interrupt not enabled, CH_CTRLLx.INT_EN =0
- Channel block interrupts are masked, MASKBLOCK [n] = 0, where n is the channel number
- Suspend in block transfer completion interrupt not enabled, CH_CFGLx.INT _SUP=0

The channel pause between blocks is used to ensure that the end-of-block ISR (interrupt service routine) of the penultimate block is served before the last block starts. This ensures that the ISR has cleared the CH_CFGLx.RELOAD_SRC and/or CH_CFGLx.RELOAD_DST bits before completing the final block. For the second-to-last block transmission, the overload bits CH_CFGLx.RELOAD_SRC and/or CH_CFGLx.RELOAD_DST should be cleared in the block end ISR of the second-to-last block.

End Multiblock Transmission

All multi-block transmissions must end in line 1 of Table 12-1. At the end of each block transmission, the DMA will sample the row number. If the DMA is in the row 1 state, the previous block transmitted is the last block and the DMA transmission terminates.

Forrows 2, 3, and 4 of Table 12-1 (CH_CFGLx.RELOAD_SRC and/or CH_CFGLx.RELOAD_DST settings), the multi-block DMA transfer continues until the CH_CFGLx.RELOAD_SRC and CH_CFGLx.RELOAD_DST registers are cleared by the software. They should be programmed to 0 in the end of block interrupt service routine that serves the penultimate block transfer, which will put the DMA into the state of the first line.

12.2.9.3 Programming channel

Two registers, CH_CTRLx and CH_CFGx, need to be programmed to determine whether a single block transfer or a multi-block transfer occurs, and which type of multi-block transfer is used. Transmission types are shown in Table 12-1.

Monoblock Transfer (Line 1)

1. Read the channel enable register to select a free channel, refer tothe CH_ENregister.
2. Clear any pending interrupts on channels in previous DMA transmissions by writing interrupt clearing registers: CLEARTFR, CLEARBLOCK, CLEARSRCTRAN, CLEARDSTTRAN,

CLEARERR, and CLEARHALFBLOCK. Reading the interrupt raw status and interrupt status registers confirms that all interrupts have been cleared.

3. Program the following channel registers:
   a) Write the starting source address of channel x in the SAx register.
   b) Write the start destination address of channel x in the DAx register.
   c) Program CH_CTRLx and CH_CFGx as per row1 of Table12-1.
   d) The control information of the DMA transfer is written in the CH_CTRLx register of channel x. For example, the following can be programmed in the register:
      i. The transfer type (memory or non-memory peripheral of source and destination) is set by programming the TT of the CH_CTRLLx register.
      ii. Set transfer properties, such as:
         ✧ Transmission width of source in SRC_TR_WIDTH field
         ✧ Transmission width of the target in theDST_ TR_WIDTH field
         ✧ Incremental/decremental or fixed address of source in SINC field
         ✧ Increment/decrement or fixed address of the target in the DINC field
   e) Write channel configuration information to the CH_CFGx register of channel x
      i. Specify the handshake interface type (hardware or software) of the source peripheral and the target peripheral, which is not required for the memory. This step requires programming the HS_SEL_SRC/HS_SEL_DST bits separately. Write 0 activates the hardware handshake interface to handle source/destination requests. Write 1 activates the software handshake interface to handle source and destination requests.
      ii. If the hardware handshake interface is activated for the source or destination peripheral, the handshake interface is assigned to the source and destination peripherals, which requires programming of the SRC_PER and DST_PER bits, respectively.
4. Before writing toCH_EN, make sure theDMA_ENregister is enabled.
5. To transmit a data block (assuming a non-memory peripheral), the source and target request a single and burst DMA transaction. The DMA acknowledges and performs block transmission upon completion of each transaction (burst and single) in the block.
6. Once the transfer is complete, the hardware sets the interrupt and disables the channel. At this point, you can respond to block transfer completion or transfer completion interrupt, or poll the transfer completion raw interrupt status register (RAWTFR [n], n = channel number) until it is set by hardware in order to detect when the transfer is complete. Note that if this kind of polling is used, the software must ensure that the transfer completion interrupt is cleared by writing the interrupt clearance register CLEARTFR [n] before the channel is enabled.

Multi-block transmission with source address auto-reloading and destination address auto-reloading (line 4)

1. Read the channel enable register to select a free channel, refer tothe CH_ENregister.

2. Clear any pending interrupts on channels in previous DMA transmissions by writing interrupt clearing registers: CLEARTFR, CLEARBLOCK, CLEARSRCTRAN, CLEARDSTTRAN, CLEARERR, and CLEARHALFBLOCK. Reading the interrupt raw status and interrupt status registers confirms that all interrupts have been cleared.
3. Program the following channel registers:
   a) Write the starting source address of channel x in the SAx register.
   b) Write the start destination address of channel x in the DAx register.
   c) Program CH_CTRLx and CH_CFGx as perline4of Table12-1.
   d) The control information of the DMA transfer is written in the CH_CTRLx register of channel x. For example, the following can be programmed in the register:
      i. By programming the TT of the CH_CTRLx register, the transfer type (memory or non-memory peripherals of the source and destination) and the flow control device are set.
      ii. Set transfer properties, such as:
         ✧ Transmission width of source in SRC_TR_WIDTH field
         ✧ Transmission width of the target in theDST_ TR_WIDTH field
         ✧ Incremental/decremental or fixed address of source in SINC field
         ✧ Increment/decrement or fixed address of the target in the DINC field
   e) Write channel configuration information into the CH_CFGx register of channel x. Ensure that CH_CFGLx.RELOAD_SRC and CH_CFGLx.RELOAD_DST are enabled.
      i. Specifies the handshake interface type (hardware or software) of the source and destination peripherals, which is not required for the memory. This step requires programming the HS_SEL_SRC/HS_SEL_DST bits separately. Write 0 activates the hardware handshake interface to handle source/destination requests. Write 1 activates the software handshake interface to handle source and destination requests.
      ii. If the hardware handshake interface is activated for the source or destination peripheral, the handshake interface is assigned to the source and destination peripherals, which requires programming of the SRC_PER and DST_PER bits, respectively.
4. Before writing toCH_EN, make sure theDMA_ENregister is enabled.
5. To transmit a data block (assuming a non-memory peripheral), the source and target request a single and burst DMA transaction. The DMA acknowledges and performs block transmission upon completion of each transaction (burst and single) in the block.
6. When block transfer is complete,the DMAreloads the SAx, DAx, and CH_CTRLx registers. The hardware setup block completes the interrupt. The DMA then samples the row numbers as shown in Table 12-1. If the DMA is in line 1, then the DMA transfer has completed. The hardware sets the transfer completion interrupt and disables the channel. You can respond to a block transfer completion or a transfer completion interrupt, or poll the transfer completion raw interrupt status register (RAWTFR [n], where n is the channel number) until it is set by hardware

in order to detect when the transfer is complete. Note that if this kind of polling is used, the software must ensure that the transfer completion interrupt is cleared by writing the interrupt clearance register CLEARTFR [n] before the channel is enabled. If the DMA is not in line 1, the next step is performed.

7. The DMA transfer process is as follows:
   a) If the interrupt is enabled (CH_CTRLLx.INT_EN = 1), and the block completion interrupt is unmasked (MASKBLOCK[x] = 1'b1, where x is the channel number), the hardware sets the block completion interrupt when the block transfer is complete. It then stops running until the block transfer completion interrupt is cleared by the software. If the next block is the last block in the DMA transfer, the block completion ISR (interrupt service routine) should clear the reload registers CH_CFGLx.RELOAD_SRC and CH_CFGLx.RELOAD_DST in CH_CFGLx. This puts the DMA on row 1, as shown in Table 12-1. If the next block is not the last block in the DMA transfer, then the reload register should remain enabled to keep DMA on line 4.
   b) If the interrupt is disabled (CH_CTRLLx.INT_EN = 0) or the block completion interrupt is masked (MASKBLOCK[x] =1 'b0, where x is the channel number), then the hardware does not stop until the write block transfer completion interrupt clear register is detected; Instead, it starts the next block transfer immediately. In this case, the software must clear the reload registers CH_CFGLx.RELOAD_SRC and CH_CFGLx.RELOAD_DST in CH_CFGLxbefore the last block of the DMA transfer is complete, thereby putting the DMAin line 1 ofTable12-1.

Multiblock transmission with source address automatic reload and destination address contiguity (line 3)

1. Read the channel enable register to select a free channel, refer tothe CH_ENregister.
2. Clear any pending interrupts on channels in previous DMA transmissions by writing interrupt clearing registers: CLEARTFR, CLEARBLOCK, CLEARSRCTRAN, CLEARDSTTRAN, CLEARERR, and CLEARHALFBLOCK. Reading the interrupt raw status and interrupt status registers confirms that all interrupts have been cleared.
3. Program the following channel registers:
   a) Write the starting source address of channel x in the SAx register.
   b) Write the start destination address of channel x in the DAx register.
   c) Program CH_CTRLx and CH_CFGx as per row3ofTable12-1.
   d) The control information of the DMA transfer is written in the CH_CTRLx register of channel x. For example, the following can be programmed in the register:
      i. The transfer type (memory or non-memory peripheral of source and destination) is set by programming the TT of the CH_CTRLLx register.
      ii. Set transfer properties, such as:
         ✧ Transmission width of source in SRC_TR_WIDTH field

- ✧ Transmission width of the target in theDST_ TR_WIDTH field
- ✧ Incremental/decremental or fixed address of source in SINC field
- ✧ Increment/decrement or fixed address of the target in the DINC field

e) Write channel configuration information into the CH_CFGx register of channel x. Ensure that CH_CFGLx.RELOAD_SRC is enabled and CH_CFGLx.RELOAD_DST is disabled.

i. Specify the handshake interface type (hardware or software) of the source peripheral and the target peripheral, which is not required for the memory.

This step requires programming the HS_SEL_SRC/HS_SEL_DST bits separately. Write 0 activates the hardware handshake interface to handle source/destination requests. Write 1 activates the software handshake interface to handle source and destination requests.

ii. If the hardware handshake interface is activated for the source or destination peripheral, the handshake interface is assigned to the source and destination peripherals, which requires programming of the SRC_PER and DST_PER bits, respectively.

4. Before writing toCH_EN, make sure theDMA_ENregister is enabled.
5. To transmit a data block (assuming a non-memory peripheral), the source and target request a single and burst DMA transaction. The DMA acknowledges and performs block transmission upon completion of each transaction (burst and single) in the block.
6. When the block transfer is complete, the DMA reloads the SAx register and the DAx register remains unchanged. The hardware setup block completes the interrupt. The DMA then samples the row numbers as shown in Table 12-1. If the DMA is in line 1, then the DMA transfer has completed. The hardware sets the transfer completion interrupt and disables the channel. You can respond to a block transfer completion or a transfer completion interrupt, or poll the transfer completion raw interrupt status register (RAWTFR [n], n = channel number) until it is set by hardware in order to detect when the transfer is complete. Note that if this kind of polling is used, the software must ensure that the transfer completion interrupt is cleared by writing the interrupt clearance register CLEARTFR [n] before the channel is enabled. If the DMA is not in line 1, the next step is performed.
7. The DMA transfer process is as follows:

a) If the interrupt is enabled (CH_CTRLLx.INT_EN = 1), and the block completion interrupt is unmasked (MASKBLOCK[x] = 1'b1, where x is the channel number), the hardware sets the block completion interrupt upon completion of the block transfer. It then stops running until the block completion interrupt is cleared by the software. If the next block is the last block in the DMA transfer, the block completion ISR (interrupt service routine) should clear the source reload register CH_CFGLx.RELOAD_SRC. This puts the DMA on row 1, as shown in Table 12-1. If the next block is not the last block of the DMA transfer, then the source reload register should remain enabled to keep the DMA in line 3, as shown in Table 12-1.

b) If the interrupt is disabled (CH _CTRLLx.INT_EN = 0) or the block completion interrupt is masked (MASKBLOCK [x] = 1'b0, where x is the channel number), then the hardware does

not stop until it detects the write block transfer completion interrupt clears the register, instead it starts the next block transfer immediately. In this case, the software must clear the source reload register CH_CFGLx.RELOAD_SRC before the last block of the DMA transfer is completed, thereby putting the DMAin line 1 ofTable12-1.

12.2.9.4 Disable channels until transfer completes

Under normal operation, the software enables the channel bywriting 1 to the channel enable register CH_EN, and the hardware disables the channel by clearing CH_EN upon completion of the transfer.

The recommended method for software to disable a channel without losing data is to use a combination of the CH_SUSP bit and the FIFO_EMPTY bit in the channel configuration register (CH_CFGLx).

1. If the software wishes to disable the channel before the DMA transfer is complete, then it can set the CH_CFGLx.CH_SUSP bit to tell the DMAtostop all transfers from the source peripheral. Therefore, the channel FIFO cannot receive new data.
2. The software may poll the CH_CFGLx.FIFO_EMPTYbit until it indicates that the channel FIFO is empty.
3. Once the channel FIFO is empty,the CH_EN bitcan be cleared by the software.

When CH_CTRLLx.SRC_TR_WIDTH < CH_CTRLLx.DST_TR_WIDTH and the CH_CFGLx.CH_SUSP bit is high, CH_CFGLx.FIFO_EMPTY is set once the data in the FIFO cannot form the unit length of CH_CTRLLx.DST_TR_WIDTH. However, there may still be data in the channel FIFO, but not enough to form a single transmission of CH_CTRLLx.DST_TR_WIDTH. In this case, the remaining data in the channel FIFO is not transferred to the target peripheral once the channel is disabled.

Allows the suspended state of the channel to be released by writing 0 to the CH_CFGLx.CH_SUSP register. The DMA transfer is done in the normal manner.

Abnormal transmission aborted

A DMA transmission may be abruptly terminated by software by clearingthe channel enable bit CH_EN. It cannot be assumed that the channel is disabled immediately after setting CH_EN. The CH_EN bit is cleared on the AHB slave interface. Considering that this is considered a request to disable the channel, you must confirm that the channel is disabled by reading back CH_ENto 0. The case where the channel is not disabled after the channel disabling request is that the source or target receives a split or retry response. The DMA must continue to reattempt the transmission to the system HADDR that originally received the split or retry response until an OKAY response is returned, otherwise the AMBA protocol is violated.

The software canabruptly terminate all channels by clearing the global enable bit in the DMA configuration register(DMA_ EN). Similarly, it cannot be assumed that all channels are disabled immediately after clearing DMA_EN[0]. DMA_EN[0] is cleared on the AHB slave interface. Consider that this is treated as a request to disable all channels. You must confirm that all channels are disabledby reading back CH_ENto 0.

If the channel enable bit is cleared when there is data in the channel FIFO, the data is not sent to the target peripheral and does not exist when the channel is re-enabled. For read-sensitive source peripherals, such as source FIFOs, these data are thus lost. When the source is not a read sensitive device, such as memory, it may be acceptable to disable the channel without waiting for the channel FIFO to clear, because data can be obtained from the source peripheral upon request and is not lost.

If the channel is disabled by the software, there is no guarantee that an active single or burst transaction will receive an acknowledgement.

It is not allowed to disable the channel by software before completing the transfer. Clearing the CH_EN bit before the channel is suspended may violate the AHB protocol.

12.3 TIMx registers

12.3.1 DMA module enable register (DMA_EN)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															DMA_EN
															RW

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	DMA_EN	RW	0	DMA enabled 0: Disabled 1: Enabled

12.3.2 DMA channel enable register (DMA_CH_EN)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		CH_EN_WE						Res.		CH_EN					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	CH_EN_WE	W	6'h0	Write enable for channel enabled (one channel per bit, e.g. CH_EN_WE [0] for CH0) 0: Disabled 1: Enabled
7: 6	Reserved	-	-	-
5: 0	CH_EN	RW	6'h0	Channel enable (one channel per bit, e.g. CH_EN [0] for CH0) 0: Disabled 1: Enabled CH_EN and CH_EN_WE have one-to-one correspondence, CH_EN and CH_EN_WE must be written simultaneously, and CH_EN can be written only when CH_EN_WE is equal to 1; When DMA_EN is 0, the register is hardware cleared; The register is cleared by hardware after the last data transfer to the target address is completed;

12.3.3 DMA Control Register (DMA_CH_CTRLLx)

Address offset: 0x10 + x * 0x58 (x is channel number)

Reset value: 0x0030 4801

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.										TT		Res.			SRC_MSIZE
										RW	RW				RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRC_MSIZE		DST_MSIZE			SINC		DINC		SRC_TR_WIDTH			DST_TR_WIDTH			Int_en
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 20	TT	RW	2'h3	Transmission Type 00: mem to mem transmission 01: mem to peripheral transfer 10: Peripheral to mem transfer 11: Peripheral-to-peripheral transfer
19: 17	Reserved	-	-	-
16: 14	SRC_MSIZE	RW	3'h1	Source burst transaction length 3 'h0: 1 3 'h1: 4 3 'h2: 8 3 'h3: 16 3 'h4: 32 3 'h5: 64 3 'h6: 128 other: forbidden The number of data, the width of each data bit is determined by SRC_TR_WIDTH
13: 11	DST_MSIZE	RW	3'h1	Target burst transaction length 3 'h0: 1 3 'h1: 4 3 'h2: 8 3 'h3: 16 3 'h4: 32 3 'h5: 64 3 'h6: 128 other: forbidden The number of data, the width of each data bit is determined by DST_TR_WIDTH
10: 9	SINC	RW	2'h0	Source address increment 00: Increment 01: Decrease 10: unchanged 11: unchanged
8: 7	DINC	RW	2'h0	Target Address Increment 2 'h0: incremental 2 'h1: decreasing 2 'h2: unchanged 2 'h3: unchanged
6: 4	SRC_TR_WIDTH	RW	3'h0	Source transmission bit width 3 'h0: 8 bits 3 'h1: 16 bits 3 'h2: 32 bits other: forbidden
3:1	DST_TR_WIDTH	RW	3'h0	Target transmission bit width 3 'h0: 8 bits 3 'h1: 16 bits 3 'h2: 32 bits other: forbidden
0	Int_en	RW	1	the interrupt enable 0: Disabled 1: Enabled

12.3.4 DMA Control Register (DMA_CH_CTRLHx)

Address offset: 0x14 + x * 0x58 (x is channel number)

Reset value: 0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				BLOCK_TS											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	BLOCK_TS	RW	12'h2	Block transfer size Read this register after the start of the transfer is the total number of data items that have been read from the source address

12.3.5 DMA Configuration Register (DMA_CH_CFGLx)

Address offset: 0x18 + x * 0x58 (x is channel number)

Reset value: 0x0000 0e (x < < 1) 0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RE-LoAD_DST	RE-LoAD_SRC	MAX_ABRST										Res.			
RW	RW	RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				HS_SEL_SRC	HS_SEL_DST	FIFO_EMPTY	CH_SUSP	CH_PRIOR			INT_SUSP	Res.			
				RW	RW	R	RW	RW			RW				

Bit	Name	R/W	Reset Value	Function
31	RELoAD_DST	RW	0	Target auto-reload 0: Disabled 1: Enabled The DA register is automatically reloaded to the initial value at the end of each block transfer of a multi-block transfer
30	RELoAD_SRC	RW	0	Source Automatic Reload 0: Disabled 1: Enabled The SA register is automatically reloaded to the initial value at the end of each block transfer of a multi-block transfer
29: 20	MAX_ABRST	RW	10'h0	Maximum burst length 0: maximum burst length is not limited other: corresponding to the set burst length
19: 12	Reserved	-	-	-
11	HS_SEL_SRC	RW	1	Source software or hardware handshake selection 0: Hardware handshake 1: Software handshake This register does not work when the source is memory
10	HS_SEL_DST	RW	1	Target software or hardware handshake selection 0: Hardware handshake 1: Software handshake This register does not work when the target is memory
9	FIFO_EMPTY	R	1	Channel fifo state 0: non-empty 1: empty
8	CH_SUSP	RW	0	Channel suspended

				0: DMA transfer from source will not be suspended 1: DMA transmission from source is suspended
7: 5	CH_PRIOR	RW	3 'hx	Channel Priority Default is channel number The higher the value of this register, the higher the priority The default priority of each channel is the channel number, that is Channel 0 priority is 0 Channel 1 priority is 1 Channel 2 priority is 2 Channel 3 priority is 3 Channel 4 priority is 4 Channel 5 priority is 5
4	INT_SUSP	RW	1	Suspended in block interrupt (multi-block transmission) 0: Continue transmission in block interrupt 1: Suspend transmission in block interrupt
3: 0	Reserved	-	-	-

12.3.6 DMA configuration register (DMA_CH_CFGHx)

Address offset: 0x1c + x * 0x58 (x is channel number)

Reset value: 0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		DST_PER			Res.	SRC_PER			Res.					FIFO_MODE	Res.
		RW	RW	RW		RW	RW	RW						RW	

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 11	DST_PER	RW	3'h0	Target hardware interface selection Allocate the request to the corresponding channel, that is 3 'h0: assign target request 0 to channel x 3 'h1: assign target request 1 to channel x 3 'h2: assign target request 2 to channel x 3 'h3: assign target request 3 to channel x 3 'h4: assign target request 4 to channel x 3 'h5: assign target request 5 to channel x other: forbidden
10	Reserved	-	-	-
9: 7	SRC_PER	RW	3'h0	Source hardware interface selection Allocate the request to the corresponding channel, that is 3 'h0: Assign source request 0 to channel x 3 'h1: Assign source request 1 to channel x 3 'h2: Assign source request 2 to channel x 3 'h3: Assign source request 3 to channel x 3 'h4: Assign source request 4 to channel x 3 'h5: Assign source request 5 to channel x other: forbidden
6: 2	Reserved	-	-	-
1	FIFO_MODE	RW	0	fifo mode selection 0: Single AHB transmission with available space of specified transmission width 1: For the target transmission, the available data is greater than or equal to half of the FIFO depth; For source transmission, the available space is greater than half the FIFO depth. There are exceptions at the end of a burst request or at the end of a block transfer.
0	Reserved	-	-	-

12.3.7 DMA channel x source address register (DMA_SAx)

Address offset: 0x20 + x * 0x58 (x is channel number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SA [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SA [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	SA [31: 0]	RW	32'h0	Current source address of DMA transfer

12.3.8 DMA channel x destination address register (DMA_DAx)

Address offset: 0x28 + x * 0x58 (x is channel number)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DA [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DA [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DA [31: 0]	RW	32'h0	Current destination address of DMA transfer

12.3.9 DMAHalf Block TransferInterrupt Register (DMA_STATUSHALFBLOCK)

Address offset: 0x2a8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	STATUS	R	6'h0	The state of half-block transmission completion interruption (each bit corresponds to one channel, for example, STATUS [0] corresponds to CH0) (MASKHALFBLOCK = 1 is valid) 0: Inactive state 1: Active status

12.3.10 DMAHalf Block TransferInterruptMaskRegister (DMA_MASKHALFBLOCK)

Address offset: 0x2b0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Half-block transmission completes write enable of interrupt mask

				(Each bit corresponds to one channel, e.g. INT_MASK_WE [0] corresponds to CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Half-block transmission completes interrupt masking (each bit corresponds to one channel, for example INT_MASK [0] corresponds to CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK corresponds to INT_MASK_WE one-to-one, and this register can only be written when the corresponding INT_MASK_WE is equal to 1

12.3.11 DMAHalf Block Transfer CompleteInterruptClearRegister (DMA_CLEARHALFBLOCK)

Address offset: 0x2b8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR half-block transfer completion interrupt (each bit corresponds to one channel, for example, CLEAR [0] corresponds to CH0) 0: No action 1: Clear interrupt

12.3.12 DMAtransfer completioninterrupt register (DMA_STATUSTFR)

Address offset: 0x2e8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	STATUS	R	6'h0	The state of transmission completion interrupt (each bit corresponds to one channel, for example, STATUS [0] corresponds to CH0) (MASKTFR = 1 is valid) 0: Inactive state 1: Active status

12.3.13 DMAblock transfer completioninterrupt register (DMA_STATUSBLOCK)

Address offset: 0x2f0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-

5: 0	STATUS	R	6'h0	The state of block transmission completion interruption (each bit corresponds to one channel, for example, STATUS [0] corresponds to CH0) (MASKBLOCK = 1 is valid) 0: Inactive state 1: Active status

12.3.14 DMASource Transaction CompleteInterrupt Register (DMA_STATUSSRCTRAN)

Address offset: 0x2f8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	STATUS	R	6'h0	The state of the source transaction completion interrupt (one channel per bit, for example, STATUS [0] corresponds to CH0) (MASKSRCTRAN = 1 is valid) 0: Inactive state 1: Active status

12.3.15 DMATarget Transaction CompleteInterrupt Register (DMA_STATUSDSTTRAN)

Address offset: 0x300

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	STATUS	R	6'h0	The state of the target transaction completion interrupt (each bit corresponds to a channel, for example, STATUS [0] corresponds to CH0) (MASKDSTTRAN = 1 is valid) 0: Inactive state 1: Active status

12.3.16 DMAtransfer errorinterrupt register (DMA_STATUSERR)

Address offset: 0x308

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										STATUS					
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	STATUS	R	6'h0	STATUS of transmission error interrupt (one channel per bit, e.g. STATUS [0] for CH0) (MASKERR = 1 valid) 0: Inactive state 1: Active status

12.3.17 DMAtransfer completioninterruptmaskregister (DMA_MASKTFR)

Address offset: 0x310

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Write Enable for Transfer Completion Interrupt Mask (Each bit corresponds to one channel, e.g. INT_MASK_WE [0] corresponds to CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Transmission completion interrupt masking (each bit corresponds to one channel, e.g. INT_MASK [0] corresponds to CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK and INT_MASK_WE have a one-to-one correspondence. INT_MASK and INT_MASK_WE must be written at the same time, and INT_MASK can only be written when INT_MASK_WE is equal to 1

12.3.18 DMAblock transfer completioninterruptmaskregister (DMA_MASKBLOCK)

Address offset: 0x318

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Block transfer completes write enable of interrupt mask (one channel per bit, e.g. INT_MASK_WE [0] corresponds to CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Block transmission completes interrupt masking (each bit corresponds to one channel, for example INT_MASK [0] corresponds to CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK and INT_MASK_WE have a one-to-one correspondence. INT_MASK and INT_MASK_WE must be written at the same time, and INT_MASK can only be written when INT_MASK_WE is equal to 1

12.3.19 DMASource Transaction CompletionInterruptMaskRegister (DMA_MASKSRCTRAN)

Address offset: 0x320

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Source transaction completes write enable of interrupt masking (one channel per bit, for example INT_MASK_WE [0] corresponds to CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Source transaction completes interrupt masking (one channel per bit, e.g. INT_MASK [0] corresponds to CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK and INT_MASK_WE have a one-to-one correspondence. INT_MASK and INT_MASK_WE must be written at the same time, and INT_MASK can only be written when INT_MASK_WE is equal to 1

12.3.20 DMATarget Transaction CompletionInterrupt MaskRegister (DMA_MASKDSTTRAN)

Address offset: 0x328

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Target transaction completes write enable of interrupt masking (each bit corresponds to one channel, for example INT_MASK_WE [0] corresponds to CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Target transaction completion interrupt masking (each bit corresponds to one channel, for example INT_MASK [0] corresponds to CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK and INT_MASK_WE have a one-to-one correspondence. INT_MASK and INT_MASK_WE must be written at the same time, and INT_MASK can only be written when INT_MASK_WE is equal to 1

12.3.21 DMATransfer ErrorInterruptMaskRegister (DMA_MASKERR)

Address offset: 0x330

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		INT_MASK_WE						Res.		INT_MASK					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	INT_MASK_WE	W	6'h0	Write enable for transmission error interrupt mask (one channel per bit, e.g. INT_MASK_WE [0] for CH0) 0: Interrupt mask write prohibition 1: Interrupt mask write enabled
7: 6	Reserved	-	-	-
5: 0	INT_MASK	RW	6'h0	Transmission error interrupt masking (one channel per bit, e.g. INT_MASK [0] for CH0) 0: Mask interrupt 1: Unmasked interrupt INT_MASK and INT_MASK_WE have a one-to-one correspondence. INT_MASK and INT_MASK_WE must be written at the same time, and INT_MASK can only be written when INT_MASK_WE is equal to 1

12.3.22 DMATransfer CompleteInterruptClearRegister (DMA_CLEARTFR)

Address offset: 0x338

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR transfer completion interrupt (one channel per bit, e.g. CLEAR [0] for CH0) 0: No action 1: Clear interrupt

12.3.23 DMABlock Transfer CompleteInterruptClearRegister (DMA_CLEARBLOCK)

Address offset: 0x340

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR block transfer completion interrupt (one channel per bit, e.g. CLEAR [0] for CH0) 0: No action 1: Clear interrupt

12.3.24 DMASourceTransactionComplete InterruptClear Register (DMA_CLEARSRCTRAN)

Address offset: 0x348

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR source transaction completion interrupts (one channel per bit, e.g. CLEAR [0] for CH0) 0: No action 1: Clear interrupt

12.3.25 DMATargetTransaction InterruptClear Register (DMA_CLEARDSTTRAN)

Address offset: 0x350

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR the target transaction completion interrupt (one channel per bit, for example, CLEAR [0] for CH0) 0: No action 1: Clear interrupt

12.3.26 DMATransfer ErrorInterruptClearRegister (DMA_CLEARERR)

Address offset: 0x358

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										CLEAR					
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	CLEAR	W	6'h0	CLEAR transmission error interrupts (one channel per bit, e.g. CLEAR [0] for CH0) 0: No action 1: Clear interrupt

12.3.27 DMAchannelinterruptstatusregister (DMA_STATUSINT)

Address offset: 0x360

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.										HALFBLOC	ERR	DSTT	SRCT	BLOCK	TFR
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-

5	HALFBLOC	R	0	Is there a half block transmission completion interrupt in all channels 0: There is no half-block transmission completion interrupt in all channels 1: Half-block transmission completion interrupt exists in all channels
4	ERR	R	0	Are there any transmission error interrupts in all channels 0: No transmission error interrupt in all channels 1: Transmission error interrupt in all channels
3	DSTT	R	0	Is there a target transmission completion interrupt in all channels 0: No target transmission completion interrupt exists in all channels 1: There is a target transmission completion interrupt in all channels
2	SRCT	R	0	Is there a source transfer completion interrupt in all channels 0: No source transmission completion interrupt exists in all channels 1: Source transfer completion interrupt exists in all channels
1	BLOCK	R	0	Is there a block transfer completion interrupt in all channels 0: No block transfer completion interrupt exists in all channels 1: There is a block transfer completion interrupt in all channels
0	TFR	R	0	Are there transmission completion interrupts in all channels 0: No transmission completion interrupt exists in all channels 1: Transmission completion interrupt exists in all channels

12.3.28 DMASource TransactionSoftware Handshake Register (DMA_REQSRC)

Address offset: 0x368

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		SRC_REQ_WE						Res.		SRC_REQ					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	SRC_REQ_WE	W	6'h0	Write enable for source transaction software requests (one channel per bit, e.g. SRC_REQ_WE [0] for CH0) 0: Source request write prohibited 1: Source request write enable
7: 6	Reserved	-	-	-
5: 0	SRC_REQ	RW	6'h0	Source transaction software request(one channel per bit, e.g. SRC_REQ [0] for CH0) 0: Source request not active 1: Source request activation When CH_EN is 0, the register is hardware cleared; SRC_REQ corresponds to SRC_REQ_WE one-to-one, and this register can be written only when the corresponding SRC_REQ_WE is equal to 1

12.3.29 DMATarget TransactionSoftware Handshake Register (DMA_REQDST)

Address offset: 0x370

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		DST_REQ_WE						Res.		DST_REQ					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	DST_REQ_WE	W	6'h0	Write enable for the target transaction software request(one channel per bit, e.g. DST_REQ_WE [0] for CH0) 0: Destination request write prohibited 1: Target request write enabled
7: 6	Reserved	-	-	-
5: 0	DST_REQ	RW	6'h0	Target transaction software request(one channel per bit, e.g. DST_REQ [0] corresponds to CH0) 0: Destination request not active 1: Target request activation When CH_EN is 0, the register is hardware cleared; DST_REQ corresponds to DST_REQ_WE one-to-one, and this register can be written only when the corresponding DST_REQ_WE is equal to 1

12.3.30 DMAsource single transactionsoftware handshake register (DMA_SGLRQSRC)

Address offset: 0x378

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		SRC_SGLREQ_WE						Res.		SRC_SGLREQ					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	SRC_SGLREQ_WE	W	6'h0	Write enable for source single transaction software requests(one channel per bit, e.g. SRC_SGLREQ_WE [0] for CH0) 0: Source request write prohibited 1: Source request write enable
7: 6	Reserved	-	-	-
5: 0	SRC_SGLREQ	RW	6'h0	Source single transaction software request(each bit corresponds to one channel, for example, SRC_SGLREQ [0] corresponds to CH0) 0: Source request not active 1: Source request activation When CH_EN is 0, the register is hardware cleared; SRC_SGLREQ corresponds to SRC_SGLREQ_WE one-to-one, and this register can be written only when the corresponding SRC_SGLREQ_WE is equal to 1

12.3.31 DMATarget Single TransactionSoftware Handshake Register (DMA_SGLRQDST)

Address offset: 0x380

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		DST_SGLREQ_WE						Res.		DST_SGLREQ					
		W	W	W	W	W	W			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 8	DST_SGLREQ_WE	W	6'h0	Write enable of target single transaction software request(one channel per bit, e.g. DST_SGLREQ_WE [0] corresponds to CH0) 0: Destination request write prohibited 1: Target request write enabled
7: 6	Reserved	-	-	-
5: 0	DST_SGLREQ	RW	6'h0	Target single transaction software request(each bit corresponds to one channel, for example, DST_SGLREQ [0] corresponds to CH0) 0: Destination request not active 1: Target request activation When CH_EN is 0, the register is hardware cleared; DST_SGLREQ corresponds to DST_SGLREQ_WE one-to-one, and this register can be written only when the corresponding DST_SGLREQ_WE is equal to 1

13. Interrupts and events

13.1 Nested vectored interrupt controller (NVIC)

The interface between nested vector interrupt controller (NVIC) and processor core is closely connected, which can realize low latency interrupt processing and efficiently handle late interrupts.

The nested vector interrupt controller manages interrupts such as kernel exceptions. Please refer to the Cortex-M4F Programming Manual for more instructions on exceptions and NVIC programming.

13.1.1 Main features

- 86 maskable interrupt channels (excluding 16 Cortex ® -M4F interrupt line);
- 8 programmable priority levels (3-bit interrupt priority is used);
- Low latency exception and interrupt handling;
- Power management control;
- Implementation of System Control Register;

13.1.2 SysTick Calibration Value Register

The system tick calibration value is fixed at 9000, and when the system tick clock is set to 9MHz, a 1ms time reference is generated.

13.1.3 Break and Exception Vector Table

Vector numbering	Priority	Type of priority	Pin name	Address	Description
-	-	-	-	0x0000_0000	Reserved
-	-3	fixed	Reset	0x0000_0004	Reset
-	-2	NMI	RCC HSE Clock Safety (HSE CSS)/ECC error for FLASH/CCM error for SRAM	0x0000_0008	NMI
-	-1	fixed	HardFault	0x0000_000C	All classes of fault
-	0	Programmable	MemManage (MemManage)	0x0000_0010	Memory management
-	1	Programmable	BusFault	0x0000_0014	Prefetch finger failure, memory access failure
-	2	Programmable	Error Application (UsageFault)	0x0000_0018	Undefined instruction or illegal status
-	-	-	-	0x0000_001C	Reserved
-	-	-	-	0x0000_0020	Reserved
-	-	-	-	0x0000_0024	Reserved
-	-	-	-	0x0000_0028	Reserved
-	3	Programmable	SVCall	0x0000_002C	System service call via SWI instruction
-	4	Programmable	DebugMonitor	0x0000_0030	Debug Monitor
-	-	-	-	0x0000_0034	Reserved
-	5	Programmable	PendSV	0x0000_0038	Pendable request for system service
-	6	Programmable	SysTick	0x0000_003C	System tick timer
0	7	Programmable	WWDG	0x0000_0040	Window Watchdog Timer Interrupt
1	8	Programmable	PVD	0x0000_0044	Power Supply Voltage Detection (PVD) Interruption to EXTI
2	9	Programmable	TAMPER	0x0000_0048	Intrusion detection interruption
3	10	Programmable	RTC	0x0000_004C	Real Time Clock (RTC) Global Interrupt
4	11	Programmable	FMC	0x0000_0050	Flash FMC Global Interrupt

5	12	Programmable	RCC and CTC	0x0000_0054	Reset and Clock Control (RCC) and CTC interrupts
6	13	Programmable	EXTI0	0x0000_0058	EXTI line 0 interrupted
7	14	Programmable	EXTI1	0x0000_005C	EXTI line 1 interrupted
8	15	Programmable	EXTI2	0x0000_0060	EXTI line 2 interrupted
9	16	Programmable	EXTI3	0x0000_0064	EXTI line 3 interrupted
10	17	Programmable	EXTI4	0x0000_0068	EXTI line 4 interrupted
11	18	Programmable	DMA1 Channel 1	0x0000_006C	DMA1 Channel 1 Global Interrupt
12	19	Programmable	DMA1 Channel 2	0x0000_0070	DMA1 Channel 2 Global Interrupt
13	20	Programmable	DMA1 Channel 3	0x0000_0074	DMA1 Channel 3 Global Interrupt
14	21	Programmable	DMA1 Channel 4	0x0000_0078	DMA1 Channel 4 Global Interrupt
15	22	Programmable	DMA1 Channel 5	0x0000_007C	DMA1 Channel 5 Global Interrupt
16	23	Programmable	DMA1 Channel 6	0x0000_0080	DMA1 Channel 6 Global Interrupt
17	24	Programmable	DMA2 Channel 6	0x0000_0084	DMA2 Channel 6 Global Interrupt
18	25	Programmable	ADC1_2	0x0000_0088	ADC1 and ADC2 global interrupts
19	26	Programmable	OTG1_FS-WKUP	0x0000_008C	USB1 OTG FS wakeup via EXTI interrupt
20	27	Programmable	CAN1 int0	0x0000_0090	CAN1 Interrupt 0
21	28	Programmable	OTG1_FS	0x0000_0094	USB1 OTG Global interrupt
22	29	Programmable	TIM18	0x0000_0098	TIM18 global interrupt
23	30	Programmable	EXTI9_5	0x0000_009C	EXTI line [9: 5] interrupted
24	31	Programmable	TIM1_BRK_TIM9	0x0000_00A0	TIMER1 abort interrupt and TIMER9 global interrupt
25	32	Programmable	TIM1_UP_TIM10	0x0000_00A4	TIMER1 update interrupt and TIMER10 global interrupt
26	33	Programmable	TIM1_TRG_COM_TIM11	0x0000_00A8	TIMER1 Trigger and Communication Interrupt and TIMER11 Global Interrupt
27	34	Programmable	TIM1_CC	0x0000_00AC	TIMER1 capture comparison interrupt
28	35	Programmable	TIM2	0x0000_00B0	TIMER2 global interrupt
29	36	Programmable	TIM3	0x0000_00B4	TIMER3 global interrupt
30	37	Programmable	TIM4	0x0000_00B8	TIMER4 global interrupt
31	38	Programmable	I2C1_EV	0x0000_00BC	I2C1 Event Interrupt
32	39	Programmable	I2C1_ER	0x0000_00C0	I2C1 Error Interrupt
33	40	Programmable	I2C2_EV	0x0000_00C4	I2C2 Event Interrupt
34	41	Programmable	I2C2_ER	0x0000_00C8	I2C2 Error Interrupt
35	42	Programmable	SPI1	0x0000_00CC	SPI1 global interrupt
36	43	Programmable	SPI2	0x0000_00D0	SPI2 global interrupt
37	44	Programmable	USART1	0x0000_00D4	USART1 global interrupt
38	45	Programmable	USART2	0x0000_00D8	USART2 global interrupt
39	46	Programmable	USART3	0x0000_00DC	USART3 global interrupt
40	47	Programmable	EXTI15_10	0x0000_00E0	EXTI line [15:10] interrupted
41	48	Programmable	RTCAlarm	0x0000_00E4	RTC alarm clock connected to EXTI is interrupted
42	49	Programmable	TIM19	0x0000_00E8	TIM19 global interrupt
43	50	Programmable	TIM8_BRK_TIM12	0x0000_00EC	TIMER8 abort interrupt and TIMER12 global interrupt
44	51	Programmable	TIM8_UP_TIM13	0x0000_00F0	TIMER8 update interrupt and TIMER13 global interrupt
45	52	Programmable	TIM8_TRG_COM_TIM14	0x0000_00F4	TIMER8 Trigger and communication interrupt TIMER14 Global interrupt
46	53	Programmable	TIM8_CC	0x0000_00F8	TIMER8 capture comparison interrupt
47	54	Programmable	ADC3	0x0000_00FC	ADC3 global interrupt
48	55	Programmable	ESMC	0x0000_0100	ESMC global interrupt
49	56	Programmable	SDIO	0x0000_0104	SDIO global interrupt
50	57	Programmable	TIM5	0x0000_0108	TIM5 global interrupt

51	58	Programmable	SPI3	0x0000_010C	SPI3 global interrupt
52	59	Programmable	UART1	0x0000_0110	UART global interrupt
53	60	Programmable	LPUART	0x0000_0114	LPUART global and wake-up interrupts
54	61	Programmable	TIM6	0x0000_0118	TIM6 global interrupt
55	62	Programmable	TIM7	0x0000_011C	TIM7 global interrupt
56	63	Programmable	DMA2 Channel 1	0x0000_0120	DMA2 Channel 1 Global Interrupt
57	64	Programmable	DMA2 Channel 2	0x0000_0124	DMA2 Channel 2 Global Interrupt
58	65	Programmable	DMA2 Channel 3	0x0000_0128	DMA2 Channel 3 Global Interrupt
59	66	Programmable	DMA2 Channel 4_5	0x0000_012C	DMA2 Channel 4 and Channel 5 Global Interrupts
60	67	Programmable	TIM15	0x0000_0130	TIMER15 global interrupt
61	68	Programmable	TIM16	0x0000_0134	TIMER16 global interrupt
62	69	Programmable	TIM17	0x0000_0138	TIMER17 global interrupt
63	70	Programmable	I2C3_EV	0x0000_013C	I2C3 Event Interrupt
64	71	Programmable	I2C3_ER	0x0000_0140	I2C3 Error Interrupt
65	72	Programmable	I2C4_EV	0x0000_0144	I2C4 Event Interrupt
66	73	Programmable	I2C4_ER	0x0000_0148	I2C4 Error Interrupt
67	74	Programmable	ETH	0x0000_014C	Ethernet global interrupt
68	75	Programmable	ETH_WKUP	0x0000_0150	Ethernet wake-up interrupt
69	76	Programmable	RNG	0x0000_0154	RNG global interrupt
70	77	Programmable	AES	0x0000_0158	AES global interrupt
71	78	Programmable	CORDIC	0x0000_015C	CORDIC global interrupt
72	79	Programmable	DAC	0x0000_0160	DAC global interrupt
73	80	Programmable	COMP1	0x0000_0164	COMP1 global interrupt
74	81	Programmable	COMP2	0x0000_0168	COMP2 global interrupt
75	82	Programmable	COMP3	0x0000_016C	COMP3 global interrupt
76	83	Programmable	COMP4	0x0000_0170	COMP4 global interrupt
77	84	Programmable	UART2	0x0000_0174	UART2 global interrupt
78	85	Programmable	UART3	0x0000_0178	UART3 global interrupt
79	86	Programmable	Reserved	0x0000_017C	-
80	87	Programmable	LPTIM	0x0000_0180	LPTIM global interrupt
81	88	Programmable	OTG2_FS-WKUP	0x0000_0084	USB2 OTG FS wake-up via EXTI interrupt
82	89	Programmable	OTG2_FS	0x0000_0088	USB2 OTG Global interrupt
83	90	Programmable	CAN1 int1	0x0000_008C	CAN1 Interruption 1
84	91	Programmable	CAN2 int0	0x0000_0090	CAN2 Interrupt 0
85	92	Programmable	CAN2 int1	0x0000_0094	CAN2 Interrupt 1

13.2 External Extended Interrupt/Event Controller (EXTI)

The extended interrupt and event controller manages CPU and system wake-up functions through configurable lines and direct input lines, and outputs the following request signals:

- Event request, event input sent to the CPU
- The wake-up request is sent to the power consumption management control module

EXTI wake-up requests allow the system to wake up from stop mode, and event requests can also be used in run mode.

EXTI allows the management of up to 31 configurable and direct input lines (22 configurable and 9 direct input lines).

There are 31 edge detectors that can generate event/interrupt requests. There are 22 configurable lines, and each input line can be independently configured with an input trigger mode (rising edge or

falling edge or double edge trigger). 31 configurable lines and direct input lines, each input line can be shielded independently. The suspend register holds the interrupt request for the configurable line.

13.2.1 Main features

The main features of the EXTI controller are as follows:

- Low power mode can be awakened via GPIO and designated modules (e.g. PVD/RTC, etc.)
- Configurable line
  - Optional valid trigger edge (rising/falling edge)
  - Interrupt pending status bits
  - Independent interrupt and event masking bits
  - SW trigger possibility
- Direct input line
  - Fixed rising edge active trigger
  - No interrupt suspend flag bit
  - No independent interrupt and event mask bits
  - No SW trigger possibility

13.2.2 Wake-up event management

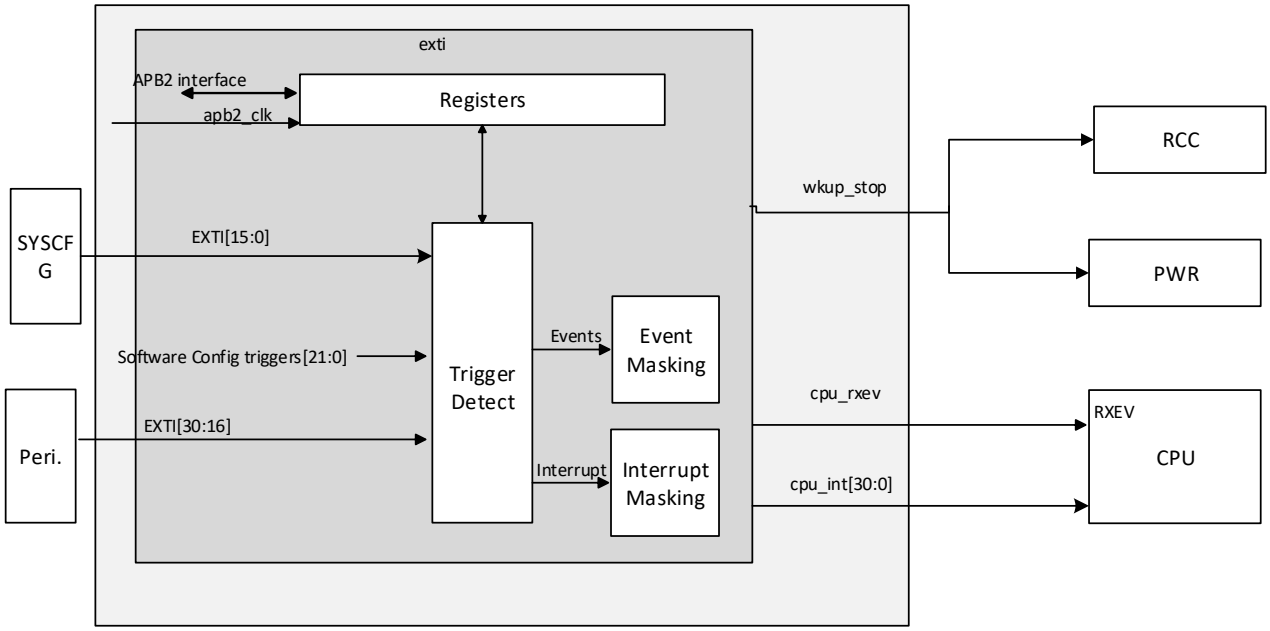
This product can handle external or internal events to wake up the kernel (WFE). The wake-up event can be generated by the following configuration:

- An interrupt is enabled in the control register of the peripheral, but not in the NVIC, while the SEVONPEND bit is enabled in the system control register of the Cortex-M4F. When the CPU recovers from the WFE, it is necessary to clear the interrupt suspend bit of the corresponding peripheral and the peripheral NVIC interrupt channel suspend bit (in the NVIC interrupt clear suspend register).
- Configure an external or internal EXTI line to event mode. When the CPU recovers from WFE, because the suspension bit of the corresponding event line is not set, it is not necessary to clear the interrupt suspension bit of the corresponding peripheral or the NVIC interrupt channel suspension bit.

13.2.3 Function description

To generate an interrupt, the interrupt line must be configured and enabled first. The 22 configurable lines set 2 trigger registers according to the required edge detection, while writing '1' to the corresponding bit of the interrupt mask register allows interrupt requests. When the expected edge occurs on the external interrupt line, an interrupt request will be generated, and the corresponding suspend bit will be set '1' accordingly. The interrupt request will be cleared by writing '1' in the corresponding bit of the suspend register. If you need to generate an event, you must first configure and enable the event line. According to the required edge detection, the event request is allowed by setting 2 trigger registers while writing '1' to the corresponding bit of the event mask register. When a desired edge occurs on the event line, an event request pulse will be generated, and the corresponding suspend bit will not be set to '1'. An interrupt/event request can also be generated by software by writing '1' in the software interrupt event register.

13.2.3.1 EXTI block diagram



13.2.3.2 Hardware interrupt selection

- Configure the mask bit of the interrupt line (EXTI_IMRx (x = 1, 2))
- Configuring trigger select bits (EXTI_RTSR and EXTI_FTSR) for the selected interrupt line;

13.2.3.3 Hardware event selection

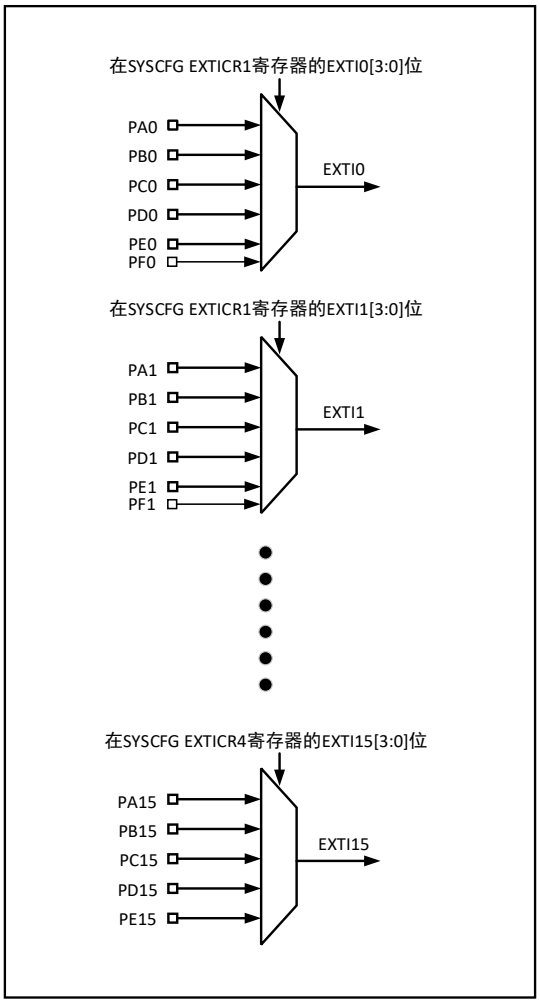
- Configure the mask bit of the event line (EXTI_EMRx (x = 1, 2))
- Configure the trigger select bits of the event line (EXTI_RTSR and EXTI_FTSR)

13.2.3.4 Selection of Software Interrupts/Events

- Configure interrupt/event line masking bits (EXTI_IMRx, EXTI_EMRx; (x = 1, 2))
- Set the request bit of the software interrupt register (EXTI_SWIER)

13.2.4 External interrupt/event line mapping

86 general purpose I/O ports are connected to 16 external interrupt/event lines in the manner shown below:



Note: To configure external interrupts/events on the GPIO line via SYSCFG_EXTICRx, the SYSCFG clock must be enabled first.

Table13-1 EXTI lines connections

EXTI	Line source	Line Type
0-15	GPIO	Configurable line
16	PVD	Configurable line
17	RTC alarm events	Configurable line
18	COMP1 Output	Configurable line
19	COMP2 Output	Configurable line
20	COMP3 output	Configurable line
21	COMP4 output	Configurable line
22	I2C1 ev wake-up	Direct input line
23	I2C2 ev wake-up	Direct input line
24	I2C3 ev wake-up	Direct input line
25	I2C4 ev wake-up	Direct input line
26	Reserved	-
27	Reserved	-
28	Reserved	-
29	Reserved	-
30	LPTIM1 wakeup	Direct input line
31	ETH Wakeup event	Direct input line
32	LPUART wake-up	Direct input line

33	USB1 OTG Wakeup event	Direct input line
34	USB2 OTG Wakeup event	Direct input line

13.3 TIMx registers

These registers must be accessed in 32 bits.

13.3.1 Interrupt Mask Register (0x00: EXTI_IMR1)

Address offset: 0x00

Reset value: 0xC3C0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMR31	IMR30	Res				IMR25	IMR24	IMR23	IMR22	IMR21	IMR20	IMR19	IMR18	IMR17	IMR16
RW	RW					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	IMRx (x = 31, 30)	RW	2'h3	**IMRx (x = 31, 30)**: interrupt masking on line x 0: Mask interrupt request from line x; 1: Open interrupt request from line x
29: 26	Reserved	-	-	**-**
25: 0	IMRx (x = 0-25)	RW	26'h3C00000	**IMRx (x = 0-25)**: interrupt mask on line x 0: Mask interrupt request from line x; 1: Open interrupt request from line x

13.3.2 Event Mask Register (0x04: EXTI_EMR1)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMR31	EMR30	Res				EMR25	EMR24	EMR23	EMR22	EMR21	EMR20	EMR19	EMR18	EMR17	EMR16
RW	RW					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EMR15	EMR14	EMR13	EMR12	EMR11	EMR10	EMR9	EMR8	EMR7	EMR6	EMR5	EMR4	EMR3	EMR2	EMR1	EMR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	EMRx (x = 31, 30)	RW	2'h0	EMRx (x = 31, 30): event masking on line x 0: Mask event requests from line x; 1: Open event requests from line x.
29: 26	Reserved	-	-	**-**
25: 0	EMRx (x = 0-25)	RW	26'h0	EMRx (x = 0-25): event masking on line x 0: Mask event requests from line x; 1: Open event requests from line x.

13.3.3 Rising Edge Trigger Select Register (0x08: EXTI_RTSR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										RTSR21	RTSR20	RTSR19	RTSR18	RTSR17	RTSR16
										RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTSR15	RTSR14	RTSR13	RTSR12	RTSR11	RTSR10	RTSR9	RTSR8	RTSR7	RTSR6	RTSR5	RTSR4	RTSR3	RTSR2	RTSR1	RTSR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 0	RTSRx (x = 0-21)	RW	22'h0	RTSRx (x = 0-21): rising edge trigger event configuration bit on line x 0: Disable rising edge triggering (interrupts and events) on input line x 1: Allow rising edge triggers on input line x (interrupts and events)

13.3.4 Falling edge trigger select register (0x0C: EXTI_FTSR)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										FTSR 21	FTS R 20	FTS R 19	FTS R 18	FTS R 17	FTS R 16
										RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTSR 15	FTSR 14	FTFR 13	FTSR 12	FTSR 11	FTSR 10	FTS R9	FTS R8	FTS R7	FTS R6	FTSR 5	FTS R4	FTS R3	FTS R2	FTS R1	FTS R0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 0	FTSRx	RW	22'h0	**RTSRx (x = 0-21**): falling edge trigger event configuration bit on line x 0: Disable falling edge triggering (interrupts and events) on input line x 1: Allow falling edge on input line x to trigger (interrupts and events)

13.3.5 Software Interrupt Event Register (0x10: EXTI_SWIER)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										SWIE R21	SWIE R20	SWIE R19	SWIE R18	SWIE R17	SWIE R16
										RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWIE R15	SWIE R14	SWIE R13	SWIE R12	SWIE R11	SWIE R10	SWI ER9	SWI ER8	SWI ER7	SWI ER6	SWIE R5	SWIE R4	SWIE R3	SWIE R2	SWIE R1	SWIE R0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 0	SWIERx	RW	22'h0	**SWIERx (x = 0-21**): Software interrupt on line x When this bit is' 0 ', writing' 1 'will set the corresponding pending bit in EXTI_PR. If the interrupt is allowed in EXTI_IMR and EXTI_EMR, an interrupt will be generated at this time. Note: This bit can be cleared to '0' by clearing the corresponding bit of EXTI_PR (write '1').

13.3.6 Suspend register (0x14: EXTI_PR)

Address offset: 0x14

Reset value: 0x000X XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										PR21	PR20	PR19	PR18	PR17	PR16
										RC_W1					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
RC_W1															

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 0	PRx	RC_W1	22 'hX	**PRx (x = 0-21)**: pending bit 0: No trigger request occurred 1: A selected trigger request occurred When a selected edge event occurs on the external interrupt line, the bit is set to '1'. Writing '1' in this bit can clear it, or it can be cleared by changing the polarity of edge detection.

13.3.7 Interrupt Mask Register (0x18: EXTI_IMR2)

Address offset: 0x18

Reset value: 0x0000 0007

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res													IMR34	IMR33	IMR32
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2: 0	IMRx	RW	3'h7	**IMRx (x = 34-32)**: interrupt mask on line x 0: Mask interrupt request from line x; 1: Open interrupt request from line x

13.3.8 Event Mask Register (0x1C: EXTI_EMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res													EMR34.	EMR33	EMR32
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2: 0	EMRx	RW	3'h0	**EMRx (x = 34-32)**: event masking on line x 0: Mask event requests from line x; 1: Open event requests from line x.

14. Analog-to-digital converters (ADC)

14.1 Introduction

A 12-bit ADC is an analog-to-digital converter using successive approximation. It has 20 multiplexed channels and can convert analog signals from multiple external channels and multiple internal channels. The analog watchdog allows the application to detect whether the input voltage exceeds the user-set high and low threshold. The A/D conversion of the individual channels may be configured in a single, continuous, scanning, or intermittent conversion mode. The results of the ADC conversion can be stored in a 16-bit data register in a left-aligned or right-aligned manner.

14.2 Main Features

- High Performance:
  - Configurable 12-, 10-, 8-and 6-bit resolutions
  - Maximum ADC sampling rate: 4 Msps
  - Self-calibration
  - Programmable sampling time
  - The data register allows configurable data alignment
  - Support DMA requests for regular channel data conversion
  - Dual ADC Mode
  - Configurable single-ended or differential inputs
- Oversampler
  - 16 bits Data register
  - Oversampling rate 2-256 adjustable
  - Programmable data shift up to 8 bits
- Data preprocessing
  - Gain compensation
  - Offset compensation
- Dynamic low power consumption characteristics:
  - Automatic delay transition mode: prevents overflow when operating with low frequency PCLK
- Analog input channel:
  - ADC1 connects 14 external channels, 5 internal channels and 1 internal VSSA channel
  - ADC2 connects 16 external channels, 3 internal channels and 1 internal VSSA channel
  - ADC3 connects 15 external channels, 4 internal channels and 1 internal VSSA channel
- Start conversion method:
  - By software
  - Hardware triggering
- Conversion mode:
  - Single-shot mode, which converts the selected input channel once per trigger;
  - Scan mode, which can scan a series of channels;
  - A continuous mode that continuously switches the selected input channel;

- Discontinuous mode, each trigger continuously converts sub-sequence channels, and multiple triggers until all channels are converted
- Synchronous mode (for devices with two ADCs).

- Analog watchdog
- Generation of interrupts:
  - End of Sampling
  - Transition End (Rule/Injection Channel)
  - End of rule sequence or injection sequence conversion
  - Simulated Watchdog 1, 2, 3 events
- ADC input range (select external reference voltage): VREF-≤ VIN ≤ VREF +

14.3 Function description

14.3.1 ADC module block diagram

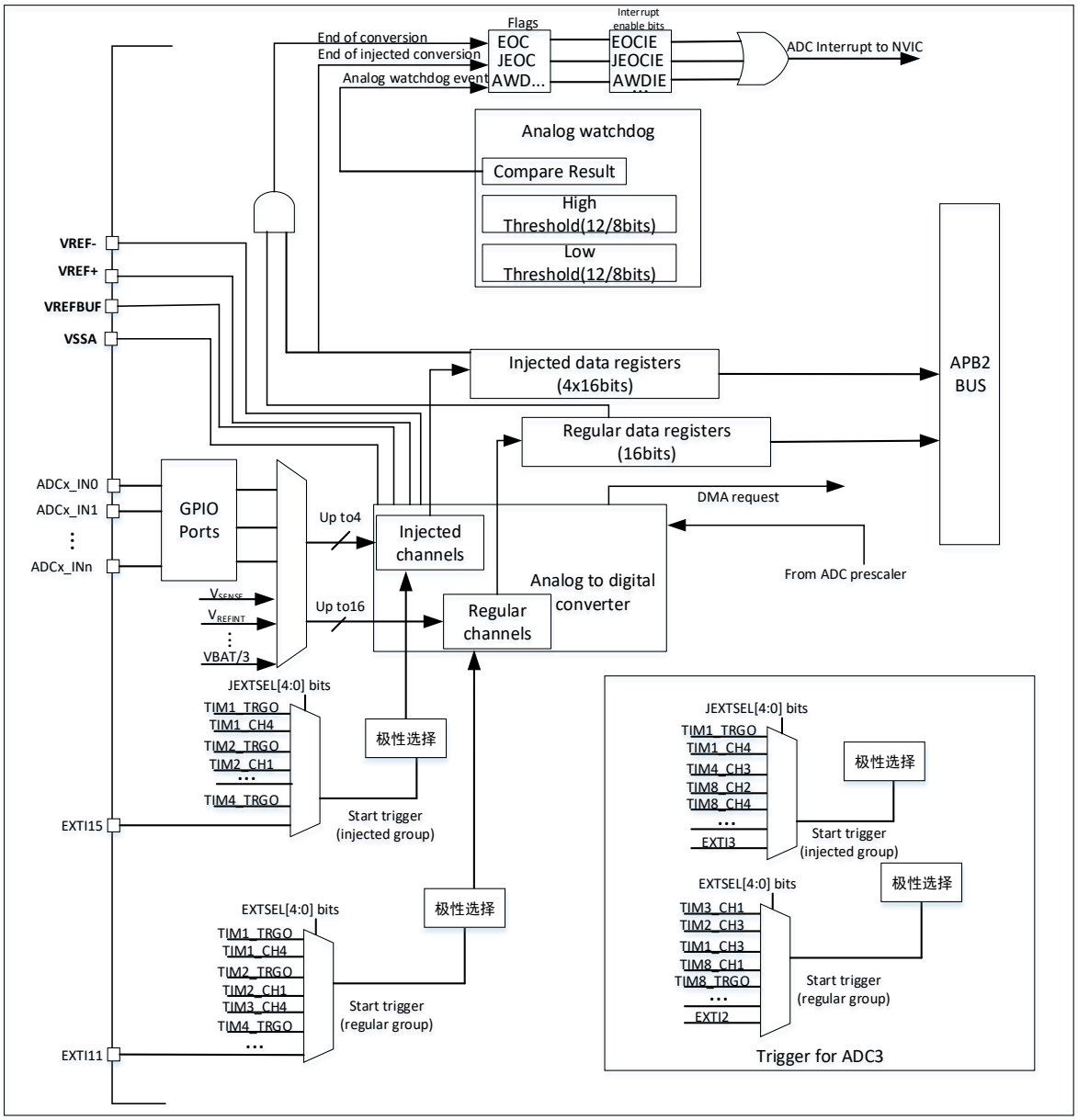


Figure 14-1 ADC module block diagram

14.3.2 ADC1/2/3 connection relationship

The external channels of ADC1, ADC2, and ADC3 share some GPIO (channel configuration is set via ADC_SQRx or ADC_JSQR), as shown in the figure below.

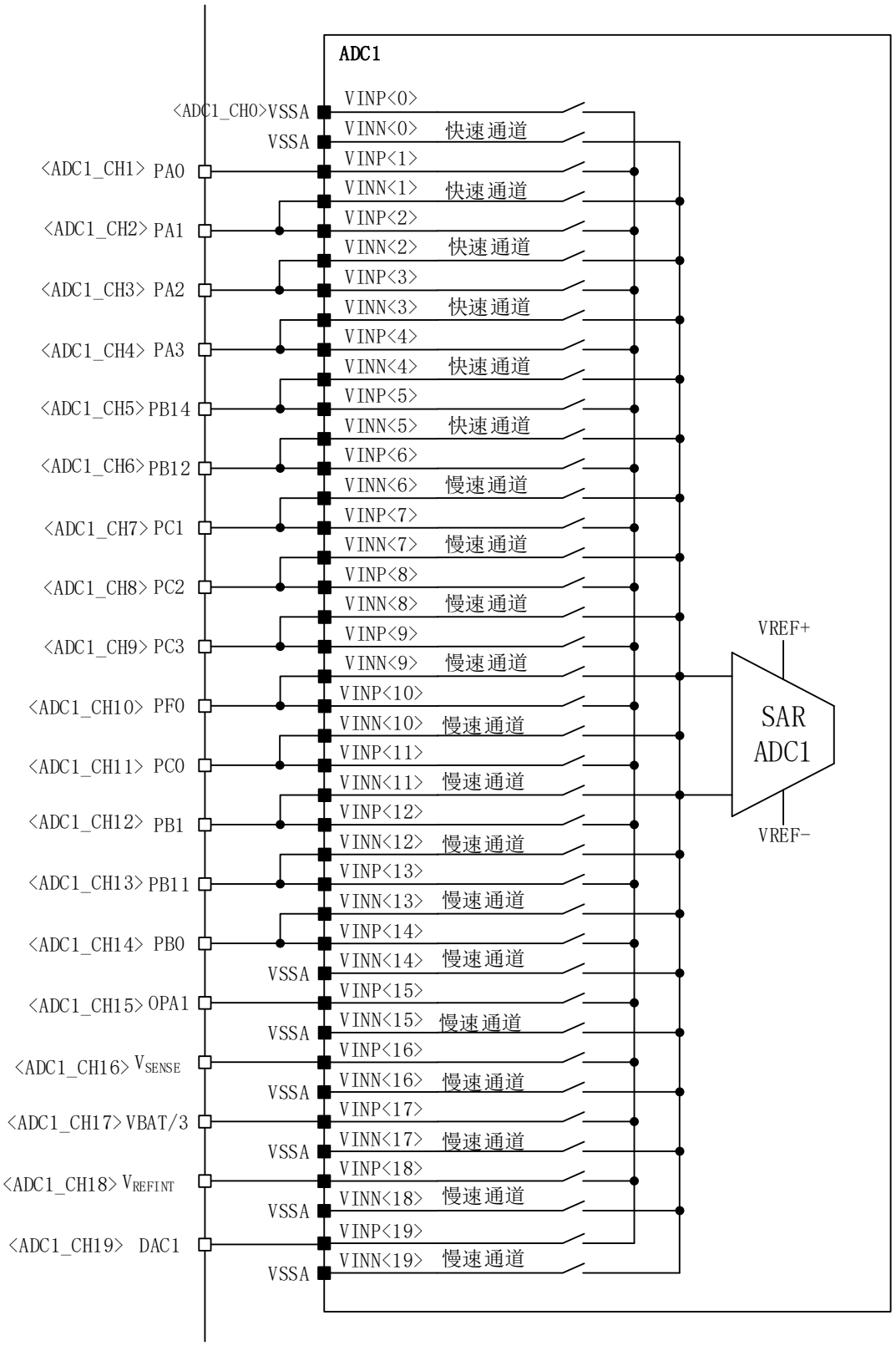


Figure 14-2 ADC1 channel connection relationship

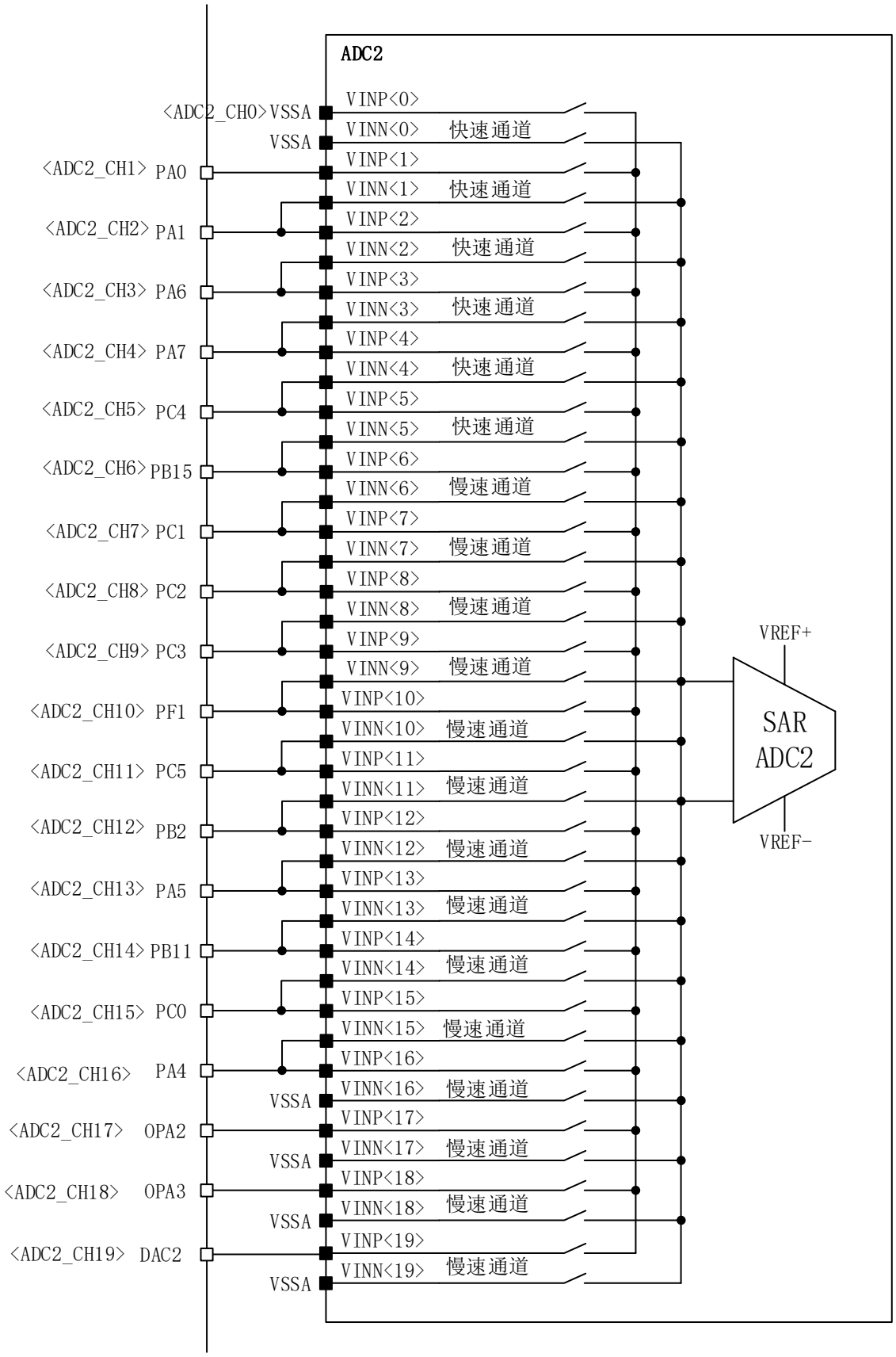


Figure 14-3 ADC2 channel connection relationship

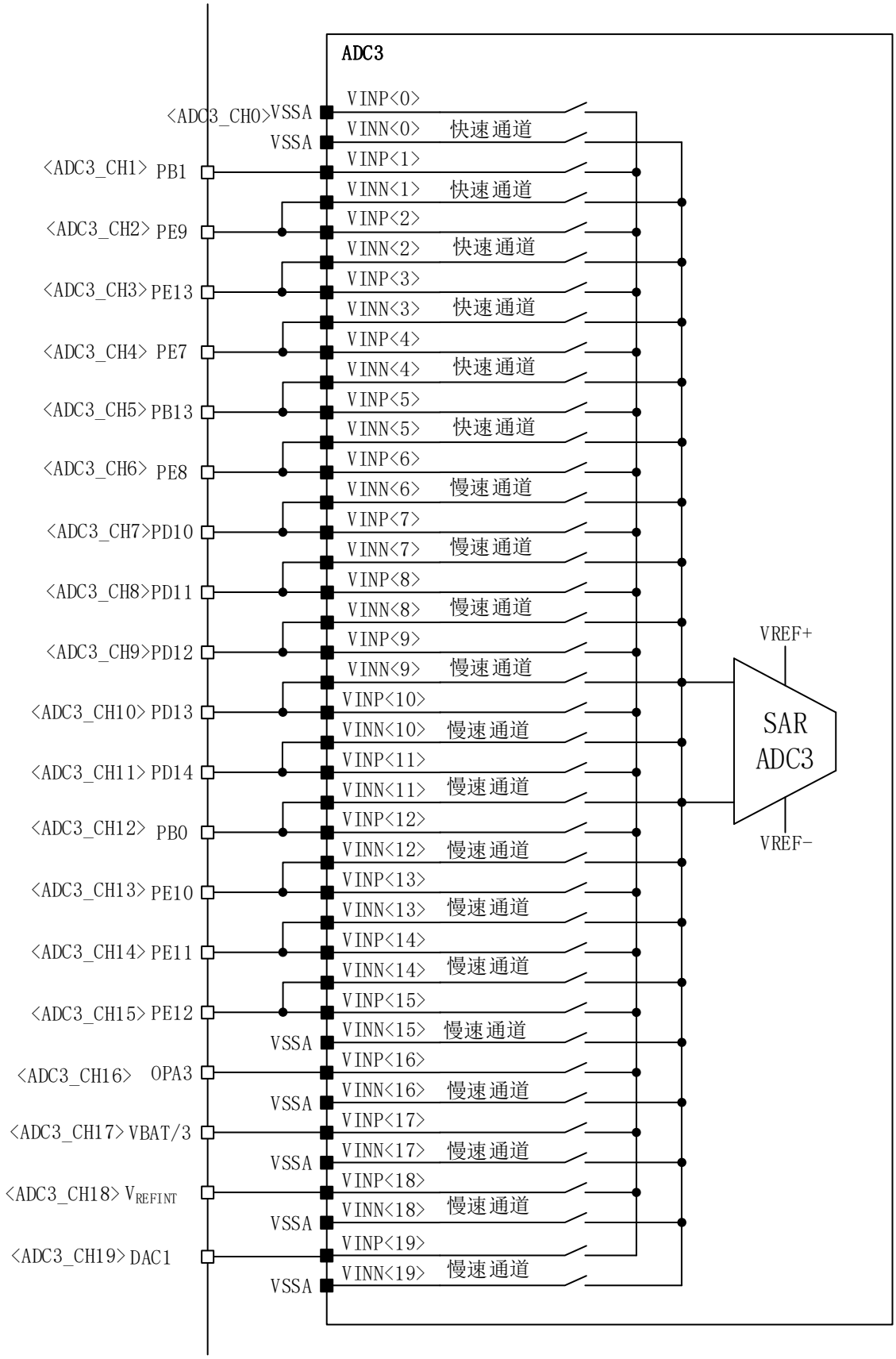


Figure 14-4 ADC3 channel connection relationship

14.3.3 ADC1/2/3 use restrictions

1. Single ADC and dual ADC modes: Discontinuous mode (DISCEN), which does not support injection interrupt rules.
2. Single ADC and dual ADC modes: Automatic delay conversion mode (AUTDLY), which does not support injection interrupt rules.
3. Single ADC and Dual ADC Modes: Automatic Injection (JAUTO) does not support Automatic Delay Conversion Mode (AUTDLY) mode
4. Single ADC and Dual ADC modes: Rule triggers during injection are ignored.
5. Single ADC and Dual ADC Modes: For single conversion mode, the software must set the rule sequence length to 0.
6. Dual ADC mode (ADC3 does not have dual ADC mode):
   a) The JEXTEN/JEXTSEL/EXTEN/EXTSEL/JADSTART/ADSTART of the Slave ADC must remain reset
   b) Injection break rules are not supported when DUAL = 7, 8, 9
   c) When DUAL = 8, the sampling time can only be selected as 2.5, 3.5 and 6.5 ADC clock cycles, and other sampling times cannot be configured.

14.3.4 Single-ended and differential input channels

The ADC channelcan be configured as a single-ended input or a differential input, configured by DIFSEL in the ADC_DIFSEL register. When DIFSEL [x] (x = 0-19) is 1, the channel that is selected to be converted is differential input mode, otherwise it is single-ended input mode.

Must be configured when the ADC is disabled (ADEN = 0). Note that the single-ended channel corresponding to DIFSEL is always kept at 0.

In single-ended mode, the analog voltage of channel i (i = 0-19) is the difference between ADCy_INx (VINP [i]) and VREF−.

In differential mode, the analog voltage of channel i (i = 0-19) is the voltage difference between ADCy_INx (VINP [i]) and ADCy_INx+1 (VINN [i]).

In differential mode, the input voltage ranges from VREF-to VREF +, reaching a full scale of 2xVREF +. When VINP [i] is equal to VREF − and VINN [i] is equal to VREF +, the maximum negative input differential voltage (VREF −) corresponds to 0x000 output. When VINP [i] is equal to VREF + and VINN [i] is equal to VREF −, the maximum positive input differential voltage (VREF +) corresponds to the 0xFFF output. When VINP [i] and VINN [i] are connected together, the zero input differential voltage corresponds to the 0x800 output.

The ADC sensitivity in differential mode is twice less than in single-ended mode. Internal channels, such as VSENSE and VREFINT, are in single-ended mode only.

For a complete description of each ADC input channel connection, see the relevant section description.

Warning:

When channel "i" is configured in differential input mode, its negative input voltage VINN [i] is already connected to another channel, so this channel is no longer available for single-ended mode or differential mode, and conversion must not be configured. ADC1/ADC2/ADC3 share some channels: this

may cause a channel on another ADC to be unavailable. The only exception is when the master ADC and the slave ADC are operating in crossover mode.

14.3.5 ADC calibration

The ADC has a calibration function, which can be enabled by users through software. During calibration, the ADC calculates a calibration factor for use inside the ADC (lost after the ADC is powered down). The application cannot use the ADC module during the ADC calibration and until the calibration is complete. Before using ADC conversion, users are advised to perform calibration operations. Calibration is used to eliminate offset errors and mismatch errors between chips due to process and temperature changes.

14.3.5.1 ADC software calibration

The software sets ADCAL = 1 to start calibration. The calibration can only be started when the ADC is not turned on (ADEN = 0), and only supports selecting the system clock as the clock of the ADC. The calibration clock frequency is 1/4 of the normal conversion clock frequency. When the calibration is complete, the ADCAL is cleared by the hardware.

When the operating conditions of the ADC change (VCC change is the main factor of offset error and mismatch error, followed by temperature change), it is recommended to recalibrate.

The number of repetitions of the calibration process is configured by setting CALNUM [2: 0] in the ADC_CFGR2 register, and the results are averaged to get more accurate calibration results. (Note: CALNUM cannot be 0 when the calibration average result is compensated to AD conversion)

The internal calibration register can be reset by setting the RSTCAL of ADC_CR, which is set to 0 by a hardware clear. This bit is cleared after the calibration register is initialized (i.e., after RSTCAL is set to 1).

Software procedure to calibrate the ADC:

1. Confirm ADEN = 0
2. Set RSTCAL (this step is optional);
3. Set the number of calibrations CALNUM (this step is optional);
4. Set ADCAL = 1
5. The ADC internally automatically delays 192 ADC CLKs to wait for the ADC to stabilize;
6. Wait until ADCAL = 0

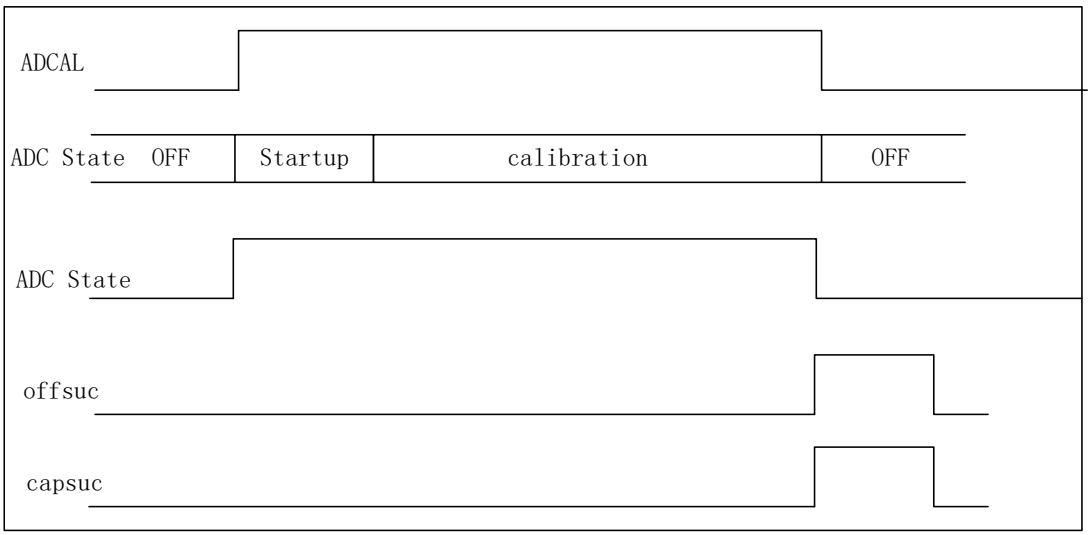


Figure 14-5 ADC Calibration

14.3.6 ADC on-off control

The ADEN of the ADC_CR register is the enable switch of the ADC module. In order to save power, when ADEN is 0, the ADC analog circuit will enter power-down mode.

Note: After the ADEN bit is set to 1, the hardware waits for a delay (tSTAB) of not less than 3 us before converting.

By clearing the ADEN bit, the transition can be stopped and the ADC state can be reset, and the ADC analog circuit can be put into power down mode.

14.3.7 Limits on Write ADC Control Bits

Only when the ADC is disabled (ADEN = 0) and there is no calibration (ADCAL = 0), the software writes RCC to enable the ADC clock (see RCC section), the software can be configured:

- CALNUM for ADC_CFGR2
- RSTCAL, ADCAL, and ADEN in the ADC_CR register.
- ADC_CALFACT

For all registers except the above registers:

- The software is only allowed to write these control bits when the ADC does not have ongoing rules and injection translations.

For the ADC_CCR register, software is allowed to write these control bits only when the ADC is disabled (ADEN = 0).

Note: If the ADC registers are not configured as described in this section, the ADC behavior may be in an unknown state. To recover from this situation, ADC must be disabled (ADEN cleared).

14.3.8 ADC clock

The ADC CLK and PCLK2 (APB2 clock) supplied by the clock controller are synchronized. The RCC controller (CLK controller) provides a dedicated programmable prescaler for the ADC clock.

14.3.9 ADC channel selection

ADC1 has 14 external channels, 5 internal channels and 1 internal VSSA channel, where the internal channels are:

- ADC1_CH15 connects OPA1

- ADC1_CH16 connects temperature sensor VSENSE,
- ADC1_CH17 connects VCC/3,
- ADC1_CH18 connects VREFINT,
- ADC1_CH19 connects DAC1.

ADC2 has 16 external channels, 3 internal channels and 1 internal VSSA channel, where the internal channels are:

- ADC2_CH17 connects OPA2,
- ADC2_CH18 connects OPA3,
- ADC2_CH19 is connected to DAC2.

ADC3 has 15 external channels, 4 internal channels and 1 internal VSSA channel, where the internal channels are:

- ADC3_ CH16 connects OPA3,
- ADC3_CH17 connects VCC/3,
- ADC3_CH18 connects VREFINT,
- ADC3_CH19 is connected to DAC1.

ADCchannel selection can be configured through ADC_SQRx or ADC_JSQR. The correspondence between configuration values and channels is described in the above chapter.

ADC divides conversions into two groups: rule conversions and injection conversions. Each group contains a sequence of transformations that can be completed on any channel in any order. For example, the sequence may be converted in the following order: ADC_IN3, ADC_IN8, ADC_IN2, ADC_IN2, ADC_IN0, ADC_IN2, ADC_IN2, ADC_IN15. All channels can be converted by injection or regular channel group.

- A rule transformation group consists of up to 16 transformations. The regular channel of the conversion sequence and its conversion order are selected in the ADC_SQRx register. The sequence length in the rule transformation group must be written L [3: 0].
- An injection conversion group consists of a maximum of 4 conversions. The injection channel and its conversion order are selected in the ADC_JSQR register. The sequence length in the injection translation group must be written to the JL [1: 0] bit. (Note: Unlike regular conversion sequences, if the length of JL [1: 0] is less than 4, the sequence order of the conversion starts from (4-JL).)

14.3.10 Forced stop of ADC

The software sets ADSTP = 1 in the ADC_CR register to stop the current conversion and let the ADC enter the idle state to prepare for the next conversion.

When ADSTP is set to 1 by the software, any current transition is aborted and the transition result is discarded (the ADC_DR register is not updated with the current transition value). Once the conversion process is finished, ADSTP is cleared by hardware.

14.3.11 Dynamic low power consumption characteristics

14.3.11.1 Automatic Delay Conversion Mode (AUTDLY)

The automatic delay conversion mode can be used to simplify software and optimize application performance when running at low speeds. Of course, ADC overflow is not easy to occur in this mode. This is a way to automatically adapt the speed of the ADC to the speed of the system that reads the data.

Automatic delay conversion (AUTDLY) mode means that after one conversion of a regular channel, you need to wait for the CPU/DMA to read away the data in the data register before starting the sampling conversion of the next channel. If the CPU/DMA does not read the last converted data, the whole sampling conversion cycle will not switch to the next channel, but will remain in a waiting state until the last converted data is read and then the conversion will continue. If a regular channel trigger event occurs during AUTDLY, the trigger event is ignored.

The AUTDLY mechanism supports all conversion modes of the regular channel. When the AUTDLY flag bit is enabled, it indicates that the ADC conversion cycle needs to wait for the data register to be read away before it can be carried out.

Note: AUTDLY mode does not support injection-triggered interrupt rule transformations.

14.3.12 Conversion modes

14.3.12.1 Single conversion mode

In the single conversion mode, the ADC performs a conversion on a single channel, which can be run in the rule group and the injection group. The SQ1 [4: 0] bit specifies the single conversion channel of the ADC rule group, and the application should keep L as 0; When injecting a single transition channel, the application should keep JL at 0. (Note: Unlike regular transition sequences, if the length of JL [1: 0] is less than 4, the sequence order of transitions starts from (4-JL).) The JSQ4 [4: 0] bit of the ADC_JSQR register configures the injection group single transition channel.

When the ADEN bit is 1 and CONT, SCAN, and DISCEN/JDISCEN are all 0, the ADC will sample and convert a channel once the corresponding external trigger occurs. (Note: Single-shot mode and continuous mode cannot be enabled at the same time, that is, CONT is 0 in single-shot mode.)

- If a regular channel was converted:

When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

-The conversion data is stored in the 16-bit ADC_DR register

After the conversion is completed, the EOC (conversion end) flag is set

-If EOCIE is set, interrupt is generated.

- If an injected channel is converted:

When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

-Conversion data is stored in a 16-bit ADC_JDRx (x = 1-4) register

JEOC (Injection Conversion End) flag is set after the conversion is completed

-If JEOCIE bit is set, an interrupt is generated.

After the rule sequence is complete:

- Set the EOS (End of Rule Sequence) flag
- If the EOSIE bit is set, an interrupt is generated

After completion of the injection sequence:

- Set the JEOS (End of Injection Sequence) flag
- If the JEOSIE bit is set, an interrupt is generated

14.3.12.2 Continuous conversion mode

In the continuous conversion mode, the ADC continuously converts the selected channel, and this mode can be run in the rule group. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

- Regular Channel Group: The successively converted channels are set by the SQ1 [4: 0] bit.
- Injection channel group: only a single conversion is performed, and the converted channel SQx [4: 0] (x = 1-4) is set. Note: Unlike regular transformation sequences, if the length of JL [1: 0] is less than 4, the sequence order of the transformation starts from (4-JL)

When the ADEN bit is 1 and the CONT is 1, the ADC samples and converts the selected channel once the corresponding external trigger occurs.

- If a regular channel was converted:

  -The conversion data is stored in the 16-bit ADC_DR register

  After the conversion is completed, the EOC (conversion end) flag is set

  -If EOCIE is set, interrupt is generated.
- If an injected channel is converted:

  -Conversion data is stored in a 16-bit ADC_JDRx (x = 1-4) register

  JEOC (Injection Conversion End) flag is set after the conversion is completed

  -If JEOCIE bit is set, an interrupt is generated.

After the sequence of conversions is complete:

- Set the EOS (End of Sequence) flag
- If the EOSIE bit is set, an interrupt is generated

The new sequence is then immediately restarted, and the ADC continuously repeats the switching sequence.

After completion of the injection sequence:

- Set the JEOS (End of Injection Sequence) flag
- If the JEOSIE bit is set, an interrupt is generated

Note: You cannot enable intermittent conversion mode and continuous mode at the same time: it is forbidden to set JDISCEN/DISCEN = 1 and CONT = 1 at the same time. The injected channel does not support continuous conversion mode.

14.3.12.3 Scan mode

When scanning conversion mode, the ADC continuously converts all channels of a selected sequence. This mode can be run in the rule group and the injection group. ADC_SQRx specifies a regular sequence length and all conversion channels, where the L [3: 0] bit specifies the sequence length. The ADC_JSQR register specifies an injection group sequence length and all conversion channels, where the JL [1: 0] bit specifies the sequence length (Note: Unlike regular conversion sequences, if the length of JL [1: 0] is less than 4, the sequence order of conversion starts from (4-JL)). When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

When the ADEN bit is 1 and the SCAN is 1, the ADC samples and converts a sequence once the corresponding external trigger occurs.

- If a sequence of rules is converted:

-The conversion data is stored in the 16-bit ADC_DR register

The EOC (End of Conversion) flag is set after the completion of each channel conversion, and the EOS flag is set after the completion of the entire sequence conversion.

-If EOCIE is set, an interrupt is generated, if EOSIE is set, an interrupt is generated.

- If an injection sequence is converted:

-Conversion data is stored in a 16-bit ADC_JDRx (x = 1-4) register

JEOC (End of Injection Transition) flag is set after each channel transition is completed, and JEOS flag is set after the entire sequence transition is completed.

-If JEOCIE bit is set, an interrupt is generated. An interrupt is generated if JEOSIE is set.

14.3.12.4 Discontinuous mode

In the intermittent conversion mode, the ADC divides the selected sequence into sub-sequences (sequence length 1-8), and completes the whole sequence conversion by externally triggering the subsequences many times. This mode can be run in the rule group and the injection group. DISCNUM specifies the regular channel subsequence length, and ADC_SQRx (x = 1-4) specifies all conversion channels of a regular sequence, where L [3: 0] bits specify the sequence length. The ADC_JSQR register specifies all translation channels of an injection group sequence, where the JL [1: 0] bit specifies the sequence length (Note: Unlike regular translation sequences, if the length of JL [1: 0] is less than 4, the sequence order of translations starts from (4-JL)). When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

Note: The subsequence of the injection channel group is 1. In this mode, DISCEN and JDISCEN are prohibited from being enabled at the same time

- Regular group

An external trigger signal initiates the ADC_SQRx (x = 1-4) description channel n (n < = 8) transitions untilall transitionsin this sequence are complete. The total sequence length is defined by L [3: 0].

For example:

n = 3, transformed channels = 0, 1, 2, 3, 6, 7, 9, 10

First trigger: The sequence of transitions is 0, 1, 2, and an EOC event is generated at the end of each channel transition.

Second trigger: The sequence of transitions is 3, 6, 7, and an EOC event is generated at the end of each channel transition.

Third trigger: The sequence of transitions is 9 and 10, and the end of each channel transition generates an EOC event. The end of the sequence produces an EOS event.

Fourth trigger: Transition sequence 0, 1, 2, EOC event is generated at the end of each channel transition.

Note: When a rule group is converted in discontinuous mode, the conversion sequence does not automatically start from the beginning. When all subgroups have been converted, the next trigger initiates the conversion of the first subgroup. In the above example, the fourth trigger re-switches channels 0, 1 and 2 of the first subset.

- Injected group

An external trigger signal initiates 1 transition of the ADC_JSQR description channel until all transitions in the sequence are completed. Total sequence length JL [1: 0] bit definition.

Examples include:

n = 1, channel being converted = 1, 2, 3

First trigger: Channel 1 is converted, and the end of each channel conversion generates a JEOC event.

Second trigger: Channel 2 is converted, and the end of each channel conversion produces a JEOC event.

Third trigger: Channel 3 is converted, and the end of each channel conversion produces a JEOC event. The end of the sequence produces a JEOS event.

Fourth trigger: Channel 1 is converted, and the end of each channel conversion generates a JEOC event.

Notes:

1 When all injection channel transitions are completed, the next trigger starts the transition of the first injection channel. In the above example, the fourth trigger re-switches the first injection channel 1.

2 Automatic injection and intermittent modes cannot be used simultaneously.

3 You must avoid setting discontinuous mode for rules and injection groups at the same time. The discontinuous mode can only act on one set of transitions.

14.3.13 Injected channel management

The injection channel external trigger has a higher priority than the regular channel external trigger, that is, the injection channel external trigger can interrupt the ongoing regular channel transition. The injection channel is interrupted in two ways: triggered injection and automatic injection.

14.3.13.1 Triggered injection

In the process of executing a rule sequence, the ADC detects an external trigger of the injection channel, and interrupts the channel that the current rule sequence is converting, and starts to execute the injection sequence conversion. When all the injection sequences are converted, the ADC restores the interrupted rule channel and completes the rule sequence conversion.

ADC_SQRx (x = 1-4) specifies a regular sequence of all conversion channels, where L [3: 0] bits specify the sequence length. The ADC_JSQR register specifies all transition channels of an injection group sequence, where the JL [1: 0] bit specifies the sequence length. Note: Unlike regular translation sequences, if the length of JL [1: 0] is less than 4, the sequence order of the translation starts from (4-JL).

Note: JAUTO must be 0 when injection is triggered. A rule trigger event occurs during an injection channel transition, which transforms after the injection channel is completed, that is, starts the rule sequence from scratch instead of resuming the interrupted rule channel. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

(Note: Discontinuous mode does not support trigger injection; AUTDLY function is not supported when trigger injection; When rule trigger and injection trigger occur simultaneously, rule trigger is discarded and only injection trigger is executed; rule trigger during injection is ignored.)

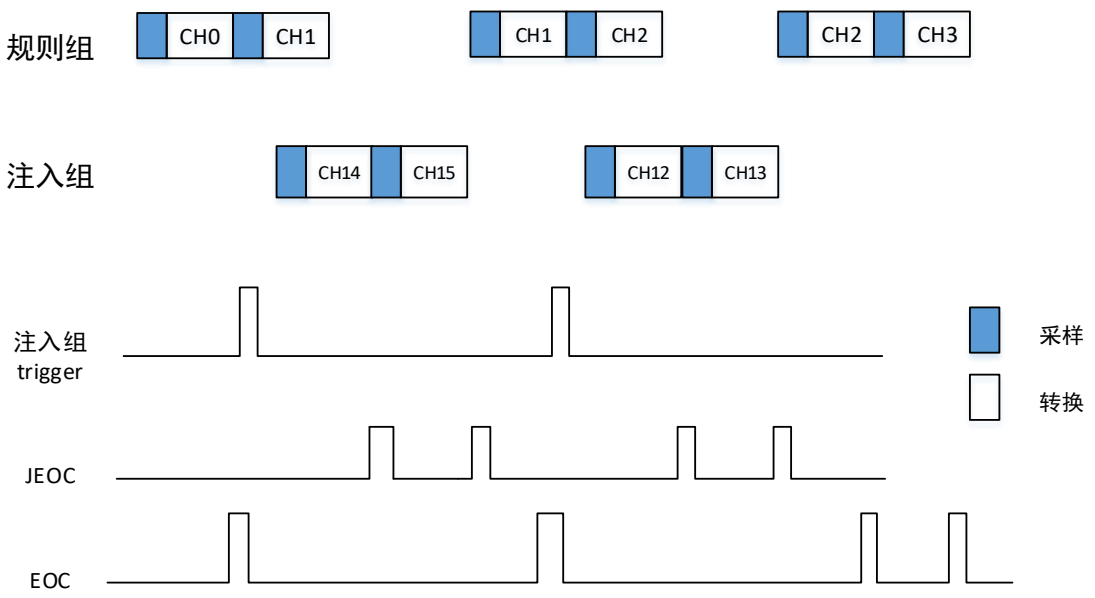


Figure 14-6 ADC Trigger Injection (Scan Mode)

14.3.13.2 Auto-injection

In automatic injection mode, the ADC automatically performs injection sequence conversion after completing a regular sequence. JAUTO enables the auto-injection mode bit. ADC_SQRx (x = 1-4) specifies a regular sequence of all conversion channels, where L [3: 0] bits specify the sequence length. The ADC_JSQR register specifies all transition channels of an injection group sequence, where the JL [1: 0] bit specifies the sequence length. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

For automatic injection in non-oversampling mode, after the injection sequence is finished, JEOS and EOS are set.

For automatic injection in oversampling mode, after the end of the regular sequence, EOS is set, and after the end of the injection sequence, JEOS is set.

(Note that this mode must disable the external trigger of the injection channel; you can't use auto-injection and intermittent modes at the same time; auto-injection does not support AUTDLY mode)

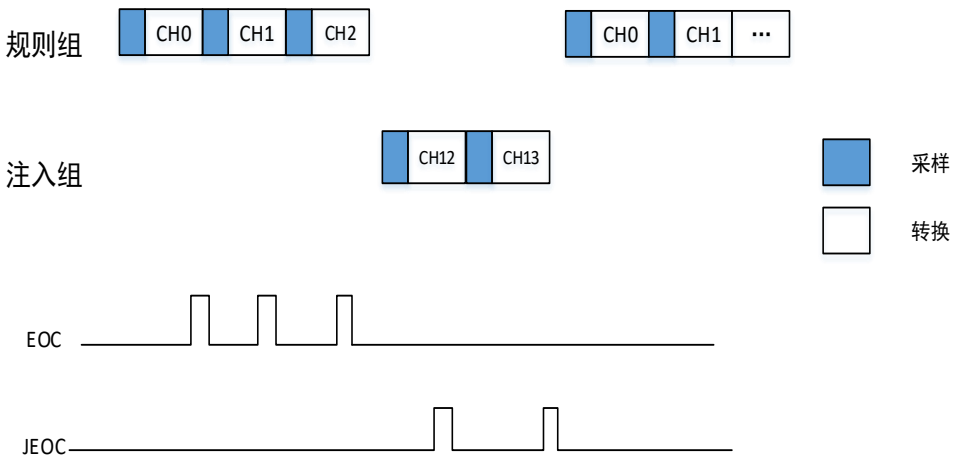


Figure 14-7 ADC automatic injection (CONT + SCAN + JAUTO)

14.3.14 Analog watchdog

Three AWD analog watchdogs monitor whether the ADC channel remains within the configured voltage range (window)

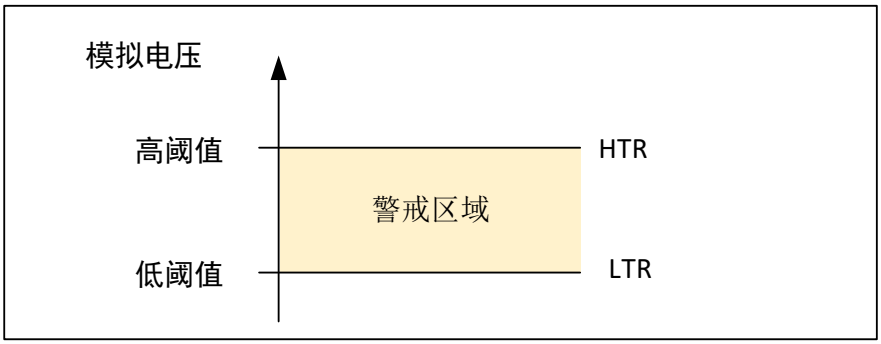


Figure 14-8 Simulated watchdog protection area

14.3.14.1 AWDx flags and interrupts

Interrupts can be enabled for each of the 3 analog watchdogs by setting AWDxIE (x = 1, 2, 3) in the ADC_IER register. The AWDx (x = 1, 2, 3) flag is cleared by the software writing 1. The ADC conversion results were compared to low threshold, high threshold before data alignment.

14.3.14.2 Analog Watchdog 1 Description

Enable AWD simulation watchdog 1 via the set JAWD1EN/AWD1EN bit. This watchdog monitors whether a selected injection/rule channel or all enabled injection and rule channels remain within the configured voltage range (window).

The following table shows how the ADC_CFGR register should be configured to enable analog watchdogs on one or more channels.

Table 14-1 Analog watchdog channel selection

Channels to be guarded by analogwatchdog	AWD1SGL bit	AWD1EN bit	JAWD1EN bit
None	x	0	0
All injected channels	0	0	1
All regular channels	0	1	0
All rules and injection channels	0	1	1

Single injection channel	1	0	1
Single regular channel	1	1	0
Single rule/injection channel	1	1	1

1. When AWD1SGL is selected by AWD1CH [4: 0], the watchdog protects the channel, and the protected channel shall be converted in a regular or injection sequence.

   If the analog voltage converted by the ADC is lower than the low threshold or higher than the high threshold, the AWD1 analog watchdog status bit is set. These thresholds are written in bits HT1 [11: 0] and LT1 [11: 0] of the ADC_TR1 register of the analog watchdog 1. When transforming data with a resolution less than 12 bits (according to bit RES [1: 0]), the LSB of the threshold must be kept at 0 because internal comparisons are always performed on the full 12-bit raw transformed data (left-aligned).

   The following table describes how the comparison is performed for all possible resolutions of the simulated watchdog 1.

Table 14-2 Simulated Watchdog 1 Comparison

Resolution (RES [1: 0])	Analog watchdog comparison between:		Description
	Original transformation left aligned	Thresholds	
00: 12 bit	DATA [11: 0]	LT1 [11: 0] and HT1 [11: 0]	-
01: 10bit	DATA [11: 2], 00	LT1 [11: 0] and HT1 [11: 0]	The user must configure LT1 [1: 0] and HT1 [1: 0] to 00
10: 8 bit	DATA [11: 4], 0000	LT1 [11: 0] and HT1 [11: 0]	The user must configure LT1 [3: 0] and HT1 [3: 0] to 00
11: 6bit	DATA [11: 6], 000000	LT1 [11: 0] and HT1 [11: 0]	The user must configure LT1 [5: 0] and HT1 [5: 0] to 00

1. For more details on analog watchdog comparisons, see the "Gain Compensation" section.

14.3.14.3 Analog Watchdog Filter for Watchdog 1

When the ADC is configured with only one input channel (multiple channels are not allowed to be selected in scan mode), the valid ADC conversion data interval can be configured through the ADC_TR1 register:

- When the conversion result exceeds the configured high/low threshold, but is within the interval defined in ADC_TR1, a DMA request is generated for each conversion result (if DMA is enabled).
- Otherwise, no DMA request is issued (i.e., multiple consecutive transition results exceed the configured threshold and exceed the interval, and the transition result exceeding the interval does not produce a DMA request (if DMA is enabled)). The RDATA register is updated with each conversion. If the data is out of range several times higher than the value specified in the AWDFILT bit of ADC_TR1, the AWDx flag is set and the corresponding interrupt is generated.

14.3.14.4 Description of Watchdogs 2 and 3

The 2nd and 3rd analog watchdogs are more flexible and can protect several selected channels by programming the corresponding bits in AWDxCH [19: 0] (x = 2, 3). When any bit of AWDxCH [19: 0] (x = 2, 3) is set, the corresponding watchdog is enabled. They are limited to a resolution of 8 bits, and only a threshold of 8 MSBs can be programmed as HTx [7: 0] and LTx [7: 0] (x = 2-3). The following table describes how the comparison is performed for all possible resolutions.

Table 14-3 Simulated Watchdog 2 and 3 Comparison

Resolution (RES [1: 0])	Analog watchdog comparison between:		Description
	Original transformation left aligned	Thresholds	
00: 12 bit	DATA [11: 4]	LTx [7: 0] and HTx [7: 0]	Data [3: 0] is not relevant to the comparison
01: 10 bit	DATA [11: 4]	LTx [7: 0] and HTx [7: 0]	Data [3: 2] are not relevant for comparison
10: 8 bit	DATA [11: 4]	LTx [7: 0] and HTx [7: 0]	-
11: 6 bit	DATA [11: 6], 00	LTx [7: 0] and HTx [7: 0]	The user must configure LTx [1: 0] and HTx [1: 0] to 00

14.3.14.5 ADCy_AWDx_OUT signal output generation

Each analog watchdog is associated with an internal hardware signal ADCy_AWDx_OUT (y = 1, 2, 3; x = 1, 2, 3), which is directly connected to the ETR input (externally triggered) of some on-chip timers. Refer to the on-chip timer section to learn how to select the ADCy_AWDx_OUT signal as the ETR.

ADCy_AWDx_OUT is activated when the associated analog watchdog is enabled:

- When the protected transition exceeds the programming threshold, ADCy_AWDx_OUT is set.
- ADCy_AWDx_OUT resets after the end of the next guard transition within the threshold (remains at 1 if the next analog watchdog channel transition is still outside the programmed threshold).
- When the ADC is disabled (when ADEN = 0 is set), ADCy_AWDx_OUT is reset. Note that stopping rules or injected transformations (setting ADSTP = 1) have no effect on the generation of ADCy_AWDx_OUT.

Note: The AWDx flag is set by the hardware and reset by the software: the AWDx flag has no effect on the generation of ADCy_AWDx_OUT (for example, if the software does not clear AWDx, the AWDx flag remains at 1 and the ADCy_AWDx_OUT can be flipped).

Table 14-4 ADCy_AWDx_OUT connection relationship

ADC1_AWDx_OUT			ADC2_AWDx_OUT			ADC3_AWDx_OUT		
AWD1	AWD2	AWD3	AWD1	AWD2	AWD3	AWD1	AWD2	AWD3
tim1_etr5	tim1_etr6	tim1_etr7	tim3_etr8	tim3_etr9	tim3_etr10	tim8_etr8	tim8_etr9	tim8_etr10
-	-	-	tim8_etr5	tim8_etr6	tim8_etr7	-	-	-

14.3.14.6 Analog watchdog with gain and offset compensation

When gain and offset compensation are enabled, the analog watchdog compares thresholds after compensating the data.

Note: When offset compensation is enabled (OFFSETy_EN in the ADC_OFRy register is set to 1), data overflows or underflows can result in false watchdog results. When overflow protection is enabled (with SATEN set to 1 in ADC_OFRy), the watchdog provides the correct results. But this prevents the use of signed data format.

14.3.15 Data processing

14.3.15.1 Data alignment

The ALIGN bit in the ADC_CFGR register is used to select the alignment mode of the data stored after conversion. You can choose left alignment and right alignment. The converted data value has been subtracted from the custom offset in the ADC_OFRy register, so the result can be a negative value. The SEXT bit represents the extended sign value.

Special case: When left alignment, except for the resolution of 6 bits, other resolution data are aligned on a half-word basis. At a resolution of 6 bits, the data is aligned on a byte basis.

Note: Left alignment is not supported in oversampling mode. When the ROVSE or JOVSE bit is set, the ALIGN bit value is ignored and the ADC only provides right-aligned data.

14.3.15.2 offset

An offset y (y = 1, 2, 3, 4) can be applied to the channel by setting OFFSETy_EN = 1 in the ADC_OFRy register. The application can select the offset channel by bit OFFSETy_CH [4: 0] of the ADC_OFRy register. In this case, the conversion result is subtracted by the user-defined offset in OFFSETy [11: 0], and the result may be negative, so the read data is signed and the SEXT bit represents the extended sign value.

Note: offset is not supported in oversampling mode. When the ROVSE or JOVSE bit is set, the value of the OFFSETy_EN bit in the ADC_OFRy register is ignored (considered a reset).

The following table describes how to perform comparisons for all possible resolutions of offset.

Table 14-5 Offset calculated values vs Data resolution

Resolution (RES [1: 0])	**Raw conversion result and offset subtraction**		**Result**	**Description**
	Original conversion result Left alignment	**Offset**		
00: 12 bit	DATA [11: 0]	OFFSET [11: 0]	Signed 12bit number	-
01: 10bit	{DATA [11: 2], 00}	OFFSET [11: 0]	Signed 10bit number	The user must configure OFFSET [1: 0] = 00
10: 8 bit	{DATA [11: 4], 0000}	OFFSET [11: 0]	Signed 8-bit number	The user must configure OFFSET [3: 0] = 0000
11: 6bit	{DATA [11: 6], 000000}	OFFSET [11: 0]	Signed 6-bit number	The user must configure OFFSET [5: 0] = 000000

When reading data from ADC_DR (regular channel) or ADC_JDRy (injection channel, y = 1, 2, 3, 4) corresponding to channel "i":

- If one of the offsets of the corresponding channel is enabled (bit OFFSETy_EN = 1), the data is read signed.
- If none of the four offsets for this channel are enabled, the read data is unsigned.

The following figure shows signed and unsigned data alignment.

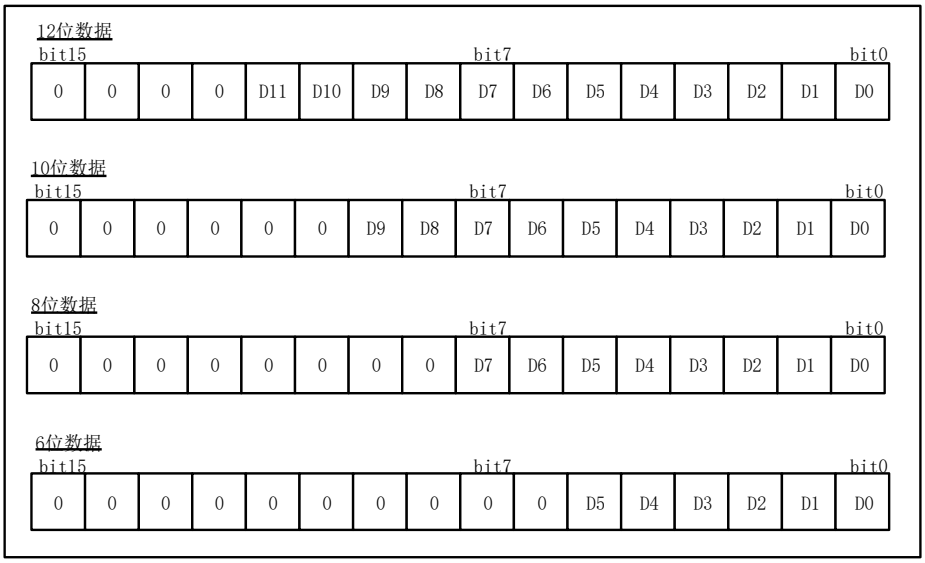


Figure 14-9 right-aligned (offset not enabled, unsigned number)

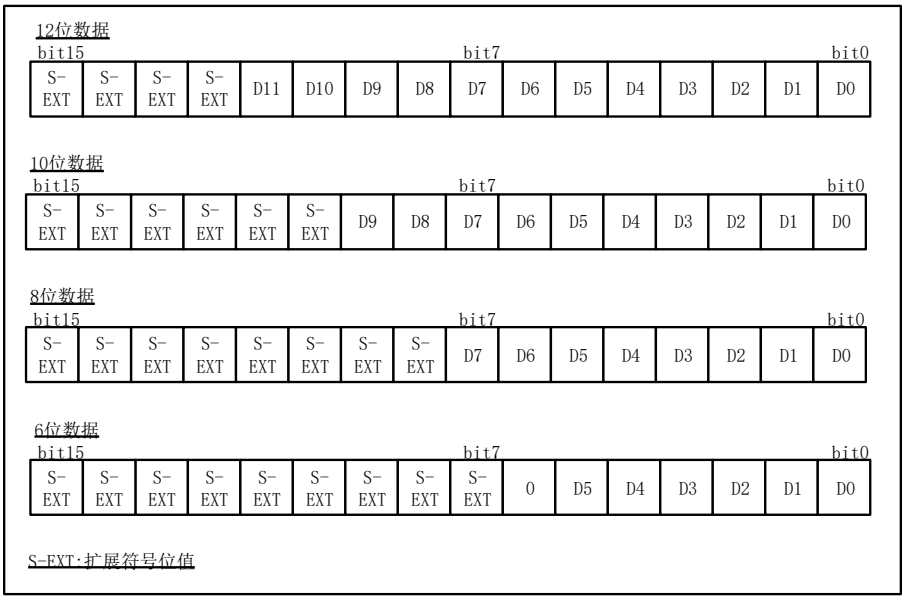


Figure 14-10 right aligned (offset enabled, signed number)

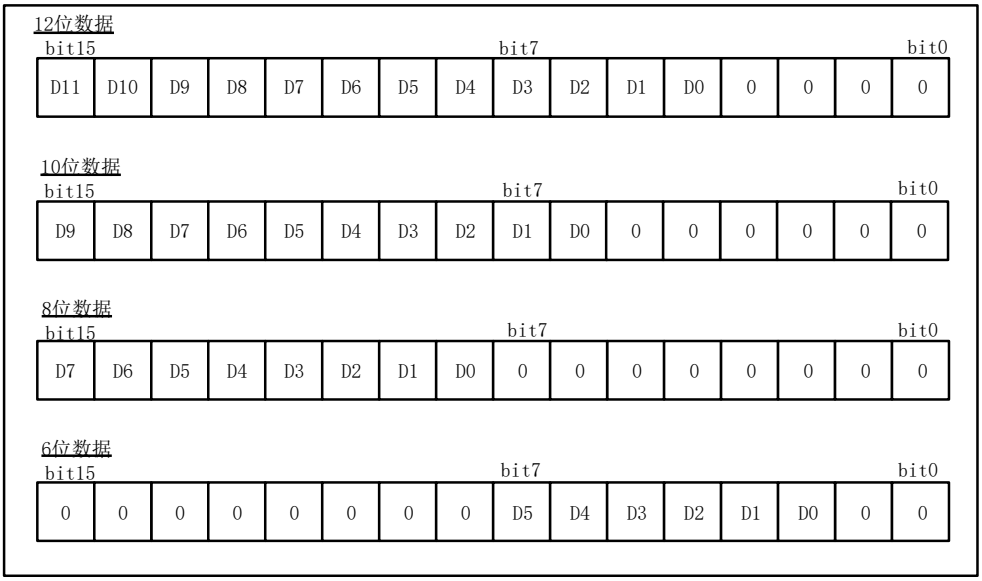


Figure 14-11 left aligned (offset not enabled, unsigned number)

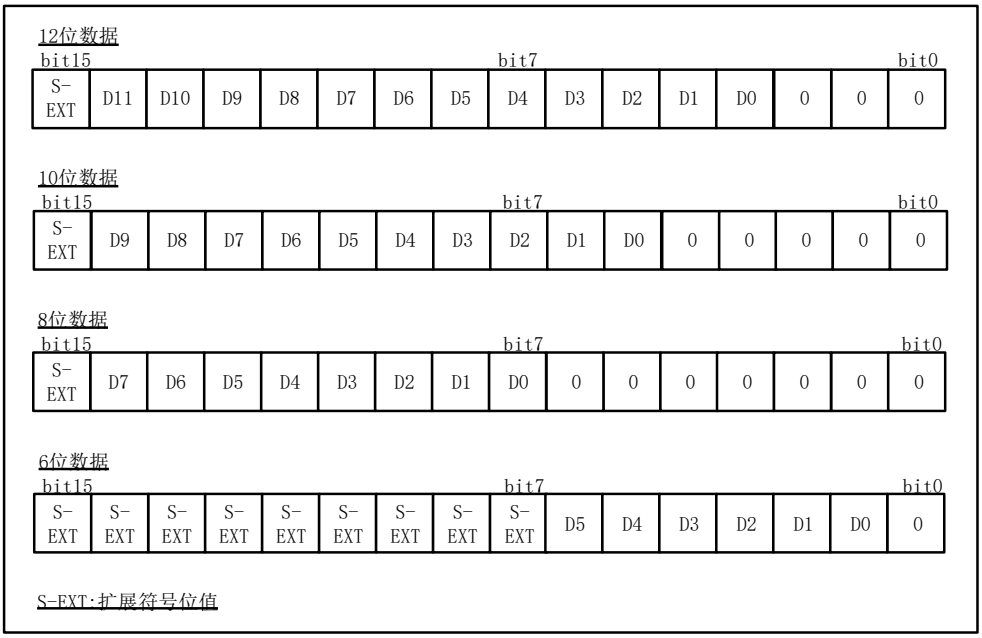


Figure 14-12 left aligned (offset enabled, signed number)

14.3.15.3 Gain compensation

When the ADC_CFGR2 register GCOMP bit is set, all converted data is gain compensated. After each conversion, the data will be calculated using the following formula.

DATA = DATA (adc result) x (GCOMPCOEFF)/4096

Since GCOMPCOEFF ranges from 0-16383, the actual gain compensation factor ranges from 0-3.999756.

The calculated LSB − 1 values are rounded and the error is minimized before the resulting data is stored in the RDATA or JDATAx register.

Gain compensation is also effective for oversampling. When gain compensation is used in the over-sampling mode, the gain calculation is performed after the accumulation and right shift operation to minimize power consumption (the gain calculation is only done once, not every transition).

14.3.15.4 Offset compensation

Whenthe offset isenabledwhilethe SATEN bit is set in the ADC_OFRy register, the data is unsigned. All offset data is saturated at 0x000 (at 12-bit resolution). When the OFFSETPOS bit is set, the offset direction is positive and the data is saturated at 0xFFF (at 12-bit resolution). In 8-bit resolution, the data is saturated at 0x00 and 0xFF, respectively.

After offset and gain compensation, an analog watchdog comparison is performed on the unsigned number. For proper watchdog operation, offset-compensated data must be in unsigned format (the SATEN bit in the ADC_OFRy register is set to 1).

14.3.16 Oversampler

The oversampling unit performs data preprocessing to offload the CPU computation load. It can handle multiple conversions and average them into a single data (increasing the data width up to 16 bits).

The oversampler provides the following formula, where N and M can be adjusted:

$$Result = \frac{1}{M} \times \sum_{n=0}^{n=N-1} Conversion(t_n)$$

The oversampler allows the following functions to be performed by hardware: averaging, reducing data rate, improving signal-to-noise ratio, basic filtering.

The oversampling rate N is defined by the OVSR [2: 0] bit in the ADC_CFGR2 register and can be adjusted in the range of 2x to 256x. The division coefficient M is implemented by a right shift, up to 8 bits. And is defined using the OVSS [3: 0] bit in the ADC_CFGR2 register.

The summing unit can produce a result of up to 20 bits (256 x 12-bit result), which is first shifted to the right. It is then truncated to 16 least significant bits before final transfer to the ADC_DR data register, rounding the least significant bits left by the shift to the nearest value.

Note: If the shifted intermediate result exceeds 16 bits, the result is truncated as is without saturation.

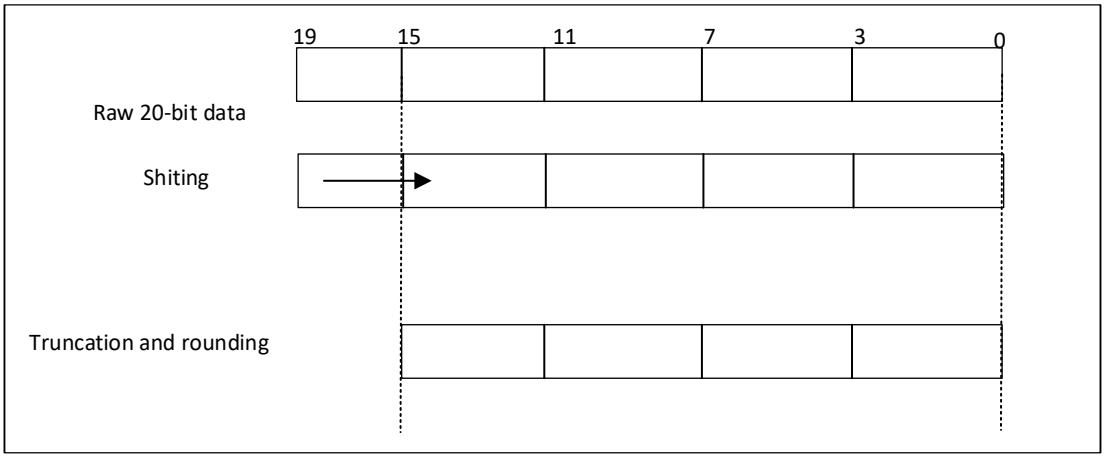


Figure 14-13 20 bits ~ 16 bits result truncation

The figure below gives a numerical example of processing, from the raw 20-bit accumulated data to the final 16-bit result.

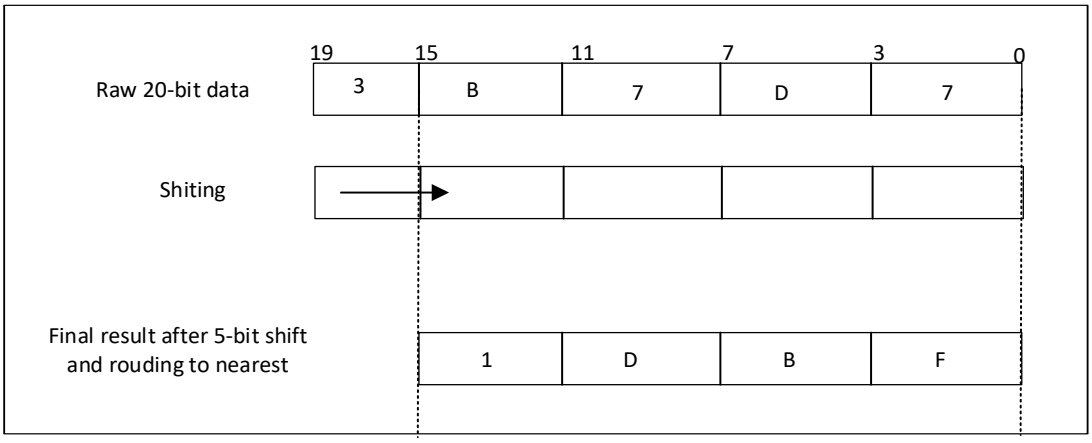


Figure. 14-14 Numerical example with 5-bit shift and rounding

The table below gives the data formats for various N and M combinations where the raw converted data is 0xFFF.

Table 14-6 Maximum Output Results vs. N and M (gray cells indicate truncation)

Over sampling ratio	Max Raw data	No-shift OVSS = 0000	1-bit shift OVSS = 0001	2-bit shift OVSS = 0010	3-bit shift OVSS = 0011	4-bit shift OVSS = 0100	5-bit shift OVSS = 0101	6-bit shift OVSS = 0110	7-bit shift OVSS = 0111	8-bit shift OVSS = 1000
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020

4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFF0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

There is no change in the transition timing in oversampling mode: the sampling times remain equal throughout the oversampling sequence. Every N transitions provide a new data, delayed by N x TCONV = N x (tSMPL + tSAR) equivalent. The flags are set as follows:

- The end of sampling phase (EOSMP) is set after each sampling phase
- End of Transition (EOC) occurs every N transitions when oversampling results are available
- The end of sequence (EOS) occurs after the completion of the oversampled data sequence (i.e. after the N x total sequence length conversion)

14.3.16.1 ADC operating modes not supported during oversampling (single ADC mode and dual ADC mode)

In oversampling mode, injection break rules are not supported.

14.3.16.2 Supported ADC operating modes when oversampling (single ADC mode)

In oversampling mode, most ADC operating modes remain the same:

- Single mode/continuous mode conversion
- Initiate ADC conversion via software or external trigger
- STOP ADC DURING TRANSFOR
- Read data via CPU/DMA
- Low Power Mode (AUTDLY)
- Programmable resolution:The result of the non-12-bits resolution conversion (according to the RES [1: 0] bit in the ADC_CFGR register) is accumulated, truncated, rounded, and shifted in the same manner as the 12-bit conversion.

Note: Data alignment is not available when using oversampled data. The ALIGN bit in ADC_CFGR is ignored and the data is always provided right-aligned. Offset correction is not supported in over-sampling mode. When the ROVSE or JOVSE bit is set, the value of the OFFSETy_EN bit in the ADC_OFRy register is ignored (treated as a reset).

14.3.16.3 Analog watchdog

The simulated watchdog functionality (AWD1SGL, AWD1EN, JAWD1EN) is maintained with the following differences:

- Ignore the RES [1: 0] bit and always use the full 12-bit values HT [11: 0] and LT [11: 0] for comparison
- Comparison is performed on the most significant 12 bits of the 16-bit oversampling result ADC_DR [15: 4]

Note: Caution must be taken when using high shift values, which reduces the comparison range. For example, if the oversampling result is shifted by 4 bits, resulting in 12 bits of data right-aligned, a valid

analog watchdog comparison can only be performed on 8 bits. A comparison is made between ADC_DR [11: 4] and HT [0: 7]/LT [0: 7], and HT [11: 8]/LT [11: 8] must be reset.

14.3.16.4 Trigger Mode

The averager can also be used for basic filtering. While not a very powerful filter (slow roll-off and limited stopband attenuation), it can be used as a notch filter to suppress constant parasitic frequencies (usually from mains or switched mode power supplies). For this purpose, a specific discontinuity mode can be enabled using the TROVS bit in ADC_CFGR2 in order to be able to have an oversampling frequency defined by the user and independent of the transition time itself.

The diagram below shows how the transition is initiated in response to a trigger in intermittent mode. If the TROVS bit is set, the contents of the DISCEN bit are ignored and treated as 1.

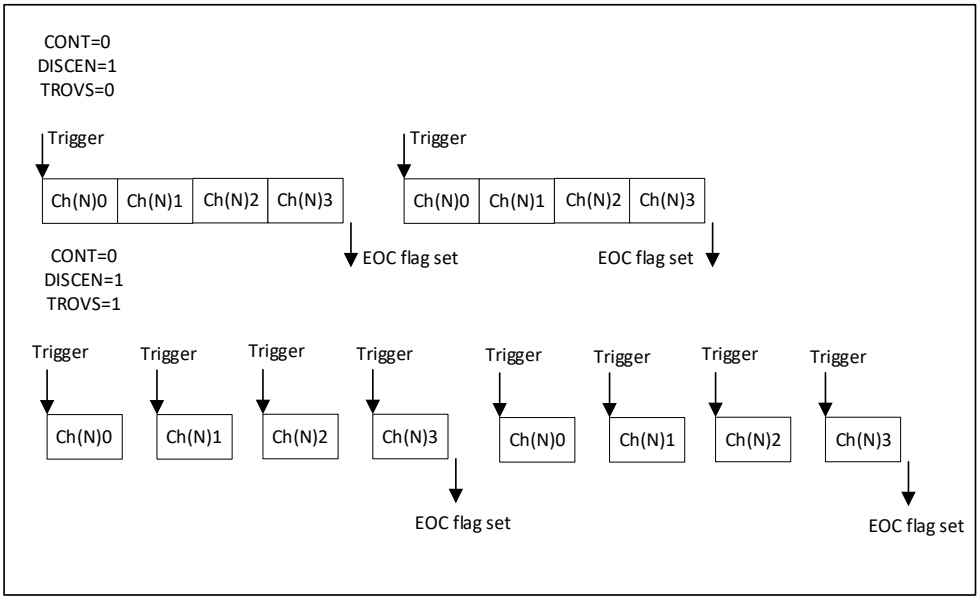


Figure 14-15 Oversampling mode triggering for rules (TROVS bit = 1)

14.3.16.5 Oversample regular channels only

The regular oversampling pattern bit ROVSM defines how a regular oversampling sequence is re-stored if the sequence is interrupted by the injected transition:

- In continuous mode, the accumulation restarts from the last regular channel valid data (the trigger of the injection causes the previous transition to abort). This ensures that oversampling will be done regardless of the injection frequency (providing at least one rule transition between triggers to be done);
- In recovery mode, accumulation restarts from 0 (previous conversion results are ignored). This mode allows to guarantee that all data used for oversampling is converted back-to-back within a single time slot. Care must be taken to make the injection trigger period longer than the over-sampling period length. If this condition is not obeyed, the oversampling will not be completed and the rule sequence will be blocked.

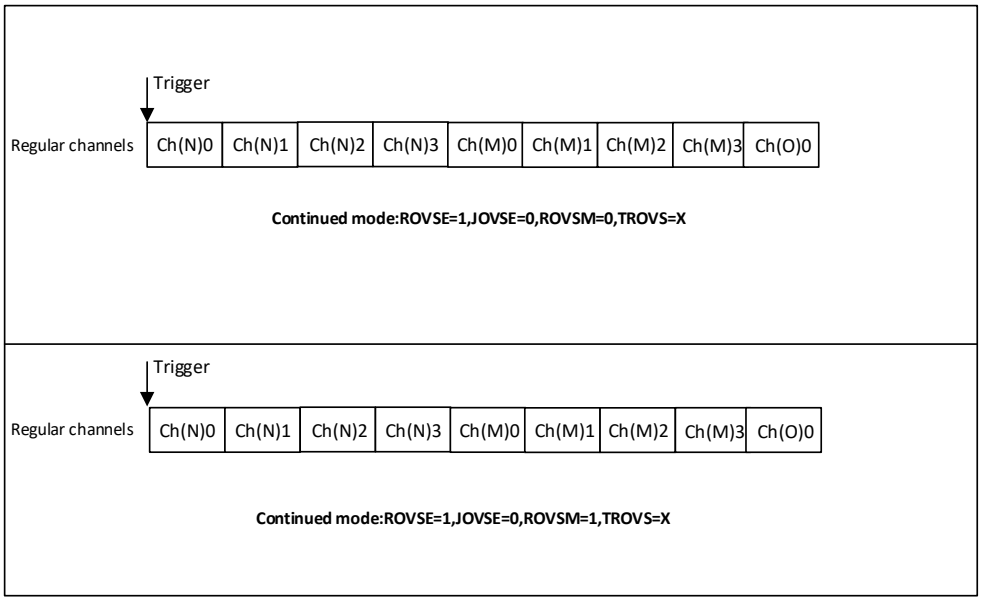


Figure 14-16 4x Example of oversampling rate

14.3.16.6 Oversample only the injection channel

The injection oversampling mode bit JOVSE enables oversampling only for transitions in the injection sequence.

14.3.16.7 Automatic injection mode

Automatically injected sequences can be oversampled and all conversion results stored in registers to save DMA resources. This mode is only available when both regular and injection oversampling are enabled: JAUTO = 1, ROVSE = 1, and JOVSE = 1, other combinations are not supported. The ROVSM bit is ignored in auto-injection mode. The following diagram shows the sequence of transformations.

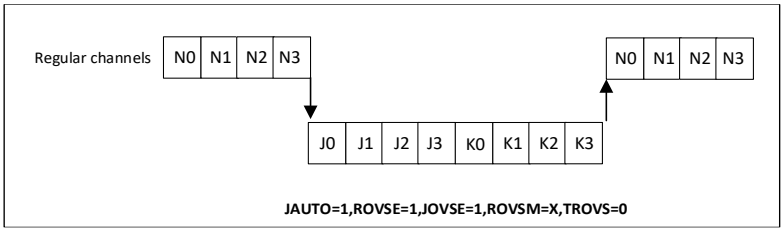


Figure 14-17 Automatic Injection Mode Oversampling

14.3.16.8 Supports dual ADC mode when oversampling

For synchronous injection mode and synchronous rule mode, oversampling can be enabled during dual ADC configuration. In this case, both ADCs must use exactly the same settings (including over-sampling).

When regular or injection oversampling is enabled (ROVSE = 1 or JOVSE = 1), all other dual ADC modes are not supported.

14.3.16.9 Summary of pattern combination

The table below summarizes all combinations, including unsupported modes.

Table 14-7 Summary of Oversampler Modes

Regular over-sampling ROVSE	Injection Over-sampling JOVSE	Oversampling mode ROVSM; 0 = contin-ued 1 = resumed	Trigger Rule Mode TROVS	Description
1	0	0	0	Regular continuous mode
1	0	0	1	-
1	0	1	0	Rule recovery model
1	0	1	1	Trigger Rule Recov-ery Mode
1	1	0	X	-
1	1	1	0	Injection and rule re-covery models
1	1	1	1	-
0	1	X	X	Injection over-sampling

14.3.17 Programmable sampling time

The ADC samples the input voltage over several ADC CLK cycles, and the number of sampling cycles is configured by SMP [2: 0] bits in the ADC_SMPR1 and ADC_SMPR2 registers. Each channel can be sampled using a different sampling time.

The total conversion time is calculated as follows:

tconv = sampling time + 12.5 cycles

Example:

fADC = 16 MHz And when sampling time = 3.5 cycles:

tconv = 3.5 + 12.5 = 16 cycles = 1 μs

14.3.18 Conversion on external trigger

Converting a single channel or converting a sequence can be triggered by software or external events (e.g. timer capture, input pins). If EXTEN [1: 0] (for regular transitions) or JEXTEN [1: 0] (for injection transitions) is not equal to 00, an external event is able to trigger transitions with the selected polarity. The rule software trigger selection is valid once the software sets the bit ADSTART = 1, while the injected software trigger selection is valid once the software sets the bit JADSTART = 1. Any hardware or software triggers that occur during the conversion process will be ignored.

Note: While STRT/JSTRT is valid, a rule trigger/injection trigger occurs and will be ignored.

The following table provides the correspondence between the EXTEN [1: 0] and JEXTEN [1: 0] values and the trigger polarity.

Table 14-8 Rule External Trigger Configuration Trigger Polarity

EXTEN [1: 0]	Source
00	Hardware trigger detection disabled, software trigger detec-tion enabled (rising edge)
01	Hardware triggering of rising edge detection
10	Hardware triggering of falling edge detection
11	Hardware triggered with detection on both rising and falling edges

Note: The polarity of rules and injection triggers cannot be changed at any time

Table 14-9 Injection External Trigger Configuration Trigger Polarity

JEXTEN [1: 0]	Source
00	Hardware trigger detection disabled, software trigger detec-tion enabled (rising edge)
01	Hardware triggering of rising edge detection

10	Hardware triggering of falling edge detection
11	Hardware triggered with detection on both rising and falling edges

Table 14-10 External Trigger Event Selection

ADC trigger selection Extsell [4: 0] or JEXTSEL [4: 0]	ADC triggers signals assignment			
	ADC1/2		ADC3	
	Regular	Injected	Regular	Injected
00000	tim1_cc1	tim1_trgo	tim3_cc1	tim1_trgo
00001	tim1_cc2	tim1_cc4	tim2_cc3	tim1_cc4
00010	tim1_cc3	tim2_trgo	tim1_cc3	tim2_trgo
00011	tim2_cc2	tim2_cc1	tim8_cc1	tim8_cc2
00100	tim3_trgo	tim3_cc4	tim3_trgo	tim4_cc3
00101	tim4_cc4	tim4_trgo	exti2	tim4_trgo
00110	exti11	exti15	tim4_cc1	tim4_cc4
00111	tim8_trgo	tim8_cc4	tim8_trgo	tim8_cc4
01000	tim8_trgo2	tim1_trgo2	tim8_trgo2	tim1_trgo2
01001	tim1_trgo	tim8_trgo	tim1_trgo	tim8_trgo
01010	tim1_trgo2	tim8_trgo2	tim1_trgo2	tim8_trgo2
01011	tim2_trgo	tim3_cc3	tim2_trgo	tim1_cc3
01100	tim4_trgo	tim3_trgo	tim4_trgo	tim3_trgo
01101	tim6_trgo	tim3_cc1	tim6_trgo	exti3
01110	tim15_trgo	tim6_trgo	tim15_trgo	tim6_trgo
01111	tim3_cc4	tim15_trgo	tim2_cc1	tim15_trgo
10000	tim9_cc1	tim2_cc3	tim18_cc3	tim18_cc4
10001	tim10_cc1	tim9_cc2	tim18_trgo	tim18_trgo
10010	tim11_cc1	tim13_cc1	tim18_cc1	tim9_cc1
10011	tim12_cc1	tim14_cc1	tim10_cc1	tim14_cc1
10100	tim18_trgo	tim12_cc2	tim13_cc1	tim18_cc2
10101	tim18_cc2	tim18_trgo	tim18_cc1	tim7_trgo
10110	tim19_cc1	tim18_cc1	tim7_trgo	Lptim_out
10111	tim7_trgo	tim16_cc1	Lptim_out	RES
11000	Lptim_out	tim7_trgo	RES	Res
11001	RES	Lptim_out	Res	Res
11010	Res	RES	Res	Res
11011	Res	Res	Res	Res
11100	Res	Res	Res	Res
11101	Res	Res	Res	Res
11110	Res	Res	Res	Res
11111	Res	Res	Res	Res

14.3.19 Configurable resolution

Fast conversion can be performed by reducing the ADC resolution. The RES bit of the ADC_CFGR register is used to select the number of bits available in the data register. The minimum conversion time for each resolution is as follows:

- 12 bits: 3.5 + 12.5 = 16 fADC cycles
- 10 bits: 3.5 + 10.5 = 14 fADC cycles
- 8-bit: 3.5 + 8.5 = 12 fADC cycles
- 6 bits: 3.5 + 6.5 = 10 fADC cycles

14.3.20 DMA request

Because the value of the regular channel conversion is stored in a unique data register, DMA needs to be used when multiple regular channels are converted, which can avoid loss of data already stored in the ADC_DR register.

Only the end of conversion of a regular channel generates a DMA request, which allows the transfer of its converted data from the ADC_DR register to the destination location selected by the user In the dual ADC mode, the data converted by the ADC2 is transmitted using the DMA function of the ADC1.

14.3.21 Dual ADC Mode

In devices with 2 ADCs, dual ADC mode can be used. In the DUAL ADC mode, the conversion is initiated by the ADC1 software/external touch trigger, depending on the mode selected by the DUAL [2: 0] bit.

Note: In dual ADC mode, when the transition is configured to be triggered by an external event, the user must set it to trigger only the master ADC and the slave ADC to be software triggered, which prevents accidental triggering of the slave transition. However, external triggers of the master and slave ADCs must be activated simultaneously.

There are 6 basic conversion modes:

-Synchronous injection mode

-Synchronization rule mode

-Fast crossover mode

-Slow crossover mode

-Alternate trigger mode

-Standalone mode

You can also use the above patterns in combination in the following ways:

-Sync Injection Mode + Sync Rule Mode

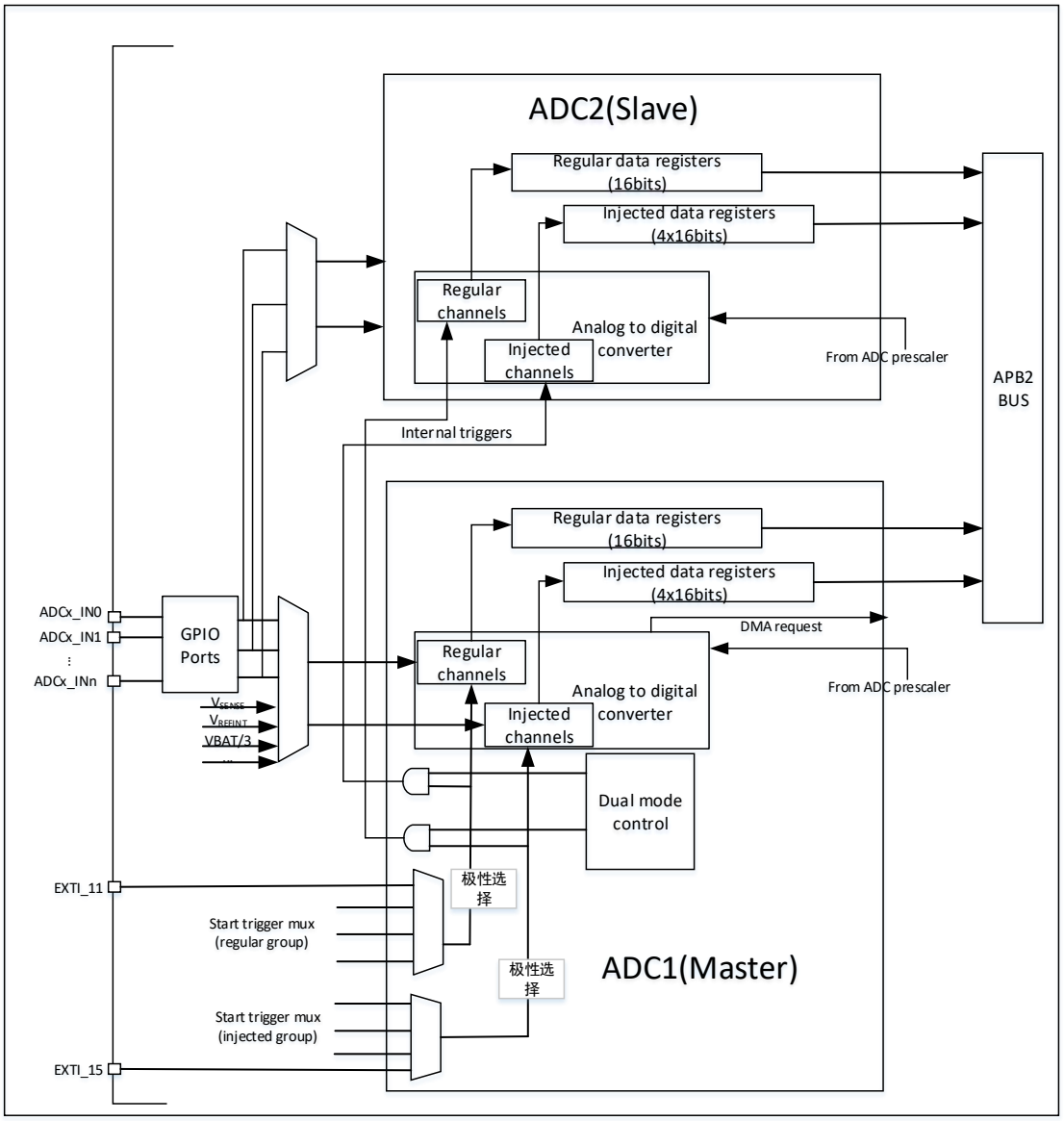


Figure 14-18 Dual ADC module diagram

1. An external trigger signal acts on ADC2, but it is not shown in this figure.

2. In dual ADC mode, the regular conversion data of ADC1 and ADC2 is included in the complete ADC1 data register (ADC1_DR).

14.3.21.1 Synchronous injection mode

In synchronous injection mode, ADC1 and ADC2 synchronously switch injection channel groups. The ADC_JSQR registers of ADC 1 and ADC 2 specify all conversion channels of an injection group sequence, where the JL [1: 0] bit of the ADC_JSQR register specifies the sequence length of the sequence.

When the ADEN bit is 1, once the ADC1 external trigger (selected by JEXTSEL [4: 0]) occurs, the ADC1 and ADC2 synchronization converts an injection sequence. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

Note:

1. Do not convert the same channel on two ADCs, that is, the sampling times of two ADCs on the same channel cannot overlap

2. The channel sampling time of ADC1 and ADC2 synchronous conversion should be consistent

At the end of the conversion of ADC1 or ADC2:

- The converted data is stored in the ADC_JDRx (x = 1-4)register of each ADC.
- When both ADC1/ADC2 injection channels are converted, the end of each channel conversion generates a JEOC event, and the end of the sequence generates a JEOS event.

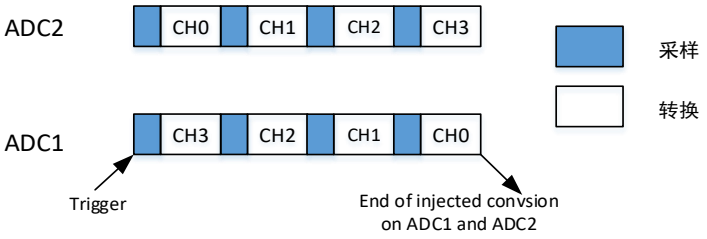


Figure 14-19 4 Synchronous injection mode on channels

14.3.21.2 Synchronization rule pattern

When synchronizing rule mode, ADC1 and ADC2 synchronously convert rule channel groups. The ADC_SQRx (x = 1-4) register of ADC 1 and ADC 2 specifies a regular sequence of all conversion channels, where the L [1: 0] bit specifies the sequence length. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

When the ADEN bit is 1, once the ADC1 external trigger (selected by EXTSEL [4: 0]) occurs, the ADC1 and ADC2 synchronization will convert a regular sequence. (Note: Users should not convert the same channel on two ADCs, that is, the sampling times of two ADCs on the same channel cannot overlap)

At the end of the conversion of ADC1 or ADC2:

- A 32-bit DMA transfer request is generated (if the DMA bit is set). The upper 16 bits of the 32-bit ADC1_DR register transferred to the SRAM contain the converted data of ADC2 and the lower 16 bits contain the converted data of ADC1.
- When all ADC1/ADC2 regular channels are converted, an EOC event is generated at the end of each channel conversion, and an EOS event is generated at the end of the sequence.

Notes:

1. In synchronization rule mode, sequences with the same length must be converted or the trigger interval must be longer than the longer sequence of the two sequences. Otherwise, when the conversion of the longer sequence is not completed, the ADC conversion with the shorter sequence will be restarted.
2. The channel sampling time of ADC1 and ADC2 synchronous conversion should be consistent.

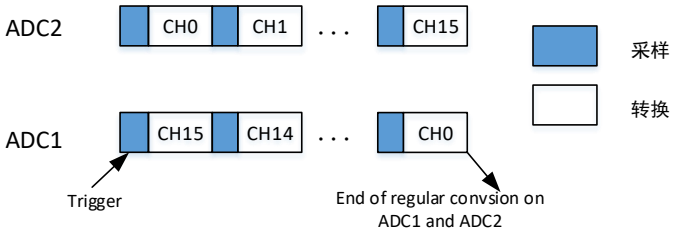


Figure 14-20 Synchronization rule pattern on 16 channels

14.3.21.3 Fast crossover mode

Fast crossover mode, ADC1 and ADC2 alternately switch one channel. This mode only applies to regular channel groups (usually one channel), and SQ1 of ADC 1 and ADC 2 defines the channel to be converted. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

When the ADEN bit is 1, once the ADC1 external trigger (selected by EXTSEL [4: 0]) occurs, ADC1 and ADC2 alternately transition a regular channel.

The external trigger source comes from the regular channel multiplexer of ADC1. After the external trigger is generated:

- ADC2 starts immediately
- ADC1 starts after 7 ADC clock cycles delay

If the CONT bits of ADC1 and ADC2 are set at the same time, the two selected ADC rule channels will be converted consecutively. The end of ADC1 conversion generates an EOC flag, which generates a 32-bit DMA transfer request (if the DMA bit is set). The 32-bit data of the ADC1_DR register is transferred to the SRAM. The upper 16 bits of ADC1_DR contain the converted data of ADC2, and the lower 16 bits contain the converted data of ADC1. The end of each channel transition produces an EOC event, and the end of the sequence produces an EOS event.

Note: The maximum allowable sampling time is < 7 ADCCLK cycles, so as to avoid the overlap of two sampling cycles when ADC1 and ADC2 convert the same channel. (Continuous Mode Maximum Sampling Time < 4 ADCCLK Cycles)

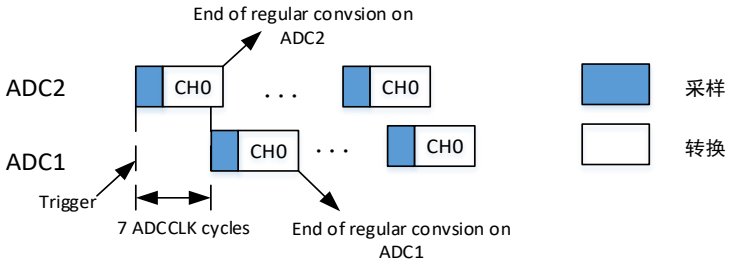


Figure 14-21 Fast crossover mode in continuous transition mode on one channel

14.3.21.4 Slow crossover mode

This mode is only available for regular channel groups (only one channel). The external trigger source comes from the regular channel multiplexer of ADC1. After the external trigger is generated:

- ADC2 starts immediately
- ADC1 starts after 14 ADC clock cycles delay
- After delaying the second 14 ADC cycles, ADC2 starts again, and so on.

Note: The maximum allowable sampling time is < 14 ADCCLK cycles to avoid overlap with the next transition.

After the ADC1 conversion is completed, an EOC flag is generated, and a 32-bit DMA transfer request is generated (if the DMA bit is set). The 32-bit data of the ADC1_DR register is transferred to the SRAM. The upper 16 bits of ADC1_DR contain the converted data of ADC2, and the lower 16 bits contain the

converted data of ADC1. The end of each channel transition produces an EOC event, and the end of the sequence produces an EOS event. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

The CONT bit cannot be set in this mode because it will continuously switch the selected regular channel.

Note: The application must ensure that when using crossover mode, no external trigger can be generated for the injection channel.

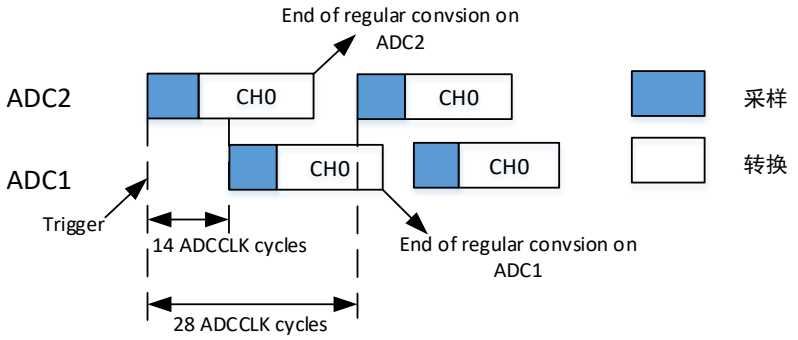


Figure 14-22 Slow crossover pattern of on one channel

14.3.21.5 Alternate trigger mode

This mode only applies to injection channel groups. The external trigger source comes from the injection channel multiplexer of ADC1. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

If the injection scan mode is used on ADC1 and ADC2:

- When the first trigger is generated, all injection group channels on ADC1 are switched.
- When the second trigger arrives, all injection group channels on ADC2 are switched.
- So cycling...

If a JEOS interrupt is allowed, a JEOS interrupt is generated after all ADC1 injection group channel transitions. The end of each channel transition produces a JEOC event, and the end of the sequence produces a JEOS event.

If allowed to generate a JEOS interrupt, generate a JEOS interrupt after all ADC2 injection group channel transitions. The end of each channel transition produces a JEOC event, and the end of the sequence produces a JEOS event.

When all injection group channels have been switched, if another external trigger is generated, the ADC1 injection group channel starts switching again.

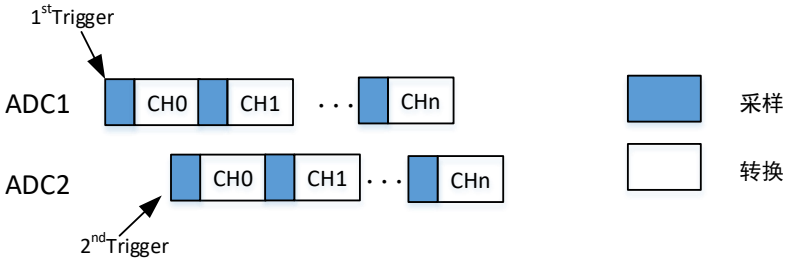


Figure 14-23 Alternate triggering-group of injection channels per ADC

If the injection interrupt mode is used on both ADC1 and ADC2:

- When the first trigger is generated, the first injection channel on ADC1 is switched.
- When the second trigger arrives, the first injection channel on ADC2 is switched.
- So cycling...

If a JEOS interrupt is allowed, a JEOS interrupt is generated after all ADC1 injection group channel transitions. The end of each channel transition produces a JEOC event, and the end of the sequence produces a JEOS event.

If allowed to generate a JEOS interrupt, generate a JEOS interrupt after all ADC2 injection group channel transitions. The end of each channel transition produces a JEOC event, and the end of the sequence produces a JEOS event.

When all injection group channels have been converted, if another external trigger is generated, the alternate trigger process is restarted.

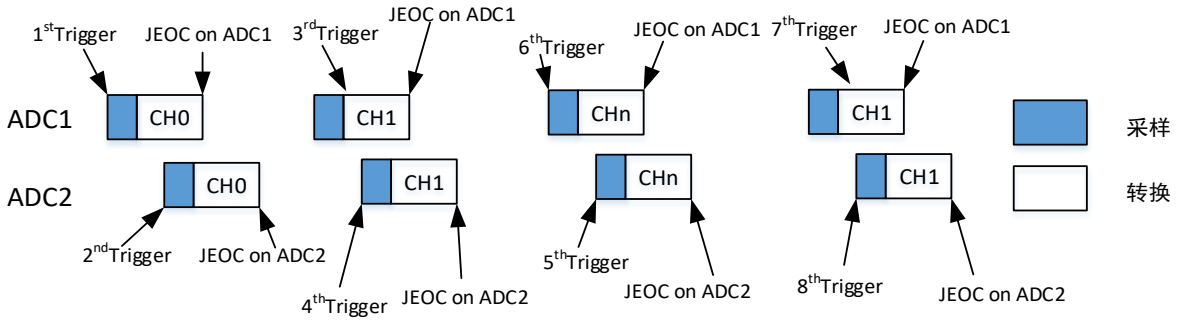


Figure 14-24 Alternate triggering-4 injection channels on each ADC in discontinuous mode

14.3.21.6 Standalone mode

In this mode, each ADC works independently and is not related to each other.

14.3.21.7 Hybrid rule/injection synchronization pattern

The rule group synchronization transition may be interrupted to initiate the synchronization transition of the injection group. When the sampling is finished, EOSMP is generated, and if EOSMPIE is set, an interrupt is generated.

Notes:

1. In the hybrid regular/injection synchronization mode, sequences with the same time length must be converted or the trigger interval must be longer than the longer sequence of the two sequences, otherwise ADC conversion with shorter sequence may be restarted when the conversion of the longer sequence has not been completed.
2. The channel sampling times of ADC1 and ADC2 synchronous conversion remain consistent

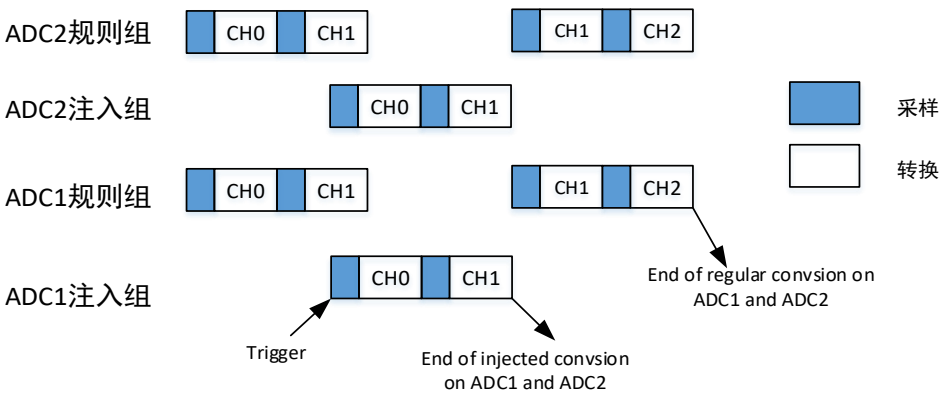


Figure 14-25 Mixed Rule/Injection Synchronization Mode-scan Mode

14.3.22 Temperature sensor and internal reference voltage

Temperature sensors may be used to measure device node temperature (TJ). The temperature sensor is internally connected to the ADC1_CH16 channel, which converts the voltage output by the sensor into a digital value. When not in use, the sensor can be put in power down mode.

The output voltage of the temperature sensor changes linearly with the temperature. Due to the change of the production process, the offset of the temperature change curve will be different on different chips. The internal temperature sensor is more suited to applications that detect temperature variations instead of absolute temperatures. If accurate temperature readings are needed, an external temperature sensor part should be used.

Note: The VSENSESEL, VREFEN bits must be set to activate the transition of the internal channels: ADC1_CH16 (temperature sensor) and ADC1_CH18 (VREFINT).

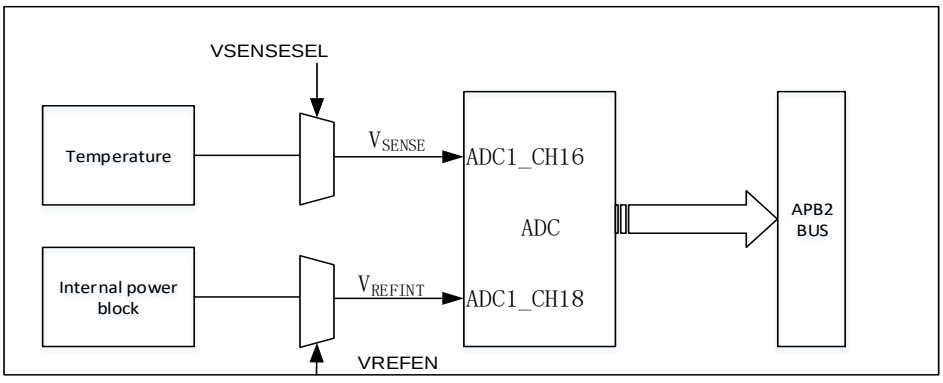


Figure 14-26 Temperature Sensor Channel Block Diagram

Main features

Reading the temperature

To use the sensor:

1. Select ADC1_CH16
2. Select a sampling time that is greater than the minimum sampling time specified in the datasheet.
3. VSENSESEL position 1 in the ADC1_CCR register to wake up the temperature sensor from power-down mode.
4. Start the ADC conversion by setting the ADEN bit (or by external trigger).
5. Read the resulting VSENSE data in the ADC data register
6. Calculate the temperature using the following formula:

Temperature (°C) = {(VSENSE-V30)/Avg_Slope} + 30 °C

with:

V30 = VSENSE value at 30 °C

Avg_Slope = Average Slope for curve between Temperature vs. VSENSE (given in mV/° C or µV/ °C). (For information on actual values of V30 and Avg_Slope, see the Electrical Characteristics section in the datasheet.)

Note: It takes a settling time for the sensor to wake up from power-down mode, and the correct VSENSE is output after the startup time. The ADC also requires a setup time after power-up, so in order to shorten the delay, ADEN and VSENSESEL should be positioned at position 1 at the same time.

14.3.23 Battery monitoring

Because the VBAT voltage can vary, the VBAT pin needs to be internally connected to the bridge distributor to ensure the correct operation of the ADC.

The bridge automatically enables VBAT/3 to connect to the ADC1/3_CH17 input channel.

14.3.24 ADC interrupts

For each ADC, the following interrupts are generated:

- At the end of any transition of the rule group (flag EOC)
- At the end of a sequence of rule groups (flag EOS)
- At the end of any transition of the injection group (flag JEOC)
- At the end of an injection group sequence (flag JEOS)
- When simulated watchdog detection occurs (flags AWD1, AWD2, and AWD3)
- When the sampling phase is over (flag EOSMP)

Independent interrupt enable bits have flexibility.

Table 14-11 Interrupts per ADC

Interrupt event	Event flag	Enable control bit
A rule conversion ends	EOC	EOCIE
A rule sequence ends	EOS	EOSIE
An injection transition ends	JEOC	JEOCIE
An injection sequence ends	JEOS	JEOSIE
Simulated Watchdog 1 State Position bit	AWD1	AWD1IE
Simulated Watchdog 2 status position bit	AWD2	AWD2IE
Simulated Watchdog 3 State Position bit	AWD3	AWD3IE
End of sampling phase	EOSMP	EOSMPIE

14.4 Register Description (for each ADC)

ADC1 register base address: 0x4001_2400

ADC2 register base address: 0x4001_2800

ADC3 register base address: 0x4001_3C00

14.4.1 ADC interrupt and status register (0x00: ADC_ISR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	STRT	JSTRT	Res	AWD3	AWD2	AWD1	JEOS	JEOC	Res	EOS	EOC	EOSMP	Res
-			R	R	-	RC_W1					-	RC_W1	RC_W1	RC_W1	-

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	STRT	R	0	Regular channel Start flag This bit is set by the hardware at the beginning of the conversion and cleared after the conversion is completed. 0: No regular channel conversion started 1: Regular channel conversion has started
11	JSTRT	R	0	Injected channel Start flag This bit is set by the hardware at the beginning of the conversion and cleared after the conversion is completed. 0: No injected group conversion started 1: njected group conversion has started
10	Reserved	-	-	-
9	AWD3	RC_W1	0	Simulated Watch Dogs 3 logo. This bit is set by hardware when the converted voltage exceeds/falls below the values of fields HT3 [7: 0] and LT3 [7: 0] of the ADC_TR3 register. It is cleared by software writing 1. 0: No simulated watchdog 3 event occurred (or the software confirmed and cleared the flag event) 1: Simulated Watchdog 3 event occurs
8	AWD2	RC_W1	0	Analog Watch Dogs 2 logo. This bit is set by hardware when the converted voltage exceeds/falls below the fields HT2 [7: 0] and LT2 [7: 0] values of the ADC_TR2 register. It is cleared by software writing 1. 0: No simulated watchdog 2 event occurred (or the software confirmed and cleared the flag event) 1: Simulated Watchdog 2 event occurs
7	AWD1	RC_W1	0	Analog Watchdog 1 logo. When the converted voltage exceeds/below the values of the fields HT1 [7: 0] and LT1 [7: 0] of the ADC_TR1 register, this bit is set by hardware. It is cleared by software writing 1. 0: No simulated watchdog 1 event occurred (or the software confirmed and cleared the flag event) 1: Simulated watchdog 1 event occurred
6	JEOS	RC_W1	0	Injects sequence conversion end flag. This bit is set by hardware when all channel transitions of the injection sequence are finished. It is cleared by software writing 1. 0: Injection sequence conversion is not complete (or flag event has been acknowledged and cleared by software) 1: Injection sequence conversion completed
5	JEOC	RC_W1	0	Injection channel transition end flag. This bit is set by the hardware when new data is accessible through the

				corresponding ADC_JDRy register at the end of each injection channel transition. Clear by software writing 1 or by reading the corresponding ADC_JDRy register. 0: Injection channel transition is not complete (or flag event has been acknowledged and cleared by software) 1: Injection channel conversion completed
4	Reserved	-	-	-
3	EOS	RC_W1	0	Rule sequence end flag The hardware sets this bit at the end of the rule sequence conversion. Software write 1 clear 0: Conversion sequence not complete (or the flag event was already acknowledged and cleared by software) 1: Conversion sequence complete
2	EOC	RC_W1	0	End of conversion flag The hardware sets this bit when a new data result can be read from the ADC_DR register after each transition of each regular channel. Software write 1 clears or read ADC_DR register clears 0: Conversion sequence not complete (or the flag event was already acknowledged and cleared by software) 1: Channel conversion completed
1	EOSMP	RC_W1	0	Sampling end flag. At the end of the sampling phase of each regular channel conversion, the hardware sets this bit and the software writes 1 to clear it 0: Not at the end of the sampling phase (or the flag event was already acknowledged and cleared by software) 1: End of sampling phase reached
0	Reserved	-	-	-

14.4.2 ADC interrupt enable register (0x04: ADC_IER)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						AWD3IE	AWD2IE	AWD1IE	JEOSIE	JEOCIE	Res	EOSIE	EOCIE	EOSMPIE	Res
-						RW	RW	RW	RW	RW	-	RW	RW	RW	-

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	AWD3IE	RW	0	Analog Watchdog 3 flag interrupt enabled. Set and clear by software to enable/disable the analog watchdog 3 flag interrupt. 0: Disable simulated Watchdog 3 event interrupt 1: Turn on simulated Watchdog 3 event interrupt
8	AWD2IE	RW	0	Analog Watchdogs 2 flag interrupt enabled. Set and clear by software to enable/disable analog Watchdog 2 flag interrupts. 0: Disable simulated Watchdog 2 event interrupt 1: Turn on simulated Watchdog 2 event interrupt
7	AWD1IE	RW	0	Analog Watchdog 1 flag interrupt enabled. Set and clear by software to enable/disable the analog watchdog 1 flag interrupt. 0: Disable simulated watchdog 1 event interrupt 1: Turn on simulated watchdog 1 event interrupt

6	JEOSIE	RW	0	Injection sequence conversion end interrupt enabled. End interrupt set and cleared by software to enable/disable injection sequence conversion. 0: JEOS interrupt disabled 1: JEOS interrupt not masked. When JEOS is set, an interrupt is generated.
5	JEOCIE	RW	0	Injection channel transition end interrupt enabled. This bit is set and cleared by the software to enable/disable the injection channel transition end interrupt. 0: JEOC interrupt disabled 1: JEOC interrupt not masked. When JEOC is set, an interrupt is generated. Note: The software is only allowed to write this bit if JADSTART = 0 (this ensures that no injection conversion is in progress).
4	Reserved	-	-	-
3	EOSIE	RW	0	End of rule sequence interrupt enable The bit is set and clear by that software to enable/disable the end interrupt of the sequence of rules 0: EOS interrupt disabled 1: EOS interrupt not masked. When EOS is set, an interrupt is generated. Note: The software is only allowed to write this bit if ADSTART = 0 (this ensures that no rule conversion is going on).
2	EOCIE	RW	0	Rule Transition End Interrupt Enable This bit is set and cleared by that software to enable/disable the rule transition end interrupt 0: EOC interrupt disabled 1: EOC interrupt not masked. When EOC is set, an interrupt is generated. Note: The software is only allowed to write this bit if ADSTART = 0 (this ensures that no rule conversion is going on).
1	EOSMPIE	RW	0	End of sampling interrupt enable This bit is set and clear by that software to enable/disable the end-of-sampling interrupt 0: EOSMP interrupt disabled 1: EOSMP interrupt not masked. When EOSMP is set, an interrupt is generated. Note: The software is only allowed to write this bit if ADSTART = 0 (this ensures that no rule conversion is going on).
0	Reserved	-	-	-

14.4.3 ADC Control Register (0x08: ADC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAL	Res	Res	Res	RSTCAL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
RS	-			RS	-										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ADSTP	JADSTART	ADSTART	Res	ADEN
-											RS	RS	RS	-	RW

Bit	Name	R/W	Reset Value	Function

31	ADCAL	RS	0	ADC calibration starts. The software sets to start ADC calibration, and the hardware will automatically clear it after the calibration is completed. Clear by hardware when calibration fails or calibration succeeds 0: Calibration completed 1: Write 1 calibration ADC, read 1 indicates calibration is in progress Note: Allow the software to initiate calibration by setting ADCAL only if ADEN = 0. Allow the software to update the calibration factor by writing ADC_CALFACT only when ADEN is invalid and there is no external trigger (ADC enabled and no conversion is in progress)
30: 28	Reserved	-	-	-
27	RSTCAL	RS	0	Reset calibration This bit is set by software and cleared by hardware. This bit is cleared after the calibration register is initialized (i.e., after RSTCAL is set to 1). 0: Calibration register initialized 1: Initialize calibration register Note: If RSTCAL is set when conversion is ongoing, additional cycles are required to clear the calibration registers.
26: 5	Reserved	-	-	-
4	ADSTP	RS	0	ADC stop conversion command This bit is set by the software to stop and discard the ongoing conversion (ADSTP command). When the transition is effectively discarded, it is cleared by hardware. After stopping the injection conversion, the ADC sequence and trigger can be reconfigured, and the ADC is ready to accept the new conversion (external trigger) 0: No ADC stops conversion command execution 1: Write 1 stops the ongoing conversion. Read 1 indicates that the ADSTP command is being executed.
3	JADSTART	RS	0	ADC software triggers start injection conversion This bit is set by software and is used to initiate ADC conversion of the injection channel. Start the conversion by software setting, and immediately after starting the conversion, it is cleared and set to 0 by hardware. 0: Reset state 1: Starts conversion of injected channels
2	ADSTART	RS	0	ADC software triggers start rule conversion This bit is set by the software and is used to initiate ADC conversion of the regular channel. Start the conversion by software, and it is cleared and set to 0 by hardware immediately after starting the conversion. 0: Reset state 1: Start converting regular channels
1	Reserved	-	-	-
0	ADEN	RW	0	A/D converter ON / OFF The ADC converter is enabled. When it is set to 1, the ADC converter wakes up. There is a delay tSTAB from the time the converter is powered on to the start of conversion. When set to 0, the ADC converter is in a power-down state. 0: Disable ADC conversion/calibration and go to power down mode. 1: Enable ADC and to start conversion.

14.4.4 ADC configuration register (0x0C: ADC_CFGR)

Address offset: 0x0C

Reset value: 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res	AWD1CH [4: 0]					JAUTO	JAWD1EN	AWD1EN	AWD1SGL	Res	JDISCEN	DISCNUM [2: 0]			DISCEN
-	RW					RW	RW	RW	RW	-	RW	RW			RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALIGN	AUTDLY	CONT	Res	EXTEN [1: 0]		EXTSEL4	EXTSEL3	EXTSEL2	EXTSEL1	EXTSEL0	RES [1: 0]		SCAN	Res	DMAEN
RW	RW	RW	-	RW		RW	RW	RW	RW	RW	RW		RW	-	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 26	AWD1CH [4: 0]	RW	5'h0	Analog Watchdog 1 Channel Select Bit These bits are set and cleared by software. They select the input channel to be guarded by the Analog watchdog. 00000: ADC analog input Channel 0 00001: ADC analog input Channel 1 …… 01111: ADC analog input Channel 15 10000: ADC analog input Channel 16 10001: ADC analog input Channel 17 10010: ADC analog input Channel 18 10011: ADC analog input Channel 19 All other numerical values are reserved. Note: The ADC analog input channel corresponding to the selection of analog watchdog 1 channel is consistent with the selection of ADC channel. See relevant chapters for detailed correspondence.
25	JAUTO	RW	0	Automatic injected group conversion This bit is used to turn on or off the regular channel group conversion and automatically perform the injection channel group conversion after completion of the regular channel group conversion 0: Automatic injected group conversion disabled 1: Automatic injected group conversion enabled
24	JAWD1EN	RW	0	Injection channel analog watchdog 1 enabled. Turn on the analog watchdog on the injection channel. 0: Disable analog watchdog 1 on injection channel 1: Using analog watchdog 1 on the injection channel
23	AWD1EN	RW	0	Regular Channel Simulation Watchdog 1 enabled. Turn on the simulated watchdog on the regular channel. 0: Disable simulated watchdog 1 on regular channel 1: Use simulated watchdog 1 on regular channel
22	AWD1SGL	RW	0	Single Channel Watchdog 1 enabled. This bit is used to turn on or off the analog watchdog 1 on the channel defined by AWD1CH [4: 0]. 0: Use simulated watchdog 1 on all channels 1: Using analog watchdog 1 on a single channel
21	Reserved	-	-	-
20	JDISCEN	RW	0	Discontinuous mode on injected channels. This bit is used to turn on or off the discontinuous mode on the injection channel group. 0: Discontinuous mode on injected channels disabled 1: Discontinuous mode on injected channels enabled
19: 17	DISCNUM [2: 0]	RW	3'h0	Discontinuous mode channel count These bits are written by software to define the number of regular channels to be converted in discontinuous mode after receiving an external trigger. 000: 1 channel 001: 2 channels … 111: 8 channels
16	DISCEN	RW	0	Discontinuous mode on regular channels This bit is used to turn on or off the discontinuous mode on the regular channel group 0: Discontinuous mode on regular channels disabled 1: Discontinuous mode on regular channels enabled
15	ALIGN	RW	0	Data alignment control bit 0: Right alignment 1: Left alignment
14	AUTDLY	RW	0	Wait mode conversion This bit is set and cleared by the software to enable/disable automatic delay transition mode. 0: Wait conversion mode off 1: Wait conversion mode on
13	CONT	RW	0	Continuous conversion enable . If set conversion takes place continuously till this bit is reset. 0: Single conversion mode

				1: Continuous conversion mode
12	Reserved	-	-	-
11: 10	EXTEN [1: 0]	RW	2'h0	External trigger enabling and polarity selection for regular channels These bits are set and cleared by the software to select the external trigger polarity and enable triggering of the rule group. 00: Disable hardware trigger detection (conversion can be initiated by software) 01: Hardware trigger detection of rising edge 10: Hardware trigger detection of falling edge 11: Both rising and falling edges have hardware trigger detection Note: The software allows these bits to be written only if it ensures that no rule translation is in progress.
9: 5	Extsell [4: 0]	RW	5'h0	External event select for regular group These bits select the external event used to trigger the start of conversion of a regular group: The external trigger events of ADC1, ADC2 are as follows: (see 4.16 for details) 00000: Event 1 00001: Event 2 00010: Event 3 00011: Event 4 … 11111 reservation The ADC3 external trigger configuration is as follows: 00000: Event 1 00001: Event 2 00010: Event 3 00011: Event 4 … 11111 reservation
4: 3	RES [1: 0]	RW	2'h0	Resolution control bit. These bits are written by software to select the resolution of the conversion. 00: 12 bits 01: 10 bits 10: 8 bits 11: 6 bits
2	SCAN	RW	0	Scan mode This bit is used to turn scan mode on or off. In Scan mode, the inputs selected through the ADC_SQRx or ADC_JSQRx registers are converted. 0: scan mode disabled 1: scan mode enabled
1	Reserved	-	-	-
0	DMAEN	RW	0	DMA enable bit 0: DMA mode disabled 1: DMA mode enabled Note: In dual ADC mode, only ADC1 can generate DMA requests.

14.4.5 ADC configuration register 2 (0x10: ADC_CFGR2)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	CALNUM [2: 0]			Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	GCOMP
-	RW	RW	RW	-											RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	ROVSM	TROVS	OVSS [3: 0]				OVSR [2: 0]			OVSE	ROVSE
-					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 28	CALNUM [2: 0]	RW	3'h0	Average number of calibrations, default 000

				000: 1 calibration and average 001: 2 calibrations and average 010: 4 calibrations and averaging 011: 8 calibrations and averaging 100: 16 calibrations and averaging 101: 32 calibrations and averaging Others: reserve
27: 17	Reserved	-	-	-
16	GCOMP	RW	0	Gain compensation mode This bit is set and cleared by the software to enable gain compensation mode. 0: Gain compensation is not enabled in ADC operating mode 1: Gain compensation is enabled and applied to all channels Note: The software allows this bit to be written only if there is no conversion in progress
15: 11	Reserved	-	-	-
10	ROVSM	RW	0	Regular oversampling mode This bit is set and cleared by the software to select a regular oversampling mode. 0: Continue mode: When injection transition is triggered, oversampling stops temporarily and continues after the injection sequence (oversampling buffer is maintained during the injection sequence) 1: Recovery mode: When the injection transition is triggered, the current oversampling is aborted and resumed from the beginning after the injection sequence (the oversampling buffer is cleared when the injection sequence starts to transition) Note: The software allows this bit to be written only if there is no conversion in progress
9	TROVS	RW	0	Trigger rule oversampling This bit is set and cleared by the software to enable triggered oversampling 0: All oversampling transitions of channels are completed continuously after triggering 1: Each oversampling transition of the channel requires a new trigger Note: The software allows this bit to be written only if there is no conversion in progress
8: 5	OVSS [3: 0]	RW	4'h0	Oversampling shift This bit field is set and cleared by the software to define the right shift applied to the original oversampled result. 0000: No shift 0001: Shift by 1 bit 0010: shift by 2 bits 0011: shift by 3 bits 0100: Shift 4 bits 0101: Shift by 5 bits 0110: Shift 6 bits 0111: Shift by 7 bits 1000: shift 8 bits Other: Reserved Note: The software allows this bit to be written only if there is no conversion in progress
4: 2	OVSR [2: 0]	RW	3'h0	Oversampling rate This bit field is set and cleared by the software to define the oversampling rate. 000: 2x 001: 4x 010: 8x 011: 16x 100: 32 x 101: 64x 110: 128 x

				111: 256 x Note: The software allows this bit to be written only if there is no conversion in progress
1	OVSE	RW	0	Injection Oversampling Enable This bit is set and cleared by the software to enable injection oversampling. 0: Disable injection oversampling 1: Enable injection oversampling Note: The software allows this bit to be written only if there is no conversion in progress
0	ROVSE	RW	0	Regular Oversampling Enable Enable This bit is set and cleared by the software to enable regular oversampling. 0: Disable regular oversampling 1: Enable regular oversampling Note: The software allows this bit to be written only if there is no conversion in progress

14.4.6 ADC Sample Time Register 1 (0x14: ADC_SMPR1)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMPPLUS	Res	SMP9 [2: 0]			SMP8 [2: 0]			SMP7 [2: 0]			SMP6 [2: 0]			SMP5 [2: 0]	
RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP5 [2: 0]	SMP4 [2: 0]			SMP3 [2: 0]			SMP2 [2: 0]			SMP1 [2: 0]			SMP0 [2: 0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	SMPPLUS	RW	0	The sampling time adds one clock cycle 1: 2.5 ADC clock cycle sampling time becomes 3.5 ADC clock cycles for the ADC_SMPR1 and ADC_SMPR2 registers. 0: Sampling time hold set to 2.5 ADC clock cycles. Make sure that no conversion is in progress before allowing software to write this bit.
30	Reserved	-	-	-
29: 0	SMP [9: 0] [2: 0]	RW	30'h0	Channel x sampling time selection These bits are written in software to select the sampling time individually for each channel. The channel select bit must remain constant during the sampling period. 000: 2.5 ADC clock cycles 001: 6.5 ADC clock cycles 010: 12.5 ADC clock cycles 011: 24.5 ADC clock cycles 100: 47.5 ADC clock cycles 101: 92.5 ADC clock cycles 110: 247.5 ADC clock cycles 111: 640.5 ADC clock cycles

14.4.7 ADC Sample Time Register 2 (0x18: ADC_SMPR2)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	SMP19 [2: 0]			SMP18 [2: 0]			SMP17 [2: 0]			SMP16 [2: 0]			SMP15 [2: 0]	
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP15 [2: 0]	SMP14 [2: 0]			SMP13 [2: 0]			SMP12 [2: 0]			SMP11 [2: 0]			SMP10 [2: 0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29: 0	SMP [19:10] [2: 0]	RW	30'h0	Channel x sampling time selection These bits are written in software to select the sampling time individually for each channel. The channel select bit must remain constant during the sampling period.

				000: 2.5 ADC clock cycles 001: 6.5 ADC clock cycles 010: 12.5 ADC clock cycles 011: 24.5 ADC clock cycles 100: 47.5 ADC clock cycles 101: 92.5 ADC clock cycles 110: 247.5 ADC clock cycles 111: 640.5 ADC clock cycles

14.4.8 ADC Watchdog Threshold Register 1 (0x20: ADC_TR1)

Address offset: 0x20

Reset value: 0x0FFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	HT1 [11: 0]											
-				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	AWDFILT [2: 0]			LT1 [11: 0]											
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 16	HT1 [11: 0]	RW	12'hFFF	Simulated Watchdog 1 High Threshold Software configurable to define simulated watchdog 1 high threshold Note: The software allows this bit to be written only if there is no conversion in progress
15	Reserved	-	-	-
14: 12	AWDFILT [2: 0]	RW	3'h0	Analog watchdog filtering parameters This bit is set and cleared by the software. 000: No filtration 001: Two consecutive detections generate AWDx flag or interrupt … 111: Eight consecutive detections generate AWDx flags or interrupts Note: The software allows this bit to be written only if there is no conversion in progress
11/0	LT1 [11: 0]	RW	12'h0	Simulated Watchdog 1 Low Threshold Software can be configured to define simulated watchdog 1 low threshold Note: The software allows this bit to be written only if there is no conversion in progress

14.4.9 ADC watchdog threshold register 2 (0x24: ADC_TR2)

Address offset: 0x24

Reset value: 0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	HT2 [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res			Res	Res	Res	Res	LT2 [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23: 16	HT2 [7: 0]	RW	8'hFF	Simulated Watchdog 2 High Threshold Software configurable to define simulated watchdog 2 high threshold Note: The software allows this bit to be written only if there is no conversion in progress
15: 8	Reserved	-	-	-
7: 0	LT2 [7: 0]	RW	8'h0	Simulated Watchdog 2 Low Threshold Software configurable to define simulated watchdog 2 low threshold

				Note: The software allows this bit to be written only if there is no conversion in progress

14.4.10 ADC watchdog threshold register 3 (0x28: ADC_TR3)

Address offset: 0x28

Reset value: 0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	HT3 [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res			Res	Res	Res	Res	LT3 [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23: 16	HT3 [8: 0]	RW	8'hFF	Simulated Watchdog 3 High Threshold Software can be configured to define simulated watchdog 3 high threshold Note: The software allows this bit to be written only if there is no conversion in progress
15: 8	Reserved	-	-	-
7: 0	LT3 [8: 0]	RW	8'h0	Simulated Watchdog 3 Low Threshold Software can be configured to define simulated watchdog 3 low threshold Note: The software allows this bit to be written only if there is no conversion in progress

14.4.11 ADC Rule Sequence Register 1 (0x30: ADC_SQR1)

Address offset: 0x30

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			SQ4 [4: 0]					Res	SQ3 [4: 0]					Res	SQ2[4]
-			RW					-	RW					-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ2 [3: 0]				Res	SQ1 [4: 0]					Res	Res	L [3: 0]			
RW				-	RW					-		RW			

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28: 24	SQ4 [4: 0]	RW	5'h0	4th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 4th time in the regular translation sequence.
23	Reserved	-	-	-
22: 18	SQ3 [4: 0]	RW	5'h0	3rd transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 3rd in the regular conversion sequence.
17	Reserved	-	-	-
16: 12	SQ2 [4: 0]	RW	5'h0	Second transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 2nd time in the regular conversion sequence.
11	Reserved	-	-	-
10: 6	SQ1 [4: 0]	RW	5'h0	1st transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 1st in the regular translation sequence.
5: 4	Reserved	-	-	-
3: 0	L [3: 0]	RW	4'h0	Regular channel sequence length These bits are defined by software. These bits are written by software to define the total number of conversions in the regular channel conversion sequence. 0000: 1 conversion 0001: 2 conversions …… 1111: 16 conversions

Some channels are not physically connected and therefore must not be selected for conversion.

14.4.12 ADC rule sequence register 2 (0x34: ADC_SQR2)

Address offset: 0x34

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			SQ9 [4: 0]					Res	SQ8 [4: 0]					Res	SQ7[4]
-			RW					-	RW					-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ7 [3: 0]				Res	SQ6 [4: 0]					Res	SQ5 [4: 0]				
RW				-	RW					-	RW				

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28: 24	SQ9 [4: 0]	RW	5'h0	The 9th transformation of the rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 9th time in the regular translation sequence.
23	Reserved	-	-	-
22: 18	SQ8 [4: 0]	RW	5'h0	8th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 8th time in the regular conversion sequence.
17	Reserved	-	-	-
16: 12	SQ7 [4: 0]	RW	5'h0	7th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 7th time in the regular translation sequence.
11	Reserved	-	-	-
10: 6	SQ6 [4: 0]	RW	5'h0	6th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 6th time in the regular conversion sequence.
5	Reserved	-	-	-
4: 0	SQ5 [4: 0]	RW	5'h0	5th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 5th time in the regular conversion sequence.

Some channels are not physically connected and therefore must not be selected for conversion.

14.4.13 ADC rule sequence register 3 (0x38: ADC_SQR3)

Address offset: 0x38

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			SQ14 [4: 0]					Res	SQ13 [4: 0]					Res	SQ12[4]
-			RW					-	RW					-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ12 [3: 0]				Res	SQ11 [4: 0]					Res	SQ10 [4: 0]				
RW				-	RW					-	RW				

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28: 24	SQ14 [4: 0]	RW	5'h0	14th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 14th time in the regular conversion sequence.
23	Reserved	-	-	-
22: 18	SQ13 [4: 0]	RW	5'h0	13th Conversion of Rule Sequence

				These bits are written by software, and the channels (0 to 19) are assigned as the 13th time in the regular conversion sequence.
17	Reserved	-	-	-
16: 12	SQ12 [4: 0]	RW	5'h0	12th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 12th time in the regular conversion sequence.
11	Reserved	-	-	-
10: 6	SQ11 [4: 0]	RW	5'h0	11th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 11th time in the regular conversion sequence.
5	Reserved	-	-	-
4: 0	SQ10 [4: 0]	RW	5'h0	10th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 10th time in the regular conversion sequence.

Some channels are not physically connected and therefore must not be selected for conversion.

14.4.14 ADC rule sequence register 4 (0x3C: ADC_SQR4)

Address offset: 0x3C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res					SQ16 [4: 0]					Res	SQ15 [4: 0]				
-					RW					-	RW				

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10: 6	SQ16 [4: 0]	RW	5'h0	16th transformation of rule sequence These bits are written by software, and the channels (0 to 19) are assigned as the 16th time in the regular conversion sequence.
5	Reserved	-	-	-
4: 0	SQ15 [4: 0]	RW	5'h0	15th Conversion of Rule Sequence These bits are written by software, and the channels (0 to 19) are assigned as the 15th time in the regular conversion sequence.

Some channels are not physically connected and therefore must not be selected for conversion.

14.4.15 ADC Rule Data Register (0x40: ADC_DR)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADC2DATA [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDATA [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	ADC2DATA [15: 0]	R	16'h0	ADC2 Converted Data The software can read the values of these bits. -In ADC1: In dual mode, these bits contain regular channel data for ADC2 conversion. -In ADC2: reserved bits.
15: 0	RDATA [15: 0]	R	16'h0	Regular data The software can read the values of these bits. These bits are read only. They contain the conversion result from regular channel x. The data is left or right aligned

14.4.16 ADC injection sequence register (0x4C: ADC_JSQR)

Address offset: 0x4C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JSQ4 [4: 0]					Res	JSQ3 [4: 0]					Res	JSQ2 [4: 1]			
RW					-	RW					-	RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JSQ2[0]	Res	JSQ1 [4: 0]					JEXTEN [1: 0]		JEXTSEL [4: 0]					JL [1: 0]	
RW	-	RW					RW		RW					RW	

Bit	Name	R/W	Reset Value	Function
31: 27	JSQ4 [4: 0]	RW	5'h0	4th transformation of injection sequence These bits are written by software, and the channels (0 to 19) are assigned as the 4th time in the injection conversion sequence. These bits are defined by software. These bits define the number (0-19) of the fourth conversion channel in the conversion sequence. Note: Unlike regular translation sequences, if the length of JL [1: 0] is less than 4, the sequence order of the translation starts from (4-JL).
26	Reserved	-	-	-
25: 21	JSQ3 [4: 0]	RW	5'h0	3rd transformation of injection sequence These bits are written by software, and the channels (0 to 19) are assigned as the 3rd time in the injection conversion sequence.
20	Reserved	-	-	-
19: 15	JSQ2 [4: 0]	RW	5'h0	2nd transformation of injection sequence These bits are written by software, and the channels (0 to 19) are assigned as the second time in the injection conversion sequence.
14	Reserved	-	-	-
13: 9	JSQ1 [4: 0]	RW	5'h0	1st transformation of injection sequence These bits are written by software, and the channels (0 to 19) are assigned as the first time in the injection conversion sequence.
8: 7	JEXTEN [1: 0]	RW	2'h0	External trigger enablement and polarity selection of injection channel These bits are set and cleared by the software to select the external trigger polarity and enable triggering of the injection group. 00: Disable hardware trigger detection (conversion can be initiated by software) 01: Hardware trigger detection of rising edge 10: Software triggered detection of falling edge 11: Both rising and falling edges have hardware trigger detection Note: The software allows these bits to be written only if there is no injected conversion in progress.
6: 2	JEXTSEL [4: 0]	RW	5'h0	External event select for injected group The external trigger events for ADC1 and ADC2 are as follows: (see 4.16 for details) 00000: Event 1 00001: Event 2 00010: Event 3 00011: Event 4 … 11111 reservation The trigger settings for ADC3 are as follows: 00000: Event 1 00001: Event 2 00010: Event 3 00011: Event 4 … 11111 reservation
1: 0	JL [1: 0]	RW	2'h0	Injected sequence length These bits are written in software to define the total number of transitions in the injection channel transition sequence. 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions

Some channels are not physically connected and therefore must not be selected for conversion.

14.4.17 ADC offset y register (0x60 + 0x04 (y-1): ADC_OFRy)

Address offset: 0x60 + 0x04 * (y-1), (y = 1 to 4)

Reset value: 0x0000_0000

(SATEN and OFFSETPOS are only available to ADC_OFR1 and are shared by ADC_OFR1-4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSETy_EN	OFFSETy_CH [4: 0]					SATEN	OFFSETPOS	Res							
RW	RW					RW	RW	-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res				OFFSET [11: 0]											
-				RW											

Bit	Name	R/W	Reset Value	Function
31	OFFSETy_EN	RW	0	OFFSET y Enable This bit is written by software to enable or disable the offset value OFFSETy [11: 0]. Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress.
30: 26	OFFSETy_CH [4: 0]	RW	5'h0	Channel selection for data offset y These bits are written by software to define the channel to which the offset of OFFSETy [11: 0] will be applied. Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress. Some channels are not physically connected and therefore cannot be selected for the data offset y.
25	SATEN	RW	0	Saturation enabling This bit is set and cleared by the software to enable the offset to saturate at 0x000 and 0xFFF. 0: No saturation control, the offset result can be signed 1: Saturation enabled, offset result unsigned and saturated at 0x000 and 0xFFF Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress.
24	OFFSETPOS	RW	0	Positive offset This bit is set and cleared by the software to enable positive offset. 0: negative offset 1: positive offset Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress.
23: 12	Reserved	-	-	-
11/0	OFFSET [11: 0]	RW	12'h0	The data offset y of the OFFSETy_CH [4: 0] channel. These bits are written by software to define the offset y that is subtracted from the raw converted data at the time of converting a channel (which can be regular or injected). The channel to which data offset y is applied must be programmed in the bit OFFSETy_CH [4: 0]. The conversion result can be read from the ADC_DR (regular conversion) or the ADC_JDRyi register (injection conversion). Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress. If multiple offsets (OFFSETy) point to the same channel, only the offset with the lowest x value is considered for subtraction. For example, if OFFSET1_CH [4: 0] = 4 and OFFSET2_CH [4: 0] = 4, this is OFFSET1 [11: 0] subtracted when switching channel 4.

14.4.18 ADC Injection Data Register (0x80 + 0x04 (y-1): ADC_JDRy)

Address offset: 0x80 + 0x04 * (y-1), (y = 1 to 4)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JDATA [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	JDATA [15: 0]	R	16'h0	Injection transformation data These bits are read-only. The conversion result of the injection channel y is placed in this register. The data are left- or right-aligned.

14.4.19 ADC analog watchdog 2 configuration register (0xA0: ADC_AWD2CR)

Address offset: 0xA0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	AWD2CH [19:16]			
-												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWD2CH [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	AWD2CH [19: 0]	RW	20'h0	Analog watchdog 2 channel selection These bits are set and cleared by the software. They enable and select the input channel to be protected by the analog watchdog 2. AWD2CH [i] = 0: AWD2 does not monitor ADC analog input channel i AWD2CH [i] = 1: AWD2 monitors ADC analog output channel i Simulated Watchdog 2 is disabled when AWD2CH [19: 0] = 000.. 0 Note: The channel selected by AWD2CH must also be selected into the SQRi or JSQRi register. Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress. Some channels are not physically connected and must not be selected for analog watchdogs.

14.4.20 ADC analog watchdog 3 configuration register (0xA4: ADC_AWD3CR)

Address offset: 0xA4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	AWD3CH [19:16]			
-												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWD3CH [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	AWD3CH [19: 0]	RW	20'h0	Analog watchdog 3 channel selection These bits are set and cleared by the software. They enable and select the input channel to be protected by the analog watchdog 3. AWD3CH [i] = 0: AWD3 does not monitor ADC analog input channel i AWD3CH [i] = 1: AWD3 monitors ADC analog output channel i Simulated Watchdog 3 is disabled when AWD3CH [19: 0] = 000.. 0 Note: The channel selected by AWD3CH must also be selected into the SQRi or JSQRi register. Note: The software allows these bits to be written only if it is ensured that there is no conversion in progress. Some channels are not physically connected and must not be selected for analog watchdogs.

14.4.21 ADC differential mode select register (0xB0: ADC_DIFSEL)

Address offset: 0xB0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DIFSEL [19:16]			
-												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Difsell [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	DIFSEL [19: 0]	RW	20'h0	Differential mode for channels 19 to 0. These bits are set and cleared by the software. They allow the selection of whether the channel is configured in single-ended mode or differential mode. DIFSEL [i] = 0: Analog input channel i is configured in single-ended mode DIFSEL [i] = 1: Analog input channel i is configured in differential mode Note: The DIFSEL corresponding channel or connected to a single-ended i/O port or connected to an internal channel must maintain a reset value (single-ended input mode). The software allows these bits to be written only if the ADC is disabled.

14.4.22 ADC calibration factor register (0xB4: ADC_CALFACT)

Address offset: 0xB4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									CALON	CAPSUC	OFFSUC	Res		Res	Res
-									R	RC_W1	RC_W1	-			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res					Res								
-															

Bit	Name	R/W	Reset Value	Function
31: 23	Reserved	-	-	-
22	CALON	R	0	Calibration status bit. 1: ADC is being calibrated; 0: ADC calibration has ended or not started.
21	CAPSUC	RC_W1	0	Capacitance calibration status bit. Indicates whether the ADC capacitance calibration is successful. Hardware setting; Software clear 0;
20	OFFSUC	RC_W1	0	Offset calibration status bit. Indicates whether the ADC offset calibration was successful. Hardware setting; Software clear 0;
19: 0	Reserved	-	-	-

14.4.23 ADC gain compensation register (0xC0: ADC_GCOMP)

Address offset: 0xC0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res						
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		GCOMPCOEFF [13: 0]													
-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 0	GCOMPCOEFF [13: 0]	RW	14'h0	Gain compensation coefficient

				These bits are set and cleared by software to program the gain compensation factor. 00 100 0 000 0 000 0: Gain factor of 0.5 … 01 000 0 000 0 000 0: Gain factor of 1 10 000 0 000 0 000 0: Gain factor of 2 11 000 0 000 0 000 0: Gain factor of 3 … This coefficient is divided by 4096 to give a gain factor ranging from 0 to 3.999756. Note: This gain compensation is applied only when the GCOMP bit of the ADC_CFGR2 register is 1.

14.5 Register Description (for Common ADC)

14.5.1 ADC common control register (0x308: ADC_CCR)

Address offset: 0x308 (only ADC1 has this register)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	VREFSEL	PWRMODE			Res	VSENSESE	VREFEN	Res					
-			RW	RW			-	RW	RW	-					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								VREFBUFSEL [1: 0]		VREFBUFEN	DUAL [4: 0]				
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28	VREFSEL	RW	0	ADC reference voltage selection, 0: VREF + as reference voltage (VREF + connected to VCCA) 1: VREFBUF as reference voltage
27: 25	PWRMODE	RW	3'h0	ADC current mode, for ADC analog circuit comparator 000: 6uA (default) 001: 7uA 010: 8uA 011: 9uA 100: 2 uA 101: 3uA 110: 4uA 111: 5 uA
24	Reserved	-	-	-
23	VSENSESE	RW	0	VSENSE Selection This bit controls the setting and disabling of VSENSE by the software. 0: Temperature sensor channel disabled 1: Temperature sensor channel enabled
22	VREFEN	RW	0	VREFINT enabled This bit is set and cleared by the software to enable/disable the VREFINT channel. 0: VREFINT channel disabled 1: VREFINT channel enabled
21: 8	Reserved	-	-	-
7: 6	VREFBUFSEL	RW	2'h0	VREFBUF output voltage selection 00: 2.048 V 01: 2.5 V 10: 2.9 V 11: 1.024 V Note: (1.024 V VREFBUF accurate value storage address: 0x1FFF 76B0 2.048 V VREFBUF precise value storage address: 0x1FFF 76B4 2.5 V VREFBUF precise value storage address: 0x1FFF 76B8 2.9 V VREFBUF precise value storage address: 0x1FFF76BC For example: the value read from address 0x1FFF 76B0 is 0x1079, indicating that the exact value of VREFBUF is 1.079 V)

5	VREFBUFEN	RW	0	VREFBUF enabled, when VREFBUFEN = = 0, the voltage output is high impedance The software writes 0 and sets 0, writes 1 and sets 1, 0: Disable VREFBUF 1: VREFBUF enabled
4: 0	DUAL [4: 0]	RW	5'h0	Dual ADC mode selection These bits are written by software to select dual ADC mode. All ADCs Independent: 00000: standalone mode 00001 to 01001: Dual ADC mode, master-slave ADC co-operation 00001: combined synchronization rule + synchronization injection mode 00010 Reservation 00011 Reservation 00100 Reservation 00101: Synchronous injection mode only 00110: Sync Rule Mode Only 00111: Fast crossover mode 01000: Slow crossover mode 01001: Alternate trigger mode only All other combinations are reserved and must not be programmed Note: The software allows these bits to be written only if the ADC is disabled.

15. Digital-to-analog converter (DAC)

15.1 Introduction

The 12-bit buffered DAC channel can be used to convert digital signals into analog voltage signal outputs. The DAC can be configured in 8-bit or 12-bit mode, or can be used in conjunction with a DMA controller. When the DAC is operating in 12-bit mode, the data can be left justified or right justified. The DAC module has two output channels, each with a separate converter. In dual-DAC mode, the two channels can be converted independently, or they can be converted simultaneously and update the output of the two channels synchronously. The DAC can input the reference voltage VREF+ through the pin for more accurate conversion results.

15.1.1 Main Features

- 2 DAC converters: 1 output channel per converter
- Left or right data alignment in 12-bit mode
- Synchronized update capability
- Noise-wave generation
- Triangular-wave generation
- Dual DAC channel independent or simultaneous conversions
- DMA capability for each channel
- Support DMA underflow error detection
- Conversion on external trigger
- Input voltage reference, VREF+

15.1.2 CAN block diagram

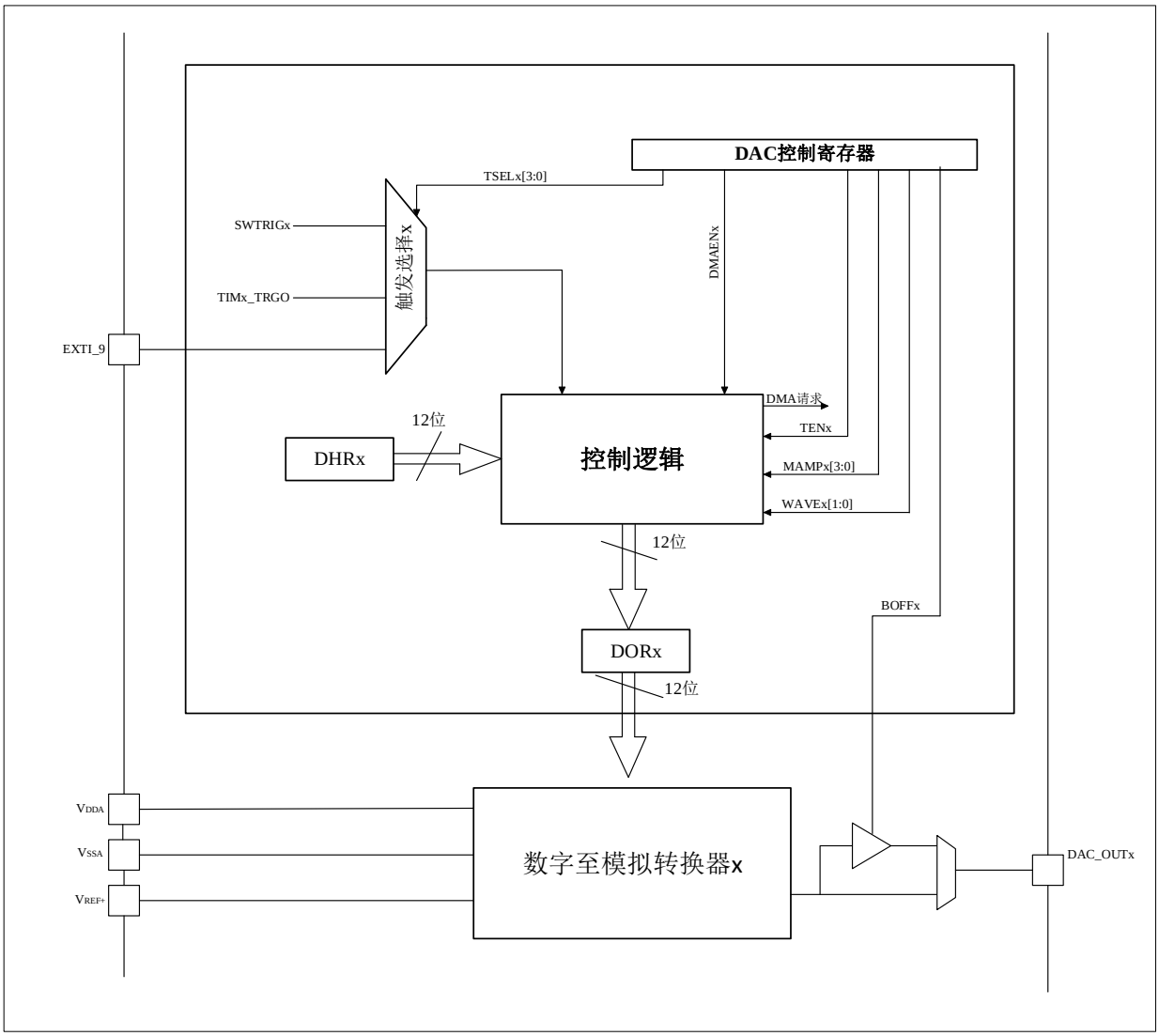


Figure 15-1 DAC module block diagram

Pin name	Signal type	Description
VCCA	Input, analog supply	Analog power supply
VSSA	Input, analog supply ground	Ground for analog power supply
VREF+	Input, analog positive reference voltage	DAC positive reference voltage, 2.3 V ≤ VREF + ≤ VCCA
DACx_OUT1	Analog output signal	DACx analog output

Note: Before enabling the DAC module, the GPIO port (PA4 corresponds to DAC1, PA5 corresponds to DAC2) should be configured with bit simulation mode.

15.2 DAC functional description

15.2.1 DAC channel enable

Each DAC channel can be powered on by setting its corresponding ENx bit in the DAC_CR register. After a startup time (tWAKEUP), the DAC channel x is enabled.

Note: The ENx bit enables the analog DAC channelx only. The DAC channelx digital interface is enabled even if the ENx bit is reset.

15.2.2 Enable DAC Output Cache

The DAC integrates two output buffers that can be used to reduce the output impedance,and to drive external loads directly without having to add an external operational amplifier. Each DAC channel output buffer can be enabled and disabled using the corresponding BOFFx bit in the DAC_CR register.

15.2.3 DAC data format

Depending on the selected configuration mode, the data has to be written in the specified register as described below:

Single DAC channelx, there are three possibilities:

- 8-bit data right-aligned: the user must write the data to register DAC_DHR8Rx [7: 0] bits (actually stored in register DAC_ DHRx [11: 4] bits)
- 12-bit data left alignment: user must write data to register DAC_DHR12Lx [15: 4] bits (actually stored in register DAC_ DHRx [11: 0] bits)
- 12-bit data right-aligned: user must write data to register DAC_DHR12Rx [11: 0] bits (actually stored in register DAC_ DHRx [11: 0] bits)

Depending on the operation of the DAC_DHRyyyx register(yyyrefers to different data formats and alignments), the written data is transferred tothe DAC_DHRxregister (DHRx is the internal data storage register x) after the corresponding shift. Subsequently, the contents of the DAC_ DHRx register are automatically transferred to the DAC_DORx register by a software trigger or an external trigger.

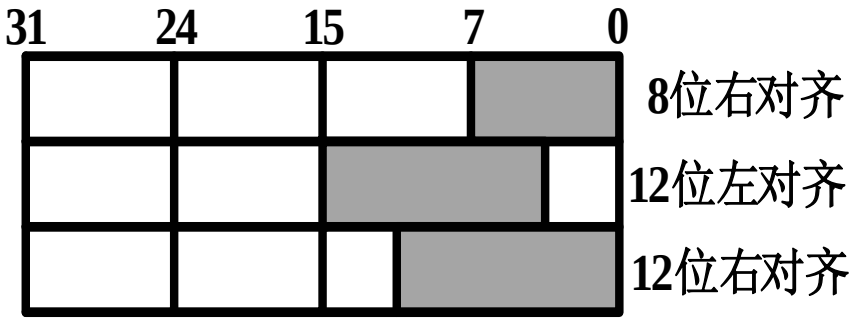


Figure 15-2 1 Data Register of DAC Channel Mode

Dual DAC channels, there are three possibilities:

- 8-bit right alignment: data for DAC channel1 to be loaded into DAC_DHR8RD [7:0] bits (stored into DAC_DHR1[11:4] bits) and data for DAC channel2 to be loaded into DAC_DHR8RD [15:8] bits (stored into DAC_DHR2[11:4] bits)
- 12-bit left alignment: data for DAC channel1 to be loaded into DAC_DHR12LD[15:4] bits (stored into DAC_DHR1[11:0] bits) and data for DAC channel2 to be loadedinto DAC_DHR12LD [31:20] bits (stored into DAC_DHR2[11:0] bits)

- 12-bit data right alignment: the user must write DAC channel 1 data into register DAC_DHR12RD [11: 0] bits (actually stored in registerDAC_ DHR1 [11: 0] bits) and DAC channel 2 data into register DAC_DHR12RD [27: 16] bits (actually stored in register DAC_ DHR2 [11: 0] bits)

Depending onthe operation of the DAC_DHRyyyD (yyyrefers to different data formats and alignments) registers, the written data is transferred to theDAC_ DHR1 andDAC_ DHR2 registersafter corresponding shifts (DAC_ DHR1 andDAC_ DHR2 are internal data holding registers). The DAC_DHR1 and DAC_DHR2 registers will then be loaded into the DAC_DOR1 and DAC_DOR2 registers, respectively, either automatically, by software trigger or by an external event trigger.

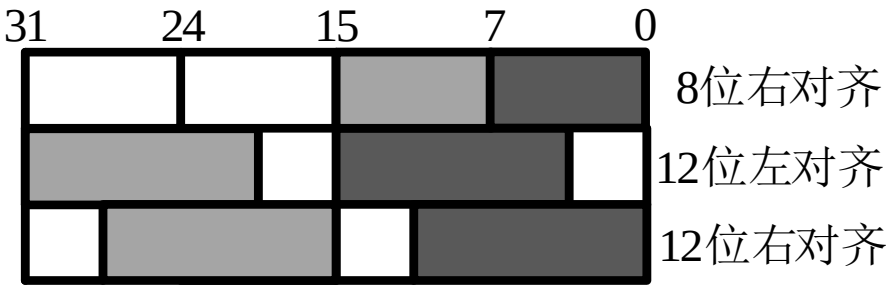


Figure 15-3 Data registers in dual-DAC channel mode

15.2.4 DAC conversion

Data cannot be written directly to the register DAC_DORx, any data output to the DAC channel x must be written tothe DAC_DHRxregister (data is actually written to the DAC_DHR8Rx, DAC_DHR12Lx, DAC_DHR12Rx, DAC_DHR8RD, DAC_DHR12LD, or DAC_DHR12RD registers).

If hardware trigger is not selected (TENx position '0' of register DAC_CR),the data stored in register DAC_ DHRx is automatically transferredtoregister DAC_DORxafter 1 APB1 clock cycle. If the hardware trigger is selected (TENx position '1' of the register DAC_CR), the data transfer is completed 3 APB1 clock cycles after the trigger occurs.

Once data is loaded fromthe DAC_DHRxregister into the DAC_DORx register,the output becomes active after the elapse of a setup time(tSETTLING), the length of whichvaries depending on the supply voltageand analog output load.

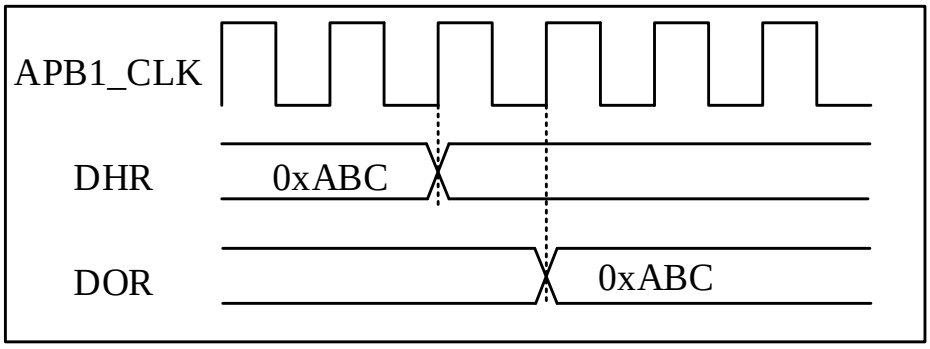


Figure 15-4 Timing diagram for conversion when TEN = 0

15.2.5 DAC output voltage

The digital input is linearly converted to an analog voltage output via DAC, which ranges from 0 to V_{REF+}.

The analog output voltages on each DAC channel pin are determined by the following equation:

DAC output = VREF + x (DOR/4096).

15.2.6 DAC trigger selection

If the TENx bit is set to 1, the DAC transition may be triggered by some external event (timing counter, external interrupt line, and software operation). The TSELx[3:0] control bits determine which one, out of 12 possible events, will trigger conversion.

Table 15-1 External triggers

TSELx [3: 0]	DAC trigger signal selection	
	DAC1	DAC2
0	sw	sw
1	tim8_trgo	tim8_trgo
2	tim7_trgo	tim7_trgo
3	tim15_trgo	tim15_trgo
4	tim2_trgo	tim2_trgo
5	tim4_trgo	tim4_trgo
6	exti9	exti9
7	tim6_trgo	tim6_trgo
8	tim3_trgo	tim3_trgo
9	tim18_trgo	tim18_trgo
10	tim1_trgo	tim1_trgo
11	tim5_trgo	tim5_trgo

Each time the DAC detects a rising edge output from the selected timer TRGO or external interrupt line 9 (event),the data most recently stored in the register DAC_ DHRx is transferred to the register DAC_DORx. After 3 APB1 clock cycles, the register DAC_DORx is updated to a new value.

If the software trigger is selected, the conversion starts once the SWTRIG bit is set. After data is transferred from the DAC_DHRx register to the DAC_DORx register, the SWTRIG bit is automatically cleared '0' by the hardware.

Note:

1. The TSELx [3: 0] bit can no longer be changed when ENx is' 1 '.

2. If software trigger is selected, only one APB1 clock cycle is required to transfer data from register DAC_DHRx to register DAC_DORx.

3.When the output cache is enabled (DAC_CR.BOFFx = 0),the trigger frequency cannot exceed 1 M; When the output buffer is turned off (DAC_CR.BOFFx = 1),the frequency of triggering cannot exceed 15 M.

15.2.7 DMA functionality

15.2.7.1 DMA request

Each DAC channel has a DMA capability. Two DMA channels are used to service DAC channel DMA requests.

If DAC_CR.DMAENx position '1', once an external trigger (not a software trigger) occurs, a DMA request will be generated after 3 clock cycles, and then the data of the DAC_DHRx register is transferred to the DAC_DORx register.

In dual DAC mode (i.e. using the DAC_DHR12RD or DAC_DHR12LD or DAC_DHR8RD register), only one bit in DAC_CR.DMAENx should be set.

15.2.7.2 DMA underrun

The DMA requests of the DAC do not have a buffer queue, so if the second external trigger arrives before receiving a reply to the first external trigger (the first request), no new DMA request will be issued and the DMA underflow flag DMAUDRx in the DAC_SR register will be set to "1", reporting an error. The DMA data transfer is then disabled and no more DMA requests are processed. The DAC channel continues to convert the original data.

The software shall clear the flag by writing "1" to the register DMAUDRx, zero the DMAEN position of the used DMA channel, and re-initialize the DMA and DAC channels in order to properly restart the DMA transfer. At the same time, the software should modify the DAC trigger conversion frequency to reduce the DMA workload to avoid the recurrence of DMA underflow.

For each DAC channel, an interrupt is also generated if its corresponding DMAUDRIEx bit in the DAC_CR register is enabled.

15.2.8 Noise generation

A linear feedback shift register (LFSR) may be utilized to generate pseudo noise of varying amplitude. Set the DAC_CR.WAVE [1: 0] bit to '01' to select the DAC noise generation function. The preloaded value in LFSR is 0xAAA According to a specific algorithm, the value of this register is updated after 3 APB1 clock cycles after each trigger event.

Setting the DAC_CR.MAMPx [3: 0] bit can mask some or all of the LFSR data, so that the resulting value of LFSR is added to the value of DAC_DHRx. If the resulting value overflows, the maximum value that the register can keep is written to DAC_DORx, and if the resulting value does not overflow, the value is written to DAC_DORx.

If the register LFSR value is 0x000, a '1' (anti-lock mechanism) is injected.

Setting the DAC_CR.WAVEx [1: 0] position '0' may reset the generation algorithm of the LFSR waveform.

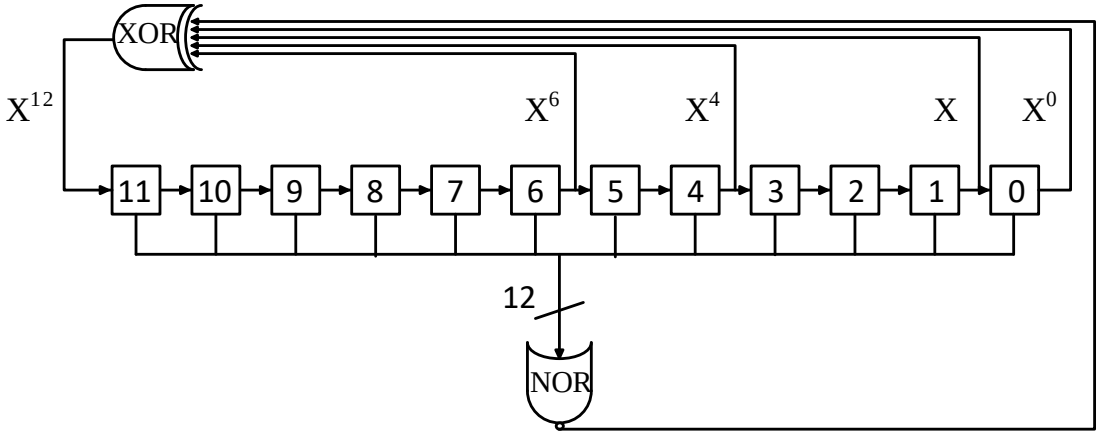


Figure 15-5 DAC LFSR register algorithm

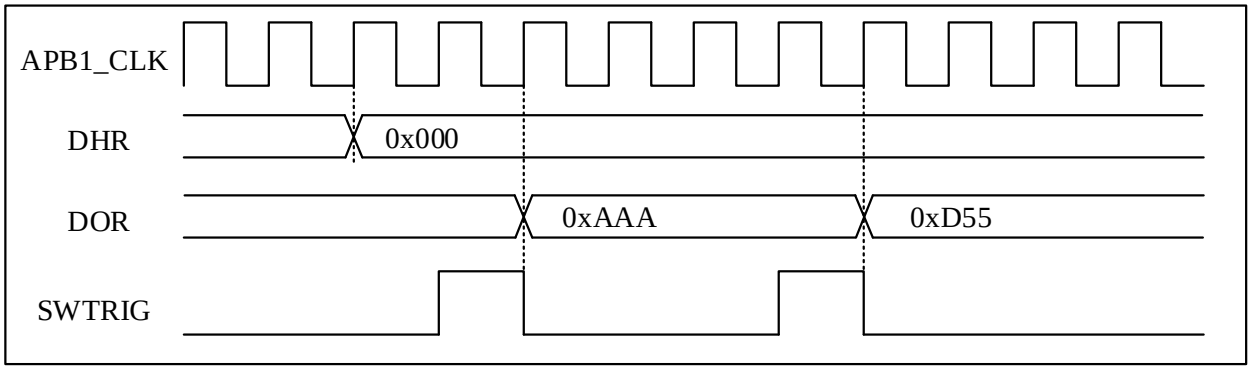


Figure 15-6 DAC conversion with LFSR waveform generation (software triggered)

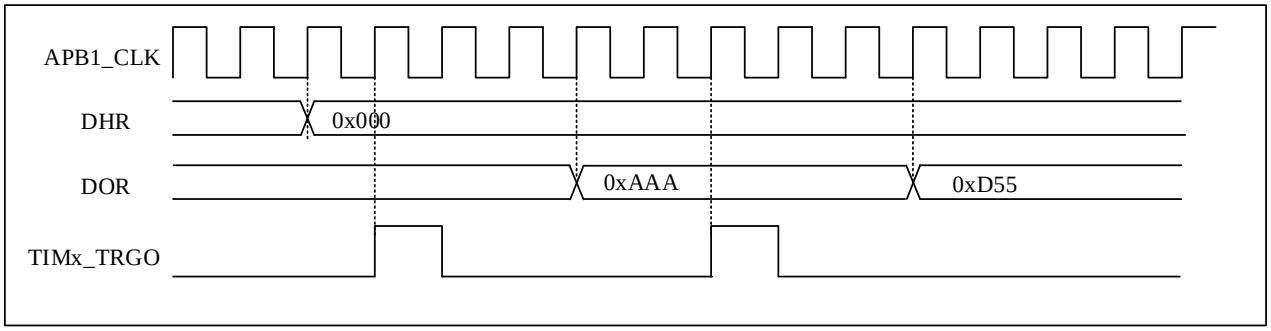


Figure 15-7 DAC conversion with LFSR waveform generation (hardware triggered)

Note:

1. In order to generate noise, the DAC trigger must be enabled, i.e. set the TENx bit of the DAC_CR register to '1';

2. If the value of DAC_DHR changes, the value of the LFSR register will be reset to 0xAAA;

3. If DAC_CR.TENxchanges from "1" to "0", the value of the LFSR register will also be reset to 0xAAA.

15.2.9 Triangle-wave generation

A small amplitude triangular wave can be added to DC or slowly changing signals. Set the DAC_CR.WAVEx [1: 0] bit to "10" or "11"to select the triangular wave generation function of the DAC. Set the DAC_CR.MAMPx[3: 0] bit to select the amplitude of the triangular wave. The internal triangular wave counter accumulates 1 for 3 APB1 clock cycles after each trigger event. Once the configured amplitude is reached, the counter is decremented down to 0, then incremented again and soon.

It is possible to reset triangle wave generation by resetting the DAC_CR.WAVEx[1:0] bits.

Note: When the base value is the maximum value of the register DAC_DORx, the output of DAC_DORx is the maximum value regardless of the value of MAMPx.

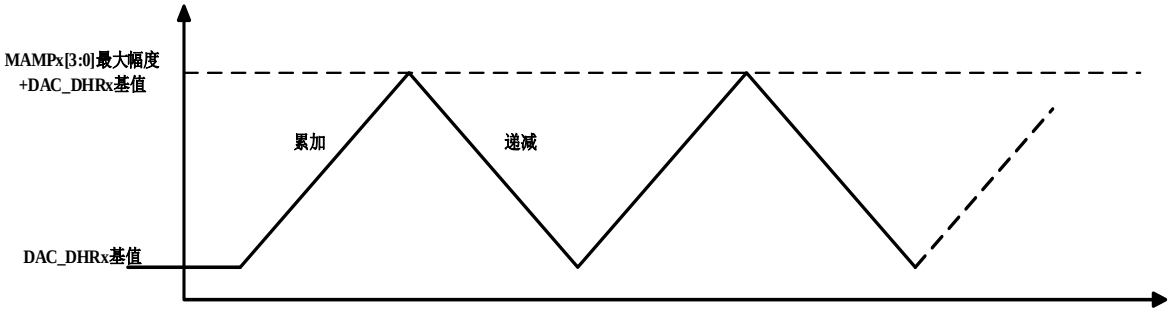


Figure 15-8 DAC Triangular Wave Generation

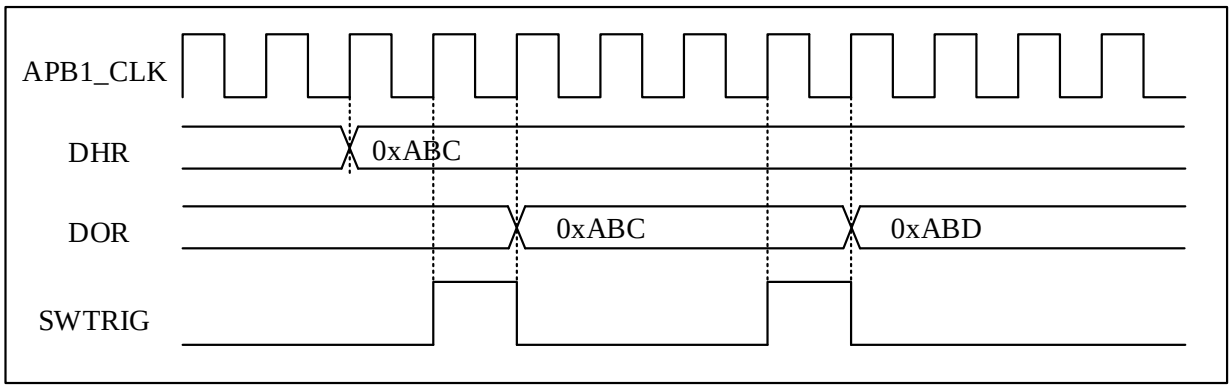
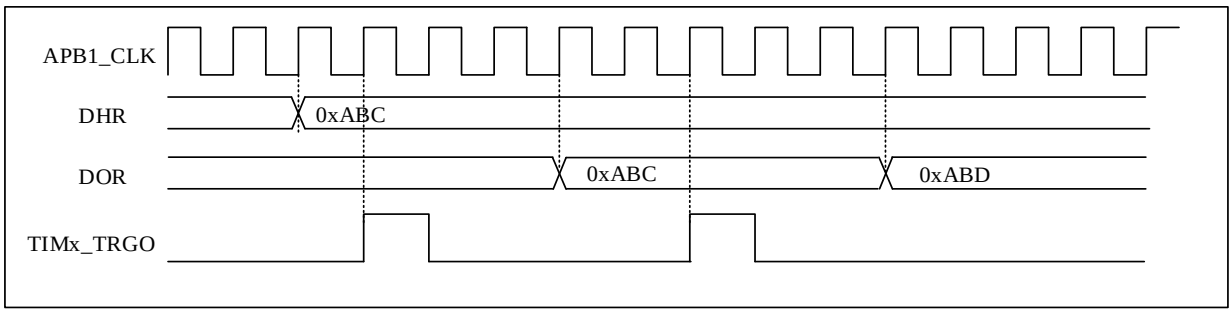


Figure 15-9 DAC conversion with triangulation generation of graphs (software triggered)



DAC conversion15-10 with triangulation generation (hardware triggered)

Note:

1. In order to generate a triangular wave, DAC trigger must be enabled, i.e. set the TENx bit of the DAC_CR register to '1';

2. The MAMPx[3:0] bits must be configured before enabling the DAC, otherwise they cannot be changed.

3. Change the value ofDAC_DHR, and the triangular wave counter is reset to 0;

4. If DAC_CR.TENx changes from "1"to"0", the triangular wave counter is also reset to 0.

15.2.10 Dual DAC channel conversion

When two DACs are required to work at the same time, in order to make more effective use of bus bandwidth, the DAC integrates three registers for dual DAC mode: DHR8RD, DHR12RD and DHR12LD. Only one register needs to be accessed to drive two DACs at the same time. Operation of DAC channels. Eleven possible conversion modes are possible using the dual DAC channels and these dual registers. These conversion modes can still be operated through a separate DAC_DHRx register when only one DAC channel is used.

All modes are described in the paragraphs below.

15.2.10.1 Independent trigger without use of waveform generator

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure different trigger sources by setting different values in the DAC_CR.TSEL1 [3:0] and DAC_CR.TSEL2 [3:0] bits;
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a DAC channel 1 trigger event occurs, the value of register DHR1 (after 3 APB1 clock cycles delay) is passed into register DAC_DOR1.

When a DAC channel 2 trigger event occurs, the value of register DHR2 (after 3 APB1 clock cycles delay) is passed into register DAC_DOR2.

15.2.10.2 Independent triggers using the same LFSR

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure different trigger sources by setting different values in the DAC_CR.TSEL1 [3:0] and DAC_CR.TSEL2 [3:0] bits;
- Configure the two DAC channel WAVEx[1:0] bits as “01” and the same LFSR mask value in the MAMPx[3:0] bits;
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a DAC channel 1 trigger event occurs, the LFSR1 counter value with the same mask isadded to the DAC_ DHR1 register value, delayed by 3 APB1 clock cycles, andthe result is passed into register DAC_DOR1, and then the LFSR1counteris updated.

When a DAC channel 2 trigger event occurs, the LFSR2 counter value with the same mask isadded to the DAC_ DHR2 register value, delayed by 3 APB1 clock cycles, andthe result is passed into register DAC_DOR2, and then the LFSR2counteris updated.

15.2.10.3 Independent triggering using different LFSRs

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure different trigger sources by setting different values in the DAC_CR.TSEL1 [3:0] and DAC_CR.TSEL2 [3:0] bits;
- Configure the two DAC channel DAC_CR.WAVEx[1:0] bits as “01” and the different LFSR mask

value in the MAMPx[3:0] bits;

- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD, or DHR8RD).

When a DAC channel 1 trigger event occurs, the LFSR1 counter value masked according to DAC_CR.MAMP1 [3: 0]is added to the DAC_ DHR1 register value, delayed by 3 APB1 clock cycles,and the result is passed into the register DAC_DOR1, and then the LFSR1 counter isupdated.

When a DAC channel 2 trigger event occurs, the LFSR2 counter value masked according to DAC_CR.MAMP2 [3: 0]is added to the DAC_ DHR2 register value, delayed by 3 APB1 clock cycles,and the result is passed into the register DAC_DOR2,and then the LFSR2 counter is updated.

15.2.10.4 Independent trigger with same triangle generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure different trigger sources by setting different values in the DAC_CR.TSEL1 [3:0] and DAC_CR.TSEL2 [3:0] bits;
- Configure the two DAC channel WAVEx [1:0] bits as “1x” and the same maximum amplitude value in the MAMPx [3:0] bits
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a DAC channel 1 trigger event occurs, the same triangular wave amplitude plus theDAC_ DHR1 register value is delayed by 3 APB1 clock cycles,and the result is passed into register DAC_DOR1, and then the DAC channel 1 triangular wavecounter is updated.

When a DAC channel 2 trigger event occurs, the same triangular wave amplitude plus theDAC_ DHR2 register value is delayed by 3 APB1 clock cycles,and the result is passed into register DAC_DOR2, and then the DAC channel 2 triangular wavecounter is updated.

15.2.10.5 Independent trigger with different triangle generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure different trigger sources by setting different values in the DAC_CR.TSEL1 [3:0] and DAC_CR.TSEL2 [3:0] bits;
- Configure the two DAC channel WAVEx [1:0] bits as “1x” and the different maximum amplitude value in the DAC_CR.MAMPx [3:0] bits
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a DAC channel 1 trigger event occurs, the triangular wave amplitude set by DAC_CR.MAMP1 [3: 0] plus theDAC_ DHR1 register value, delay 3 APB1 clock cycles, andthe result is passed into the register DAC_DOR1, and then the DAC channel 1 triangular wave counter isupdated.

When a DAC channel 2 trigger event occurs, the triangular wave amplitude set by DAC_CR.MAMP2 [3: 0] plus theDAC_ DHR2 register value, delay 3 APB1 clock cycles, andthe result is passed into the register DAC_DOR2, and then the DAC channel 2 triangular wave counter isupdated.

15.2.10.6 Simultaneous non-hardware conversion

Follow the following procedure to set the DAC conversion mode:

- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

In this configuration, after one APB1 clock cycle,the values of the DAC_DHR1 and DAC_ DHR2 registers are passed into the DAC_DOR1 and DAC_DOR2 registers, respectively.

15.2.10.7 Simultaneous trigger without wave generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure the same trigger source for both DAC channels by setting the same value in the DAC_CR.TSEL1[3:0] and DAC_CR.TSEL2[3:0] bits;
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a trigger event occurs, after a delay of 3 APB1 clock cycles, the values of the DAC_DHR1 and DAC_ DHR2 registers are passed into the DAC_DOR1 and DAC_DOR2 registers,respectively.

15.2.10.8 Simultaneous trigger with same LFSR generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- By setting the DAC_CR.TSEL1 [3: 0] and DAC_CR.TSEL2 [3: 0] bits to the same value, two DAC channels are configured to use the same trigger source respectively;
- Configure the two DAC channel DAC_CR.WAVEx[1:0] bits as “01” and the same LFSR mask value in the MAMPx[3:0] bits;
- Loading dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD, or DHR8RD);

When a trigger event occurs, the masked LFSR1 counter value set by DAC_CR.MAMP1 [3: 0]is added to the DAC_ DHR1 register value, delayed by 3 APB1 clock cycles,and the result is passed into the DAC_DOR1 register, and then the LFSR1 counter is updated.

Similarly, the masked LFSR2 counter value set by DAC_CR.MAMP1 [3: 0]is added to the DAC_ DHR2 register value, delayed by 3 APB1 clock cycles,and the result is passed into register DAC_DOR2, and then the LFSR2counter is updated.

15.2.10.9 Simultaneous trigger with different LFSR generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure the same trigger source for both DAC channels by setting the same value in the DAC_CR.TSEL1[3:0] and DAC_CR.TSEL2[3:0] bits;
- Configure the two DAC channel DAC_CR.WAVEx[1:0] bits as “01” and the different LFSR mask value in the MAMPx[3:0] bits;
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a trigger event occurs, the LFSR1 counter value with the same mask isadded to the DAC_ DHR1 register value, delayed by 3 APB1 clock cycles, and the result is passed into register DAC_DOR1, and then the LFSR1 counter is updated.

At the same time, the LFSR2 counter value with the same mask isadded to the DAC_ DHR2 register value, delayed by 3 APB1 clock cycles, and the result is passed into the register DAC_DOR2, and then the LFSR2 counter is updated.

15.2.10.10 Simultaneous trigger with same triangle generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure the same trigger source for both DAC channels by setting the same value in the DAC_CR.TSEL1[3:0] and DAC_CR.TSEL2[3:0] bits
- Configure the two DAC channel WAVEx [1:0] bits as “1x” and the same maximum amplitude value in the MAMPx [3:0] bits
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a trigger event occurs, the same triangular wave amplitude is added to theDAC_ DHR1 register value. After a delay of 3 APB1 clock cycles,the result is passed intotheregister DAC_DOR1, and then the triangular wave counter is updated.

At the same time, the same triangular wave amplitude is added to theDAC_ DHR2 register value, and after a delay of 3 APB1 clock cycles,the result is passed intotheregister DAC_DOR2, and then the triangular wave counter is updated.

15.2.10.11 Simultaneous trigger with different triangle generation

Set the DAC conversion mode in the following order:

- Set the two DAC channel trigger enable bits DAC_CR.TEN1 and DAC_CR.TEN2;
- Configure the same trigger source for both DAC channels by setting the same value in the DAC_CR.TSEL1[3:0] and DAC_CR.TSEL2[3:0] bits
- Configure the two DAC channel WAVEx [1:0] bits as “1x” and the different maximum amplitude value in the DAC_CR.MAMPx [3:0] bits
- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD, or DHR8RD).

When a trigger event occurs, the triangular wave amplitude set by DAC_CR.MAMP1 [3: 0]is added to the DAC_ DHR1 register value, and after a delay of 3 APB1 clock cycles,the result is passed into theregister DAC_DOR1, and then the triangular wave counter is updated.

At the same time, the triangular wave amplitude set by DAC_CR.MAMP2 [3: 0]is added to the DAC_ DHR2 register value, delayed by 3 APB1 clock cycles,and the result is passed into theregister DAC_DOR2, and then the triangular wave counter is updated.

15.2.11 DAC output clock

The DAC output clock (DAC_CLK) is the analog sample data clock, which is only valid when DAC_CR.ENx is "1". When selecting hardware trigger, it is detected that 5 APB1 cycles after the rising edge of the hardware trigger TRGO are set to "1", and 0 is cleared after holding 3 APB1 cycles; When

selecting software trigger, it is detected that 3 APB1 cycles after the rising edge of software trigger are set to "1", and 0 is cleared after maintaining 3 APB1 cycles; When not triggering is selected, 2 APB1 periods in which data changes are detected are set to "1", and 0 is cleared after holding 3 APB1 periods. Note: After1 APB1 cyclewhenTENxchangesfrom "1"to "0", the data of DAC_DOR is updated to the data in the DAC_DHR register. When DAC_DOR is updated to a new value, a DAC_CLK is generated; Otherwise, no DAC_CLK will be generated.

15.3 TIMx registers

These peripheral registers must be operated in a word (32-bit) manner.

15.3.1 DAC Control Register (DAC_CR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	DMAUDRIE2	DMAEN2	MAMP2 [3: 0]				WAVE2 [1: 0]		TSEL2 [3: 0]				TEN2	BOFF2	EN2
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	DMAUDRIE1	DMAEN1	MAMP1 [3: 0]				WAVE1 [1: 0]		TSEL1 [3: 0]				TEN1	BOFF1	EN1
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30	DMAUDRIE2	RW	0	DMAUDRIE2: DAC Channel 2 DMA Underflow Interrupt Enable 0: Masked DAC channel 2 DMA underflow interrupt; 1: DAC channel 2 DMA underrun interrupt enabled.
29	DMAEN2	RW	0	DMAEN2: DAC channel2 DMA enable 0: Masked DAC channel 2 DMA mode; 1: DAC channel 2 DMA mode enabled.
28: 25	MAMP2	RW	4'h0	MAMP2 [3: 0]: DAC channel 2 mask/amplitude selector 0000: Unmasked LSFR bit 0/triangular wave amplification value equal to 1; 0001: Unmasked LSFR bit [1: 0]/triangular wave amplification value equal to 3; 0010: Unmasked LSFR bit [2: 0]/triangular wave amplification value equal to 7; 0011: Unmasked LSFR bit [3: 0]/triangular wave amplification value equal to 15; 0100: Unmasked LSFR bit [4: 0]/triangular wave amplification value equal to 31; 0101: Unmasked LSFR bit [5: 0]/triangular wave amplification value equal to 63; 0110: Unmasked LSFR bit [6: 0]/triangular wave amplification value equal to 127; 0111: Unmasked LSFR bit [7: 0]/triangular wave amplification value equal to 255; 1000: unmasked LSFR bit [8: 0]/triangular wave amplification value equal to 511; 1001: unmasked LSFR bit [9: 0]/triangular wave amplification value equal to 1023; 1010: unmasked LSFR bit [10: 0]/triangular wave amplification value equal to 2047; ≥ 1011: unmasked LSFR bit [11: 0]/triangular wave amplification value equal to 4095;
24: 23	WAVE2	RW	2'h0	WAVE2 [1: 0]: DAC channel2 noise/triangle wave generation enable 00: wave generation disabled 01: noise wave generation enabled 1x: triangle wave generation enabled
22: 19	TSEL2	RW	4'h0	TSEL2 [3: 0]: DAC channel 2 trigger selection

				0000: sw 0001: tim8_trgo 0010: tim7_trgo 0011: tim15_trgo 0100: tim2_trgo 0101: tim4_trgo 0110: exti9 0111: tim6_trgo 1000: tim3_trgo 1001 tim18_trgo 1010: tim1_trgo 1011 tim5_trgo Other: Reserved
18	TEN2	RW	0	TEN2: DAC channel2 trigger enable 0: Close DAC channel 2 trigger 1: Enable DAC channel 2 trigger
17	BOFF2	RW	0	BOFF2: DAC channel2 output buffer disable 0: DAC channel 2 output buffer enabled; 1: DAC channel 2 output buffer disabled.
16	EN2	RW	0	EN2: DAC channel2 enable 0: Close DAC channel 2; 1: Enable DAC channel 2;
15	Reserved	-	-	-
14	DMAUDRIE1	RW	0	DMAUDRIE1: DAC Channel 1 DMA Underflow Interrupt Enable 0: Masked DAC channel 1 DMA underflow interrupt; 1: DAC channel 1 DMA underrun interrupt enabled.
13	DMAEN1	RW	0	DMAEN1: DAC channel 1 DMA enable 0: Mask DAC channel 1 DMA mode; 1: DAC channel 1 DMA mode enabled.
12: 9	MAMP1	RW	4'h0	MAMP1 [3: 0]: DAC channel 1 mask/amplitude selector These bits are written by software to select mask in wave generation mode or amplitude in triangle generation mode. 0000: Unmasked LSFR bit 0/triangular wave amplification value equal to 1; 0001: Unmasked LSFR bit [1: 0]/triangular wave amplification value equal to 3; 0010: Unmasked LSFR bit [2: 0]/triangular wave amplification value equal to 7; 0011: Unmasked LSFR bit [3: 0]/triangular wave amplification value equal to 15; 0100: Unmasked LSFR bit [4: 0]/triangular wave amplification value equal to 31; 0101: Unmasked LSFR bit [5: 0]/triangular wave amplification value equal to 63; 0110: Unmasked LSFR bit [6: 0]/triangular wave amplification value equal to 127; 0111: Unmasked LSFR bit [7: 0]/triangular wave amplification value equal to 255; 1000: unmasked LSFR bit [8: 0]/triangular wave amplification value equal to 511; 1001: unmasked LSFR bit [9: 0]/triangular wave amplification value equal to 1023; 1010: unmasked LSFR bit [10: 0]/triangular wave amplification value equal to 2047; ≥ 1011: unmasked LSFR bit [11: 0]/triangular wave amplification value equal to 4095;
8: 7	WAVE1	RW	2'h0	WAVE1 [1: 0]: DAC channel 1 noise/triangle wave generation enable These bits are set and cleared by the software. 00: wave generation disabled 01: noise wave generation enabled 1x: triangle wave generation enabled
6: 3	TSEL1	RW	4'h0	TSEL1 [3: 0]: DAC channel 1 trigger selection 0000: sw 0001: tim8_trgo 0010: tim7_trgo 0011: tim15_trgo 0100: tim2_trgo

				0101: tim4_trgo 0110: exti9 0111: tim6_trgo 1000: tim3_trgo 1001 tim18_trgo 1010: tim1_trgo 1011 tim5_trgo Other: Reserved
2	TEN1	RW	0	TEN1: DAC channel1 trigger enable 0: Close DAC channel 1 trigger 1: Enable DAC channel 1 trigger
1	BOFF1	RW	0	BOFF1: DAC channel1 output buffer disable 0: DAC channel 2 output buffer enabled; 1: DAC channel 2 output buffer disabled.
0	EN1	RW	0	EN1: DAC channel 1 enable 0: Close DAC channel 1; 1: Enable DAC channel 1;

15.3.2 DAC software trigger register (DAC_SWTRIGR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														SWTRIG2	SWTRIG1
														W	W

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	SWTRIG2	W	0	**SWTRIG2**: DAC channel2 software trigger 0: Close DAC channel 2 software trigger; 1: Enable DAC Channel 2 software trigger.
0	SWTRIG1	W	0	**SWTRIG1**: DAC channel1 software trigger 0: No trigger 1: Trigger

15.3.3 12-bit right-aligned data hold register for DAC channel 1 (DAC_DHR12R1)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
Res				DACC1DHR [11: 0]										
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	DACC1DHR	RW	12'h0	**DACC1DHR [11: 0]**: 12-bit right-aligned data of DAC channel 1 These bits are written by software. They specify 12-bit data for DAC channel1.

15.3.4 12-bit left-aligned data hold register for DAC channel 1 (DAC_DHR12L1)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACC1DHR [11: 0]												Res			

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 4	DACC1DHR	RW	12'h0	**DACC1DHR [11: 0]**: 12-bit left-aligned data of DAC channel 1 These bits are written by software. They specify 12-bit data for DAC channel1.
3: 0	Reserved	-	-	-

15.3.5 8-bit right-aligned data hold register for DAC channel 1 (DAC_DHR8R1)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								DACC1DHR [7: 0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	DACC1DHR	RW	8'h0	**DACC1DHR [7: 0]**: 8-bit right-aligned data of DAC channel 1 These bits are written by software. They specify 8-bit data for DAC channel1.

15.3.6 12-bit right-aligned data hold register for DAC channel 2 (DAC_DHR12R2)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res				DACC2DHR [11: 0]										
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	DACC2DHR	RW	12'h0	**DACC2DHR [11: 0]**: 12-bit right-aligned data of DAC channel 2 These bits are written by software. They specify 12-bit data for DAC channel2.

15.3.7 12-bit left-aligned data hold register for DAC channel 2 (DAC_DHR12L2)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHR [11: 0]												Res			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 4	DACC2DHR	RW	8'h0	**DACC2DHR [11: 0]**: 12-bit left-aligned data of DAC channel 2 These bits are written by software. They specify 12-bit data for DAC channel2.

3: 0	Reserved	-	-	-

15.3.8 8-bit right-aligned data hold register for DAC channel 2 (DAC_DHR8R2)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								DACC2DHR [7: 0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	DACC2DHR	RW	8'h0	**DACC2DHR [7: 0]**: 8-bit right-aligned data of DAC channel 2 These bits are written by software which specifies 8-bit data for DAC channel2.

15.3.9 12-bit right-aligned data hold register for dual DAC (DAC_DHR12RD)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				DACC2DHR [11: 0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				DACC1DHR [11: 0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 16	DACC2DHR	RW	12'h0	**DACC2DHR [11: 0]**: 12-bit right-aligned data of DAC channel 2 These bits are written by software. They specify 12-bit data for DAC channel2.
15: 12	Reserved	-	-	-
11/0	DACC1DHR	RW	12'h0	**DACC1DHR [11: 0]**: 12-bit right-aligned data of DAC channel 1 These bits are written by software. They specify 12-bit data for DAC channel1.

15.3.10 12-bit left-aligned data holding register for dual DAC (DAC_DHR12LD)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DACC2DHR [11: 0]												Res			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC1DHR [11: 0]												Res			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31: 20	DACC2DHR	RW	12'h0	**DACC2DHR [11: 0]**: 12-bit left-aligned data of DAC channel 2 These bits are written by software. They specify 12-bit data for DAC channel2.
19: 16	Reserved	-	-	-
15: 4	DACC1DHR	RW	12'h0	**DACC1DHR [11: 0]**: 12-bit left-aligned data of DAC channel 1 These bits are written by software. They specify 12-bit data for DAC channel1.
3: 0	Reserved	-	-	-

15.3.11 8-bit right-aligned data hold register for dual DAC (DAC_DHR8RD)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACC2DHR [7: 0]								DACC1DHR [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 8	DACC2DHR	RW	8'h0	**DACC2DHR [7: 0]**: 8-bit right-aligned data of DAC channel 2 These bits are written by software which specifies 8-bit data for DAC channel2.
7: 0	DACC1DHR	RW	8'h0	**DACC1DHR [7: 0]**: 8-bit right-aligned data of DAC channel 1 These bits are written by software. They specify 8-bit data for DAC channel1.

15.3.12 DAC channel 1 data output register (DAC_DOR1)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res				DACC1DOR [11: 0]										
				R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	DACC1DOR	R	12'h0	**DACC1DOR [11: 0]**: DAC channel 1 data output This bit is read-only which specifies the output data of DAC channel 1.

15.3.13 DAC channel 2 data output register (DAC_DOR2)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				DACC2DOR [11: 0]											
				R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	DACC2DOR	R	12'h0	**DACC2DOR [11: 0]**: DAC channel2 data output This bit is read-only which specifies the output data of DAC channel 2.

15.3.14 DAC status register (DAC_SR)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		DMAUDR2	Res												
		RC_W1													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		DMAUDR1	Res												
		RC_W1													

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29	DMAUDR2	RC_W1	0	DMAUDR2: DAC channel 2DMA underrun flag This bit is set by hardware and cleared by software (by writing it to 1). 0: No DMA underrun error condition occurred for DAC channel2; 1: DMA underrun error condition occurred for DAC channel2. Note: Only after DMAUDR2 is set to "1", the software can write "1" and clear "0"
28: 14	Reserved	-	-	-
13	DMAUDR1	RC_W1	0	DMAUDR1: DAC channel 1 DMA underrun flag This bit is set by hardware and cleared by software (by writing it to 1). 0: No DMA underrun error condition occurred for DAC channel1; 1: DMA underrun error condition occurred for DAC channel1. Note: Only after DMAUDR1 is set to "1", the software can write "1" and clear "0"
12: 0	Reserved	-	-	-

15.3.15 DAC output select register (DAC_OC)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			Res												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			Res											INTEREN2	INTEREN1
														RW	RW

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	INTEREN2	RW	0	**INTEREN2:** DAC2 voltage output to on-chip peripheral/GPIO select signal. This bit is controlled by software. 0: DAC2 voltage output to GPIO 1: DAC2 voltage output to on-chip peripheral (ADC, COMP)
0	INTEREN1	RW	0	**INTEREN1:** DAC1 voltage output to on-chip peripheral/GPIO select signal. This bit is controlled by software. 0: DAC1 voltage output to GPIO 1: DAC1 voltage output to on-chip peripheral (ADC, COMP)

16. Voltage Reference Buffer

16.1 Introduction

The embedded VREFBUF is used as the ADC reference voltage, and this voltage can also be output through VREF+.

16.2 Functional description

The embedded reference voltage supports four voltages and can be configured through the VREFBUFSEL of ADC_CCR:

- VREFBUFSEL = 00: 1.08 V
- VREFBUFSEL = 01: 2.048 V
- VREFBUFSEL = 10: 2.5 V
- VREFBUFSEL = 11: 2.9 V

By setting VREFBUFEN of ADC_CCR to enable VREFBUF, the user must set VREFBUFSELand the VREFBUF outputreaches the desired value.

17. Comparator (COMP)

17.1 Introduction

The chip integrates four general-purpose comparators (COMP), namely COMP1,COMP2,COMP3 and COMP4. The COMP1/4 module can be used as a separate module or in combination with timer.

The I/O used as the comparator analog function must be configured in analog mode in the GPIO register and need to enable the PAD of the comparator mapping in the SYSCFG.P *_ANA2EN (* = A, B, C, F) register.

The COMP features:

- Low power mode wake-up
- Analog signal conditioning
- Current control of Cycle by cycle when connected to the PWM output from the timer

17.1.1 Main features

- Supports voltage comparison function, each comparator has configurable positive or negative input for flexible voltage selection
  - Multiple I/O pins
  - 64 gear voltage division of VCC/VREFCMP
  - Temperature sensor output
  - DAC output
- Programmable speed and power consumption
- Programmable hysteresis function
- Write protection for configuration registers (LOCK function)
- The output can be connected to the input of I/O or timer
  - OCREF_CLR event (cycle by cycle current control)
  - Brakes for fast PWM shutdown
- Each COMP has interrupt generation capability and is used as a chip to wake up from low power mode (sleep/low power sleep/stop mode) (via EXTI)
- Provides software to configure the digital filtering time to enhance the anti-interference capability of the device
- It supports output blanking to reduce switching noise.
- Support window comparison function

17.1.2 CAN block diagram

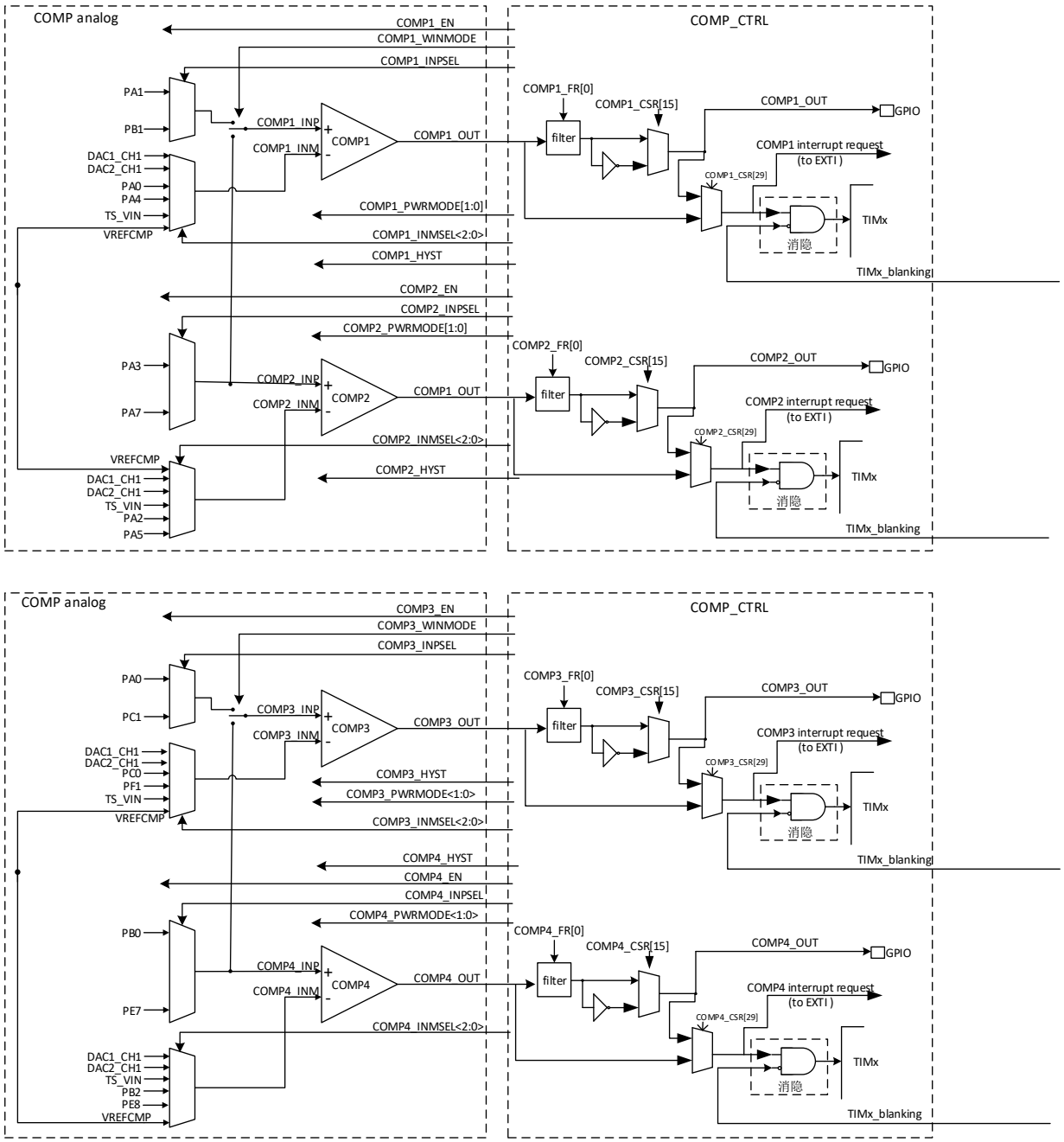


Figure 17-1 COMP structural block diagram

17.2 COMP functional description

17.2.1 Inputs and outputs of COMP

The comparator output can be connected to the I/O pin through an alternate function at the GPIO.

The output can also be internally redirected to a variety of timer input for the following purposes:

- Emergency shut-down braking of PWM signal when brake input is connected
- OCREF_CLR event (cycle by cycle current control)
- Input capture for timing measures

See the following table for the connection relationship between comparator output and other modules.

Table 17-1 COMP Module Digital Output

COMP1_OUT	COMP2_OUT	COMP3_OUT	COMP4_OUT
PA0	PB9	PB15	PB14
PA6	PA2	PC2	PB1
PA11	PA7	PB7	PB6
PB8	PA12	EXTI20	EXTI21
EXTI18	EXTI19	lptim1_ext_trig8	lptim1_ext_trig9
lptim1_ext_trig6	lptim1_ext_trig7	tim1_bk2	tim1_bk2
tim1_bk2	tim1_bk2	tim8_bk2	tim8_bk2
tim8_bk2	tim8_bk2	tim20_bk2	tim20_bk2
tim20_bk2	tim20_bk2	tim1_bk	tim1_bk
tim1_bk	tim1_bk	tim8_bk	tim8_bk
tim8_bk	tim8_bk	tim15_bk	tim15_bk
tim15_bk	tim15_bk	tim16_bk	tim16_bk
tim16_bk	tim16_bk	tim17_bk	tim17_bk
tim17_bk	tim17_bk	tim3_ti3_in1	tim2_ti3_in1
tim2_ti4_in1	tim2_ti4_in2	tim2_ti2_in3	tim2_ti2_in4
tim2_ti2_in1	tim2_ti2_in2	tim3_ti2_in3	tim3_ti2_in4
tim3_ti2_in1	tim3_ti2_in2	tim4_ti2_in3	tim4_ti2_in4
tim4_ti2_in1	tim4_ti2_in2	tim5_ti2_in3	tim5_ti2_in4
tim5_ti2_in1	tim5_ti2_in2	tim18_ti2_in3	tim18_ti2_in4
tim18_ti2_in1	tim18_ti2_in2	tim15_ti2_in2	tim1_ti1_in4
tim1_ti1_in1	tim15_ti2_in1	tim1_ti1_in3	tim2_ti1_in4
tim2_ti1_in1	tim1_ti1_in2	tim2_ti1_in3	tim3_ti1_in4
tim3_ti1_in1	tim2_ti1_in2	tim3_ti1_in3	tim4_ti1_in4
tim4_ti1_in1	tim3_ti1_in2	tim4_ti1_in3	tim8_ti1_in4
tim8_ti1_in1	tim4_ti1_in2	tim8_ti1_in3	tim18_ti1_in4
tim18_ti1_in1	tim8_ti1_in2	tim18_ti1_in3	tim5_ti1_in7
tim15_ti1_in2	tim18_ti1_in2	tim5_ti1_in6	tim1_ocref_clr3
tim5_ti1_in4	tim15_ti1_in3	tim1_ocref_lr2	tim2_ocref_clr3
tim1_ocref_clr0	tim5_ti1_in5	tim2_ocref_lr2	tim3_ocref_clr3
tim2_ocref_clr0	tim1_ocref_clr1	tim3_ocref_lr2	tim4_ocref_clr3
tim3_ocref_clr0	tim2_ocref_clr1	tim4_ocref_lr2	tim5_ocref_clr3
tim4_ocref_clr0	tim3_ocref_clr1	tim5_ocref_lr2	tim8_ocref_clr3
tim5_ocref_clr0	tim4_ocref_clr1	tim8_ocref_lr2	tim15_ocref_clr3
tim8_ocref_clr0	tim5_ocref_clr1	tim15_ocref_lr2	tim16_ocref_clr3
tim15_ocref_clr0	tim8_ocref_clr1	tim16_ocref_lr2	tim17_ocref_clr3
tim16_ocref_clr0	tim15_ocref_clr1	tim17_ocref_lr2	tim18_ocref_clr3
tim17_ocref_clr0	tim16_ocref_clr1	tim18_ocref_lr2	tim1_etr4
tim18_ocref_clr0	tim17_ocref_clr1	tim1_etr3	tim2_etr4
tim1_etr1	tim18_ocref_clr1	tim2_etr3	tim3_etr4
tim2_etr1	tim1_etr2	tim3_etr3	tim4_etr4
tim3_etr1	tim2_etr2	tim4_etr3	tim5_etr4
tim4_etr1	tim3_etr2	tim5_etr3	tim8_etr4
tim5_etr1	tim4_etr2	tim8_etr3	tim18_etr4
tim8_etr1	tim5_etr2	tim18_etr3	-
tim18_etr1	tim8_etr2	-	-
-	tim18_etr2	-	-

17.2.2 Clock and Reset of COMP

The COMP module has two clock sources:

1. PCLK (APB clock) for configuring registers

2. COMP clock, used to simulate the clock of the circuit after the output of the comparator (analog output latch circuit, glitch filter circuit, etc.), can be selected as PCLK, LSE or LSI. When it is necessary to work in stop mode, LSE or LSI is selected(selected byRCC _CCIPR.COMPxSEL).

Note:

1. PCLK and COMP CLK are simultaneously enabled by the RCC_APB2ENR.COMPEN bit. If enabled, PCLK and COMP CLK are turned off, and if enabled, PCLK and COMP CLK are turned on simultaneously.

2. Before entering Stop mode, it is recommended to configure COMP CLK to LSI or LSE; Then go into stop mode.

The reset signal of the COMP module includes APB reset source and COMP module software reset source

1. APB reset, used to reset the COMP register

2. COMP software reset, used to reset the circuit after the analog comparator output (analog output latch circuit, glitch filter circuit, etc.)

Note: When the reset signal in RCC_APB2RSTRis enabled, the entire COMP module is reset, including the registers and the circuit used to simulate the output of the comparator.

17.2.3 Filtering function of COMP

If the working environment of the chip is harsh, the output of the comparator will appear noisy signals. When the digital filter module is enabled, all noise signals whose pulse width is less than the set time of COMPx_ FR.FLTCNT [15: 0] in the output waveform of the comparator can be filtered out. If the digital filter module is disabled, the output signal of the digital filter module is the same as the input signal.

Note: To set the COMP filtering time, enabling filtering should be completed before COMPx_CSR.EN is enabled.

The filtering schematic is as follows:

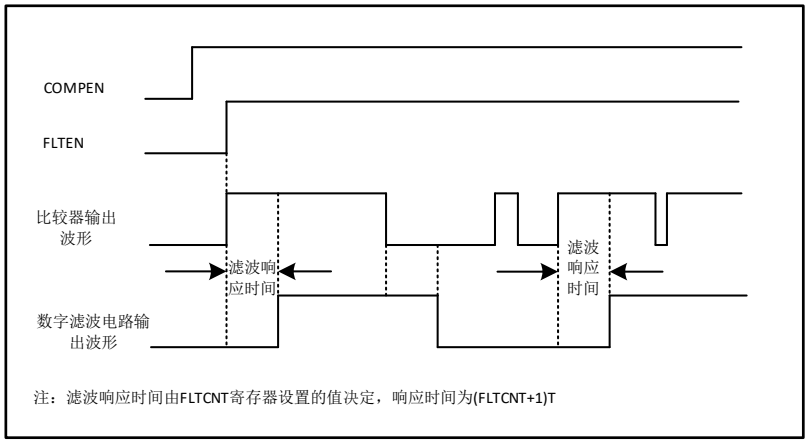


Figure 17-2 COMP filtering schematic diagram

17.2.4 COMP's Locking Mechanism

The comparator can be used in safety aspects, such as overcurrent and temperature protection. For applications with specific functional safety requirements, it is necessary to ensure that the comparator registers cannot be overwritten in case of false register access and PC (program counter) confusion.

Thus, the comparator control and status registers can be write-protected (read-only).

If the writing of the register is complete,the COMPx_CSR.LOCK bitis set to1, which makes the entire register read-only, includingthe COMPx_CSR.LOCK bit.

The write protection can only be reset by the system reset of the chip.

17.2.5 Output blanking of COMP

The purpose of the blanking function is to prevent current from tripping at the beginning of the regulated PWM cycle due to a short current spike (usually the recovery current in the anti-parallel diode of the power switch). The blanking function is realized by setting the output comparison signal of the timer. Theblanking source for each comparator channel is individually selected by the software viathe BLANKSEL [2: 0] bit of the corresponding COMPx_ CSR register. The inverted blanking signal is logically AND with the comparator output to produce the output of the comparator channel.

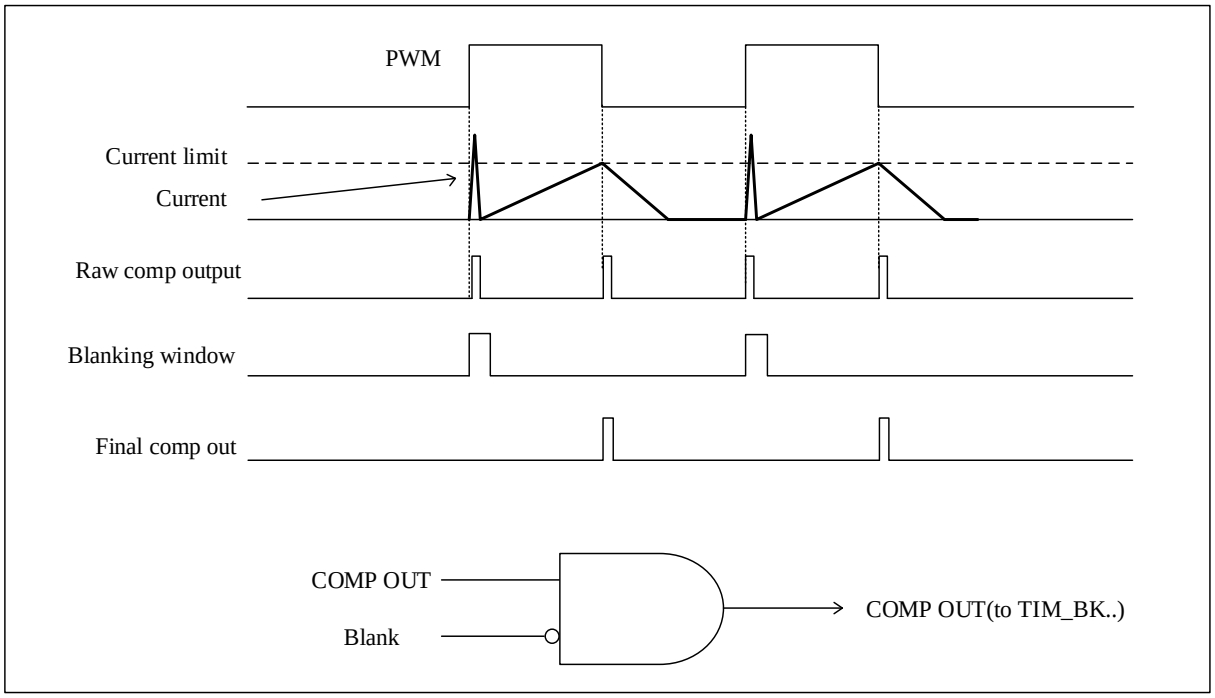


Figure 17-3 comparator blanking

17.2.6 Window Mode of COMP

The comparator in window mode is simply referred to as a window comparator, and the function of the window comparator is to monitor whether the analog voltage is within the low and high threshold ranges. Window comparators can be created using COMP1 and COMP2, and COMP3 and COMP4. The monitored analog voltages are simultaneously connected to the positive inputs of both comparators, and the high threshold and the low threshold are connected to the negative inputs of both comparators, respectively.

By enabling theCOMP1/3_ CSR.WINMODE bit, the positive inputsof the COMP1 and COMP2, or COMP3 and COMP4 comparators can beconnected together, saving an I/Opin.

For example, when WINMODE of Comparator 1 is set to 1, the positive input of the comparator will be connected to COMP2_INP, and when WINMODE is cleared to 0, the positive input of the comparator will be connected to COMP1_INP. Doing so can save an IO pin.

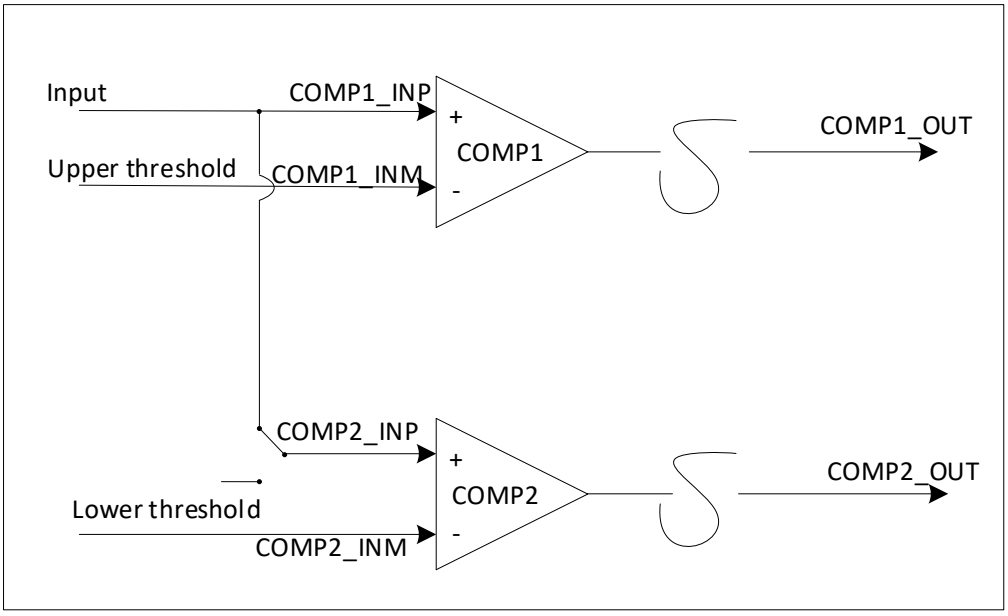


Figure 17-4 Window comparator

17.2.7 Hysteresis function of COMP

The comparator has a programmable hysteresis to avoid false output jumps in the presence of noisy signals. By enablingthe HYST bits of COMP1_CSR, COMP2_CSR, COMP3_CSR, and COMP4_CSR, the hysteres is function of COMP1, COMP2, COMP3, and COMP4, respectively, can be turned on.

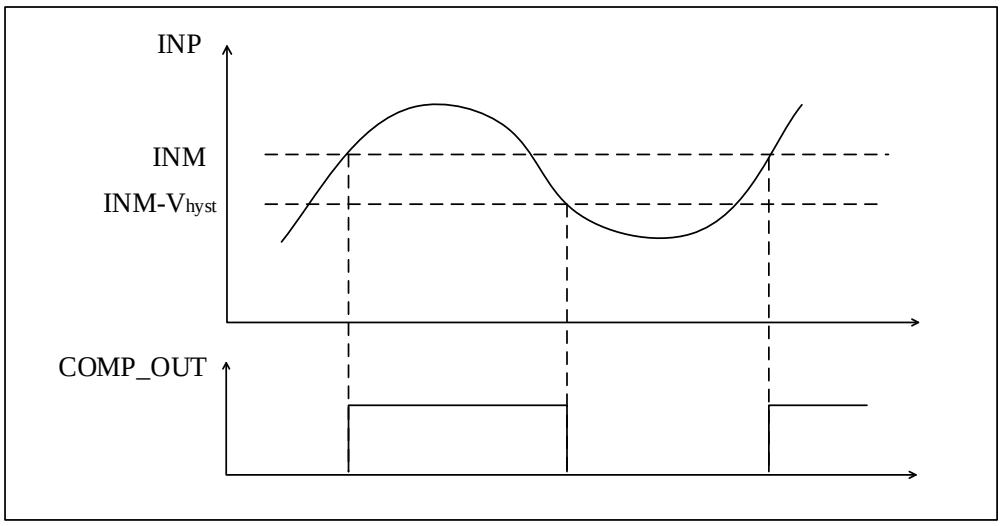


Figure 17-5 Hysteresis schematic diagram of comparator

17.2.8 Power Consumption Mode of COMP

The power consumption and transmission delay of the comparator can be adjusted to achieve specific applications. This function is selected bythe PWRMODE [1: 0] bitofthe COMPx_CSR register.

17.2.9 Low Power Mode of COMP

Mode	Description
Sleep Mode	There was no effect on COMP. The comparator interrupt may cause the device to exit sleep mode
Low power operating mode	No effect
Low power sleep mode	No impact. The comparator interrupt may cause the device to exit a low-power sleep mode
Stop mode	There was no effect on COMP. The comparator interrupt may cause that device to exit the stop mode
Standby mode	The COMP register is powered down and must be reinitialized after exiting standby mode

17.2.10 Interruption of COMP

The output of the comparator is connected to the EXTI controller (Extended interrupts and events) inside the chip. Each comparator has its own EXTI line and is capable of generating interrupts or events. The same mechanism is used as a low power wake-up.

TIMx registers

17.2.11 COMP1 Control and Status Register (COMP1_CSR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP1_OUT	TIM_EXTI_OUT_SEL	Res.	VCSEL	VCDIV_EN	VCDIV						PWRMODE		Res.	HYST
RW	R	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW			RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL			WINMODE	Res.	INPSEL	Res			INNSEL			COMP_VSEL		EN
RW	RW			RW		RW				RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP1_CSR register lock Software set, system reset and clear. When set, all 32 bits of the COMP1_CSR register are locked 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP1_OUT	R	0	COMP1 output This read-only flag reflects the level of the comparator 1 output before the polarity selector.
29	TIM_EXTI_OUT_SEL	RW	0	COMP1 output to TIM/EXTI select. 0: Direct output to TIM/EXTI without filtering 1: After filtering, output to TIM/EXTI
28	Reserved	-	-	-
27	VCSEL	RW	0	Voltage source selection signal 0: VREFCMP 1: VCC
26	VCDIV_EN	RW	0	VREFCMP enable signal 0: Close VREFCMP 1: Enable VREFCMP
25: 20	VCDIV	RW	6'h0	VREFCMP output voltage selection signal 6 'h0: 1/64 partial voltage source (VREFCMP) 6 'h1: 2/64 partial voltage source (VREFCMP) 6 'h2: 3/64 partial voltage source (VREFCMP) … 6 'h3E: 63/64 partial voltage source (VREFCMP) 6 'h3F: 64/64 partial voltage source (VREFCMP)
19: 18	PWRMODE	RW	2'h0	Comparator 1 power mode selector

				It selects the power consumption and as a consequence the speed of the comparator 1 2 'h0: High speed mode (250 uA) 2 'h1: medium speed mode (5 uA) 2'h2: Reserved 2'h3: Reserved
17	Reserved	-	-	-
16	HYST	RW	0	Comparator 1 hysteresis selector 0: Hysteresis function turned off 1: Hysteresis function enabled
15	POLARITY	RW	0	Comparator 1 polarity selector 0: Non-inverted 1: Inverted
14: 12	BLANKSEL	RW	3'h0	Comparator 1 Blanking Signal Selection 3 'h0: no blanking signal 3 'h1: tim1_oc5 3 'h2: tim2_oc3 3 'h3: tim3_oc3 3 'h4: tim8_oc5 3 'h5: tim18_oc3 3 'h6: tim15_oc1 3 'h7: tim4_oc3
11	WINMODE	RW	0	COMP1 window mode enabled 0: Window mode is not enabled, and the positive input of COMP1 is COMP1_INP 1: Window mode enabled, the positive input of COMP1 is the same as the positive input of COMP2
10	Reserved	-	-	-
9	INPSEL	RW	0	Signal selection of COMP1 non-reverse input 0: PA1 1: PB1
8: 6	Reserved	-	-	-
5: 3	INNSEL	RW	3'h0	Signal selection of COMP1 reverse input 3 'h0: PA0 3 'h1: PA4 3 'h2: DAC1_CH1 3 'h3: DAC2_CH1 3 'h4: VREFCMP 3 'h5: TS_VIN 3'h6: Reserved
2: 1	COMP_VSEL	RW	2'h0	VREFCMP voltage divider source voltage selection signal 2 'h0: The output voltage divider source is 0.6 V 2 'h1: The output voltage divider is 1.5 V 2 'h2: The output voltage divider is 2.048 V 2 'h3: The output voltage divider source is 2.5 V Note: (0.6 V VREFCMP accurate value storage address: 0x1FFF 7E40 1.5 V VREFCMP accurate value storage address: 0x1FFF 7E44 2.048 V VREFCMP accurate value storage address: 0x1FFF 7E48 2.5 V VREFCMP precise value storage address: 0x1FFF 7E4C For example: the value read from address 0x1FFF 7E40 is 0x0599, indicating that the accurate value of VrefBuf is 0.599 V)
0	EN	RW	0	Comparator 1 enable 0: Close COMP1 1: Enable COMP1

17.2.12 COMP1 Filter Register(COMP1_FR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31: 16	FLTCNT1	RW	16'h0	Comparator 1 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count period = FLTCNT [15: 0]
15: 1	Reserved	-	-	-
0	FLTEN1	RW	0	Comparator 1 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when EN of COMP1 is 0

17.2.13 Comparator 2 control and status register (COMP2_CSR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP2_OUT	Tim_EXTI_OUT_SEL.	Res.									PWRMODE		Res.	
RW	R	RW										RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL			Res		INPSEL	Res			INNSEL			Res	HYST	EN
RW	RW	RW	RW			RW				RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP2_CSR register lock Software set, system reset and clear. When set, all 32 bits of the COMP2_CSR register are locked 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP2_OUT	R	0	COMP2 output This read-only flag reflects the level of the comparator 2 output before the polarity selector.
29	TIM_EXTI_OUT_SEL	RW	0	COMP2 output to TIM/EXTI select. 0: Without filtering, ana_out (which can be blanked) is directly output to TIM/EXTI 1: Select comp_out output to PAD (can be filtered, can be blanked) and output to TIM/EXTI
28: 20	Reserved	-	-	-
19: 18	PWRMODE	RW	2'h0	Comparator 2 power mode selector It selects the power consumption and as a consequence the speed of the comparator 2. 2 'h0: High speed mode (250 uA) 2 'h1: medium speed mode (5 uA) 2'h2: Reserved 2'h3: Reserved
17: 16	Reserved	-	-	-
15	POLARITY	RW	0	COMP2 polarity selection 0: Non-inverted 1: Inverted
14: 12	BLANKSEL	RW	3'h0	COMP2 blanking signal selection 3 'h0: no blanking signal 3 'h1: tim1_oc5 3 'h2: tim2_oc3 3 'h3: tim3_oc3 3 'h4: tim8_oc5 3 'h5: tim18_oc3 3 'h6: tim15_oc1 3 'h7: tim4_oc3
11: 10	Reserved	-	-	-

9	INPSEL	RW	0	Signal selection of COMP2 non-reverse input 0: PA3 1: PA7
8: 6	Reserved	-	-	-
5: 3	INNSEL	RW	3'h0	Signal selection of COMP2 reverse input 3 'h0: PA2 3 'h1: PA5 3 'h2: DAC1_CH1 3 'h3: DAC2_CH1 3 'h4: VREFCMP 3 'h5: TS_VIN 3'h6: Reserved
2	Reserved	-	-	-
1	HYST	RW	0	Comparator 2 hysteresis selector 0: Hysteresis function turned off 1: Hysteresis function enabled
0	EN	RW	0	Comparator 2 enable 0: Close COMP2 1: Enable COMP2

17.2.14 COMP2Filter Register(COMP2_FR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT2 [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.															FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31: 16	FLTCNT2	RW	16'h0	Comparator 2 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count period = FLTCNT [15: 0]
15: 1	Reserved	-	-	-
0	FLTEN2	RW	0	Comparator 2 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when EN of COMP2 is 0

17.2.15 COMP3Control and Status Register(COMP3_CSR)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP3_OUT	Tim_EXTI_OUT_SEL.	Res.									PWRMODE		Res.	
RW	R	RW										RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	Blanksell [2: 0]			WINMODE	Res.	INPSEL	Res			INNSEL			Res	HYST	EN
RW	RW	RW	RW	RW		RW				RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP3_CSR register lock Software set, system reset and clear. When set, all 32 bits of the COMP3_CSR register are locked 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP3_OUT	R		COMP3 output

				This read-only flag reflects the level of the comparator 3 output before the polarity selector.
29	TIM_EXTI_OUT_SEL	RW	0	COMP3 output to TIM/EXTI select. 0: Without filtering, ana_out (which can be blanked) is directly output to TIM/EXTI 1: Select comp_out output to PAD (can be filtered, can be blanked) and output to TIM/EXTI
28: 20	Reserved	-	-	-
19: 18	PWRMODE	RW	2'h0	Comparator 3 power mode selector It selects the power consumption and as a consequence the speed of the comparator 3. 2 'h0: High speed mode (250 uA) 2 'h1: medium speed mode (5 uA) 2'h2: Reserved 2'h3: Reserved
17: 16	Reserved	-	-	-
15	POLARITY	RW		COMP3 polarity selection These bits can be read and written by software. 0: Non-inverted 1: Inverted
14: 12	BLANKSEL	RW	3'h0	COMP3 blanking signal selection 3 'h0: no blanking signal 3 'h1: tim1_oc5 3 'h2: tim3_oc3 3 'h3: tim2_oc4 3 'h4: tim8_oc5 3 'h5: tim18_oc4 3 'h6: tim15_oc1 3 'h7: tim4_oc3
11	WINMODE	RW		COMP3 window mode enabled 0: Window mode is not enabled, and the positive input of COMP3 is COMP3_INP 1: Window mode enabled, the positive input of COMP3 is the same as the positive input of COMP4
10	Reserved	-	-	-
9	INPSEL	RW	0	Signal selection of COMP3 non-reverse input 0: PA0 1: PC1
8: 6	Reserved	-	-	-
5: 3	INNSEL	RW	3'h0	COMP3 Signal Selection without Reverse Input 3 'h0: PC0 3 'h1: PF1 3 'h2: DAC1_CH1 3 'h3: DAC2_CH1 3 'h4: VREFCMP 3 'h5: TS_VIN 3'h6: Reserved
2	Reserved	-	-	-
1	HYST	RW	0	Comparator 3 hysteresis selector 0: Hysteresis function turned off 1: Hysteresis function enabled
0	EN	RW		COMP3 enable bit 0: Close COMP3 1: Enable COMP3

17.2.16 COMP3Filter Register(COMP3_FR)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT3 [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															FLTEN3
															RW

Bit	Name	R/W	Reset Value	Function
31: 16	FLTCNT3	RW	16'h0	Comparator 3 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count period = FLTCNT [15: 0]
15: 1	Reserved	-	-	-
0	FLTEN3	RW	0	Comparator 3 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when EN of COMP3 is 0

17.2.17 COMP4 Control and Status Register(COMP4_CSR)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK.	COMP4_OUT	Tim_EXTI_OUT_SEL.	Res.									PWRMODE		Res.	
RW	R	RW										RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	Blanksell [2: 0]			Res		INPSEL	Res			INNSEL			Res	HYST	EN
RW	RW	RW	RW			RW				RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP4_CSR register lock Software set, system reset and clear. When set, all 32 bits of the COMP4_CSR register are locked 0: Unlocked, can read and write the entire register 1: Lock, entire register read-only
30	COMP4_OUT	R	0	COMP4 output This bit is read-only and reflects the output level of COMP4 after polarity selection.
29	TIM_EXTI_OUT_SEL	RW	0	COMP4 output to TIM/EXTI select. 0: Without filtering, ana_out (which can be blanked) is directly output to TIM/EXTI 1: Select comp_out output to PAD (can be filtered, can be blanked) and output to TIM/EXTI
28: 20	Reserved	-	-	-
19: 18	PWRMODE	RW	0	COMP4 power consumption mode selection It selects the power consumption and as a consequence the speed of the comparator 3. 2 'h0: High speed mode (250 uA) 2 'h1: medium speed mode (5 uA) 2'h2: Reserved 2'h3: Reserved
17: 16	Reserved	-	-	-
15	POLARITY	RW	0	COMP4 polarity selection These bits can be read and written by software. 0: Non-inverted 1: Inverted
14: 12	BLANKSEL	RW	3'h0	Comparator 4 Blanking Signal Selection 3 'h0: no blanking signal 3 'h1: tim3_oc4 3 'h2: tim8_oc5 3 'h3: tim15_oc2 3 'h4: tim1_oc5 3 'h5: tim18_oc4 3 'h6: tim15_oc1 3 'h7: tim4_oc3
11: 10	Reserved	-	-	-
9	INPSEL	RW	0	COMP4 does not reverse input signal selection, software can be read and written 0: PB0 1: PE7

8: 6	Reserved	-	-	-
5: 3	INNSEL	RW	3'h0	COMP4 Signal selection without reverse input 3 'h0: PB2 3 'h1: PE8 3 'h2: DAC1_CH1 3 'h3: DAC2_CH1 3 'h4: VREFCMP 3 'h5: TS_VIN 3'h6: Reserved
2	Reserved	-	-	-
1	HYST	RW	0	COMP4 hysteresis function enable control 0: Hysteresis function turned off 1: Hysteresis function enabled
0	EN	RW	0	COMP4 enable bit These bits can be read and written by software. 0: Close COMP4 1: Enable COMP4

17.2.18 COMP4Filter Register(COMP4_FR)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT4 [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.															FLTEN4
															RW

Bit	Name	R/W	Reset Value	Function
31: 16	FLTCNT4	RW	16'h0	Comparator 4 Sampling Filter Counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count period = FLTCNT [15: 0]
15: 1	Reserved	-	-	-
0	FLTEN4	RW	0	Comparator 4 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when EN of COMP4 is 0

18. Operational amplifier (OPA)

18.1 Overview

There are 3 op amps embedded, each with two inputs and one output. Three I/Os can be connected to external pins, enabling any type of external interconnection. Operational amplifiers can be internally configured as programmable gain amplifiers, as an amplifier with forward gains ranging from 2 to 32 and inverting gains ranging from -1 to -31.

The positive input can be connected to an internal DAC. The output is connected to the internal ADC.

18.1.1 Main features

- Rail-to-rail input and output voltage ranges
- Low input bias current
- Low input offset voltage
- High frequency gain bandwidth
- High speed mode for better conversion rates

18.1.2 CAN block diagram

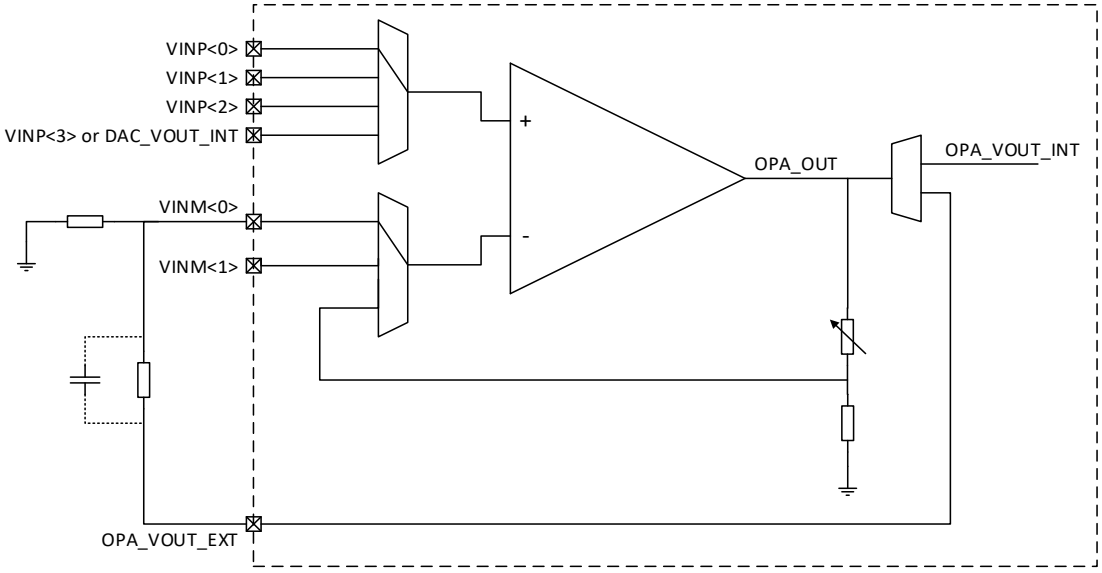


Figure 18-1 OPA module block diagram

18.2 OPA functional description

There are multiple modes of OPA. Each OPA can be enabled individually and the output is high impedance when disabled.

18.2.1 The OPA output redirects to the internal ADC channel

By setting the OPAINTOEN bit in the OPAx_CSR register, the op amp output can be internally redirected to the ADC channel. In this case, the GPIO mapped to OPAx_VOUT is released and may be used for other purposes. The ADC may measure the OPAx_VOUT voltage internally.

18.2.2 OPA reset and clocks

The OPA clock provided by the RCC is synchronized with the PCLK2 (APB2 clock). A clock enable control bit is provided in the RCC controller. The OPAx_CSR.OPAEN bit is used toenable and disablethe OPA module. If the OPA register configuration needs to be changed, it must be before enabling the OPAEN bit to avoid impact on the output. When the output of the op amp is no longer needed, the OPA can be disabled to save power consumption. When OPA is disabled, all previously set configurations are retained.

18.2.3 Initial configuration

Underthe default configuration conditions,OPAwill be innormal operationmodeusing standard performance.SettingOPAx_ CSR.OPAHSM willcauseOPAto switch to high-speed modeandgeta better conversion rate. By default, three input/output signals of OPA are connected to external pins.

Once the OPAEN bit in the OPAx_CSR register is set,OPA works. The two input pins and output pins are connected as shown in the table below and the default connection settings can also be changed.

Note: The input and output pins must be configured in analog mode (default state) in the corresponding GPIOx_MODER register.

18.2.4 Signal connection

The connection of the op amp pins is determined by the OPAx_CSR register. The connection of 3 operational amplifiers (OPAx, x = 1 … 3) is described in the table below.

Table 18-1 Possible Connections of Operational Amplifier

Signal	IO pin	Internal	Comment
OPA1_VINM	PA3 (VINM < 0 >) PC5 (VINM < 1 >)	OPA1_OUT or PGA	Controlled by OPA1_VMSEL and OPA1_PGA_GAIN
OPA1_VINP	PA1 (VINP < 0 >) PA3 (VINP < 1 >) PA7 (VINP < 2 >)	DAC1_VOUT_INT	Controlled by OPA1_VPSEL
OPA1 output	PA2	ADC1_IN13	Controlled by OPA1_INTOEN
OPA2_VINM	PA5 (VINM < 0 >) PC5 (VINM < 1 >)	OPA2_OUT or PGA	Controlled by OPA2_VMSEL and OPA2_PGA_GAIN
OPA2_VINP	PA7 (VINP < 0 >) PB14 (VINP < 1 >) PB0 (VINP < 2 >) PD14 (VINP < 3 >)	-	Controlled by OPA2_VPSEL
OPA2 output	PA6	ADC2_IN16	Controlled by OPA2_INTOEN
OPA3_VINM	PB2 (VINM < 0 >) PB10 (VINM < 1 >)	OPA3_OUT or PGA	Controlled by OPA3_VMSEL and OPA3_PGA_GAIN
OPA3_VINP	PB0 (VINP < 0 >) PB13 (VINP < 1 >) PA1 (VINP < 2 >)	DAC2_VOUT_INT	Controlled by OPA3_VPSEL
OPA3 output	PB1	ADC2_IN18,ADC3_IN13	Controlled by OPA3_INTOEN

18.3 OPA mode

Both the inputs and outputs of the op amp are accessible via the GPIO. The op amps can be used in a variety of configuration environments:

- Independent mode (external gain setting mode)
- PGA mode (programmable gain amplifier mode)

18.3.1 Independent mode (external gain setting mode)

Starting first with the OPAx_CSR default value and the GPIOx_MODER default state, once the OPAx_CSR.OPAEN bit isset, the two input pins and the output pins are connected to the op amp.

This default configuration causes OPA to run in normal mode. OPA can modify the configuration through the following process:

- By setting OPAx_CSR.OPAHSM, OPA is set to the op amp high speed mode toobtain a high conversion rate.

OPA standalone mode configuration process:

1) Configure OPAx_CSR.VP_SEL/OPAx_CSR.VM_SEL Select VINP/VINM
2) Configure the OPAx_CSR.INTOEN bitto enableOPAx_VOUT_EXT
3) The voltage isoutput to OPAx_VOUT_EXTas long as the OPAx _CSR.OPAEN bitis configured

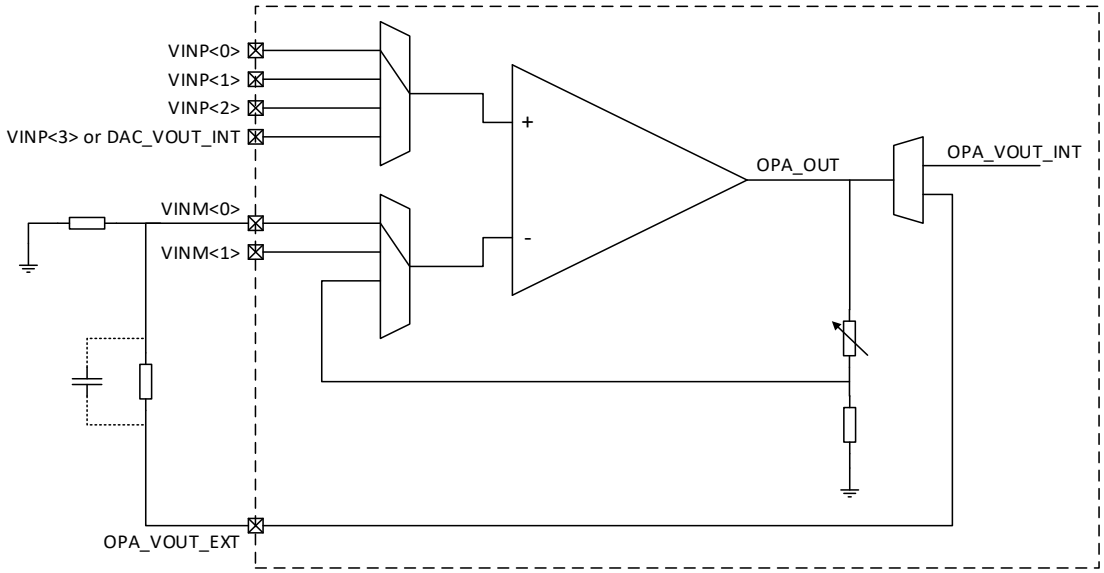


Figure 18-2 Independent mode (external gain setting mode)

18.3.2 Programmable gain amplifier mode

OPA programmable gain amplifier mode configuration process:

1) Configure the OPAx_CSR.VM_SELbit to 2'b10and connect the feedback resistorto the inputof OPAx_VINM
2) Configure the OPAx_CSR.PGA _GAIN bits to5 'b00000~5'b00101so that the internal gain is2, 4, 8, 16or32
3) Configure the OPAx_CSR.VP_SELbit to 2'b00 and connectthe input of thecorresponding GPIOto the OPAx_VINP terminal
4) Configure the OPAx_CSR.INTOEN bittoenable OPAx_VOUT_EXTorOPAx_VOUT_INT

5) As long as the OPAx _CSR.OPAEN bit isset,the voltageon the OPAx_VINP pin isamplified according to the selectedgainand isvisible on the OPAx_VOUT_EXTorOPAx_VOUT_INT pin.

Note: To avoid saturation, the input voltage must be kept below VCCA divided by the selected gain.

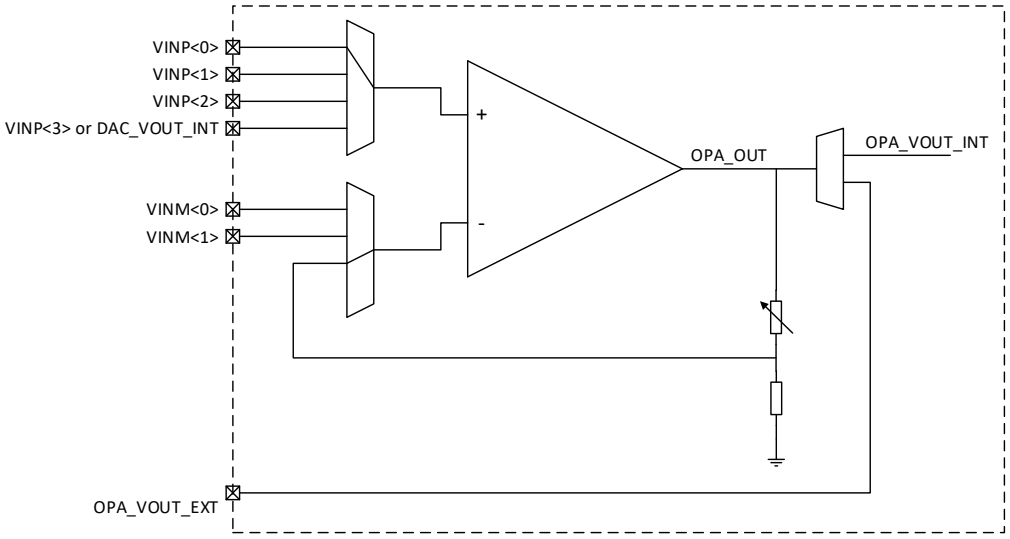


Figure 18-3 PGA mode, internal gain setting (x2/x4/x8/x16/x32) reverse input not used

18.3.3 Programmable gain amplifier mode with external filtering

OPA programmable gain amplifier mode configuration process with external filtering:

1) Configure the OPAx_CSR.VM_SELbit to 2'b10so that the feedback resistor is connected tothe input ofOPAx_VINM
2) Configure the OPAx_CSR.PGA _GAIN bits to 5'b10000 ~ 5'b10101so that the internal gain of the filteredVINM < 0 >pin is2, 4, 8, 16or32
3) Configure the OPAx_CSR.VP_SELbit to connectthe input of thecorrespondingGPIOto the OPAx_VINP
4) Configure the OPAx_CSR.INTOEN bitto enableOPAx_VOUT_EXT
5) As long as the OPAx _CSR.OPAEN bit isconfigured,the voltageon the OPAx_VINP pin is amplified according to the selected gainandis visible on the OPAx_VOUT_EXT pin.

Any external connection on the VINM can be used in parallel with the internal PGA, for example, a capacitor can be connected between OPA_OUT and the VINM for filtering purposes.

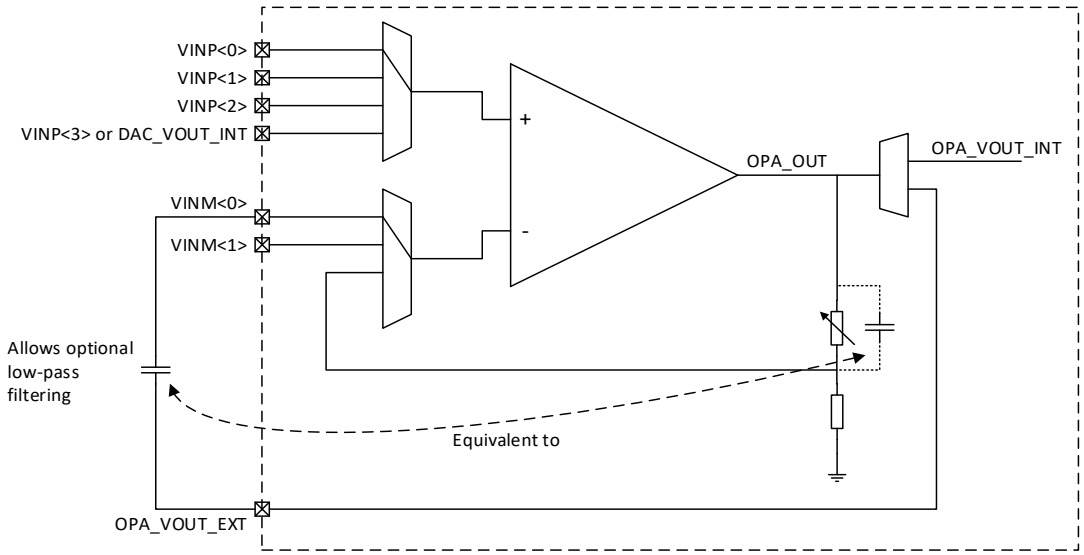


Figure 18-4 PGA mode, internal gain settings (x2/x4/x8/x16/x32) reverse input for filtering

18.3.4 Programmable gain amplifier, forward or inverting mode with external bias

OPA forward or inverting mode configuration process with external bias:

1) Configure the OPAx_CSR.VM_SELbit to 2'b10so that the feedback resistor is connected tothe input ofOPAx_VINM
2) The OPAx_CSR.PGA _GAIN bits are configured to be 5'b01000to 5'b01101such thatthe reverse gain at the VINM < 0 >terminal is− 1, − 3,
   -7,-15,-31/non-reverse gain of 2, 4, 8, 16, 32
3) Configure the OPAx_CSR.VP_SELbit to connectthe input of thecorrespondingGPIOto the OPAx_VINP
4) Configure the OPAx_CSR.INTOEN bitto enableOPAx_VOUT_EXTorOPAx_VOUT_INT
5) As long as the OPAx _CSR.OPAEN bit isset,the voltageon the OPAx_VINP pin is amplified according to the selectedgainand isvisibleon the OPAx_VOUT_EXTorOPAx_VOUT_INT pin.

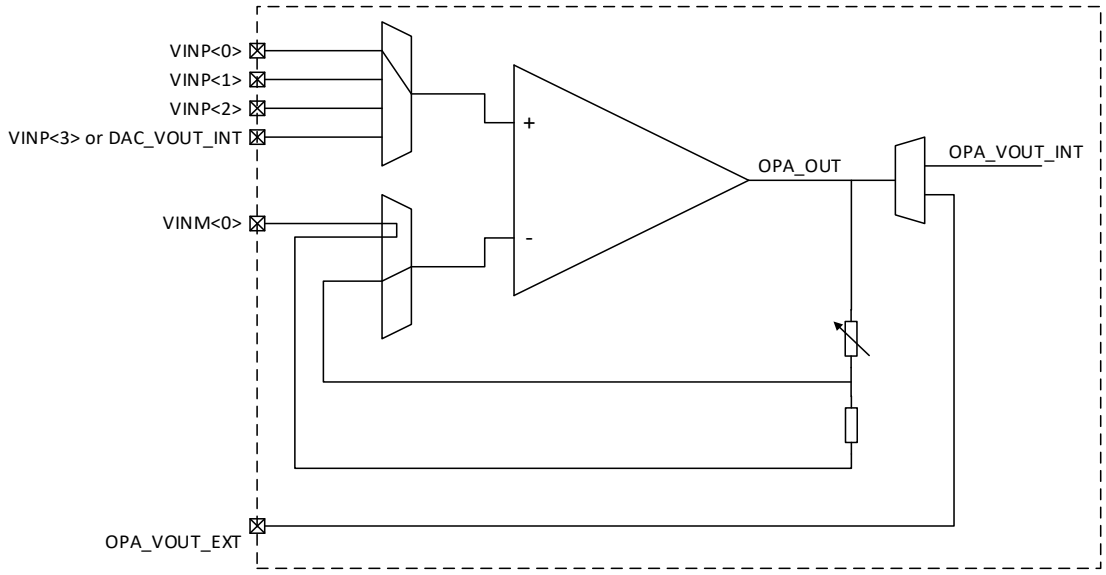


Figure 18-5 PGA mode, non-reverse gain setting (x2/x4/x8/x16/x32) or reverse gain setting (x-1/x-3/x-7/x-15/x-31)

18.3.5 Programmable gain amplifier, forward with external bias or inverted mode with filtering

OPA forward with external bias or inverted mode with filtering configuration process:

1) Configure the OPAx_CSR.VM_SELbit to 2'b10and connect the feedback resistorto the inputof OPAx_VINM
2) Configure the OPAx_CSR.PGA _GAIN bits to 5'b11000~ 5'b11101so thatthe reverse gain on the filtered VINM < 0 >and VINM< 1 >pinsis-1,-3,-7,-15,-31/the non-reverse gain is2, 4, 8, 16, 32
3) Configure the OPAx_CSR.VP_SELbit toconnect the corresponding GPIO inputto the OPAx_VINP
4) Configure the OPAx_CSR.INTOEN bitto enableOPAx_VOUT_EXT
5) As long as the OPAx__CSROPAEN bit is configured, the voltage onthe OPAx_VINP pinis amplifiedaccording to the selectedgainand isvisible on the OPAx_VOUT_EXT pin.

Any external connection on VINM < 1 > can be used in parallel with the internal PGA, for example, a capacitor can be connected between OPA_OUT and VINM < 1 > for filtering purposes.

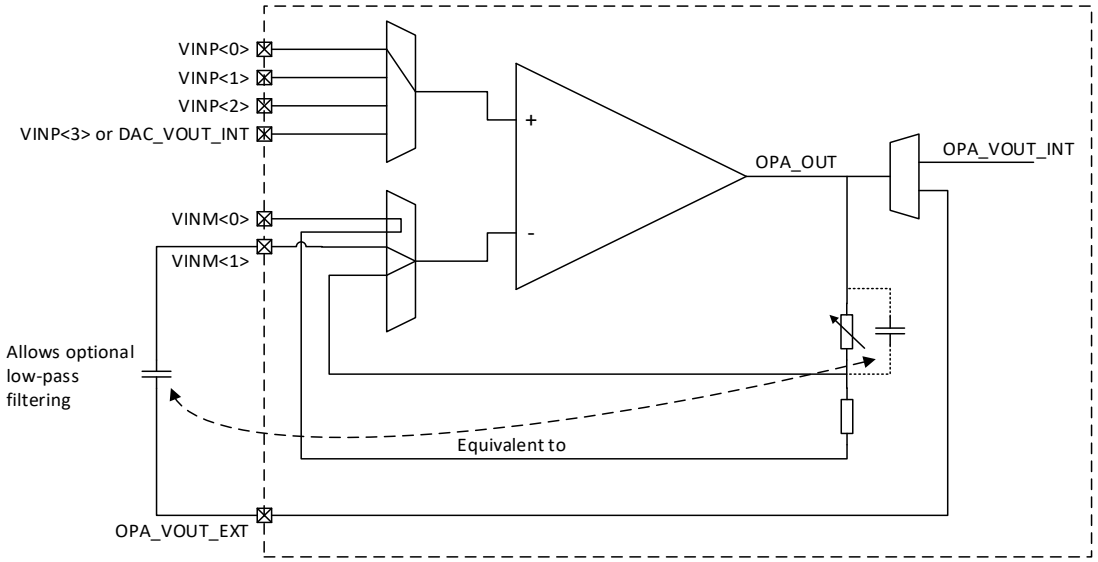


Figure 18-6 PGA mode, non-inverted gain setting (x2/x4/x8/x16/x32) or inverted gain setting (x-1/x-3/x-7/x-15/x-31) for filtering

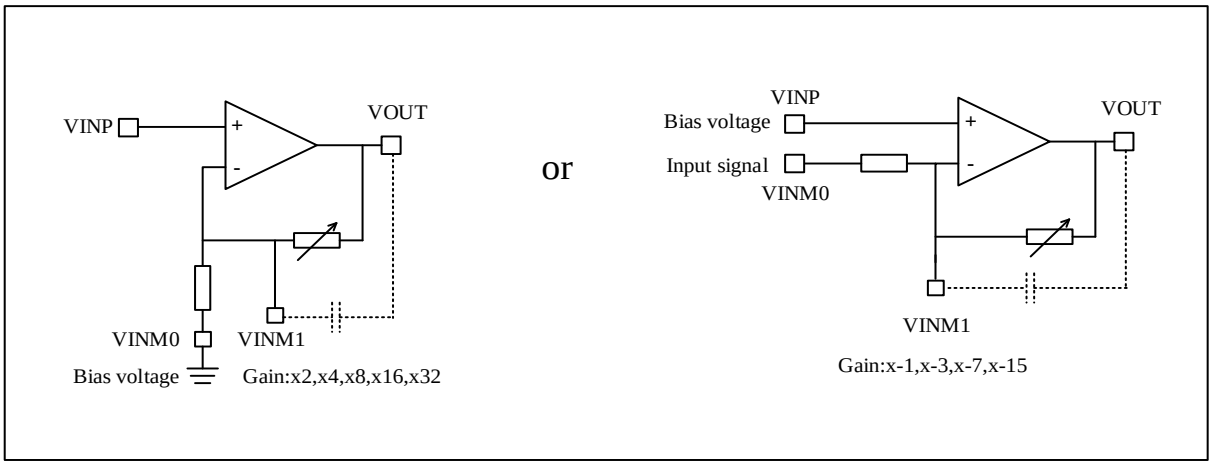


Figure 18-7 configuration example

18.3.6 Current sampling mode

In motor applications, you can choose the common mode voltage built-in, and the built-in voltage can be selected as VCC/2 and VCC/5. An external voltage may also be used.

However, common mode voltage external and built-in modes cannot exist at the same time.

Select the external voltage when OPAx_VBSEL = 0 in the OPAx_CSR register.

Selecting the built-in voltage by configuring OPAx_VBSEL = 1 in the OPAx_CSR register;

The built-in voltage selection VCC/2 or VCC/5 of OPAx is controlled by configuring OPA_VBIAS in the OPA1_CSR register.

Note: In current sampling mode, OPA input and output need to use external input/output, that is, internal signal cannot be used as OPA input.

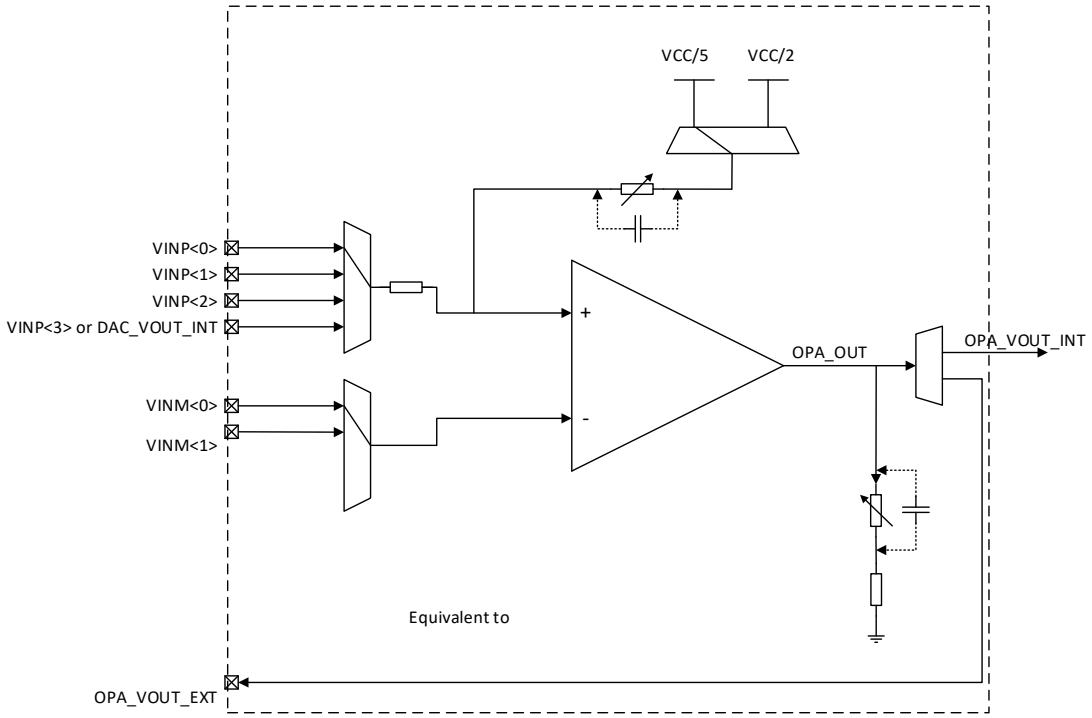


Figure 18-8 Current sampling circuit

18.4 OPA PGA Gain

When the OPA is configured in PGA mode, the gain is programmable to x2, x4, x8, x16, x32 in the forward configuration and x-1, x-3, x-7, x-15, x-31 in the inverted configuration.

When the OPA is configured in forward mode, the gain depends only on the internal resistance divider. When configured in inverted mode, the gain factor defines not only the on-chip feedback resistance but also the signal source output impedance. If the output impedance of the signal source is not negligible compared to the input feedback resistance of the PGA, a gain error will occur.

18.5 Timer Control OPA VINP/VINM Select Mode

The selection of OPA inverting and non-inverting inputs can be done automatically. This automatic switching is triggered by either the TIM1 CC6 or TIM8 CC6 output arriving at the OPA input multiplexer. This is useful for dual-motor controls that require simultaneous measurement of three-phase currents on the first motor and the second motor.

Enable the automatic switch by settingthe OPAx_ TCMR.TxCM_EN bit(x = 1, 8). Inverting and non-inverting input selections are configured via OPAx_ TCMR.VPS_SEL and VMS_SEL. If the TxCM_EN bit is cleared, completion is selected using VP_SEL and VM_SELinOPAx _CSR.

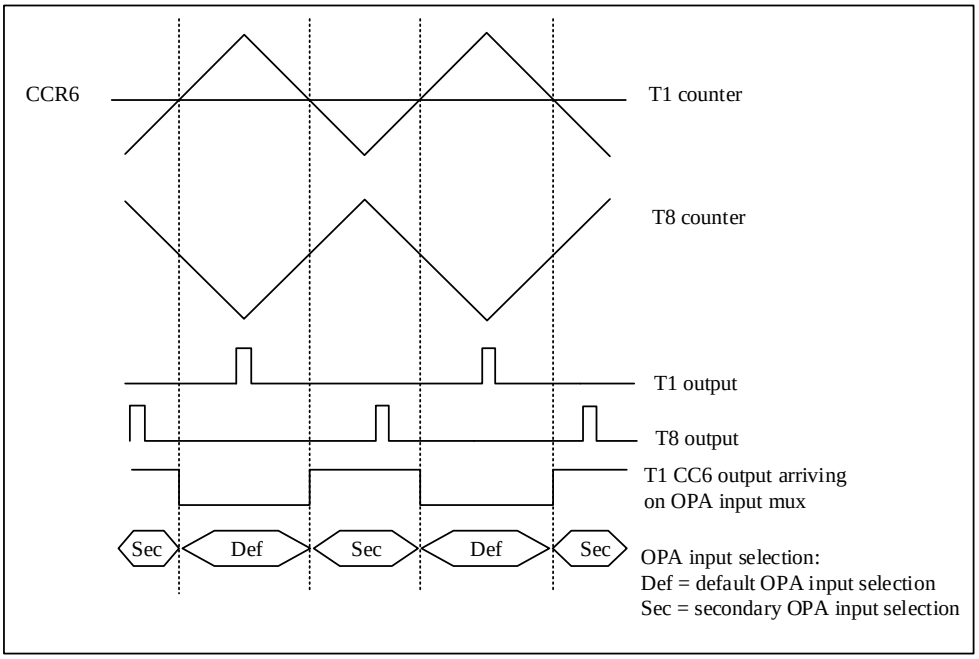


Figure 18-9 timer-controlled multiplexer mode

18.6 OPA low-power modes

Table 18-2 Effect of Low Power Mode on OPA

Mode	Description
Sleep	No effect
Low-power run	No effect
Low-power sleep	No effect
Stop0/Stop1	OPA register value holds, OPA does not work
Standby	The OPA register is powered down and must be reinitialized after exiting Standby mode.

18.7 TIMx registers

18.7.1 OPA1 Control/Status Register (OPA1_CSR)

Offset address 0x00

Reset value 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res.	OPA1_VBSEL	Res										PGA_GAIN		
RW		RW											RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PGA_GAIN		Res.			VBIAS		OPAINTOEN	OPAHSM	VM_SEL		Res	VP_SEL		Res.	OPAEN
RW	RW				RW	RW	RW	RW	RW	RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPA1_CSR register as read-only. 0: OPA1_CSR readable and writeable 1: OPA1_CSR read-only
30	Reserved	-	-	-

29	OPA1_VBSEL	RW	0	OPA1 Common Mode Voltage Selection 0: Common mode voltage select external voltage 1: Common mode voltage select built-in voltage
28: 19	Reserved	-	-	-
18: 14	PGA_GAIN	RW	5'h0	OPA configurable amplifier gain value 5 'b00000: non-inverted internal gain = 2 5 'b00001: non-inverted internal gain = 4 5 'b00010: non-inverted internal gain = 8 5 'b00011: non-inverted internal gain = 16 5 'b00100: non-inverted internal gain = 32 5 'b00101: Not used 5 'b00110: Not used 5 'b00111: Not used 5 'b01000: Inverting gain of VINM0 pin for input or bias =-1/forward gain = 2 5 'b01001: Inverted gain of VINM0 pin for input or bias =-3/forward gain = 4 5 'b01010: Inverting gain of VINM0 pin for input or bias =-7/forward gain = 8 5 'b01011: Inverting gain of VINM0 pin for input or bias =-15/forward gain = 16 5 'b01100: Inverted gain of VINM0 pin for input or bias =-31/forward gain = 32 5 'b01101: Not used 5 'b01110: Not used 5 'b01111: Not used 5 'b10000: Non-inverted gain on VINM0 pin with filtering function = 2 5 'b10001: Non-inverted gain on VINM0 pin with filtering function = 4 5 'b10010: Non-inverted gain on VINM0 pin with filtering function = 8 5 'b10011: Non-inverted gain on VINM0 pin with filtering function = 16 5 'b10100: Non-inverted gain on VINM0 pin with filtering function = 32 5 'b10101: Not used 5 'b10110: not used 5 'b10111: not used 5 'b11000: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-1/non-inverted gain = 2 5 'b11001: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-3/forward gain = 4 5 'b11010: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-7/forward gain = 8 5 'b11011: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-15/forward gain = 16 5 'b11100: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-31/forward gain = 32 5 'b11101: not used 5 'b11110: not used 5 'b11111: not used
13: 11	Reserved	-	-	-
10: 9	OPA_VBIAS	RW	2'h0	OPA1 common mode voltage built-in voltage selection 2'h0: Reserved 2 'h1: VCC/2 2 'h2: VCC/5 2'h3: Reserved
8	OPAINTOEN	RW	0	OPA internal output enabled. 0: OPA output connected to output pin 1: The OPA output is connected to an ADC channel while not connected to the output pin.
7	OPAHSM	RW	0	OPA High Speed Mode. The op-amp must be disabled to change this configuration. 0: OPA works in normal mode 1: OPA works in high-speed mode
6: 5	VM_SEL	RW	2'h0	Reverse input selection 2 'h0: VINM0 pin is connected to OPA1 VINM input 2 'h1: VINM1 pin is connected to OPA1 VINM input 2 'h2: Feedback resistor connected to OPA1 VINM input

				(PGA mode), inverting input selection depends on PGA_GAIN setting 2'h3: Reserved
4	Reserved	-	-	-
3: 2	VP_SEL	RW	2'h0	Non-reverse input selection 2 'h0: VINP0 pin is connected to OPA1 VINP input 2 'h1: VINP1 pin is connected to OPA1 VINP input 2 'h2: VINP2 pin is connected to OPA1 VINP input 2 'h3: DAC1_CH1 pin connected to OPA1 VINP input
1	Reserved	-	-	-
0	OPAEN	RW	0	OPA enabled 0: OPA prohibited 1: Enable OPA

18.7.2 OPA2 Control/Status Register (OPA2_CSR)

Offset address 0x04

Reset value 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res.	OPA2_VBSEL	Res										PGA_GAIN		
RW		RW											RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PGA_GAIN		Res.					OPAINTOEN	OPAHSM	VM_SEL		Res	VP_SEL		Res.	OPAEN
RW	RW						RW	RW	RW	RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPA1_CSR register as read-only. 0: OPA2_CSR readable and writeable 1: OPA2_CSR read-only
30	Reserved	-	-	-
29	OPA2_VBSEL	RW	0	OPA2 Common Mode Voltage Selection 0: Common mode voltage select external voltage 1: Common mode voltage select built-in voltage
28: 19	Reserved	-	-	-
18: 14	PGA_GAIN	RW	5'h0	OPA configurable amplifier gain value 5 'b00000: non-inverted internal gain = 2 5 'b00001: non-inverted internal gain = 4 5 'b00010: non-inverted internal gain = 8 5 'b00011: non-inverted internal gain = 16 5 'b00100: non-inverted internal gain = 32 5 'b00101: Not used 5 'b00110: Not used 5 'b00111: Not used 5 'b01000: Inverting gain of VINM0 pin for input or bias =-1/forward gain = 2 5 'b01001: Inverted gain of VINM0 pin for input or bias =-3/forward gain = 4 5 'b01010: Inverting gain of VINM0 pin for input or bias =-7/forward gain = 8 5 'b01011: Inverting gain of VINM0 pin for input or bias =-15/forward gain = 16 5 'b01100: Inverted gain of VINM0 pin for input or bias =-31/forward gain = 32 5 'b01101: Not used 5 'b01110: Not used 5 'b01111: Not used 5 'b10000: Non-inverted gain on VINM0 pin with filtering function = 2 5 'b10001: Non-inverted gain on VINM0 pin with filtering function = 4 5 'b10010: Non-inverted gain on VINM0 pin with filtering function = 8

				5 'b10011: Non-inverted gain on VINM0 pin with filtering function = 16 5 'b10100: Non-inverted gain on VINM0 pin with filtering function = 32 5 'b10101: Not used 5 'b10110: not used 5 'b10111: not used 5 'b11000: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-1/non-inverted gain = 2 5 'b11001: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-3/forward gain = 4 5 'b11010: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-7/forward gain = 8 5 'b11011: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-15/forward gain = 16 5 'b11100: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-31/forward gain = 32 5 'b11101: not used 5 'b11110: not used 5 'b11111: not used
13: 9	Reserved	-	-	-
8	OPAINTOEN	RW	0	OPA internal output enabled. 0: OPA output connected to output pin 1: The OPA output is connected to an ADC channel while not connected to the output pin.
7	OPAHSM	RW	0	OPA High Speed Mode. The op-amp must be disabled to change this configuration. 0: OPA works in normal mode 1: OPA works in high-speed mode
6: 5	VM_SEL	RW	2'h0	Reverse input selection 2 'h0: VINM0 pin is connected to OPA2 VINM input 2 'h1: VINM1 pin is connected to OPA2 VINM input 2 'h2: Feedback resistor connected to OPA2 VINM input (PGA mode), inverting input selection depends on PGA_GAIN setting 2'h3: Reserved
4	Reserved	-	-	-
3: 2	VP_SEL	RW	2'h0	Non-reverse input selection 2 'h0: VINP0 pin is connected to OPA2 VINP input 2 'h1: VINP1 pin is connected to OPA2 VINP input 2 'h2: The VINP2 pin is connected to the OPA2 VINP input 2 'h3: VINP3 pin is connected to OPA2 VINP input
1	Reserved	-	-	-
0	OPAEN	RW	0	OPA enabled 0: OPA prohibited 1: Enable OPA

18.7.3 OPA3Control/Status Register (OPA3_CSR)

Offset address 0x08

Reset value 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res.	OPA3_VBSEL	Res										PGA_GAIN		
RW		RW											RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PGA_GAIN		Res.					OPAINTOEN	OPAHSM	VM_SEL		Res	VP_SEL		Res.	OPAEN
RW	RW						RW	RW	RW	RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPA1_CSR register as read-only. 0: OPA3_CSR readable and writeable

				1: OPA3_CSR read-only
30	Reserved	-	-	-
29	OPA3_VBSEL	RW	0	OPA3 Common Mode Voltage Selection 0: Common mode voltage select external voltage 1: Common mode voltage select built-in voltage
28: 19	Reserved	-	-	-
18: 14	PGA_GAIN	RW	5'h0	OPA configurable amplifier gain value 5 'b00000: non-inverted internal gain = 2 5 'b00001: non-inverted internal gain = 4 5 'b00010: non-inverted internal gain = 8 5 'b00011: non-inverted internal gain = 16 5 'b00100: non-inverted internal gain = 32 5 'b00101: Not used 5 'b00110: Not used 5 'b00111: Not used 5 'b01000: Inverting gain of VINM0 pin for input or bias =-1/forward gain = 2 5 'b01001: Inverted gain of VINM0 pin for input or bias =-3/forward gain = 4 5 'b01010: Inverting gain of VINM0 pin for input or bias =-7/forward gain = 8 5 'b01011: Inverting gain of VINM0 pin for input or bias =-15/forward gain = 16 5 'b01100: Inverted gain of VINM0 pin for input or bias =-31/forward gain = 32 5 'b01101: Not used 5 'b01110: Not used 5 'b01111: Not used 5 'b10000: Non-inverted gain on VINM0 pin with filtering function = 2 5 'b10001: Non-inverted gain on VINM0 pin with filtering function = 4 5 'b10010: Non-inverted gain on VINM0 pin with filtering function = 8 5 'b10011: Non-inverted gain on VINM0 pin with filtering function = 16 5 'b10100: Non-inverted gain on VINM0 pin with filtering function = 32 5 'b10101: Not used 5 'b10110: not used 5 'b10111: not used 5 'b11000: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-1/non-inverted gain = 2 5 'b11001: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-3/forward gain = 4 5 'b11010: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-7/forward gain = 8 5 'b11011: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-15/forward gain = 16 5 'b11100: Inverted gain of VINM0 pin for input or bias and VINM1 pin for filtering =-31/forward gain = 32 5 'b11101: not used 5 'b11110: not used 5 'b11111: not used
13: 9	Reserved	-	-	-
8	OPAINTOEN	RW	0	OPA internal output enabled. 0: OPA output connected to output pin 1: The OPA output is connected to an ADC channel while not connected to the output pin.
7	OPAHSM	RW	0	OPA High Speed Mode. The op-amp must be disabled to change this configuration. 0: OPA works in normal mode 1: OPA works in high-speed mode
6: 5	VM_SEL	RW	2'h0	Reverse input selection 2 'h0: VINM0 pin is connected to OPA3 VINM input 2 'h1: VINM1 pin is connected to OPA3 VINM input 2 'h2: Feedback resistor connected to OPA3 VINM input (PGA mode), inverting input selection depends on PGA_GAIN setting 2'h3: Reserved

4	Reserved	-	-	-
3: 2	VP_SEL	RW	2'h0	Non-reverse input selection 2 'h0: VINP0 pin is connected to OPA3 VINP input 2 'h1: VINP1 pin is connected to OPA3 VINP input 2 'h2: VINP2 pin is connected to OPA3 VINP input 2 'h3: DAC2_CH1 pin is connected to OPA3 VINP input
1	Reserved	-	-	-
0	OPAEN	RW	0	OPA enabled 0: OPA prohibited 1: Enable OPA

18.7.4 OPA1 timer control mode register (OPA1_TCMR)

Offset address 0x0C

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res														
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res											T8CM_EN	T1CM_EN	VPS_SEL		VMS_SEL
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	OPA1_TCMR lock. The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPAx_TCMR register as read-only. 0: OPA1_TCMR readable and writeable 1: OPA1_TCMR read-only
30: 5	Reserved	-	-	-
4	T8CM_EN	RW	0	TIM8 controls multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM8CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
3	T1CM_EN	RW	0	TIM1 controls the multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM1 CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
2: 1	VPS_SEL	RW	2'h0	OPA1 non-inverting input secondary selection. These bits are set and cleared by the software. They are used to select the OPA1 non-inverting input when controlled multiplexing mode is enabled (T1CM_EN = 1 or T8CM_EN = 1 2 'h0: VINP0 pin is connected to OPA1 VINP input 2 'h1: VINP1 pin is connected to OPA1 VINP input 2 'h2: VINP2 pin is connected to OPA1 VINP input 2 'h3: DAC1_CH1 pin connected to OPA1 VINP input
0	VMS_SEL	RW	0	OPA1 inverting input secondary selection. This bit is set and cleared by the software. When controlled multiplexer mode is enabled (T1CM_EN = 1 or T8CM_EN = 1), when standalone mode is used (i.e. VM_SEL = "2 'h0" or "2' h1"), it is used to select the OPA1 inverting input: 0: Input from VINM0 1: Input from VINM1 When using PGA (VM_SEL = "2 'h2"):This bit can only be set to 0.

18.7.5 OPA2 timer control mode register(OPA2_TCMR)

Offset address 0x10

Reset value 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res														
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res											T8CM_EN	T1CM_EN	VPS_SEL		VMS_SEL
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	OPA2_TCMR lock. The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPAx_TCMR register as read-only. 0: OPA2_TCMR readable and writeable 1: OPA2_TCMR read-only
30: 5	Reserved	-	-	-
4	T8CM_EN	RW	0	TIM8 controls multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM8CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
3	T1CM_EN	RW	0	TIM1 controls the multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM1 CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
2: 1	VPS_SEL	RW	2'h0	OPA2 non-inverting input secondary selection. These bits are set and cleared by the software. They are used to select the OPA2 non-inverting input when controlled multiplexing mode is enabled (T1CM_EN = 1 or T8CM_EN = 1 2 'h0: VINP0 pin is connected to OPA2 VINP input 2 'h1: VINP1 pin is connected to OPA2 VINP input 2 'h2: The VINP2 pin is connected to the OPA2 VINP input 2 'h3: VINP3 pin is connected to OPA2 VINP input
0	VMS_SEL	RW	0	OPA2 inverting input secondary selection. This bit is set and cleared by the software. When controlled multiplexer mode is enabled (T1CM_EN = 1 or T8CM_EN = 1), when standalone mode is used (i.e. VM_SEL = "2 'h0" or "2' h1"), it is used to select the OPA2 inverting input: 0: Input from VINM0 1: Input from VINM1 When using PGA (VM_SEL = "2 'h2"):This bit can only be set to 0.

18.7.6 OPA3 timer control mode register (OPA3_TCMR)

Offset address 0x14

Reset value 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	Res														
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res											T8CM_EN	T1CM_EN	VPS_SEL		VMS_SEL
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	OPA3_TCMR lock. The bit is written once. It is set by software. It can only be cleared by system reset or module soft reset. This bit is used to configure the OPAx_TCMR register as read-only. 0: OPA3_TCMR readable and writeable 1: OPA3_TCMR read-only
30: 5	Reserved	-	-	-
4	T8CM_EN	RW	0	TIM8 controls multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM8CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
3	T1CM_EN	RW	0	TIM1 controls the multiplexing mode enabled. This bit is set and cleared by the software. It is used to automatically control the switching between default selections (VP_SEL and VM_SEL) and secondary selections (VPS_SEL and VMS_SEL) for inverting and non-inverting inputs. This automatic switch is triggered by the TIM1 CC6 output arriving at the OPAx input multiplexer. 0: Automatic switching off 1: Automatic switching enabled
2: 1	VPS_SEL	RW	2'h0	OPA3 non-inverting input secondary selection. These bits are set and cleared by the software. They are used to select the OPA3 non-inverting input when controlled multiplexing mode is enabled (T1CM_EN = 1 or T8CM_EN = 1 2 'h0: VINP0 pin is connected to OPA3 VINP input 2 'h1: VINP1 pin is connected to OPA3 VINP input 2 'h2: VINP2 pin is connected to OPA3 VINP input 2 'h3: DAC2_CH1 pin is connected to OPA3 VINP input
0	VMS_SEL	RW	0	OPA3 inverting input secondary selection. This bit is set and cleared by the software. When controlled multiplexer mode is enabled (T1CM_EN = 1 or T8CM_EN = 1), when standalone mode is used (i.e. VM_SEL = "2 'h0" or "2' h1"), it is used to select the OPA3 inverting input: 0: Input from VINM0 1: Input from VINM1 When using PGA (VM_SEL = "2 'h2"):This bit can only be set to 0.

19. Advanced Timer (TIM1 and TIM8)

19.1 Introduction

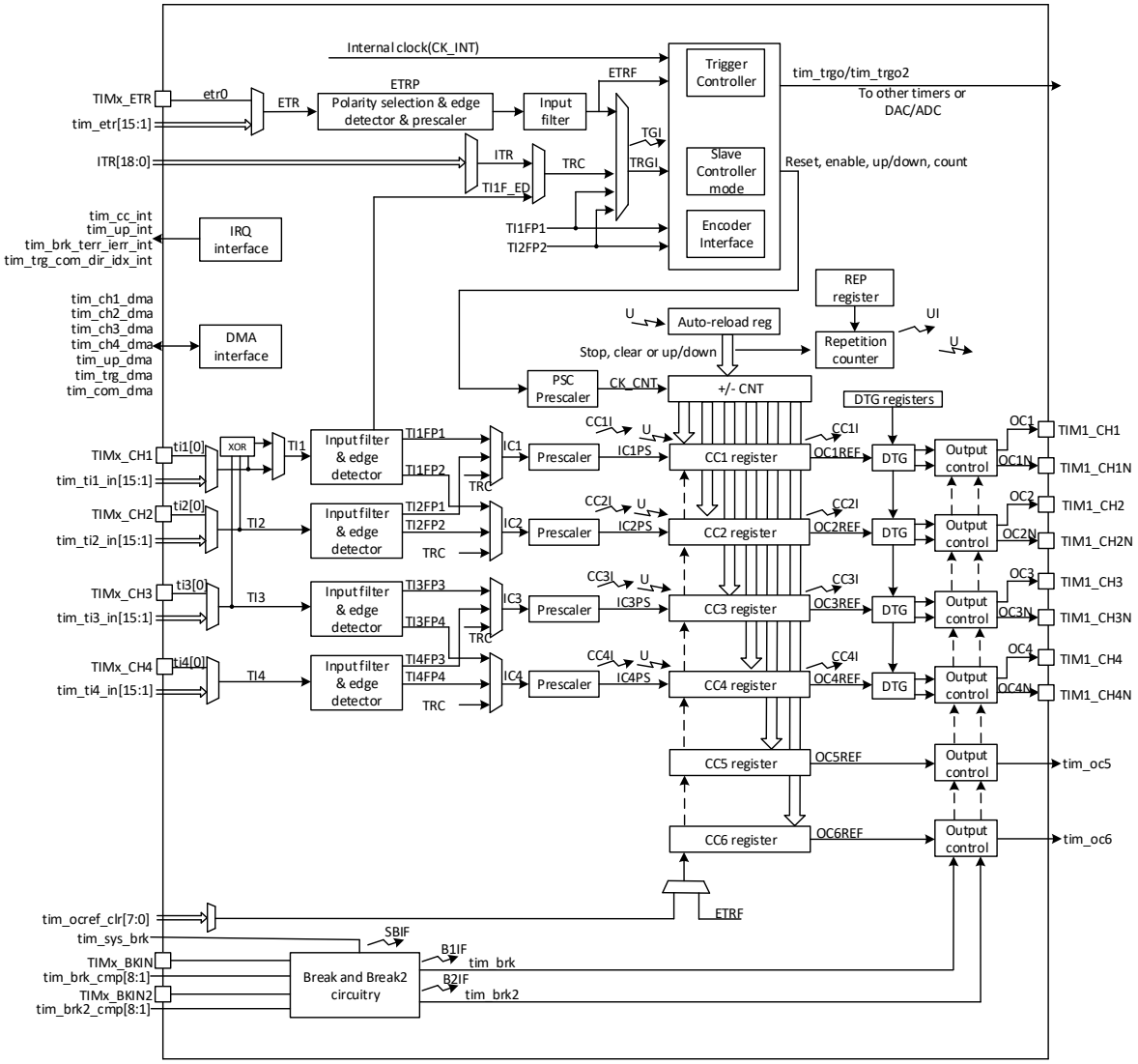
The advanced control timers (TIM1, TIM8) consist of a 16-bit auto-load counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The advanced-control (TIM1) and general-purpose (TIMx) timers are completely independent, and do not share any resources. They can be synchronized together.

19.1.1 TIM1/TIM8 Main Features

The functions of the TIM1 timer include:

- 16-bit up, down, up/down auto load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- Up to 6 independent channels:
  - Input capture (5/6 channel not supported)
  - Output compare
  - PWM generation (edge or middle alignment mode)
  - One-pulse mode output
- Complementary output with programmable dead time
- Synchronization circuit for controlling timer and interconnection between timer by external signal
- Allows the repetition counter of the timer register to be updated after a specified number of counter cycles
- The two brake input signals can put the timer output signal into a reset state or a known state
- An interrupt/DMA occurs when the following events occur:
  - Update: Counter upflow/downflow, counter initialization (via software or internal/external trigger)
  - Trigger event (counter starts, stops, initializes or counts by internal/external triggers)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoders and Hall sensor circuits for positioning
- Trigger input for external clock or cycle-by-cycle current management

19.1.2 CAN block diagram



19.2 TIM1/TIM8 function description

19.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)
- Repetition Register (TIMx_RCR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition (overflow

or underflow) and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set. (See the description of the slave mode controller for more details on enabling counters).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

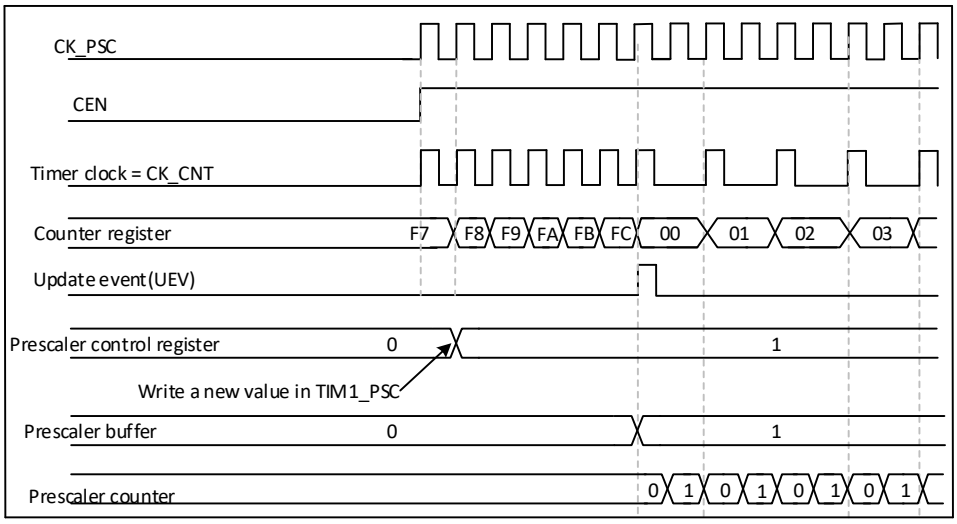


Figure 19-1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

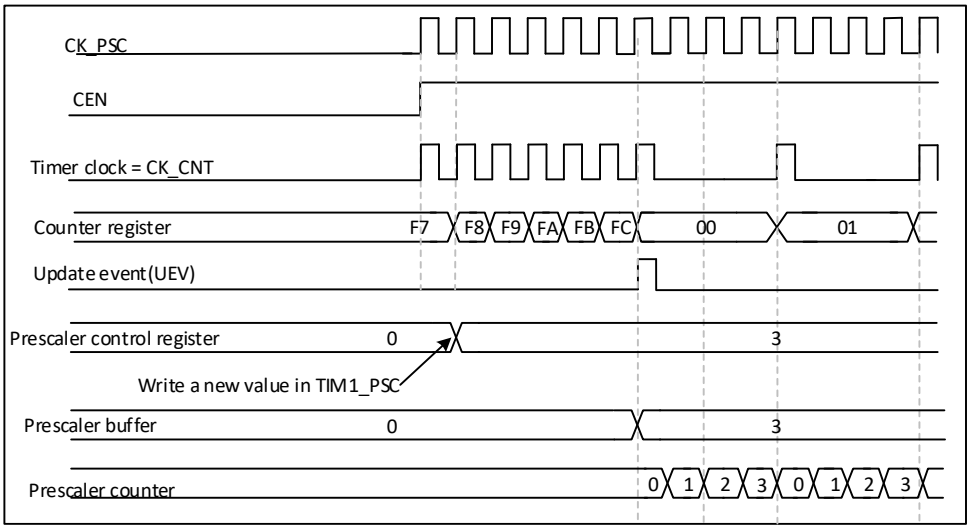


Figure 19-2 Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

19.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event.

If a repetition counter is used, the update event (UEV) will not be generated until the number of overflows reaches the value of the configured repetition count register plus one (i.e. TIMx_RCR+1); If a repetition counter is not used (i.e., TIMx_RCR = 0), an update event will be generated every time the count overflows.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt or DMA request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

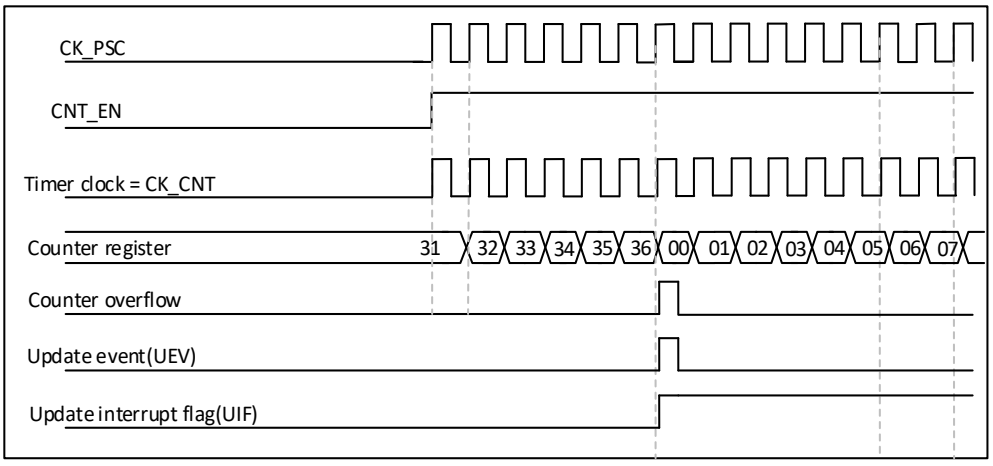


Figure 19-3 Counter timing diagram, internal clock divided by 1

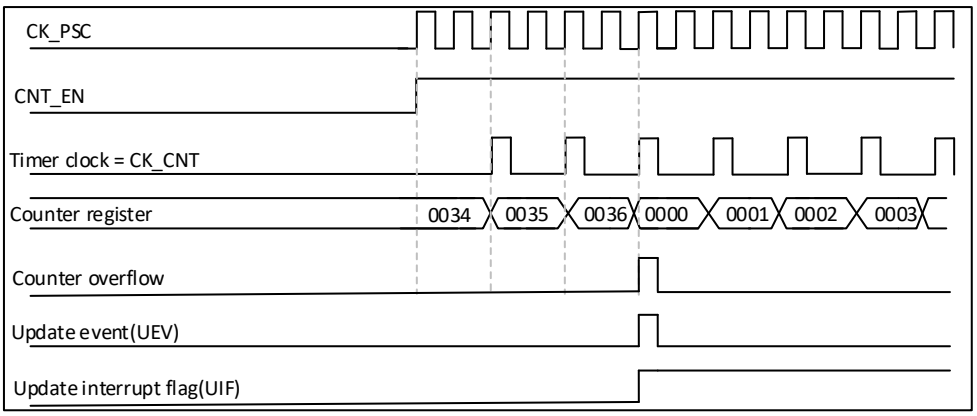


Figure 19-4 Counter timing diagram, internal clock divided by 2

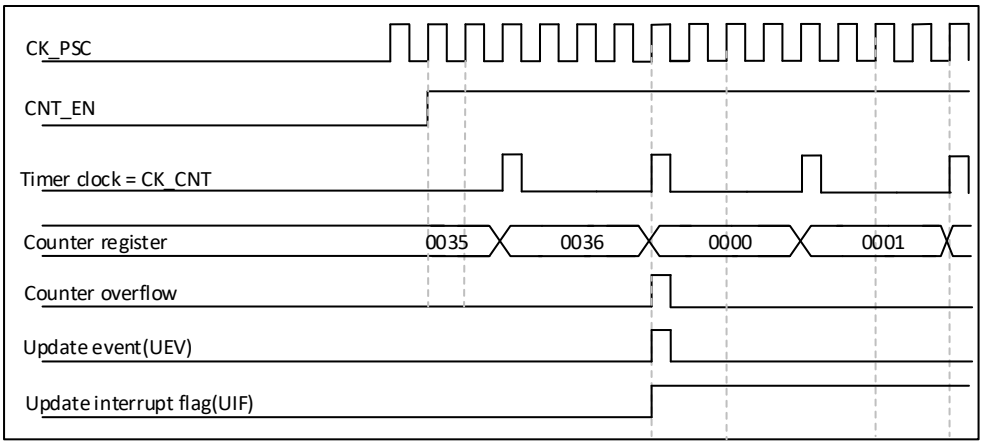


Figure 19-5 Counter timing diagram, internal clock divided by 4

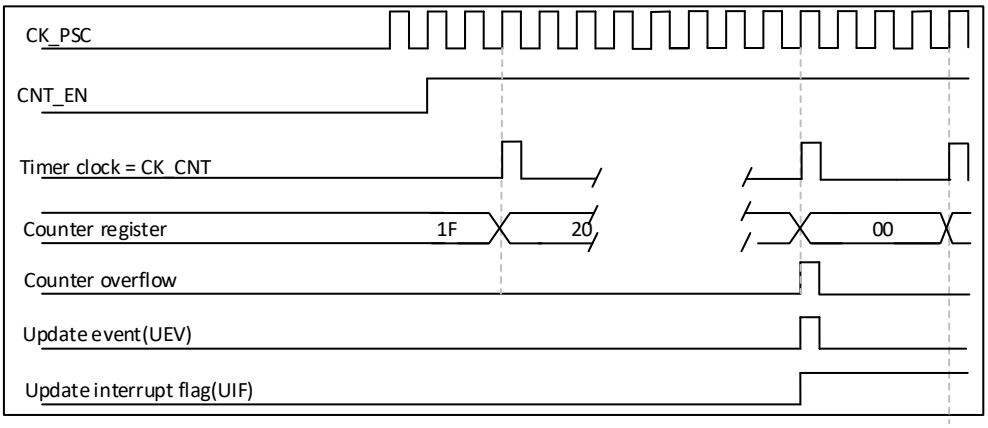


Figure 19-6 Counter timing diagram, internal clock divided by N

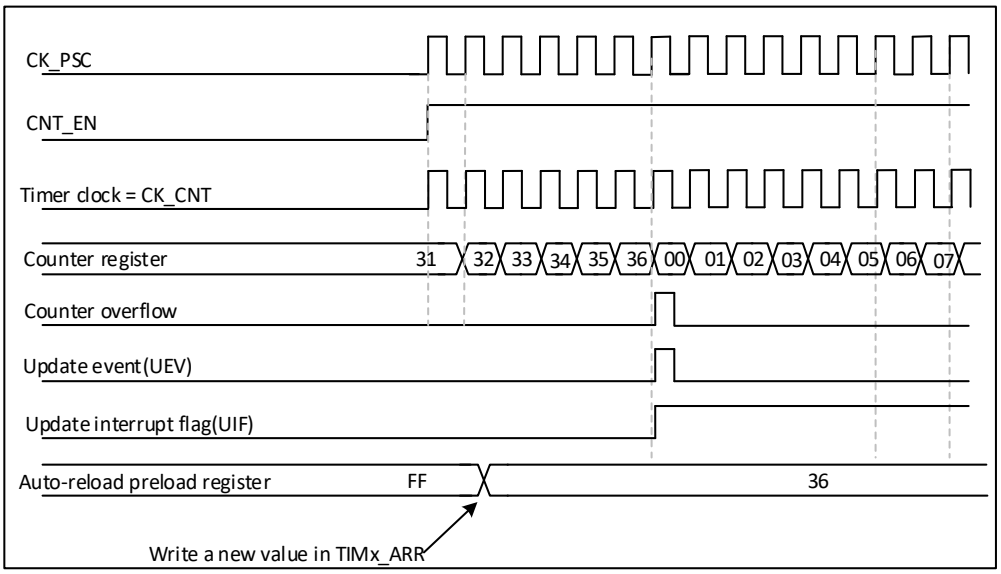


Figure 19-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

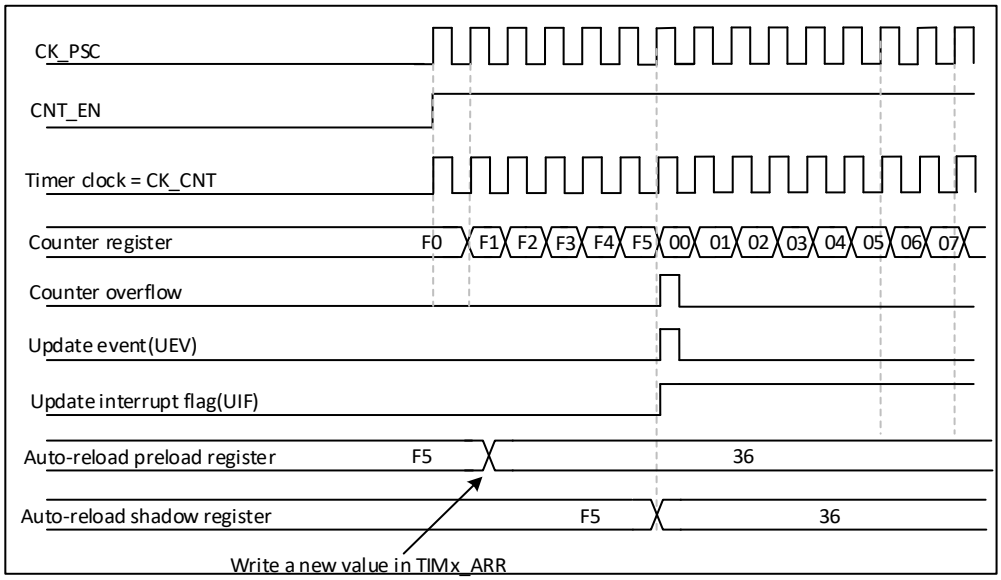


Figure 19-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

Downcounting mode

In down-count mode, the counter starts from the auto-loaded value (the contents of TIMx_ARR) and counts down to 0, then restarts from the auto-loaded value and generates a count underflow event.

If a repetition counter is used, the update event (UEV) will not be generated until the number of underflows reaches the value of the configured repetition count register plus one (i.e. TIMx_RCR+1); If a repetition counter is not used (i.e., TIMx_RCR = 0), an update event will be generated every time the count underflows.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. Even then, when an update event should

occur, the counter will restart counting from the current autoload value while the counter inside the prescaler is cleared '0' (but the prescaling factor remains unchanged).

In addition, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit is not set (i.e., no interrupt or DMA request will be generated), in order to avoid clearing the counter when a capture event occurs, and generating both an update and a capture interrupt.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The repetition counter is reloaded as the contents of the TIMx_RCR register.
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note: The autoload value is updated before the counter is reloaded, so the next cycle will be the expected value.

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

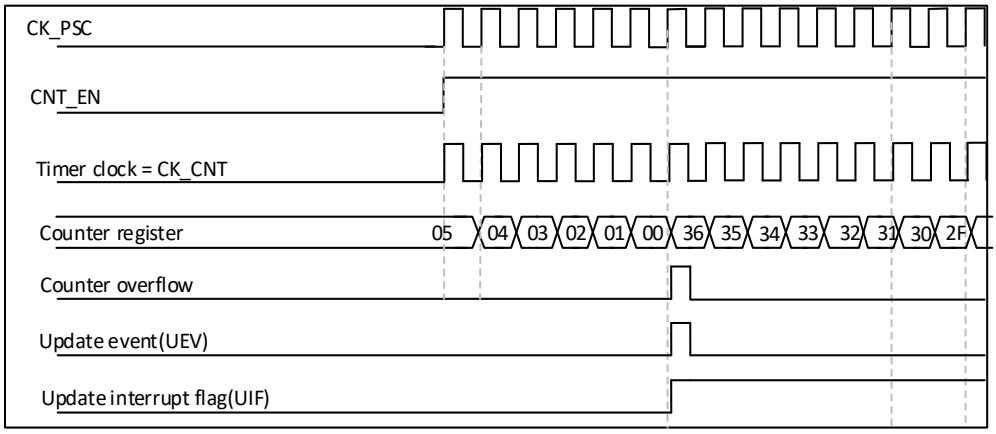


Figure 19-9 Counter timing diagram, internal clock divided by 1

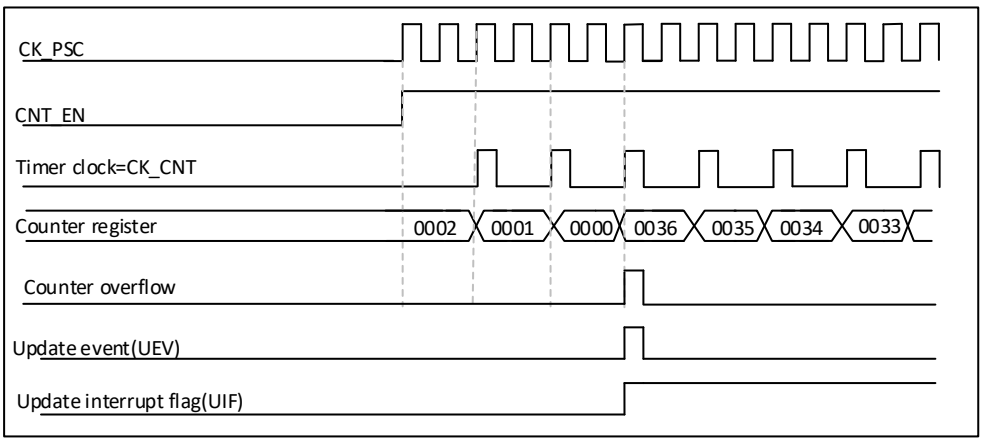


Figure 19-10 Counter timing diagram, internal clock divided by 2

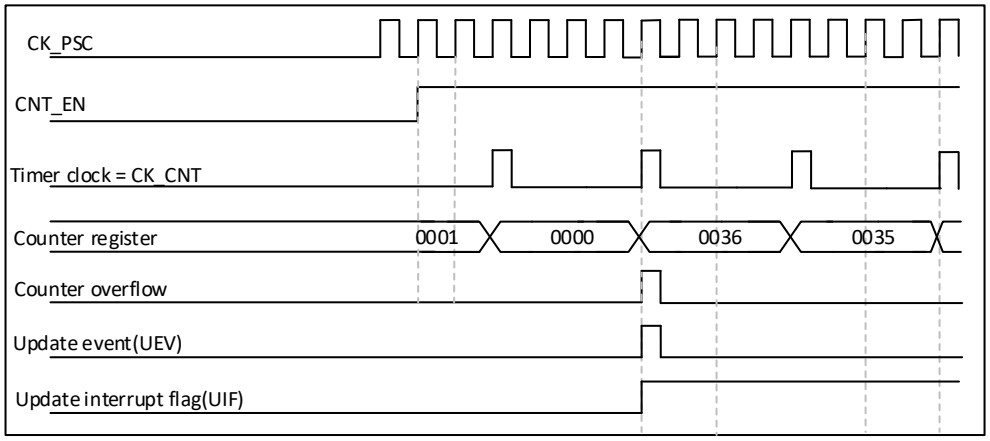


Figure 19-11 Counter timing diagram, internal clock divided by 4

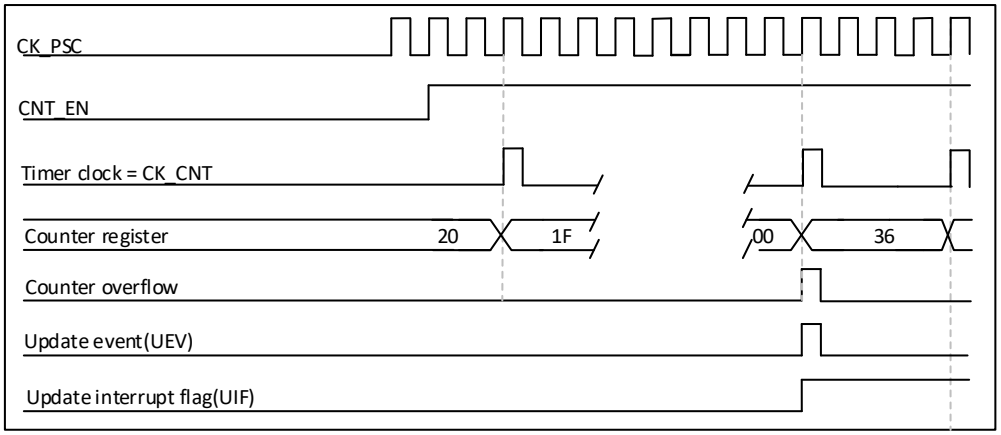


Figure 19-12 Counter timing diagram, internal clock divided by N

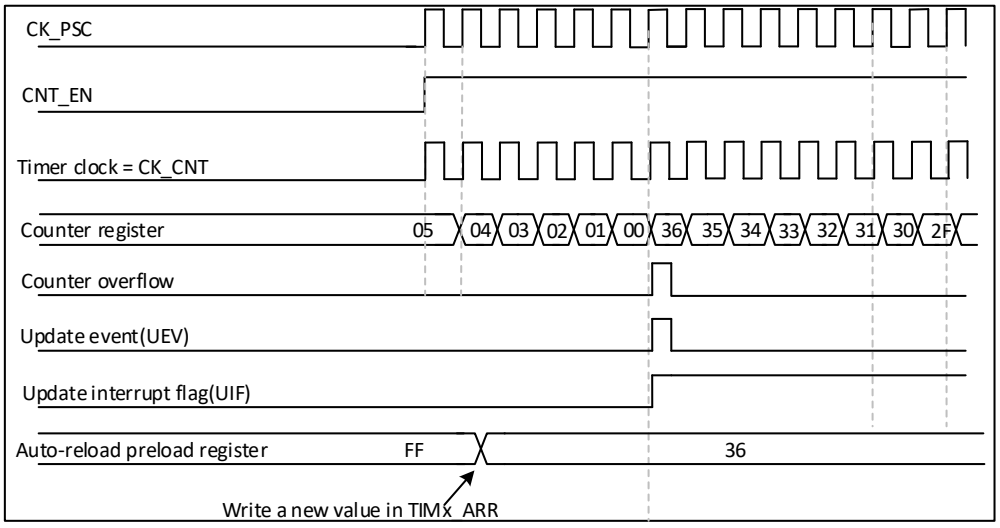


Figure 19-13 Counter timing diagram, update event when repetition counter is not used

Central alignment mode (up/down count)

In central alignment mode, the counter starts counting from 0 to the auto-loaded value (TIMx_ARR register)minus1, generating a counter overflow event, and then counts down to 1 and generating a counter underflow event; Then re-count from 0 again.

The center alignment mode can be obtained by configuring the CMS bit in the TIMx_CR1 register not to be '00'. The output comparison flag bit with the channel configured as output mode is set during the following counting processes: when counting down (center alignment mode 1, CMS = '01'), when counting up (center alignment mode 2, CMS = '10'), and when counting up and down (center alignment mode 3, CMS = '11').

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event. The counter then starts counting again from 0, and the prescaler internal counter also starts counting again from 0.

Setting the UDIS bit in the TIMx_CR1 register can disable the update event. This is to avoid updating the shadow registers while writing new values in the preload registers. Although no update event will occur until the UDIS bit is cleared to 0, the counter will continue to count up or down based on the value of the current auto-reload.

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), by setting the UG bit, an update event UEV will be generated but the UIF flag will not be set (thus no interrupts and DMA requests are generated), in order to avoid clearing the counter when a capture event occurs, while generating both an update and a capture interrupt.

When an update event occurs, all registers are updated and (according to the setting of the URS bit) an update flag bit (UIF bit in the TIMx_SR register) is also set:

- The repetition counter is reloaded with the content of TIMx_RCR register
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register). Note: If an update occurs due to a counter overflow, the automatic reload will be updated before the counter is reloaded, so the next cycle will be the expected value (the counter is loaded as a new value).

Here are some examples of how the counter operates at different clock frequencies:

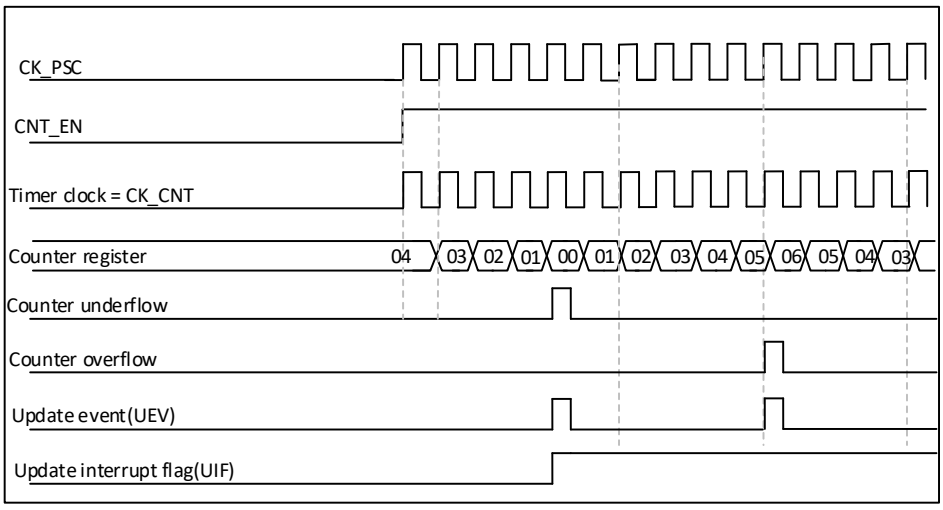


Figure 19-14 Counter timing diagram, internal clock division factor is 1, TIMx_ARR = 0x6

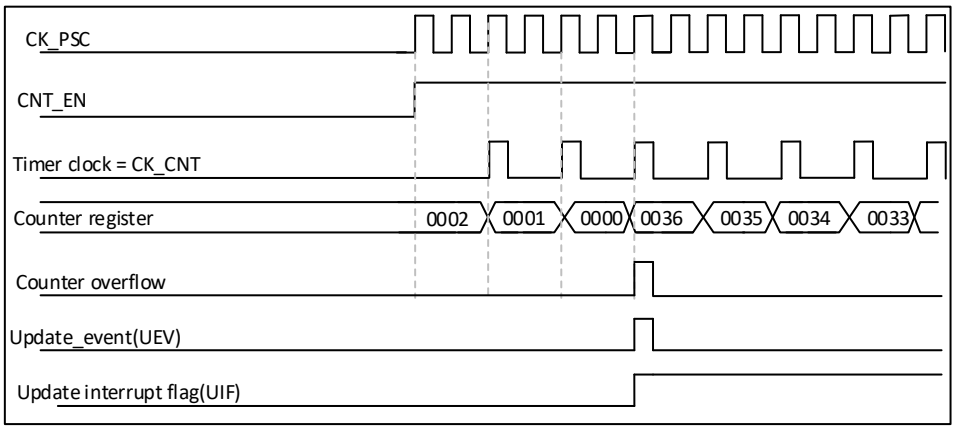


Figure 19-15 Counter timing diagram, internal clock divided by 2

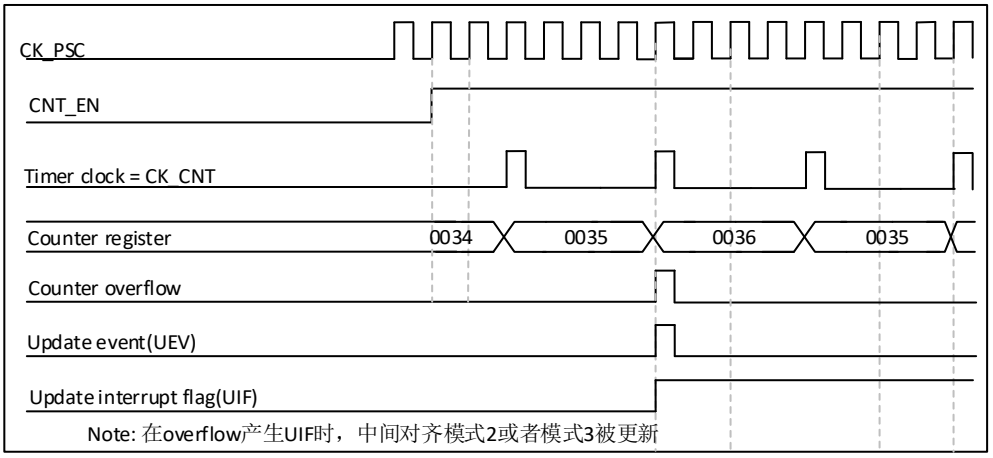


Figure 19-16 Counter timing diagram, internal clock division factor is 4, TIMx_ARR = 0x36

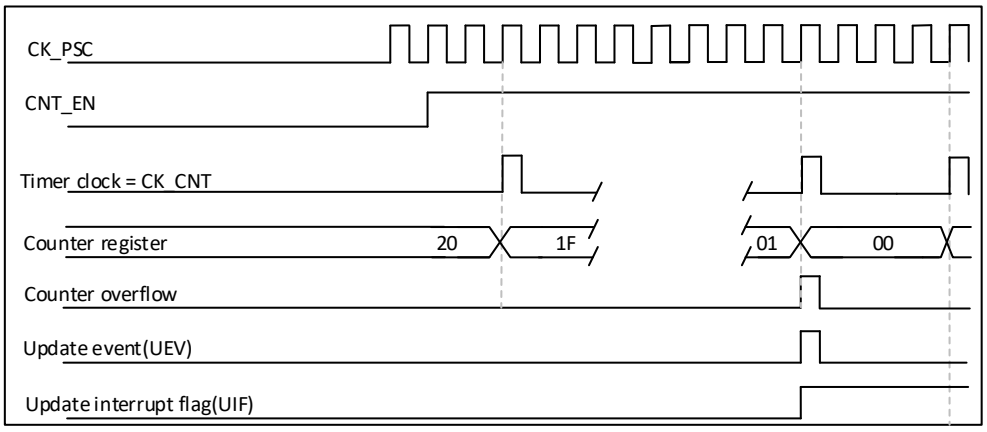


Figure 19-17 Counter timing diagram, internal clock divided by N

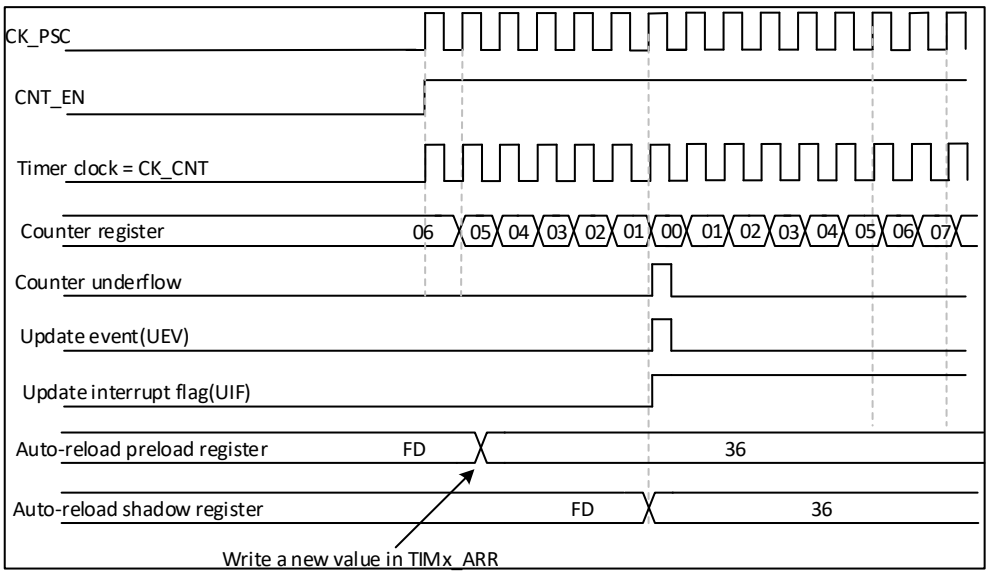


Figure 19-18 Counter timing diagram, update event with ARPE=1 (counter underflow)

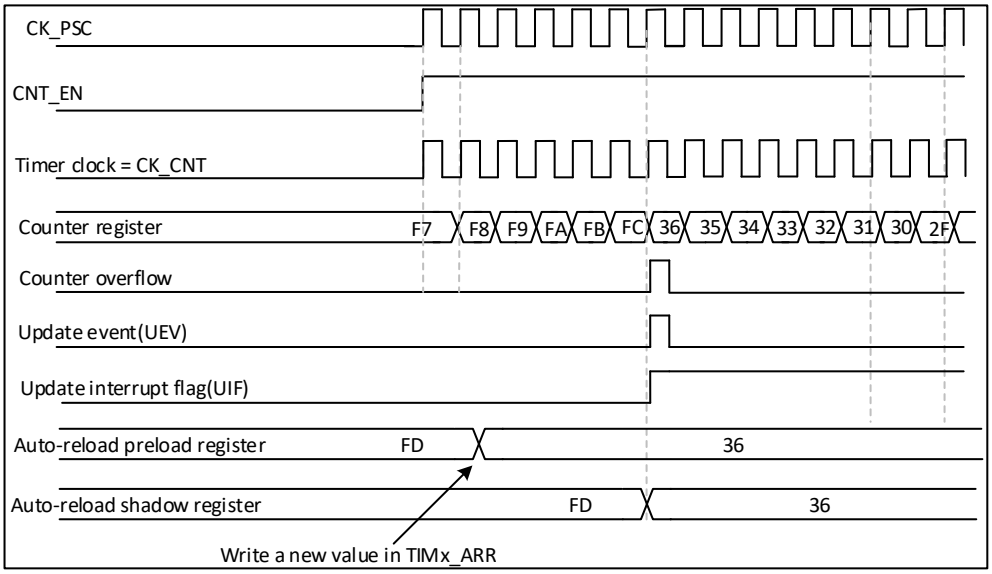


Figure 19-19 Counter Timing Chart, Update Event (Counter Overflow) at ARPE = 1

19.2.3 Repetition counter

The "Time Base Unit" explains how an update event (UEV) occurs when the counter overflows/underflows, in fact it can only occur when the repetition counter count reaches 0. This feature is very useful for generating PWM signals.

This means that data is transferred from the preload register to the shadow register (TIMx_ARR auto-reload register, TIMx_PSC preload register, and capture/compare register TIMx_CCRx in comparison mode) only every N+1 count overflow or underflow, N being the value in the TIMx_RCR repeat count register.

The repetition counter is decremented when either of the following conditions is true:

- Each time the counter overflows in up-count mode,
- Each time the counter underflows in down-count mode,
- At each counter overflow and at each counter underflow in center-aligned mode.

The central alignment modelimits the maximum cycle period of the PWM to 128(TIMx_RCRis an 8-bit counter), but it is able to update the duty cycle twice per PWM cycle.

The repetition counter is automatically reloaded and the repetition rate is defined by the value of the TIMx_RCR register. When the update event is generated by software (by setting the UG bit in TIMx_EGR) or by the slave mode controller of hardware, the update event occurs immediately regardless of the value of the repetition counter, and the contents of the TIMx_RCR register are reloaded to the repetition counter.

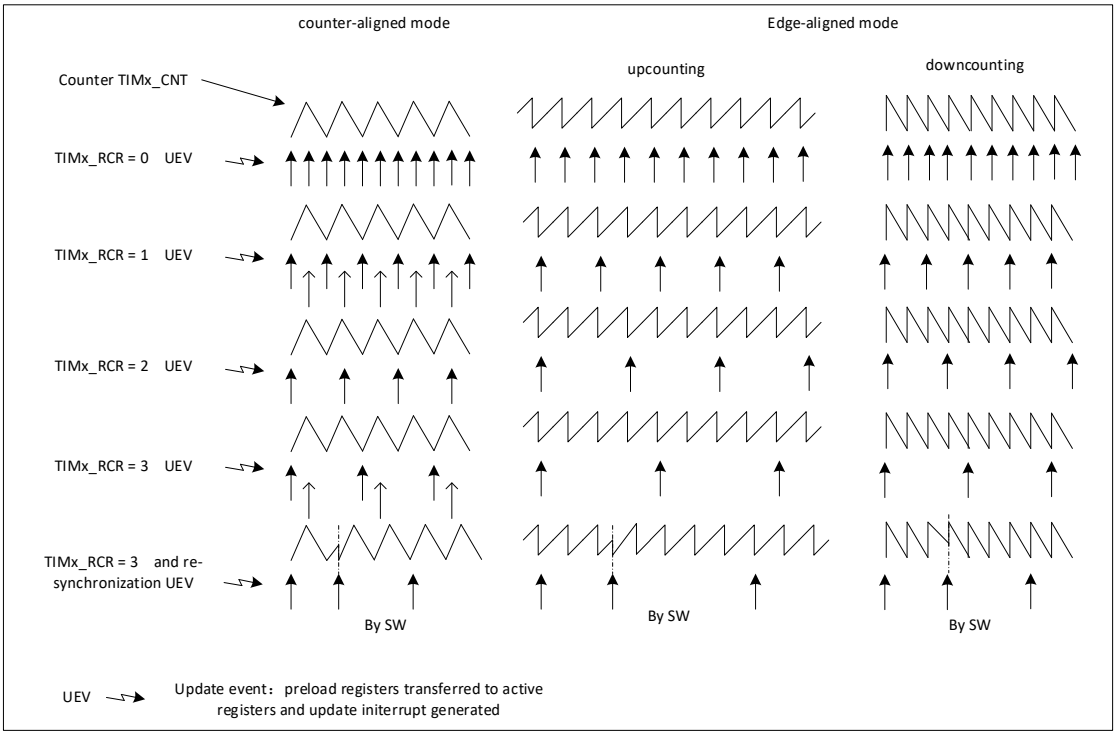


Figure 19-20 Examples of update rates in different modes, and register settings for TIMx_RCR

19.2.4 External trigger input

The timer external trigger input tim_etr can be used to:

- External clock
- From the trigger of the pattern
- As a PWM reset input in periodic current regulation

19.2.5 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pins (TI1 and TI2)
- External clock mode2: external trigger input ETR
- Internal Trigger Input (ITRx): Uses one timer as a prescaler for the other. For example, one timer Timer1 may be configured as a prescaler for another timer Timer2.
- Encoder Mode

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 0000), the CEN, DIR (TIMx_CR1 register), and UG bits (TIMx_EGR register) are de facto control bits and can only be modified by software (the UG bits are still automatically cleared). As long as the CEN bit is written as' 1 ', the clock of the prescaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

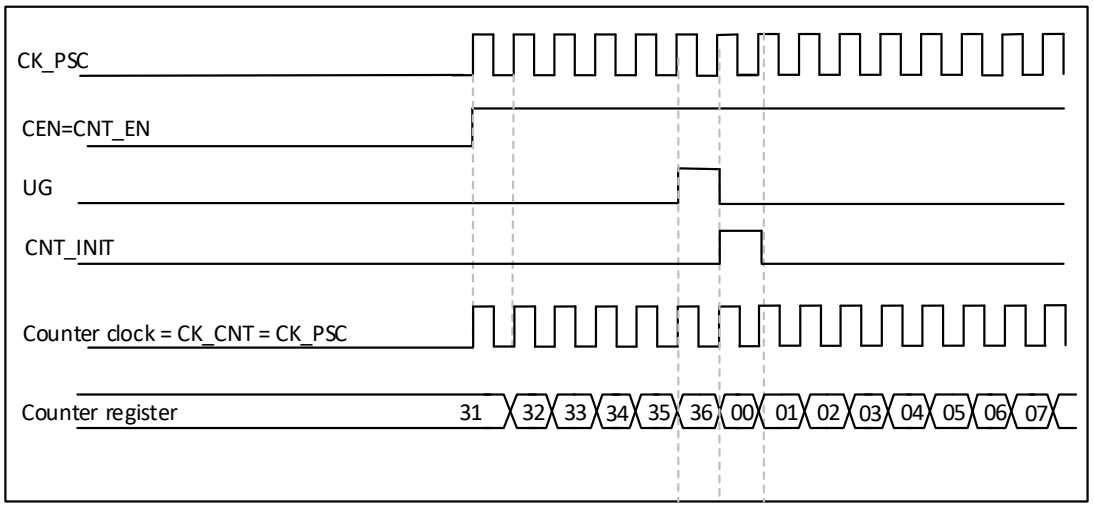


Figure 19-21 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

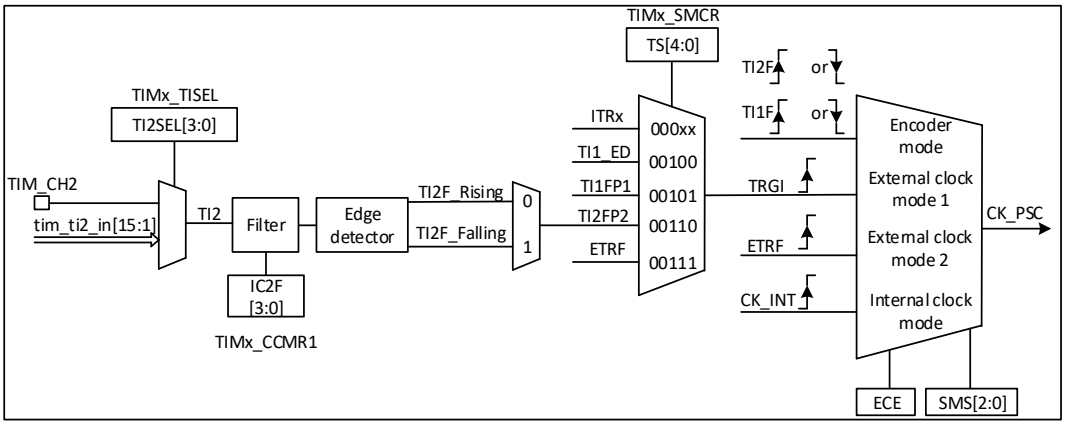


Figure 19-22 TI2 Example of external clock connection

For example, to configure the counter to count up on the rising edge of the TI2 input, use the following steps:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = ‘01’ in the TIMx_CCMR1 register.

2. Configure IC2F [3: 0] of the TIMx_CCMR1 register and select the input filter bandwidth (if no filter is needed, keep IC2F = 0000);

3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.

4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=00110 in the TIMx_SMCR register;

6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the synchronization circuit at the input of TI2.

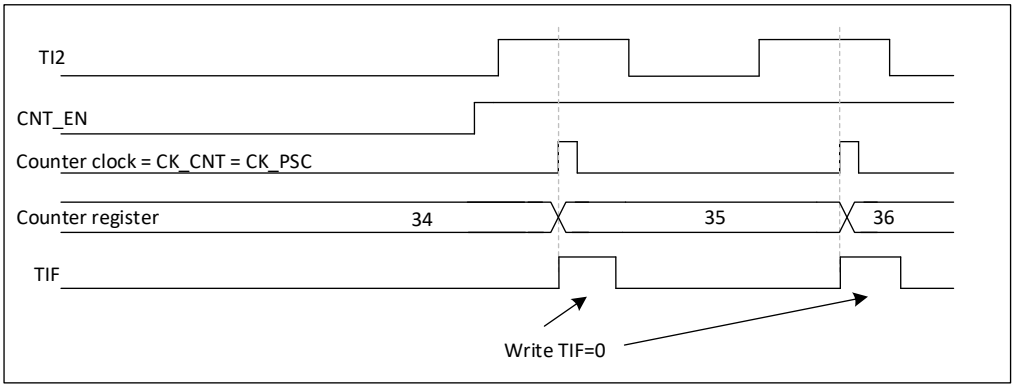


Figure 19-23 Control circuit in external clock mode 1

External clock source mode 2

This mode is selected by making ECE = 1 in the TIMx_SMCR register, and the counter can externally trigger each rising or falling edge of the ETR to count.

The following figure is a block diagram of the external trigger input:

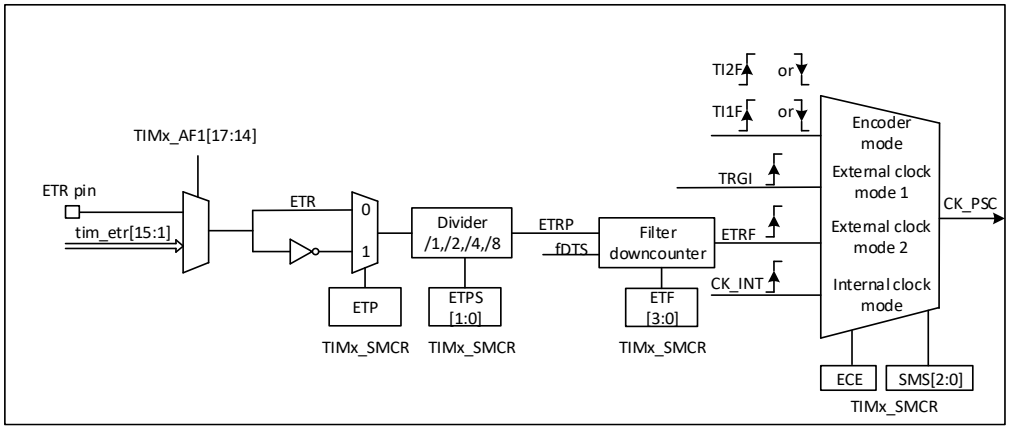


Figure 19-24 External trigger input block diagram

For example, to configure an up-counter that counts every 2 rising edges under ETR, use the following steps:

1. No filter is needed in this example, set ETF [3: 0] = 0000 in the TIMx_SMCR register;

2. Set the prescaler and set ETPS [1: 0] = 01 in the TIMx_SMCR register;

3. Select the rising edge of the ETR input terminal and set ETP = 0 in the TIMx_SMCR register;

4. Turn on external clock mode 2 and write ECE = 1 in the TIMx_SMCR register;

5. Start counter, write CEN = 1 in TIMx_CR1 register;

The counter counts at every 2 ETR rising edges.

The delay between the rising edge of the ETR and the actual clock of the counter depends on the synchronization circuit of the ETRP signal.

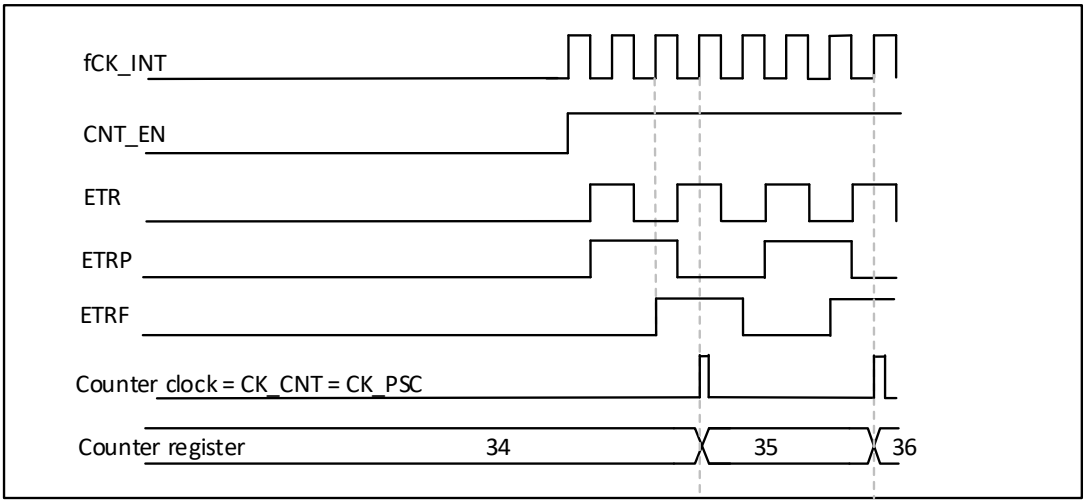


Figure 19-25 Control circuit in external clock mode 2

19.2.6 Capture/Compare channels

Each capture/compare channel is surrounded by a capture/compare register (including a shadow register), including an input portion of the capture (digital filtering, multiplexing and prescaler, except for channels 5 and 6), and an output portion (comparator and output control).

The following figures are capture/compare channel overviews.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge monitor with polarity selection then generates a signal (TIxFPx) that can be triggered as an input from the slave mode controller or as a capture control. It is prescaled before the capture register (ICxPS).

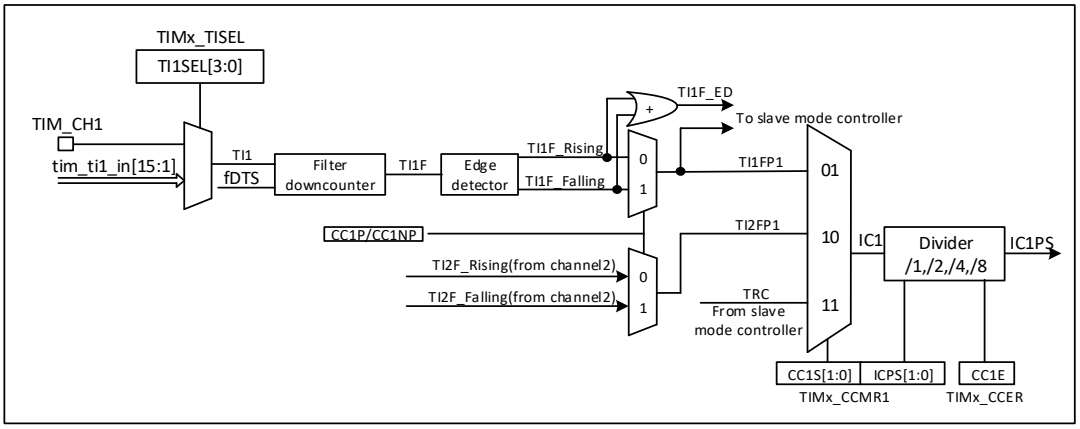


Figure 19-26 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

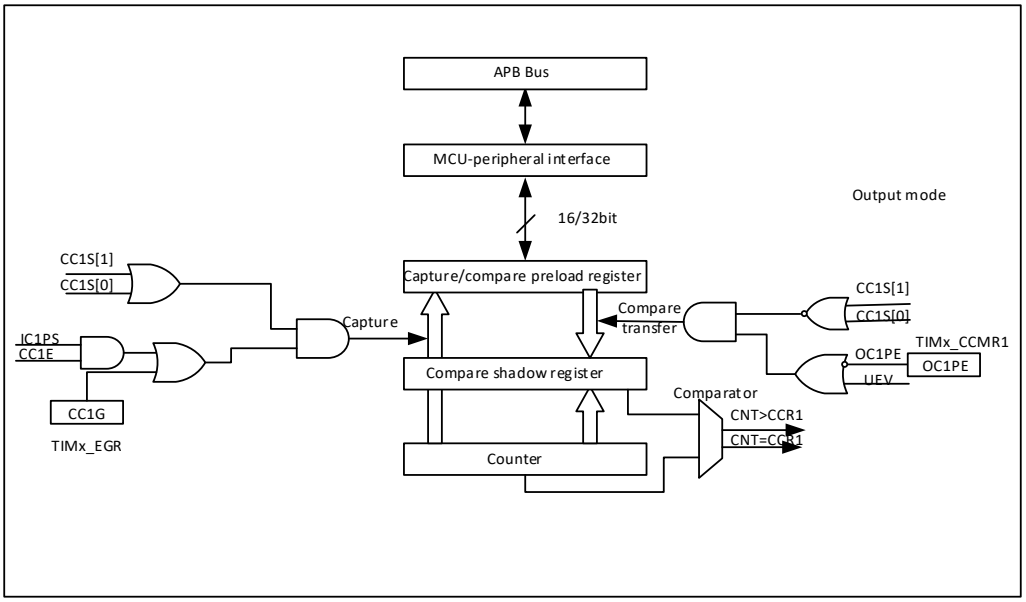


Figure 19-27 Capture/Compare the main circuit of channel 1

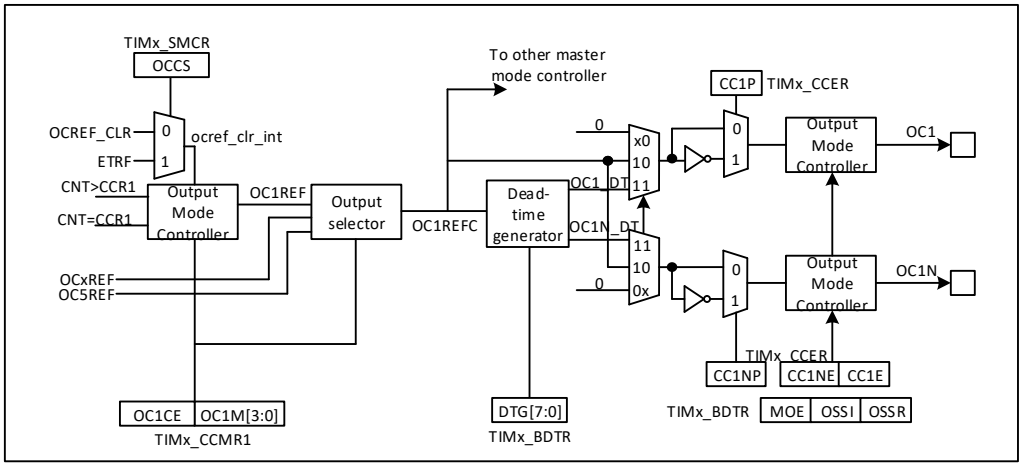


Figure 19-28 Output stage of capture/compare channel (channel 1 to 4)

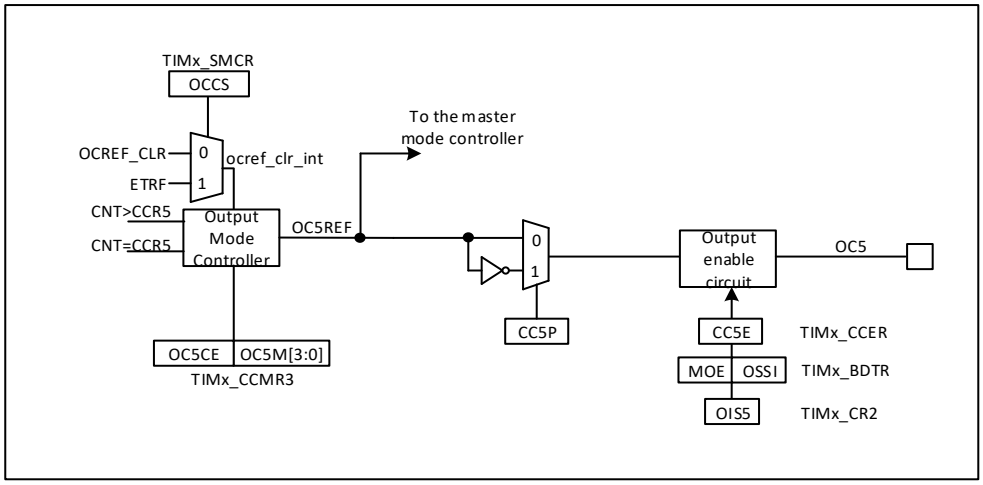


Figure 19-29 Output stage of capture/compare channel (channel 5,6)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

19.2.7 Input capture mode:

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched into the capture/compare register (TIMx_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx_SR register) is set to 1, and if an interrupt or DMA operation is enabled, an interrupt or DMA request will be generated. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (TIMx_SR register) is set to 1. The CCxIF can be cleared by writing CCxIF = 0, or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to ‘0’.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select valid input: TIMx_CCMR1 must be connected to the TI1 input, so CC1S = 01 written in the TIMx_CCMR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let’s imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (set to rising edge) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to ‘00’ in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If desired, the associated interrupt request may be allowed by setting the CC1IE bit in the TIMx_DIER register, or the DMA request may be allowed by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- The CC1IF flag bit is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

19.2.8 PWM input mode

This mode is a special case of the input capture mode, and the rest of the operation is the same as the input capture mode except for the following differences:

- Both ICx signals are mapped to the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, the user can measure the cycle (TIMx_CCR1 register) and the duty cycle (TIMx_CCR2 register) of the PWM signal input to TI1 as follows

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to ‘0’ (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to ‘1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

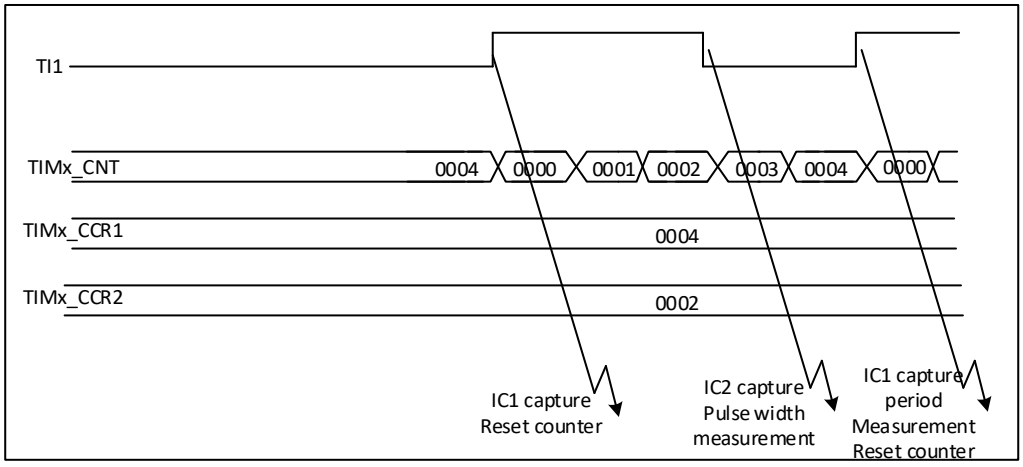


Figure 19-30 PWM input mode timing

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use the TIMx_CH1/TIMx_CH2 signal.

19.2.9 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

Set the corresponding OCxM = 0101 in the TIMx_CCMRx register to force the output comparison signal (OCxREF/OCx) to an active state. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIMx_CCMRx register. Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

19.2.10 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has expired. Channels 1 through 4 can output, but channels 5 and 6 are only used internally in the MCU (e.g. for generating mixed waveforms or for ADC triggering).

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=0000), be set active (OCxM=0001), be set inactive (OCxM=0010) or can toggle (OCxM=0011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- If the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register selects the DMA request function), a DMA request is generated.

You can select whether the TIMx_CCRx register needs to use a preload register by configuring the OCxPE bit in TIMx_CCMRx.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

— Write OCxM = 0011 to toggle OCx output pin when CNT matches CCRx

— Write OCxPE = 0 to disable preload register

— Write CCxP = 0 to select active high polarity

— Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated by software at any time to control the output waveform, provided the preload register is not used (OCxPE = '0', otherwise the shadow register of TIMx_CCRx can only be updated when the next update event occurs). An example is given in the figure below.

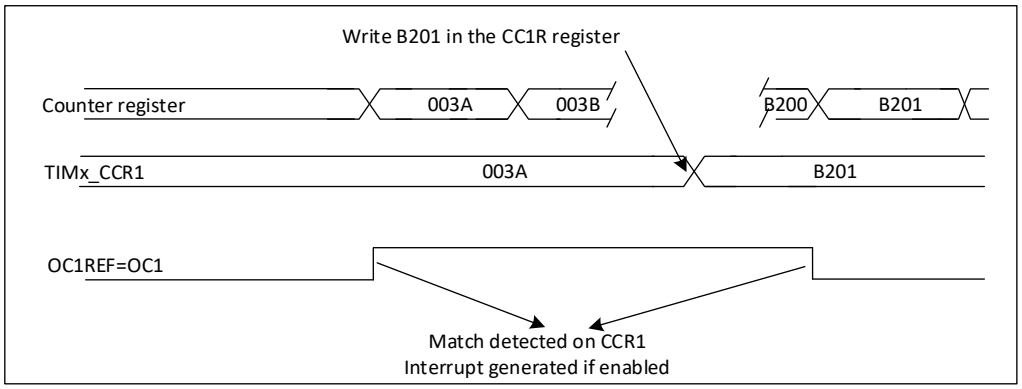


Figure 19-31 Output comparison mode, flip OC1

19.2.11 PWM mode

The pulse width modulation mode may produce a signal whose frequency is determined by the TIMx_ARR register and whose duty cycle is determined by the TIMx_CCRx register.

The OCxM bit in the TIMx_CCMRx register is written to '0110' (PWM mode 1) or '0111' (PWM mode 2), and each OCx output channel can be independently set to generate a PWM. The corresponding preload register must be enabled by setting the OCxPE bit of the TIMx_CCMRx register, and finally the ARPE bit of the TIMx_CR1 register to enable the preload register that is automatically reloaded (in up-count or centrosymmetric mode).

The preload register can only be transferred to the shadow register when an update event occurs, so the user must initialize all registers by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

■ Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the comparison value in TIMx_CCRx is greater than the auto-reload value (TIMx_ARR), OCxREF remains' 1 '. If the comparison value is 0,

then OCxREF remains' 0 '. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

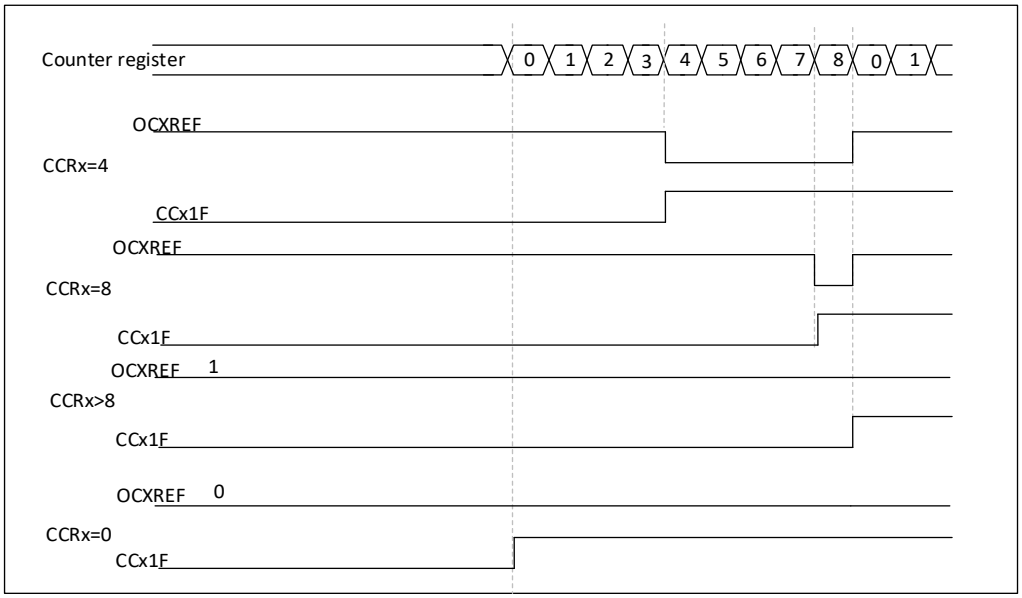


Figure 19-32 Edge-aligned PWM waveforms (ARR=8)

- Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long asTIMx_CNT > TIMx_CCRx else it becomes high. If the comparison value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, OCxREF remains '1'. 0% PWM is not possible in this mode.

PWM Central Alignment Mode

The center alignment mode is when the CMS bit in the TIMx_CR1 register is not '00' (all other configurations of the CMS bit have the same effect on the OCxREF/OCx signal). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software.

The following figure gives some examples of centrally aligned PWM waveforms

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- CMS = 01 for the TIMx_CR1 register, in center alignment mode 1, the compare flag is set when the counter counts down.

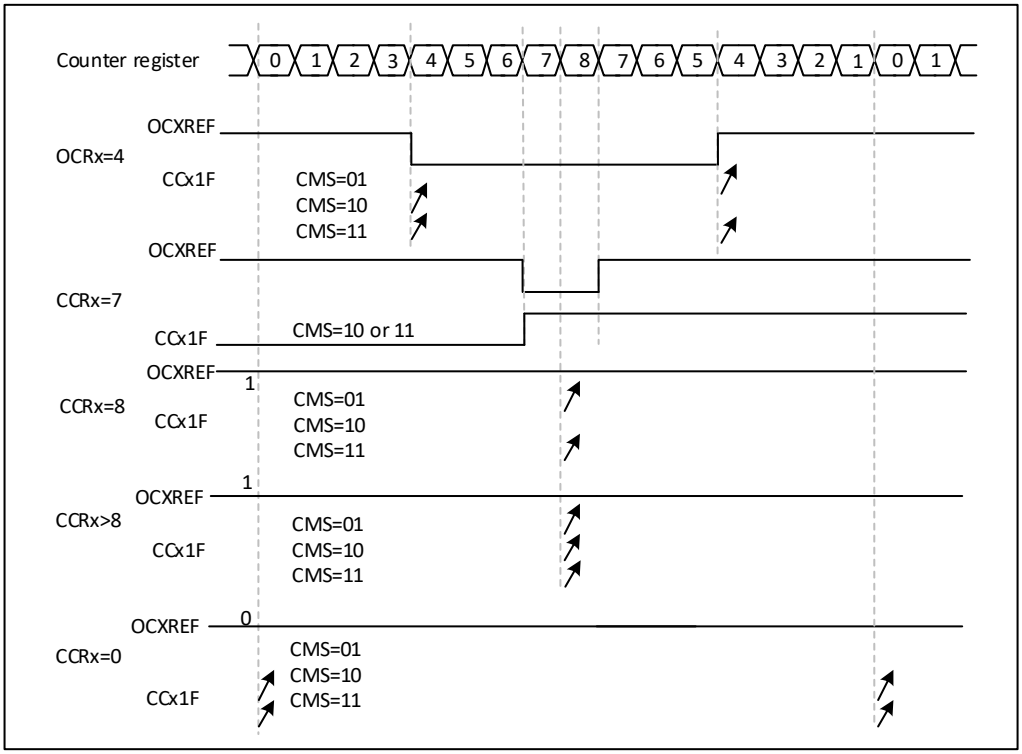


Figure 19-33 Centre-aligned PWM waveform (APR = 8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Also, do not modify the DIR and CMS bits simultaneously through software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:

-If the value of the write counter is greater than the value of the auto-reload (TIMx_CNT > TIMx_ARR), the direction is not updated.

For example, if the counter was counting up, it continues to count up.

-If a value of 0 or TIMx_ARR is written to the counter, the direction is updated, but no update event UEV is generated.

- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

Jitter mode

The DITHEN bit of the TIMx_CR1 register may be enabled to turn on the jitter mode to improve the effective resolution of the PWM mode. Can be applied to CCR (improved duty cycle resolution) and ARR (improved resolution of PWM frequency)

The principle of operation is to make the actual CCR (or ARR) value change slightly (with or without increasing a timer cycle) over 16 consecutive PWM cycles, and how the change can be set by predefined settings. When calculating the average duty cycle, you can achieve a 16-fold increase in resolution. The figure below shows the jitter principle of four consecutive PWM periods

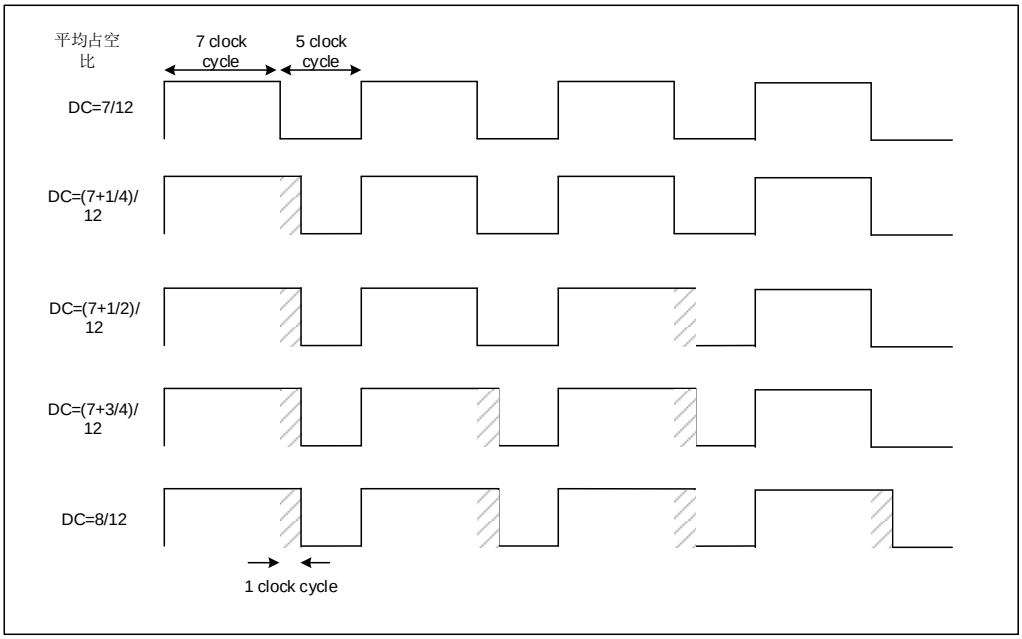


Figure 19-34 jitter principle

When jitter mode is enabled, the value of the register will automatically change as follows:

- The lowest 4 bits is the encoding for improving resolution (fractional part)
- The high bit is shifted left to 19: 4 for encoding as the base value (integer part)

Note: When dither mode is on or off, the values of ARR and CCR are automatically updated (for example, ARR = 0x05 when DITHEN = 0, then ARR = 0x50 when DITHEN = 1), the following steps must be followed when resetting the DITHEN bit

1. CEN and ARPE bits must be reset
2. The ARR [3: 0] bit must be reset
3. The DITHEN bit must be reset
4. The CCIF flag must be cleared
5. CEN bit can be set (ARPE = 1 can be set)

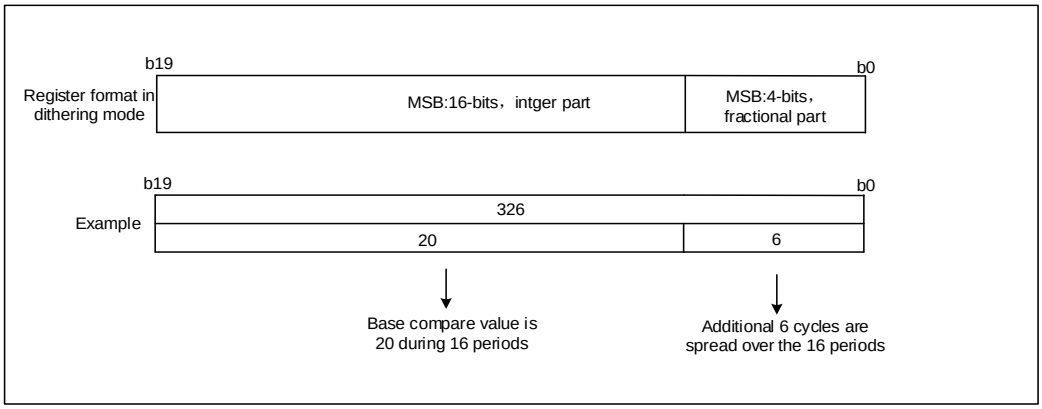


Figure 19-35 Data mapping and register coding in dither mode

The minimum frequency is calculated as follows:

$$\text{According to resolution} = \frac{\text{定时器频率}}{\text{PWM 频率}} \quad \text{push out: minimum PWM frequency} = \frac{\text{定时器频率}}{\text{最大分辨率}}$$

$$\text{In jitter-free mode: Minimum PWM frequency} = \frac{\text{定时器频率}}{65536}$$

When there is jitter mode: minimum PWM frequency = $\frac{\text{定时器频率}}{65535+\frac{15}{16}}$

Note: The maximum values of TIMx_arr and TIMxCCRy in jitter mode are limited to 0xFFFEF (corresponding to 65534 in the integer part and 15 in the jitter part). Exceeding 0xFFFEF, there will be an overflow situation, that is, arr = 0XFFFF+1 = 0.

The figure below shows that in jitter mode, the resolution of the PWM can be improved regardless of the PWM frequency

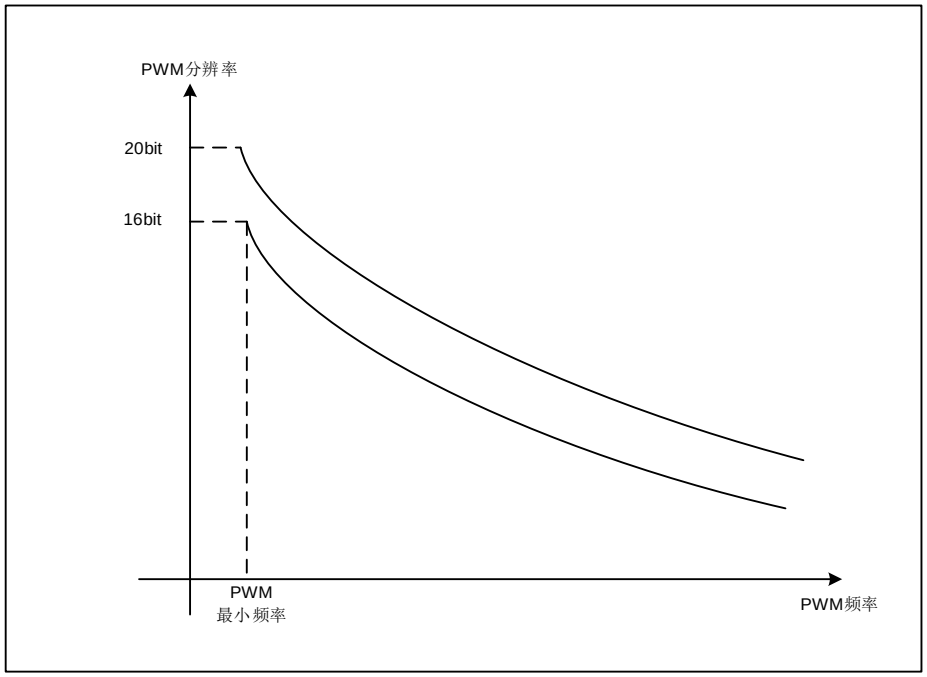


Figure 19-36 PWM Resolution VS Frequency

The duty cycle and cycle changes in 16 consecutive cycles are as follows:

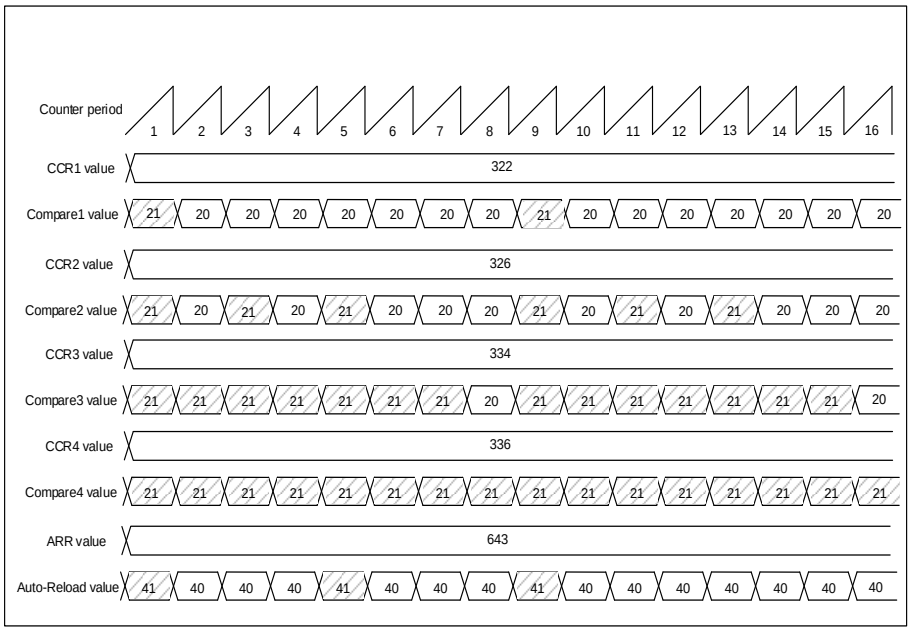


Figure 19-37 PWM Jitter Mode

The auto-reload and compare value increments are distributed in a specific pattern as described in the following table, with the incremental distribution of the jitter sequence as uniformly as possible, minimizing the overall ripple

Table 19-1 CCR and ARR register jitter changes

LSB value	PWM Cycle															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000																
0001	+1															
0010	+1								+1							
0011	+1				+1				+1							
0100	+1				+1				+1				+1			
0101	+1		+1		+1				+1				+1			
0110	+1		+1		+1				+1		+1		+1			
0111	+1		+1		+1		+1		+1		+1		+1			
1000	+1		+1		+1		+1		+1		+1		+1		+1	
1001	+1	+1	+1		+1		+1		+1		+1		+1		+1	
1010	+1	+1	+1		+1		+1		+1	+1	+1		+1		+1	
1011	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1		+1	
1100	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1	+1	+1	
1101	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1		+1	+1	+1	
1110	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1	+1	+1	+1	+1	
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	

The dither mode can also be applied in the intermediate alignment PWM mode (CMS! = 00). The figure below considers the case where the dither mode is applied to 8 consecutive cycles when counting up and down.

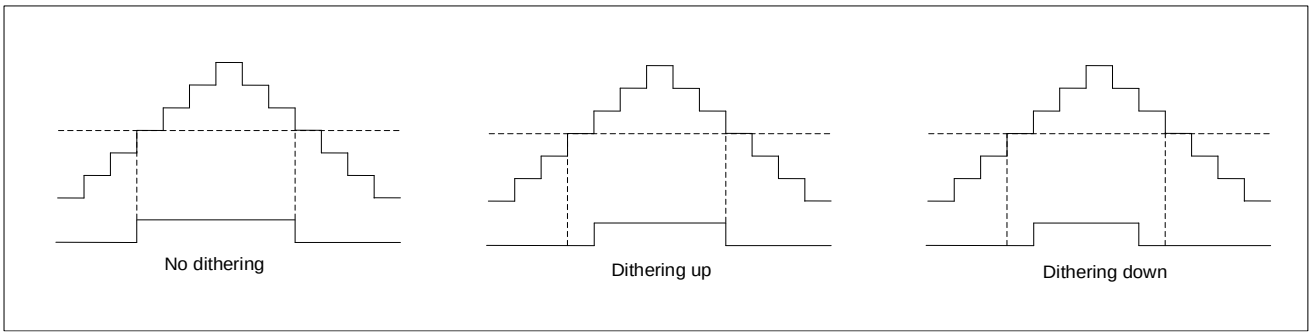


Figure 19-38 Effect of jitter on duty cycle in aligned PWM mode in middle of the figure

Table 19-2 Change of CCR Register Jitter in Middle Aligned PWM Mode

LSB value	PWM Cycle															
	1		2		3		4		5		6		7		8	
	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN
0000																
0001	+1															
0010	+1								+1							
0011	+1				+1				+1							
0100	+1				+1				+1				+1			
0101	+1		+1		+1				+1				+1			
0110	+1		+1		+1				+1		+1		+1			
0111	+1		+1		+1		+1		+1		+1		+1			
1000	+1		+1		+1		+1		+1		+1		+1		+1	
1001	+1	+1	+1		+1		+1		+1		+1		+1		+1	
1010	+1	+1	+1		+1		+1		+1	+1	+1		+1		+1	
1011	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1		+1	
1100	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1	+1	+1	
1101	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1		+1	+1	+1	
1110	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1	+1	+1	+1	+1	
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	

19.2.12 Asymmetric PWM mode

The asymmetric mode allows two centrally aligned PWM signals to produce a programmable phase shift. When the frequency is determined by TIMx_ARR and the duty cycle and phase shift are determined by a pair of TIMx_CCRx registers. The asymmetric mode allows the generation of two center-aligned PWM signals using programmable phase shifts. One TIMx_CCRx register is controlled during the up-count phase and the other during the down-count phase so that the PWM can be adjusted every half PWM cycle

- The OC1REFC(orOC2REFC) is controlled by TIMx_CCR1 and TIMx_CCR2
- The OC3REFC(orOC4REFC) is controlled by TIMx_CCR3 and TIMx_CCR4

By writing 1110 to OCxM (asymmetric PWM mode 1) and 1111 to OCxM (asymmetric PWM mode 2), asymmetric PWM mode can be independently selected on dual channels (each pair of CCR registers controls one OCx output)

Note: For compatibility reasons, the OCxM [3: 0] field is divided into two parts, with the highest and lower 3 bits not contiguous.

When a given channel is used as an asymmetric channel, its adjacent channels (channel 1 adjacent to channel 2) can also be used. For example, if OC1REFC is generated by channel 1 (asymmetric PWM mode 1), channel 2 may also output OC2REF, or OC2REFC through asymmetric PWM mode 1

The figure below represents an example of using asymmetric PWM mode to generate signals (channels 1 through 4 are configured as asymmetric PWM mode 2), which, together with the dead time generator, controls a full-bridge DC-DC phase shift inverter.

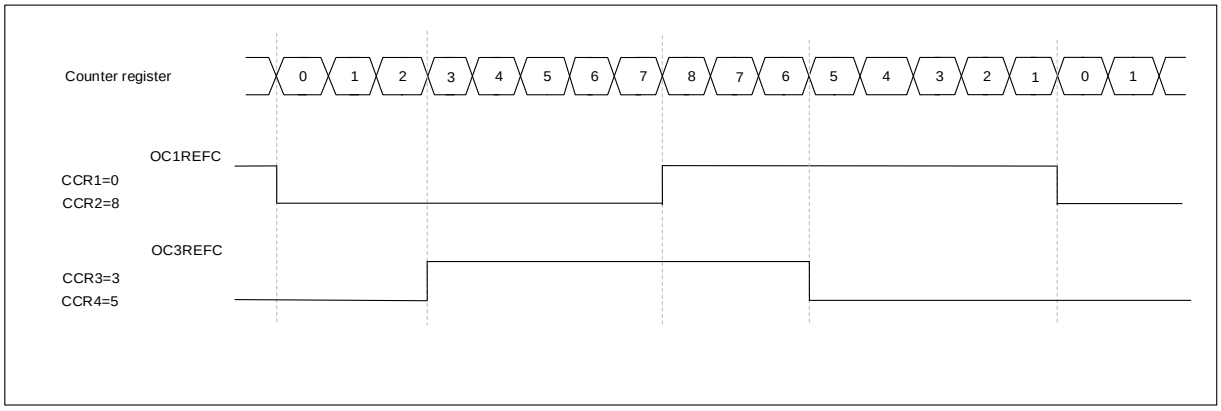


Figure 19-39 2 50% duty cycle phase shift PWM

19.2.13 Combined PWM mode

The combined PWM mode allows an edge or center aligned PWM signal to be generated by programmable delay and phase shift between pulses. The frequency is determined by the ARR and the duty cycle and delay are determined by the two CCRs. The resulting signal OCxREFC is generated by AND/OR logic between two reference PWMs

- The OC1REFC(orOC2REFC) is controlled by TIMx_CCR1 and TIMx_CCR2
- The OC3REFC(orOC4REFC) is controlled by TIMx_CCR3 and TIMx_CCR4

Asymmetric PWM modes can be independently selected on dual channels (each pair of CCR registers controls one OCx output) by writing to OCxM 1100 (combined PWM mode 1, logic OR output) and OCxM 1101 (combined PWM mode 2, logic AND output

When a given channel is used as a combined PWM channel, its complementary channels must be configured in opposite PWM modes (e.g., one for combined PWM mode 1 and one for combined PWM mode 2).

The following figure shows an example of a combined mode generation signal, including

- Channel 1 is combined PWM mode 2
- Channel 2 is PWM Mode 1
- Channel 3 is combined PWM mode 2
- Channel 4 is PWM Mode 1

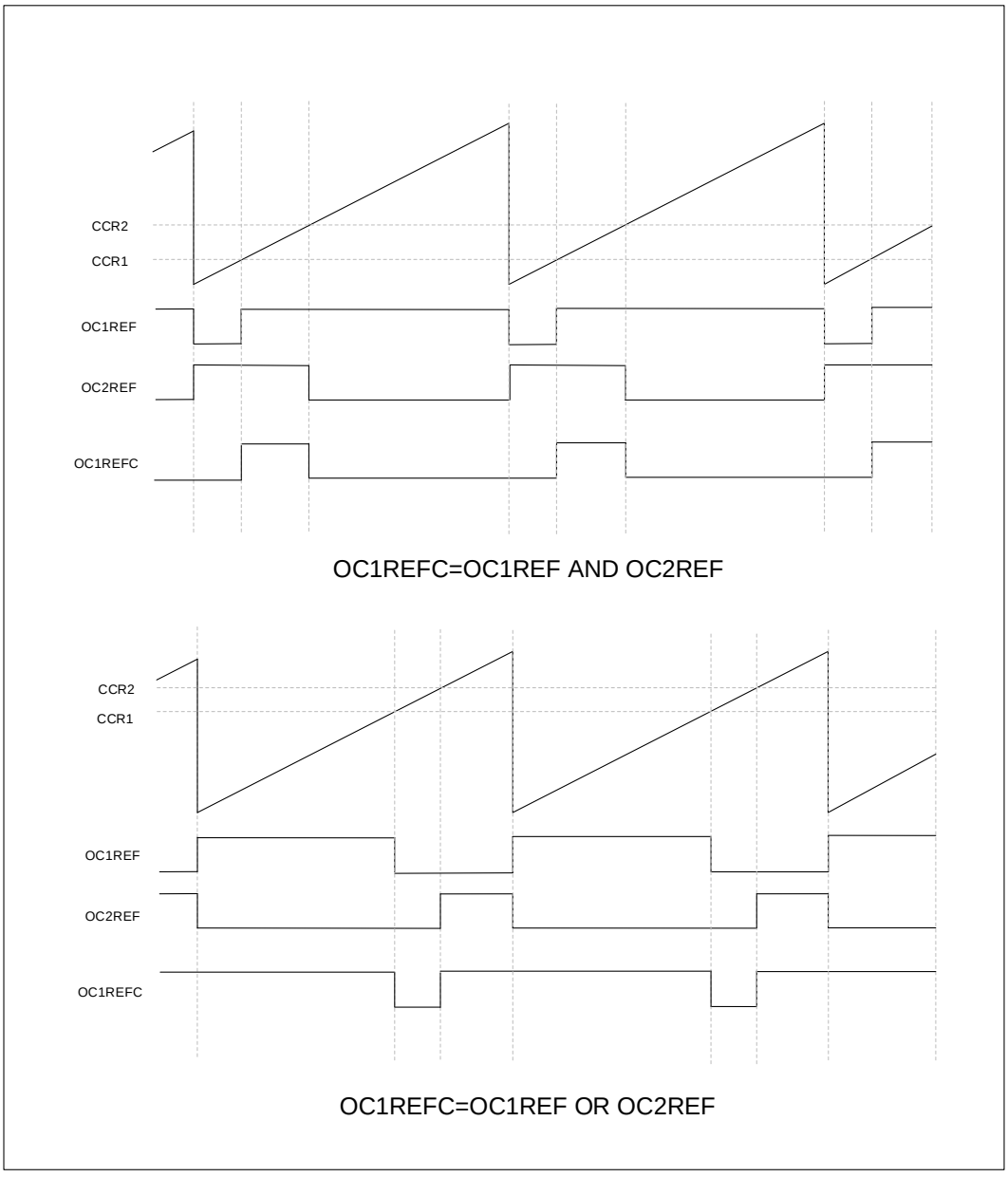


Figure 19-40 Combined PWM patterns for channel 1 and channel 3

19.2.14 Three-phase PWM combination mode

The three-phase combined PWM mode allows three centrally aligned PWM signals to be generated by a single programming signal and logic in the middle of a pulse. The OC5REF is used to define the resulting combined signal. GC5C1/GC5C2/GC5C3of TIM_CCR5 is used to select by which reference

signal the OC5REF is combined. The final output signal OCxREFC is generated by AND logic of two reference PWM signals

- GC5C1 = 1, OC1REFC is controlled by CCR1 and CCR5
- GC5C2 = 1, OC2REFC is controlled by CCR2 and CCR5
- GC5C3 = 1, OC3REFC is controlled by CCR3 and CCR5
- Three-phase combined PWM mode can be independently selected from channels 1 to 3 via GC5C1/GC5C2/GC5C3

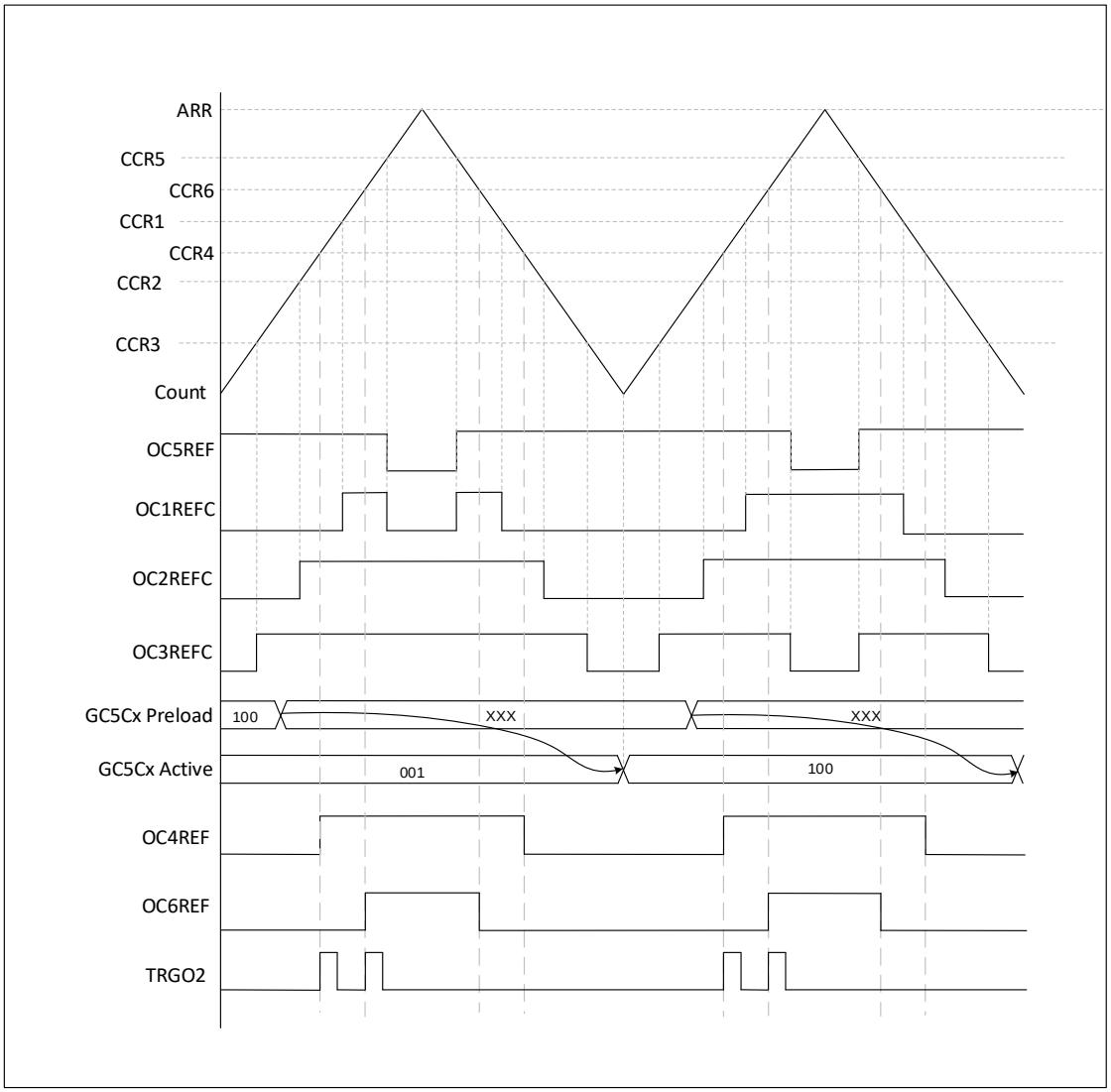


Figure 19-41 Three-phase PWM combination mode

The Trgo2 waveform shows how the ADC is synchronized in a given three-phase PWM signal

19.2.15 Complementary outputs and dead-time insertion

The advanced-control timers (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs.

This time is generally known as dead-time, and you have to adjust it depending on the devices you have connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

You can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: the CCxE and CCxNE bits of the TIMx_CCER register, the MOE, OISx, OISxN, OSSI and OSSR bits of the TIMx_BDTR and TIMx_CR2 registers, as detailed in the control bits of the complementary output channels OCx and OCxN with brake function. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. By configuring the DTG [7: 0] bit in the TIMx_BDTR register, you can control the dead time generator for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP=0, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

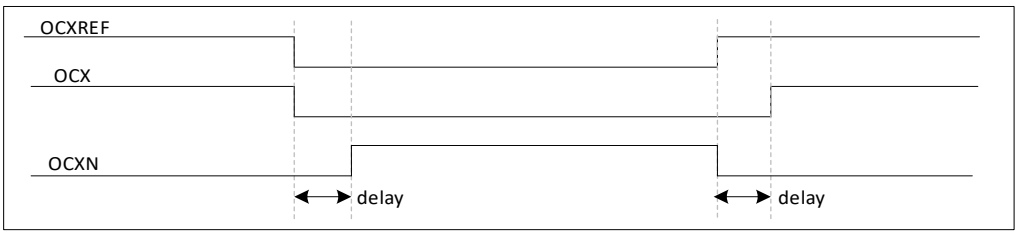


Figure 19-42 Complementary output with dead-time insertion

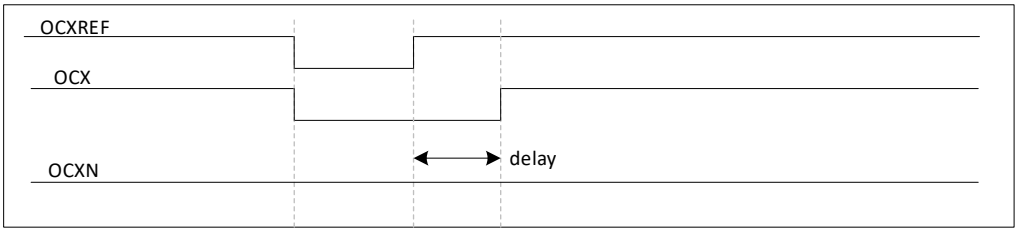


Figure 19-43 Dead-time waveforms with delay greater than the negative pulse

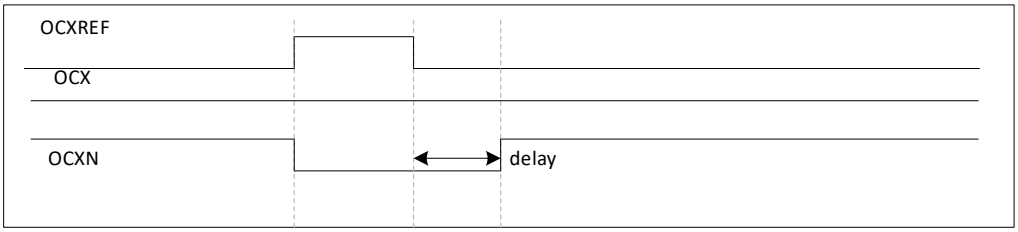


Figure 19-44 Dead-time waveforms with delay greater than the positive pulse.

When DTAE = 1, the rising edge dead time is configured by the DTG [7: 0] bit of TIMx_BDTR, and the falling edge dead time is configured by the DTGF [7: 0] bit of TIMx_DTR2. The writing of the DTAE needs to be before the counter is enabled. When CEN = 1, it is not allowed to modify the DTAE.

The dead time can be modified during the PWM operation through the mechanism of preloading. When the DTPE of TIMX_DTR2 is set, the dead time configuration registers DTG [7: 0] and DTGF [7: 0] are preloaded, and the preloaded data is loaded intothe cache register at the next update event.

Note: If the DTPE bit is set while the counter is enabled, new data written since the last update is discarded and the previous value is used

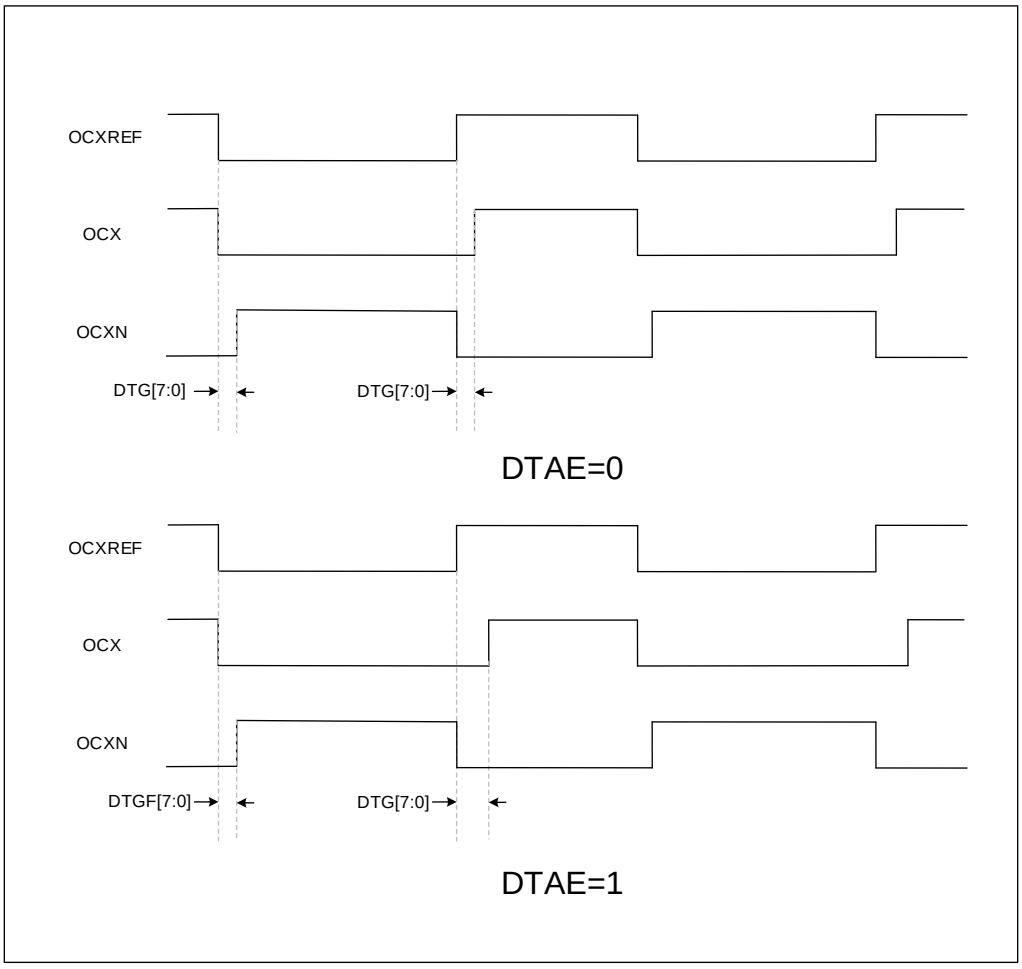


Figure 19-45 asymmetric dead time

19.2.15.1 Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register.

This allows you to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other alternative possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it does not invert and becomes high immediately when OCxREF is active. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

19.2.16 Using the break function

The purpose of the brake function is to protect the power switch driven by the PWM output from the TIMER. The input to the two-way brakes is usually the fault output connected to the three-phase inverter and power stage chip. When the brake signal is active, it will immediately turn off all outputs of

the PWM and force them to a preset safe level state. Some internal events of the MCU can also be used as trigger signals to turn off the output.

There are two channels of braking events. Channel 1 includes system-level faults (clock failure, ECC, parity, etc.) and application-level faults (through input pins or built-in comparators), and can be forced to output to a preset level (whether working or idle) after a dead zone. Channel 2 contains only some application layer faults and forces the output to an invalid state.

The output enable signal and output level during braking are controlled by the following bits

- The MOE bit of TIMx_BDTR can be enabled and turned off by software or 2-way brake event reset
- The OSSI bit of TIMx_BDTR is used to define whether the timer also controls the output when idle, or releases control of the GPIO controller (high impedance state)
- The OISx and OISxN bits of TIMx_CR2 are used to set the output to an invalid level regardless of the working or idle state, andthe OCxandOCxN outputscannot be set to an active level at the same time at the same time, regardless of the values of OISx and OISxN.

When exiting from reset, the brake function is turned off and the MOE is 0. The brake function can be turned on by enabling the BKE and BK2E bits of TIMxBDTR. The polarity of the brake input is configurable via BKP and BK2P. BKEx and BKPx can be modified simultaneously. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1APB clock period to correctly read back the bit after the write operation.

MOE falling edge can be asynchronous, thus a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

A source of the brake channel 1;

- An external source can be connected to one of the TIMx_BKIN pins (select GPIO and configuration registers), plus polarity selection and filtering
- The internal source contains 2 parts
  - Signal from brake comparator (tim_brk_cmpx)
  - From system brake request

A source of the brake channel 2;

- An external source can be connected to one of the TIMx_BKIN2 pins (select GPIO and configuration registers), polarity selection and filtering
- Internal source from brake comparator signal (tim_brk2_cmpx)

The brake event may also be generated by software, and the BG and B2G bits of TIMx_EGR may be set, with all brake sources being OR as brake inputs for tim_brk and tim_brk2. Figure below:

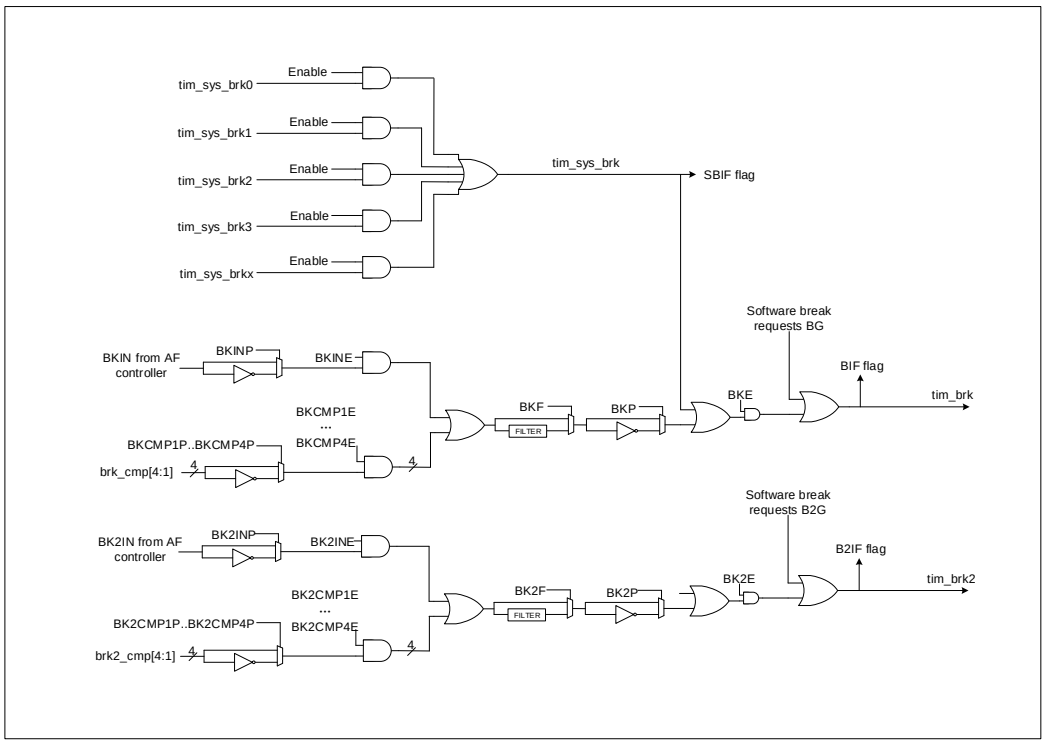


Figure 19-46 Brake and Brake 2 Circuit

Note: Asynchronous operation can only be guaranteed when there is no filtering. If filtering is turned on, a failsafe clock mode must be used to ensure that brake interrupt events are handled (e.g. using internal PLL or CSS, no filtering).

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, placing the output in an invalid state, an idle state, or releasing control of the GPIO controller (selected by the OSSI bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSSI = 0, the timer releases output control, otherwise the enable output is always high.
- When complementary outputs are used:

-The output is first put into a reset state, i.e. an invalid state (depending on polarity). This is done asynchronously so that it works even if no clock is provided to the timer.

— If the clock of the timer still exists, the dead time generator will re-take effect, driving the output port according to the level indicated by the OISx and OISxN bits after the dead time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that the dead time is longer than usual (approximately 2 clock cycles) because of the resynchronization of the MOE.

-if OSSI = 0, the timer releases the output, otherwise keeps the output enabled; Or once one of CCxE and CCxNE becomes high, the enable output becomes high.

- The brake status flag is pulled up (SBIF, BIF, B2IF for TIMx_SR). If the BIE bit in the TIMx_DIER register is set, an interrupt is generated when the brake status flag (BIF bit in the TIMx_SR register) is' 1 '. If the TDE bit in the TIMx_DIER register is set, a DMA request is generated.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next

update event UEV. This can be used to perform a regulation, for instance. Otherwise, the MOE remains low until it is set to '1' again; At this time, this feature can be used in safety. You can connect the brake input to the power-driven alarm output, thermal sensor or other safety devices.

Note: When AOE is 1, if MOE is reset by the CPU, the output will enter the idle state and be forced to an invalid level or high impedance (depending on the value of OSSI). If both MOE and AOE are reset by the CPU, the output will enter the invalid state and the output level is driven by the OISx bit.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). At the same time, the status flags BIF and B2IF cannot be cleared.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows the user to freeze several configuration parameters (dead time duration, OCx/OCxN polarity and disabled state, OCxM configuration, brake enable and polarity). The user can select one of the three levels of protection by setting the LOCK bit in the TIMx_BDTR register, see Brake and Dead Time Register (TIMx_BDTR). The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

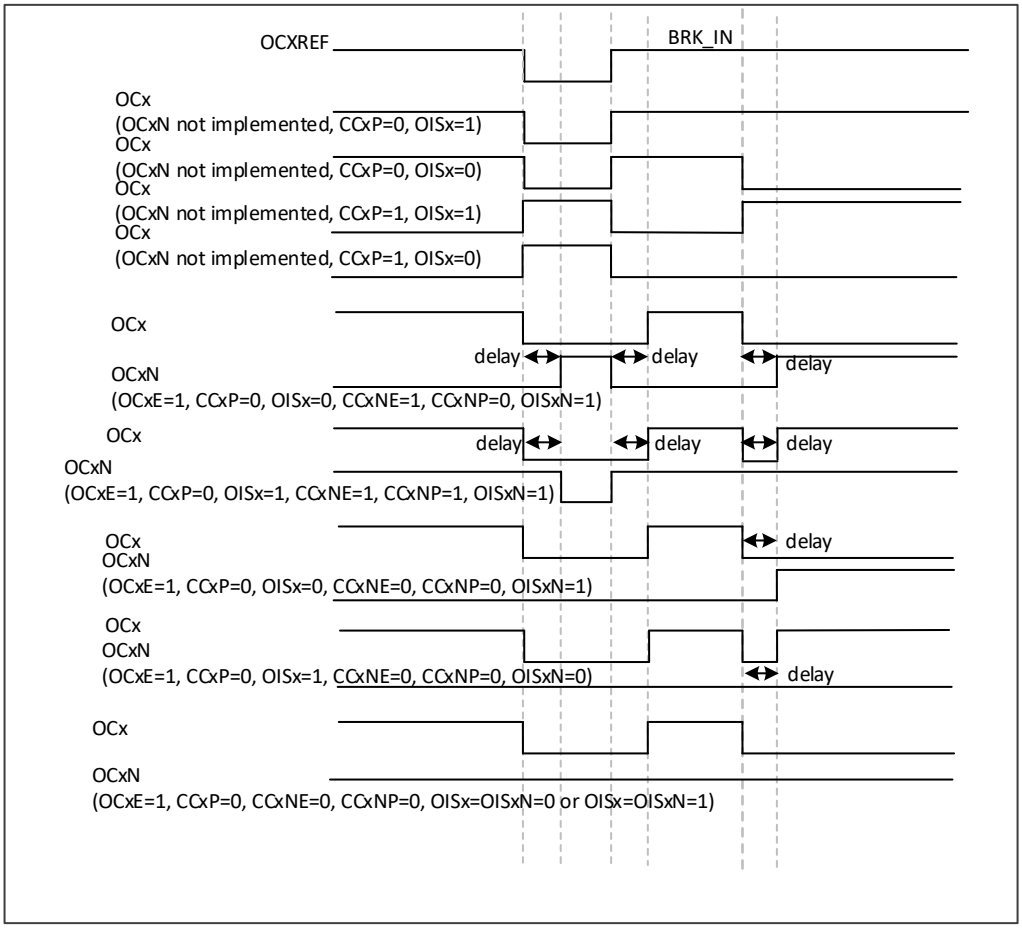


Figure 19-47 Output behavior in response to a break

The two brake input signals have different behaviors at the output of the timer

Tim_brk can either invalidate the output or force it to a preconfigured secure state

Tim_brk2 can only make the output invalid.

The priority of tim_brk is higher than that of tim_brk2, which is described by the following figure:

Note: tim_brk2 is only used when OSSR = OSSI = 1 (output is not high impedance)

Table 19-3 Output Behavior at Brake and Brake 2 Input

BRK	BRK2	TIM output status	Typical situation	
			OCxN output (low side switch)	OCx output (high side switch)
Active	X	-Output is invalid and force output state (after dead zone) -Output forbidden if OSSI = 0	On after dead zone insertion	OFF
Inactive	Active	Not valid	OFF	OFF

The following figure depicts the output behavior of OCx and OCxN when tim_brk and tim_brk2 are valid. All polarities are highly effective (CCxP = CCxNP = 0 of the TIMx_CCER register)

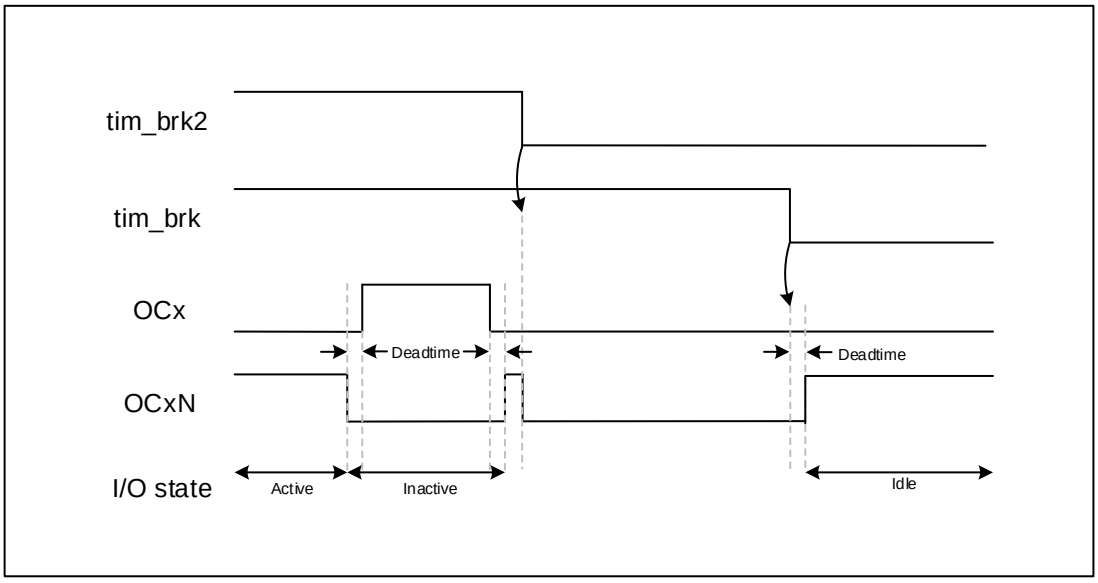


Figure 19-48 PWM output state after braking and braking 2 (OSSI = 1)

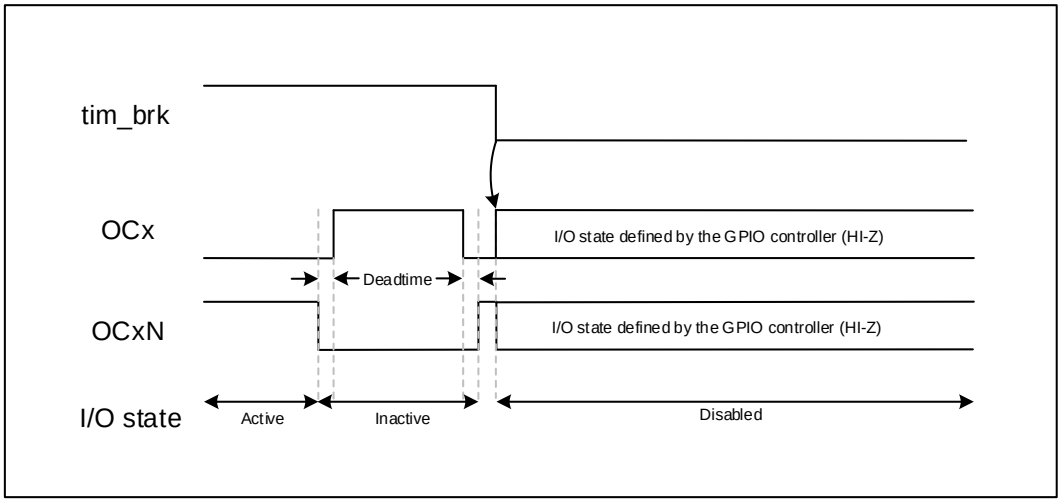


Figure 19-49 PWM output state after braking and braking 2 (OSSI = 0)

19.2.17 Bidirectional brake input

TIM1 has two-way brake I/O, as shown in the figure below:

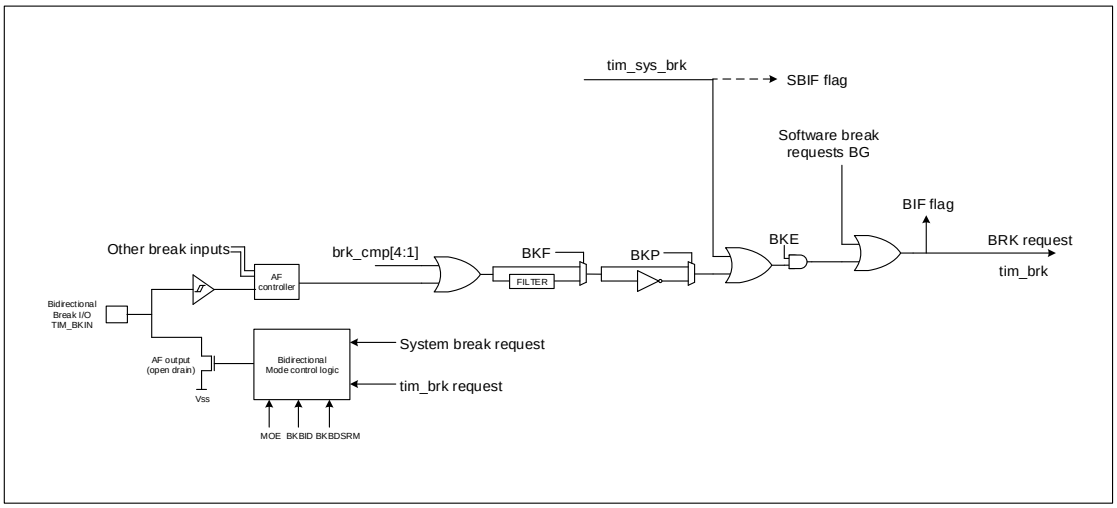


Figure 19-50 Output Redirection (when Brake 2 is not requested)

I/O support features:

- Provides a board-level global brake signal that sends a fault signal to the external MCU and gate driver through a unique pin that can be used as both input and output
- When it is necessary to combine a plurality of internal and external brake sources, the internal brake source and the plurality of external open-drain sources are OR calculated as the unique trigger signal of the braking event

The inputs of tim_brk and tim_brk2 are controlled to be bidirectional by the BKBID and BK2BID bits of TIMxBDTR. BKBID and BK2BID can be set to 1 using the LOCK bit (LOCK level 1) of TIMxBDTR, and the implementation is locked into read-only mode.

Both tim_brk and tim_brk2 inputs can usebidirectional modeand require I/O configuration in open drain mode with low active polarity (via BKINP, BKP, BK2INP, BK2P control bits). Whether from the system (e.g. CCS), from the board-level peripheral, or the brake input forcing a low level indicates a brake input failure event, it can be used as a brake request. However, for safety reasons, when the polarity bit is not effectively configured (when configured to be active high), the bidirectional mode is disabled. Software braking events (BG, B2G) will also cause the brake I/O to be pulled down, which is used to show external devices that this timer has entered the braking state. Of course, it is only possible when BKE or B2KE is 1. When BKE or B2KE is 0, a software brake event occurs, the output is set to a safe state and a brake flag signal is generated, but it has no effect on TIMx_BKIN and TIMx_BKIN2 I/O

A safe release mechanism prevents the system from being restrictively locked (a low level at the brake input triggers a brake, which in turn forces the input to low level)

When the BKDSRM (BK2DSRM) bit is set to 1, the brake output control can be released, the fault flag signal is cleared and the possibility of reconfiguring the system is provided.

Under no circumstances should the brake protection circuit be turned off:

- The brake input path is always active: even if the BKDSRM (BK2DSRM) bit is set to 1 and the open-drain control is released, the brake event is still active. This prevents that PWM output from being restarted while still in the brake state
- When the output is enabled (MOE = 1), the BKDSRM (BK2DSRM) bit cannot release brake protection

Table 19-4 Brake protection status

MOE	BKBID（BK2BID)	BKDSRM（BK2DSRM)	Brake protection status
0	0	X	Enable
0	1	0	Enable
0	1	1	1 1 Disarmed
1	X	X	Enable

19.2.17.1 Enable and re-enable the brake circuit

By default, the brake circuit (in input or bi-directional mode) is enabled

Follow the following procedure to restart the protection after braking (or brake 2):

- The BKDSRM (BK2DSRM) bit must be set to 1 to release output control
- The software must wait until the system brake status disappears and then clear the SBIF status flag (or systematically clear it before re-enabling)
- The software must poll the BKDSRM (BK2DSRM) bit until it is cleared by the hardware (when the application brake state disappears)

In this way, the brake circuit is activated and the MOE bit can be re-enabled for the PWM output

19.2.18 Clearing the OCxREF signal on an external event

For a given channel, setting the corresponding OCxCE bit in the TIMx_CCMRx register to '1' enables the OCxREF signal to be pulled low with a high level at the OCREF_CLR_INT input, and the OCxREF signal will remain low until the update event UEV generated by the next count overflow occurs. This function can only be used in output compare and PWM modes. It does not work in Forced mode.

While OCREF_CLR_INT can be selected between OCREF_CLR and ETRF (after ETR filtering) by configuring the OCCS bit in the TIMx_SMCR register.

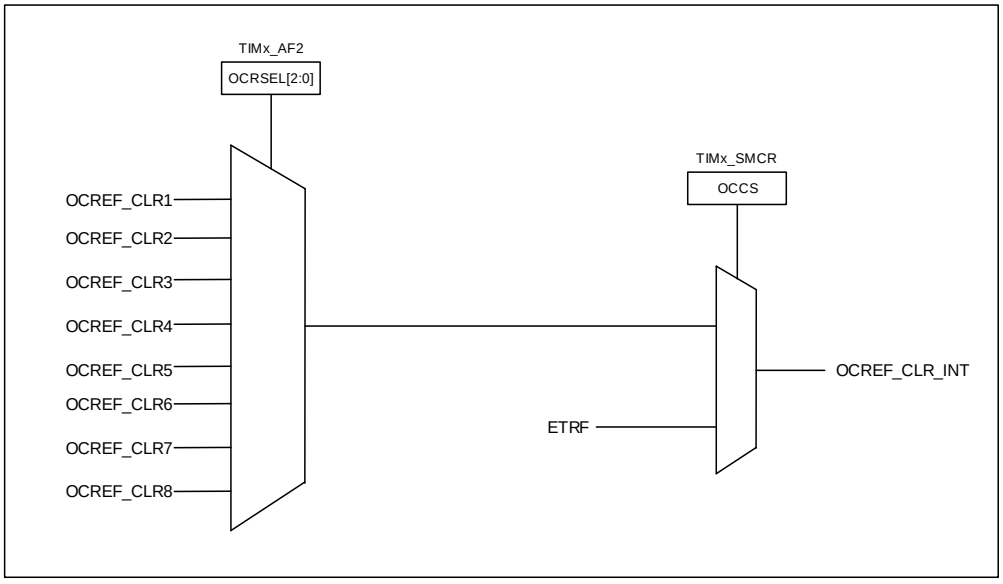


Figure 19-51 OCREF_CLR input selection

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, ETR must be configured as follows:

1. The external trigger prescaler should be kept off: bits ETPS[1:0] in the TIMx_SMCR register are cleared to 00.

2. The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.

3. The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured as needed.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is placed in PWM1 mode.

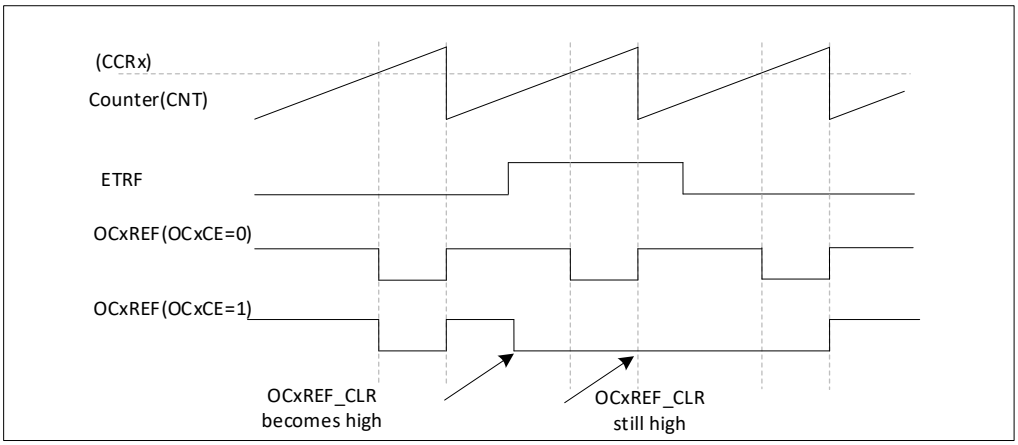


Figure 19 -52OCxREF for clearing TIMx

19.2.19 Generate a six-step PWM output

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. These preload bits are transferred to the shadow register when a COM commutation event occurs. In this way, you can pre-set the configuration of the next step and change the configuration of all channels at the same time. COM can be generated by software by setting the COMG bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

When a COM event occurs, a flag bit (COMIF bit in the TIMx_SR register) will be set. At this time, if the COMIE bit of the TIMx_DIER register has been set, an interrupt will be generated; If the COMDE bit of the TIMx_DIER register has been set, a DMA request is generated.

The figure below describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

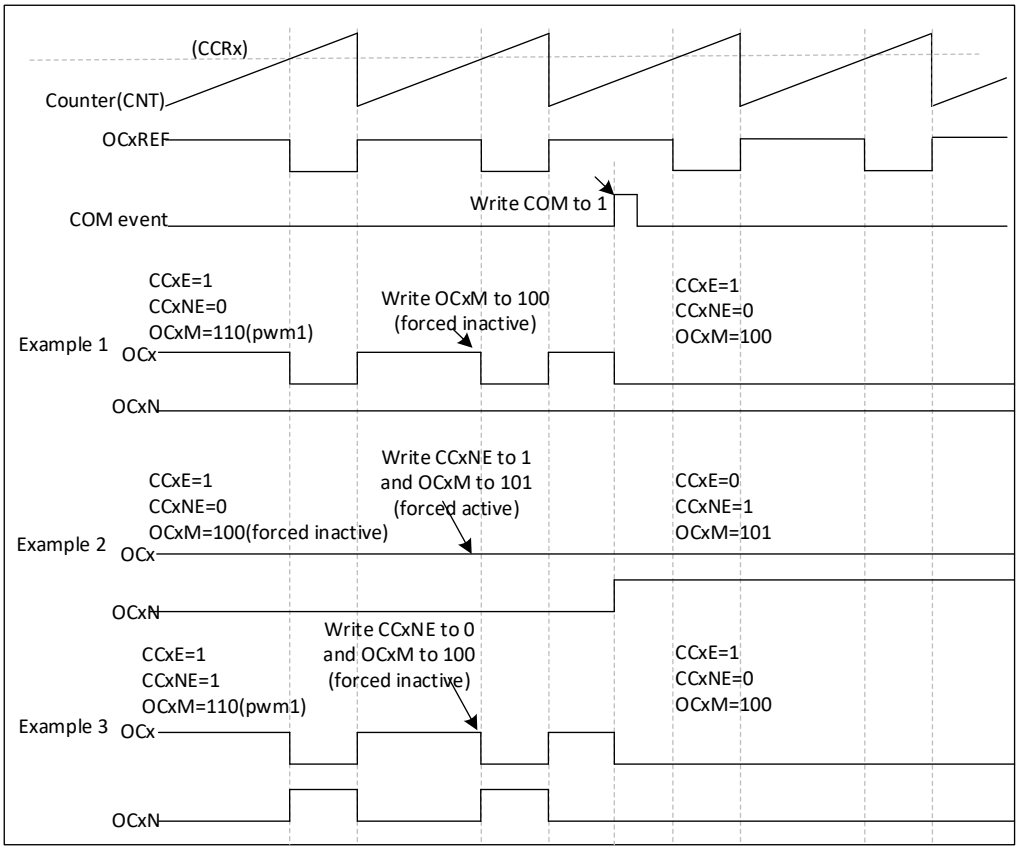


Figure 19-53 generates a six-step PWM, using COM example (OSSR = 1)

19.2.20 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV. A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up count mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx),
- Down counting mode: Counter CNT > CCRx.

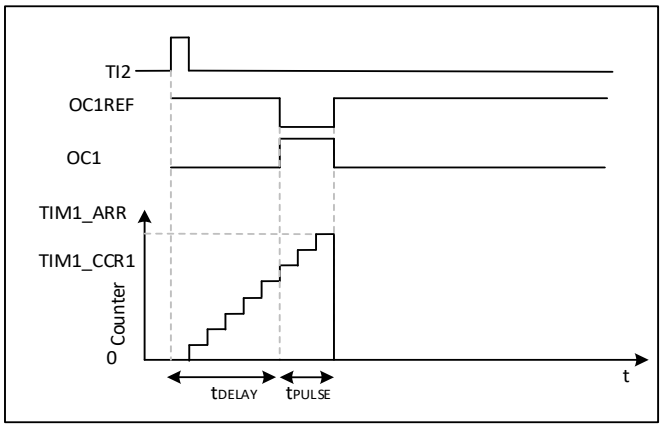


Figure 19-54 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Assume TI2FP2 as the trigger:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let's say one want to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '0' to '1' when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0) When OPM bit in the TIMx_CR1 register is set to '0', so the repetitive mode is selected.

19.2.20.1 Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay tDELAY min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF (and OCx) is forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

19.2.21 Retriggerable Single Pulse Mode

This mode allows the counter to start in response to excitation and generate pulses of programmable length, which differs from the non-retriggerable single pulse mode described in the previous section as follows:

- The pulse starts as soon as the trigger occurs (no programmable delay).
- If a new trigger occurs before the pulse generated by the previous trigger is completed, the pulse is extended.

To use the retriggerable monopulse mode, the timer must be in slave mode, the bit SMS [3: 0] = "1000" in the TIMx_SMCR register (combined mode-reset + trigger), and the OCxM [3: 0] bit set to "1000" or "1001" (retriggerable OPM mode 1 or 2).

If the timer is configured in Up-counting mode, the corresponding CCRx must be set to 0 (the ARR register sets the pulse length). If the timer is configured in Down-counting mode, CCRx must be above or equal to ARR.

Note: For compatibility reasons, the OCxM [3: 0] and SMS [3: 0] bit fields are split into two parts, with the most significant bit not contiguous with the 3 least significant bit positions.

This mode must not be used in conjunction with the central alignment count mode. CMS [1: 0] = 00 must be in TIMx_CR1.

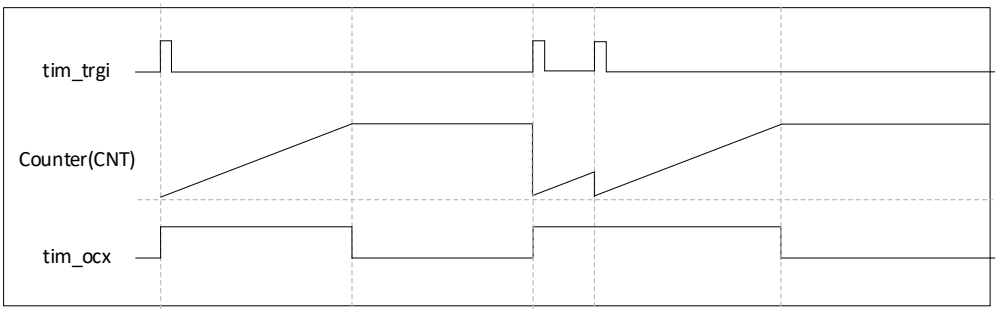


Figure 19-55 Example of a retriggerable monopulse mode

19.2.22 Comparison mode generation pulse

Pulses may be generated by comparing matching events. When the counter value is equal to the given comparison value, a programmable width pulse signal is generated for debugging and synchronization purposes.

This mode can be used with any slave mode, including encoder mode. It is only available for 3/4 channels in edge and center alignment counting mode. The pulse generator is unique and shared by these two channels, as shown in the following figure:

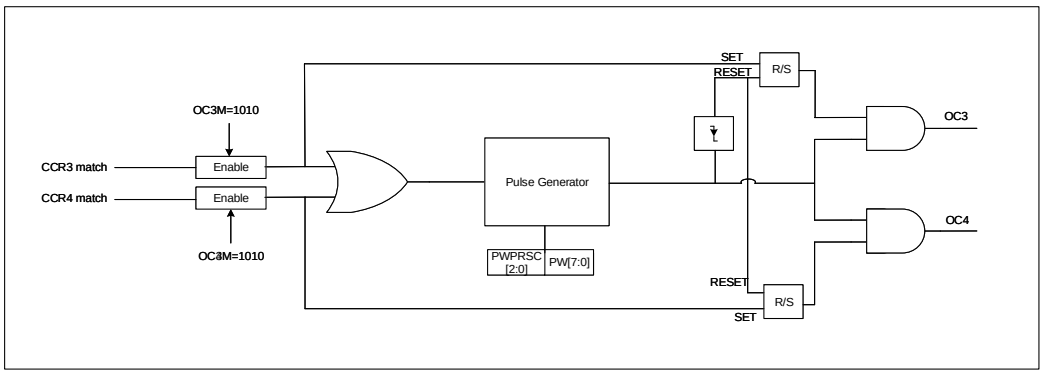


Figure 19-56 Pulse generator circuit

The following figure shows how pulses are generated in edge counting and encoder modes:

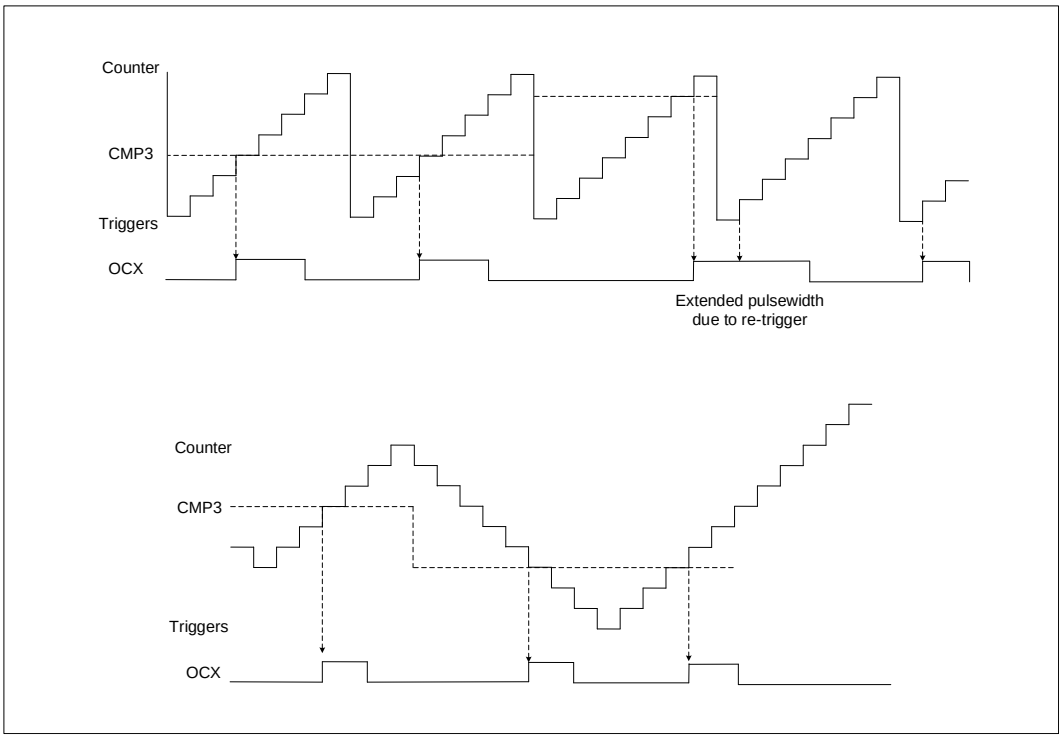


Figure 19-57 Generate pulses by comparing events in edge alignment and center alignment modes

The output comparison mode is selected by OC3M [3: 0] and OC4M [3: 0] in TIMx_CCMR2

The pulse width is programmed via the PW [7: 0] register, using the clock generated by clock prescalation via PWPRSC [2: 0]

t_{PW}= PW [7: 0] xt_{PWG}

t_{PWG}= xCK $2^{(PWPRSC[2:\ 0])}$_INT

The resolution and maximum value depend on the value of the prescaler

Pulse re-trigger: When the pulse is output normally, a new trigger comes, which will cause the pulse to be extended

Note: If both channels are enabled at the same time, the pulses output are independent as long as the trigger on one channel does not overlap with the pulses generated by the other channel. If the two triggers overlap, the trigger 1 generation pulse will be extended (will be re-triggered), and the last trigger generation pulse width will be correct.

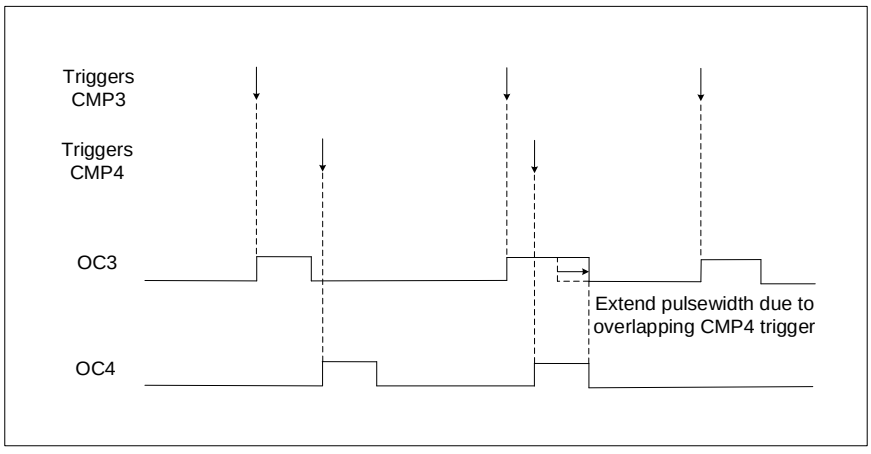


Figure 19-58 Generate pulses by comparing events in edge alignment and center alignment modes

19.2.23 Encoder interface mode

19.2.23.1 Quadrature encoder

To select Encoder Interface mode: write SMS='0001' in the TIMx_SMCR register if the counter is counting on TI1 edges only, SMS=0010 if it is counting on TI2 edges only and SMS=0011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, you can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Referring to the table below, assuming that the counter has been started (CEN = 1 in the TIMx_CR1 register), the counter is driven by each valid jump on TI1FP1 or TI2FP2. TI1FP1=TI1 if not filtered and not inverted,TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit inTIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIMx_ARR must be configured before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together. In this mode, the counter is modified automatically following the speed and the direction of the quadrature encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 19-5 Relationship between counting direction and encoder signal (CC1P = CC2P = 0)

Effective edge	SMS [3: 0]	Reverse signal level (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
			Rising	Falling	Rising	Falling
Count only at TI1 single edge x1 mode	1110	High	Minus	Add	-	-
		Low	-	-	-	-
Count only at TI2 single edge x1 mode	1111	High	-	-	Add	Minus
		Low	-	-	-	-
Count only at TI1 double edge x2 mode	0001	High	Minus	Add	-	-
		Low	Add	Minus	-	-
Count only at TI2 double edge x2 mode	0010	High	-	-	Add	Minus
		Low	-	-	Minus	Add
Counting at TI1 and TI2 double edges x4 mode	0011	High	Minus	Add	Add	Minus
		Low	Add	Minus	Minus	Add

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally used to convert the encoder's differential outputs to digital signals.

This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S = '01' (TIMx_CCMR1 register, TI1FP1 mapped to IC1)
- CC2S = '01' (TIMx_CCMR2 register, TI2FP2 mapped to IC2)
- CC1P = '0' (TIMx_CCER register, TI1FP1 is not inverted, TI1FP1 = TI1)
- CC2P = '0' (TIMx_CCER register, TI2FP2 is not inverted, TI2FP2 = TI2)
- SMS = '011' (TIMx_SMCR register, all inputs are valid on rising and falling edges).
- CEN = '1' (TIMx_CR1 register, counter enabled)

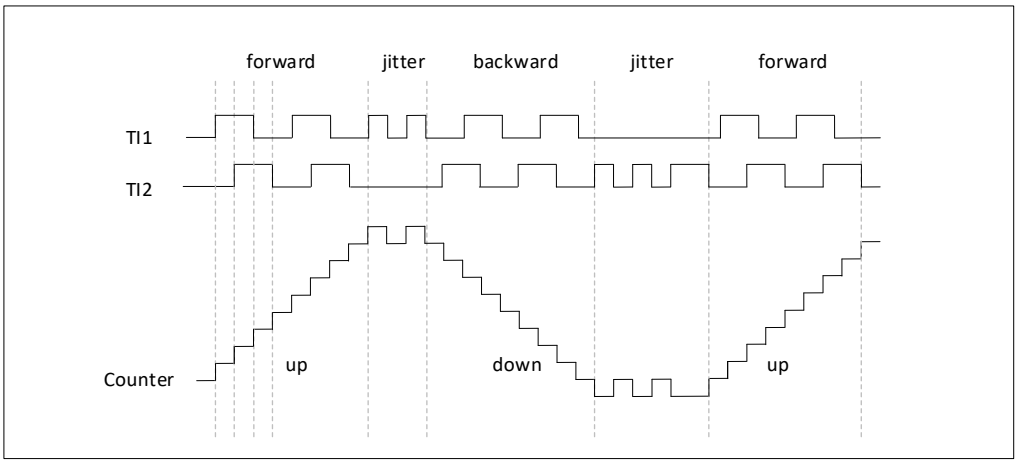


Figure 19-59 Example of counter operation in encoder interface mode

The following figure shows an operation example of the counter when the polarity of IC1FP1 is inverted (CC1P = '1', other configurations are the same as the above example)

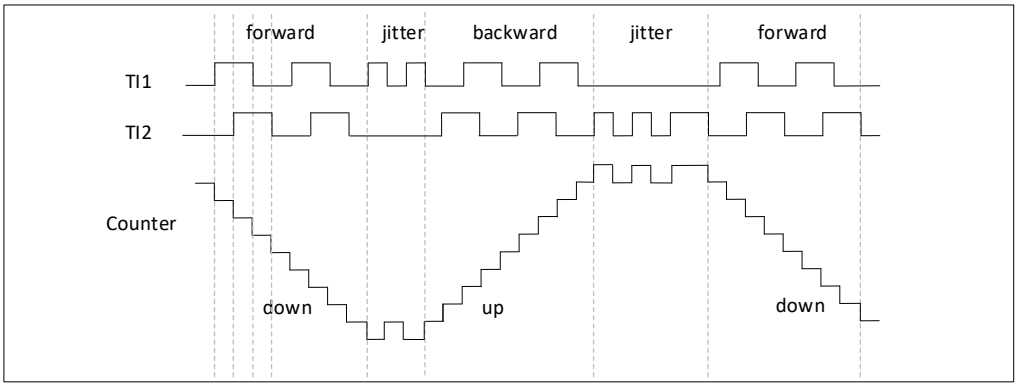


Figure 19-60 IC1FP1 Inverted Encoder Interface Mode Example

The figure below shows the count value when the steering turns over in different modes:

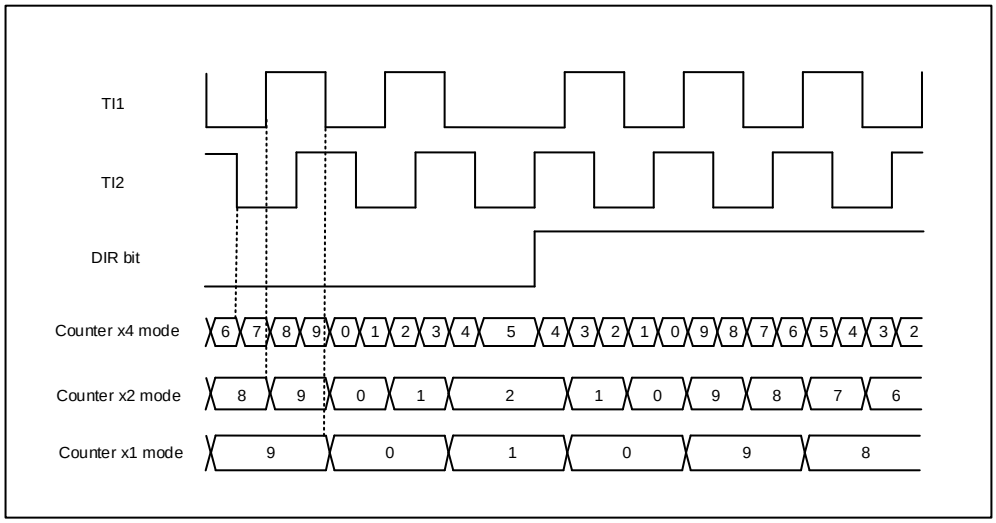


Figure 19-61 Quadrature Encoder Counting Mode

When the timer is configured in the encoder interface mode, information of the current position of the sensor may be provided. By configuring the second timer in the capture mode, the interval between the two encoder events can be measured, and dynamic information (speed, acceleration, deceleration) can be obtained. An encoder output indicating a mechanical zero may be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. If possible, you can latch the value of the counter to the third input capture register (the capture signal must be periodic and can be generated by another timer); Its value can also be read by a DMA request generated by a real-time clock.

The UIFREMAP bit of the TIMxCR1 register forces the continuous copy of the update interrupt flag (UIF) to the highest bit of the timer counter (TIMxCNT [31]). This allows the count value and a state generated by the UIFCPY flag to be readable. This simplifies the calculation of angular velocity by avoiding causing race conditions. For example, by sharing processing between background tasks (counter readout) and interrupts (interrupt updates

There is no delay between UIF and UIFCPY flag assertions

In a 32-bit counter design, when the UIFREMAP bit is set, the 31st bit of the counter is overwritten by the UIFCPY flag when reading (the most significant bit of the counter is only accessible in write mode)

19.2.23.2 Clock plus directional encoder mode

In addition to quadrature encoders, the timer also supports other different types of encoder modes

Clock plus direction mode As shown in the figure, the clock is provided by TI2 and the direction is input by TI1, this mode is enabled by configuring the SMS [3: 0] of the TIMx_SMCR register as follows:

- 1010: x2 mode, counter updates on rising and falling edges of clock
- 1011: x1 mode, counter is updated at one edge, rising edge is updated at CC2P = 0, falling edge is updated at CC2P = 1

The polarity of the directional signal input to TI1 is determined by the CC1P bit, CC1P = 0: positive polarity (count up when TI1 is high, count down when TI1 is low); CC1P = 1: negative polarity (count down when TI1 is high, count up when TI1 is low)

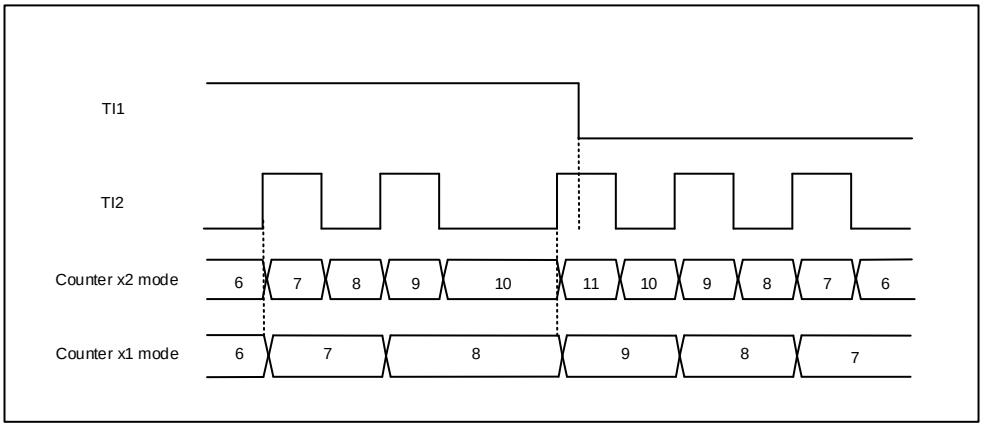


Figure 19-62 direction plus clock encoder mode

19.2.23.3 Directional clock encoder mode

In directional clock mode (shown in the figure below), the clock is provided by two signals, one at a time depending on the direction, thus having an up-count clock signal (TI2) and a down-count clock signal (TI1). This mode is enabled by configuring the SMS [3: 0] of the TIMx_SMCR register as follows:

- 1100: x2 mode, regardless of which clock line, the counter is updated on both the rising and falling edges of the clock. The clock initial state may be configured by the CC1P and CC2P bits. CCxP = 0 represents a high level initial state and CCxP = 1 represents a low level initial state
- 1101: x1 mode, the counter is updated at one edge, depending on the values of CCx1P and CCx2P. CCxP = 0 indicates a falling edge count and the initial state is high. CCxP = 1 indicates a rising edge count and the initial state is low.

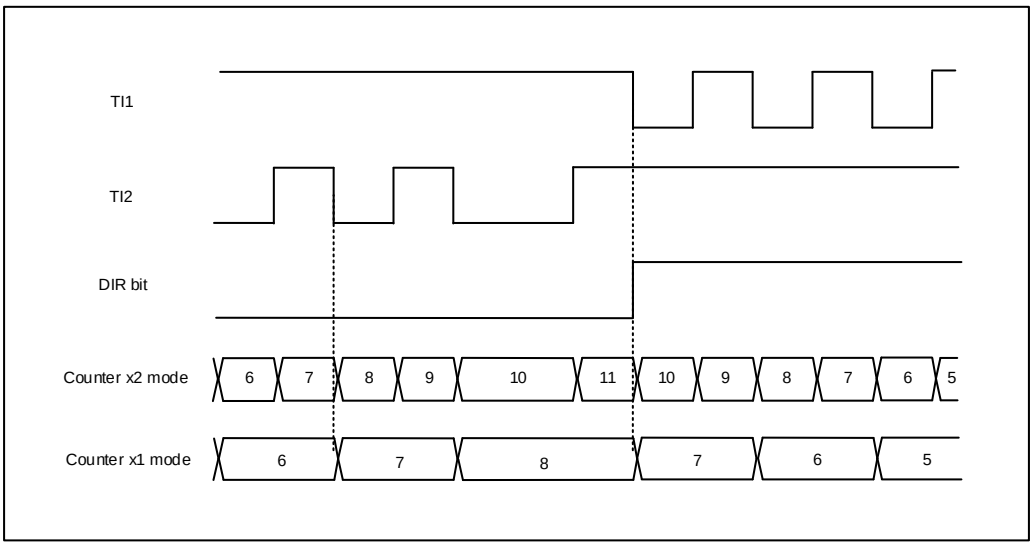


Figure 19-63 Directional Clock Encoder Mode (CC1P = CC2P = 0)

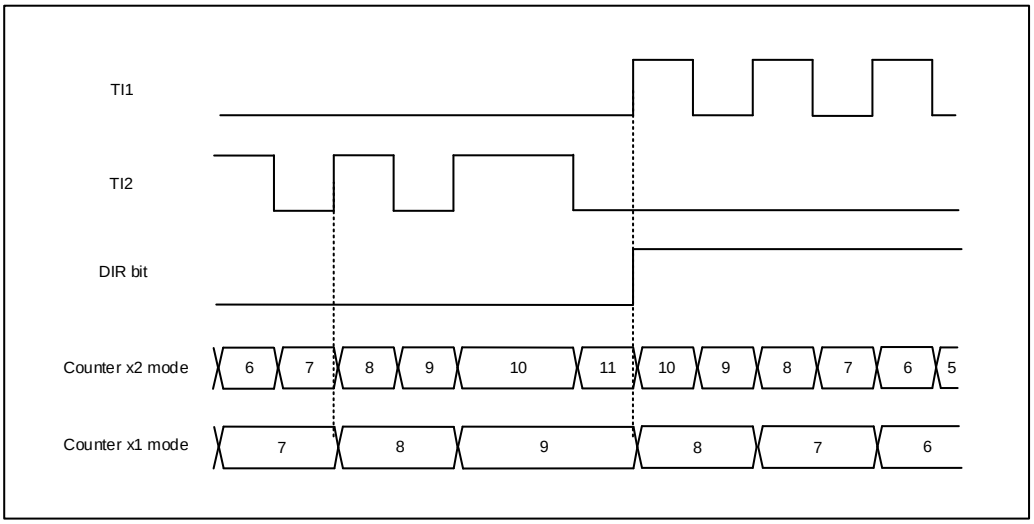


Figure 19-64 Directional Clock Encoder Mode (CC1P = CC2P = 1)

The following table details how the directional clock mode operates for any input transition:

Table 19-6 Counting direction versus encoder signals

Effective edge	SMS [3: 0]	Reverse signal level (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
			Rising	Falling	Rising	Falling
x2 mode CCxP = 0	1100	High	Minus	Minus	Add	Add
		Low	-	-	-	-
x2 mode CCxP = 1	1100	High	-	-	-	-
		Low	Minus	Minus	Add	Add
x1 mode CCxP = 0	1101	High	-	Minus	-	Add
		Low	-	-	-	-
x1 mode CCxP = 1	1101	High	-	-	-	-
		Low	Minus	-	Add	-

19.2.23.4 Index input

The counter may be reset by an index signal output by the encoder for indicating an absolute reference position. The index signal must be connected to the tim_etr input. It may be filtered by a digital filter. The IE bit of the TIMx_ECR register enables the index function, and the IE bit can only be enabled in encoder mode, that is, when the SMS [3: 0] is configured to the following values: 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111

As shown in the figure below, commercial encoders have several options for index pulse adjustment

- Gating of A and B: The pulse width is 1/4 of a channel, aligned with the edges of A and B
- Gating of A (gating of B): The pulse width is 1/2 of a channel, aligned with the edge of A
- Ungated: Pulse width up to one channel period, not aligned with channel edge

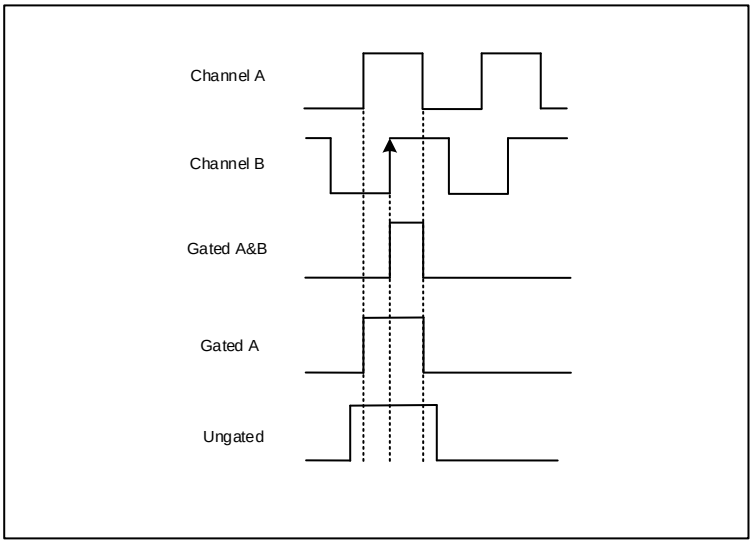


Figure 19-65 Index gating selection

The figure below shows that the circuit can tolerate the jitter of the index signal in any gating mode. In ungated mode, the index signal pulse width needs to be strictly less than 2 encoder cycles. If greater than or equal to 2 cycles, the counter will be reset multiple times.

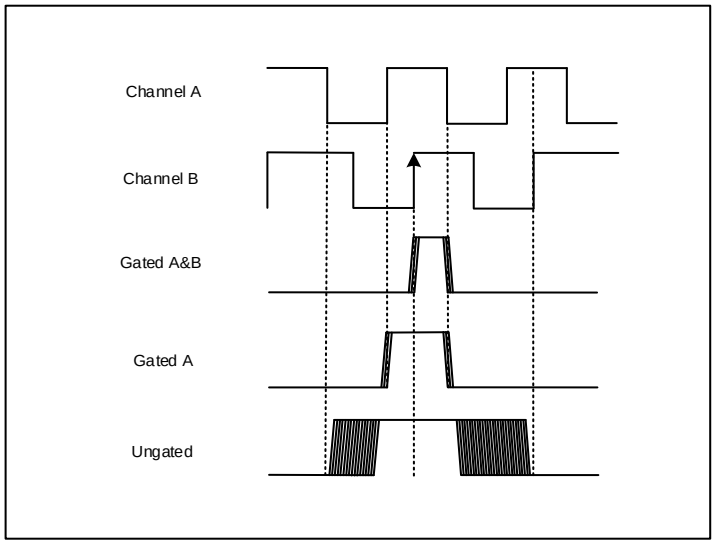


Figure 19-66 Index signals for jitter

The timer fully supports 3 gating modes and does not require special programming. It only needs to define what state of the encoder (the combination of channel A and channel B states) the index signal needs to be synchronized. By configuring the IPOS [1: 0] of the TIMx_ECR register.

The detection event of the index signal varies depending on the counting direction to ensure symmetrical operation during the rotation direction flip.

- The counter is reset during up count (DIR = 0)
- When counting down, the counter value is set to TIMx_ARR

This allows indexes to be generated at the same mechanical angular position regardless of the counting direction. The figure below shows which position indexes are generated, for a simple example (one encoder provides 4 mechanical rotations along each standard)

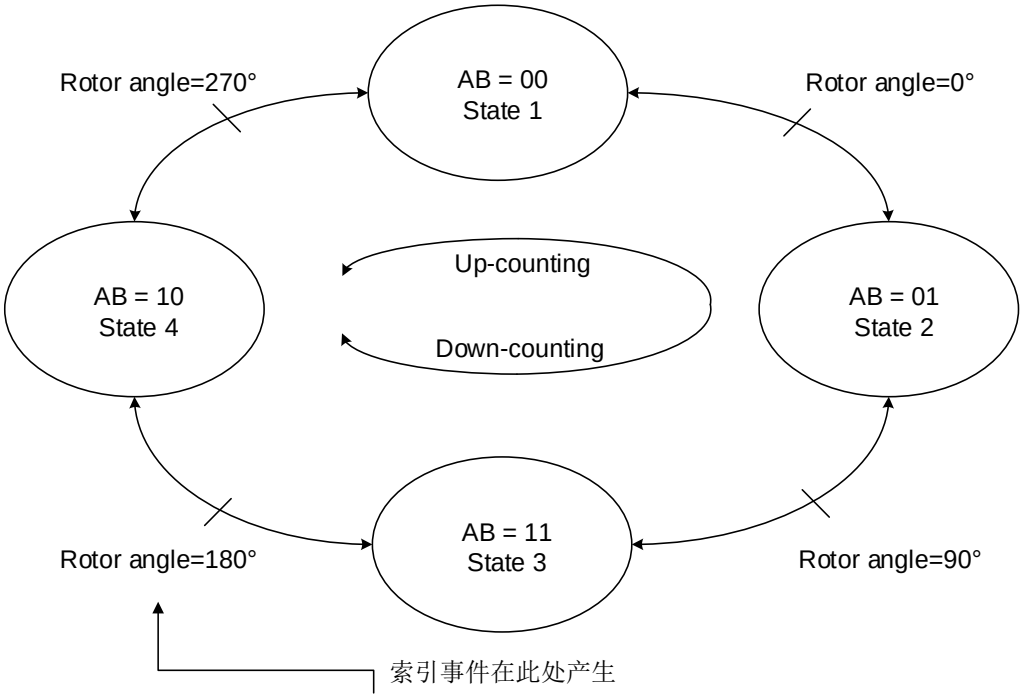


Figure 19-67 Index Generation at IPOS [1: 0] = 11

The following figure shows the waveform and corresponding values at IPOS [1: 0] = 11, which indicates that the moment when the counter value is forced is automatically adjusted according to the counting direction

- Counter cleared When the encoder enters 11 state (channel A = 1, channel B = 1), counting up (DIR = 0)
- The counter value is set to TIMx_ARR when leaving the 11 state, counting down (DIR = 1)

Index detection events can generate interrupts

The arrow indicates which conversion process index event interrupt is generated

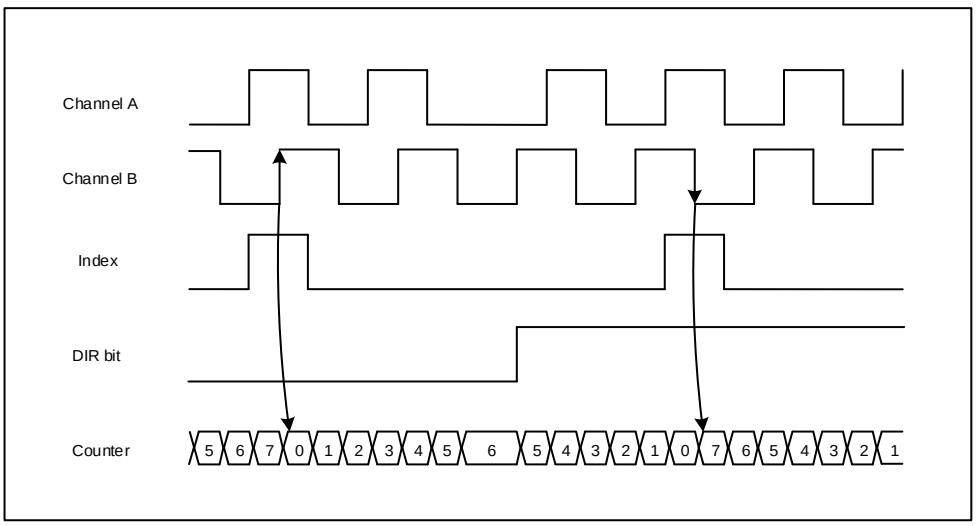


Figure 19-68 Index at IPOS [1: 0] = 11 and count values at channel A gated mode

The following figure shows the waveform and corresponding values in ungated mode, with arrows indicating which transition the index event interrupt occurred

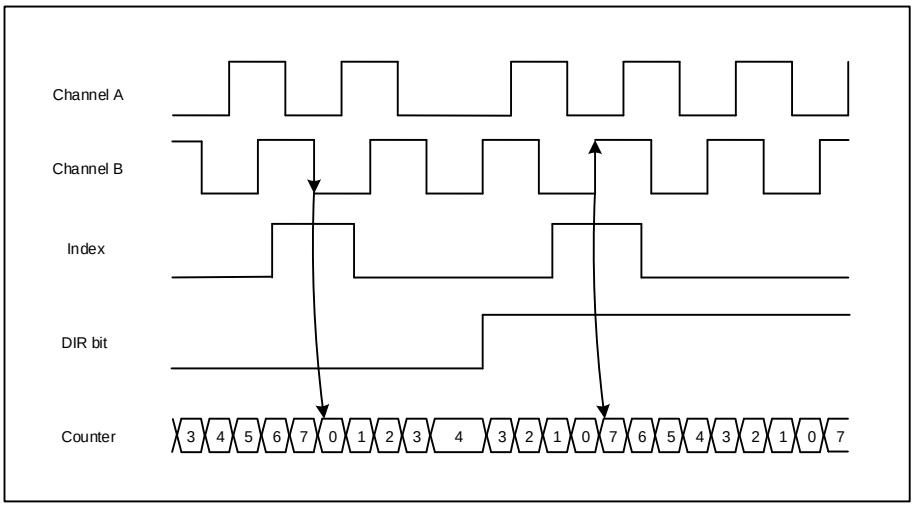


Figure 19-69 Count values in ungated index mode at IPOS [1: 0] = 00

The following figure shows the waveforms and corresponding values in A and B gated modes, with arrows indicating which transition process index event interrupt occurs

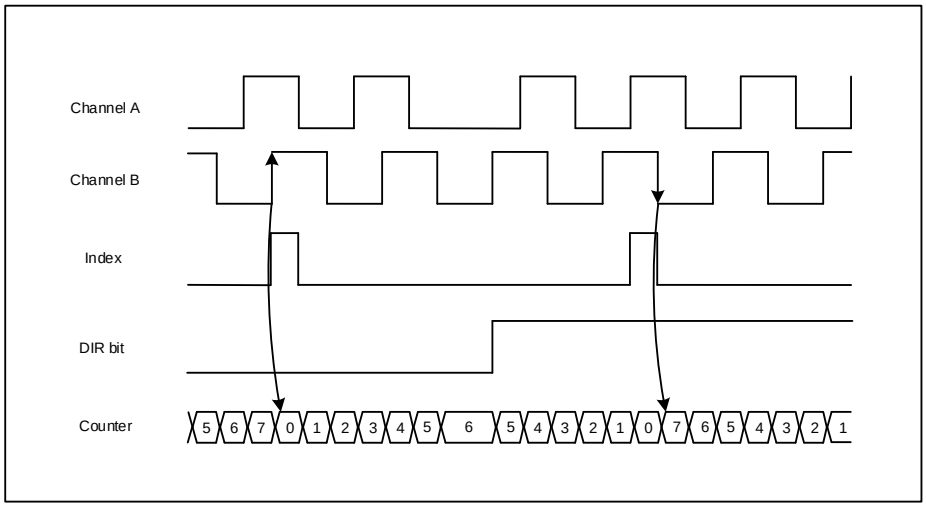


Figure 19-70 indexes and count values when channels A and B are gated

The following two figures illustrate in detail cases where the index pulse width is less than 1/4 of the encoder period

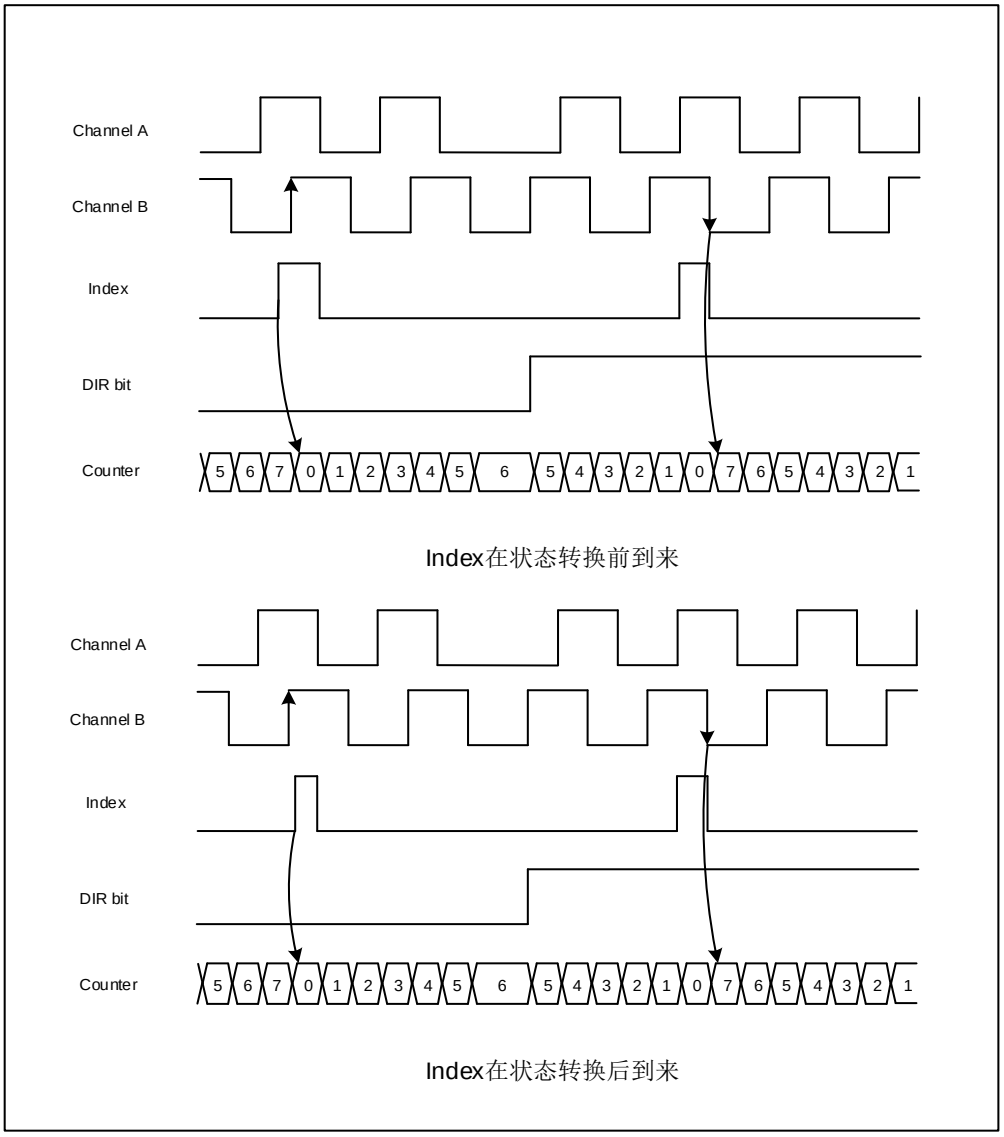


Figure 19-71 Operation of encoder mode with narrow index pulse at IPOS [1: 0] = 11

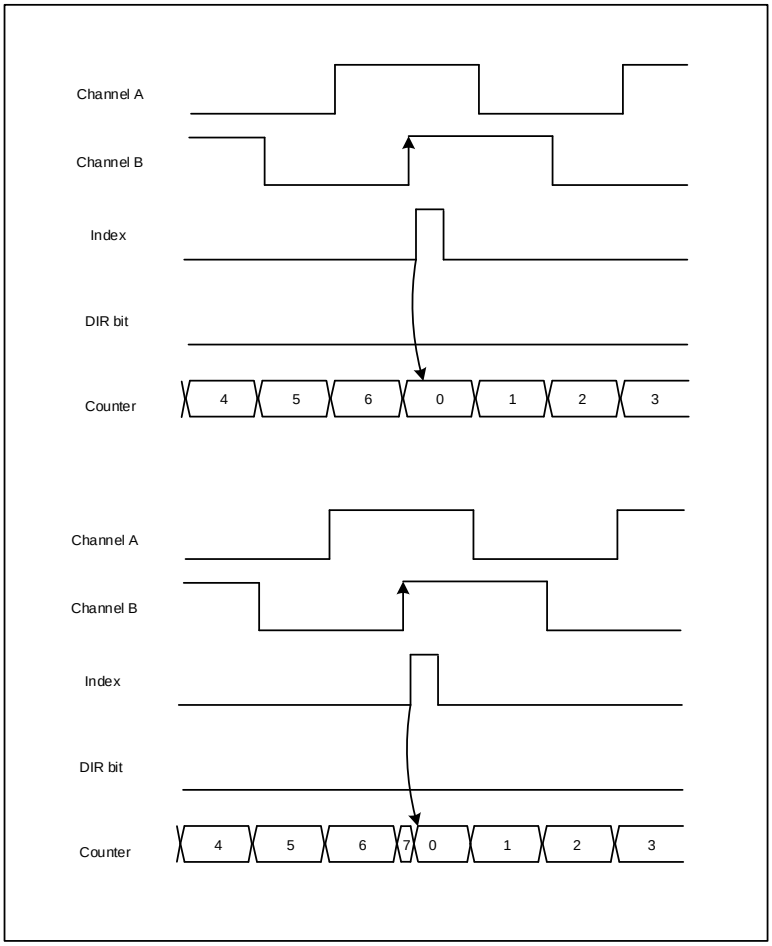


Figure 19-72 ARR = 0x07 time-to-time narrow index pulse reset counter

The following figure shows in detail how indexes are managed in x1 and x2 schemas

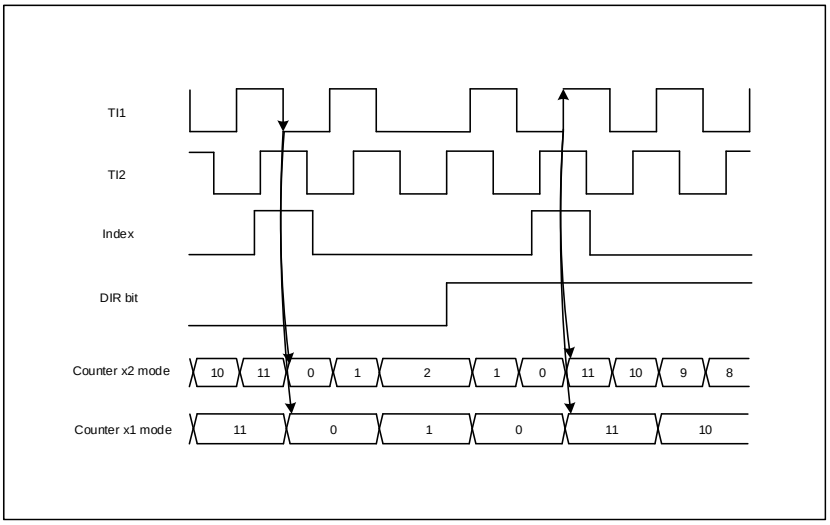


Figure 19-73 IPOS [1: 0] = 01 in x1 and x2 modes

■ Directional index sensitivity

You can allow the index to be valid only in the selected counting direction by configuring the IDIR [1: 0] of the TIMx_ECR register.

The following figure shows the relationship between values, indexes, and count reset events based on IDIR [1: 0].

Note: IDIR must be written until the IE bit is reset (index mode is off)

Note: Directional index sensitivity does not support clock + directional mode, when SMS = 1010 or 1011, IDIR must be set to 00

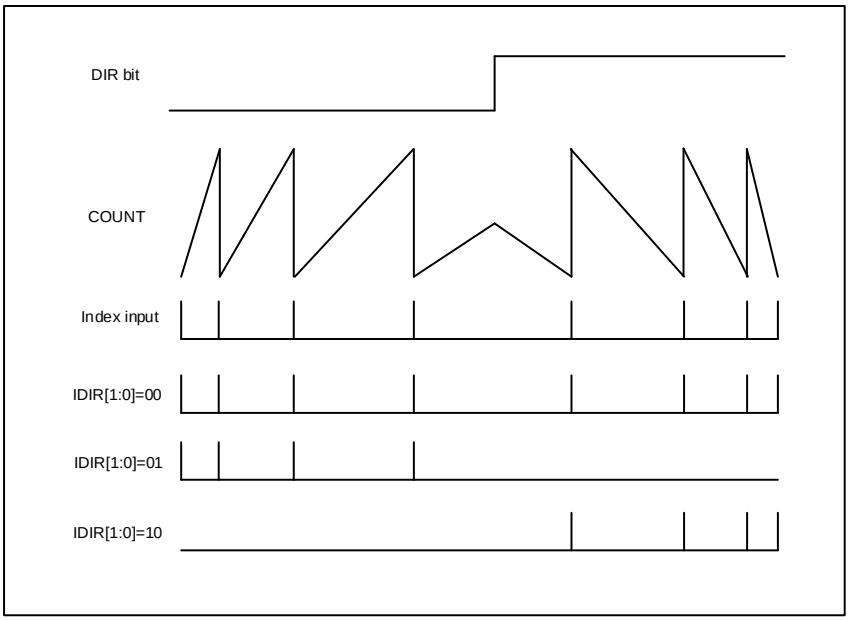


Figure 19-74 oriented index sensitivity

■ Special First Index Event Management

The FIDX bit of the TIMx_ECR register allows indexing to be performed only once, as shown in the figure below. When the first index arrives, the subsequent indexes are ignored. If desired, the circuit can be re-enabled by writing the FIDX bit to 0 and then to 1.

Note: When FIDX = 1, if the direction changes at position 0 (index active), index may be issued twice (IDXF flag set)

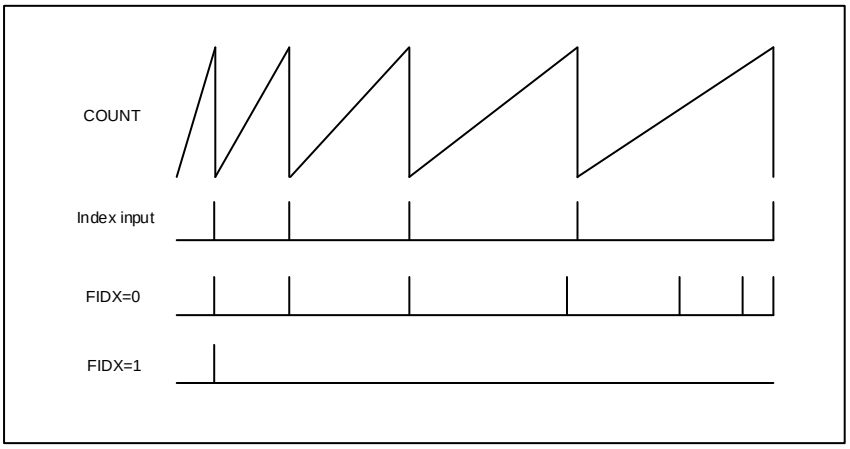


Figure 19-75 Counter reset when FIDX bit is set

■ Index management in non-orthogonal mode

The following two figures describe in detail how indexes are managed in directional clock mode and clock + directional mode. I.e. SMS = 1010 or 1011

In these modes, the sensitivity of the index is determined by the IPOS

IPOS [0] = 0: Detect index at low level of clock

IPOS [0] = 1: Detect index at high level of clock

IPOS [1] has no effect.

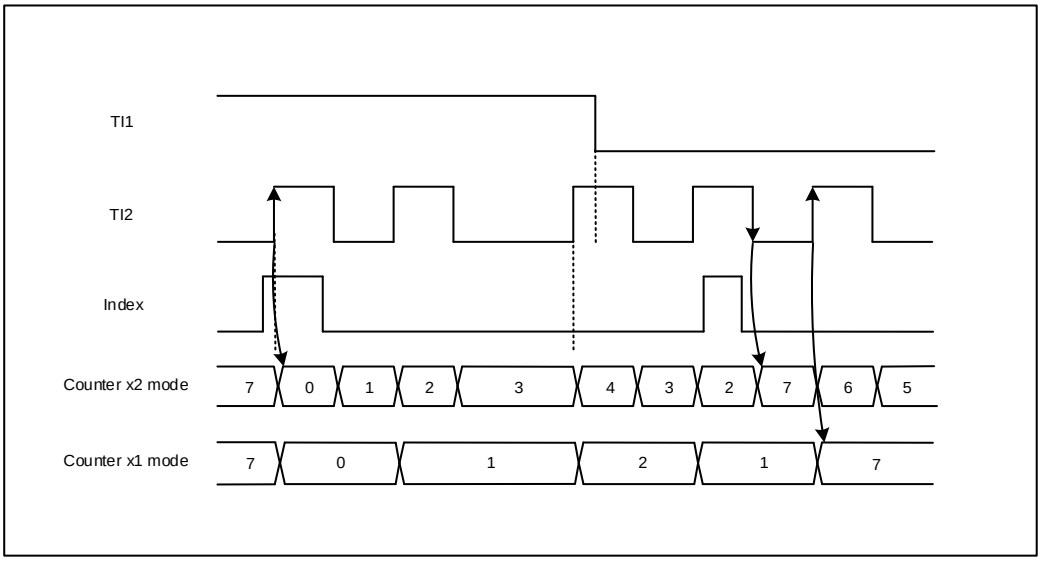


Figure 19-76 clock plus directional mode index behavior (IPOS [0] = 1)

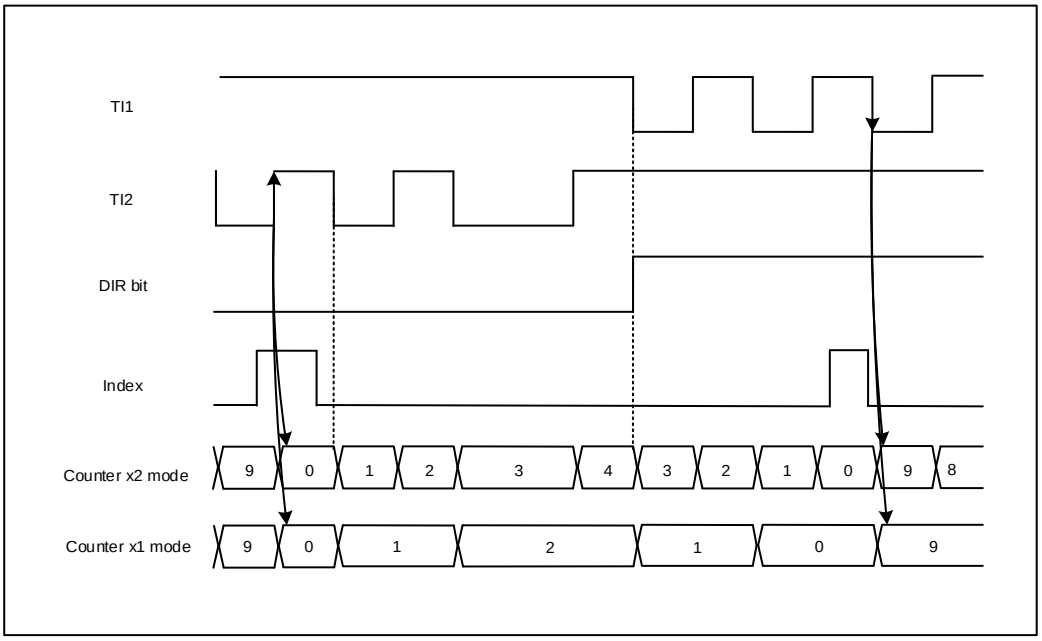


Figure 19-77 oriented clock mode index behavior (IPOS [0] = 1)

■ Encoder Error Management

There are 2 quadrature signals in the encoder configuration that are valid, and it is necessary to detect conversion errors. The readings on the 2 inputs correspond to 2-digit Gray codes to represent the state diagram, as shown in the figure. You can only change 1 bit at a time. The erroneous transition pulls up the interrupt flag TERRF in the TIMx_SR status register. If the TERRIE position of the TIMx_DIER register is 1, a transition error interrupt will be generated.

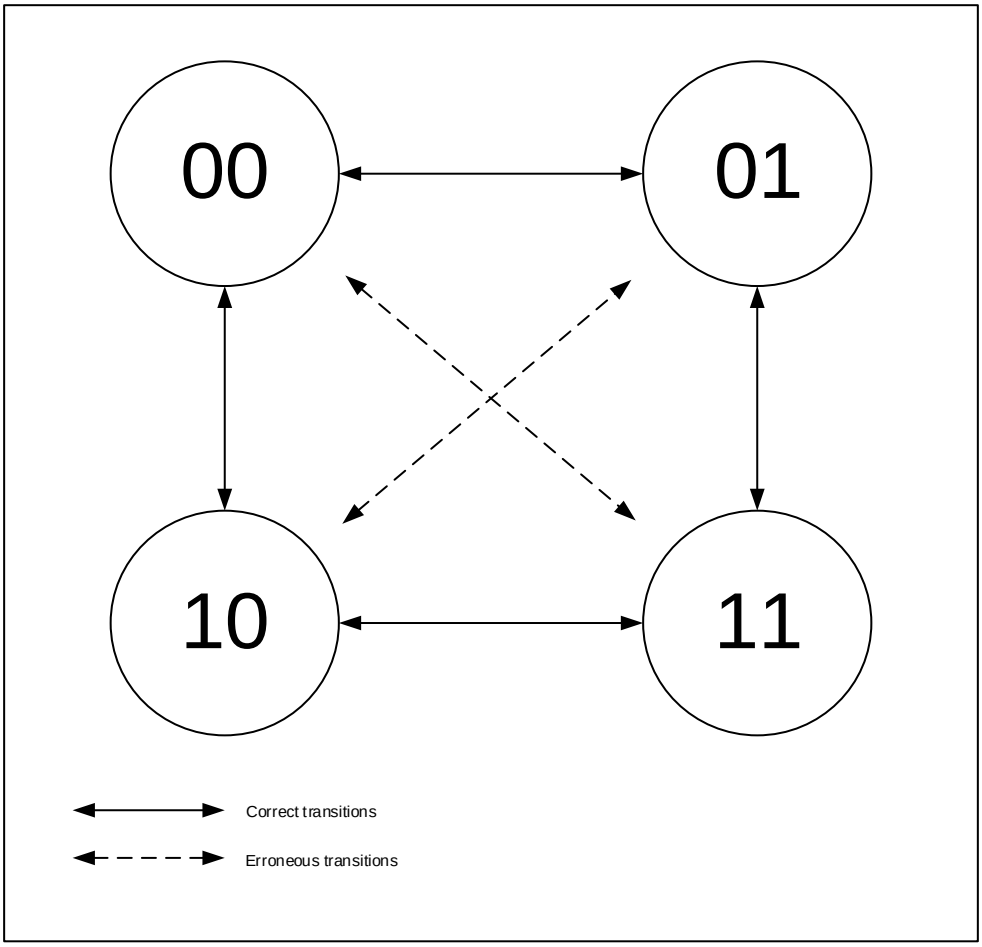


Figure 19-78 Schematic diagram of orthogonal coded signal states

After the encoder has the index signal, it can detect abnormal operation that causes pulse excess during each rotation. The encoder will provide N pulses per rotation and will produce a 4 * N count. The index signal resets the counter every 4 * N clock cycles.

If the counter value is tapered from TIMx_ARR to 0 or from 0 to TIMx_ARR without an index event, this is reported as an index position error.

The overflow threshold is configured by the TIMx_ARR register. The resulting count value of the 1000-line encoder ranges from 0 to 3999 (in 4x read mode). The overflow detection threshold must be set to TIMx_ARR = 3999 +1 = 4000

When counting up, the error is asserted when the transition is delayed to 0 to 1. The figure below shows the processing of narrow index signals in A and B gated modes.

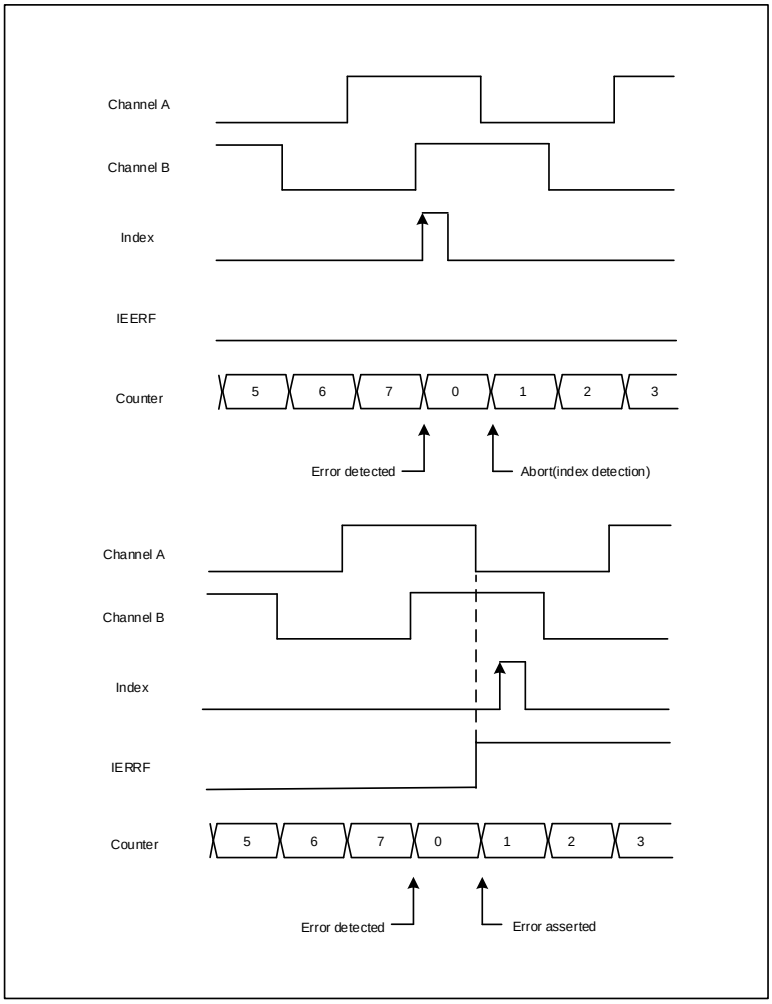


Figure 19-79 Error detection when is coded up

When counting down, the detection must be based on the previous transition from 1 to 0. The figure below shows the processing of narrow index signals in A and B gated modes. In order to avoid the encoder immediately after index detection jitter between the two values of TIMx_ARR and 0, and erroneously generate false determination.

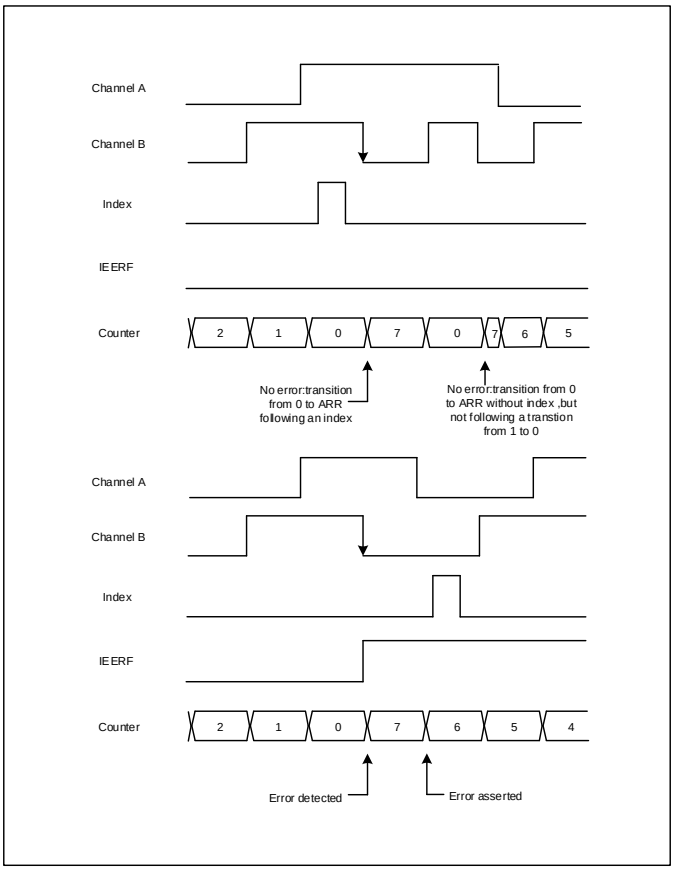


Figure 19-80 Error Detection when Downcoding

An index error will pull up the IERRF interrupt flag bit of the TIMx_SR status register. If the TERRIE position of the TIMx_DIER register is 1, a transition error interrupt will be generated.

- Functional encoder interrupt

The following interrupt are all valid in encoder mode

- Direction change: In encoder mode, any change in counting direction will cause the DIR bit of TIMx_CR1 to flip, and the change in direction will pull up the DIRF interrupt flag bit of the TIMx_SR status register. If the DIRIE bit of the TIMx_DIER register is enabled, a change of direction interrupt is also generated
- Index event: The index event will pull up the IDXF interrupt flag bit of the TIMx_SR status register. An index interrupt is also generated if the IDXIE bit of the TIMx_DIER register is enabled

- Runtime updates encoder mode from mode preload function

It is necessary to switch from one encoder mode to another operation mode, which is usually done at high speeds to reduce the rate of update interrupts. The figure below shows switching from x4 mode to x2 mode and then switching to x1 mode.

For this purpose, SMS [3: 0] can be preloaded. The enable bit is controlled by the SMSPE bit of the TIM_SMCR register. The selection of the trigger signal from the SMS [3: 0] preload to the active value can be controlled using the SMSPS bit of the TIMx_SMCR register.

- SMSPS = 0: Transmission is triggered by UEV, this mode must turn off the index function (IE = 0)
- SMSPS = 1: Transfer triggered by index event

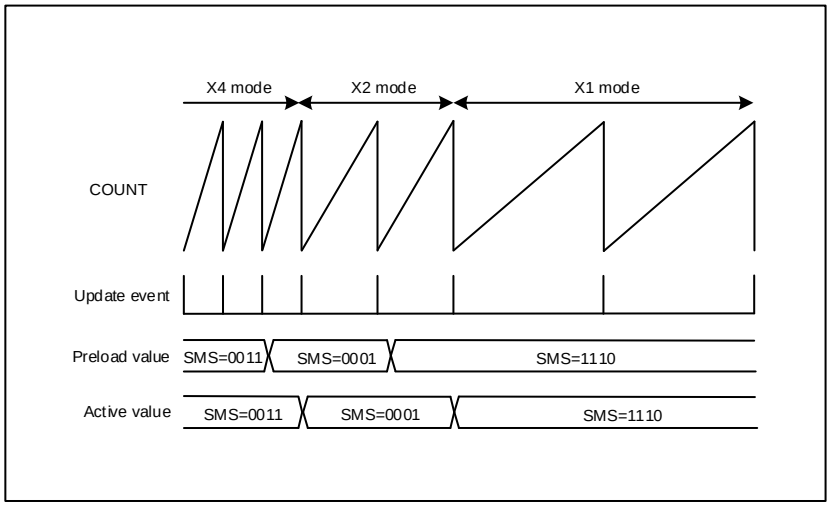


Figure 19-81 The encoder mode is updated by preload when event updates

- Encoder clock output

The working principle of the encoder is not entirely suitable for high-resolution fast measurements. At low speeds, it requires a considerable integration time to have a sufficient number of clock edges and accurate measurements.

At low speeds, the best way is to measure the edge-to-edge clock cycle directly. You may need to use a slave timer at this time. The timer may output the clock signal of the encoder through tim_trgo. The slave timer may perform cycle measurements and provide speed information for each encoder clock edge.

This mode is enabled by setting MMS [3: 0] = 1000 of the TIMx_CR2 register. And can only be used if the SMS [3: 0] is configured to 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111. Other SMS [3: 0] values are not allowed and may produce unexpected behavior.

19.2.24 Directional bit output

It is also necessary to output the direction bit of the timer, which is output from the OC3N anddOC4channels by configuring the OC3M [3: 0] and OC4M [3: 0] bit segments of the TIMx_CCMR2 register to 1011(copying the DIR bit of the TIMx_CR1 register).

This feature is used to monitor the counting direction (or rotation direction) of the encoder, or can provide an indication signal of the upper/lower stages in the intermediate alignment PWM mode.

19.2.25 UIF bit remapping

The UIFREMAP bit of the TIMx_CR1 register can force a continuous copy of the UIF to the 31st bit of the timer counter (TIMxCNT [31]). This allows both the value of the counter and a potential rolling state exhibited by the UIFCPY flag signal to be read out. In special cases, this can simplify the calculation by avoiding race conditions, for example by simultaneous processing of counter readout and update interrupts.

The UIF and UIFCPY flags are generated without delay.

19.2.26 Timer input XOR function

The TI1S bit in the TIMx_CR2 register allows the input filter of channel 1 to be connected to the output of an exclusive OR gate, whose three inputs are TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature being used to connect a Hall sensor is given in the next section.

19.2.27 Interfacing with Hall sensors

When using an advanced control timer (TIM1 or TIM8) to generate a PWM signal to drive the motor, another general-purpose TIMx (TIM2, TIM3, TIM4 or TIM5) timer can be used as an "interface timer" to connect the Hall sensor, asshown in the figure below. The timer input pins (TI1,TI2, TI3) are connected to the TI1 input channel through an XOR gate (selected by setting the TI1S bit in the TIMx_CR2 register), at which time the "interface timer" is used to capture this signal.

The slave mode controller is configuredtoreset mode, and the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

The capture/compare channel 1 on the "interface timer" is configured in the capture mode and the capture signal is TRC (as shown below). The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The "interface timer" can be used to generate a pulse in the output mode, which can be used (by triggering a COM event) to change the properties of each channel of the advanced timer TIM1 or TIM8, while the advanced control timer generates a PWM signal to drive the motor.

Therefore, the "interface timer" channel must be programmed to generate a positive pulse after a specified delay (output comparison or PWM mode), which is sent to the advanced control timer TIM1 or TIM8 through the TRGO output.

Example: one wants to change the PWM configuration of the advanced-control timer after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

- Set the TI1S bit of the TIMx_CR2 register to '1', and configure three timer inputs to logic XOR to the TI1 input;
- Program the time base: write the TIMx_ARR to the max value (the counter must be cleared by the TI1 change. Set the prescaler to obtain a maximum counter period, which is longer than the time interval between two changes on the sensor;
- Set channel 1 to capture mode (select TRC): set CC1S = 01 in the TIMx_CCMR1 register, and set a digital filter if necessary;
- Set channel 2 to PWM2 mode with the required delay: set OC2M = 111 and CC2S = 00 in the TIMx_CCMR1 register;
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIMx_CR2 register to ‘101’.

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the TIMx_CR2 register) and the COM event is controlled by the trigger input (CCUS=1 in the TIMx_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step, which can be done in an interrupt subroutine generated by the rising edge of OC2REF).

The following figure shows an example of this

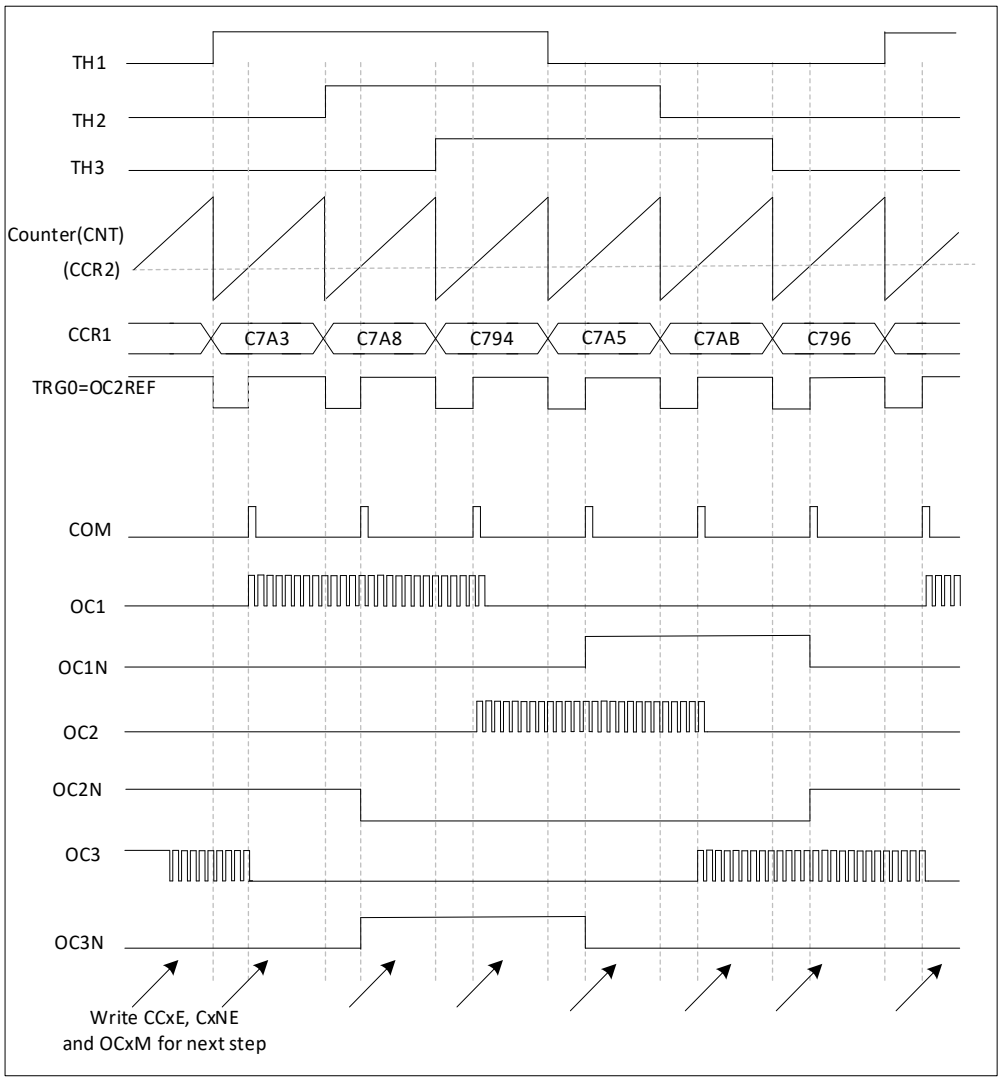


Figure 19-82 Example of Hall sensor interface

19.2.28 Synchronization of TIMx timer and externally triggered

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gated mode, trigger, reset + trigger and gated + reset mode.

19.2.28.1 Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; Meanwhile, if the UDIS bit of the TIMx_CR1 register is low, an update event UEV is also generated; All preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=0100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, a trigger flag (TIF bit in the TIMx_SR register) is set, and an interrupt request or a DMA request is generated according to the setting of the TIE (Interrupt Enable) bit and the TDE (DMA Enable) bit in the TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

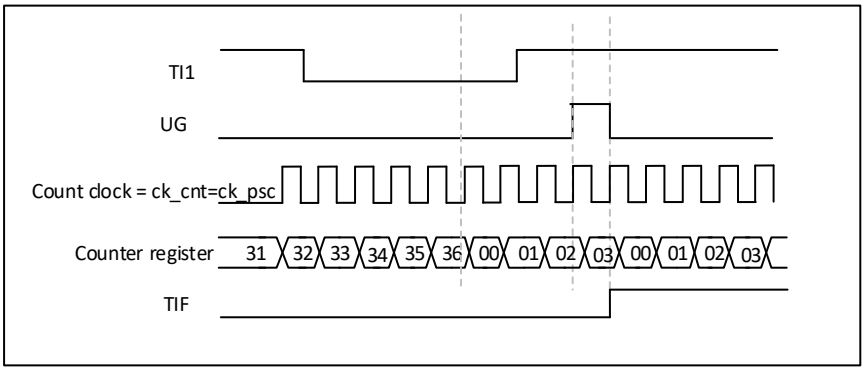


Figure 19-83 Control circuit in reset mode

19.2.28.2 Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. SelectTI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

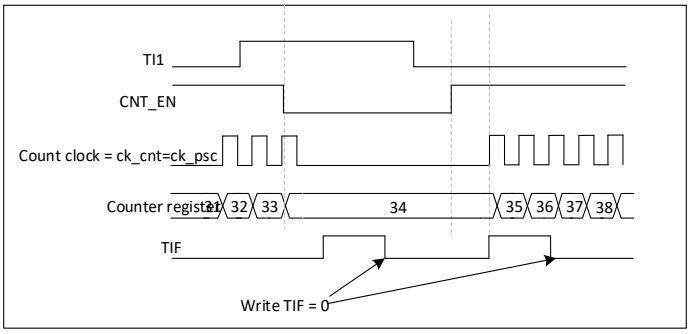


Figure 19-84 Control circuit in Gated mode

19.2.28.3 Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMx_CCMR1 register. Write CC2P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

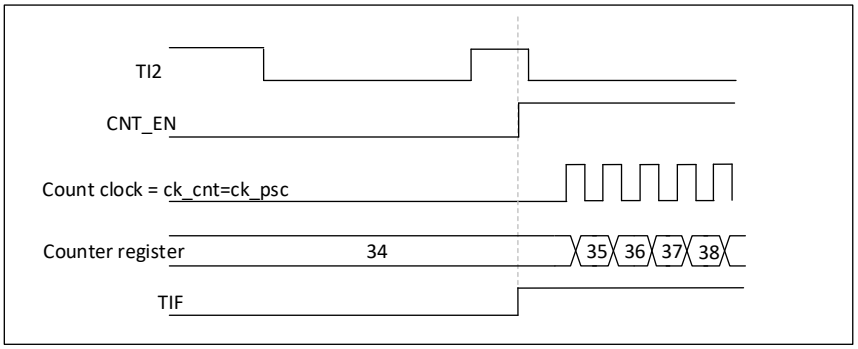


Figure 19-85 Control circuit in trigger mode

19.2.28.4 Slave mode: reset plus trigger combination mode

In this mode, the rising edge of the trigger input initializes the counter and generates an update signal for the register, turning on the counting

This mode is used for a single pulse mode

19.2.28.5 Slave mode: gating plus reset combination mode

The clock of the counter is enabled only when the trigger input is high. The counter stops when the trigger signal is pulled low. The start and stop of the counter can be controlled

This mode allows detection of out-of-range PWM signals (duty cycle exceeding maximum expected value)

19.2.28.6 Slave mode: External clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit through the TIMx_SMCR register:

— ETF = 0000: no filter

— ETPS = 00: prescaler disabled

— ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.

2. Channel 1 is configured as follows to detect the rising edge of TI1:

— IC1F = 0000: no filter

— The capture prescaler is not used for triggering, so it does not need to be configured.

— CC1S=01 in TIMx_CCMR1 register to select only the input capture source

— Set CC1P = 0 in the TIMx_CCER register to determine polarity (only rising edges are detected)

3. Set SMS = 110 in the TIMx_SMCR register, and configure the timer to trigger mode. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

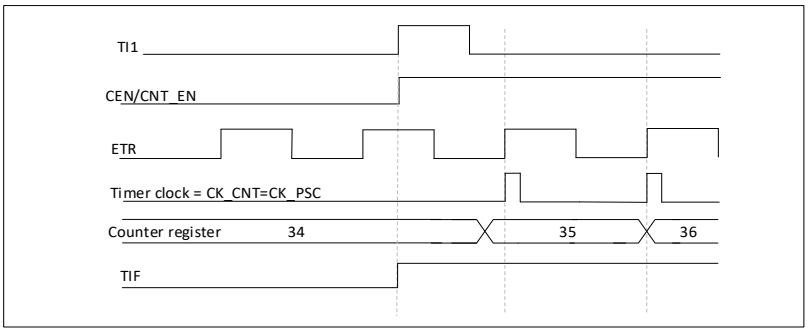


Figure 19-86 Control Circuit in External Clock Mode 2 + Trigger Mode

19.2.29 DMA Burst Mode

The TIMx timer has the ability to generate multiple DMA requests when a single event occurs. The main purpose is to be able to reprogram some functions of the timer multiple times without the overhead of software. But it is also possible to read several registers in a row at fixed intervals.

The destination of the DMA controller is unique and must point to the virtual register TIMx_DMAR. When a given timer event occurs, the timer initiates a series of DMA requests. Each write to the TIMx_DMAR register actually redirects a timer register.

The DBL [4: 0] bit in the TIMx_DCR register sets the DMA burst length. When a read or write access is made to a TIMx_DMR address, the timer recognizes burst transmissions, that is, the number of transmissions (in half a word or byte)

The DBA [4: 0] bit of TIMx_DCR defines the DMA base address for DMA transmission (when read and write operations are completed through the TIMx_DMAR address). The DBA is defined as the offset from the TIMx_CR1 register address.

Example:

00000: TIMx_CR1

00001: TIMx_CR2

00010: TIMx_SMCR

As an example, the timer DMA burst characteristic is used to update the contents in the CCRx register when an update event occurs, transmitting a half-word to CCRx by DMA.

This is done through the following steps:

Configure the corresponding DMA channel as follows:

The DMA channel peripheral address is the address of the DMAR register

The DMA channel storage address is the BUFF address in RAM, which contains the data transmitted by DMA to CCRx

Number of data transferred = 3 (see note)

Polling Mode Off

Configure DCR registers by configuring DBA and DBL: DBL = 3, DBA = 0XE

Enable timer update DMA request (UDE = 1)

Enable timer

Enable the DMA channel

This example applies to the case where each CCRx register is updated once. If each CCRx is updated twice, the number of data transfers needs to be set to 6. For example, suppose that the RAM buffer contains data1, data2, data3, data4, data5 and

data6. Data is sequentially transmitted to CCRx: the first update DMA request, data1 is transmitted to CCR2, data2 is transmitted to CCR3, and data3 is transmitted to CCR4; For the second update DMA request, data4 is transmitted to CCR2, data5 is transmitted to CCR3, and data6 is transmitted to CCR4.

Note: Null values can be written to the reserved register

19.2.30 TIM1/TIM8 DMA Request

TIM1/TIM8 can generate DMA requests as shown in the following table:

Table 19-7 DMA Request

DMA request source	DMA enable control bit
update	UDE
Compare/Capture 1	CC1DE
Compare/Capture 2	CC2DE
Comparison/Capture 3	CC3DE
Comparison/Capture 4	CC4DE
COM	COMDE
Trigger	TDE

19.2.31 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. See Section 20.2.22: Timer Synchronization for details.

19.2.32 Debug Mode

When the microcontroller enters debug mode (Cortex-M4F core stops), according to the setting of DBG_TIMx_STOP in the DBG module,

The TIMx counter may either continue normal operation or stop. See subsequent DBG sections for details.

Behavior in debug mode can be programmed using a dedicated configuration for each timer in the Debug Support Module (DBG)

For safety reasons, when the counter stops, the output is disabled (if the MOE is reset). The output can be forced to an invalid level (OSSI = 1), or taken over by the GPIO controller (OSSI = 0), typically forced to high impedance.

See DBG module description for details

19.2.33 TIM1/TIM8 Interrupt

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
Comparison/Capture 3	CC3IF	CC3IE
Comparison/Capture 4	CC4IF	CC4IE
COM	COM	COMIE
Trigger	TIF	TIE
Encoder Index	IDXF	IDXIE
Encoder direction	DIRF	DIRIE
Brake	BIF	BIE
Brake 2	B2IF	
System brake	SBIF	
Encoder index error	IERRF	IERRIE
Encoder transmission error	TERRF	TERRIE

19.3 TIMx registers

19.3.1 TIM1 and TIM8 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DITHEN	UIFREMAP	Res	CKD [1: 0]		ARPE	CMS [1: 0]		DIR	OPM	URS	UDIS	CEN
			RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 13	Reserved	-	-	-
12	DITHEN	RW	0	Jitter enable (Dithering enbale) 0: Jitter off 1: Jitter on Note: This bit can only be modified when the CEN bit is reset
11	UIFREMAP	RW	0	UIF status bit remapping enable 0: No remapping, UIF status bit is not copied to bit 31 of TIMx_CNT register 1: Remap, UIF status bit copied to bit 31 of TIMx_CNT register
10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the division ratio of the timer clock (CK_INT) frequency and dead time and the division ratio between the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 5	CMS	RW	2'h0	Center-aligned mode selection

				00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. The output comparison interrupt flag bit of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) is set only when the counter is counted down. 10: Center-aligned mode 2. The counter counts up and down alternatively. The output comparison interrupt flag bit of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) is set only when the counter is counted up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The output of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) compares the interrupt flag bit, which is set when the counter counts up and down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
4	DIR	RW	0	Direction 0: Counter counts up; 1: The counter counts down. Note: This bit is read-only when the counter is configured in center alignment mode or encoder mode.
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

19.3.2 TIM1 and TIM8 control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						MMS[3]	Res	MMS2 [3: 0]				Res	OIS6	Res	OIS5
						RW		RW	RW	RW	RW		RW		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OIS4N	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS [2: 0]			CCDS	CCUS	Res	CCPC
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	MMS[3]	RW	0	See MMS [2: 0] description for details

24	Reserved	-	-	-
23: 20	MMS2 [3: 0]	RW	4'h0	Main mode selection (Master mode selection 2) These four bits are used to select synchronization information (TRGO2) to send to the ADC, and possible combinations are as 0000: Reset-The UG bit of the TIMx_EGR register is used as a trigger output (TRGO2). If the trigger input (slave mode controller in reset mode) produces a reset, the signal on TRGO is relative to the actual reset There will be a delay. 0001: Enable-Counter enable signal CNT_EN may be used as a trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timeris enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 0010: UPDATE-The update event is selected as the trigger output (TRGO2). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Comparison Pulse-Once a catch occurs or a comparison is successful, when the CC1IF flag is to be set (i.e. it is already high), the trigger output sends a positive pulse (TRGO2). 0100: Compare-OC1REFC signal is used as trigger output (TRGO2). 0101: Compare-OC2REFC signal is used as trigger output (TRGO2). 0110: Compare-OC3REFC signal is used as trigger output (TRGO2). 0111: Compare-OC4REFC signal is used as trigger output (TRGO2). 1000: Compare-OC5REFC signal is used as trigger output (TRGO2). 1001: Compare-OC6REFC signal is used as trigger output (TRGO2). 1010: Comparison pulse-OC4REFC signal rising or falling edge is used as trigger output (TRGO2). 1011: Comparison pulse-rising or falling edge of OC6REFC signal is used as trigger output (TRGO2) 1100: Comparison pulse-OC4REFC signal rising edge or OC6REFC signal rising edge used as trigger output (TRGO2) 1101: Comparison pulse-OC4REFC signal rising edge or OC6REFC signal falling edge is used as trigger output (TRGO2) 1110: Comparison pulse-OC5REFC signal rising edge or OC6REFC signal rising edge is used as trigger output (TRGO2) 1111: Comparison pulse-OC5REFC signal rising edge or OC6REFC signal falling edge is used as trigger output (TRGO2) NOTE: The clock of the slave timer or ADC must be turned on before receiving the master timer event and not changed while receiving it
19	Reserved	-	-	-
18	OIS6	RW	0	Output idle state 6 (OC6 output) Refer to OIS1 bit.
17	Reserved	-	-	-
16	OIS5	RW	0	Output idle state 5 (OC5 output) Refer to OIS1 bit.
15	OIS4N	RW	0	Output idle state 4 (OC4N output) Refer to OIS1N bit
14	OIS4	RW	0	Output idle state 4 (OC4 output) Refer to OIS1 bit.
13	OIS3N	RW	0	Output idle state 3 (OC3N output) Refer to OIS1N bit
12	OIS3	RW	0	Output idle state 3 (OC3 output) Refer to OIS1 bit.
11	OIS2N	RW	0	Output idle state 2 (OC2N output) Refer to OIS1N bit
10	OIS2	RW	0	Output idle state 2 (OC2 output) Refer to OIS1 bit.
9	OIS1N	RW	0	Output Idle state 1 0: OC1N = 0 after dead time when MOE = 0; 1: OC1N = 1 after dead zone when MOE = 0. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
8	OIS1	RW	0	Output Idle state 1 0: when MOE = 0, OC1 = 0 after dead time if OC1N is achieved; 1: When MOE = 0, OC1 = 1 after dead time if OC1N is implemented. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
7	TI1S	RW	0	TI1 selection 0: TIMx_CH1 pin connected to TI1 input; 1: The TIMx_CH1, TIMx_CH2, and TIMx_CH3 pins are connected to the TI1 input via XOR.
6: 4	MMS [2: 0]	RW	3'h0	Main mode selection (Master mode selection) For selecting synchronization information (TRGO) to be sent to the slave timer in master mode. The combination is as follows:

				0000: Reset-The UG bit of the TIMx_EGR register is used as a trigger output (TRGO). If the reset is generated by the trigger input (the slave mode controller is in reset mode), the signal on the TRGO will have a delay relative to the actual reset. 0001: Enable-Counter enable signal CNT_EN may be used as a trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timeris enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 0010: Update-The update event is selected as Trigger Output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Comparison Pulse-When a catch occurs or a comparison is successful, when the CC1IF flag is to be set (even if it is already high), the trigger output sends a positive pulse (TRGO). 0100: Compare - OC1REFC signal is used as trigger output (TRGO) 0101: Compare - OC2REFC signal is used as trigger output (TRGO) 0110: Compare - OC3REFC signal is used as trigger output (TRGO) 0111: Compare - OC4REFC signal is used as trigger output (TRGO) 1000: Encoder clock output-encoder clock signal as trigger output (TRGO). This encoded value is only valid if the SMS [3: 0] has values of 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111. Other SMS values may result in unexpected effects. 1001-1111: Reserved Note: 1. The clock of the slave timer or ADC must be enabled prior to receive events from the master timer, and must not be changed. 2. If the master and slave timers are not on the same bus, the master mode should be configured to the width that can be picked by the slave timer.
3	CCDS	RW	0	Capture/compare DMA selection 0: When a CCx event occurs, send a DMA request for CCx; 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/Compare Control Update Selection (Capture/compare control update selection) 0: If the capture/compare control bits are preloaded (CCPC = 1), they can only be updated by setting the COM bit; 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	-
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE, and OCxM bits are not preloaded; 1: CCxE, CCxNE, and OCxM bits are preloaded; When this bit is set, they are only updated after the COM bit is set (COMG bit setting or rising edge detected on tim_trgi, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

19.3.3 TIM1and TIM8slave mode control registers (TIMx_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						SMSPS	SMSPE	Res		TS [4: 3]		Res			SMS[3]
						RW	RW			RW	RW				RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS [1: 0]		ETF [3: 0]				MSM	TS [2: 0]			OCCS	SMS [2: 0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	SMSPS	RW	0	SMS preload source This bit selects whether the event can trigger SMS [3: 0] 0: Transmission triggered by an update event of a timer 1: Transmission triggered by index event

24	SMSPE	RW	0	SMS preload enable This bit selects whether SMS [3: 0] can be preloaded 0: SMS [3: 0] can be preloaded 1: SMS [3: 0] cannot be preloaded
23: 22	Reserved	-	-	-
21: 20	TS [4: 3]	RW	2'h0	See TS [2: 0] description for details
19: 17	Reserved	-	-	-
16	SMS[3]	RW	0	See SMS [2: 0] description for details
15	ETP	RW	0	External trigger polarity External trigger polarity This bit selects whether ETR or the inversion of the ETR is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge; 1: ETR is inverted, active at low level or falling edge
14	ECE	RW	0	External clock enable bit (External clock enable) This bit enables external clock mode 2 0: External clock mode 2 disabled; 1: External clock mode 2 enabled. The counter is drived by any active edge on the ETRF signal. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=00111). Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gate mode and trigger mode; However, the TRGI cannot be connected to the ETRF at this time (the TS bit cannot be '00111'). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13: 12	ETPS	RW	2'h0	Externally triggered prescalation External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. When a faster external clock is input, the frequency of the ETRP can be reduced using prescalation. 00: Turn off prescaling 01: ETRP frequency divided by 2; 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8.
11: 8	ETF	RW	4'h0	Externally triggered filtering External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
7	MSM	RW	0	Master/slave mode 0: No action; 1: The event on the trigger input (TRGI) is delayed to allow perfect synchronization between the current timer (via TRGO) and its slave timers. It is useful if we want to synchronize several timers on a single external event.
6: 4	TS	RW	3'h0	Trigger selection Trigger selection Select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2)

				00111 external trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) 01010: Internal Trigger 6 (ITR6) 01011: Internal Trigger 7 (ITR7) 01100: Internal Trigger 8 (ITR8) 01101: Internal Trigger 9 (ITR9) 01110: Internal Trigger 10 (ITR10) 01111: Internal Trigger 11 (ITR11) 10000: Internal Trigger 12 (ITR12) 10001: Internal Trigger 13 (ITR13) 10010: Internal Trigger 14 (ITR14) 10011: Internal Trigger 15 (ITR15) 10100: Internal Trigger 16 (ITR16) 10101: Internal Trigger 17 (ITR17) 10110: Internal Trigger 18 (ITR18) Others: Reserved For more details about ITRx, see the System Cascade Table Note: These bits can only be changed when they are not used (e.g. SMS = 0000) to avoid erroneous edge detection when changing.
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT connects OCREF_CLR input 1: OCREF_CLR_INT connection ETRF
2: 0	SMS	RW	3'h0	Select from Mode (Slave mode selection) When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 0000: Slave mode disabled - if CEN = 1 then the prescaler is clocked directly by the internal clock. 0001: Quadrature Encoder Mode 1, x2 Mode-The counter counts up/down on the edge of TI1FP1 depending on the level of TI2FP2. 0010: Quadrature Encoder Mode 2, x2 Mode-The counter counts up/down on the edge of TI2FP2 depending on the level of TI1FP1. 0011: Quadrature Encoder Mode 3, x4 Mode-The counter counts up/down on the edges of TI1FP1 and TI2FP2 depending on the input level of the other signal. 0100: Reset Mode-The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a signal to update the register. 0101: Gated mode-When the trigger input (TRGI) is high, the clock of the counter is turned on. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger mode-The counter starts (but does not reset) on the rising edge of the trigger input TRGI, only the start of the counter is controlled. 0111: External clock mode 1-The rising edge of the selected trigger input (TRGI) drives the counter. 1000: Reset + Trigger Combination Mode:-Rising edge of Trigger Input (TRGI) initializes counter, generates register update and starts counter 1001: Gate + Reset Combination Mode:-Trigger input (TRGI) is high when counter clock enabled. The counter stops and resets once the trigger signal is pulled low. The start and stop of the counter are controllable. 1010: Encoder mode: clock + direction, x2 mode 1011: Encoder mode: clock + direction, x1 mode, sensitivity of TI2FP2 set by CC2P 1100: encoder mode: directional clock, x2 mode 1101: Encoder mode: directional clock, x1 mode, sensitivity of TI1FP1, TI2FP2 set by CC1P and CC2P 1110: quadrature encoder mode: x1 mode. Count only at the edge of TI1FP1, the edge sensitivity is set by CC1P 1111: quadrature encoder mode: x1 mode. Counted only at the edge of TI2FP2, the edge sensitivity is set by CC2P Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS=00100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. Note: The slave device (timer, ADC) must be enabled to receive TRGO or TRGO2 before receiving the master timer event, and the clock frequency can no longer be changed during reception Note: Do not use UEV as the tim_trgo output signal in encoder mode (i.e. mms cannot be configured to 010)

19.3.4 TIM1 and TIM8 DMA/Interrupt enable registers (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								TERRIE	IERRIE	DIRIE	IDXIE	Res			
								RW	RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Re-served	-	-	-
23	TERRIE	RW	0	Transmission Error Interrupt Enable Transition error interrupt enable 0: Transmission error interrupt not enabled 1: Transmission Error Interrupt Enable
22	IERRIE	RW	0	Index Error Interrupt Enable (Index error interrupt enable) 0: Index error interrupt not enabled 1: Index error interrupt enable
21	DIRIE	RW	0	Direction change interrupt enable (Direction change interrupt enable) 0: Direction change interrupt disabled 1: Direction change interrupt enable
20	IDXIE	RW	0	Index interrupt enable (Index interrupt enable) 0: Index interrupt not enabled 1: Index interrupt enabled
19: 15	Re-served	-	-	-
14	TDE	RW	0	Trigger DMA request enable 0: prohibit triggering of DMA request; 1: Allow DMA request to be triggered.
13	COMDE	RW	0	COM DMA request enable 0: DMA request of COM is prohibited; 1: Allow DMA requests for COM.
12	CC4DE	RW	0	Capture/Compare 4 DMA request enable 0: DMA request for capture/compare 4 is prohibited; 1: DMA request allowed to capture/compare 4.
11	CC3DE	RW	0	Capture/Compare 3 DMA request enable 0: DMA request for capture/compare 3 is prohibited; 1: DMA request allowed to capture/compare 3. 。
10	CC2DE	RW	0	Capture/Compare 2 DMA request enable 0: DMA request for capture/compare 2 is prohibited; 1: DMA request allowed to capture/compare 2.
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: DMA request to capture/compare 1 is prohibited; 1: DMA request allowed to capture/compare 1. 。
8	UDE	RW	0	Update DMA request enable 0: DMA request to prohibit update; 1: Allow updated DMA requests.
7	BIE	RW	0	Allow brake interruption (Break interrupt enable) 0: Brake interruption is prohibited; 1: Allow the brake to be interrupted.
6	TIE	RW	0	Trigger interrupt enable (Trigger interrupt enable) 0: prohibit triggering interrupt; 1: Enable trigger interrupt.
5	COMIE	RW	0	COM interrupt enable 0: Disable COM interrupt; 1: COM interrupt enabled
4	CC4IE	RW	0	Allow capture/compare 4 interrupts (Capture/Compare 4 interrupt enable) 0: Disable capture/compare 4 interrupt; 1: Allow capture/compare 4 interrupts.
3	CC3IE	RW	0	Allow capture/compare 3 interrupts (Capture/Compare 3 interrupt enable) 0: Disable capture/compare 3 interrupt; 1: Allow capture/compare 3 interrupts.

2	CC2IE	RW	0	Allow capture/compare 2 interrupts (Capture/Compare 2 interrupt enable) 0: Disable capture/compare 2 interrupt; 1: Allow capture/compare 2 interrupts.
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable) 0: Disable capture/compare 1 interrupt; 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

19.3.5 TIM1and TIM8status registers (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	TERRF	IERRF	DIRF	IDXF	IC2IR	IC1IR	CC6IF	CC5IF
		RC_W0													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		SBIF	CC4OF	CC3OF	CC2OF	CC1OF	B2IF	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
		RC_W0													

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29	IC4IF	RC_W0	0	Falling edge capture 4 flag Refer to IC1IF description
28	IC3IF	RC_W0	0	Falling edge capture 3 flag Refer to IC1IF description
27	IC2IF	RC_W0	0	Falling edge capture 2 flag Refer to IC1IF description
26	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by falling edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A falling edge capture event occurs.
25	IC4IR	RC_W0	0	Rising edge capture 4 flag Refer to IC1IR description
24	IC3IR	RC_W0	0	Rising edge capture 3 flag Refer to IC1IR description
23	TERRF	RC_W0	0	Transmission Error Interrupt Flag (Transition error interrupt flag) This flag is set by the hardware when a transmission error occurs in the encoder mode. Can be cleared by software or write cleared 0: No encoder transmission error detected 1: Encoder transmission error detected
22	IERRF	RC_W0	0	Index error interrupt flag (Index error interrupt flag) The flag is set by the hardware when an index error is detected. Can be cleared by software or write cleared
21	DIRF	RC_W0	0	Direction change interrupt flag Direction change interrupt flag This flag is set by hardware when a direction change occurs in the encoder mode (DIR bit change in TIMx_CR). Can be cleared by software or write cleared 0: No direction change detected 1: Direction change detected
20	IDXF	RC_W0	0	Index interrupt flag (Index interrupt flag) The flag is set by the hardware when an index event is detected. Can be cleared by software or write cleared 0: No index event occurred 1: An index event occurs
19	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IR description
18	IC1IR	RC_W0	0	Rising edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by rising edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A rising edge capture event occurs.

17	CC6IF	RC_W0	0	Capture/Compare 6 Interrupt Markers Capture/Compare 6 interrupt flag Refer to CC1IF description Note: Channel 6 can only be configured as output
16	CC5IF	RC_W0	0	Capture/Compare 5 Interrupt Markers (Capture/Compare 5 interrupt flag) Refer to CC1IF description Note: Channel 5 can only be configured as output
15:14	Reserved	-	-	-
13	SBIF	RC_W0	0	System brake interruption flag System break interrupt flag When the system brake input is active, the flag signal will be set hardware. When the system brake input is invalid, it can be cleared by software This flag must be reset to restart the PWM operation 0: No braking event occurred 1: The system brake input is active. An interrupt is generated if the BIE bit of the TIMx_DIER register = 1
12	CC4OF	RC_W0	0	Capture/Compare 4 Repeat Capture Markers Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 Repeat Capture Markers (Capture/Compare 3 overcapture flag) Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 Repeat Capture Markers (Capture/Compare 2 overcapture flag) Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 Repeat Capture Marker (Capture/Compare 1 overcapture flag) This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been generated. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	B2IF	RC_W0	0	Break2 interrupt flag Once the brake input is active, the position '1' is applied by the hardware. If the brake input is invalid, the bit may be cleared '0' by the software. 0: No break event occurred. 1: An active level has been detected on the break input. Generate an interrupt if BIE = 1 for TIM_DIER
7	BIF	RC_W0	0	Brake interruption mark (Break interrupt flag) Once the brake input is active, the position '1' is applied by the hardware. If the brake input is invalid, the bit may be cleared '0' by the software. 0: No break event occurred. 1: An active level has been detected on the break input. Generate an interrupt if BIE = 1 for TIM_DIER
6	TIF	RC_W0	0	Trigger interrupt flag (Trigger interrupt flag) This position is' 1 'by hardware when a trigger event occurs (a valid edge is detected at the TRGI input when the slave mode controller is in a mode other than gated mode, or either edge in gated mode). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	RC_W0	0	COM interrupt flag Once a COM event is generated (when the capture/compare control bits: CCxE, CCxNE, OCxM have been updated) this bit is set '1' by the hardware. It is cleared by software. 0: No update occurred. 1: COM interrupt waiting for response.
4	CC4IF	RC_W0	0	Capture/Compare 4 Interrupt Markers (Capture/Compare 4 interrupt flag) Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 Interrupt Markers (Capture/Compare 3 interrupt flag) Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare 2 Interrupt Markers (Capture/Compare 2 interrupt flag) Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register.

				When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes high under the counter overflow condition in the up or up/down count mode, or under the counter underflow condition in the down count mode If channel TI1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0, when the repetition counter value overflows or underflows (an update event occurs when the repetition counter = 0). − If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

19.3.6 TIM1and TIM8event generation registers (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							B2G	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
							W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	B2G	W	0	Break generation This bit is set '1' by software to generate a braking event, and '0' is automatically cleared by hardware. 0: No action 1: A Brake 2 event is generated. MOE = 0, B2IF = 1 at this time
7	BG	W	0	Break generation This bit is set '1' by software to generate a braking event, and '0' is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	W	0	Trigger generation This bit is set '1' by the software to generate a trigger event, and '0' is automatically cleared by the hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	COMG	W	0	Capture/compare events, generate control updates Capture/Compare control update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit is only valid for channels with complementary outputs.
4	CC4G	W	0	Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation

				This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action 1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged). The counter is cleared '0' if in centrosymmetric mode or DIR = 0 (count up); If DIR = 1 (count down), the counter takes the value of TIMx_ARR.

19.3.7 TIM1and TIM8Capture/Compare Mode Control Register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							OC2M[3]	Res.							OC1M[3]
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M [2: 0]			OC2PE	OC2FE	CC2S [1: 0]		OC1CE	OC1M [2: 0]			OC1PE	OC1FE	CC1S [1: 0]	
IC2F [3: 0]				IC2PSC [1: 0]				ICIF [3: 0]				ICIPSC [1: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.3.7.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	OC2M[3]	RW	0	See OC2M [2: 0] description for details
23: 17	Reserved	-	-	-
16	OC1M[3]	RW	0	See OC1M [2: 0] description for details
15	OC2CE	RW	0	Output Compare 2 Clear Enable (Output Compare 2 clear enable)
14: 12	OC2M	RW	3'h0	Output Comparison 2 Mode Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 Preload Enable (Output Compare 2 preload enable)
10	OC2FE	RW	0	Output Compare 2 Fast Enable (Output Compare 2 fast enable)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection Capture/Compare 2 selection This bit defines the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	OutputCompare1mode These bits define the action of outputting the reference signal OC1REF, and OC1REF determines the values of OC1, OC1N. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high.

				0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 0011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 0100: Force inactive level OC1REF is forced low. 0101: Force active level OC1REF is forced high. 0110: PWM Mode 1-On up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 0111: PWM Mode 2-On up count, channel 1 is invalid level once TIMx_CNT < TIMx_CCR1, active level otherwise; When counting down, channel 1 is active once TIMx_CNT > TIMx_CCR1, otherwise it is invalid. 1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes invalid again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In the decrement count mode, the channel is in an active state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update 1010: Reserved 1011: Reserved 1100: Combined PWM Mode 1-OC1REF has the same behavior as in PWM Mode 1, OC1REFC is composed of OC1REF and OC2REF or logic 1101: Combined PWM Mode 2-OC1REF has the same behavior as in PWM Mode 2, OC1REFC is composed of OC1REF and OC2REF AND logic 1110: Asymmetric PWM Mode 1: OC1REF has the same behavior as in PWM Mode 1, OC1REFC outputs OC1REF when counting up and OC2REF when counting down 1111: Asymmetric PWM Mode 2: OC1REF has the same behavior as in PWM Mode 2, OC1REFC outputs OC1REF when counting up and OC2REF when counting down Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: In PWM mode, the OC1REF level is changed only when the comparison result is changed or when the output comparison mode is switched from freeze mode to PWM mode. Note 3: This bit segment can be preloaded when the channel has complementary outputs. If the CCPC position of TIMx_CR2 is 1, then the OC1M valid bit is updated with a new value from the preload only when the COM event is generated
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: Only in monopulse mode (OPM = 1 of TIMx_CR1 register), PWM mode can be used without confirming the preload register, otherwise its operation is uncertain.
2	OC1FE	RW	0	Output Compare 1 Fast Enable (Output Compare 1 fast enable) This bit is used to speed up the output response to the triggering input event. Must be used in single pulse mode (OPM position 1 of the TIMx_CR1 register) to achieve fast pulse output when the trigger comes.

				0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. Only available for PWM Mode 1 and PWM Mode 2
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

19.3.7.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 12	IC2F	RW	4'h0	Input Capture 2 Filter (Input capture 2 filter)
11: 10	IC2PSC	RW	2'h0	Input/Capture 2 Prescaler (Input capture 2 prescaler)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1.

				10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

19.3.8 TIM1and TIM8capture/compare mode control register 2 (TIMx_CCMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

See the above description of the CCMR1 register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							OC4M[3]	Res.							OC3M[3]
							RW								RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC4CE	OC4M [2: 0]			OC4PE	OC4FE	CC4S [1: 0]		OC3CE	OC3M [2: 0]			OC3PE	OC3FE	CC3S [1: 0]	
IC4F [3: 0]				IC4PSC [1: 0]				IC3F [3: 0]				IC3PSC [1: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.3.8.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	OC4M[3]	RW	0	See OC4M [2: 0] description for details
23: 17	Reserved	-	-	-
16	OC3M[3]	RW	0	See OC3M [2: 0] description for details
15	OC4CE	RW	0	Output Compare 4 Clear Enable (Output Compare 4 clear enable)
14: 12	OC4M	RW	3'h0	Output Comparison 4 Mode Output Compare 4 mode
11	OC4PE	RW	0	Output Comparison 4 Preload Enable (Output Compare 4 preload enable)
10	OC4FE	RW	0	Output Comparison 4 Fast Enable (Output Compare 4 fast enable)
9: 8	CC4S	RW	2'h0	Capture/Compare 4 selection Capture/Compare 4 selection The 2 bits define the direction of the channel (input/output), and the selection of the input pins: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7	OC3CE	RW	0	Output Compare 3 clear '0' enable (Output Compare 1clear enable)
6: 4	OC3M	RW	3'h0	OutputCompare3mode These bits define the action of outputting the reference signal OC3REF, and OC3REF determines the values of OC3, OC3N. OC3REF is active high whereas OC3 and OC3N active level depends on CC3P and CC3NP bits. 0000: Frozen. The comparison between the output comparison register TIMx_CCR3 and the counter TIMx_CNT has no effect on OC3REF; 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR3), OC3REF is forced to be high. 0010: Set channel 1 to inactive level on match. OC3REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR3). 0011: Toggle OC3REF toggles when TIMx_CNT=TIMx_CCR3. 0100: Force inactive level OC3REF is forced low. 0101: Force active level OC3REF is forced high. 0110: PWM Mode 1-On up count, channel 3 is active once TIMx_CNT < TIMx_CCR3, otherwise invalid level; On counting down, channel 3 is an inactive level (OC3REF = 0) once TIMx_CNT > TIMx_CCR3, otherwise it is an active level (OC3REF = 1). 0111: PWM Mode 2-On up count, channel 1 is invalid level once TIMx_CNT < TIMx_CCR3, active level otherwise; On counting down, channel 1 is active once TIMx_CNT > TIMx_CCR3, otherwise it is invalid.

				1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes inactive again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In the decrement count mode, the channel is in an active state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update 1010: Compare pulse: generate pulse on OC3REF when CCR3 match event occurs, pulse configurable by PWPRSC [2: 0] and PW [7: 0] of TIMx_ECR 1011: Directional output: OC3REF signal covered by DIR bit 1100: Combined PWM Mode 1-OC3REF has the same behavior as in PWM Mode 1, OC3REFC is composed of OC3REF and OC4REF or logic 1101: Combined PWM Mode 2-OC3REF has the same behavior as in PWM Mode 2, OC3REFC is composed of OC3REF and OC4REF AND logic 1110: Asymmetric PWM Mode 1: OC3REF has the same behavior as in PWM Mode 1, OC3REFC outputs OC3REF when counting up and OC4REF when counting down 1111: Asymmetric PWM Mode 2: OC3REF has the same behavior as in PWM Mode 2, OC3REFC outputs OC3REF when counting up and OC4REF when counting down Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC3S='00' (the channel is configured in output). Note 2: In PWM mode, the OC3REF level is changed only when the comparison result is changed or when the output comparison mode is switched from freeze mode to PWM mode. Note 3: This bit segment can be preloaded when the channel has complementary outputs. If the CCPC position of TIMx_CR2 is 1, then the value of the OC3M valid bit updated from the preload only when the COM event is generated Note 4: When using repeatable OPM mode, do not configure the counting mode as central counting mode
3	OC3PE	RW	0	Output Comparison 3 Preload Enable (Output Compare 3 preload enable)
2	OC3FE	RW	0	Output Comparison 3 Fast Enable (Output Compare 3 fast enable)
1: 0	CC3S	RW	2'h0	Capture/Compare 3 selection Capture/Compare 3 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

19.3.8.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 12	IC4F	RW	4'h0	Input Capture 4 Filter (Input capture 4 filter)
11: 10	IC4PSC	RW	2'h0	Input/Capture 4 Prescaler (Input capture 4 prescaler
9: 8	CC4S	RW	2'h0	Capture/Compare 4 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3.

				11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7: 4	IC3F	RW	4'h0	Input Capture 3 Filter (Input capture 3 filter)
3: 2	IC3PSC	RW	2'h0	Input/Capture 3 Prescaler (Input capture 3 prescaler)
1: 0	CC3S	RW	2'h0	Capture/Compare 3 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

19.3.9 TIM1and TIM8capture/compare enable registers (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										CC6P	CC6E	Res		CC5P	CC5E
										RW	RW			RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	CC4NE	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21	CC6P	RW	0	Input/capture 6 output polarity Capture/Compare 4 output polarity Refer to CC1P description.
20	CC6E	RW	0	Input/Capture 6 Output Enable (Capture/Compare 4 output enable) Refer to CC1E description.
19: 18	Reserved	-	-	-
17	CC5P	RW	0	Input/Capture 5 Output Polarity Capture/Compare 4 output polarity Refer to CC1P description.
16	CC5E	RW	0	Input/Capture 5 Output Enable (Capture/Compare 4 output enable) Refer to CC1E description.
15	CC4NP	RW	0	Input/capture 4 complementary output polarity Capture/Compare 3 output polarity Refer to CC1NP description.
14	CC4NE	RW	0	Input/capture 4 complementary output enable (Capture/Compare 3 output enable) Refer to CC1NE description.
13	CC4P	RW	0	Input/Capture 4 Output Polarity Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 Output Enable (Capture/Compare 4 output enable) Refer to CC1E description.
11	CC3NP	RW	0	Input/capture 3 complementary output polarity Capture/Compare 3 output polarity Refer to CC1NP description.
10	CC3NE	RW	0	Input/capture 3 complementary output enable (Capture/Compare 3 output enable) Refer to CC1NE description.
9	CC3P	RW	0	Input/capture 3 output polarity Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 Output Enable (Capture/Compare 3 output enable) Refer to CC1E description.
7	CC2NP	RW	0	Input/capture 2 complementary output polarity (Capture/Compare 2 output polarity) Refer to CC1NP description.
6	CC2NE	RW	0	Input/capture 2 complementary output enable (Capture/Compare 2 output enable) Refer to CC1NE description.

5	CC2P	RW	0	Input/capture 2 output polarity (Capture/Compare 2 output polarity) Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 Output Enable (Capture/Compare 2 output enable) Refer to CC1E description.
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configuration as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/capture 1 complementary output enable (Capture/Compare 1 implementary output enable) 0: OFF-OC1N disables output, so the level of OC1N depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. 1: ON-The OC1N signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output, so the output level of OC1 depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: ON-The OC1 signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Note: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 19-8 Control bits for complementary output channels OCx and OCxN with brake function

Control bits					Output states	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output status	OCxN output state
1	x	0	0	0	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	0	1	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
		1	0	0	Output disabled (not driven by the timer anymore). OCx = CCxP, OCx = 0 OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = CCxNP, OCx = 0 OCxN_EN = 0
		1	0	1	Off-State (output enabled with in-active state) OCx = CCxP, OCx_EN = 1	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
0	0	x	x	x	Output disabled (not driven by the timer anymore).	
	1		0	0		
	1		0	1	Off state (output enabled but invalid level) Asynchronous: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCx_EN = 1; if brake or brake 2 is triggered Then if the clock exists (only valid when brake 1 is triggered, brake 2 is not needed): OCx = OISx, OCxN = OISxN after a dead time, assuming that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN Note: Brake 2 is only for OSSI = OSSR = 1	
	1		1	0		
	1		1	1		

Especially when the dead band is active, the output is always output according to the rules of the dead band output, although at this time moe may have been set to 1 by software or hardware. If neither of the two outputs of a channel is used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP, and CCxNP must all be cleared.

19.3.10 TIM1and TIM8counters (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIFCPY	Res														
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	UIFCPY	R	0	This bit is a read-only register, and the UIF bit of the TIMx_ISR register is copied. If the UIF remap bit in TIMx_CR1 is reset, the bit is reserved and read out as 0
30: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

19.3.11 TIM1and TIM8prescalers (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes the counter being cleared '0' by the UG bit of TIM_EGR or cleared '0' by the slave controller operating in re-set mode.

19.3.12 TIM1and TIM8Automatic Reload Registers (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												ARR [19:16]			
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	ARR	RW	20'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null. Jitter-free mode (DITHEN = 0): This register holds the auto-load value Dither Mode (DITHEN = 1): The register holds an integer portion ARR [19: 4], and ARR [3: 0] contains a dither portion

19.3.13 TIM1and TIM8repeat count registers (TIMx_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REP															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	REP	RW	16'h0	Value of repeat counter Repetition counter value When the preload function is turned on, these bits allow the user to set the update rate of the comparison register (i.e., periodically transfer from the preload register to the current register); If an update interrupt is allowed, the rate at which the update interrupt is generated will also be affected. Each time the down counter reaches 0, an update event is generated and the repeat counter resumes counting from the REP value. The new value written to the TIMx_RCR register only takes effect when the next update event occurs. This means that in PWM mode, (REP+1) corresponds to: - the number of PWM periods in edge-aligned mode - the number of half PWM period in center-aligned mode.

19.3.14 TIM1and TIM8capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR1 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR1	RW/ RO	20'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR1 [15: 0] and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR1 [19: 4], and CCR1 [3: 0] contains a dither portion If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR1 [15: 0], and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR1 [19: 4], and the CCR1 [3: 0] bit is reset

19.3.15 TIM1and TIM8capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR2 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR2	RW/RO	20'h0	Capture/Compare 2 value for channel 2 If channel CC2 is configured as output: CCR2 contains the value loaded into the current capture/compare 2 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC2PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR2 [15: 0], and the CCR2 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR2 [19: 4], and CCR2 [3: 0] contains a dither portion If channel CC2 is configured as input: CCR2 contains the counter value transmitted by the last input capture 2 event (IC2). TIMx_CCR2 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR2 [15: 0], and the CCR2 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR2 [19: 4], and the CCR2 [3: 0] bit is reset

19.3.16 TIM1and TIM8capture/compare register 3 (TIMx_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR3 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR3	RW/RO	20'h0	Capture/Compare 3 value for channel 3 If channel CC3 is configured as output: CCR3 contains the value loaded into the current capture/compare 3 register (preload value). If the preload function is not selected in the TIMx_CCMR2 register (OC3PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR3 [15: 0] and the CCR3 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR3 [19: 4], and CCR3 [3: 0] contains a dither portion If channel CC3 is configured as input: CCR3 contains the counter value transmitted by the last input capture 3 event (IC3). TIMx_CCR3 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR3 [15: 0], and the CCR3 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR3 [19: 4], and the CCR3 [3: 0] bit is reset

19.3.17 TIM1and TIM8capture/compare register 4 (TIMx_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR4 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR4	RW/RO	20'h0	Capture/Compare 4 value for channel 4 If channel CC4 is configured as output: CCR4 contains the value loaded into the current capture/compare 4 register (preload value). If the preload function is not selected in the TIMx_CCMR2 register (OC4PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR4 [15: 0] and the CCR4 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR4 [19: 4], and CCR4 [3: 0] contains a dither portion If channel CC4 is configured as input: CCR4 contains the counter value transmitted by the last input capture 4 event (IC4). TIMx_CCR4 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR4 [15: 0], and the CCR4 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR4 [19: 4], and the CCR4 [3: 0] bit is reset

19.3.18 TIM1and TIM8brake and dead registers (TIMx_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		BK2BID	BKBID	BK2DSRM	BKDSRM	BK2P	BK2E	BK2F [3: 0]				BKF [3: 0]			
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1: 0]		DTG [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Note: The BKBID/BK2BID/BK2P/BK2E/BK2F/BKF/AOE/BKP/BKE/OSSI, OSSR, and DTG [7: 0] bits are write-protected depending on the lock settings, and it is necessary to configure them when the TIMx_BDTR register is first written.

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29	BK2BID	RW	0	Break2 bidirectional Reference BKBID Description
28	BKBID	RW	0	Break bidirectional 0: Brake input BRK in input mode 1: Brake input BRK in bi-directional mode In the bi-directional mode (BKBID is set to 1), the brake input is configured as both an input mode and an open-drain output mode. Any valid braking event will pull the brake input level down, indicating to external devices that a braking event has occurred internally

				Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified. Note: Any write operation to this bit will delay the APB clock from taking effect.
27	BK2DSRM	RW	0	Break2 disarm Reference BKDSRM Description
26	BKDSRM	RW	0	Brake release (Break disarm) 0: Brake input BRK activated 1: Brake input BRK released This bit is cleared by hardware when there is no brake source The BKDSRM bit must be set by software to release the bidirectional output control (open drain output high impedance state) and then poll it until it is reset by hardware, indicating that the fault state has disappeared. Note: Any write operation to this bit will delay the APB clock from taking effect.
25	BK2P	RW	0	Break 2 polarity 0: The brake input BRK2 active level is low 1: The brake input BRK2 active level is high Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified. Note: Any write operation to this bit will delay the APB clock from taking effect.
24	BK2E	RW	0	Break 2 enable This position enables protection of Brake 2 0: Brake 2 function off 1: Brake function turned on Note: Brake 2 must be used when OSSR = OSSI = 1 Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified. Note: Any write operation to this bit will delay the APB clock from taking effect.
23: 20	BK2F [3: 0]	RW	4'h0	Break 2 filter These bits define the frequency at which the BRK2 signal is sampled and the bandwidth at which the BRK2 is digitally filtered. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK2 is asynchronous 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8 Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified.
19: 16	BKF [3: 0]	RW	4'h0	Brake filter (Break filter) These bits define the frequency at which the BRK signal is sampled and the bandwidth at which the BRK is digital filtered. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK2 is asynchronous 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8

				1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8 Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified.
15	MOE	RW	0	Main output enabled (Main output enable) Once the brake input is valid, the bit is asynchronously cleared '0' by the hardware. Depending on the setting value of the AOE bit, this bit can be cleared '0' by software or automatically set to 1. It is only valid for channels configured as output. 0: OC and OCN outputs are disabled or forced to idle state. 1: Turn on the OC and OCN outputs if the corresponding enable bits (CCxE, CCxNE bits of the TIMx_CCER register) are set. See TIMx Capture/Compare Enable Register (TIMx_CCER) for details of OC/OCN enabling.
14	AOE	RW	0	Automatic output enable (Automatic output enable) 0: MOE can only be set '1' by software; 1: The MOE can be set '1' by software or automatically set '1' at the next update event (if the brake input is invalid). Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified.
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
12	BKE	RW	0	Break enable 0: Disable brake input (BRK and CCS clock failure events); 1: Turn on the brake input (BRK and CCS clock failure events). Note: When LOCK level 1 is set (the LOCK bit in the TIMx_BDTR register), this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
11	OSSR	RW	0	"Off state" selection in running mode (Off-state selection for Run mode) This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0); 1: When the timer is not working, once CCxE = 1 or CCxNE = 1, first turn on OC/OCN and output an invalid level, and then set OC/OCN enable output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
10	OSSI	RW	0	"Off state" selection in idle mode (Off-state selection for Idle mode) This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0); 1: When the timer is not working, once CCxE = 1 or CCxNE = 1, the OC/OCN first outputs its idle level and then the OC/OCN enables the output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
9: 8	LOCK	RW	2'h0	Lock settings (Lock configuration) These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot be written to DTG, BKBID, BK2BID, BKE, BKP, AOE bits of TIMx_BDTR register and OISx/OISxN bits of TIMx_CR2 register; 10: Lock level 2, cannot write bits in lock level 1, nor can you write CC polarity bits (once the relevant channel is set to output by CCxS bits, the

				CC polarity bits are CCxP/CCNxP bits of the TIMx_CCER register) and OSSR/OSSI bits; 11: Lock level 3, cannot write bits in lock level 2, and cannot write CC control bits (CC control bits are OCxM/OCxPE bits of TIMx_CCMRx register once the relevant channel is set as output through CCxS bits); Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7: 0	DTG	RW	8'h0	Dead Generator Settings (Dead-time generator setup) This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG [7: 5] = 0xx = > DT = DTG [7: 0] × Tdtg, Tdtg = TDTS; DTG [7: 5] = 10x = > DT = (64 + DTG [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTG [7: 5] = 110 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTG [7: 5] = 111 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

19.3.19 TIM1and TIM8capture/compare register 5 (TIMx_CCR5)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GC5C3	GC5C2	GC5C1	Res									CCR5 [19:16]			
RW	RW	RW										RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR5 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31	GC5C3	RW	0	Packet channel 5 and channel (Group channel 5 and channel 3) Channel 3 output deformation 0: OC5REF has no effect on OC3REFC 1: OC3REFC consists of OC5REF and OC3REF AND logic This bit can have an immediate impact or be preloaded and only updated when an update event occurs (if the preload feature is selected in TIMxCCMR2) Note: This variation can also be applied to combined PWM signals
30	GC5C2	RW	0	Packet channel 5 and channel (Group channel 5 and channel 2) Channel 2 output deformation 0: OC5REF has no effect on OC2REFC 1: OC2REFC consists of OC5REF and OC2REF AND logic This bit can have an immediate impact or be preloaded and only updated when an update event occurs (if the preload feature is selected in TIMxCCMR1) Note: This variation can also be applied to combined PWM signals
29	GC5C1	RW	0	Packet channel 5 and channel Group channel 5 and channel 1 Channel 1 output deformation 0: OC5REF has no effect on OC1REFC 1: OC1REFC consists of OC5REF and OC1REF AND logic This bit can have an immediate impact or be preloaded, and is updated when an update event occurs (enabled if the preload feature is selected in TIMxCCMR1) Note: This variation can also be applied to combined PWM signals
28: 20	Reserved	-	-	-
19: 0	CCR5	RW/RO	20'h0	Capture/Compare 5 value If channel CC5 is configured as output: CCR5 contains the value loaded into the current capture/compare 5 register (preload value). If the preload function is not selected in the TIMx_CCMR3 register (OC5PE bit), the written value is immediately transferred to the current

				register. Else the preload value is copied in the active capture/compare 5 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC5 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR5 [15: 0] and the CCR5 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR5 [19: 4], and CCR5 [3: 0] contains a dither portion

19.3.20 TIM1and TIM8capture/compare register 6 (TIMx_CCR6)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR6 [19:16]			
												RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR6 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR6	RW/RO	20'h0	Capture/Compare 6 value for channel 6 If channel CC6 is configured as output: CCR6 contains the value loaded into the current capture/compare 6 register (preload value). If the preload function is not selected in the TIMx_CCMR3 register (OC6PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 6 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC6 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR6 [15: 0] and the CCR6 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR6 [19: 4], and CCR6 [3: 0] contains a dither portion

19.3.21 TIM1and TIM8capture/compare mode control register 3 (TIMx_CCMR3)

Address offset: 0x50

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							OC6M[3]	Res							OC5M[3]
							RW								RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC6CE	OC6M [2: 0]			OC6PE	OC6FE	Res		OC5CE	OC5M [2: 0]			OC5PE	OC5FE	Res	
RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	OC6M[3]	RW	0	See OC6M [2: 0] description for details
23: 17	Reserved	-	-	-
16	OC5M[3]	RW	0	See OC5M [2: 0] description for details
15	OC6CE	RW	0	Output Compare 6 Clear Enable (Output Compare 6 clear enable)
14: 12	OC6M	RW	3'h0	Output Comparison 6 Mode Output Compare 6 mode Description with reference to OC5M [3: 0]
11	OC6PE	RW	0	Output Comparison 6 Preload Enable (Output Compare 6 preload enable)
10	OC6FE	RW	0	Output Compare 6 Fast Enable (Output Compare 6 fast enable)
9: 8	Reserved	-	-	-
7	OC5CE	RW	0	Output Compare 5 clear '0' enable (Output Compare 5 clear enable)
6: 4	OC5M	RW	3'h0	Output Comparison 5 Mode (Output Compare 5 mode)

				These bits define the action of the output reference signal OC5REF, and OC5REF determines the values of OC5, OC5N. OC5REF is active high whereas OC5 and OC5N active level depends on CC5P and CC5NP bits. 0000: Frozen. The comparison between the output comparison register TIMx_CCR5 and the counter TIMx_CNT has no effect on OC5REF; 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR5), OC5REF is forced to be high. 0010: Set channel 1 to inactive level on match. OC5REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR5). 0011: Toggle OC5REF toggles when TIMx_CNT=TIMx_CCR5. 0100: Force inactive level OC5REF is forced low. 0101: Force active level OC5REF is forced high. 0110: PWM Mode 1-On up count, channel 5 is active once TIMx_CNT < TIMx_CCR5, otherwise it is invalid; On counting down, channel 5 is an inactive level (OC5REF = 0) once TIMx_CNT > TIMx_CCR5, otherwise it is an active level (OC5REF = 1). 0111: PWM mode 2 - In upcounting, channel 5 is inactive as long as TIMx_CNT<TIMx_CCR5 else active. In downcounting, channel 5 is active as long as TIMx_CNT>TIMx_CCR5 else inactive. 1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes inactive again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In decrement count mode, the channel is active until a trigger event (TRGI signal) is detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update Others: Reserved Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: In PWM mode, the OC5REF level is changed only when the comparison result is changed or when the output comparison mode is switched from freeze mode to PWM mode. Note 3: This bit segment can be preloaded when the channel has complementary outputs. If the CCPC position of TIMx_CR2 is 1, then the OC5M valid bit is updated with a new value from the preload only when the COM event is generated Note 4: When using repeatable OPM mode, do not configure the counting mode as central counting mode
3	OC5PE	RW	0	Output Comparison 5 Preload Enable (Output Compare 5 preload enable)
2	OC5FE	RW	0	Output Comparison 5 Fast Enable (Output Compare 5 fast enable)
1: 0	Reserved	-	-	-

19.3.22 TIM1and TIM8dead time register 2 (TIMx_DTR2)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														DTPE	DTAE
														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								DTGF [7: 0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	DTPE	RW	0	Deadtime preload enable

				0: The value of dead time cannot be preloaded 1: The value of the dead time can be preloaded Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
16	DTAE	RW	0	Deadtime asymmetric enable 0: The rising and falling edges have the same dead time, both configured by DTG [7: 0] 1: Rising edge dead time is configured by DTG [7: 0] and falling edge dead time is configured by DTGF [7: 0] Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
15: 8	Reserved	-	-	-
7: 0	DTGF [7: 0]	RW	8'h0	Falling edge dead time generator Dead-time falling edge generator setup This bit segment defines the dead time of the falling edge when the complementary output is inserted, assuming DT denotes its duration: DTGF [7: 5] = 0xx = > DTF = DTGF [7: 0] × Tdtg, Tdtg = TDTS; DTGF [7: 5] = 10x = > DTF = (64 + DTGF [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTGF [7: 5] = 110 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTGF [7: 5] = 111 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified

19.3.23 TIM1and TIM8Encoder Control Registers (TIMx_ECR)

Address offset: 0x58

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					PWPRSC [2: 0]			PW [7: 0]							
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								IPOS		FIDX	Res		IDIR		IE
								RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 27	Reserved	-	-	-
26: 24	PWPRSC	RW	3'h0	Pulse width prescalation Pulse width prescaler This bit sets the clock prescalation of the pulse generator, and the formula is as follows: t_{PWG}= xCK_$2^{(PWPRSC[2:0])}$t_INT
23: 16	PW	RW	8'h0	Pulse width Pulse width This bit defines the pulse duration as follows: t_{PW}= PW [7: 0] xt_{PWG}
15: 8	Reserved	-	-	-
7: 6	IPOS	RW	2'h0	Index positioning Index positioning In orthogonal encoder mode (SMS [3: 0] = 0001, 0010, 0011, 1110, 1111), this bit is used to indicate in which configuration the AB input index event resets the counter 00: When AB = 00, the index resets the counter 01: When AB = 01, the index resets the counter 10: When AB = 10, the index resets the counter 11: When AB = 11, the index resets the counter In directional clock mode or clock + directional mode (SMS [3: 0] = 1010, 1011, 1100, 1101), this bit segment is used to indicate which level index will reset the counter. In directional clock mode, clock inputs are applied to both ways. x0: When clock is 0, index resets counter x1: When clock is 1, index resets counter Note: IPOS [1] bit is invalid
5	FIDX	RW	0	First index

				This bit indicates whether only the first index is considered 0: Index always valid 1: Only the first index can reset the counter
4: 3	Reserved	-	-	-
1	IDIR	RW	0	Index direction Index direction This bit indicate in which count direction the index event resets the counter 00: Index can reset counter regardless of counting direction 01: Index can reset counter only when counting up 10: Index can reset counter only when counting down 11: Reserved
0	IE	RW	0	Index enable Index enable This bit indicates whether the index resets the counter 0: Index closed 1: Index enable

19.3.24 TIM1and TIM8input select registers (TIMx_TISEL)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				TI4SEL [3: 0]				Res				TI3SEL [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				TI2SEL [3: 0]				Res				TI1SEL [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 24	TI4SEL	RW	4'h0	TI4 input selection (tim_TI4 [15: 0] input) 0000: TIMx_CH4 Other: Reserved
23: 20	Reserved	-	-	-
19: 16	TI3SEL	RW	4'h0	TI3 input selection (tim_TI3 [15: 0] input) 0000: TIMx_CH3 Other: Reserved
15: 12	Reserved	-	-	-
11: 8	TI2SEL	RW	4'h0	TI2 input selection (tim_TI2 [15: 0] input) 0000: TIMx_CH2 Other: Reserved
7: 4	Reserved	-	-	-
3: 0	TI1SEL	RW	4'h0	TI1 input selection (tim_TI1 [15: 0] input) 0000: TIMx_CH1 0001:tim _ti1_in1 0010:tim _ti1_in2 0011:tim _ti1_in3 0100:tim _ti1_in4 Other: Reserved

19.3.25 TIM1and TIM8Alternate Function Option Register 1 (TIMx_AF1)

Address offset: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														ETRSEL [3: 2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL [1: 0]		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	Res				BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 14	ETRSEL	RW	4'h0	External trigger source selection (etr_in source selection) This bit segment is used to select an ETR input source 0000: TIMx_ETR 0001:tim_etr1

				0010:tim_etr2 0011:tim_etr 3 0100:tim_etr 4 0101:tim_etr 5 0110:tim_etr 6 0111:tim_etr 7 1000:tim_etr 8 1001:tim_etr 9 1010:tim_etr 10 1011:tim_etr 11 1100:tim_etr 12 1101:tim_etr 13 1110:tim_etr 14 1111:tim_etr 15 Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
13	BKCMP4P	RW	0	Tim_brk_cmp4 input polarity This bit select that input polarity of tim_brk_cmp4 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp4 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp4 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
12	BKCMP3P	RW	0	Tim_brk_cmp3 input polarity This bit select that input polarity of tim_brk_cmp3 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp3 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp3 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
11	BKCMP2P	RW	0	Tim_brk_cmp2 input polarity This bit selects that brk_cmp2 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp2 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp2 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
10	BKCMP1P	RW	0	Tim_brk_cmp1 input polarity This bit selects that brk_cmp1 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp1 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp1 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
9	BKINP	RW	0	TIMx_BKIN input polarity This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
8: 5	Reserved	-	-	-
4	BKCMP4E	RW	0	Tim_brk_cmp4 input enable This bit is used to enable tim_brk_cmp4 on brake input. The tim_brk_cmp4 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp4 input off 1: tim_brk_cmp4 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified

3	BKCMP3E	RW	0	Tim_brk_cmp3 input enable This bit is used to enable tim_brk_cmp3 on brake input. The tim_brk_cmp3 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp3 input off 1: tim_brk_cmp3 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
2	BKCMP2E	RW	0	Tim_brk_cmp2 input enable This bit is used to enable tim_brk_cmp2 on brake input. The tim_brk_cmp2 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp2 input off 1: tim_brk_cmp2 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
1	BKCMP1E	RW	0	Tim_brk_cmp1 input enable This bit is used to enable tim_brk_cmp1 on brake input. The tim_brk_cmp1 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp1 input off 1: tim_brk_cmp1 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
0	BKINE	RW	1	TIMx_BKIN input enable This bit is used to enable the standby function of BKIN when the brake is input. The BKIN input is OR logic with other brake sources as a brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.

19.3.26 TIM1and TIM8Alternate Function Option Register 2 (TIMx_AF2)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res													OCRSEL [2: 0]		
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res		BK2 CMP4P	BK2 CMP3P	BK2 CMP2P	BK2 CMP1P	BK2INP	Res				BK2 CMP4E	BK2 CMP3E	BK2 CMP2E	BK CMP1E	BK2INE
		RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 19	Reserved	-	-	-
18: 16	OCRSEL	RW	3'h0	OCREF Reset source selection (OCREF_clr source selection) This bit segment is used to select that ocref_clr input source 0000: ocref_clr0 0001: ocref_clr1 0010 ocref_clr2 0011 ocref_clr3 Others: Reserved Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
15: 14	Reserved	-	-	-
13	BK2CMP4P	RW	0	Tim_brk2_cmp4 input polarity This bit select that input polarity of tim_brk2_cmp4 and must be configured at the same time as the BK2P polarity bit 0: tim_brk2_cmp4 input polarity does not flip (low active when BK2P = 0, high active when BK2P = 1) 1: tim_brk2_cmp4 input polarity flip (high active when BK2P = 0, low active when BK2P = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
12	BK2CMP3P	RW	0	Tim_brk2_cmp3 input polarity This bit select that input polarity of tim_brk2_cmp3 and must be configured at the same time as the BK2P polarity bit

				0: tim_brk2_cmp3 input polarity does not flip (low active when BK2P = 0, high active when BK2P = 1) 1: tim_brk2_cmp3 input polarity flip (high active when BK2P = 0, low active when BK2P = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
11	BK2CMP2P	RW	0	Tim_brk2_cmp2 input polarity This bit select that input polarity of tim_brk2_cmp2 and must be configured at the same time as the BK2P polarity bit 0: tim_brk2_cmp2 input polarity does not flip (low active when BK2P = 0, high active when BK2P = 1) 1: tim_brk2_cmp2 input polarity flip (high active when BK2P = 0, low active when BK2P = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
10	BK2CMP1P	RW	0	Tim_brk2_cmp1 input polarity This bit select that input polarity of tim_brk2_cmp1 and must be configured simultaneously with the BK2P polarity bit 0: tim_brk2_cmp1 input polarity does not flip (low active when BK2P = 0, high active when BK2P = 1) 1: tim_brk2_cmp1 input polarity flip (high active when BK2P = 0, low active when BK2P = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
9	BK2INP	RW	0	TIMx_BKIN2 input polarity This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BK2P polarity bit 0: BKIN2 input polarity does not flip (low active when BK2P = 0, high active when BK2P = 1) 1: BKIN2 input polarity flip (high active when BK2P = 0, low active when BK2P = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
8: 5	Reserved	-	-	-
4	BK2CMP4E	RW	0	Tim_brk2_cmp4 input enable This bit is used to enable tim_brk2_cmp4 on brake input. The tim_brk2_cmp4 input is OR logic with other brake sources as a brake input. 0: brk2_cmp4 input off 1: brk2_cmp4 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
3	BK2CMP3E	RW	0	Tim_brk2_cmp3 input enable This bit is used to enable tim_brk2_cmp3 on brake input. The tim_brk2_cmp3 input is OR logic with other brake sources as a brake input. 0: tim_brk2_cmp3 input off 1: tim_brk2_cmp3 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
2	BK2CMP2E	RW	0	Tim_brk2_cmp2 input enable This bit is used to enable tim_brk2_cmp2 on brake input. The tim_brk2_cmp2 input is OR logic with other brake sources as a brake input. 0: tim_brk2_cmp2 input off 1: tim_brk2_cmp2 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
1	BK2CMP1E	RW	0	Tim_brk2_cmp1 input enable This bit is used to enable tim_brk2_cmp1 on brake input. The tim_brk2_cmp1 input is OR logic with other brake sources as a brake input. 0: tim_brk2_cmp1 input off 1: tim_brk2_cmp1 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
0	BK2INE	RW	0	TIMx_BKIN input enable This bit is used to enable the standby function of BKIN2 on brake input. The BKIN2 input is OR logic with other brake sources as a brake input.

				0: BKIN2 input off 1: BKIN2 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.

19.3.27 TIM1and TIM8DMA control registers (TIMx_DCR)

Address offset: 0x3DC

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DBL [4: 0]					Res			DBA [4: 0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12: 8	DBL	RW	5'h0	DMA burst length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register, the timer makes a continuous transfer), i.e. defines the number of transfers, which can be half words (double bytes) or bytes: 00000: 1 transfer 00001: 2 transfers 00010: 3 transmissions..... ..... 11001: 26 transfers Example: We consider such a transmission: DBL = 7 bytes, DBA = TIMx_CR1 If DBL = 7 bytes and DBA = TIMx_CR1 represents the address of the data to be transmitted, then the transmitted address is given by: (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL Where (address of TIMx_CR1) + DBA plus 7 gives the address where data will be written or read out, so that the transfer of data will occur in 7 registers starting from address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following can occur: -If the data is set to a half word (16 bits), then the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, for the timer, the user must specify the data width to be transmitted by the DMA.
7: 5	Reserved	-	-	-
4: 0	DBA	RW	5'h0	DMA base address These bits define the base address of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register), and the DBA is defined as the offset from the address where the TIMx_CR1 register is located: 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR …….

19.3.28 DMA address for TIM1and TIM8continuous modes (TIMx_DMAR)

Address offset: 0x3E0

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DMAB	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA index) x4 where: The "TIMx_CR1 address" is an address where the control register 1 (TIMx_CR1) is located; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

20. General Purpose Timer (TIM2 to TIM5 / TIM18)

20.1 Introduction

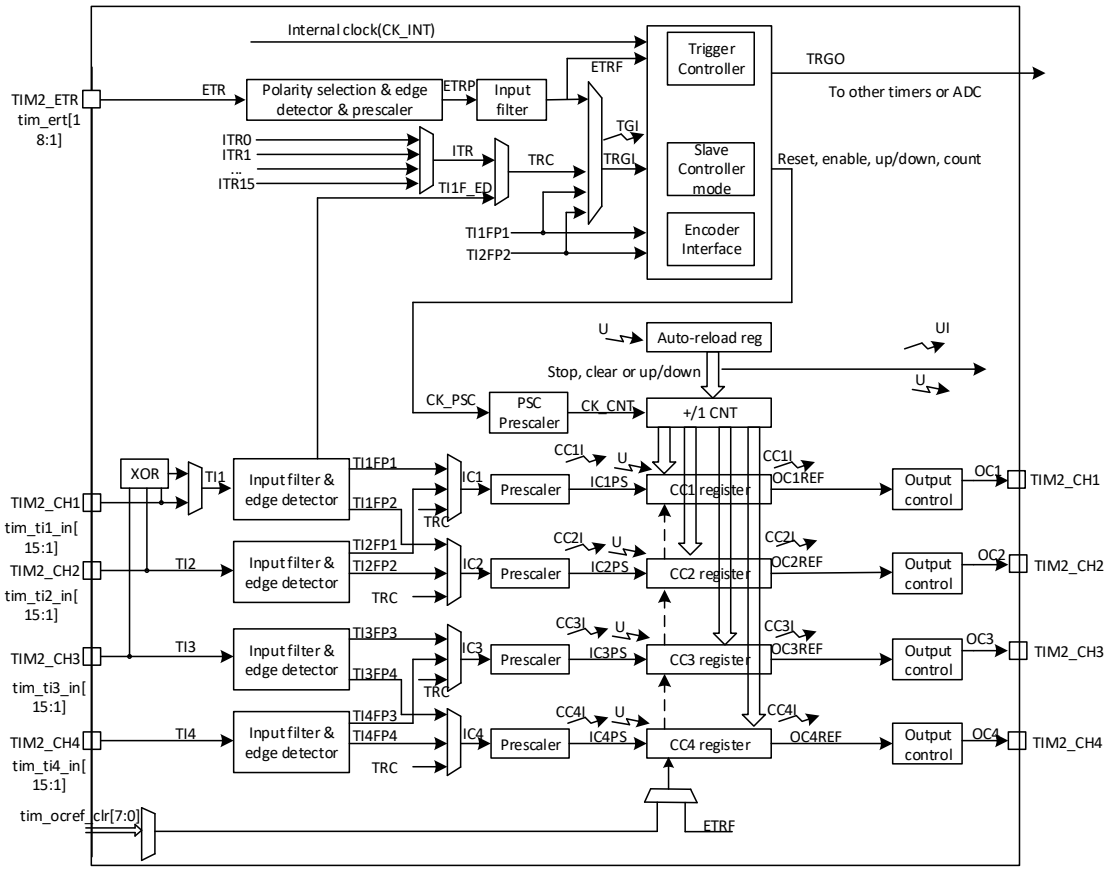
The Universal Control Timer (TIMx) consists of a 32-bit or 16-bit auto-load counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The Universal Control Timer (TIMx) and the Advanced Timer (TIMx) are completely independent, they do not share any resources. They can be synchronized together.

20.1.1 TIM2/3/4/5/18 Main Features

Key features include:

- 32-bit (TIM2) or 16-bit (TIM3/4/5/18) up, down, up/down auto-load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- Up to 4 independent channels:
  - Input capture
  - Output compare
  - PWM generation (edge or center alignment mode)
  - One-pulse mode output
- Synchronization circuit for controlling timer and interconnection between timer by external signal
- An interrupt/DMA occurs when the following events occur:
  - Update: Counter upflow/downflow, counter initialization (via software or internal/external trigger)
  - Trigger event (counter starts, stops, initializes or counts by internal/external triggers)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoders for positioning
- Trigger input for external clock or cycle-by-cycle current management

20.1.2 CAN block diagram



20.2 TIM2/3/4/5/18 Functional Description

20.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition (overflow or underflow) and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set. (See the description of the slave mode controller for more details on enabling counters).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

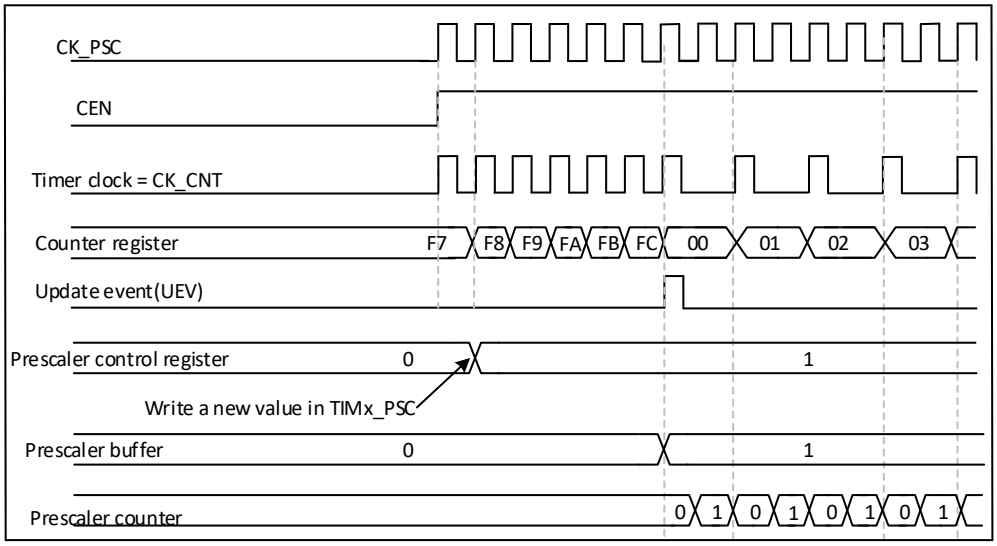


Figure 20-1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

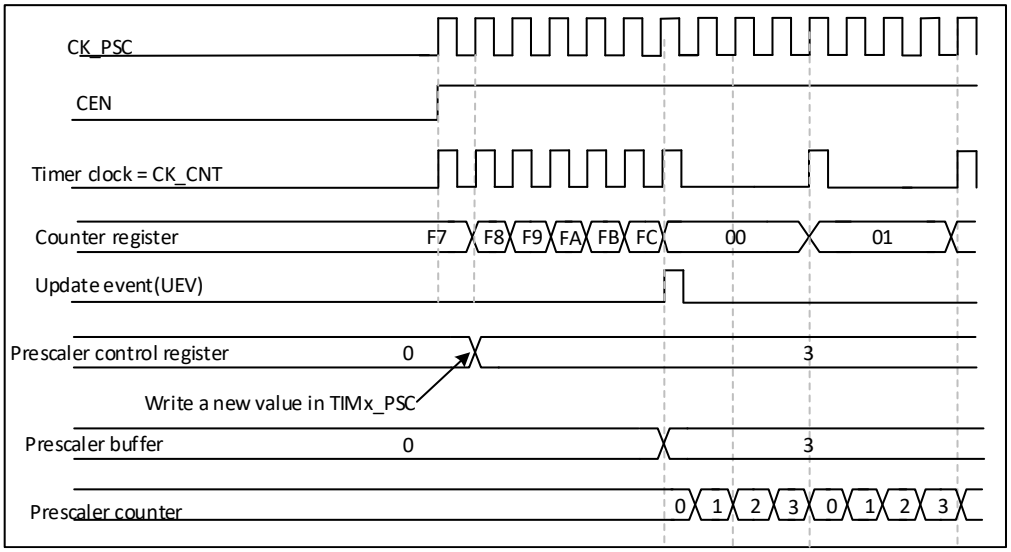


Figure 20-2 Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

20.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event. Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt or DMA request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

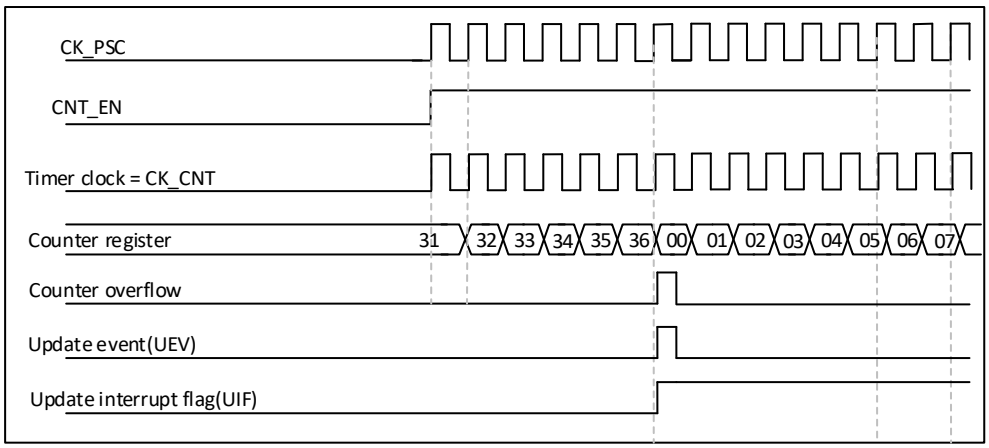


Figure 20-3 Counter timing diagram, internal clock divided by 1

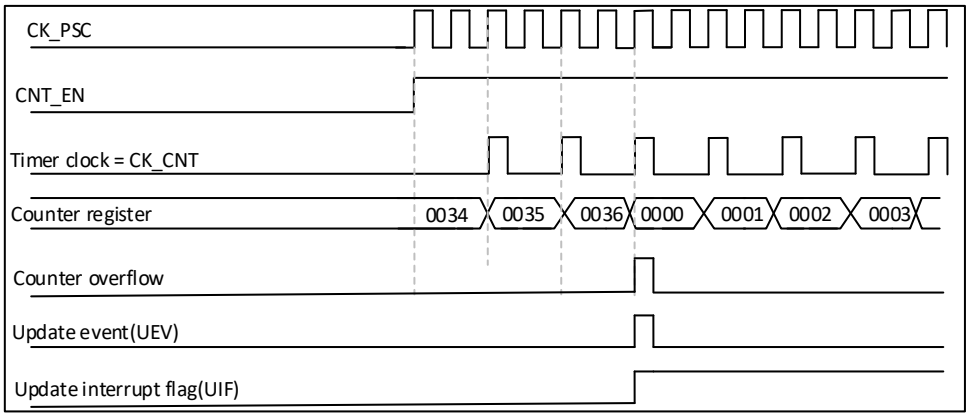


Figure 20-4 Counter timing diagram, internal clock divided by 2

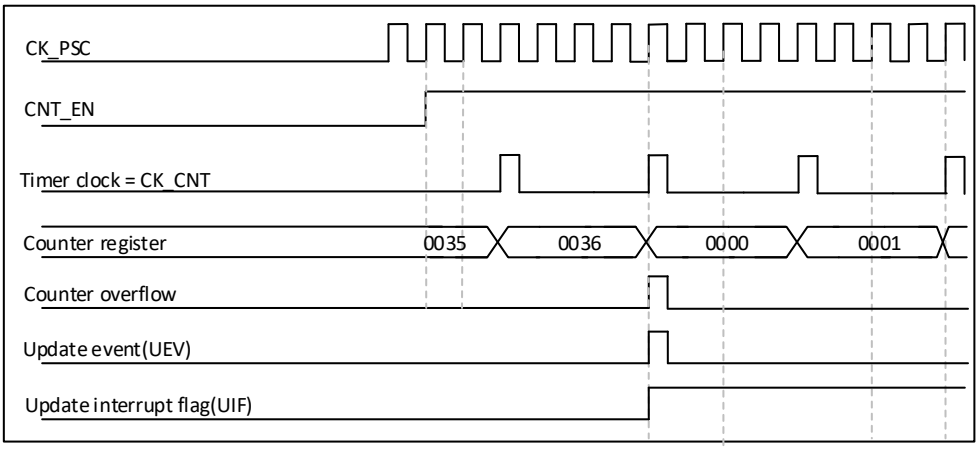


Figure 20-5 Counter timing diagram, internal clock divided by 4

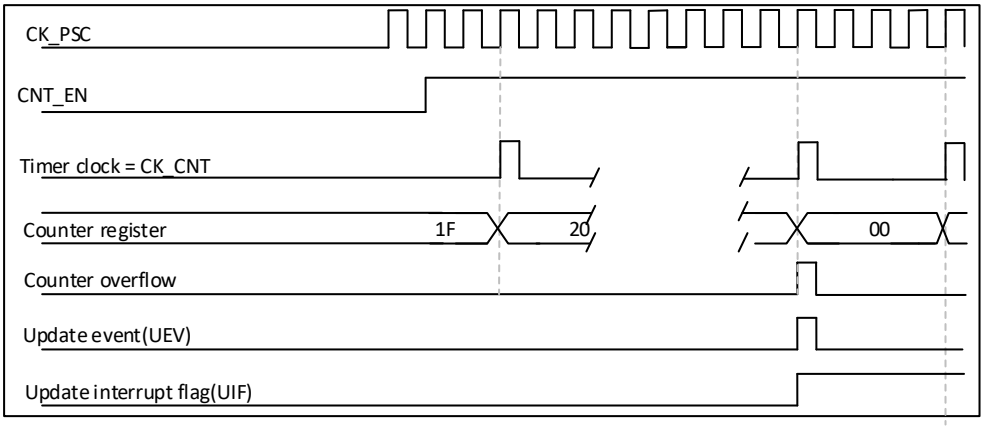


Figure 20-6 Counter timing diagram, internal clock divided by N

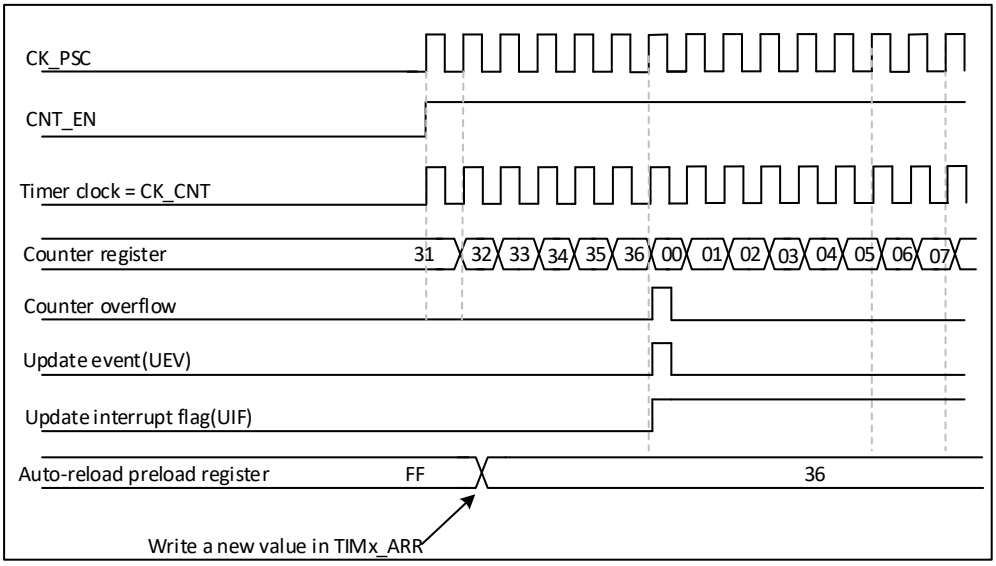


Figure 20-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

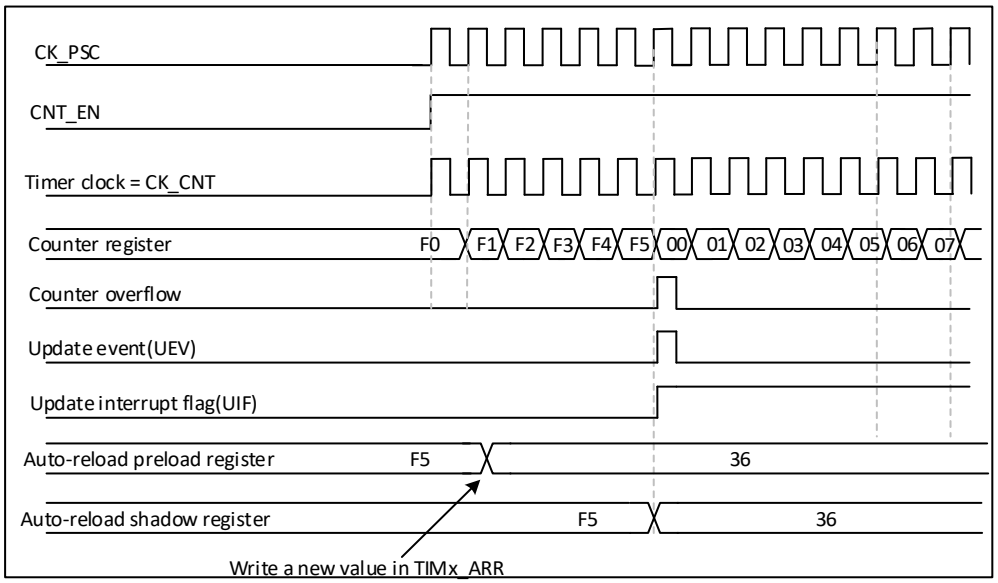


Figure 20-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

Downcounting mode

In down-count mode, the counter starts from the auto-loaded value (the contents of TIMx_ARR) and counts down to 0, then restarts from the auto-loaded value and generates a count underflow event. Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. Even then, when an update event should occur, the counter will restart counting from the current autoload value while the counter inside the prescaler is cleared '0' (but the prescaling factor remains unchanged).

In addition, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit is not set (i.e., no interrupt or

DMA request will be generated), in order to avoid clearing the counter when a capture event occurs, and generating both an update and a capture interrupt.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note: The autoload value is updated before the counter is reloaded, so the next cycle will be the expected value.

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

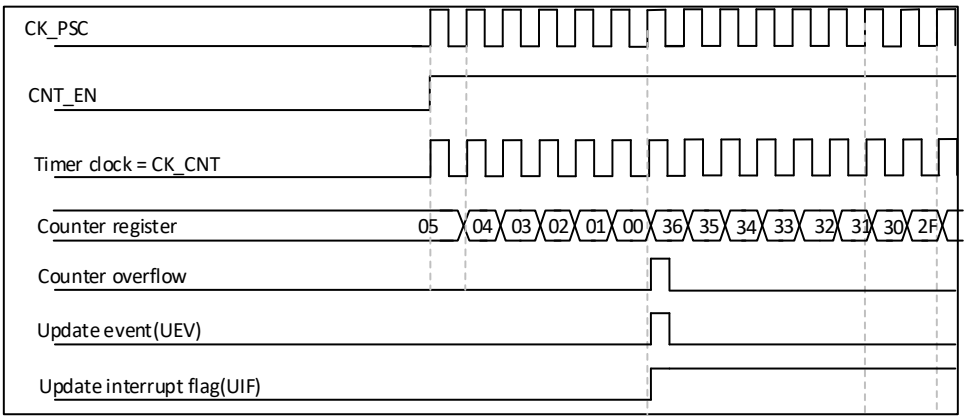


Figure 20-9 Counter timing diagram, internal clock divided by 1

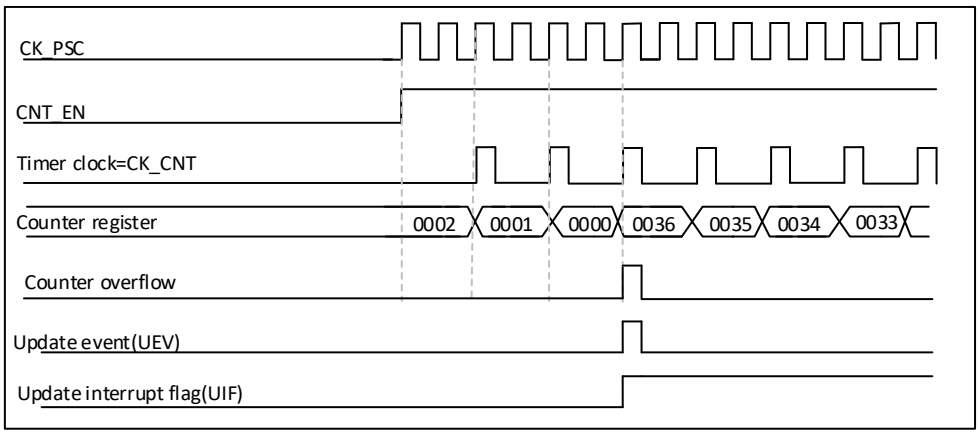


Figure 20-10 Counter timing diagram, internal clock divided by 2

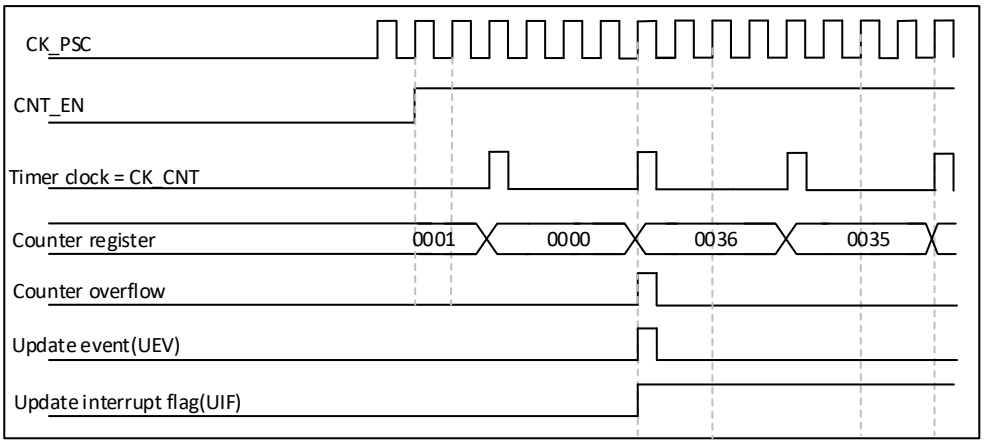


Figure 20-11 Counter timing diagram, internal clock divided by 4

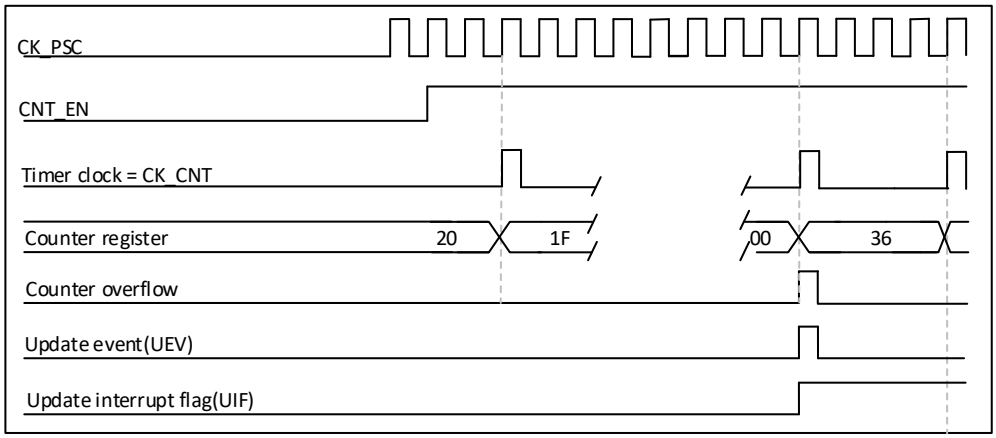


Figure 20-12 Counter timing diagram, internal clock divided by N

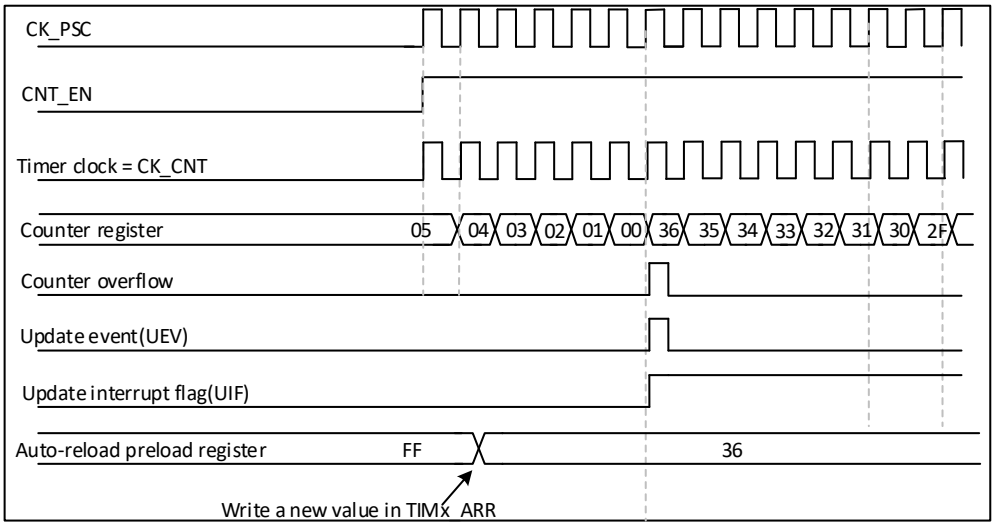


Figure 20-13 Counter timing diagram, update event when repetition counter is not used

Central alignment mode (up/down count)

In central alignment mode, the counter starts counting from 0 to the auto-loaded value (TIMx_ARR register)minus1, generating a counter overflow event, and then counts down to 1 and generating a counter underflow event; Then re-count from 0 again.

The center alignment mode can be obtained by configuring the CMS bit in the TIMx_CR1 register not to be '00'. The output comparison flag bit with the channel configured as output mode is set during the following counting processes: when counting down (center alignment mode 1, CMS = '01'), when counting up (center alignment mode 2, CMS = '10'), and when counting up and down (center alignment mode 3, CMS = '11').

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event. The counter then starts counting again from 0, and the prescaler internal counter also starts counting again from 0.

Setting the UDIS bit in the TIMx_CR1 register can disable the update event. This is to avoid updating the shadow registers while writing new values in the preload registers. Although no update event will occur until the UDIS bit is cleared to 0, the counter will continue to count up or down based on the value of the current auto-reload.

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), by setting the UG bit, an update event UEV will be generated but the UIF flag will not be set (thus no interrupts and DMA requests are generated), in order to avoid clearing the counter when a capture event occurs, while generating both an update and a capture interrupt.

When an update event occurs, all registers are updated and (according to the setting of the URS bit) an update flag bit (UIF bit in the TIMx_SR register) is also set:

- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIMx_ARR register). Note: If an update occurs due to a counter overflow, the automatic reload will be updated before the counter is reloaded, so the next cycle will be the expected value (the counter is loaded as a new value).

Here are some examples of how the counter operates at different clock frequencies:

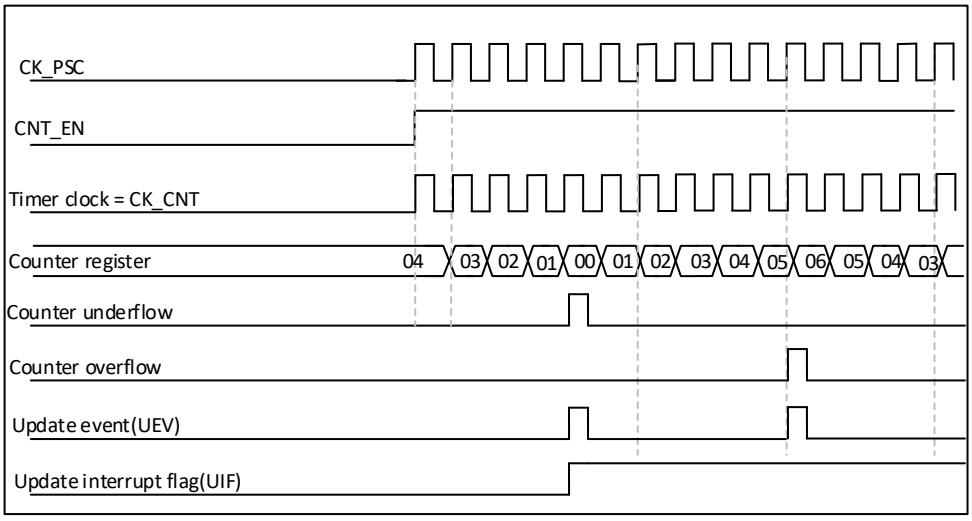


Figure 20-14 Counter timing diagram, internal clock division factor is 1, TIMx_ARR = 0x6

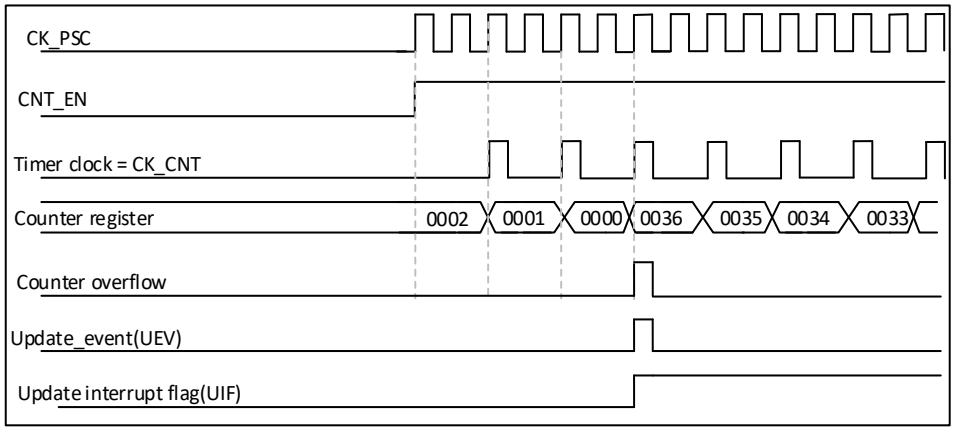


Figure 20-15 Counter timing diagram, internal clock divided by 2

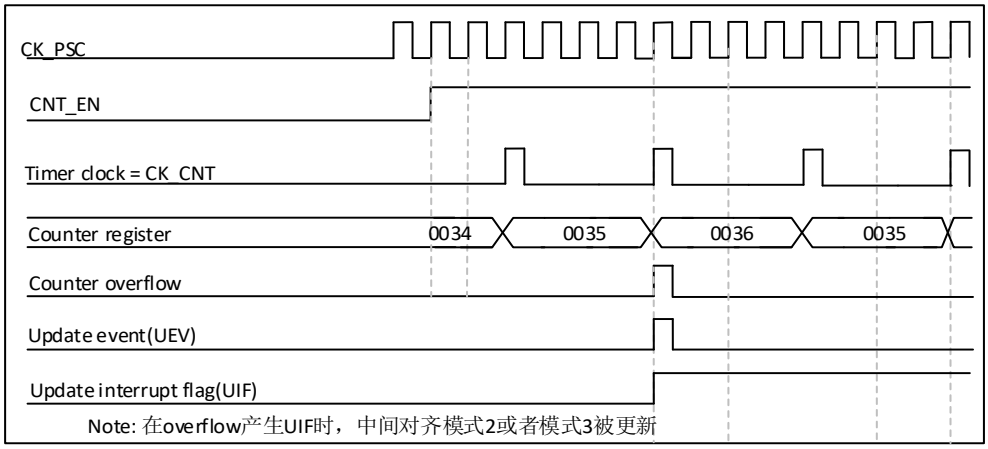


Figure 20-16 Counter timing diagram, internal clock division factor is 4, TIMx_ARR = 0x36

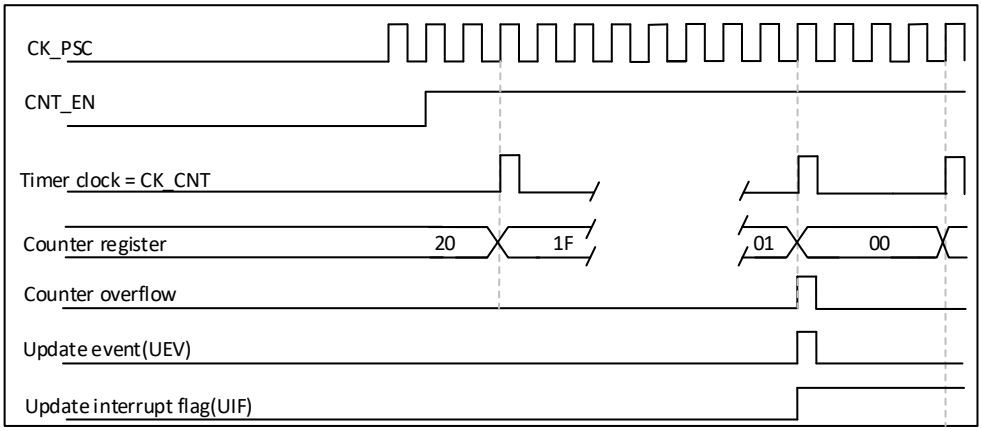


Figure 20-17 Counter timing diagram, internal clock divided by N

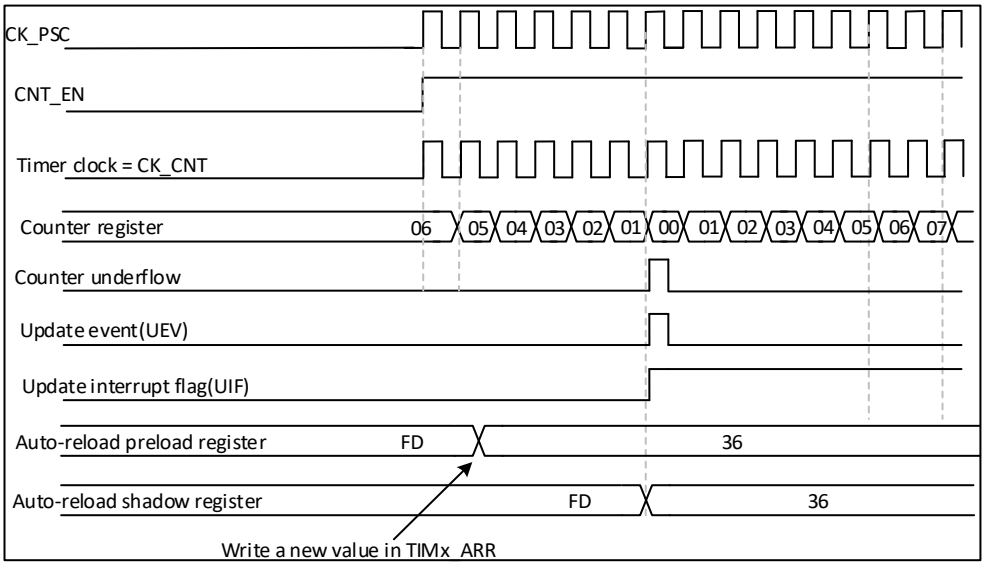


Figure 20-18 Counter timing diagram, update event with ARPE=1 (counter underflow)

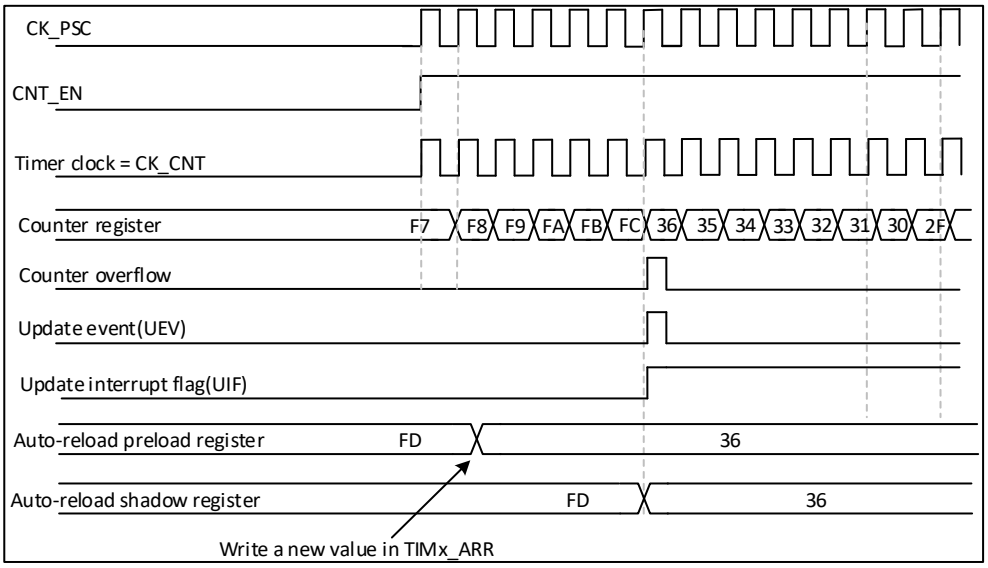


Figure 20-19 Counter Timing Chart, Update Event (Counter Overflow) at ARPE = 1

20.2.3 External trigger input

The timer external trigger input tim_etr can be used to:

- External clock
- From the trigger of the pattern
- As a PWM reset input in periodic current regulation

20.2.4 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pins (TI1 and TI2)
- External clock mode2: external trigger input ETR
- Internal Trigger Input (ITRx): Uses one timer as a prescaler for the other. For example, one timer Timer1 may be configured as a prescaler for another timer Timer2.

■ Encoder Mode

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 0000), the CEN, DIR (TIMx_CR1 register), and UG bits (TIMx_EGR register) are de facto control bits and can only be modified by software (the UG bits are still automatically cleared). As long as the CEN bit is written as' 1 ', the clock of the prescaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

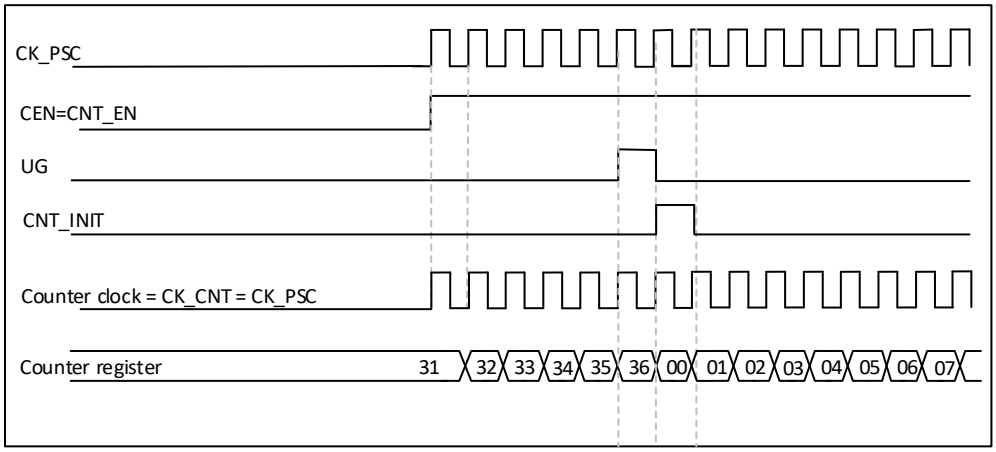


Figure 20-20 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

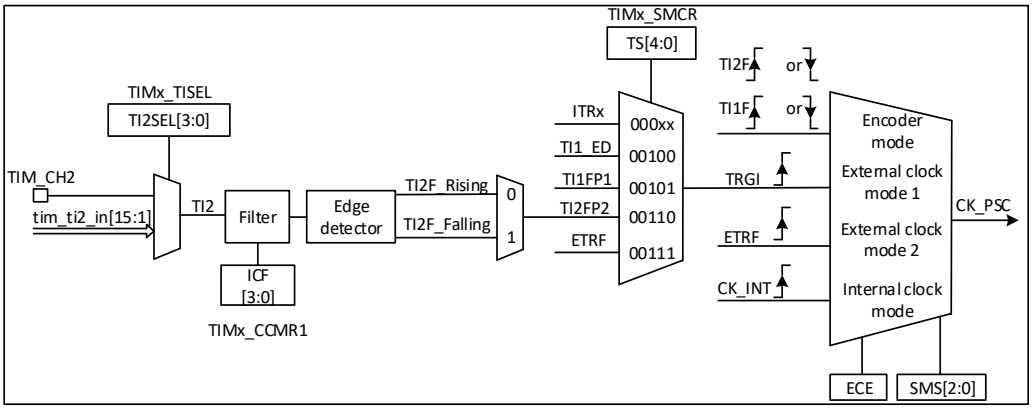


Figure 20-21 TI2 Example of external clock connection

For example, to configure the counter to count up on the rising edge of the TI2 input, use the following steps:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = ‘01’ in the TIMx_CCMR1 register.

2. Configure IC2F [3: 0] of the TIMx_CCMR1 register and select the input filter bandwidth (if no filter is needed, keep IC2F = 0000);

3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.

4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=00110 in the TIMx_SMCR register;

6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the synchronization circuit at the input of TI2.

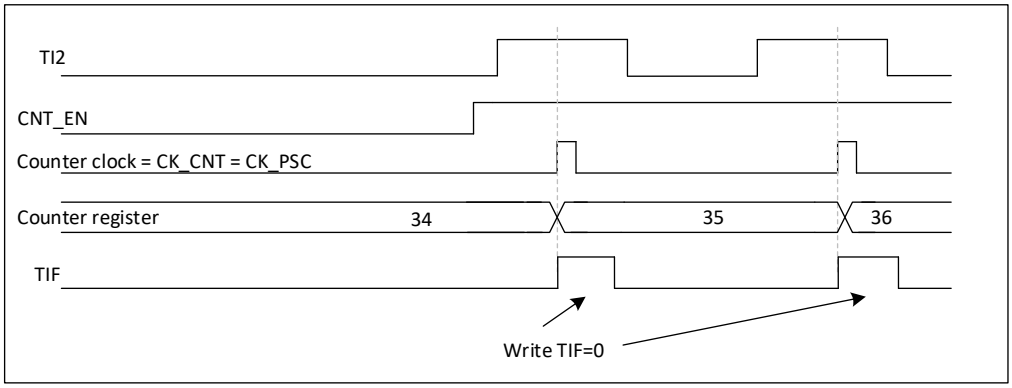


Figure 20-22 Control circuit in external clock mode 1

External clock source mode 2

This mode is selected by making ECE = 1 in the TIMx_SMCR register, and the counter can externally trigger each rising or falling edge of the ETR to count.

The following figure is a block diagram of the external trigger input:

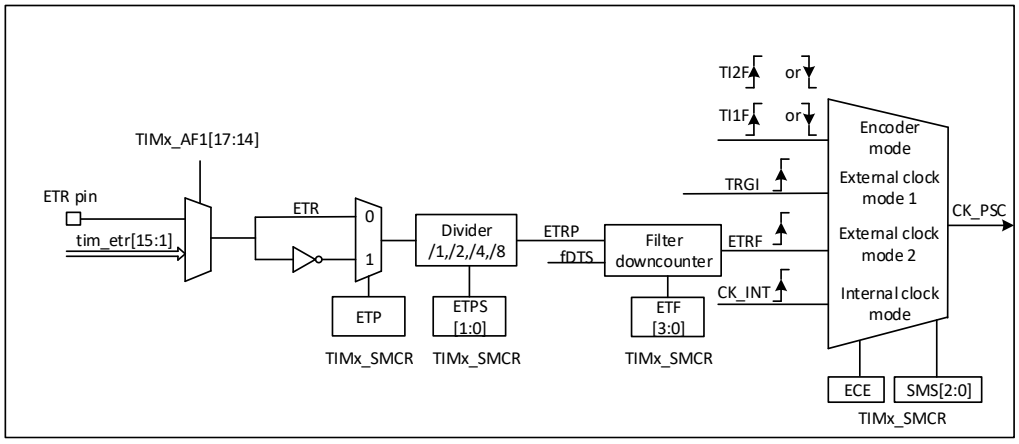


Figure 20-23 External trigger input block diagram

For example, to configure an up-counter that counts every 2 rising edges under ETR, use the following steps:

1. No filter is needed in this example, set ETF [3: 0] = 0000 in the TIMx_SMCR register;

2. Set the prescaler and set ETPS [1: 0] = 01 in the TIMx_SMCR register;

3. Select the rising edge of the ETR input terminal and set ETP = 0 in the TIMx_SMCR register;

4. Turn on external clock mode 2 and write ECE = 1 in the TIMx_SMCR register;

5. Start counter, write CEN = 1 in TIMx_CR1 register;

The counter counts at every 2 ETR rising edges.

The delay between the rising edge of the ETR and the actual clock of the counter depends on the synchronization circuit of the ETRP signal.

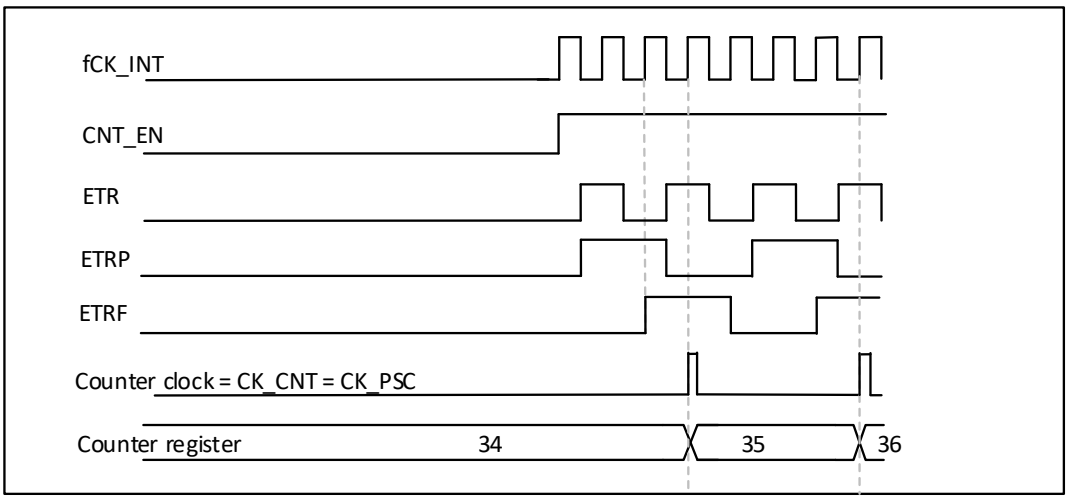


Figure 20-24 Control circuit in external clock mode 2

20.2.5 Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures are capture/compare channel overviews.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge monitor with polarity selection then generates a signal (TIxFPx) that can be triggered as an input from the slave mode controller or as a capture control. It is prescaled before the capture register (ICxPS).

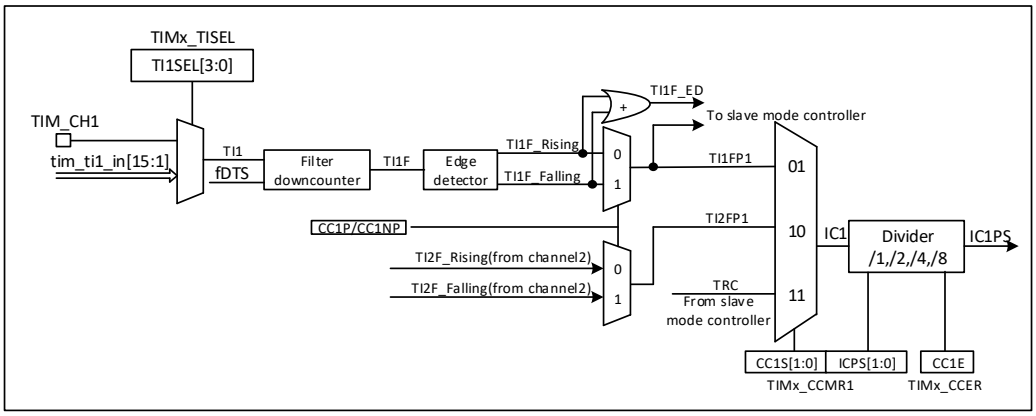


Figure 20-25 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

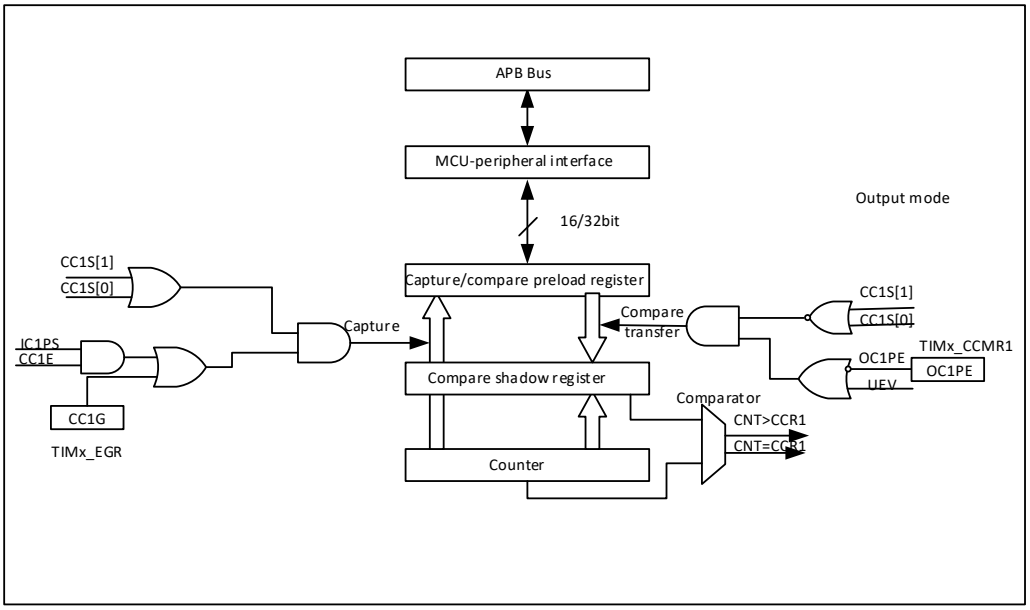


Figure 20-26 Capture/Compare the main circuit of channel 1

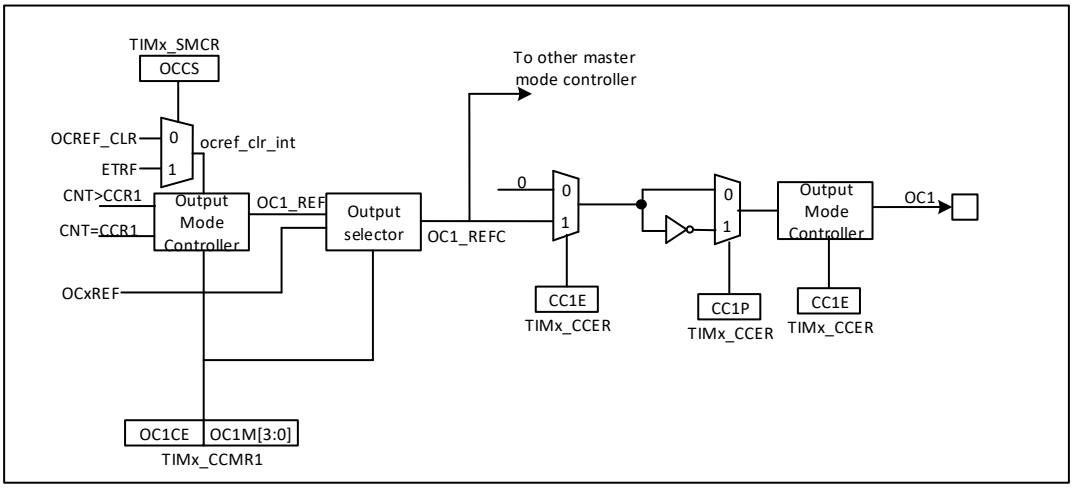


Figure 20-27 Output stage of capture/compare channel (channel 1 to 4)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

20.2.6 Input capture mode:

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched into the capture/compare register (TIMx_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx_SR register) is set to 1, and if an interrupt or DMA operation is enabled, an interrupt or DMA request will be generated. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (TIMx_SR register) is set to 1. The CCxIF can be cleared by writing CCxIF = 0, or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select valid input: TIMx_CCMR1 must be connected to the TI1 input, so CC1S = 01 written in the TIMx_CCMR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (set to rising edge) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If desired, the associated interrupt request may be allowed by setting the CC1IE bit in the TIMx_DIER register, or the DMA request may be allowed by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- The CC1IF flag bit is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

20.2.7 PWM input mode

This mode is a special case of the input capture mode, and the rest of the operation is the same as the input capture mode except for the following differences:

- Both ICx signals are mapped to the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, the user can measure the cycle (TIMx_CCR1 register) and the duty cycle (TIMx_CCR2 register) of the PWM signal input to TI1 as follows

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

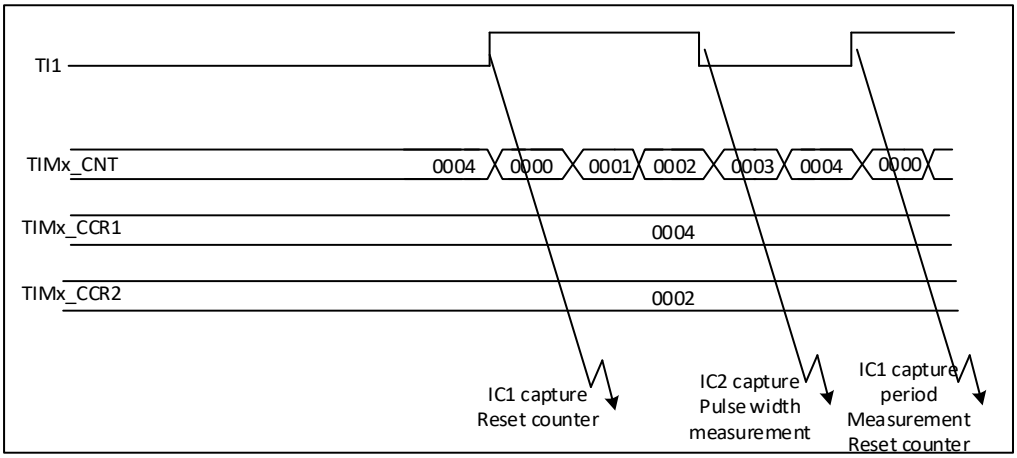


Figure 20-28 PWM input mode timing

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use the TIMx_CH1/TIMx_CH2 signal.

20.2.8 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

Set the corresponding OCxM = 0101 in the TIMx_CCMRx register to force the output comparison signal (OCxREF/OCx) to an active state. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIMx_CCMRx register. Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

20.2.9 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has expired. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=0000), be set active (OCxM=0001), be set inactive (OCxM=0010) or can toggle (OCxM=0011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- If the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register selects the DMA request function), a DMA request is generated.

You can select whether the TIMx_CCRx register needs to use a preload register by configuring the OCxPE bit in TIMx_CCMRx.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

— Write OCxM = 0011 to toggle OCx output pin when CNT matches CCRx

— Write OCxPE = 0 to disable preload register

— Write CCxP = 0 to select active high polarity

— Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated by software at any time to control the output waveform, provided the preload register is not used (OCxPE = '0', otherwise the shadow register of TIMx_CCRx can only be updated when the next update event occurs). An example is given in the figure below.

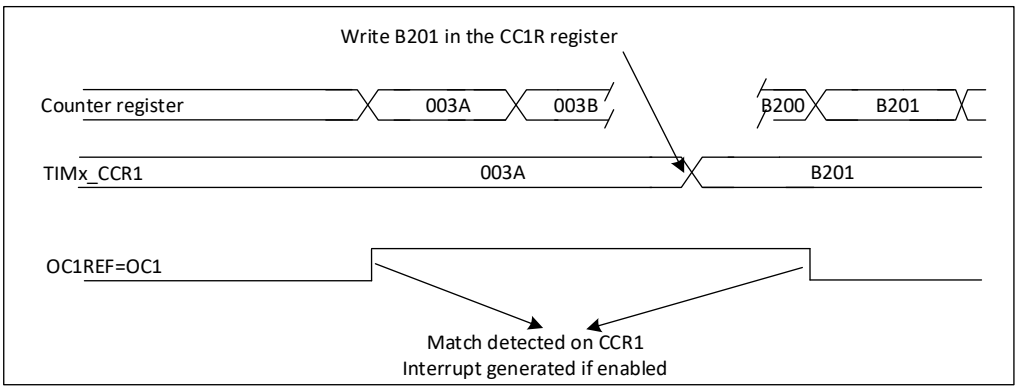


Figure 20-29 Output comparison mode, flip OC1

20.2.10 PWM mode

The pulse width modulation mode may produce a signal whose frequency is determined by the TIMx_ARR register and whose duty cycle is determined by the TIMx_CCRx register.

The OCxM bit in the TIMx_CCMRx register is written to '0110' (PWM mode 1) or '0111' (PWM mode 2), and each OCx output channel can be independently set to generate a PWM. The corresponding preload register must be enabled by setting the OCxPE bit of the TIMx_CCMRx register, and finally the ARPE bit of the TIMx_CR1 register to enable the preload register that is automatically reloaded (in up-count or centrosymmetric mode).

The preload register can only be transferred to the shadow register when an update event occurs, so the user must initialize all registers by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. The output of OCx is enabled to be controlled by (TIMx_CCER) CCxE, CCxNE. Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

- Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the comparison value in TIMx_CCRx is greater than the auto-reload value (TIMx_ARR), OCxREF remains' 1 '. If the comparison value is 0, then OCxREF remains' 0 '. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

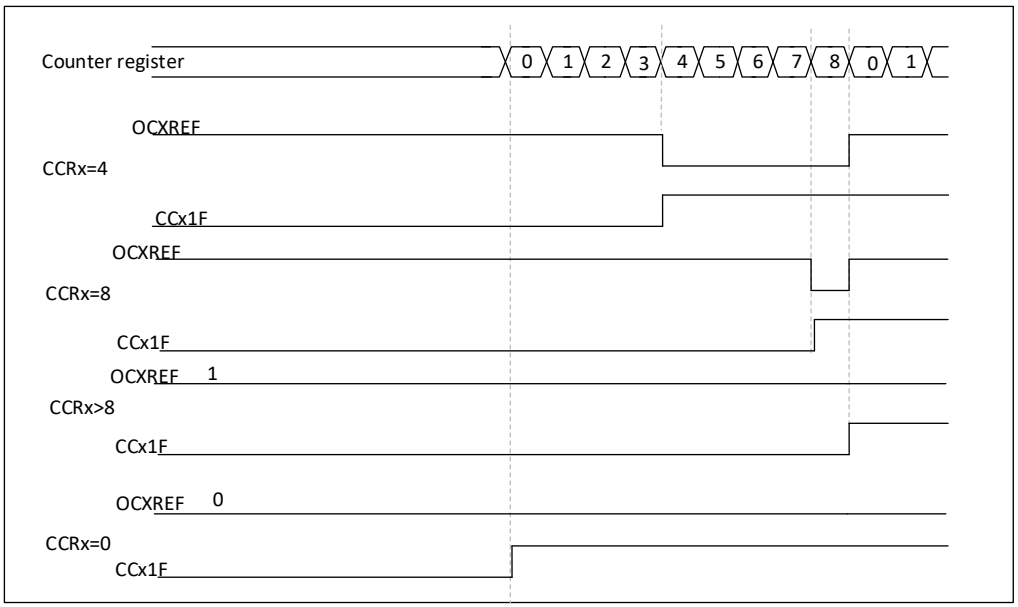


Figure 20-30 Edge-aligned PWM waveforms (ARR=8)

- Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long asTIMx_CNT > TIMx_CCRx else it becomes high. If the comparison value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, OCxREF remains' 1 '. 0% PWM is not possible in this mode.

PWM Central Alignment Mode

The center alignment mode is when the CMS bit in the TIMx_CR1 register is not '00' (all other configurations of the CMS bit have the same effect on the OCxREF/OCx signal). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software.

The following figure gives some examples of centrally aligned PWM waveforms

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- CMS = 01 for the TIMx_CR1 register, in center alignment mode 1, the compare flag is set when the counter counts down.

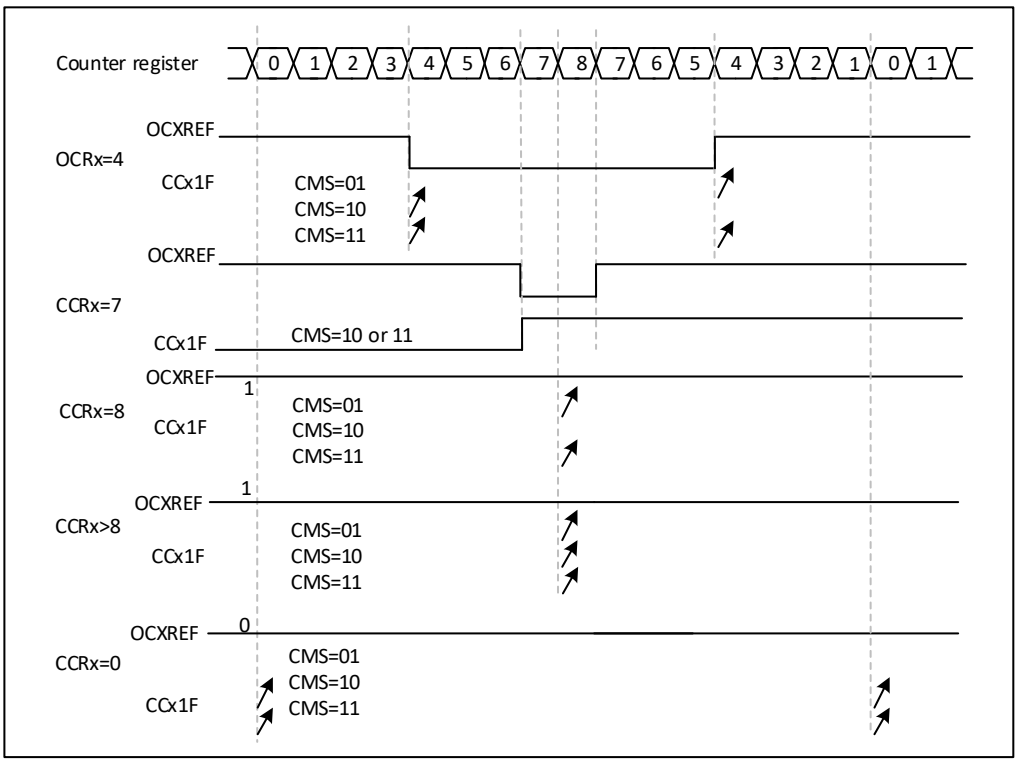


Figure 20-31 Centre-aligned PWM waveform (APR = 8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Also, do not modify the DIR and CMS bits simultaneously through software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:

-If the value of the write counter is greater than the value of the auto-reload (TIMx_CNT > TIMx_ARR), the direction is not updated.

For example, if the counter was counting up, it continues to count up.

-If a value of 0 or TIMx_ARR is written to the counter, the direction is updated, but no update event UEV is generated.

- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

Jitter mode

The DITHEN bit of the TIMx_CR1 register may be enabled to turn on the jitter mode to improve the effective resolution of the PWM mode. Can be applied to CCR (improved duty cycle resolution) and ARR (improved resolution of PWM frequency)

The principle of operation is to make the actual CCR (or ARR) value change slightly (with or without increasing a timer cycle) over 16 consecutive PWM cycles, and how the change can be set by predefined settings. When calculating the average duty cycle, you can achieve a 16-fold increase in resolution. The figure below shows the jitter principle of four consecutive PWM periods

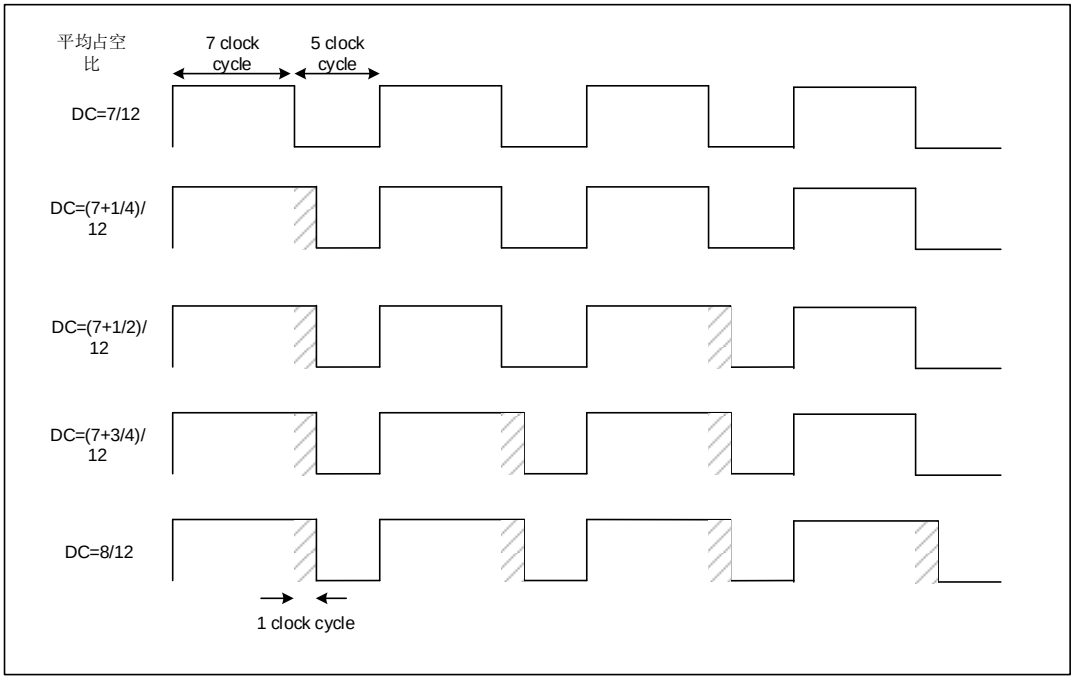


Figure 20-32 jitter principle

When jitter mode is enabled, the value of the register will automatically change as follows:

- The lowest 4 bits is the encoding for improving resolution (fractional part)
- The high bit is shifted left to 19: 4 for encoding as the base value (integer part)

Note: When dither mode is on or off, the values of ARR and CCR are automatically updated (for example, ARR = 0x05 when DITHEN = 0, then ARR = 0x50 when DITHEN = 1), the following steps must be followed when resetting the DITHEN bit

1. CEN and ARPE bits must be reset
2. The ARR [3: 0] bit must be reset
3. The DITHEN bit must be reset
4. The CCIF flag must be cleared
5. CEN bit can be set (ARPE = 1 can be set)

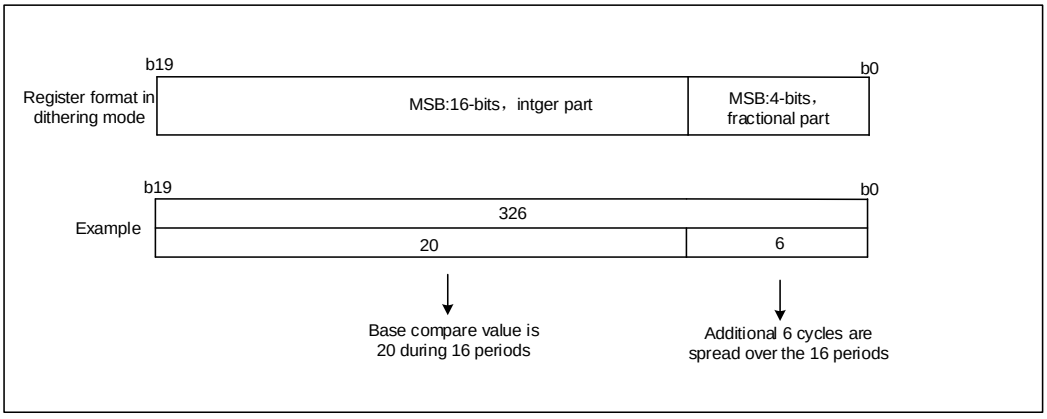


Figure 20-33 Data mapping and register coding in dither mode

The minimum frequency is calculated as follows:

According to resolution = $\frac{\text{定时器频率}}{\text{PWM 频率}}$ push out: minimum PWM frequency = $\frac{\text{定时器频率}}{\text{最大分辨率}}$

In jitter-free mode: Minimum PWM frequency = $\frac{\text{定时器频率}}{65536}$

When there is jitter mode: minimum PWM frequency = $\frac{\text{定时器频率}}{65535+\frac{15}{16}}$

Note: The maximum values of TIMx_arr and TIMxCCRy in jitter mode are limited to 0xFFFEF (corresponding to 65534 in the integer part and 15 in the jitter part). Exceeding 0xFFFEF, there will be an overflow situation, that is, arr = 0XFFFF+1 = 0.

The figure below shows that in jitter mode, the resolution of the PWM can be improved regardless of the PWM frequency

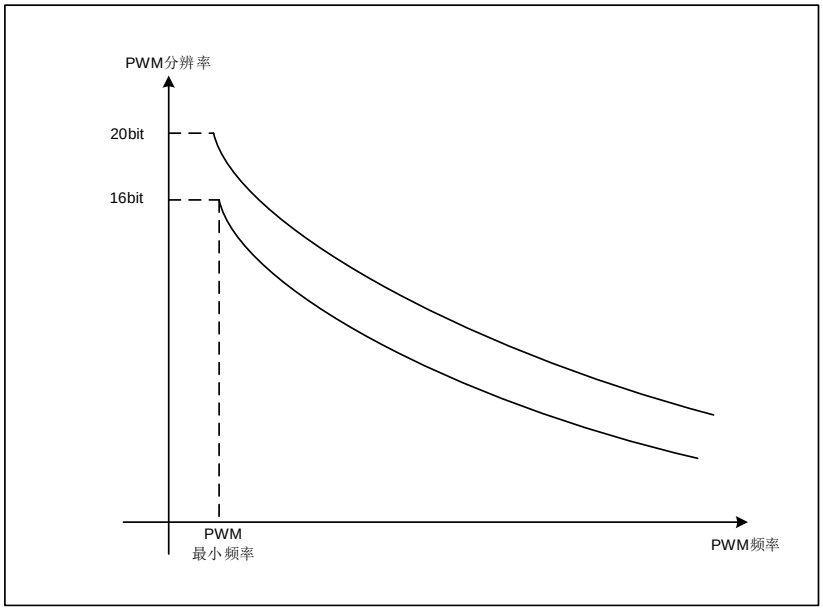


Figure 20-34 PWM resolution vs frequency

The duty cycle and cycle changes in 16 consecutive cycles are as follows:

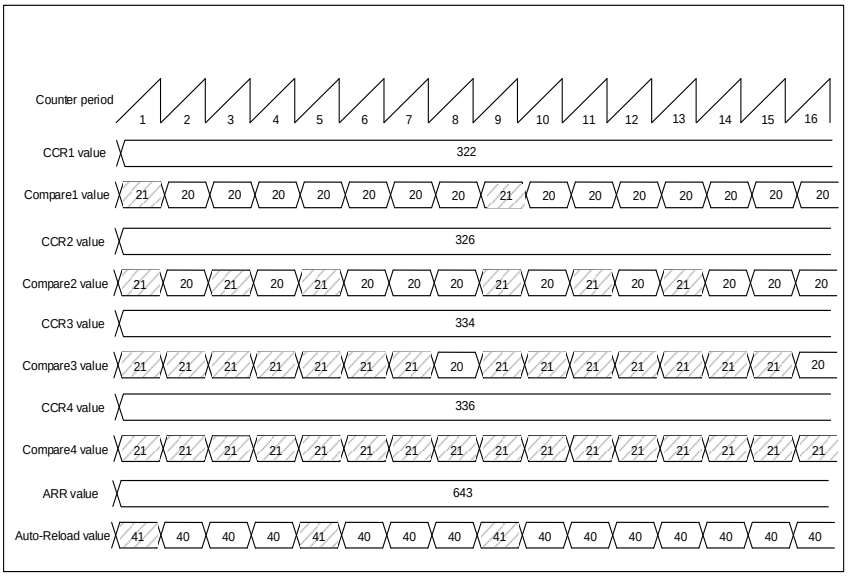


Figure 20-35 PWM Jitter Mode

The auto-reload and compare value increments are distributed in a specific pattern as described in the table below, with the incremental distribution of the jitter sequence as uniformly as possible, minimizing overall ripple.

Table 20-1 CCR and ARR Register Jitter Changes

LSB value	PWM Cycle															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000																
0001	+1															
0010	+1								+1							
0011	+1				+1				+1							
0100	+1				+1				+1				+1			
0101	+1		+1		+1				+1				+1			
0110	+1		+1		+1				+1		+1		+1			
0111	+1		+1		+1		+1		+1		+1		+1			
1000	+1		+1		+1		+1		+1		+1		+1		+1	
1001	+1	+1	+1		+1		+1		+1		+1		+1		+1	
1010	+1	+1	+1		+1		+1		+1	+1	+1		+1		+1	
1011	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1		+1	
1100	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1	+1	+1	
1101	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1		+1	+1	+1	
1110	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1	+1	+1	+1	+1	
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	

The dither mode can also be applied in the intermediate alignment PWM mode (CMS! = 00). The figure below considers the case where the dither mode is applied to 8 consecutive cycles when counting up and down.

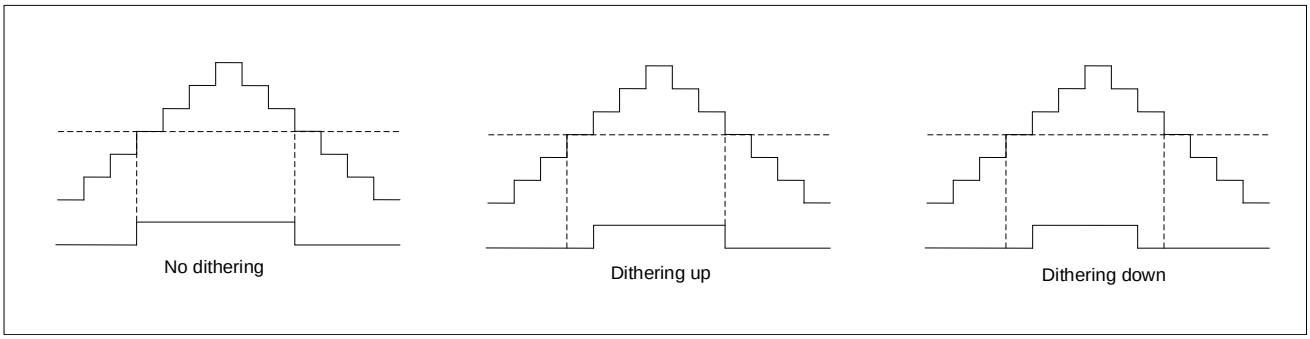


Figure 20-36 Effect of jitter on duty cycle in aligned PWM mode in middle of the figure

Table 20-2 Change of CCR Register Jitter in Middle Aligned PWM Mode

LSB value	PWM Cycle															
	1		2		3		4		5		6		7		8	
	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN	UP	DN
0000																
0001	+1															
0010	+1								+1							
0011	+1				+1				+1							
0100	+1				+1				+1				+1			
0101	+1		+1		+1				+1				+1			
0110	+1		+1		+1				+1		+1		+1			
0111	+1		+1		+1		+1		+1		+1		+1			
1000	+1		+1		+1		+1		+1		+1		+1		+1	
1001	+1	+1	+1		+1		+1		+1		+1		+1		+1	
1010	+1	+1	+1		+1		+1		+1	+1	+1		+1		+1	
1011	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1		+1	
1100	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1	+1	+1	
1101	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1		+1	+1	+1	
1110	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1	+1	+1	+1	+1	
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	

20.2.11 Asymmetric PWM mode

The asymmetric mode allows two centrally aligned PWM signals to produce a programmable phase shift. When the frequency is determined by TIMx_ARR and the duty cycle and phase shift are determined by a pair of TIMx_CCRx registers. The asymmetric mode allows the generation of two center-aligned PWM signals using programmable phase shifts. One TIMx_CCRx register is controlled during the up-count phase and the other during the down-count phase so that the PWM can be adjusted every half PWM cycle

- The OC1REFC(orOC2REFC) is controlled by TIMx_CCR1 and TIMx_CCR2
- The OC3REFC(orOC4REFC) is controlled by TIMx_CCR3 and TIMx_CCR4

By writing 1110 to OCxM (asymmetric PWM mode 1) and 1111 to OCxM (asymmetric PWM mode 2), asymmetric PWM mode can be independently selected on dual channels (each pair of CCR registers controls one OCx output)

Note: For compatibility reasons, the OCxM [3: 0] field is divided into two parts, with the highest and lower 3 bits not contiguous.

When a given channel is used as an asymmetric channel, its adjacent channels (channel 1 adjacent to channel 2) can also be used. For example, if OC1REFC is generated by channel 1 (asymmetric PWM mode 1), channel 2 may also output OC2REF, or OC2REFC through asymmetric PWM mode 1

The figure below represents an example of using asymmetric PWM mode to generate signals (channels 1 through 4 are configured as asymmetric PWM mode 2), which, together with the dead time generator, controls a full-bridge DC-DC phase shift inverter.

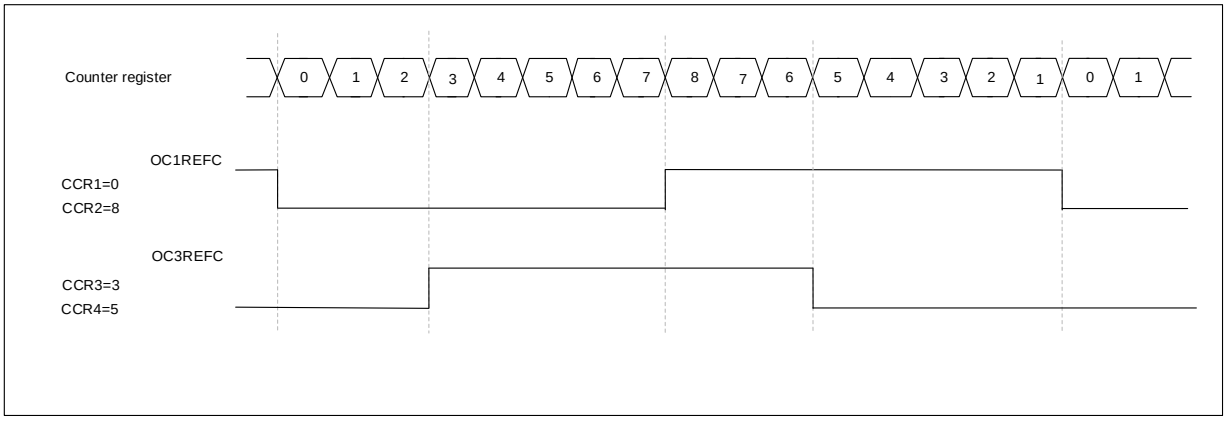


Figure 20-37 2 50% duty cycle phase shift PWM

20.2.12 Combined PWM mode

The combined PWM mode allows an edge or center aligned PWM signal to be generated by programmable delay and phase shift between pulses. The frequency is determined by the ARR and the duty cycle and delay are determined by the two CCRs. The resulting signal OCxREFC is generated by AND/OR logic between two reference PWMs

- The OC1REFC(orOC2REFC) is controlled by TIMx_CCR1 and TIMx_CCR2
- The OC3REFC(orOC4REFC) is controlled by TIMx_CCR3 and TIMx_CCR4

Asymmetric PWM modes can be independently selected on dual channels (each pair of CCR registers controls one OCx output) by writing to OCxM 1100 (combined PWM mode 1, logic OR output) and OCxM 1101 (combined PWM mode 2, logic AND output

When a given channel is used as a combined PWM channel, its complementary channels must be configured in opposite PWM modes (e.g., one for combined PWM mode 1 and one for combined PWM mode 2).

The following figure shows an example of a combined mode generation signal, including

- Channel 1 is combined PWM mode 2
- Channel 2 is PWM Mode 1
- Channel 3 is combined PWM mode 2
- Channel 4 is PWM Mode 1

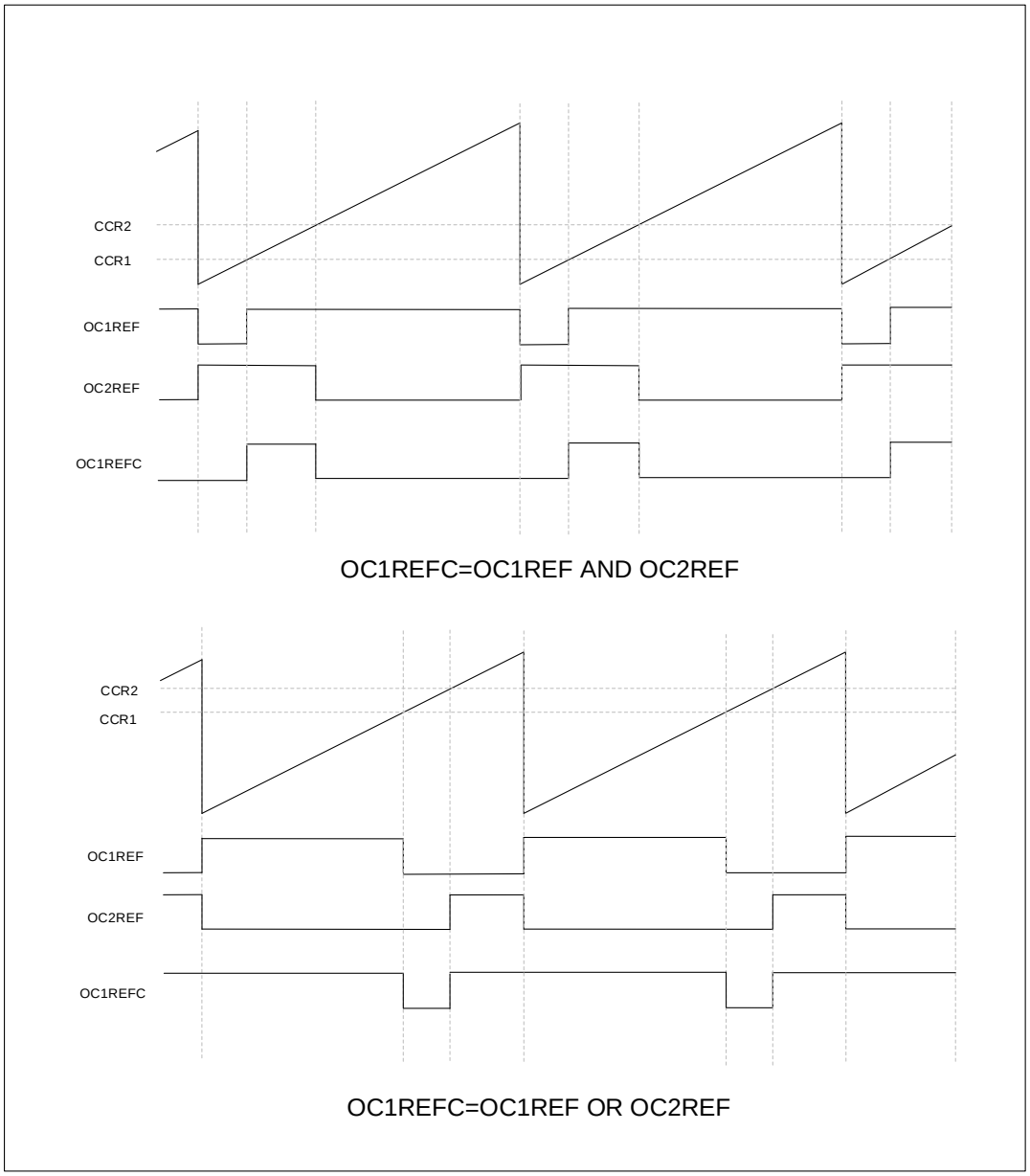


Figure 20-38 Combined PWM patterns for channel 1 and channel 3

20.2.13 Clearing the OCxREF signal on an external event

For a given channel, setting the corresponding OCxCE bit in the TIMx_CCMRx register to '1' enables the OCxREF signal to be pulled low with a high level at the OCREF_CLR_INT input, and the OCxREF

signal will remain low until the update event UEV generated by the next count overflow occurs. This function can only be used in output compare and PWM modes. It does not work in Forced mode.

While OCREF_CLR_INT can be selected between OCREF_CLR and ETRF (after ETR filtering) by configuring the OCCS bit in the TIMx_SMCR register.

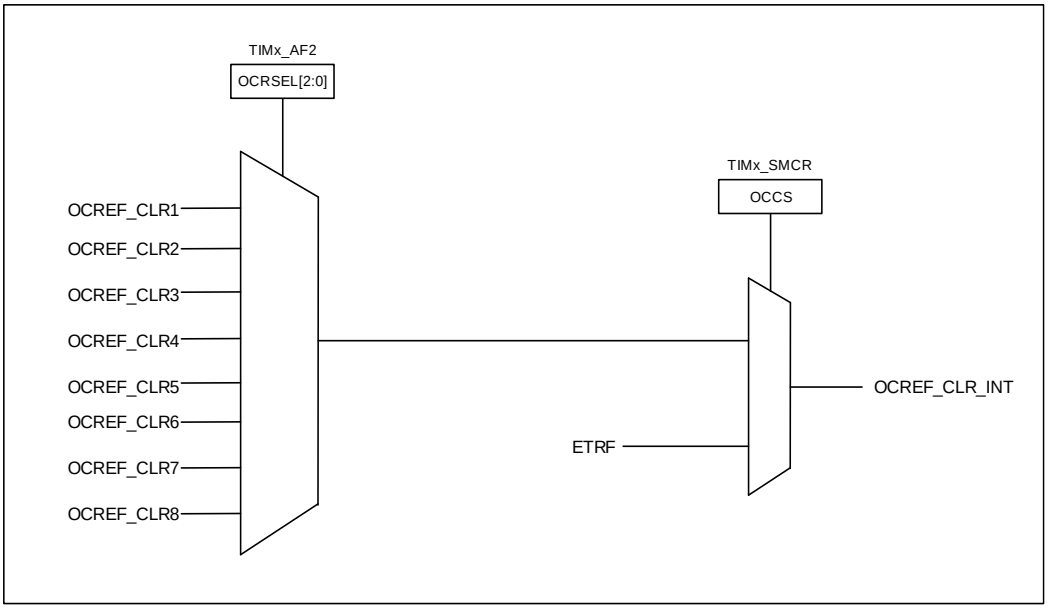


Figure 20-39 OCREF_CLR input selection

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, ETR must be configured as follows:

1. The external trigger prescaler should be kept off: bits ETPS[1:0] in the TIMx_SMCR register are cleared to 00.
2. The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
3. The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured as needed.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is placed in PWM1 mode.

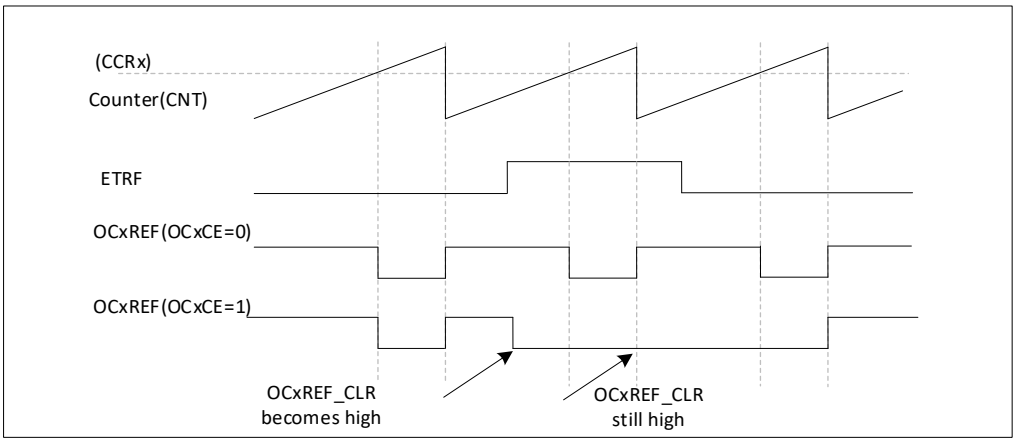


Figure 20-40 OCxREF for clearing TIMx

20.2.14 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV. A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up count mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx),
- Down counting mode: Counter CNT > CCRx.

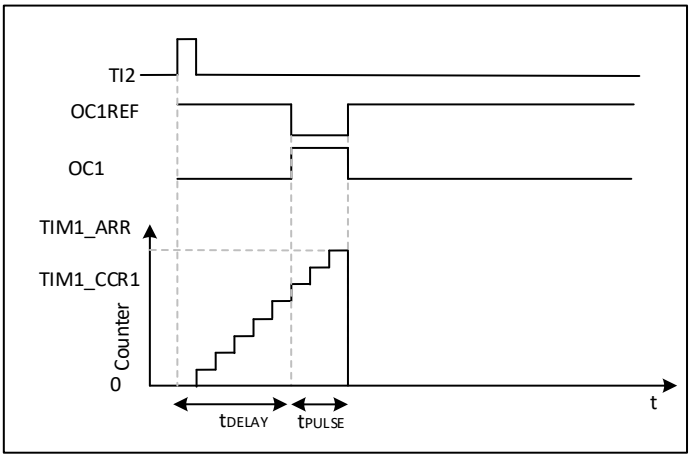


Figure 20-41 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Assume TI2FP2 as the trigger:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let’s say one want to build a waveform with a transition from ‘0’ to ‘1’ when a compare match occurs and a transition from ‘0’ to ‘1’ when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register

and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to ‘0’ in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0) When OPM bit in the TIMx_CR1 register is set to '0', so the repetitive mode is selected.

20.2.14.1 Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay tDELAY min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF (and OCx) is forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

20.2.15 Retriggerable Single Pulse Mode

This mode allows the counter to start in response to excitation and generate pulses of programmable length, which differs from the non-retriggerable single pulse mode described in the previous section as follows:

- The pulse starts as soon as the trigger occurs (no programmable delay).
- If a new trigger occurs before the pulse generated by the previous trigger is completed, the pulse is extended.

To use the retriggerable monopulse mode, the timer must be in slave mode, the bit SMS [3: 0] = "1000" in the TIMx_SMCR register (combined mode-reset + trigger), and the OCxM [3: 0] bit set to "1000" or "1001" (retriggerable OPM mode 1 or 2).

If the timer is configured in Up-counting mode, the corresponding CCRx must be set to 0 (the ARR register sets the pulse length). If the timer is configured in Down-counting mode, CCRx must be above or equal to ARR.

Note: For compatibility reasons, the OCxM [3: 0] and SMS [3: 0] bit fields are split into two parts, with the most significant bit not contiguous with the 3 least significant bit positions.

This mode must not be used in conjunction with the central alignment count mode. CMS [1: 0] = 00 must be in TIMx_CR1.

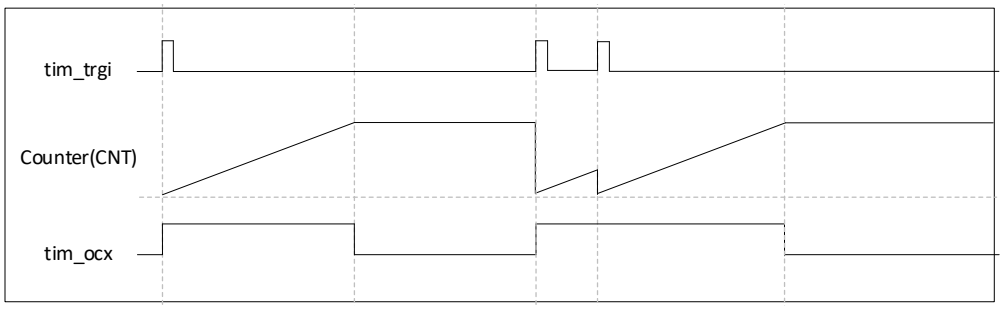


Figure 20-42 Example of a retriggerable monopulse mode

20.2.16 Comparison mode generation pulse

Pulses may be generated by comparing matching events. When the counter value is equal to the given comparison value, a programmable width pulse signal is generated for debugging and synchronization purposes.

This mode can be used with any slave mode, including encoder mode. It is only available for 3/4 channels in edge and center alignment counting mode. The pulse generator is unique and shared by these two channels, as shown in the following figure:

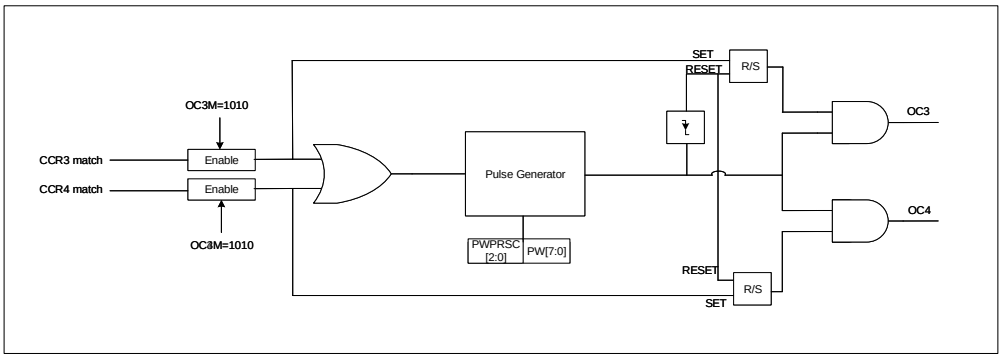


Figure 20-43 Pulse generator circuit

The following figure shows how pulses are generated in edge counting and encoder modes:

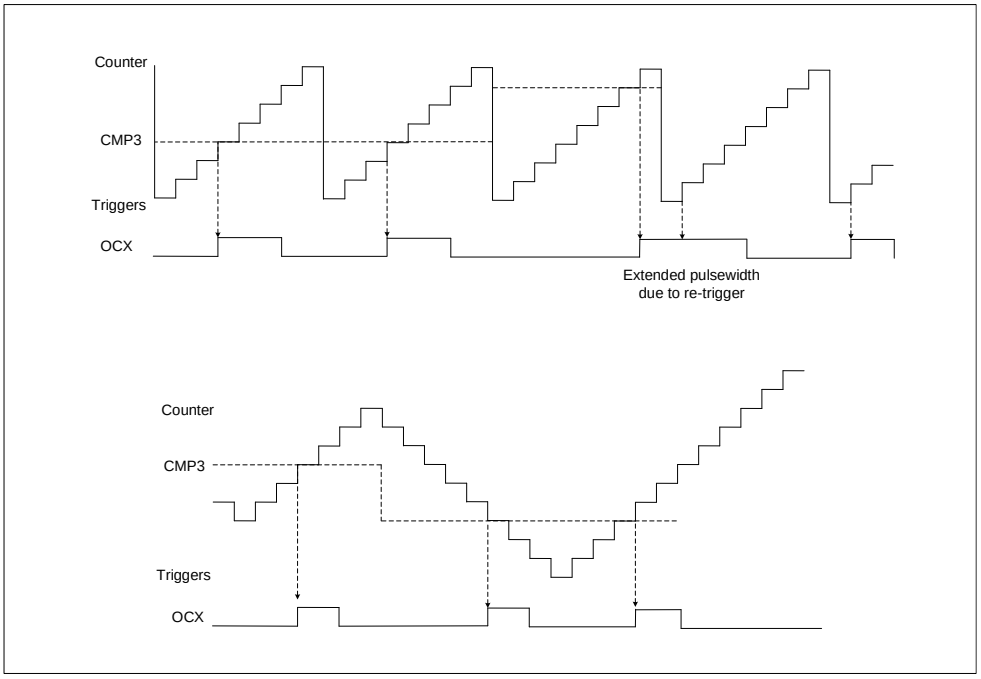


Figure 20-44 Generate pulses by comparing events in edge alignment and center alignment modes

The output comparison mode is selected by OC3M [3: 0] and OC4M [3: 0] in TIMx_CCMR2

The pulse width is programmed via the PW [7: 0] register, using the clock generated by clock prescalation via PWPRSC [2: 0]

t_{PW}= PW [7: 0] xt_{PWG}

t_{PWG}= xCK $2^{(PWPRSC[2:\ 0])}$_INT

The resolution and maximum value depend on the value of the prescaler

Pulse re-trigger: When the pulse is output normally, a new trigger comes, which will cause the pulse to be extended

Note: If both channels are enabled at the same time, the pulses output are independent as long as the trigger on one channel does not overlap with the pulses generated by the other channel. If the two triggers overlap, the trigger 1 generation pulse will be extended (will be re-triggered), and the last trigger generation pulse width will be correct.

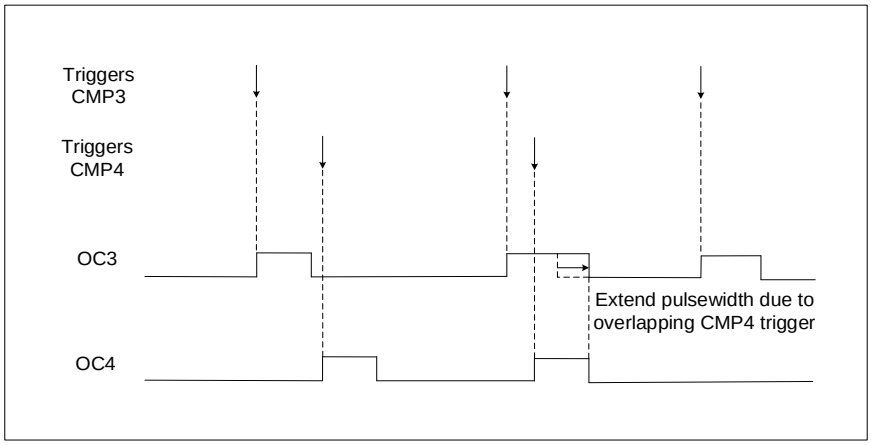


Figure 20-45 Generate pulses by comparing events in edge alignment and center alignment modes

20.2.17 Encoder interface mode

20.2.17.1 Quadrature encoder

To select Encoder Interface mode: write SMS='0001' in the TIMx_SMCR register if the counter is counting on TI1 edges only, SMS=0010 if it is counting on TI2 edges only and SMS=0011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, you can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Referring to the table below, assuming that the counter has been started (CEN = 1 in the TIMx_CR1 register), the counter is driven by each valid jump on TI1FP1 or TI2FP2. TI1FP1=TI1 if not filtered and not inverted,TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit inTIMx_CR1 register written to ‘1’). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIMx_ARR must be configured before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together. In this mode, the counter is modified automatically following the speed and the direction of the quadrature encoder and its content, therefore, always represents the encoder’s position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 20-3 Relationship between counting direction and encoder signal (CC1P = CC2P = 0)

Effective edge	SMS [3: 0]	Reverse signal level (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
			Rising	Falling	Rising	Falling
Count only at TI1 single edge x1 mode	1110	High	Minus	Add	-	-
		Low	-	-	-	-
Count only at TI2 single edge x1 mode	1111	High	-	-	Add	Minus
		Low	-	-	-	-
Count only at TI1 double edge x2 mode	0001	High	Minus	Add	-	-
		Low	Add	Minus	-	-
Count only at TI2 double edge x2 mode	0010	High	-	-	Add	Minus
		Low	-	-	Minus	Add
Counting at TI1 and TI2 double edges x4 mode	0011	High	Minus	Add	Add	Minus
		Low	Add	Minus	Minus	Add

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S = '01' (TIMx_CCMR1 register, TI1FP1 mapped to IC1)
- CC2S = '01' (TIMx_CCMR2 register, TI2FP2 mapped to IC2)
- CC1P = '0' (TIMx_CCER register, TI1FP1 is not inverted, TI1FP1 = TI1)
- CC2P = '0' (TIMx_CCER register, TI2FP2 is not inverted, TI2FP2 = TI2)
- SMS = '011' (TIMx_SMCR register, all inputs are valid on rising and falling edges).
- CEN = '1' (TIMx_CR1 register, counter enabled)

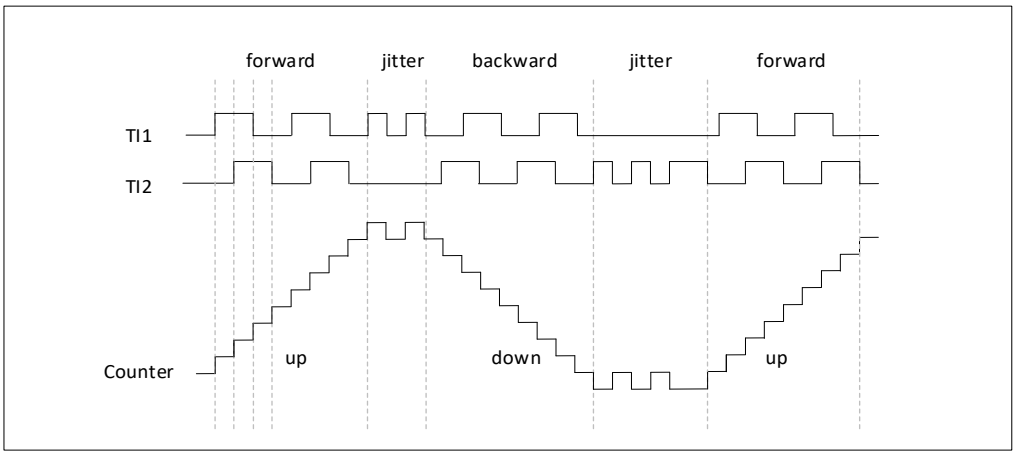


Figure 20-46 Example of counter operation in encoder interface mode

The following figure shows an operation example of the counter when the polarity of IC1FP1 is inverted (CC1P = '1', other configurations are the same as the above example)

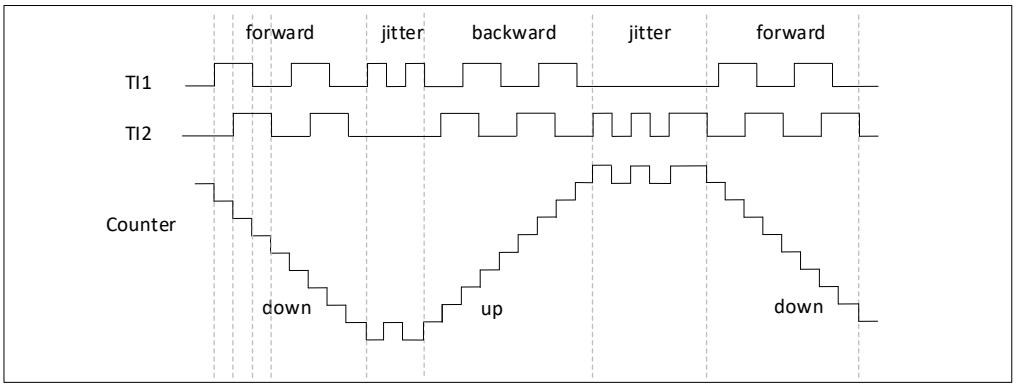


Figure 20-47 IC1FP1 Inverted Encoder Interface Mode Example

The figure below shows the count value when the steering turns over in different modes:

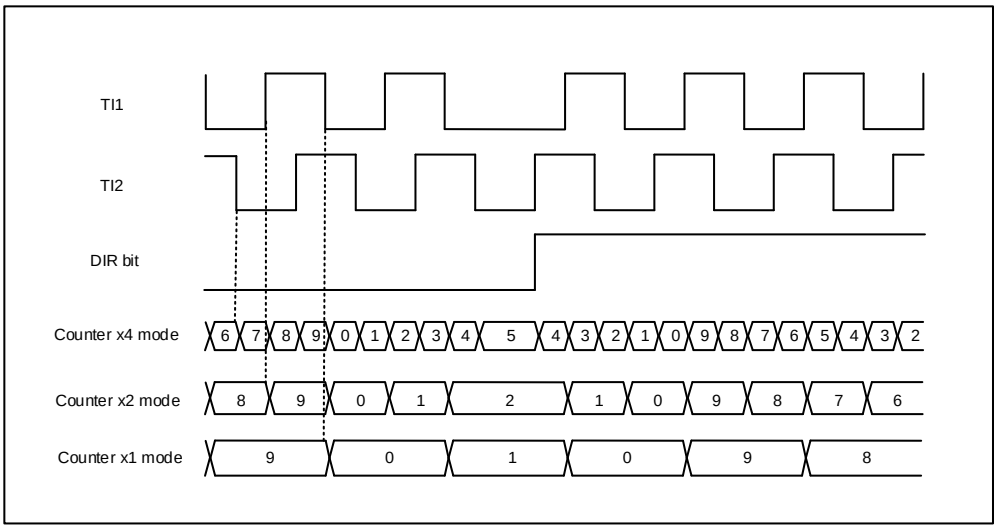


Figure 20-48 Quadrature Encoder Counting Mode

When the timer is configured in the encoder interface mode, information of the current position of the sensor may be provided. By configuring the second timer in the capture mode, the interval between the two encoder events can be measured, and dynamic information (speed, acceleration, deceleration) can be obtained. An encoder output indicating a mechanical zero may be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. If possible, you can latch the value of the counter to the third input capture register (the capture signal must be periodic and can be generated by another timer); Its value can also be read by a DMA request generated by a real-time clock.

The UIFREMAP bit of the TIMxCR1 register forces the continuous copy of the update interrupt flag (UIF) to the highest bit of the timer counter (TIMxCNT [31]). This allows the count value and a state generated by the UIFCPY flag to be readable. This simplifies the calculation of angular velocity by avoiding causing race conditions. For example, by sharing processing between background tasks (counter readout) and interrupts (interrupt updates

There is no delay between UIF and UIFCPY flag assertions

In a 32-bit counter design, when the UIFREMAP bit is set, the 31st bit of the counter is overwritten by the UIFCPY flag when reading (the most significant bit of the counter is only accessible in write mode)

20.2.17.2 Clock plus directional encoder mode

In addition to quadrature encoders, the timer also supports other different types of encoder modes

Clock plus direction mode As shown in the figure, the clock is provided by TI2 and the direction is input by TI1, this mode is enabled by configuring the SMS [3: 0] of the TIMx_SMCR register as follows:

- 1010: x2 mode, counter updates on rising and falling edges of clock
- 1011: x1 mode, counter is updated at one edge, rising edge is updated at CC2P = 0, falling edge is updated at CC2P = 1

The polarity of the directional signal input to TI1 is determined by the CC1P bit, CC1P = 0: positive polarity (count up when TI1 is high, count down when TI1 is low); CC1P = 1: negative polarity (count down when TI1 is high, count up when TI1 is low)

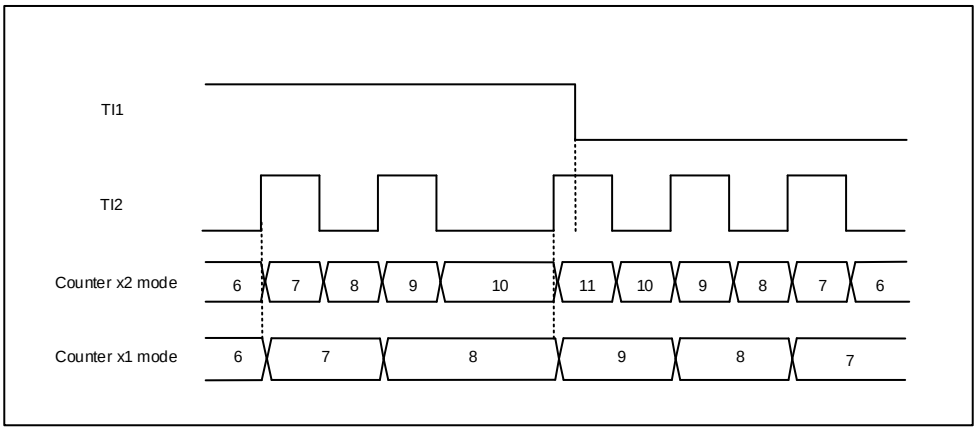


Figure 20-49 direction plus clock encoder mode

20.2.17.3 Directional clock encoder mode

In directional clock mode (shown in the figure below), the clock is provided by two signals, one at a time depending on the direction, thus having an up-count clock signal (TI2) and a down-count clock signal (TI1). This mode is enabled by configuring the SMS [3: 0] of the TIMx_SMCR register as follows:

- 1100: x2 mode, regardless of which clock line, the counter is updated on both the rising and falling edges of the clock. The clock initial state may be configured by the CC1P and CC2P bits. CCxP = 0 represents a high level initial state and CCxP = 1 represents a low level initial state
- 1101: x1 mode, the counter is updated at one edge, depending on the values of CCx1P and CCx2P. CCxP = 0 indicates a falling edge count and the initial state is high. CCxP = 1 indicates a rising edge count and the initial state is low.

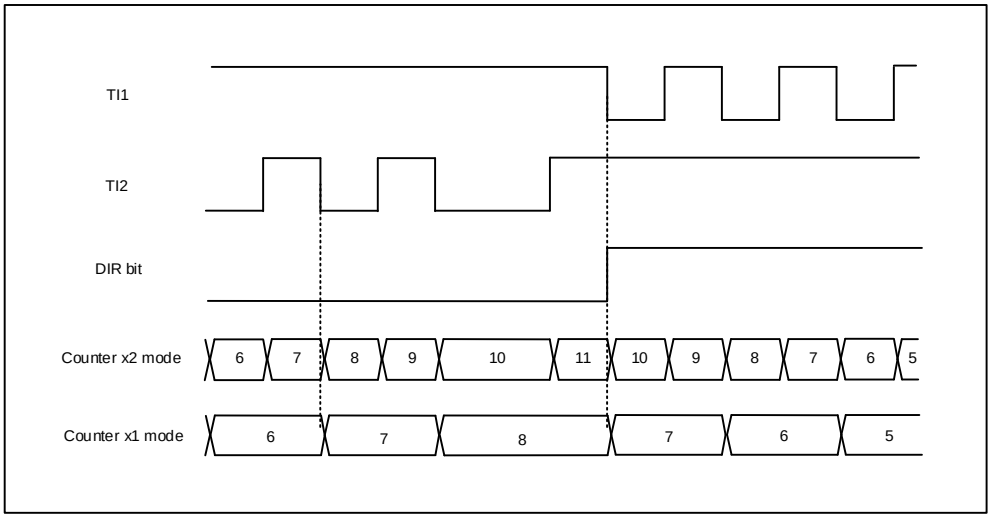


Figure 20-50 Directional Clock Encoder Mode (CC1P = CC2P = 0)

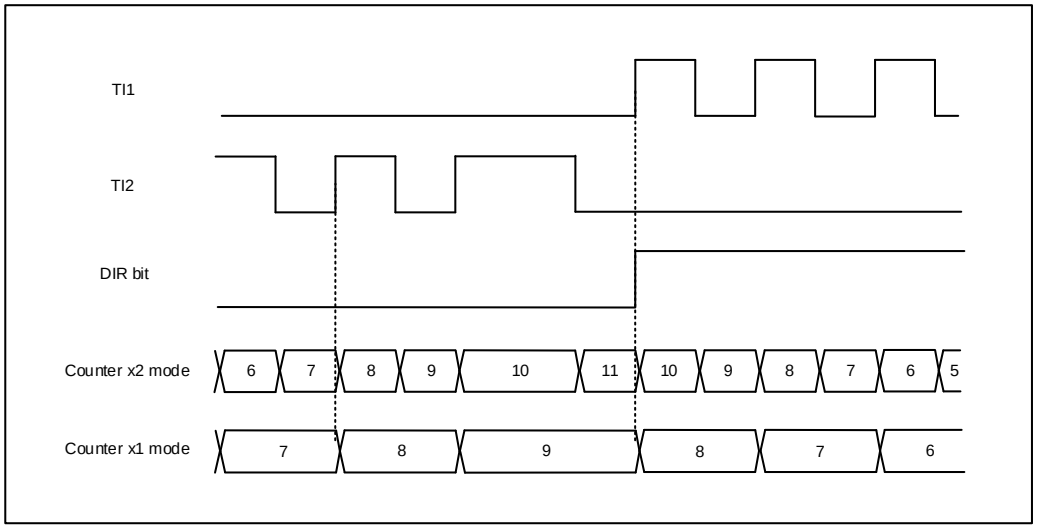


Figure 20-51 Directional Clock Encoder Mode (CC1P = CC2P = 1)

The following table details how the directional clock mode operates for any input transition:

Table 20- 4 Relationship between counting direction and encoder signal polarity

Effective edge	SMS [3: 0]	Reverse signal level (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
			Rising	Falling	Rising	Falling
x2 mode CCxP = 0	1100	High	Minus	Minus	Add	Add
		Low	-	-	-	-
x2 mode CCxP = 1	1100	High	-	-	-	-
		Low	Minus	Minus	Add	Add
x1 mode CCxP = 0	1101	High	-	Minus	-	Add
		Low	-	-	-	-
x1 mode CCxP = 1	1101	High	-	-	-	-
		Low	Minus	-	Add	-

20.2.17.4 Index input

The counter may be reset by an index signal output by the encoder for indicating an absolute reference position. The index signal must be connected to the tim_etr input. It may be filtered by a digital filter. The IE bit of the TIMx_ECR register enables the index function, and the IE bit can only be enabled in encoder mode, that is, when the SMS [3: 0] is configured to the following values: 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111

As shown in the figure below, commercial encoders have several options for index pulse adjustment

- Gating of A and B: The pulse width is 1/4 of a channel, aligned with the edges of A and B
- Gating of A (gating of B): The pulse width is 1/2 of a channel, aligned with the edge of A
- Ungated: Pulse width up to one channel period, not aligned with channel edge

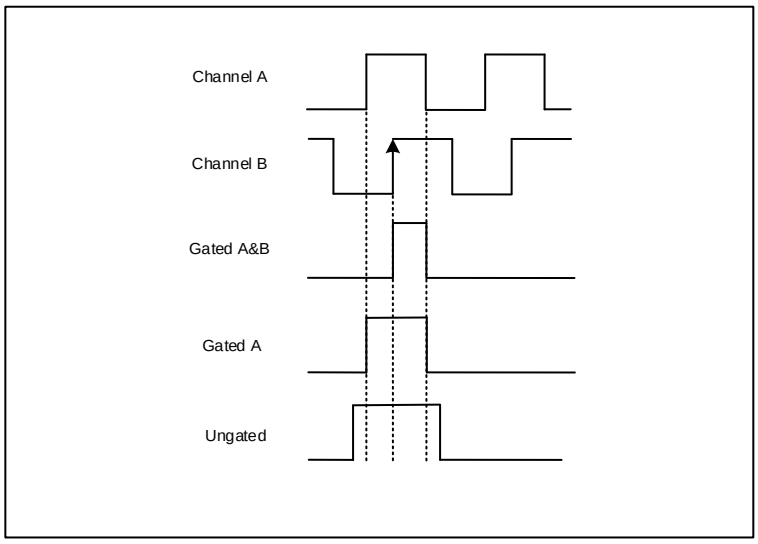


Figure 20-52 Index gating selection

The figure below shows that the circuit can tolerate the jitter of the index signal in any gating mode. In ungated mode, the index signal pulse width needs to be strictly less than 2 encoder cycles. If greater than or equal to 2 cycles, the counter will be reset multiple times.

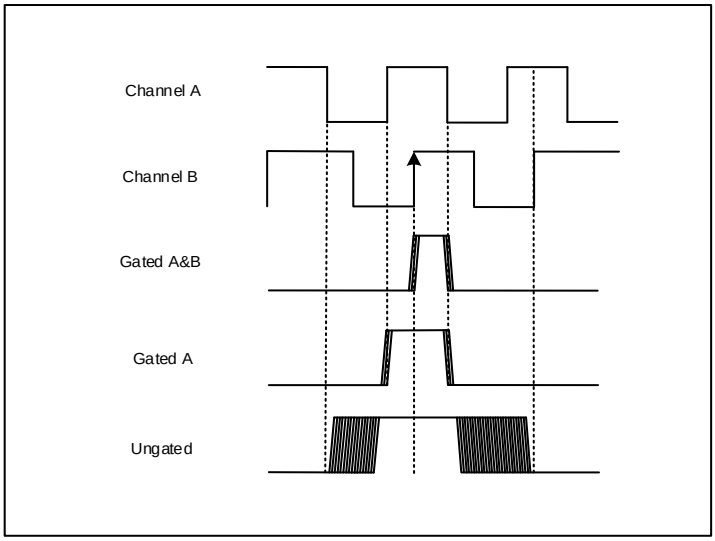


Figure 20-53 Index signals for jitter

The timer fully supports 3 gating modes and does not require special programming. It only needs to define what state of the encoder (the combination of channel A and channel B states) the index signal needs to be synchronized. By configuring the IPOS [1: 0] of the TIMx_ECR register.

The detection event of the index signal varies depending on the counting direction to ensure symmetrical operation during the rotation direction flip.

- The counter is reset during up count (DIR = 0)
- When counting down, the counter value is set to TIMx_ARR

This allows indexes to be generated at the same mechanical angular position regardless of the counting direction. The figure below shows which position indexes are generated, for a simple example (one encoder provides 4 mechanical rotations along each standard)

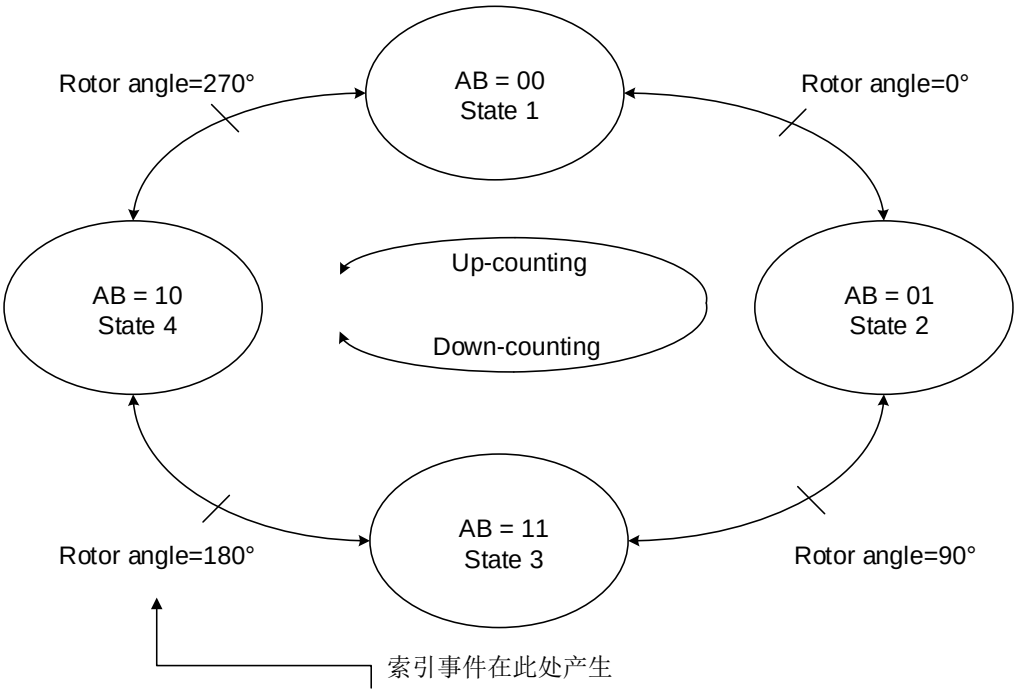


Figure 20-54 Index Generation at IPOS [1: 0] = 11

The following figure shows the waveform and corresponding values at IPOS [1: 0] = 11, which indicates that the moment when the counter value is forced is automatically adjusted according to the counting direction

- Counter cleared When the encoder enters 11 state (channel A = 1, channel B = 1), counting up (DIR = 0)
- The counter value is set to TIMx_ARR when leaving the 11 state, counting down (DIR = 1)

Index detection events can generate interrupts

The arrow indicates which conversion process index event interrupt is generated

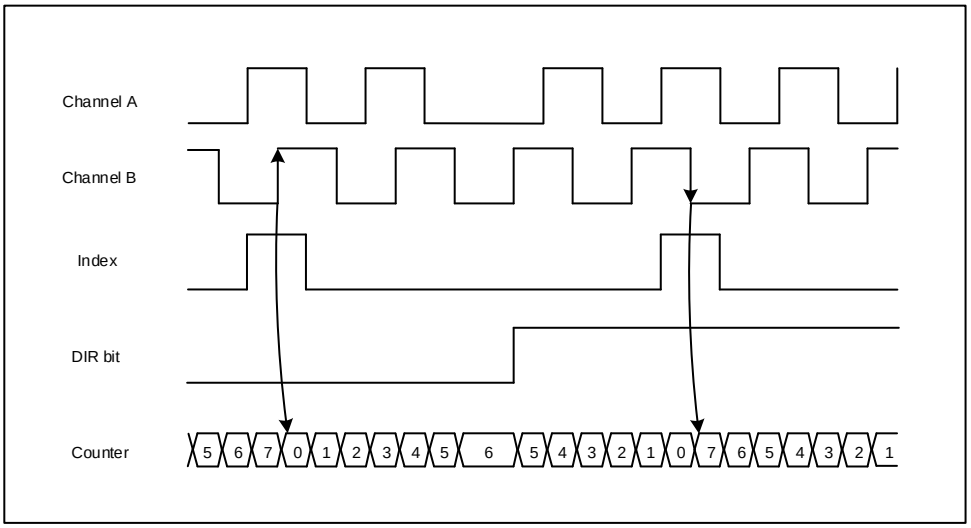


Figure 20-55 Index at IPOS [1: 0] = 11 and count values at channel A gated mode

The following figure shows the waveform and corresponding values in ungated mode, with arrows indicating which transition the index event interrupt occurred

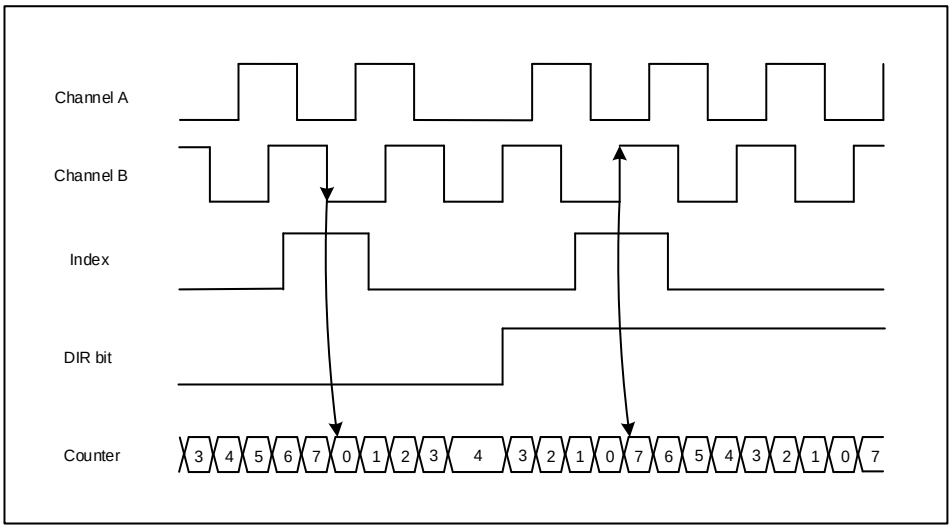


Figure 20-56 Count values in ungated index mode at IPOS [1: 0] = 00

The following figure shows the waveforms and corresponding values in A and B gated modes, with arrows indicating which transition process index event interrupt occurs

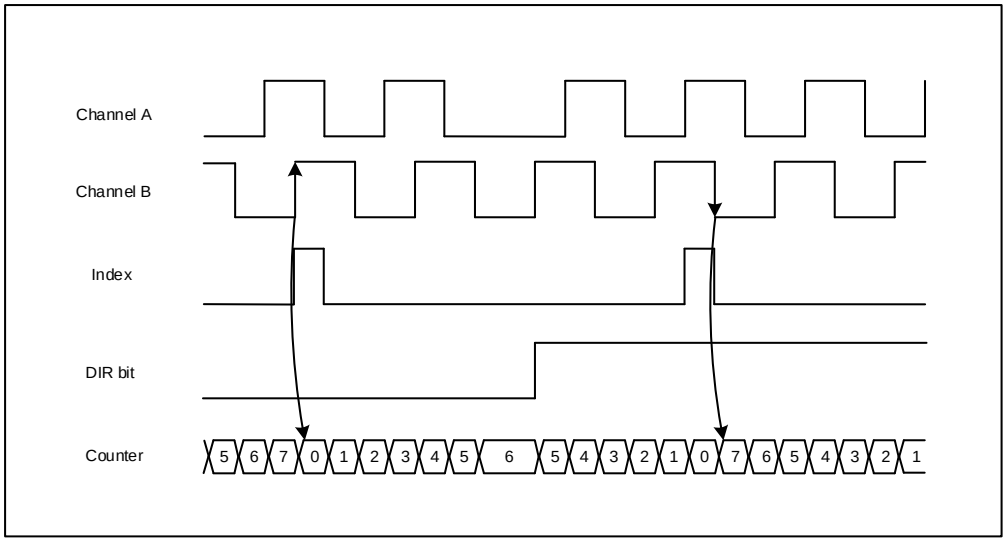


Figure 20-57 indexes and count values when channels A and B are gated

The following two figures illustrate in detail cases where the index pulse width is less than 1/4 of the encoder period

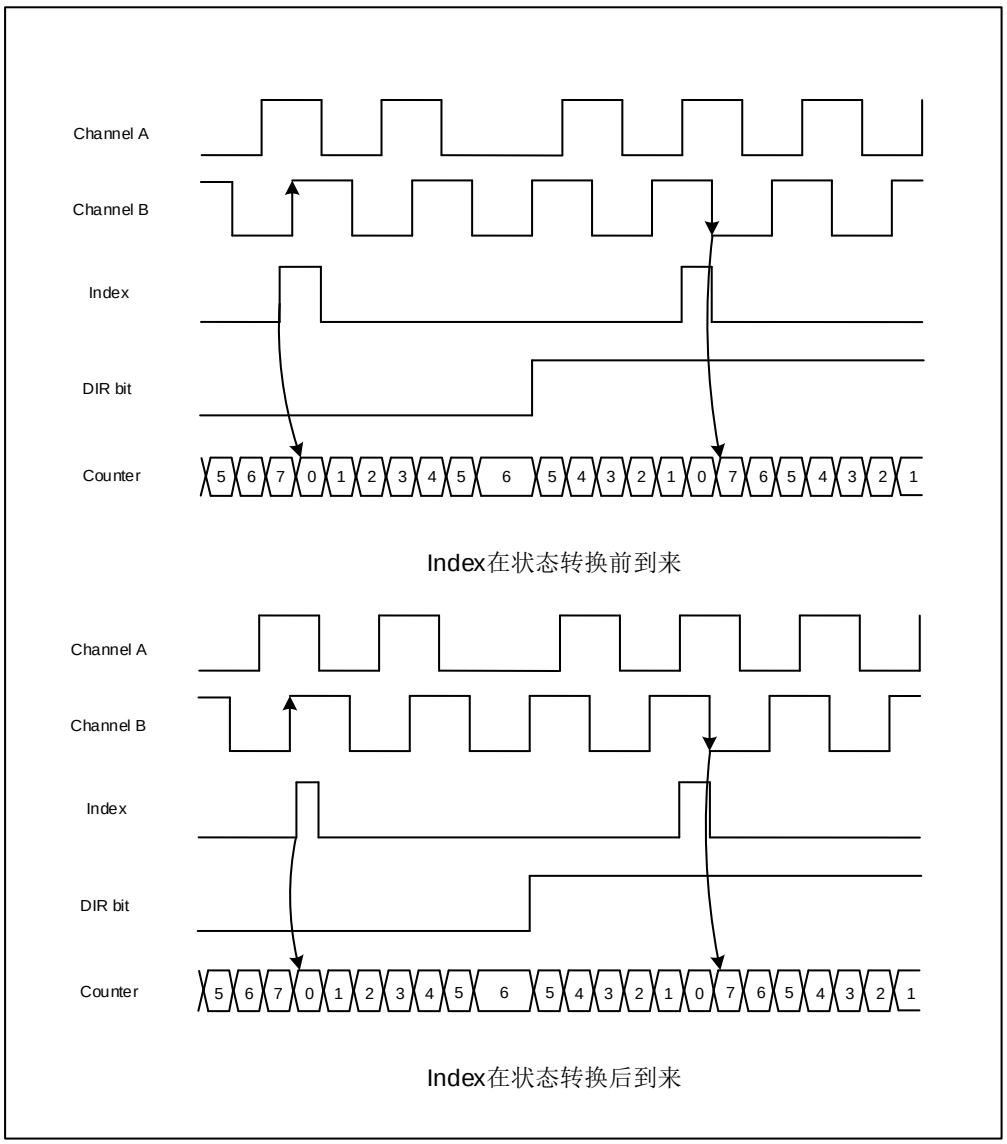


Figure 20-58 Operation of encoder mode with narrow index pulse at IPOS [1: 0] = 11

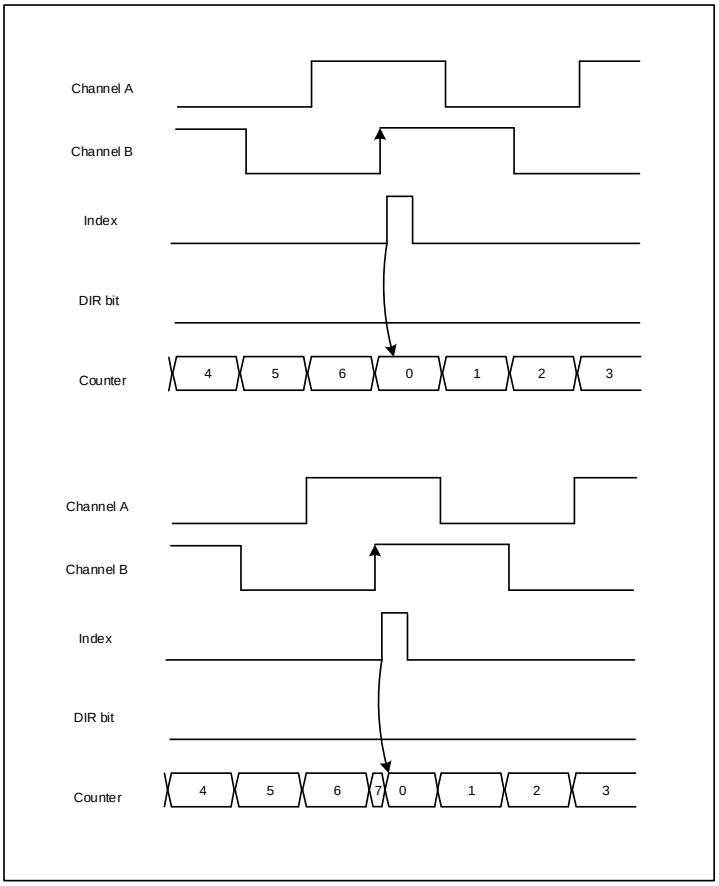


Figure 20-59 ARR = 0x07 time-to-time narrow index pulse reset counter

The following figure shows in detail how indexes are managed in x1 and x2 schemas

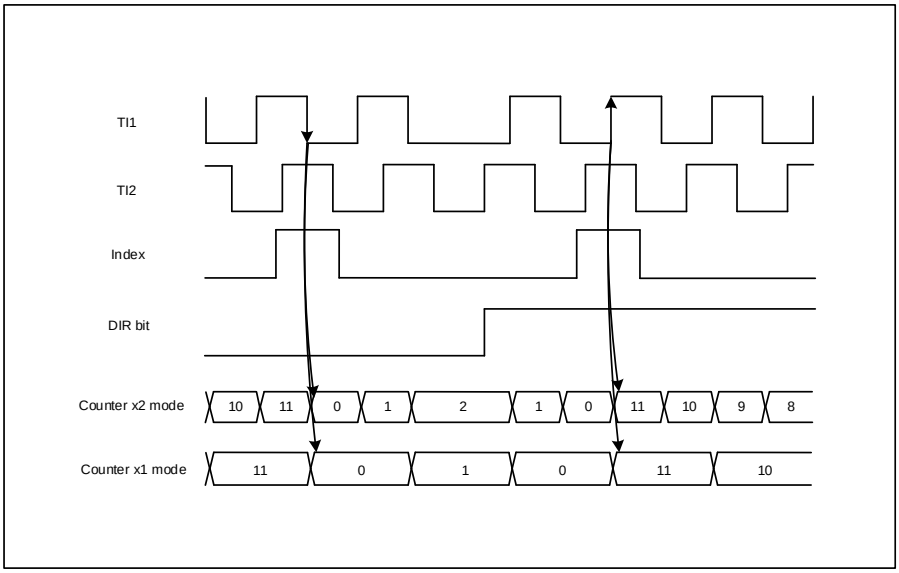


Figure 20-60 IPOS [1: 0] = 01 in x1 and x2 modes

■ Directional index sensitivity

You can allow the index to be valid only in the selected counting direction by configuring the IDIR [1: 0] of the TIMx_ECR register.

The following figure shows the relationship between values, indexes, and count reset events based on IDIR [1: 0].

Note: IDIR must be written until the IE bit is reset (index mode is off)

Note: Directional index sensitivity does not support clock + directional mode, when SMS = 1010 or 1011, IDIR must be set to 00

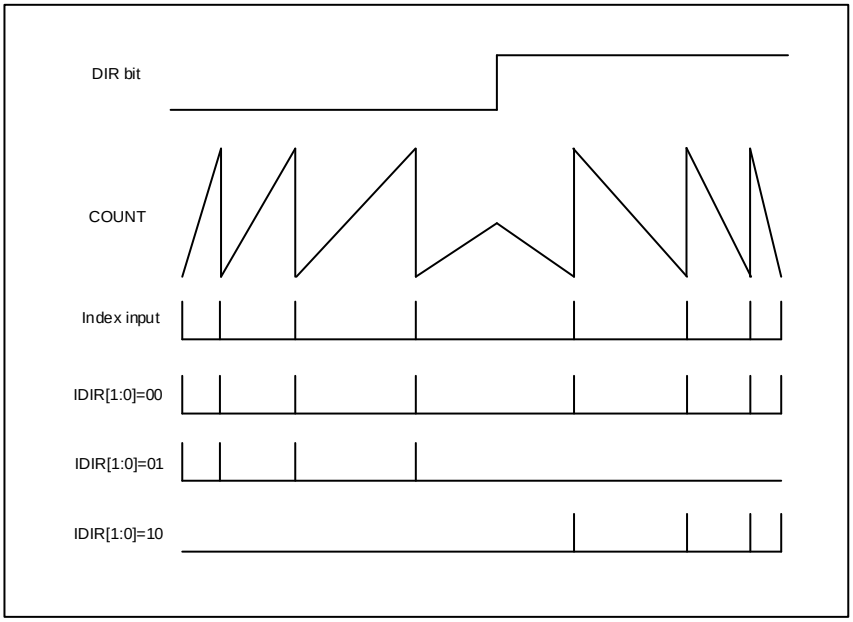


Figure 20-61 oriented index sensitivity

■ Special First Index Event Management

The FIDX bit of the TIMx_ECR register allows indexing to be performed only once, as shown in the figure below. When the first index arrives, the subsequent indexes are ignored. If desired, the circuit can be re-enabled by writing the FIDX bit to 0 and then to 1.

Note: When FIDX = 1, if the direction changes at position 0 (index active), index may be issued twice (IDXF flag set)

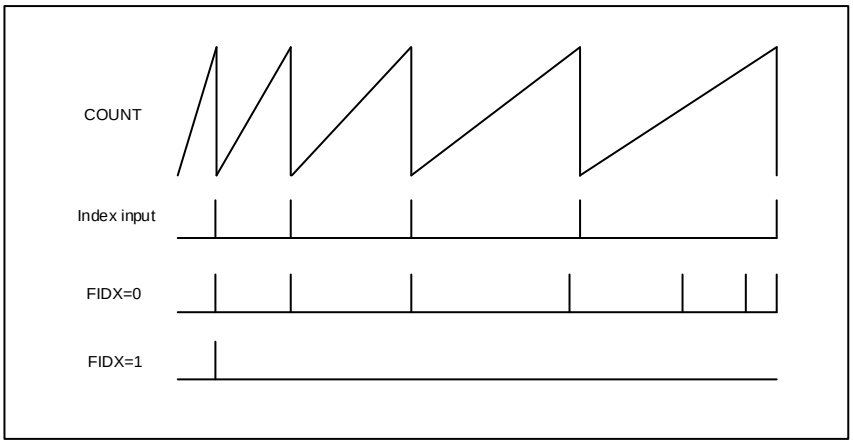


Figure 20-62 Counter reset when FIDX bit is set

■ Index management in non-orthogonal mode

The following two figures describe in detail how indexes are managed in directional clock mode and clock + directional mode. I.e. SMS = 1010 or 1011

In these modes, the sensitivity of the index is determined by the IPOS

IPOS [0] = 0: Detect index at low level of clock

IPOS [0] = 1: Detect index at high level of clock

IPOS [1] has no effect.

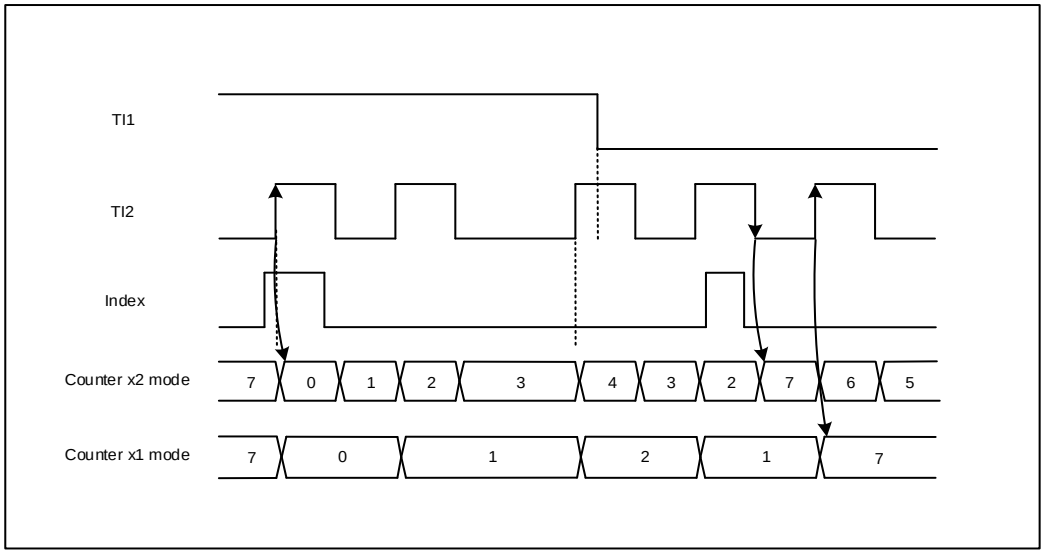


Figure 20-63 clock plus directional mode index behavior (IPOS [0] = 1)

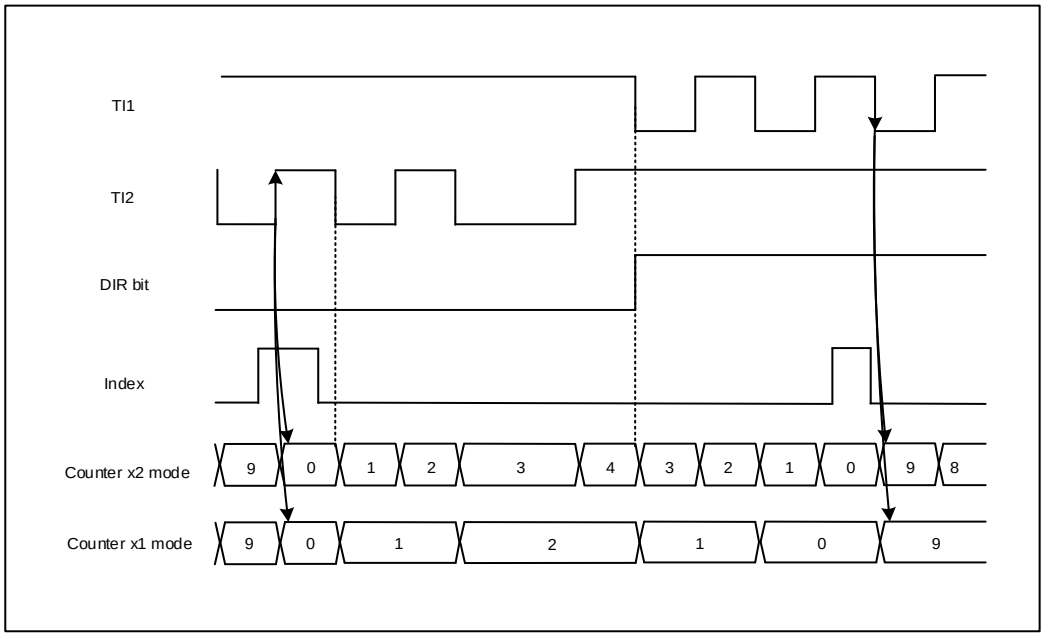


Figure 20-64 oriented clock mode index behavior (IPOS [0] = 1)

■ Encoder Error Management

There are 2 quadrature signals in the encoder configuration that are valid, and it is necessary to detect conversion errors. The readings on the 2 inputs correspond to 2-digit Gray codes to represent the state diagram, as shown in the figure. You can only change 1 bit at a time. The erroneous transition pulls up the interrupt flag TERRF in the TIMx_SR status register. If the TERRIE position of the TIMx_DIER register is 1, a transition error interrupt will be generated.

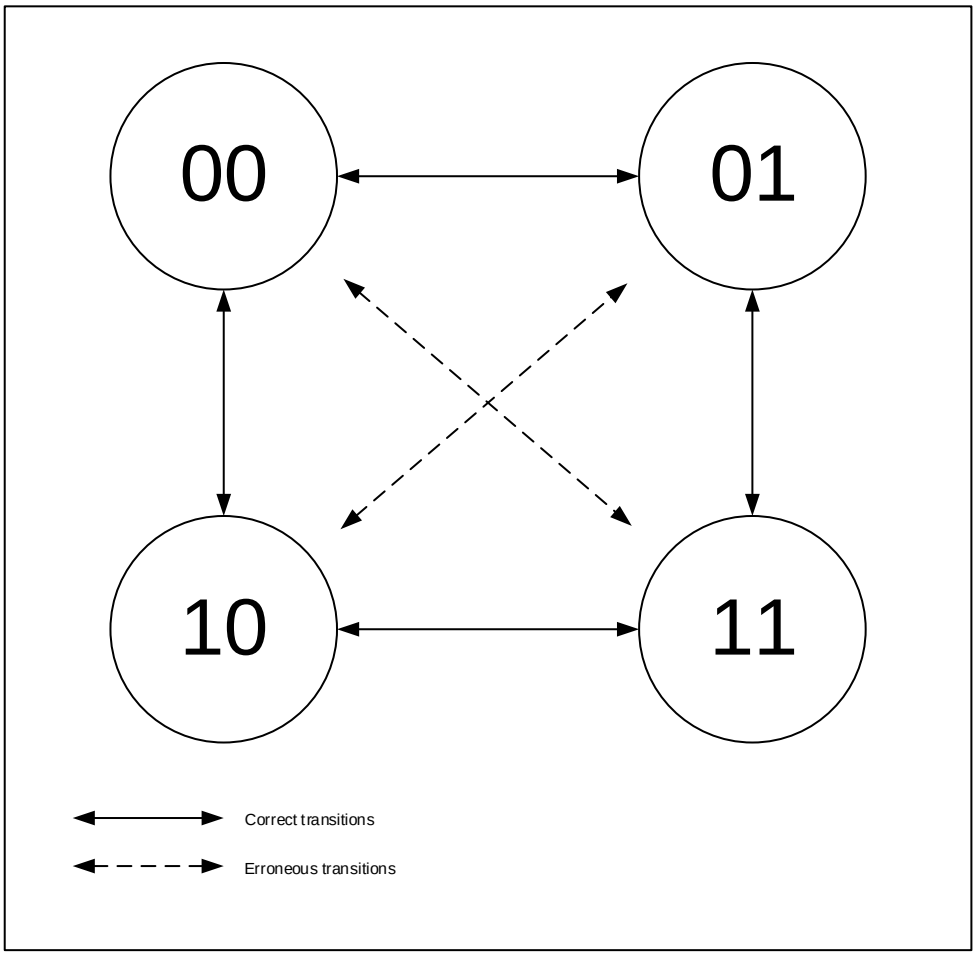


Figure 20-65 Schematic diagram of orthogonal coded signal states

After the encoder has the index signal, it can detect abnormal operation that causes pulse excess during each rotation. The encoder will provide N pulses per rotation and will produce a 4 * N count. The index signal resets the counter every 4 * N clock cycles.

If the counter value is tapered from TIMx_ARR to 0 or from 0 to TIMx_ARR without an index event, this is reported as an index position error.

The overflow threshold is configured by the TIMx_ARR register. The resulting count value of the 1000-line encoder ranges from 0 to 3999 (in 4x read mode). The overflow detection threshold must be set to TIMx_ARR = 3999 +1 = 4000

When counting up, the error is asserted when the transition is delayed to 0 to 1. The figure below shows the processing of narrow index signals in A and B gated modes.

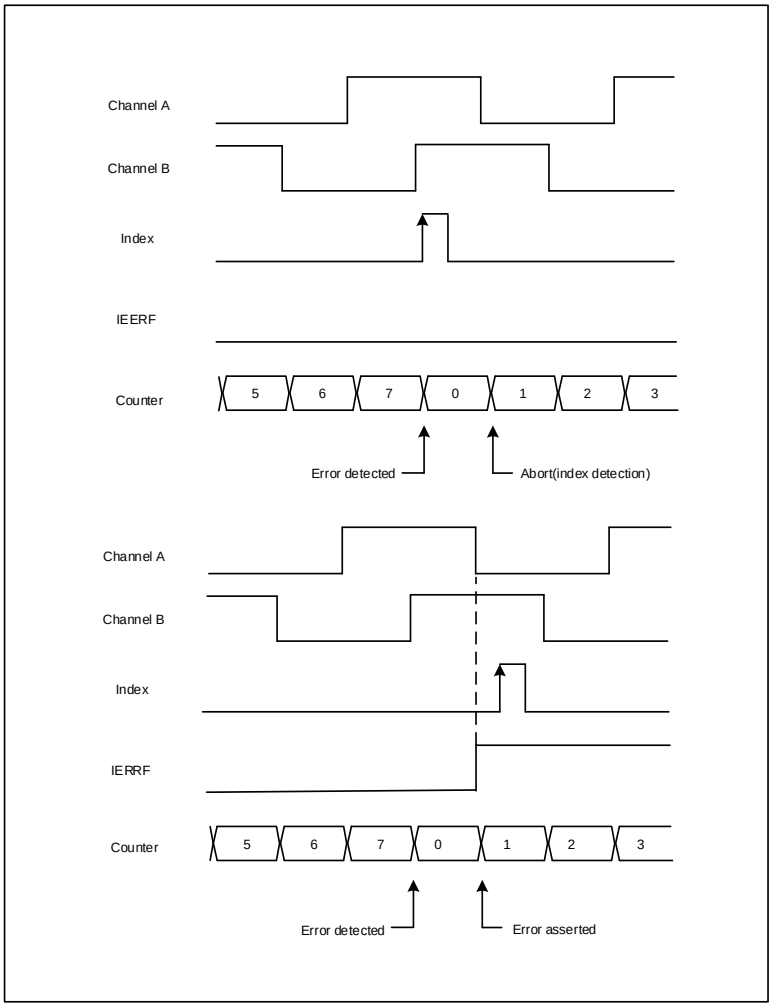


Figure 20-66 Error detection when is coded up

When counting down, the detection must be based on the previous transition from 1 to 0. The figure below shows the processing of narrow index signals in A and B gated modes. In order to avoid the encoder immediately after index detection jitter between the two values of TIMx_ARR and 0, and erroneously generate false determination.

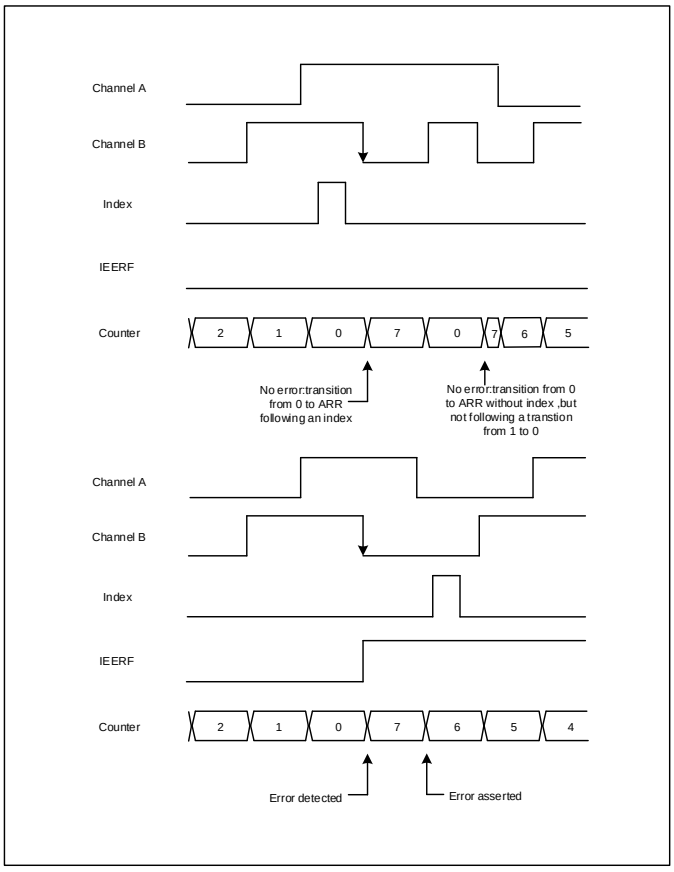


Figure 20-67 Error Detection when Downcoding

An index error will pull up the IERRF interrupt flag bit of the TIMx_SR status register. If the TERRIE position of the TIMx_DIER register is 1, a transition error interrupt will be generated.

- Functional encoder interrupt

The following interrupt are all valid in encoder mode

- Direction change: In encoder mode, any change in counting direction will cause the DIR bit of TIMx_CR1 to flip, and the change in direction will pull up the DIRF interrupt flag bit of the TIMx_SR status register. If the DIRIE bit of the TIMx_DIER register is enabled, a change of direction interrupt is also generated
- Index event: The index event will pull up the IDXF interrupt flag bit of the TIMx_SR status register. An index interrupt is also generated if the IDXIE bit of the TIMx_DIER register is enabled
- Runtime updates encoder mode from mode preload function

It is necessary to switch from one encoder mode to another operation mode, which is usually done at high speeds to reduce the rate of update interrupts. The figure below shows switching from x4 mode to x2 mode and then switching to x1 mode.

For this purpose, SMS [3: 0] can be preloaded. The enable bit is controlled by the SMSPE bit of the TIM_SMCR register. The selection of the trigger signal from the SMS [3: 0] preload to the active value can be controlled using the SMSPS bit of the TIMx_SMCR register.

- SMSPS = 0: Transmission is triggered by UEV, this mode must turn off the index function (IE = 0)
- SMSPS = 1: Transfer triggered by index event

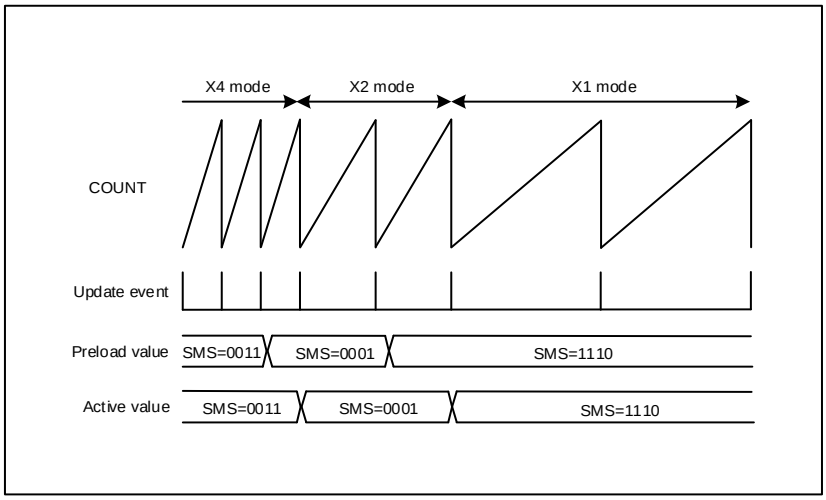


Figure 20-68 The encoder mode is updated by preload when event updates

- Encoder clock output

The working principle of the encoder is not entirely suitable for high-resolution fast measurements. At low speeds, it requires a considerable integration time to have a sufficient number of clock edges and accurate measurements.

At low speeds, the best way is to measure the edge-to-edge clock cycle directly. You may need to use a slave timer at this time. The timer may output the clock signal of the encoder through tim_trgo. The slave timer may perform cycle measurements and provide speed information for each encoder clock edge.

This mode is enabled by setting MMS [3: 0] = 1000 of the TIMx_CR2 register. And can only be used if the SMS [3: 0] is configured to 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111. Other SMS [3: 0] values are not allowed and may produce unexpected behavior.

20.2.18 Directional bit output

It is also necessary to output the direction bit of the timer, which is output from the OC3andOC4channels by configuring the OC3M [3: 0] and OC4M [3: 0] bit segments of the TIMx_CCMR2 register to 1011(copying the DIR bit of the TIMx_CR1 register).

This feature is used to monitor the counting direction (or rotation direction) of the encoder, or can provide an indication signal of the upper/lower stages in the intermediate alignment PWM mode.

20.2.19 UIF bit remapping

The UIFREMAP bit of the TIMx_CR1 register can force a continuous copy of the UIF to the 31st bit of the timer counter (TIMxCNT [31]). This allows both the value of the counter and a potential rolling state exhibited by the UIFCPY flag signal to be read out. In special cases, this can simplify the calculation by avoiding race conditions, for example by simultaneous processing of counter readout and update interrupts.

The UIF and UIFCPY flags are generated without delay.

In a 32-bit timer implementation, when the UIFREMAP bit is set, 31 bits of the counter are overwritten by the UIFCPY flag on read access (the most significant bit of the counter is only accessible in write mode).

20.2.20 Timer input XOR function

The TI1S bit in the TIMx_CR2 register allows the input filter of channel 1 to be connected to the output of an exclusive OR gate, whose three inputs are TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

20.2.21 Synchronization of TIMx timer and externally triggered

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gated mode, trigger, reset + trigger and gated + reset mode.

20.2.21.1 Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; Meanwhile, if the UDIS bit of the TIMx_CR1 register is low, an update event UEV is also generated; All preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=0100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, a trigger flag (TIF bit in the TIMx_SR register) is set, and an interrupt request or a DMA request is generated according to the setting of the TIE (Interrupt Enable) bit and the TDE (DMA Enable) bit in the TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

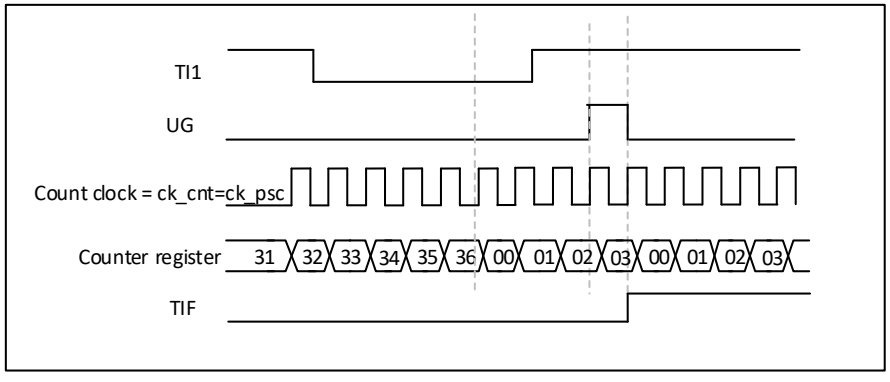


Figure 20-69 Control circuit in reset mode

20.2.21.2 Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this

example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only).

- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. SelectTI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

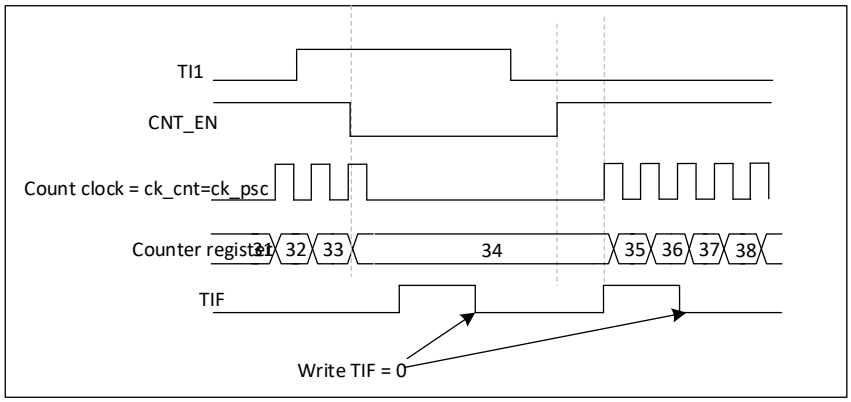


Figure 20-70 Control circuit in Gated mode

20.2.21.3 Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMx_CCMR1 register. Write CC2P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

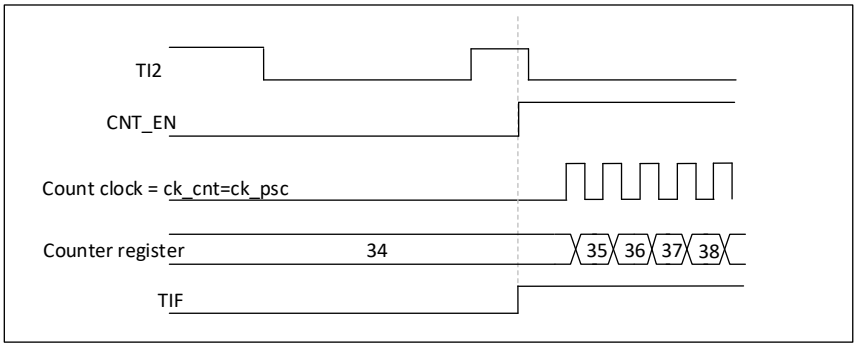


Figure 20-71 Control circuit in trigger mode

20.2.21.4 Slave mode: reset plus trigger combination mode

In this mode, the rising edge of the trigger input initializes the counter and generates an update signal for the register, turning on the counting

This mode is used for a single pulse mode

20.2.21.5 Slave mode: gating plus reset combination mode

The clock of the counter is enabled only when the trigger input is high. The counter stops when the trigger signal is pulled low. The start and stop of the counter can be controlled

This mode allows detection of out-of-range PWM signals (duty cycle exceeding maximum expected value)

20.2.21.6 Slave mode: External clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit through the TIMx_SMCR register:

— ETF = 0000: no filter

— ETPS = 00: prescaler disabled

— ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.

2. Channel 1 is configured as follows to detect the rising edge of TI1:

— IC1F = 0000: no filter

— The capture prescaler is not used for triggering, so it does not need to be configured.

— CC1S=01 in TIMx_CCMR1 register to select only the input capture source

— Set CC1P = 0 in the TIMx_CCER register to determine polarity (only rising edges are detected)

3. Set SMS = 110 in the TIMx_SMCR register, and configure the timer to trigger mode. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

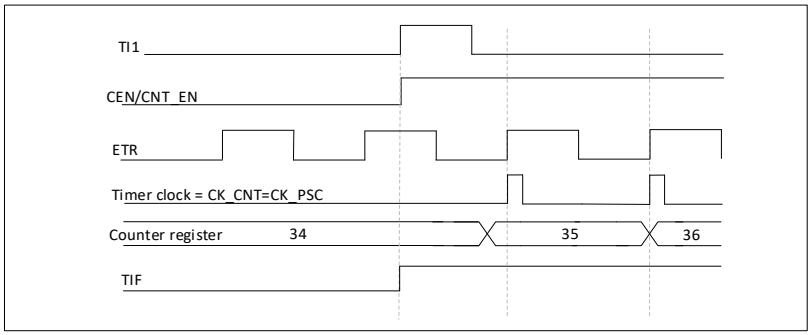


Figure 20-72 Control Circuit in External Clock Mode 2 + Trigger Mode

20.2.22 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

The following presents an overview of the trigger selection and the master mode selection blocks.

20.2.22.1 Using one timer as prescaler for another timer

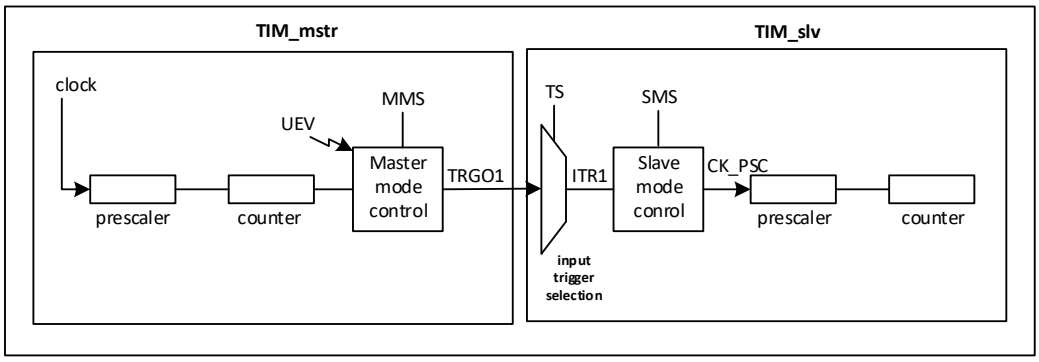


Figure 20-73 Master/Slave timer example

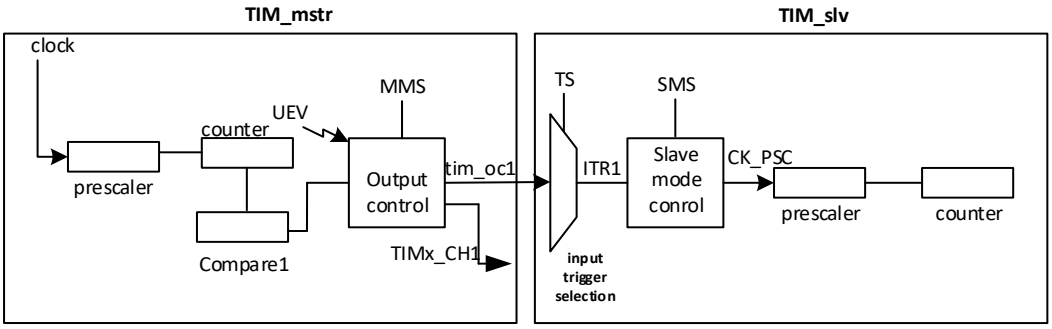


Figure 20-74 Example of master-slave connection with only 1 channel timer

A timer with only one channel (see image above) does not have a master mode. However, the tim_oc1 output signal may be used as a flip-flop for the slave timer. The tim_oc1 signal pulse width must be programmed to at least 2 clock cycles of the target timer to ensure that the slave timer detects a trigger. For example, if the target timer clock is 4 times slower than the source timer, the OC1 pulse width must be 8 clock cycles.

For example, TIM_mstr can be configured as the prescaler of TIM_slv. Referring to FIGS. 20-73, the following is performed:

- TIM_mstr is configured as the main mode, which can output a periodic trigger signal every time the event UEV is updated. When MMS = '010' of the TIM_mst_ CR2 register, a rising edge signal is output on TRGO1 every time an update event occurs.
- The TRGO1 output of TIM_mstr is connected to TIM_slv,TS = '00000' of the TIM_slv_ SMCR register isset, and TIM_slv is configured to be a slave mode using ITR1 as an internal trigger.
- The slave mode controller is then placed in external clock mode 1 (SMS = 111 for the TIM_slv_ SMCR register); In this way, TIM_slv can be driven by the periodic rising edge of TIM_mstr (i.e., counter overflow of TIM_mstr) signal.
- Finally, the respective CEN bits must be set to start two timers respectively.

Note: If OCx has been selected as the trigger output of TIM_mstr (MMS = 1xx), its rise is used to drive the counter of TIM_slv.

20.2.22.2 Using one timer to enable another timer

In this example, the enablement of TIM_slv is controlled by the output comparison of TIM_mstr. TIM_slv counts the divided internal clock only when the OC1REF of TIM_mstr is high. The clock frequencies of both timers are obtained by dividing CK_INT by 3 by the prescaler (fCK_CNT = fCK_INT/3).

- Configure TIM_mstr as the main mode, and send its output comparison reference signal (OC1REF) as the trigger output (MMS of TIM_mstr_ CR2 register = 100)
- OC1REF waveform configuring TIM_ mstr (TIM_mst_ CCMR1 register)
- Configure TIM_slv to get input trigger from TIM_mstr (TS = 00000for TIM_slv_ SMCR register)
- Configure TIM_slv in gated mode (SMS = 101 forTIM_slv_ SMCR register)
- Set CEN = 1 of the TIM _slv_ CR1 register to enable TIM_slv
- Set CEN= 1 of TIM_mst_ CR1 register to start TIM_mstr

Note: The clock of TIM_slv is not synchronized with the clock of TIM_mstr. This mode only affects the enable signal of the TIM_slv counter.

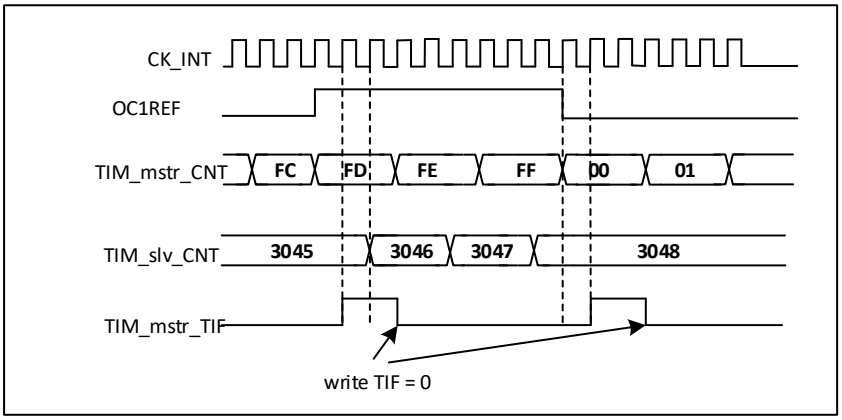


Figure 20-75 OC1REF Control TIM_slv for TIM_mstr

Inthe example above, their counters and prescalers are not initialized before TIM_slv starts, so they start counting from the current value. You can reset the two timers before starting TIM_mstr, so that they start with a given value, that is, write any value desired in the timer counter. The timers can easily be reset by software using the UG bit in the TIMx_EGR registers.

In the next example, TIM_mstr and TIM_slv need to be synchronized. TIM_mstr is the master mode and starts from 0, TIM_slv is the slave mode and starts from 0xE7; The prescaler coefficients of the two timers are the same. Writing '0' to the CEN bit of TIM_mstr_CR1 disables TIM_mstr, and TIM_slv stops.

- TIM_mstr is configured as the main mode, and the output comparison 1 reference signal (OC1REF) is sent as the trigger output (MMS of the TIM_mst_ CR2 register = 100).
- The OC1REF waveform of TIM_mstr is configured (TIM_mstr_ CCMR1 register).
- Configure TIM_slv to get input trigger from TIM_mstr (TS = 00000for TIM_slv_ SMCR register)
- Configure TIM_slv in gated mode (SMS = 101 forTIM_slv_ SMCR register)
- Set UG= '1' of the TIM_mstr_ EGR register to reset TIM_mstr.
- Set UG= '1' of the TIM_slv_ EGR register to reset TIM_slv.
- Write '0xE7' to the counter of TIM_slv (TIM_slv_ CNT), initializing it to 0xE7.
- SetCEN = '1' of the TIM_slv_ CR1 register to enable TIM_slv.
- SetCEN = '1' of the TIM_mstr_ CR1 register to start TIM_mstr.
- SetCEN = '0' of the TIM_mstr_ CR1 register to stop TIM_mstr.

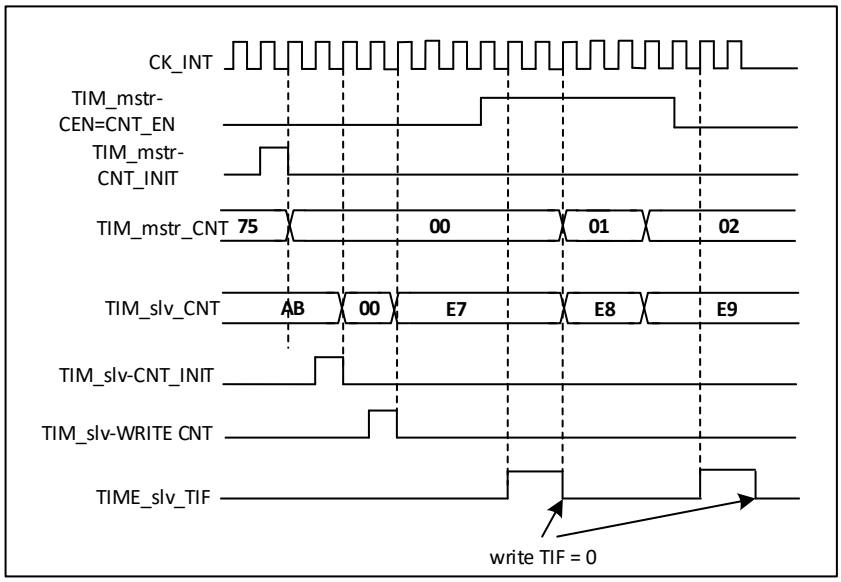


Figure 20-76 TIM_slvcan be controlled by enabling TIM_mstr

20.2.22.3 Using one timer to start another timer

In this example, TIM_slv is enabled using the update event of TIM_mstr. Once TIM_mstr generates an update event, TIM_slv starts counting from its current value (which may be non-0) according to the divided internal clock. Upon receipt of the trigger signal, the CEN bit of TIM_slv is automatically set to '1' and the counter starts counting until '0' is written to the CEN bit of the TIM_slv_CR1 register. The clock frequency of both timers is divided by 3 by the prescaler pair of CK_INT (fCK_CNT = fCK_INT/3).

- Configure TIM_mstr as the main mode and send its update event (UEV) as the trigger output (MMS = 010 of the TIM_mstr_CR2 register).
- Configure the cycle of TIM_mstr (TIM_mstr_ARR register).
- Configure TIM_slv to get input trigger from TIM_mstr (TS = 00000for TIM_slv_ SMCR register)
- Configure TIM_slv to trigger mode (SMS of TIM_slv_SMCR register = 110)

- Set CEN = 1 of the TIM_mstr_CR1 register to start TIM_mstr.

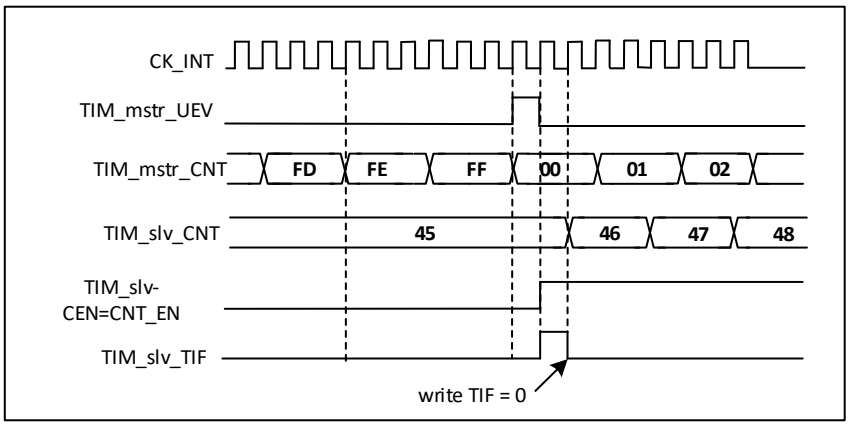


Figure 20-77 triggers TIM_slv using the update of TIM_mstr

As in the previous example, the user can initialize both counters before starting counting. The following figureshows the action of using the trigger mode instead of the gated mode (SMS = 110 of the TIM_slv_SMCR register) with the same configuration as 0.

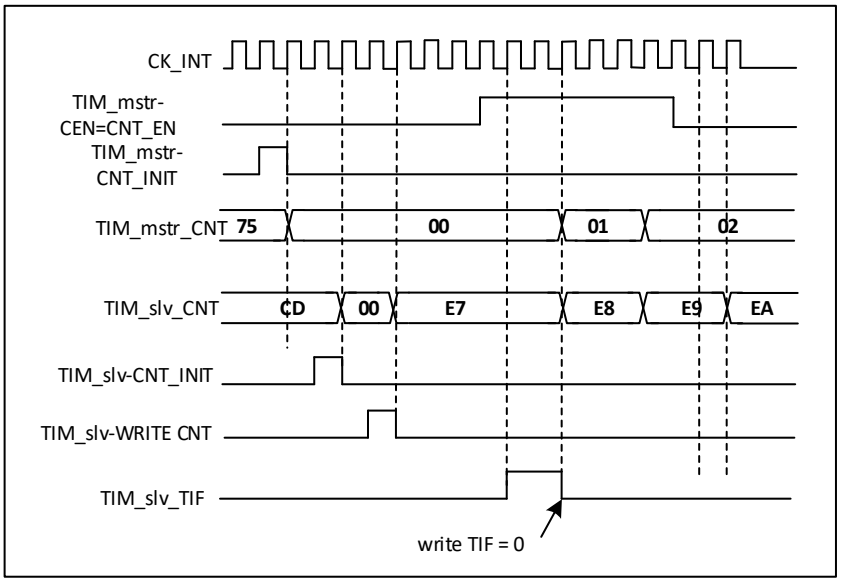


Figure 20-78 triggers TIM_slv with the enable of TIM_mstr

20.2.22.4 Starting 2 timers synchronously in response to an external trigger

This example enables TIM_mstr when the TI1 input of TIM_mstr rises, and enables TIM_mstr while enabling TIM_slv. To ensure the alignment of counters, TIM_mstr must be configured in master/slave mode (corresponding to TI1 as slave, corresponding to TIM_slv as master):

- Configure TIM_mstr as the main mode and send its enable as the trigger output (MMS = 001 of the TIM_mstr_CR2 register).
- Configure TIM_mstr in slave mode and obtain input trigger from TI1 (TS =00100of TIM_mstr_SMCR register).
- Configure TIM_mstr to trigger mode (SMS = 110 of TIM_mstr_SMCR register).
- Configure TIM_mstr in master/slave mode with MSM = 1 of the TIM_mstr_SMCR register.
- Configure TIM_slv to get input trigger from TIM_mstr (TS = 00000for TIM_slv_ SMCR register)

- Configure TIM_slv to trigger mode (SMS = 110 of TIM_slv_SMCR register).

When a rising edge appears on TI1 of TIM_mstr, the two timers start counting according to the internal clock synchronously, and the two TIF flags are also set at the same time.

Note: In this example, both timers are initialized (set the corresponding UG bits) before starting, and both counters start at 0, but an offset can be inserted between timers by writing to either counter register (TIMx_CNT). In the figure below, you can see that there is a delay between CNT_EN and CK_PSC of TIM_mstr in master/slave mode.

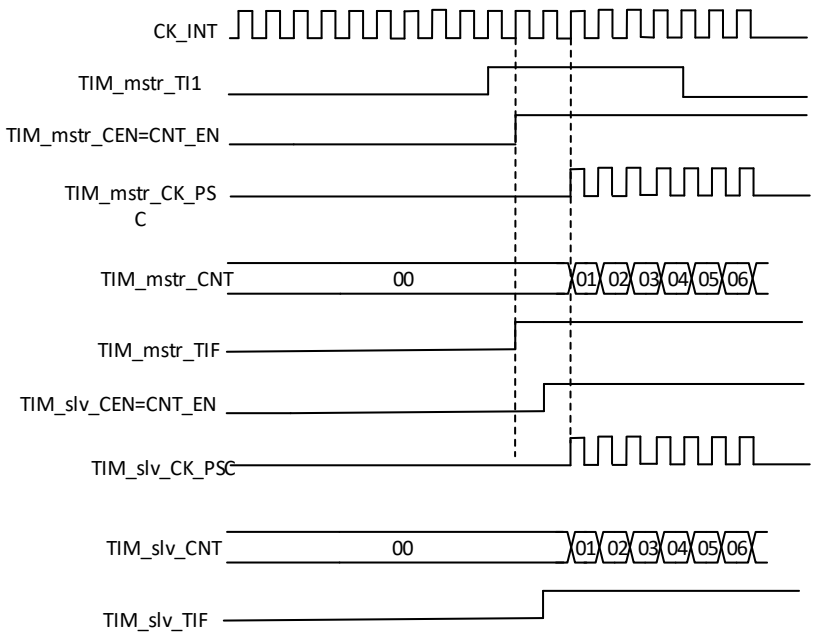


Figure 20-79 Triggers TIM_mstr and TIM_slv using the TI1 input of TIM_mstr

20.2.23 DMA Burst Mode

The TIMx timer has the ability to generate multiple DMA requests when a single event occurs. The main purpose is to be able to reprogram some functions of the timer multiple times without the over-head of software. But it is also possible to read several registers in a row at fixed intervals.

The destination of the DMA controller is unique and must point to the virtual register TIMx_DMAR. When a given timer event occurs, the timer initiates a series of DMA requests. Each write to the TIMx_DMAR register actually redirects a timer register.

The DBL [4: 0] bit in the TIMx_DCR register sets the DMA burst length. When a read or write access is made to a TIMx_DMR address, the timer recognizes burst transmissions, that is, the number of transmissions (in half a word or byte)

The DBA [4: 0] bit of TIMx_DCR defines the DMA base address for DMA transmission (when read and write operations are completed through the TIMx_DMAR address). The DBA is defined as the offset from the TIMx_CR1 register address.

Example:

00000: TIMx_CR1

00001: TIMx_CR2

00010: TIMx_SMCR

As an example, the timer DMA burst characteristic is used to update the contents in the CCRx register when an update event occurs, transmitting a half-word to CCRx by DMA.

This is done through the following steps:

1. Configure the corresponding DMA channel as follows:
   1) The DMA channel peripheral address is the address of the DMAR register
   2) The DMA channel storage address is the BUFF address in RAM, which contains the data transmitted by DMA to CCRx
   3) Number of data transmitted = 3
   4) Polling Mode Off
2. Configure DCR registers by configuring DBA and DBL: DBL = 3, DBA = 0XE
3. Enable timer update DMA request (UDE = 1)
4. Enable timer
5. Enable the DMA channel

This example applies to the case where each CCRx register is updated once. If each CCRx is updated twice, the number of data transfers needs to be set to 6. For example, assume that the buffer of RAM contains data1, data2, data3, data4, data5 and data6. Data is sequentially transmitted to CCRx: the first update DMA request, data1 is transmitted to CCR2, data2 is transmitted to CCR3, and data3 is transmitted to CCR4; For the second update DMA request, data4 is transmitted to CCR2, data5 is transmitted to CCR3, and data6 is transmitted to CCR4.

Note: Null values can be written to the reserved register

20.2.24 TIM2/3/4/5/18 DMA Request

TIM2/3/4/5/18 can generate DMA requests as shown in the following table:

Table 20-4 DMA Request

DMA request source	DMA enable control bit
update	UDE
Compare/Capture 1	CC1DE
Compare/Capture 2	CC2DE
Comparison/Capture 3	CC3DE
Comparison/Capture 4	CC4DE
Trigger	TDE

20.2.25 Debug mode

When the microcontroller enters debug mode (Cortex-M4F core stops), according to the setting of DBG_TIMx_STOP in the DBG module,

The TIMx counter may either continue normal operation or stop. See subsequent DBG sections for details.

Behavior in debug mode can be programmed using a dedicated configuration for each timer in the Debug Support Module (DBG)

For safety reasons, when the counter stops, the output is disabled (if the MOE is reset). The output can be forced to an invalid level (OSSI = 1), or taken over by the GPIO controller (OSSI = 0), typically forced to high impedance.

For more details, see the DBG module description

20.2.26 TIM2/3/4/5/18 low power mode

Mode	Description
Sleep	No effect, peripheral is active. Interruption may cause device to exit sleep
Stop	The timer operation stops and the register contents remain. Cannot generate an interrupt
Standby	The timer has been powered down and must be reinitialized after exiting standby mode.

20.2.27 TIM2/3/4/5/18 Interrupted

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
Comparison/Capture 3	CC3IF	CC3IE
Comparison/Capture 4	CC4IF	CC4IE
COM	COM	COMIE
Trigger	TIF	TIE
Encoder Index	IDXF	IDXIE
Encoder direction	DIRF	DIRIE
Encoder index error	IERRF	IERRIE
Encoder transmission error	TERRF	TERRIE

20.3 TIMx registers

20.3.1 TIMx Control Register 1 (TIMx_CR1) (x = 2/3/4/5/18)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DITHEN	UIFREMAP	Res	CKD [1: 0]		ARPE	CMS [1: 0]		DIR	OPM	URS	UDIS	CEN
			RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 13	Reserved	-	-	-
12	DITHEN	RW	0	Jitter enable (Dithering enbale) 0: Jitter off 1: Jitter on Note: This bit can only be modified when the CEN bit is reset
11	UIFREMAP	RW	0	UIF status bit remapping enable 0: No remapping, UIF status bit is not copied to bit 31 of TIMx_CNT register 1: Remap, UIF status bit copied to bit 31 of TIMx_CNT register
10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the frequency division ratio between the timer clock (CK_INT) frequency and dead time and the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 5	CMS	RW	2'h0	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR).

				01: Center-aligned mode 1. The counter counts up and down alternatively. The output comparison interrupt flag bit of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) is set only when the counter is counted down. 10: Center-aligned mode 2. The counter counts up and down alternatively. The output comparison interrupt flag bit of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) is set only when the counter is counted up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The output of the channel configured to output (CCxS = 00 in the TIMx_CCMRx register) compares the interrupt flag bit, which is set when the counter counts up and down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
4	DIR	RW	0	Direction 0: Counter counts up; 1: The counter counts down. Note: This bit is read-only when the counter is configured in center alignment mode or encoder mode.
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

20.3.2 TIMx Control Register 2 (TIMx_CR2) (x = 2/3/4/5/18)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						MMS[3]	Res								
						RW									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								TI1S	MMS [2: 0]			CCDS	Res		
								RW	RW	RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	MMS[3]	RW	0	See MMS [2: 0] description for details

24: 8	Reserved	-	-	-
7	TI1S	RW	0	TI1 selection 0: TIMx_CH1 pin connected to TI1 input; 1: The TIMx_CH1, TIMx_CH2, and TIMx_CH3 pins are connected to the TI1 input via XOR.
6: 4	MMS	RW	3'h0	Main mode selection (Master mode selection) For selecting synchronization information (TRGO) to be sent to the slave timer in master mode. The combination is as follows: 0000: Reset-The UG bit of the TIMx_EGR register is used as a trigger output (TRGO). If the reset is generated by the trigger input (the slave mode controller is in reset mode), the signal on the TRGO will have a delay relative to the actual reset. 0001: Enable-Counter enable signal CNT_EN may be used as a trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timeris enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 0010: Update-The update event is selected as Trigger Output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 0011: Comparison Pulse-When a catch occurs or a comparison is successful, when the CC1IF flag is to be set (even if it is already high), the trigger output sends a positive pulse (TRGO). 0100: Compare - OC1REFC signal is used as trigger output (TRGO) 0101: Compare - OC2REFC signal is used as trigger output (TRGO) 0110: Compare - OC3REFC signal is used as trigger output (TRGO) 0111: Compare - OC4REFC signal is used as trigger output (TRGO) 1000: Encoder clock output-encoder clock signal as trigger output (TRGO). This encoded value is only valid if the SMS [3: 0] has values of 0001, 0010, 0011, 1010, 1011, 1100, 1101, 1110, 1111. Other SMS values may result in unexpected effects. 1001-1111: Reserved Note: 1. The clock of the slave timer or ADC must be enabled prior to receive events from the master timer, and must not be changed. 2. If the master and slave timers are not on the same bus, the master mode should be configured to the width that can be picked by the slave timer.
3	CCDS	RW	0	Capture/compare DMA selection 0: When a CCx event occurs, send a DMA request for CCx; 1: CCx DMA requests sent when update event occurs
2: 0	Reserved	-	-	-

20.3.3 TIMx Slave Mode Control Register (TIMx_SMCR) (x = 2/3/4/5/18)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						SMSPS	SMSPE	Res		TS [4: 3]		Res			SMS[3]
						RW	RW			RW	RW				RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS [1: 0]		ETF [3: 0]				MSM	TS [2: 0]			OCCS	SMS [2: 0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	SMSPS	RW	0	SMS preload source This bit selects whether the event can trigger SMS [3: 0] 0: Transmission triggered by an update event of a timer 1: Transmission triggered by index event
24	SMSPE	RW	0	SMS preload enable This bit selects whether SMS [3: 0] can be preloaded

				0: SMS [3: 0] can be preloaded 1: SMS [3: 0] cannot be preloaded
23: 22	Re-served	-	-	-
21: 20	TS [4: 3]	RW	2'h0	See TS [2: 0] description for details
19: 17	Re-served	-	-	-
16	SMS[3]	RW	0	See SMS [2: 0] description for details
15	ETP	RW	0	External trigger polarity External trigger polarity This bit selects whether ETR or the inversion of the ETR is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge; 1: ETR is inverted, active at low level or falling edge
14	ECE	RW	0	External clock enable bit (External clock enable) This bit enables external clock mode 2 0: External clock mode 2 disabled; 1: External clock mode 2 enabled. The counter is drived by any active edge on the ETRF signal. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=00111). Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gate mode and trigger mode; However, the TRGI cannot be connected to the ETRF at this time (the TS bit cannot be '00111'). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13: 12	ETPS	RW	2'h0	Externally triggered prescalation External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. When a faster external clock is input, the frequency of the ETRP can be reduced using prescalation. 00: Turn off prescaling 01: ETRP frequency divided by 2; 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8.
11: 8	ETF	RW	4'h0	Externally triggered filtering External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
7	MSM	RW	0	Master/slave mode 0: No action; 1: The event on the trigger input (TRGI) is delayed to allow perfect synchronization between the current timer (via TRGO) and its slave timers. It is useful if we want to synchronize several timers on a single external event.
6: 4	TS	RW	3'h0	Trigger selection Trigger selection Select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1)

				00110: Filtered timer input 2 (TI1FP2) 00111 external trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) 01010: Internal Trigger 6 (ITR6) 01011: Internal Trigger 7 (ITR7) 01100: Internal Trigger 8 (ITR8) 01101: Internal Trigger 9 (ITR9) 01110: Internal Trigger 10 (ITR10) 01111: Internal Trigger 11 (ITR11) 10000: Internal Trigger 12 (ITR12) 10001: Internal Trigger 13 (ITR13) 10010: Internal Trigger 14 (ITR14) 10011: Internal Trigger 15 (ITR15) 10100: Internal Trigger 16 (ITR16) 10101: Internal Trigger 17 (ITR17) 10110: Internal Trigger 18 (ITR18) Others: Reserved For more details about ITRx, see the System Cascade Table Note: These bits can only be changed when they are not used (e.g. SMS = 0000) to avoid erroneous edge detection when changing.
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT connects OCREF_CLR input 1: OCREF_CLR_INT connection ETRF
2: 0	SMS	RW	3'h0	Select from Mode (Slave mode selection) When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 0000: Slave mode disabled - if CEN = 1 then the prescaler is clocked directly by the internal clock. 0001: Quadrature Encoder Mode 1, x2 Mode-The counter counts up/down on the edge of TI1FP1 depending on the level of TI2FP2. 0010: Quadrature Encoder Mode 2, x2 Mode-The counter counts up/down on the edge of TI2FP2 depending on the level of TI1FP1. 0011: Quadrature Encoder Mode 3, x4 Mode-The counter counts up/down on the edges of TI1FP1 and TI2FP2 depending on the input level of the other signal. 0100: Reset Mode-The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a signal to update the register. 0101: Gated mode-When the trigger input (TRGI) is high, the clock of the counter is turned on. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger mode-The counter starts (but does not reset) on the rising edge of the trigger input TRGI, only the start of the counter is controlled. 0111: External clock mode 1-The rising edge of the selected trigger input (TRGI) drives the counter. 1000: Reset + Trigger Combination Mode:-Rising edge of Trigger Input (TRGI) initializes counter, generates register update and starts counter 1001: Gate + Reset Combination Mode:-Trigger input (TRGI) is high when counter clock enabled. The counter stops and resets once the trigger signal is pulled low. The start and stop of the counter are controllable. 1010: Encoder mode: clock + direction, x2 mode 1011: Encoder mode: clock + direction, x1 mode, sensitivity of TI2FP2 set by CC2P 1100: encoder mode: directional clock, x2 mode 1101: Encoder mode: directional clock, x1 mode, sensitivity of TI1FP1, TI2FP2 set by CC1P and CC2P 1110: quadrature encoder mode: x1 mode. Count only at the edge of TI1FP1, the edge sensitivity is set by CC1P 1111: quadrature encoder mode: x1 mode. Counted only at the edge of TI2FP2, the edge sensitivity is set by CC2P Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS=00100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. Note: Do not use UEV as the tim_trgo output signal in encoder mode(i.e. mms cannot be configured to 010).

20.3.4 TIMx DMA/Interrupt Enable Register (TIMx_DIER) (x = 2/3/4/5/18)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								TERRIE	IERRIE	DIRIE	IDXIE	Res			
								RW	RW	RW	RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW			

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	TERRIE	RW	0	Transmission Error Interrupt Enable Transition error interrupt enable 0: Transmission error interrupt not enabled 1: Transmission Error Interrupt Enable
22	IERRIE	RW	0	Index Error Interrupt Enable (Index error interrupt enable) 0: Index error interrupt not enabled 1: Index error interrupt enable
21	DIRIE	RW	0	Direction change interrupt enable (Direction change interrupt enable) 0: Direction change interrupt disabled 1: Direction change interrupt enable
20	IDXIE	RW	0	Index interrupt enable (Index interrupt enable) 0: Index interrupt not enabled 1: Index interrupt enabled
19: 15	Reserved	-	-	-
14	TDE	RW	0	Trigger DMA request enable 0: prohibit triggering of DMA request; 1: Allow DMA request to be triggered.
13	Reserved	-	-	-
12	CC4DE	RW	0	Capture/Compare 4 DMA request enable 0: DMA request for capture/compare 4 is prohibited; 1: DMA request allowed to capture/compare 4.
11	CC3DE	RW	0	Capture/Compare 3 DMA request enable 0: DMA request for capture/compare 3 is prohibited; 1: DMA request allowed to capture/compare 3. 。
10	CC2DE	RW	0	Capture/Compare 2 DMA request enable 0: DMA request for capture/compare 2 is prohibited; 1: DMA request allowed to capture/compare 2.
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: DMA request to capture/compare 1 is prohibited; 1: DMA request allowed to capture/compare 1. 。
8	UDE	RW	0	Update DMA request enable 0: DMA request to prohibit update; 1: Allow updated DMA requests.
7	BIE	RW	0	Allow brake interruption (Break interrupt enable) 0: Brake interruption is prohibited; 1: Allow the brake to be interrupted.
6	TIE	RW	0	Trigger interrupt enable (Trigger interrupt enable) 0: prohibit triggering interrupt; 1: Enable trigger interrupt.
5	Reserved	-	-	-
4	CC4IE	RW	0	Allow capture/compare 4 interrupts (Capture/Compare 4 interrupt enable) 0: Disable capture/compare 4 interrupt; 1: Allow capture/compare 4 interrupts.
3	CC3IE	RW	0	Allow capture/compare 3 interrupts (Capture/Compare 3 interrupt enable) 0: Disable capture/compare 3 interrupt; 1: Allow capture/compare 3 interrupts.
2	CC2IE	RW	0	Allow capture/compare 2 interrupts (Capture/Compare 2 interrupt enable) 0: Disable capture/compare 2 interrupt; 1: Allow capture/compare 2 interrupts.
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable) 0: Disable capture/compare 1 interrupt;

				1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

20.3.5 TIMx Status Register (TIMx_SR) (x = 2/3/4/5/18)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								TERRF	IERRF	DIRF	IDXF	Res			
								RC_W0	RC_W0	RC_W0	RC_W0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			CC4OF	CC3OF	CC2OF	CC1OF	Res		TIF	Res	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			RC_W0	RC_W0	RC_W0	RC_W0			RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	TERRF	RC_W0	0	Transmission Error Interrupt Flag (Transition error interrupt flag) This flag is set by the hardware when a transmission error occurs in the encoder mode. Can be cleared by software or write cleared 0: No encoder transmission error detected 1: Encoder transmission error detected
22	IERRF	RC_W0	0	Index error interrupt flag (Index error interrupt flag) The flag is set by the hardware when an index error is detected. Can be cleared by software or write cleared
21	DIRF	RC_W0	0	Direction change interrupt flag Direction change interrupt flag This flag is set by hardware when a direction change occurs in the encoder mode (DIR bit change in TIMx_CR). Can be cleared by software or write cleared 0: No direction change detected 1: Direction change detected
20	IDXF	RC_W0	0	Index interrupt flag (Index interrupt flag) The flag is set by the hardware when an index event is detected. Can be cleared by software or write cleared 0: No index event occurred 1: An index event occurs
19: 13	Reserved	-	-	-
12	CC4OF	RC_W0	0	Capture/Compare 4 Repeat Capture Markers Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 Repeat Capture Markers (Capture/Compare 3 overcapture flag) Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 Repeat Capture Markers (Capture/Compare 2 overcapture flag) Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 Repeat Capture Marker (Capture/Compare 1 overcapture flag) This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been generated. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8: 7	Reserved	-	-	-
6	TIF	RC_W0	0	Trigger interrupt flag (Trigger interrupt flag) This position is' 1 'by hardware when a trigger event occurs (a valid edge is detected at the TRGI input when the slave mode controller is in a mode other than gated mode, or either edge in gated mode). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved	-	-	-

4	CC4IF	RC_W0	0	Capture/Compare 4 Interrupt Markers (Capture/Compare 4 interrupt flag) Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 Interrupt Markers (Capture/Compare 3 interrupt flag) Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare 2 Interrupt Markers (Capture/Compare 2 interrupt flag) Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes high under the counter overflow condition in the up or up/down count mode, or under the counter underflow condition in the down count mode If channel TI1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0, when the repetition counter value overflows or underflows (an update event occurs when the repetition counter = 0). − If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

20.3.6 TIMx Event Generation Register (TIMx_EGR) (x = 2/3/4/5/18)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TG	Res	CC4G	CC3G	CC2G	CC1G	UG
									W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	TG	W	0	Trigger generation This bit is set '1' by the software to generate a trigger event, and '0' is automatically cleared by the hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	Reserved	-	-	-
4	CC4G	W	0	Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action

				1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged). The counter is cleared '0' if in centrosymmetric mode or DIR = 0 (count up); If DIR = 1 (count down), the counter takes the value of TIMx_ARR.

20.3.7 TIMx Capture/Compare Mode Control Register 1 (TIMx_CCMR1) (x = 2/3/4/5/18)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							OC2M[3]	Res.							OC1M[3]
							RW								RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M [2: 0]			OC2PE	OC2FE	CC2S [1: 0]		OC1CE	OC1M [2: 0]			OC1PE	OC1FE	CC1S [1: 0]	
IC2F [3: 0]				IC2PSC [1: 0]				ICIF [3: 0]				ICIPSC [1: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

20.3.7.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	OC2M[3]	RW	0	See OC2M [2: 0] description for details
23: 17	Reserved	-	-	-
16	OC1M[3]	RW	0	See OC1M [2: 0] description for details
15	OC2CE	RW	0	Output Compare 2 Clear Enable (Output Compare 2 clear enable)
14: 12	OC2M	RW	3'h0	Output Comparison 2 Mode Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 Preload Enable (Output Compare 2 preload enable)
10	OC2FE	RW	0	Output Compare 2 Fast Enable (Output Compare 2 fast enable)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection Capture/Compare 2 selection This bit defines the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	Output Comparison 1 Mode Output Compare 1 mode These bits define the action of outputting the reference signal OC1REF, and OC1REF determines the values of OC1, OC1N. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high.

				0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 0011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 0100: Force inactive level OC1REF is forced low. 0101: Force active level OC1REF is forced high. 0110: PWM Mode 1-On up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 0111: PWM Mode 2-On up count, channel 1 is invalid level once TIMx_CNT < TIMx_CCR1, active level otherwise; When counting down, channel 1 is active once TIMx_CNT > TIMx_CCR1, otherwise it is invalid. 1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes invalid again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In the decrement count mode, the channel is in an active state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update 1010: Reserved 1011: Reserved 1100: Combined PWM Mode 1-OC1REF has the same behavior as in PWM Mode 1, OC1REFC is composed of OC1REF and OC2REF or logic 1101: Combined PWM Mode 2-OC1REF has the same behavior as in PWM Mode 2, OC1REFC is composed of OC1REF and OC2REF AND logic 1110: Asymmetric PWM Mode 1: OC1REF has the same behavior as in PWM Mode 1, OC1REFC outputs OC1REF when counting up and OC2REF when counting down 1111: Asymmetric PWM Mode 2: OC1REF has the same behavior as in PWM Mode 2, OC1REFC outputs OC1REF when counting up and OC2REF when counting down
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives.
2	OC1FE	RW	0	Output Compare 1 Fast Enable (Output Compare 1 fast enable) This bit is used to speed up the output response to the triggering input event. Must be used in single pulse mode (OPM position 1 of the TIMx_CR1 register) to achieve fast pulse output when the trigger comes. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. Only available for PWM Mode 1 and PWM Mode 2
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register).

				Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

20.3.7.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 12	IC2F	RW	4'h0	Input Capture 2 Filter (Input capture 2 filter)
11: 10	IC2PSC	RW	2'h0	Input/Capture 2 Prescaler (Input capture 2 prescaler)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

20.3.8 TIMx Capture/Compare Mode Control Register 2 (TIMx_CCMR2) (x = 2/3/4/5/18)

Address offset: 0x1C

Reset value: 0x0000 0000

See the above description of the CCMR1 register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							OC4M[3]	Res.							OC3M[3]
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M [2: 0]			OC4PE	OC4FE	CC4S [1: 0]		OC3CE	OC3M [2: 0]			OC3PE	OC3FE	CC3S [1: 0]	
IC4F [3: 0]				IC4PSC [1: 0]				IC3F [3: 0]				IC3PSC [1: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

20.3.8.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	OC4M[3]	RW	0	See OC4M [2: 0] description for details
23: 17	Reserved	-	-	-
16	OC3M[3]	RW	0	See OC3M [2: 0] description for details
15	OC4CE	RW	0	Output Compare 4 Clear Enable (Output Compare 4 clear enable)
14: 12	OC4M	RW	3'h0	Output Comparison 4 Mode Output Compare 4 mode
11	OC4PE	RW	0	Output Comparison 4 Preload Enable (Output Compare 4 preload enable)
10	OC4FE	RW	0	Output Comparison 4 Fast Enable (Output Compare 4 fast enable)
9: 8	CC4S	RW	2'h0	Capture/Compare 4 selection Capture/Compare 4 selection The 2 bits define the direction of the channel (input/output), and the selection of the input pins: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7	OC3CE	RW	0	Output Compare 3 clear '0' enable (Output Compare 1clear enable)
6: 4	OC3M	RW	3'h0	Output Comparison 3 Mode (Output Compare 3 mode) These bits define the action of outputting the reference signal OC3REF, and OC3REF determines the values of OC3, OC3N. OC3REF is active high whereas OC3 and OC3N active level depends on CC3P and CC3NP bits. 0000: Frozen. The comparison between the output comparison register TIMx_CCR3 and the counter TIMx_CNT has no effect on OC3REF; 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR3), OC3REF is forced to be high. 0010: Set channel 1 to inactive level on match. OC3REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR3). 0011: Toggle OC3REF toggles when TIMx_CNT=TIMx_CCR3. 0100: Force inactive level OC3REF is forced low. 0101: Force active level OC3REF is forced high. 0110: PWM Mode 1-On up count, channel 3 is active once TIMx_CNT < TIMx_CCR3, otherwise invalid level; On counting down, channel 3 is an inactive level (OC3REF = 0) once TIMx_CNT > TIMx_CCR3, otherwise it is an active level (OC3REF = 1). 0111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR3 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR3 else inactive. 1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes inactive again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In the decrement count mode, the channel is in an active state until a trigger event (tim_trgi signal) is

				detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update 1010: Compare pulse: generate pulse on OC3REF when CCR3 match event occurs, pulse configurable by PWPRSC [2: 0] and PW [7: 0] of TIMx_ECR 1011: Directional output: OC3REF signal covered by DIR bit 1100: Combined PWM Mode 1-OC3REF has the same behavior as in PWM Mode 1, OC3REFC is composed of OC3REF and OC4REF or logic 1101: Combined PWM Mode 2-OC3REF has the same behavior as in PWM Mode 2, OC3REFC is composed of OC3REF and OC4REF AND logic 1110: Asymmetric PWM Mode 1: OC3REF has the same behavior as in PWM Mode 1, OC3REFC outputs OC3REF when counting up and OC4REF when counting down 1111: Asymmetric PWM Mode 2: OC3REF has the same behavior as in PWM Mode 2, OC3REFC outputs OC3REF when counting up and OC4REF when counting down
3	OC3PE	RW	0	Output Comparison 3 Preload Enable (Output Compare 3 preload enable)
2	OC3FE	RW	0	Output Comparison 3 Fast Enable (Output Compare 3 fast enable)
1: 0	CC3S	RW	2'h0	Capture/Compare 3 selection Capture/Compare 3 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

20.3.8.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 12	IC4F	RW	4'h0	Input Capture 4 Filter (Input capture 4 filter)
11: 10	IC4PSC	RW	2'h0	Input/Capture 4 Prescaler (Input capture 4 prescaler
9: 8	CC4S	RW	2'h0	Capture/Compare 4 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC4S is writable only when the channel is closed (CC4E = 0 in the TIMx_CCER register).
7: 4	IC3F	RW	4'h0	Input Capture 3 Filter (Input capture 3 filter)
3: 2	IC3PSC	RW	2'h0	Input/Capture 3 Prescaler (Input capture 3 prescaler)
1: 0	CC3S	RW	2'h0	Capture/Compare 3 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3. 10: CC3 channel is configured as input, IC3 is mapped on TI4. 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx_CCER register).

20.3.9 TIMx capture/compare enable register (TIMx_CCER) (x = 2/3/4/5/18)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	CC4NP	RW	0	Input/capture 4 complementary output polarity Capture/Compare 3 output polarity Refer to CC1NP description.
14	Reserved	-	-	-
13	CC4P	RW	0	Input/Capture 4 Output Polarity Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 Output Enable (Capture/Compare 4 output enable) Refer to CC1E description.
11	CC3NP	RW	0	Input/capture 3 complementary output polarity Capture/Compare 3 output polarity Refer to CC1NP description.
10	Reserved	-	-	-
9	CC3P	RW	0	Input/capture 3 output polarity Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 Output Enable (Capture/Compare 3 output enable) Refer to CC1E description.
7	CC2NP	RW	0	Input/capture 2 complementary output polarity (Capture/Compare 2 output polarity) Refer to CC1NP description.
6	Reserved	-	-	-
5	CC2P	RW	0	Input/capture 2 output polarity (Capture/Compare 2 output polarity) Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 Output Enable (Capture/Compare 2 output enable) Refer to CC1E description.
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low.
2	Reserved	-	-	-
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode.
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output, so the output level of OC1 depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits.

				1: ON-The OC1 signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Notes: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 20-5 Control bits for complementary output channels OCx and OCxN with brake function

CCxE bit	OCx output status
0	Output disabled (disconnected from timer) OCx = 0, OCx_EN = 0
1	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1

20.3.10 TIMx counter (TIMx_CNT) (x = 2)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIFCPY	CNT [30:16]														
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	UIFCPY	RW	0	If UIFREMAP = 0 CNT [31]: most significant bit of counter value If UIFREMAP = 1 UIFCPY: UIF Copy This bit is a read-only copy of the UIF bit of the TIMx_ISR register
30: 0	CNT	RW	31'h0	Counter value

20.3.11 TIMx Counter (TIMx_CNT) (x = 3/4/5/18)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIFCPY RW	Res														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	UIFCPY	RW	0	If UIFREMAP = 0 CNT [31]: most significant bit of counter value If UIFREMAP = 1 UIFCPY: UIF Copy This bit is a read-only copy of the UIF bit of the TIMx_ISR register
30: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

20.3.12 TIMx Prescaler (TIMx_PSC) (x = 2/3/4/5/18)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes the counter being cleared '0' by the UG bit of TIM_EGR or cleared '0' by the slave controller operating in reset mode.

20.3.13 TIMx Auto Reload Register (TIMx_ARR) (x = 2)

Address offset: 0x2C

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	ARR	RW	32'hFFFFFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null. No jitter mode (DITHEN = 0): This register holds that auto-load value Dither Mode (DITHEN = 1): The register holds an integer portion ARR [31: 4], and ARR [3: 0] contains a dither portion

20.3.14 TIMx Auto Reload Register (TIMx_ARR) (x = 3/4/5/18)

Address offset: 0x2C

Reset value: 0x0FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												ARR [19:16]			
												RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	ARR	RW	20'h0FFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null. No jitter mode (DITHEN = 0): This register holds that auto-load value Dither Mode (DITHEN = 1): The register holds an integer portion ARR [19: 4], and ARR [3: 0] contains a dither portion

20.3.15 TIMx capture/compare register 1 (TIMx_CCR1) (x = 2)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1 [31: 16]															
RW/RO															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

CCR1 [15: 0]
RW/RO

Bit	Name	R/W	Reset Value	Function
31: 0	CCR1	RW/RO	32'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): This register holds an integer portion CCR1 [31: 4], and CCR1 [3: 0] contains a dither portion If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR1 [31: 4] and the CCR1 [3: 0] bit is reset

20.3.16 TIMx capture/compare register 1 (TIMx_CCR1) (x = 3/4/5/18)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR1 [19:16]			
												RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR1	RW/RO	20'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): The register holds an integer portion CCR1 [19: 4], and CCR1 [3: 0] contains a dither portion If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR1 [19: 4], and the CCR1 [3: 0] bit is reset

20.3.17 TIMx capture/compare register 2 (TIMx_CCR2) (x = 2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2 [31: 16]															
RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2 [15: 0]															
RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO	RW/RO

Bit	Name	R/W	Reset Value	Function
31: 0	CCR2	RW/RO	32'h0	Capture/Compare 2 value for channel 2 If channel CC2 is configured as output: CCR2 contains the value loaded into the current capture/compare 2 register (preload value). If the preload function is not selected in the TIMx_CCMR2 register (OC2PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): The register holds an integer portion CCR2 [31: 4], and CCR2 [3: 0] contains a dither portion If channel CC2 is configured as input: CCR2 contains the counter value transmitted by the last input capture 2 event (IC2). TIMx_CCR2 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR2 [31: 4], and the CCR2 [3: 0] bit is reset

20.3.18 TIMx capture/compare register 2 (TIMx_CCR2) (x = 3/4/5/18)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR2 [19:16]			
												RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR2	RW/RO	20'h0	Capture/Compare 2 value for channel 2 If channel CC2 is configured as output: CCR2 contains the value loaded into the current capture/compare 2 register (preload value). If the preload function is not selected in the TIMx_CCMR2 register (OC2PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): The register holds an integer portion CCR2 [19: 4], and CCR2 [3: 0] contains a dither portion If channel CC2 is configured as input: CCR2 contains the counter value transmitted by the last input capture 2 event (IC2). TIMx_CCR2 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR2 [19: 4], and the CCR2 [3: 0] bit is reset

20.3.19 TIMx capture/compare register 3 (TIMx_CCR3) (x = 2)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3 [31: 16]															
RW/RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 0	CCR3	RW/RO	32'h0	Capture/Compare 3 value for channel 3 If channel CC3 is configured as output: CCR3 contains the value loaded into the current capture/compare 3 register (preload value). If the preload function is not selected in the TIMx_CCMR3 register (OC3PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): This register holds an integer portion CCR3 [31: 4], and CCR3 [3: 0] contains a dither portion If channel CC3 is configured as input: CCR3 contains the counter value transmitted by the last input capture 3 event (IC3). TIMx_CCR3 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR3 [31: 4], and the CCR3 [3: 0] bit is reset

20.3.20 TIMx capture/compare register 3 (TIMx_CCR3) (x = 3/4/5/18)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR3 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR3	RW/RO	20'h0	Capture/Compare 3 value for channel 3 If channel CC3 is configured as output: CCR3 contains the value loaded into the current capture/compare 3 register (preload value). If the preload function is not selected in the TIMx_CCMR3 register (OC3PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): The register holds an integer portion CCR3 [19: 4], and CCR3 [3: 0] contains a dither portion If channel CC3 is configured as input:

				CCR3 contains the counter value transmitted by the last input capture 3 event (IC3). TIMx_CCR3 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR3 [19: 4], and the CCR3 [3: 0] bit is reset

20.3.21 TIMx capture/compare register 4 (TIMx_CCR4) (x = 2)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4 [31: 16]															
RW/RO															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 0	CCR4	RW/RO	32'h0	Capture/Compare 4 value for channel 4 If channel CC4 is configured as output: CCR4 contains the value loaded into the current capture/compare 4 register (preload value). If the preload function is not selected in the TIMx_CCMR4 register (OC4PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. No jitter mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): This register holds an integer portion CCR4 [31: 4], and CCR4 [3: 0] contains a dither portion If channel CC4 is configured as input: CCR4 contains the counter value transmitted by the last input capture 4 event (IC4). TIMx_CCR4 Register Read Only No jitter mode (DITHEN = 0): This register holds that capture value Dither Mode (DITHEN = 1): This register holds the capture value to CCR4 [31: 4], and the CCR4 [3: 0] bit is reset

20.3.22 TIMx capture/compare register 4 (TIMx_CCR4) (x = 3/4/5/18)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR4 [19:16]			
												RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR4	RW/RO	20'h0	Capture/Compare 4 value for channel 4 If channel CC4 is configured as output: CCR4 contains the value loaded into the current capture/compare 4 register (preload value). If the preload function is not selected in the TIMx_CCMR4 register (OC4PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 4 register when an update event occurs.

				The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. Jitter-free mode (DITHEN = 0): This register holds the comparison value. Dither Mode (DITHEN = 1): The register holds an integer portion CCR4 [19: 4], and CCR4 [3: 0] contains a dither portion If channel CC4 is configured as input: CCR4 contains the counter value transmitted by the last input capture 4 event (IC4). TIMx_CCR4 Register Read Only Jitter-free mode (DITHEN = 0): This register holds the capture value Jitter Mode (DITHEN = 1): This register holds the capture value to CCR4 [19: 4], the CCR4 [3: 0] bit is reset

20.3.23 TIMx Encoder Control Register (TIMx_ECR) (x = 2/3/4/5/18)

Address offset: 0x58

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					PWPRSC [2: 0]			PW [7: 0]							
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								IPOS		FIDX	Res		IDIR		IE
								RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 27	Reserved	-	-	-
26: 24	PWPRSC	RW	3'h0	Pulse width prescalation Pulse width prescaler This bit sets the clock prescalation of the pulse generator, and the formula is as follows: t_{PWG}= xCK $2^{(PWPRSC[2:\ 0])}$t_INT
23: 16	PW	RW	8'h0	Pulse width Pulse width This bit defines the pulse duration as follows: t_{PW}= PW [7: 0] xt_{PWG}
15: 8	Reserved	-	-	-
7: 6	IPOS	RW	2'h0	Index positioning Index positioning In orthogonal encoder mode (SMS [3: 0] = 0001, 0010, 0011, 1110, 1111), this bit is used to indicate in which configuration the AB input index event resets the counter 00: When AB = 00, the index resets the counter 01: When AB = 01, the index resets the counter 10: When AB = 10, the index resets the counter 11: When AB = 11, the index resets the counter In directional clock mode or clock + directional mode (SMS [3: 0] = 1010, 1011, 1100, 1101), this bit segment is used to indicate which level index will reset the counter. In directional clock mode, clock inputs are applied to both ways. x0: When clock is 0, index resets counter x1: When clock is 1, index resets counter Note: IPOS [1] bit is invalid
5	FIDX	RW	0	First index This bit indicates whether only the first index is considered 0: Index always valid 1: Only the first index can reset the counter
4: 3	Reserved	-	-	-
1	IDIR	RW	0	Index direction Index direction This bit indicate in which count direction the index event resets the counter 00: Index can reset counter regardless of counting direction 01: Index can reset counter only when counting up 10: Index can reset counter only when counting down 11: Reserved
0	IE	RW	0	Index enable Index enable This bit indicates whether the index resets the counter 0: Index closed 1: Index enable

20.3.24 TIMx input select register (TIMx_TISEL) (x = 2/3/4/5/18)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				TI4SEL [3: 0]				Res				TI3SEL [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				TI2SEL [3: 0]				Res				TI1SEL [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27:24	TI4SEL	RW	4'h0	TI4 input selection (tim_TI4 [15: 0] input) 0000: TIMx_CH4 0001:tim _ti4_in1 0010:tim _ti4_in2 Other: Reserved
23:20	Reserved	-	-	-
19:16	TI3SEL	RW	4'h0	TI3 input selection (tim_TI3 [15: 0] input) 0000: TIMx_CH3 0001:tim _ti3_in1 Other: Reserved
15:12	Reserved	-	-	-
11: 8	TI2SEL	RW	4'h0	TI2 input selection (tim_TI2 [15: 0] input) 0000: TIMx_CH2 0001:tim _ti2_in1 0010:tim _ti2_in2 0011:tim _ti2_in3 0100:tim _ti2_in4 Other: Reserved
7: 4	Reserved	-	-	-
3: 0	TI1SEL	RW	4'h0	TI1 input selection (tim_TI1 [15: 0] input) 0000: TIMx_CH1 0001:tim _ti1_in1 0010:tim _ti1_in2 0011:tim _ti1_in3 0100:tim _ti1_in4 Other: Reserved

20.3.25 TIMx Alternate Function Option Register 1 (TIMx_AF1) (x = 2/3/4/5/18)

Address offset: 0x60

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														ETRSEL [3: 2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL [1: 0]		Res													
RW	RW														

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 14	ETRSEL	RW	4'h0	External trigger source selection (etr_in source selection) This bit segment is used to select an ETR input source 0000: TIMx_ETR 0001:tim_etr1 0010:tim_etr2 0011:tim_etr 3 0100:tim_etr 4 0101:tim_etr 5 0110:tim_etr 6 0111:tim_etr 7 1000:tim_etr 8 1001:tim_etr 9 1010:tim_etr 10

				1011:tim_etr 11 1100:tim_etr 12 1101:tim_etr 13 1110:tim_etr 14 1111:tim_etr 15
13: 0	Reserved	-	-	-

20.3.26 TIMx Alternate Function Option Register 2 (TIMx_AF2) (x = 2/3/4/5/18)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res													OCRSEL [2: 0]		
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															

Bit	Name	R/W	Reset Value	Function
31: 19	Reserved	-	-	-
18:16	OCRSEL	RW	3'h0	OCREF Reset source selection (OCREF_clr source selection) This bit segment is used to select that ocref_clr input source 0000: ocref_clr0 0001: ocref_clr1 0010 ocref_clr2 0011 ocref_clr3 Others: Reserved
15: 0	Reserved	-	-	-

20.3.27 TIMx DMA Control Register (TIMx_DCR) (x = 2/3/4/5/18)

Address offset: 0x3DC

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DBL [4: 0]					Res			DBA [4: 0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12: 8	DBL	RW	5'h0	DMA burst length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register, the timer makes a continuous transfer), i.e. defines the number of transfers, which can be half words (double bytes) or bytes: 00000: 1 transfer 00001: 2 transfers 00010: 3 transmissions..... ..... 11001: 26 transfers Example: We consider such a transmission: DBL = 7 bytes, DBA = TIMx_CR1 If DBL = 7 bytes and DBA = TIMx_CR1 represents the address of the data to be transmitted, then the transmitted address is given by: (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL Where (address of TIMx_CR1) + DBA plus 7 gives the address where data will be written or read out, so that the transfer of data will occur in 7 registers starting from address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following can occur: -If the data is set to a half word (16 bits), then the data will be transferred to all 7 registers.

				-If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, for the timer, the user must specify the data width to be transmitted by the DMA.
7: 5	Reserved	-	-	-
4: 0	DBA	RW	5'h0	DMA base address These bits define the base address of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register), and the DBA is defined as the offset from the address where the TIMx_CR1 register is located: 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ……

20.3.28 DMA address for TIMx continuous mode (TIMx_DMAR) (x = 2/3/4/5/18)

Address offset: 0x3E0

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DMAB	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA index) x4 where: The "TIMx_CR1 address" is an address where the control register 1 (TIMx_CR1) is located; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

21. General Purpose Timer (TIM9/TIM12)

21.1 Introduction

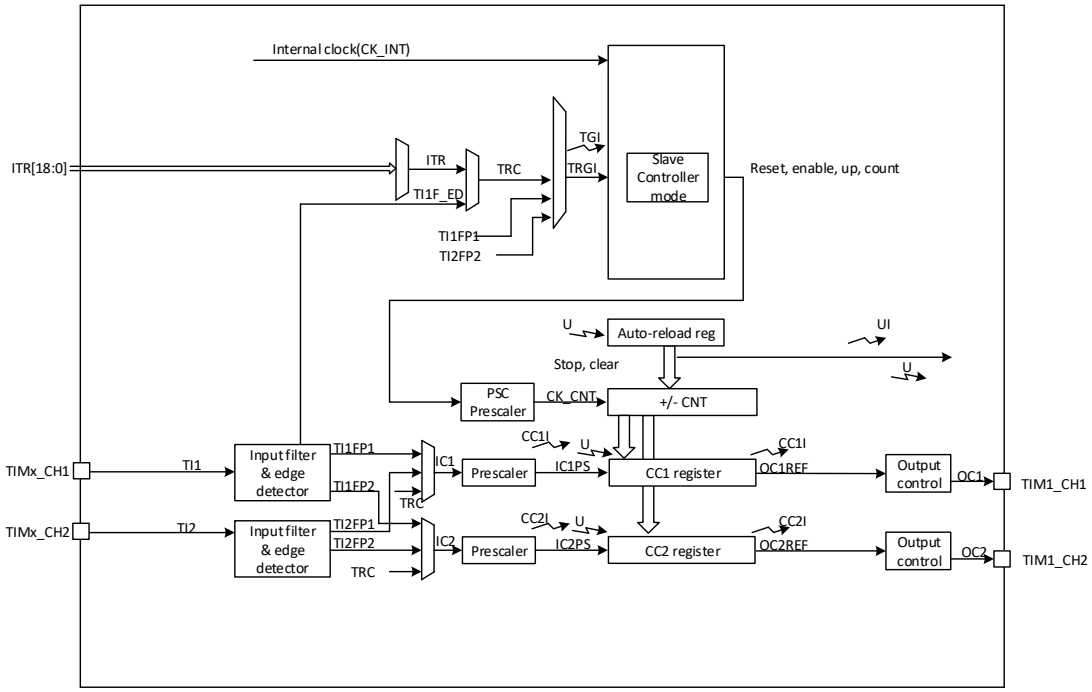
The universal timers (TIM9 and TIM12) consist of a 16-bit auto-load counter driven by a programmable prescaler. It is suitable for a variety of uses, including measuring the pulse width of the input signal (input capture), or generating the output waveform (output comparison, PWM). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The general purpose timers TIM9 and TIM12 are completely independent and they do not share any resources. They can be synchronized together.

21.1.1 TIM9 and TIM12 Main Features

The functions of the TIM9 and TIM12 timers include:

- 16-bit up auto-load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- Up to 2 independent channels:
  - Input capture
  - Output compare
  - PWM generation (edge alignment mode)
  - One-pulse mode output
- Synchronization circuit for controlling timer and interconnection between timer by external signal
- An interrupt occurs when the following events occur:
  - Update: Counter overflow, counter initialization (via software or internal trigger)
  - Trigger event (counter starts, stops, initializes, or counts by internal trigger)
  - Input capture
  - Output compare

21.1.2 CAN block diagram



21.2 TIM9/TIM12 functional description

21.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. This counter can count up. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition (overflow or underflow) and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set. (See the description of the slave mode controller for more details on enabling counters).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on

the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

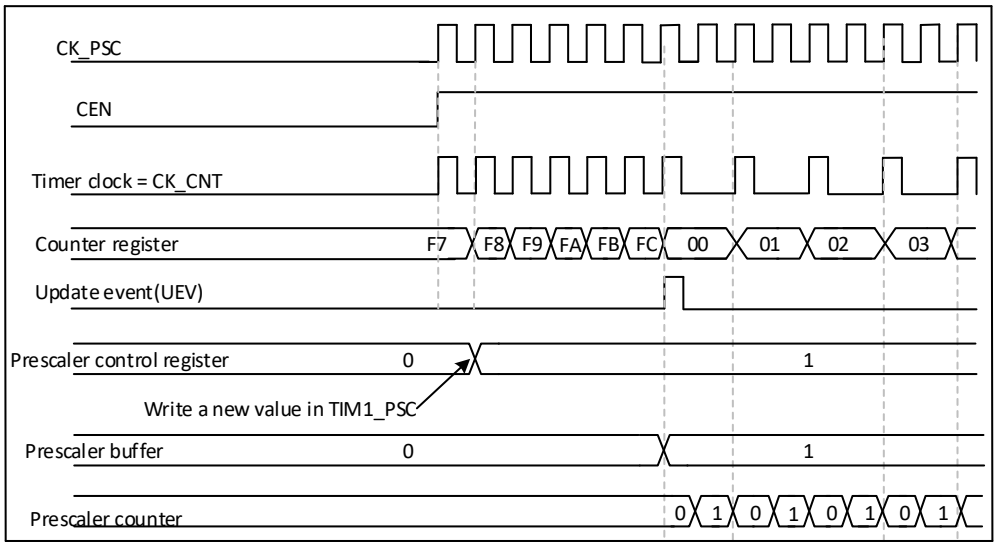


Figure 21-1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

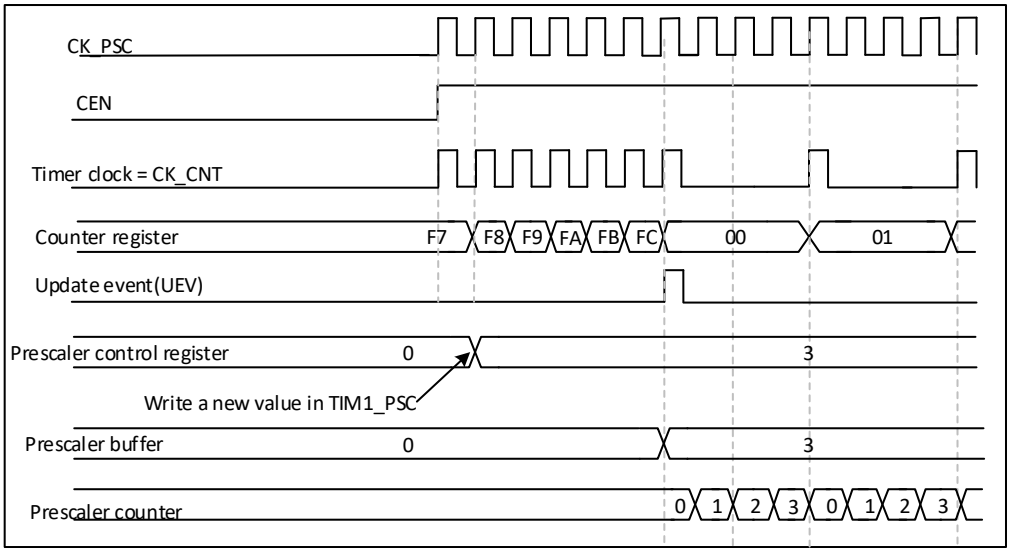


Figure 21-2 Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

21.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

In addition, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

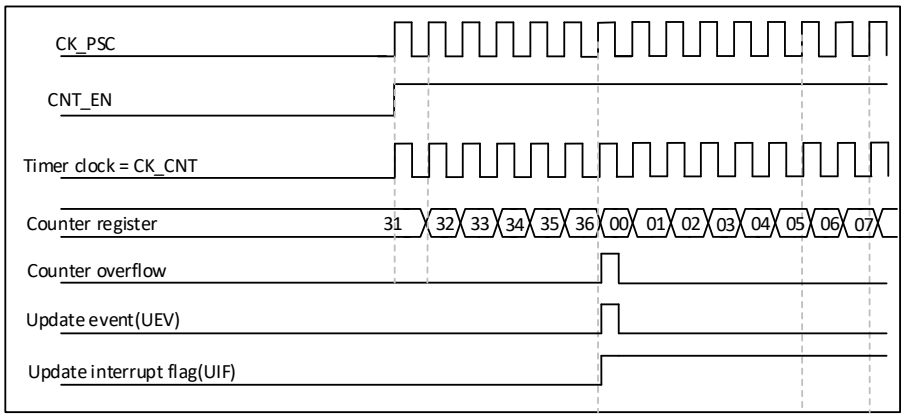


Figure 21-3 counter timing diagram with internal clock division factor of 1

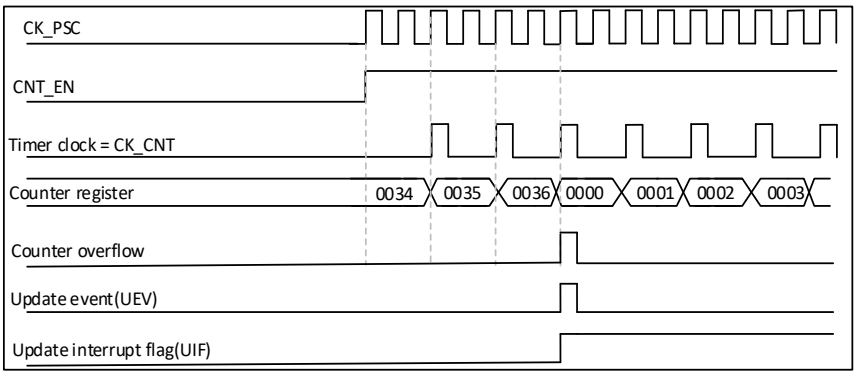


Figure 21-4 counter timing diagram with internal clock division factor of 2

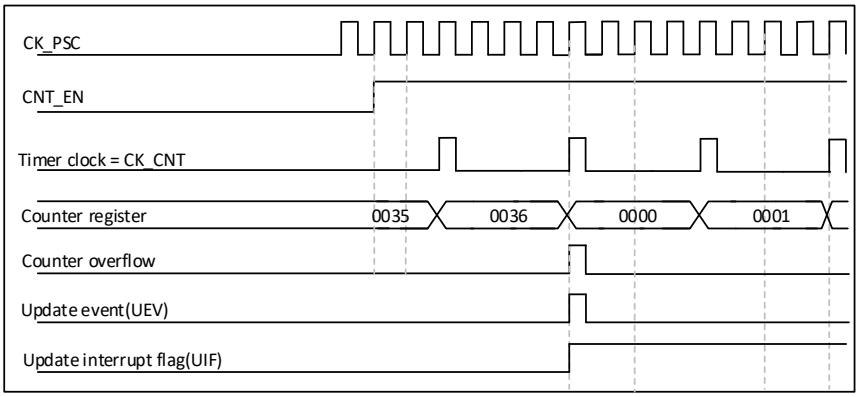


Figure 21-5 Counter timing diagram, internal clock divided by 4

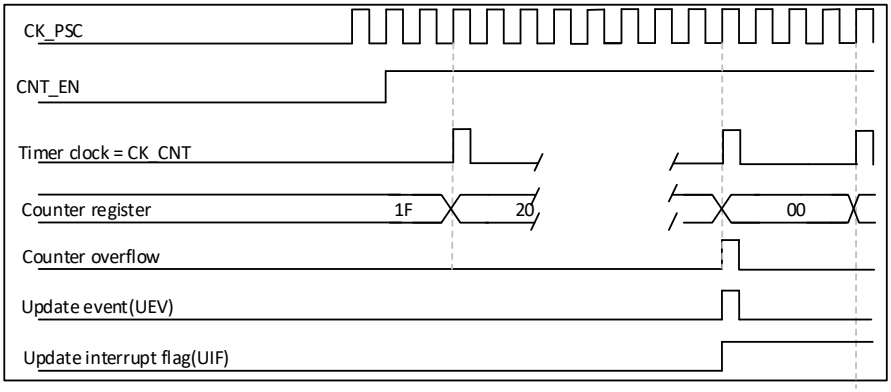


Figure 21-6 Counter timing diagram, internal clock divided by N

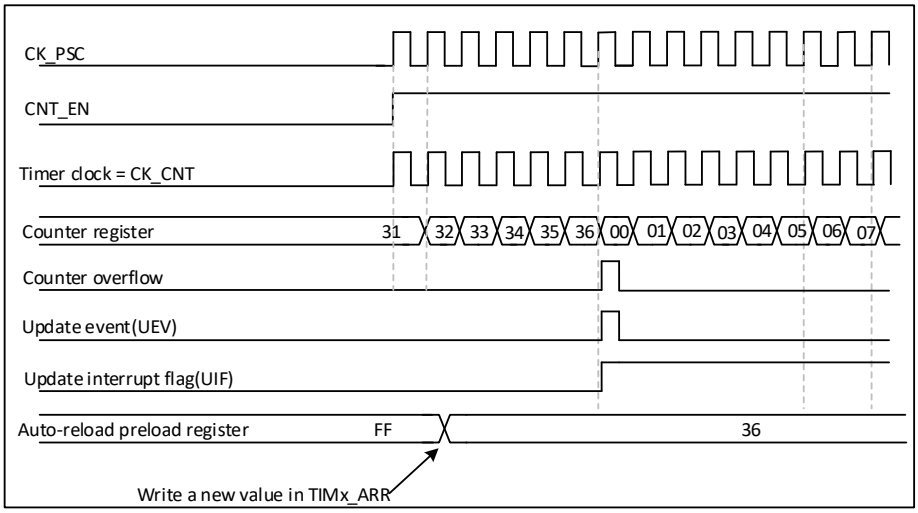


Figure 21-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

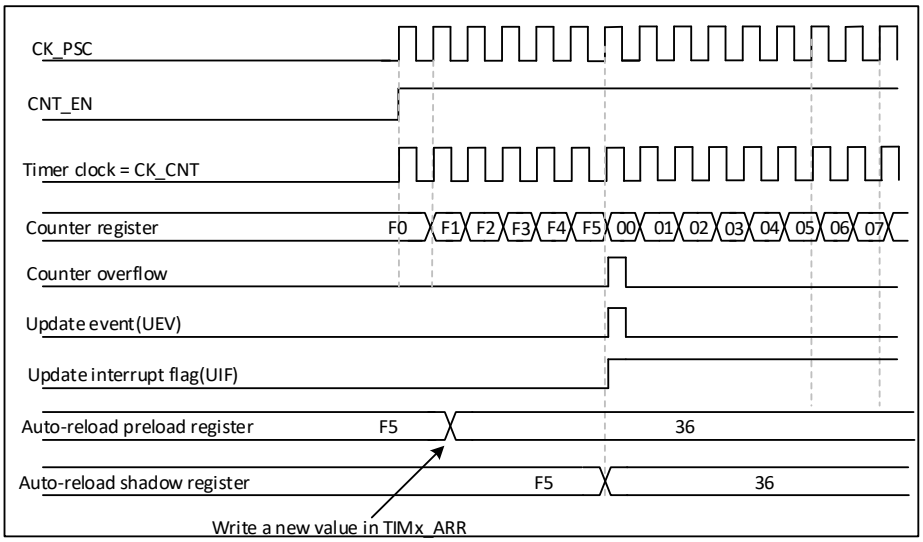


Figure 21-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

21.2.3 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pin (TIx)
- Internal Trigger Input (ITRx): Uses one timer as a prescaler for the other.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 000), the CEN and UG bits (TIMx_EGR register) are de facto control bits and can only be modified by software (the UG bits are still automatically cleared). As long as the CEN bit is written as' 1 ', the clock of the prescaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

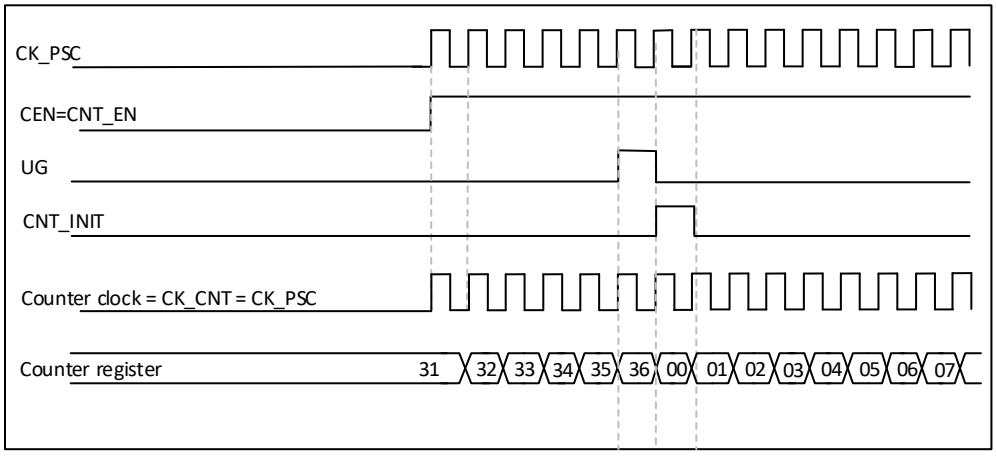


Figure 21-9 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

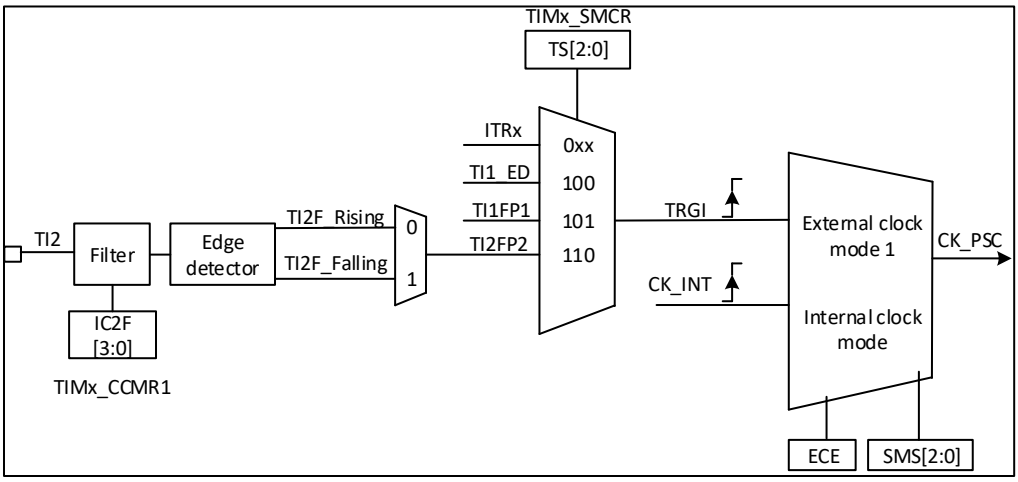


Figure 21-10 TI2 Example of external clock connection

For example, to configure the counter to count up on the rising edge of the TI2 input, use the following steps:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = '01' in the TIMx_CCMR1 register.

2. Configure IC2F [3: 0] of the TIMx_CCMR1 register and select the input filter bandwidth (if no filter is needed, keep IC2F = 0000);

3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.

4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=00110 in the TIMx_SMCR register;

6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the synchronization circuit at the input of TI2.

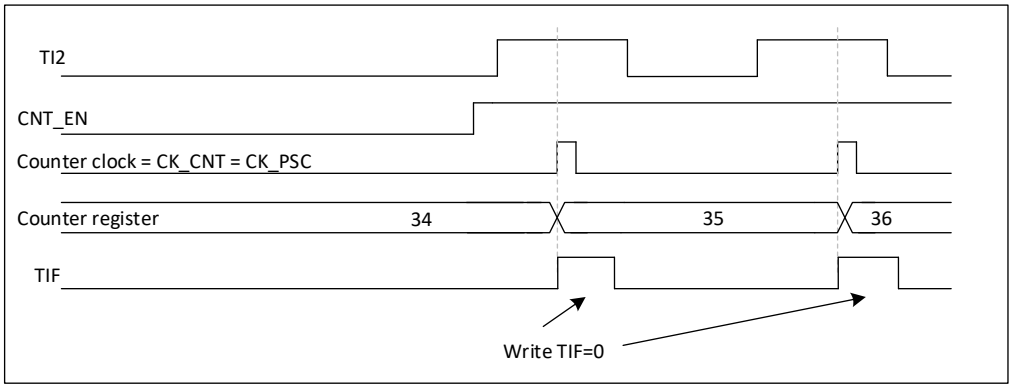


Figure 21-11 Control circuit in external clock mode 1

21.2.4 Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures are a capture/compare channel overview.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge monitor with polarity selection then generates a signal (TIxFPx) that can be triggered as an input from the slave mode controller or as a capture control. It is prescaled before the capture register (ICxPS).

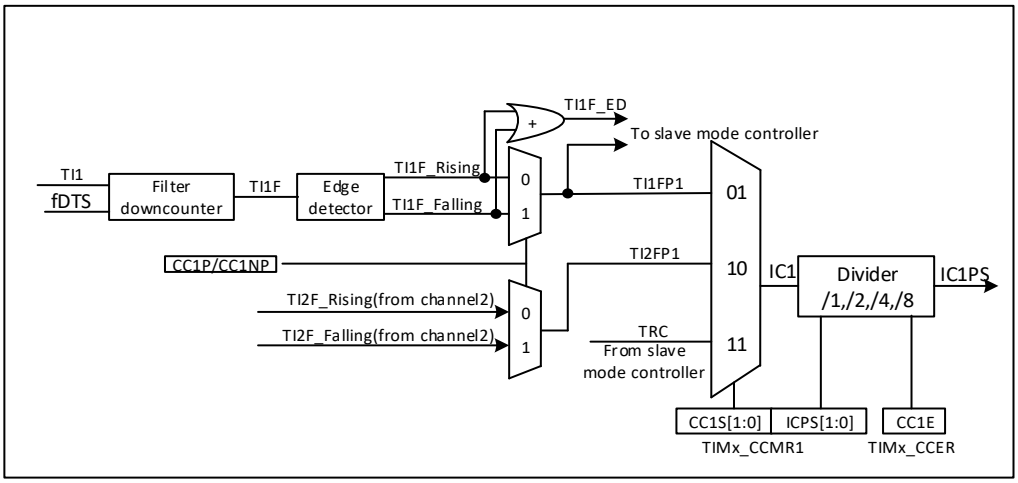


Figure 21-12 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

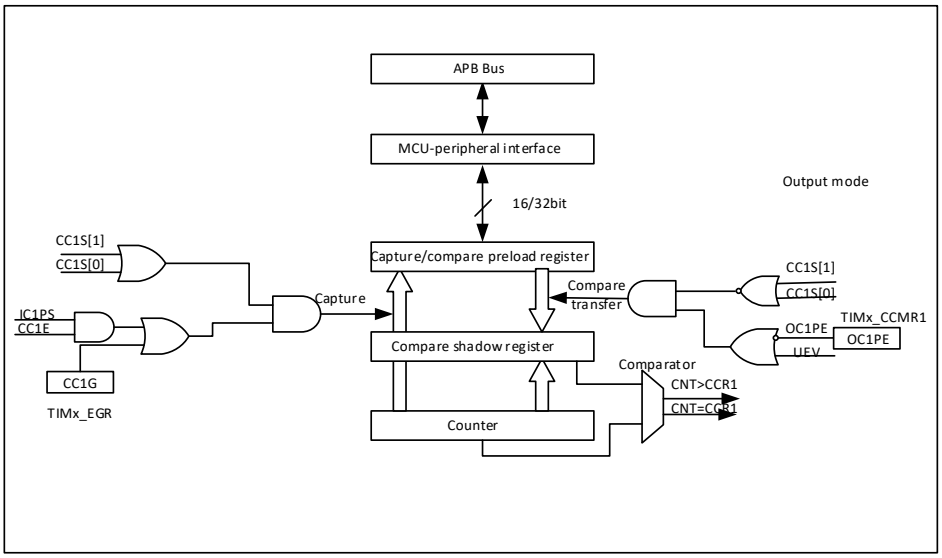


Figure 21-13 Capture/Compare the main circuit of channel 1

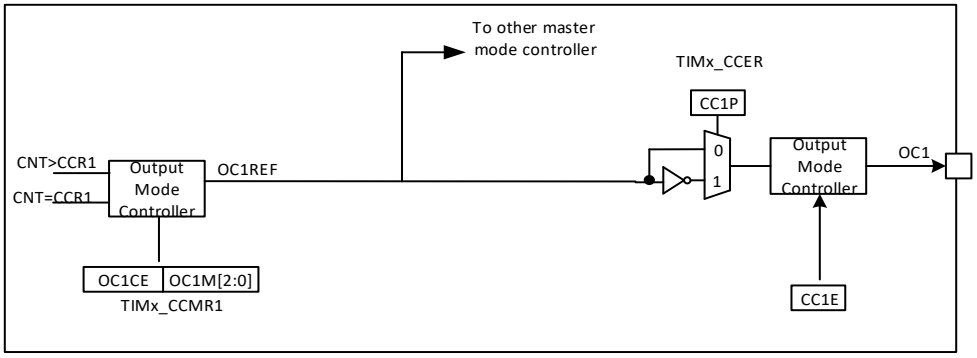


Figure 21-14 Output stage of capture/compare channel

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

21.2.5 Input capture mode:

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched into the capture/compare register (TIMx_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx_SR register) is set to 1. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (TIMx_SR register) is set to 1. The CCxIF can be cleared by writing CCxIF = 0, or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select valid input: TIMx_CCMR1 must be connected to the TI1 input, so CC1S = 01 written in the TIMx_CCMR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (set to rising edge) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If desired, the associated interrupt request may be allowed by setting the CC1IE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- The CC1IF flag bit is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

21.2.6 PWM input mode

This mode is a special case of the input capture mode, and the rest of the operation is the same as the input capture mode except for the following differences:

- Both ICx signals are mapped to the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, the user can measure the cycle (TIMx_CCR1 register) and the duty cycle (TIMx_CCR2 register) of the PWM signal input to TI1 as follows

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

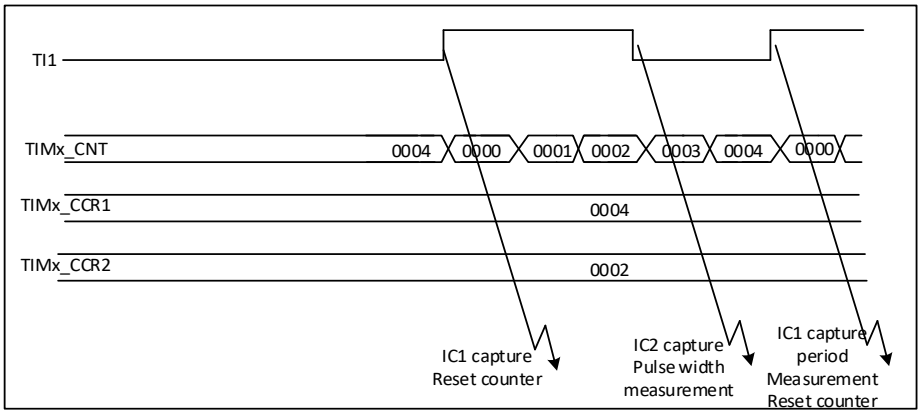


Figure 21-15 PWM input mode timing

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use the TIMx_CH1/TIMx_CH2 signal.

21.2.7 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, you just need to write101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

21.2.8 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has expired. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

You can select whether the TIMx_CCRx register needs to use a preload register by configuring the OCxPE bit in TIMx_CCMRx.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

— Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx

— Write OCxPE = 0 to disable preload register

— Write CCxP = 0 to select active high polarity

— Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated by software at any time to control the output waveform, provided the preload register is not used (OCxPE = '0', otherwise the shadow register of TIMx_CCRx can only be updated when the next update event occurs). An example is given in the figure below.

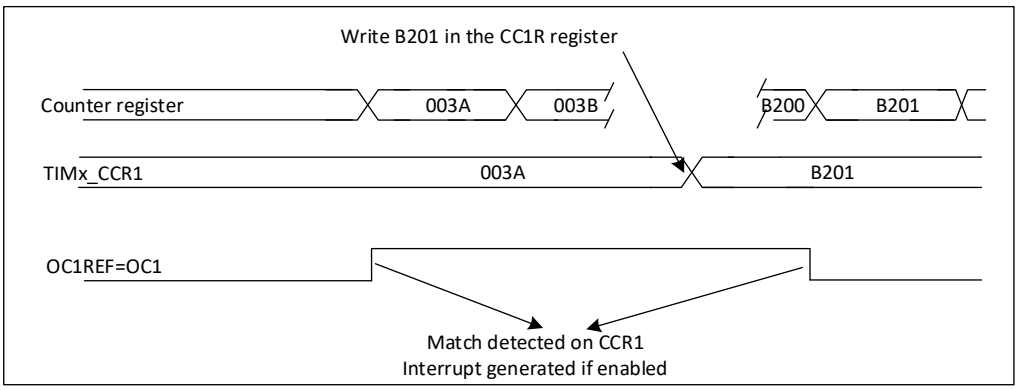


Figure 21-16 Output comparison mode, flip OC1

21.2.9 PWM mode

The pulse width modulation mode may produce a signal whose frequency is determined by the TIMx_ARR register and whose duty cycle is determined by the TIMx_CCRx register.

The OCxM bit in the TIMx_CCMRx register is written to '110' (PWM mode 1) or '111' (PWM mode 2), and each OCx output channel can be independently set to generate a PWM. The corresponding pre-load register must be enabled by setting the OCxPE bit of the TIMx_CCMRx register, and finally the ARPE bit of the TIMx_CR1 register to enable the auto-reload preload register.

The preload register can only be transferred to the shadow register when an update event occurs, so the user must initialize all registers by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. The output of OCx is enabled by CCxE control (in TIMx_CCER). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

PWM edge-aligned mode

- Upcounting configuration

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the comparison value in TIMx_CCRx is greater than the auto-reload value (TIMx_ARR), OCxREF remains' 1 '. If the comparison value is 0, then OCxREF remains' 0 '. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

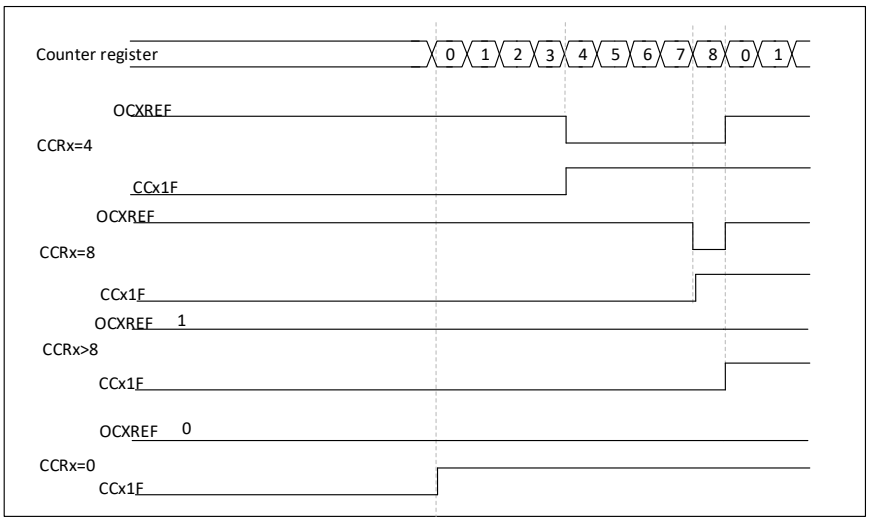


Figure 21-17 Edge-aligned PWM waveforms (ARR=8)

21.2.10 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV. A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up count mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx),

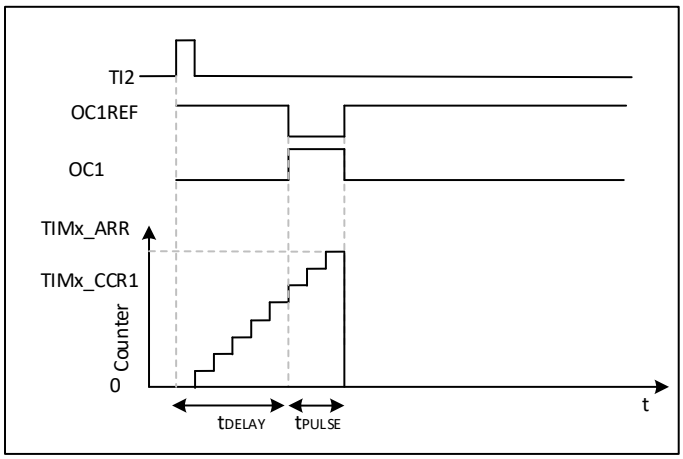


Figure 21-18 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Assume TI2FP2 as the trigger:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the

TIMx_SMCR register.

- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let's say one want to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '0' to '1' when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

21.2.10.1 Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay tDELAY min we can get. If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF (and OCx) is forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

21.2.11 Synchronization of TIMx timer and externally triggered

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gated mode, trigger mode.

21.2.11.1 Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; Meanwhile, if the UDIS bit of the TIMx_CR1 register is low, an update event UEV is also generated; All preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIMx_SMCR register. Select TI1 as the

input source by writing TS=00101 in TIMx_SMCR register.

- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, the trigger flag (TIF bit in the TIMx_SR register) is set, and the following figure shows the action when the auto-reload register TIMx_ARR = 0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

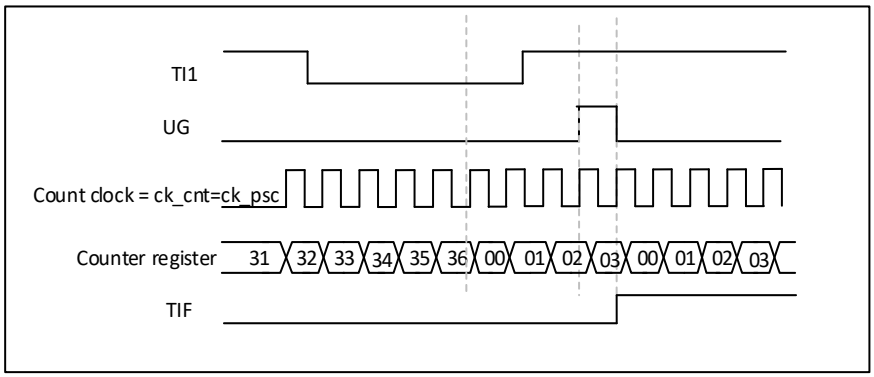


Figure 21-19 Control circuit in reset mode

21.2.11.2 Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. SelectTI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

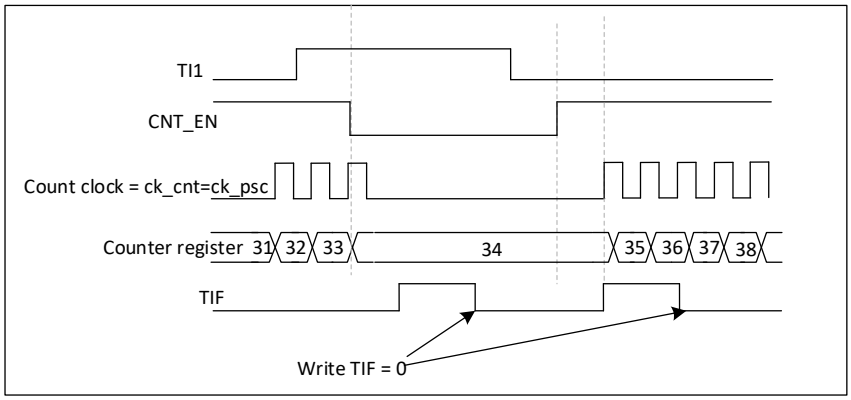


Figure 21-20 Control circuit in Gated mode

21.2.11.3 Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMx_CCMR1 register. Write CC2P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=00110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

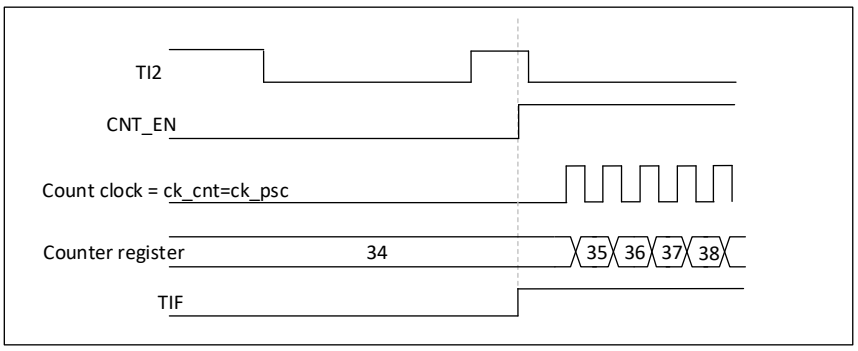


Figure 21-21 Control circuit in trigger mode

21.2.12 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. See Section 20.2.22: Timer Synchronization for details.

21.2.13 Debug mode

When the microcontroller enters the debug mode (Cortex-M4F core stops), the TIMx counter may either continue normal operation or stop according to the setting of DBG_TIMx_STOP in the DBG module. See subsequent DBG sections for details.

21.3 TIMx registers

21.3.1 TIM9 and TIM12 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CKD [1: 0]		ARPE	Res			OPM	URS	UDIS	CEN
						RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the frequency division ratio between the timer clock (CK_INT) frequency and dead time and the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 4	Reserved	-	-	-
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

21.3.2 TIM9 and TIM12 slave mode control registers (TIMx_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							TS [4: 0]					Res	SMS [2: 0]		
							RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Re-served	-	-	-
8: 4	TS	RW	5'h0	Trigger selection Trigger selection These 5 bits select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) 01010: Internal Trigger 6 (ITR6) 01011: Internal Trigger 7 (ITR7) 01100: Internal Trigger 8 (ITR8) 01101: Internal Trigger 9 (ITR9) 01110: Internal Trigger 10 (ITR10) 01111: Internal Trigger 11 (ITR11) 10000: Internal Trigger 12 (ITR12) 10001: Internal Trigger 13 (ITR13) 10010: Internal Trigger 14 (ITR14) 10011: Internal Trigger 15 (ITR15) 10100: Internal Trigger 16 (ITR16) 10101: Internal Trigger 17 (ITR17) 10110: Internal Trigger 18 (ITR18) Others: Reserved For more details about ITRx, see the System Cascade Table Note: These bits can only be changed when they are not used (e.g. SMS = 00000) to avoid erroneous edge detection when changing.
3	Re-served	-	-	-
2: 0	SMS	RW	3'h0	Select from Mode (Slave mode selection) When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 000: Slave mode disabled - if CEN = 1 then the prescaler is clocked directly by the internal clock. 001: Reserved 010: Reserved 011: Reserved 100: Reset mode-The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a signal to update the register. 101: Gated mode-When the trigger input (TRGI) is high, the clock of the counter is turned on. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode-the counter starts (but does not reset) on the rising edge of the trigger input TRGI, only the start of the counter is controlled. 111: External clock mode 1-rising edge of selected trigger input (TRGI) drives counter.

21.3.3 TIM9 and TIM12 DMA/Interrupt enable registers (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TIE	Res			CC2IE	CC1IE	UIE
									RW				RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	TIE	RW	0	Trigger interrupt enable (Trigger interrupt enable) 0: prohibit triggering interrupt; 1: Enable trigger interrupt.
5: 3	Reserved	-	-	-
2	CC2IE	RW	0	Allow capture/compare 2 interrupts (Capture/Compare 2 interrupt enable) 0: Disable capture/compare 2 interrupt; 1: Allow capture/compare 2 interrupts.
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable) 0: Disable capture/compare 1 interrupt; 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

21.3.4 TIM9 and TIM12 status registers (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TIF	Res			CC2IF	CC1IF	UIF
									RC_W0				RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	TIF	RC_W0	0	Trigger interrupt flag (Trigger interrupt flag) This position is' 1 'by hardware when a trigger event occurs (a valid edge is detected at the TRGI input when the slave mode controller is in a mode other than gated mode, or either edge in gated mode). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5: 3	Reserved	-	-	-
2	CC2IF	RC_W0	0	Capture/Compare 2 Interrupt Markers (Capture/Compare 2 interrupt flag) Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes high under the counter overflow condition in the up or up/down count mode, or under the counter underflow condition in the down count mode If channel TI1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag

				This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0, when the repetition counter value overflows or underflows (an update event occurs when the repetition counter = 0). − If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

21.3.5 TIM9 and TIM12 event generation registers (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TG	Res			CC2G	CC1G	UG
									W				W	W	W

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	TG	W	0	Trigger generation This bit is set '1' by the software to generate a trigger event, and '0' is automatically cleared by the hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5: 3	Reserved	-	-	-
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action 1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged).

21.3.6 TIM9 and TIM12 capture/compare mode control register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M [2: 0]			OC2PE	OC2FE	CC2S [1: 0]		OC1CE	OC1M [2: 0]			OC1PE	OC1FE	CC1S [1: 0]	
IC2F [3: 0]				IC2PSC [1: 0]				ICIF [3: 0]				ICIPSC [1: 0]			

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

21.3.6.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	OC2CE	RW	0	Output Compare 2 Clear Enable (Output Compare 2 clear enable)
14: 12	OC2M	RW	3'h0	Output Comparison 2 Mode Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 Preload Enable (Output Compare 2 preload enable)
10	OC2FE	RW	0	Output Compare 2 Fast Enable (Output Compare 2 fast enable)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection Capture/Compare 2 selection This bit defines the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	Output Comparison 1 Mode Output Compare 1 mode These bits define the action of the output reference signal OC1REF, and OC1REF determines OC1. OC1REF is high active, while the active level of OC1 depends on the CC1P bit. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high. 010: Set channel 1 to inactive level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be low. 011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110: PWM mode 1-on up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 else inactive.
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives.
2	OC1FE	RW	0	Output Compare 1 Fast Enable (Output Compare 1 fast enable) This bit is used to speed up the output response to the triggering input event. Must be used in single pulse mode (OPM position 1 of the TIMx_CR1 register) to achieve fast pulse output when the trigger comes. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. Only available for PWM Mode 1 and PWM Mode 2
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin:

				00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

21.3.6.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Re-served	-	-	-
15: 12	IC2F	RW	4'h0	Input Capture 2 Filter (Input capture 2 filter)
11: 10	IC2PSC	RW	2'h0	Input/Capture 2 Prescaler (Input capture 2 prescaler)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

21.3.7 TIM9 and TIM12 capture/compare enable registers (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
								RW		RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Re-served	-	-	-
7	CC2NP	RW	0	Input/capture 2 complementary output polarity (Capture/Compare 2 output polarity) Refer to CC1NP description.
6	Re-served	-	-	-
5	CC2P	RW	0	Input/capture 2 output polarity (Capture/Compare 2 output polarity) Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 Output Enable (Capture/Compare 2 output enable) Refer to CC1E description.
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low.
2	Re-served	-	-	-
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode.
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output, so the output level of OC1 depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: ON-The OC1 signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

				Notes: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 21-1 Control bits for complementary output channels OCx and OCxN with brake function

CCxE bit	OCx output status
0	Output disabled (disconnected from timer) OCx = 0, OCx_EN = 0
1	OCx = OCxREF + Polarity, OCx_EN = 1

21.3.8 TIM9 and TIM12 counters (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

21.3.9 TIM9 and TIM12 prescalers (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes a counter being The UG bit of TIM_EGR is cleared '0' or is cleared '0' by a slave controller operating in reset mode.

21.3.10 TIM9 and TIM12 Automatic Reload Registers (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Re-served	-	-	-
15: 0	ARR	RW	16'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null.

21.3.11 TIM9 and TIM12 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CCR1	RW/ RO	16'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only

21.3.12 TIM9 and TIM12 capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CCR2	RW/RO	16'h0	Capture/Compare 2 value for channel 2 If channel CC2 is configured as output: CCR2 contains the value loaded into the current capture/compare 2 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC2PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 contains the counter value transmitted by the last input capture 2 event (IC2). TIMx_CCR2 Register Read Only

22. General Purpose Timer (TIM10/TIM11/TIM13/TIM14/TIM19)

22.1 Introduction

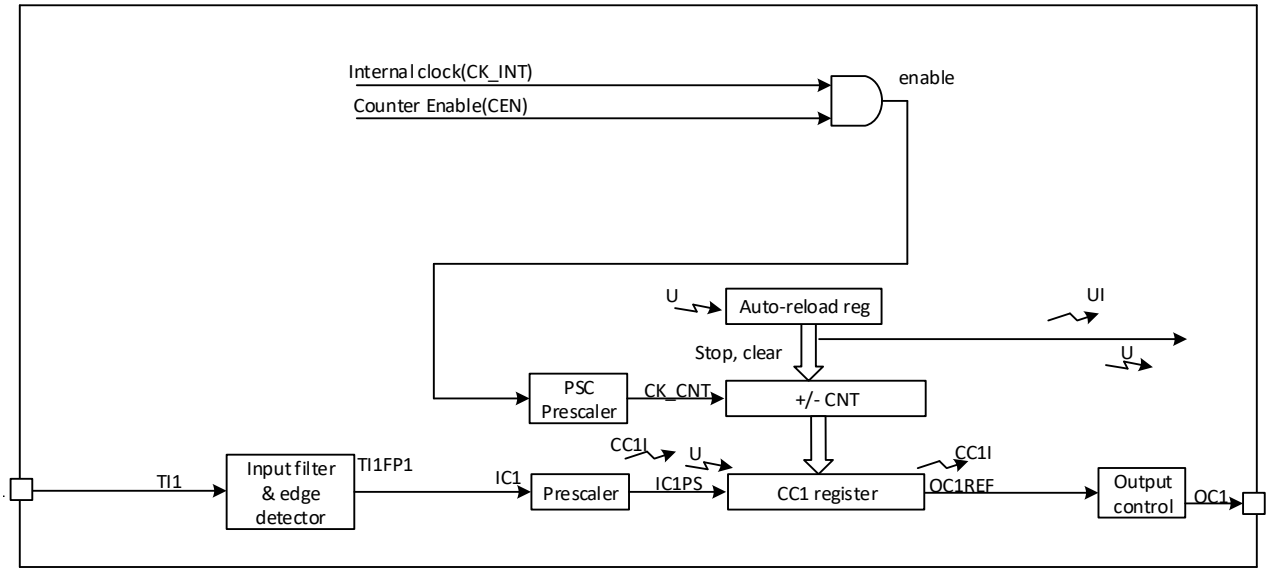
The general purpose timer (TIM10/TIM11/TIM13/TIM14/TIM19, hereinafter referred to as TIMx) consists of a 16-bit auto-load counter driven by a programmable prescaler. It is suitable for a variety of uses, including measuring the pulse width of the input signal (input capture), or generating the output waveform (output comparison, PWM). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. Universal timers (TIMx) are completely independent, they do not share any resources. They can be synchronized together.

22.1.1 TIMx Main Features

The functions of the TIMx timer include:

- 16-bit up auto-load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- 1 independent channel:
  - Input capture
  - Output compare
  - PWM generation (edge mode)
  - One-pulse mode output
- An interrupt occurs when the following events occur:
  - Counter overflow, counter initialization (via software)
  - Input capture
  - Output compare

22.1.2 CAN block diagram



22.2 TIMx functional description

22.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. This counter can count up. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

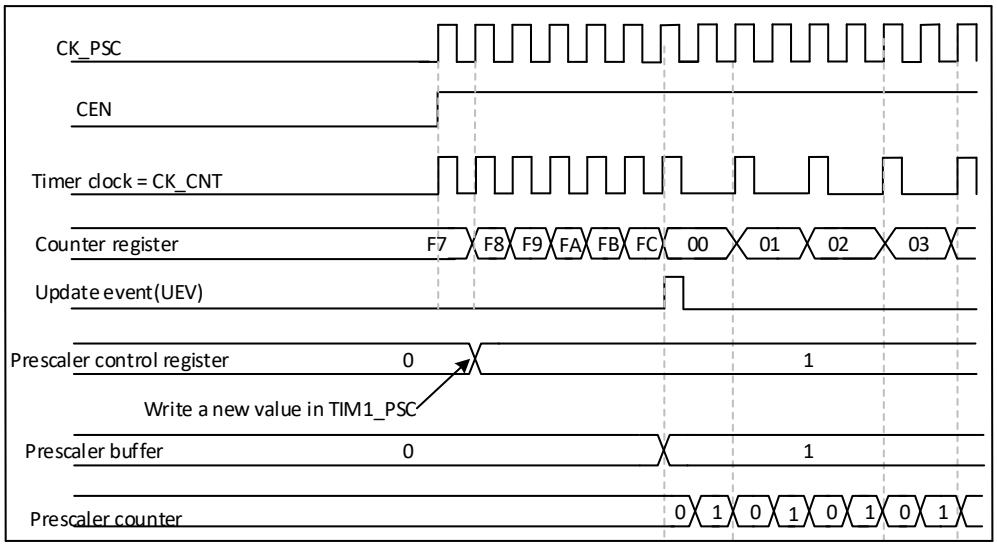


Figure 22-1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

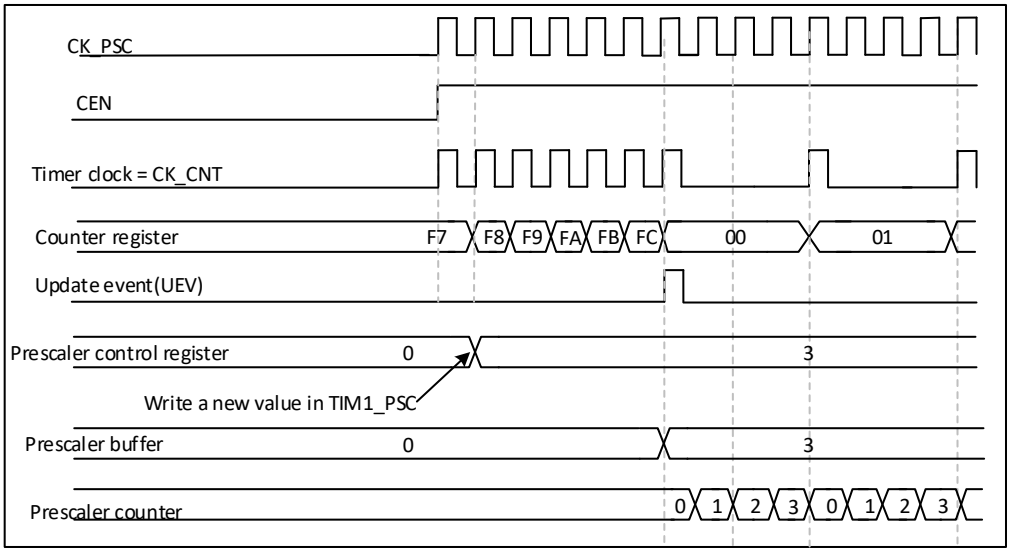


Figure 22-2 Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

22.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur,

the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

In addition, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

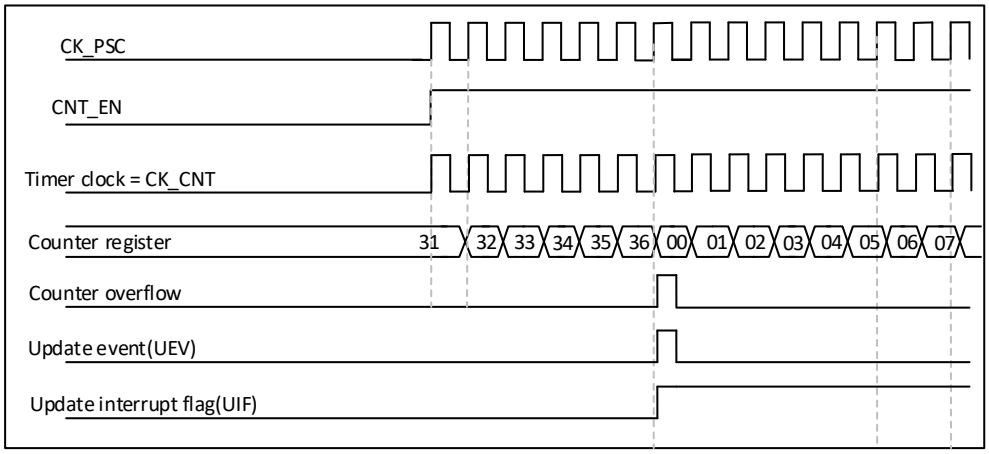


Figure 22-3 counter timing diagram with internal clock division factor of 1

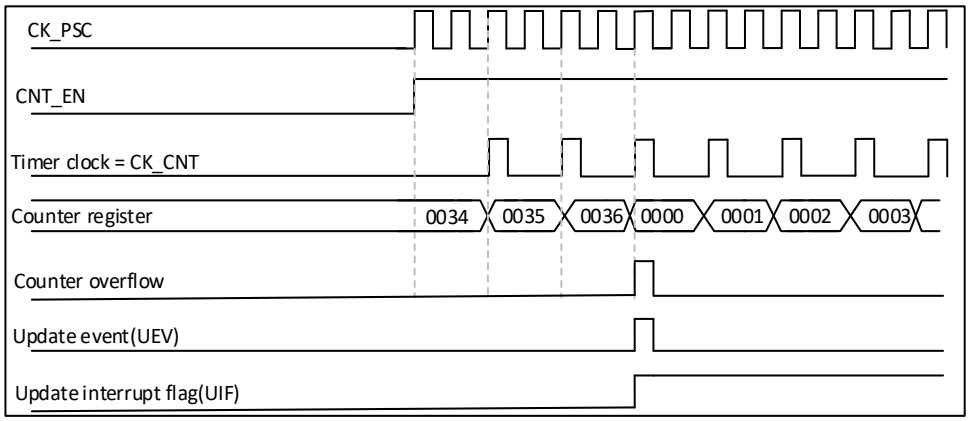


Figure 22-4 counter timing diagram with internal clock division factor of 2

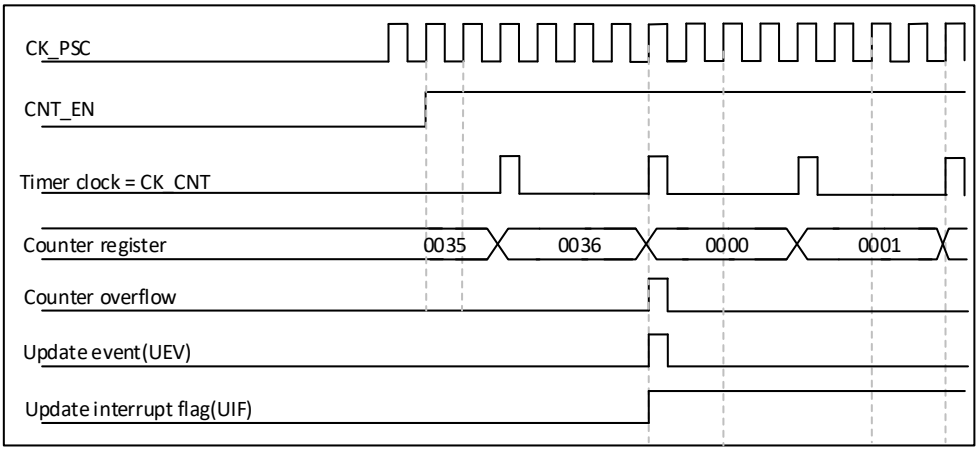


Figure 22-5 Counter timing diagram, internal clock divided by 4

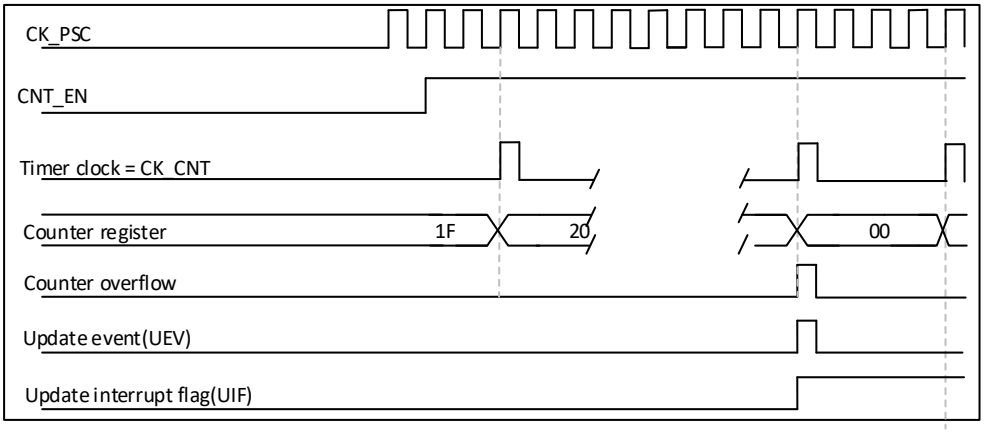


Figure 22-6 Counter timing diagram, internal clock divided by N

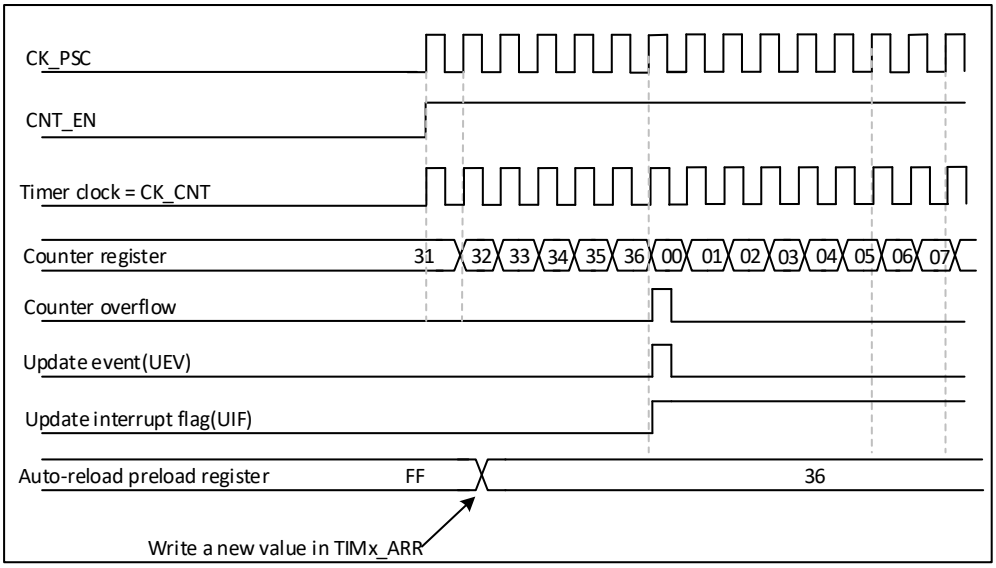


Figure 22-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

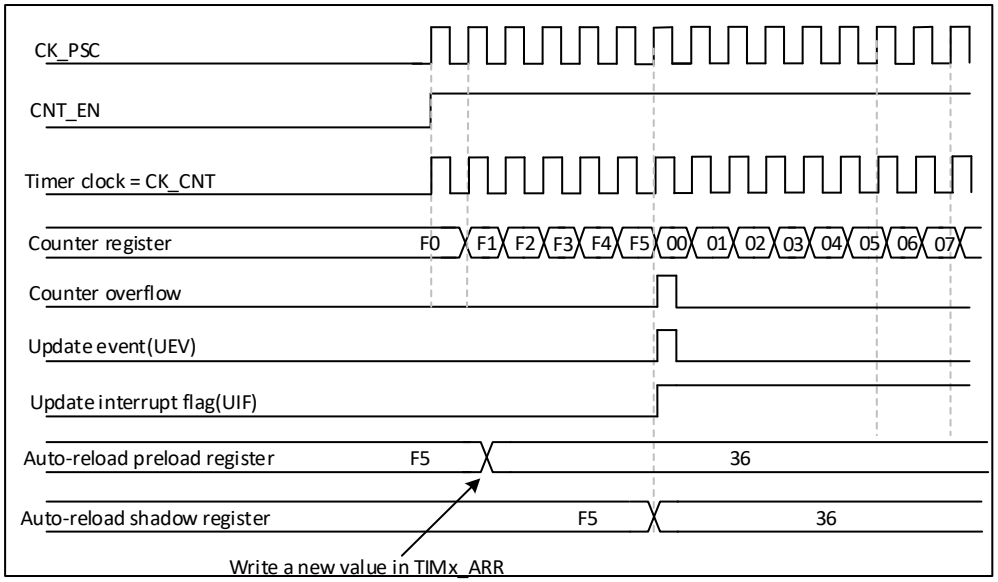


Figure 22-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

22.2.3 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)

Internal clock source (CK_INT)

For TIM10/TIM11/TIM13/TIM14/TIM19, the internal clock source is the default clock, then the CEN and UG bits (TIMx_EGR registers) are de facto control bits and can only be modified by software (the UG bits are still automatically cleared). As long as the CEN bit is written as' 1 ', the clock of the pre-scaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

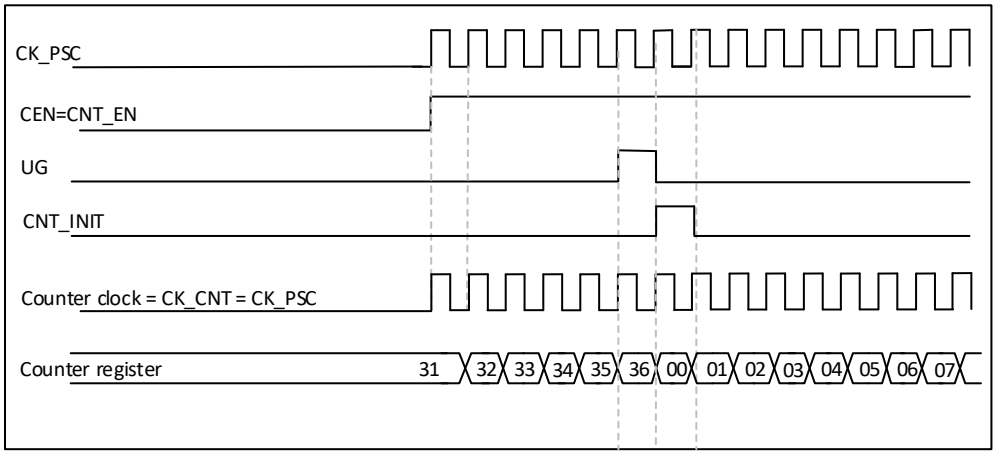


Figure 22-9 Control circuit in normal mode, internal clock divided by 1

22.2.4 Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The following figures are capture/compare channel overviews.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge monitor with polarity selection then produces a signal (TIxFPx), which can be used as a capture control. It is prescaled before the capture register (ICxPS).

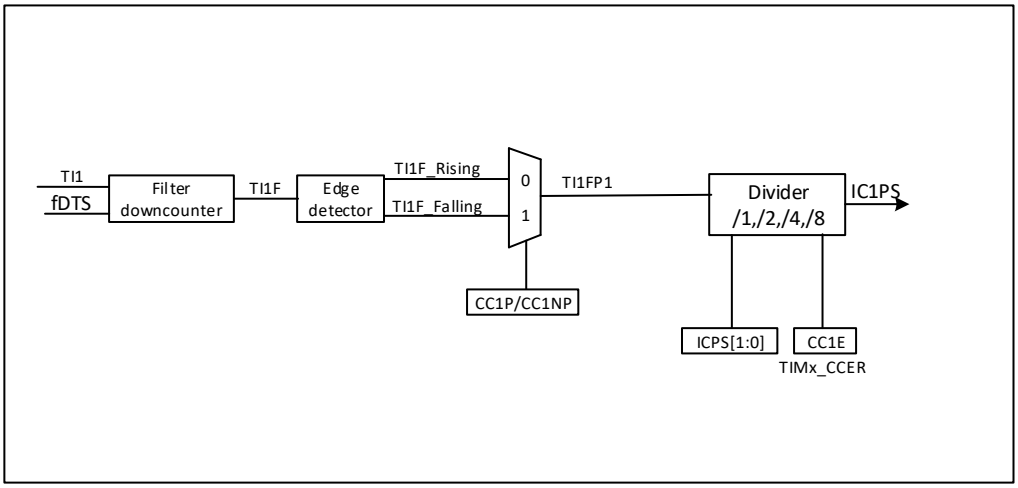


Figure 22-10 capture channel

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

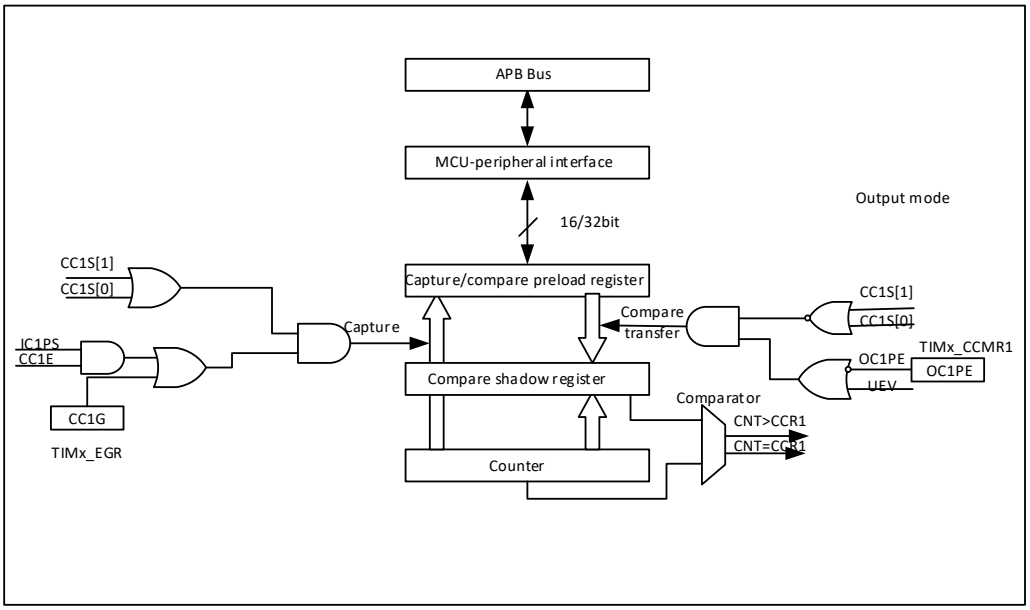


Figure 22-11 Capture/Compare the main circuit of channel 1

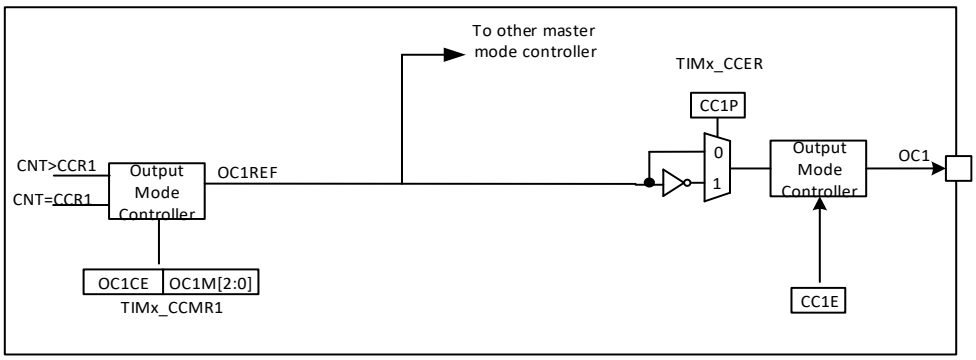


Figure 22-12 Output stage of capture/compare channel

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

22.2.5 Input capture mode:

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched into the capture/compare register (TIMx_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx_SR register) is set to 1. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (TIMx_SR register) is set to 1. The CCxIF can be cleared by writing CCxIF = 0, or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select valid input: TIMx_CCMR1 must be connected to the TI1 input, so CC1S = 01 written in the TIMx_CCMR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (set to rising edge) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If desired, the associated interrupt request may be allowed by setting the CC1IE bit in the

TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- The CC1IF flag bit is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

22.2.6 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, you just need to write101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

22.2.7 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has expired.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

You can select whether the TIMx_CCRx register needs to use a preload register by configuring the OCxPE bit in TIMx_CCMRx.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

— Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx

— Write OCxPE = 0 to disable preload register

— Write CCxP = 0 to select active high polarity

— Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated by software at any time to control the output waveform, provided the preload register is not used (OCxPE = '0', otherwise the shadow register of TIMx_CCRx can only be updated when the next update event occurs). An example is given in the figure below.

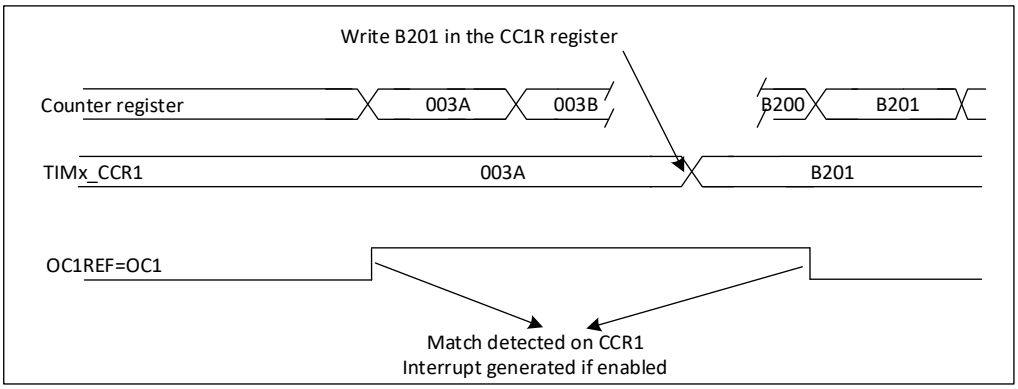


Figure 22-13 Output comparison mode, flip OC1

22.2.8 PWM mode

The pulse width modulation mode may produce a signal whose frequency is determined by the TIMx_ARR register and whose duty cycle is determined by the TIMx_CCRx register.

The OCxM bit in the TIMx_CCMRx register is written to '110' (PWM mode 1) or '111' (PWM mode 2), and each OCx output channel can be independently set to generate a PWM. The corresponding preload register must be enabled by setting the OCxPE bit of the TIMx_CCMRx register, and finally setting the ARPE bit of the TIMx_CR1 register to enable the preload register that can be automatically reloaded.

The preload register can only be transferred to the shadow register when an update event occurs, so the user must initialize all registers by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. The output of OCx enables control by the CCxE bit (in TIMx_CCER). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

PWM edge-aligned mode

- Upcounting configuration

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the comparison value in TIMx_CCRx is greater than the auto-reload value (TIMx_ARR), OCxREF remains' 1 '. If the comparison value is 0, then OCxREF remains' 0 '. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

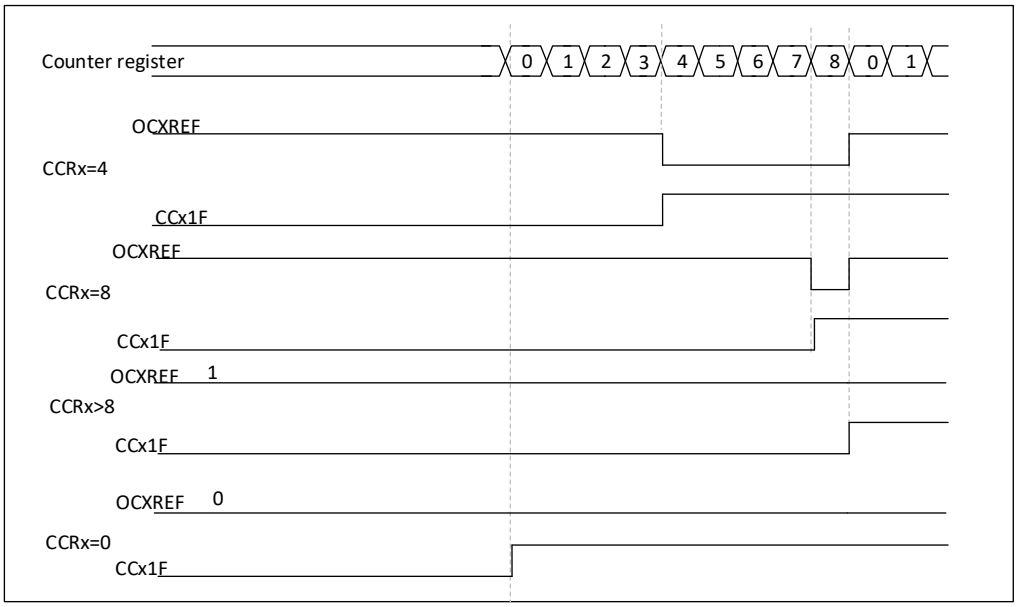


Figure 22-14 Edge-aligned PWM waveforms (ARR=8)

22.2.9 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

22.2.10 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. See Section 20.2.22: Timer Synchronization for details.

22.2.11 Debug mode

When the microcontroller enters debug mode (Cortex-M4F core stops), according to the setting of DBG_TIMx_STOP in the DBG module,

The TIMx counter may either continue normal operation or stop. See subsequent DBG sections for details.

22.3 TIMx registers

22.3.1 TIMX control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CKD [1: 0]		ARPE	Res			OPM	URS	UDIS	CEN
						RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the frequency division ratio between the timer clock (CK_INT) frequency and dead time and the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 4	Reserved	-	-	-
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

22.3.2 TIMX DMA/interrupt enable register (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														CC1IE	UIE
														RW	RW

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable) 0: Disable capture/compare 1 interrupt; 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

22.3.3 TIMX Status Register (TIMX_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CC1OF	Res							CC1IF	UIF
						RC_W0								RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	CC1OF	RC_W0	0	Capture/Compare 1 Repeat Capture Marker (Capture/Compare 1 overcapture flag) This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been generated. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8: 3	Reserved	-	-	-
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes high under the counter overflow condition in the up or up/down count mode, or under the counter underflow condition in the down count mode If channel TI1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by soft-ware. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0, when the repetition counter value overflows or underflows (an update event occurs when the repetition counter = 0).

				− If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

22.3.4 TIMX Event Generation Register (TIMX_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														CC1G	UG
														W	W

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	CC1G	W	0	Capture/Compare 1 generation This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action 1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged).

22.3.5 TIMX Capture/Compare Mode Control Register 1 (TIMX_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								OC1CE	OC1M [2: 0]			OC1PE	Res	CC1S [1: 0]	
								ICIF [3: 0]				ICIPSC [1: 0]			
								RW	RW	RW	RW	RW	RW	RW	RW

22.3.5.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	Output Comparison 1 Mode Output Compare 1 mode These bits define the action of outputting the reference signal OC1REF, and OC1REF determines the values of OC1, OC1N. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs.

				001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high. 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110: PWM mode 1-on up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 else inactive.
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives.
2	Reserved	-	-	-
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

22.3.5.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection

				These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

22.3.6 TIMX capture/compare enable register (TIMX_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low.
2	Reserved	-	-	-
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode.
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output. 1: ON-The OC1 signal is output to the corresponding output pin. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table 22-1 Control bits for complementary output channels OCx and OCxN with brake function

CCxE bit	OCx output status
0	Output disabled (disconnected from timer)OCx = 0, OCx_EN = 0
1	OCx = OCxREF + Polarity, OCx_EN = 1

22.3.7 TIMX Counter (TIMX_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

22.3.8 TIMX Prescaler (TIMX_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes the counter being cleared '0' by the UG bit of TIM_EGR or cleared '0' by the slave controller operating in reset mode.

22.3.9 TIMX Automatic Reload Register (TIMX_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ARR	RW	16'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null.

22.3.10 TIMX Capture/Compare Register 1 (TIMX_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15: 0]															
RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO	RW/ RO

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CCR1	RW/ RO	16'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only

23. General Purpose Timer (TIM15/TIM16/TIM17)

23.1 Introduction

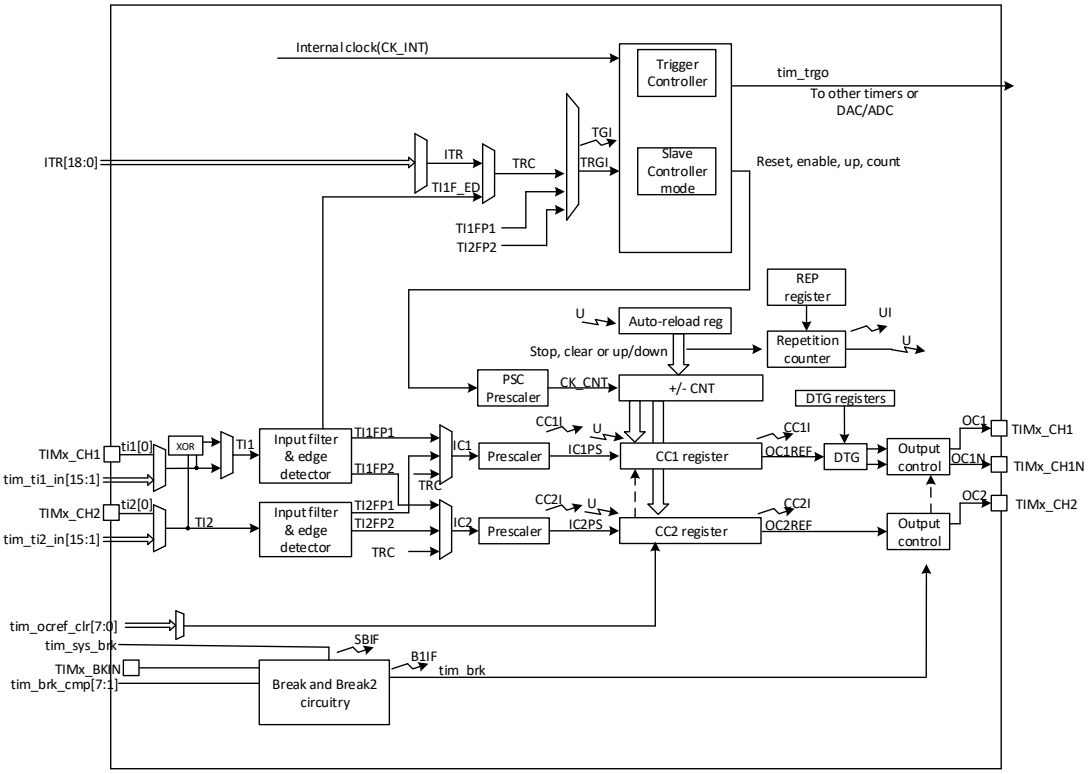
The universal control timer (TIM15_16_17) consists of a 16-bit auto-load counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion). Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The universal control timer (TIM15_16_17) and the universal timer (TIMx) are completely independent and do not share any resources. They can be synchronized together.

23.1.1 TIM15 features

The functions of the TIM15 timer include:

- 16-bit up auto-load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- Up to 2 independent channels:
  - Input capture
  - Output compare
  - PWM generation (edge alignment mode)
  - One-pulse mode output
- Complementary output with programmable dead time (channel 1 only)
- Synchronization circuit for controlling timer and interconnection between timer by external signal
- Allows the repetition counter of the timer register to be updated after a specified number of counter cycles
- The brake input signal may place the timer output signal into a reset state or a known state
- An interrupt/DMA occurs when the following events occur:
  - Update: Counter overflow, counter initialization (via software or internal/external trigger)
  - Trigger event (counter starts, stops, initializes or counts by internal/external triggers)
  - Input capture
  - Output compare
  - Brake signal input

23.1.2 CAN block diagram

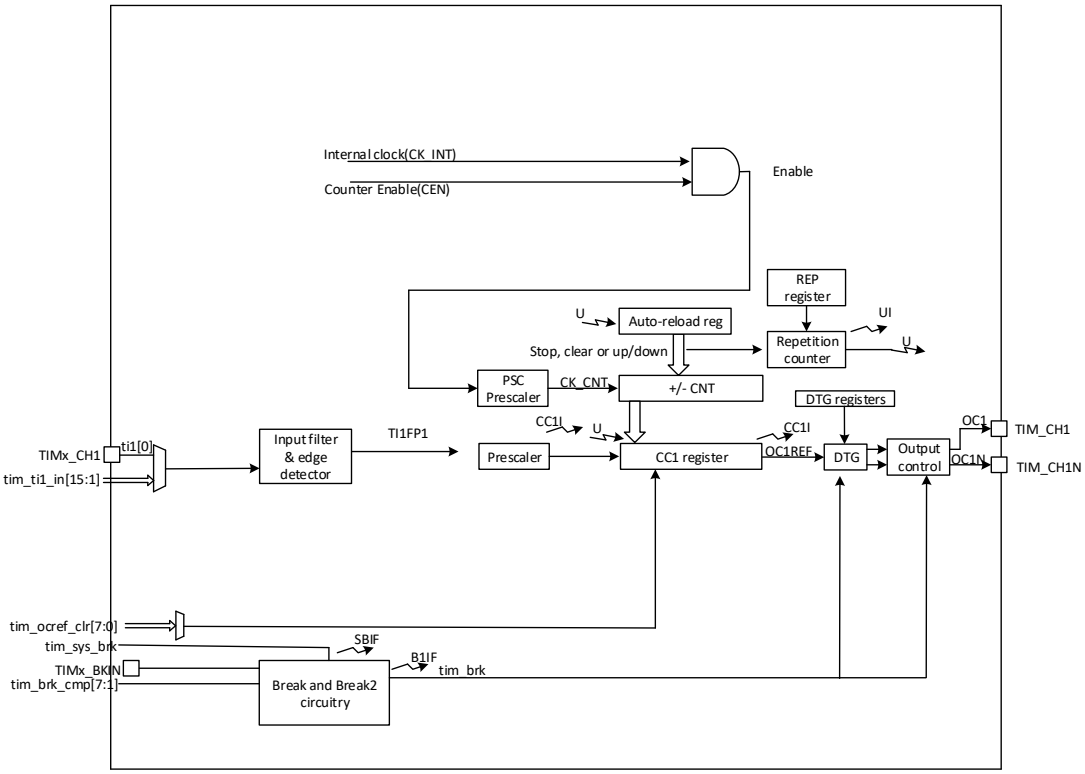


23.1.3 TIM16_17 features

The functions of the TIM16_17 timer include:

- 16-bit up, down, up/down auto load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65536
- 1 independent channel:
  - Input capture
  - Output compare
  - PWM generation (edge alignment mode)
  - One-pulse mode output
- Complementary output with programmable dead time
- Allows the repetition counter of the timer register to be updated after a specified number of counter cycles
- The brake input signal may place the timer output signal into a reset state or a known state
- An interrupt/DMA occurs when the following events occur:
  - Counter overflows up
  - Input capture
  - Output compare
  - Brake signal input

23.1.4 CAN block diagram



23.2 TIM15_16_17 functional description

23.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)
- Repetition Register (TIMx_RCR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition (overflow or underflow) and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set. (See the description of the slave mode controller for more details on enabling counters).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

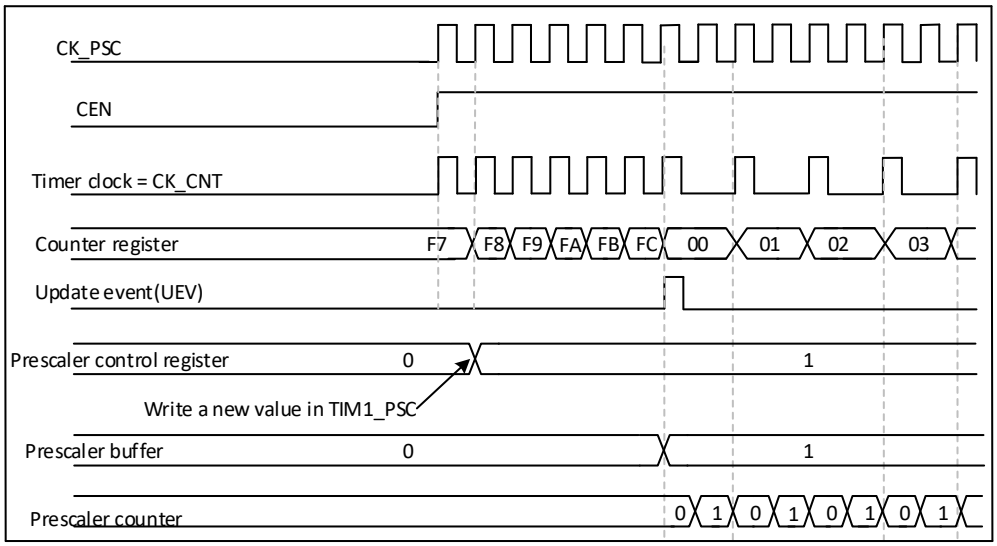


Figure 23-.1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

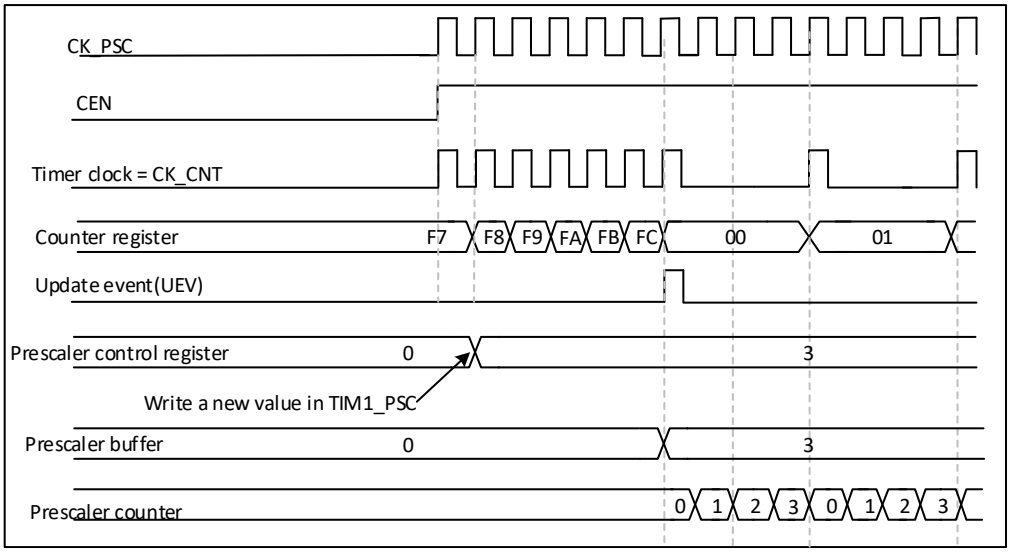


Figure 23-2. Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

23.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event.

If a repetition counter is used, the update event (UEV) will not be generated until the number of overflows reaches the value of the configured repetition count register plus one (i.e. TIMx_RCR+1); If a repetition counter is not used (i.e., TIMx_RCR = 0), an update event will be generated every time the count overflows.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

Furthermore, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt or DMA request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

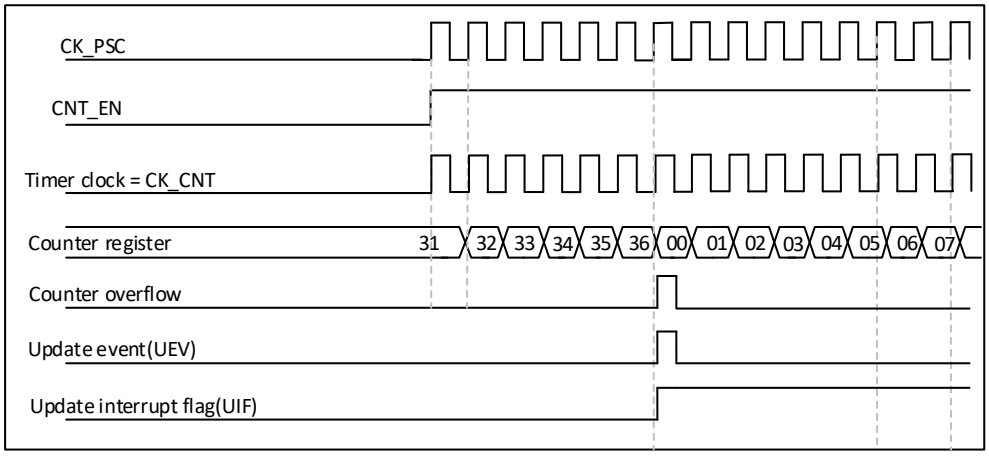


Figure 23-3 counter timing diagram with internal clock division factor of 1

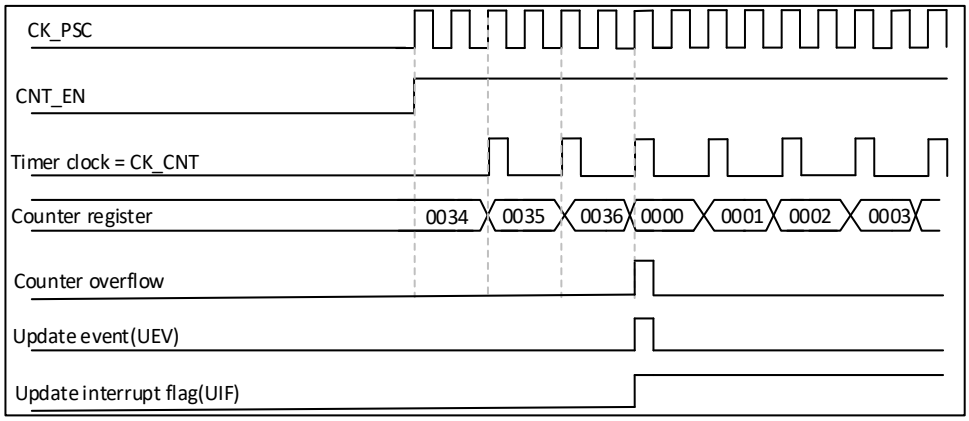


Figure 23-4 counter timing diagram with internal clock division factor of 2

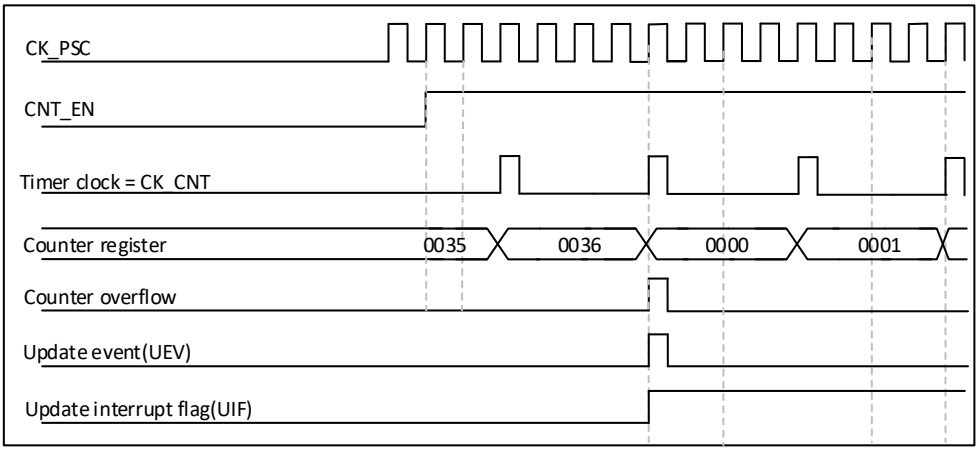


Figure 23-5 Counter timing diagram, internal clock divided by 4

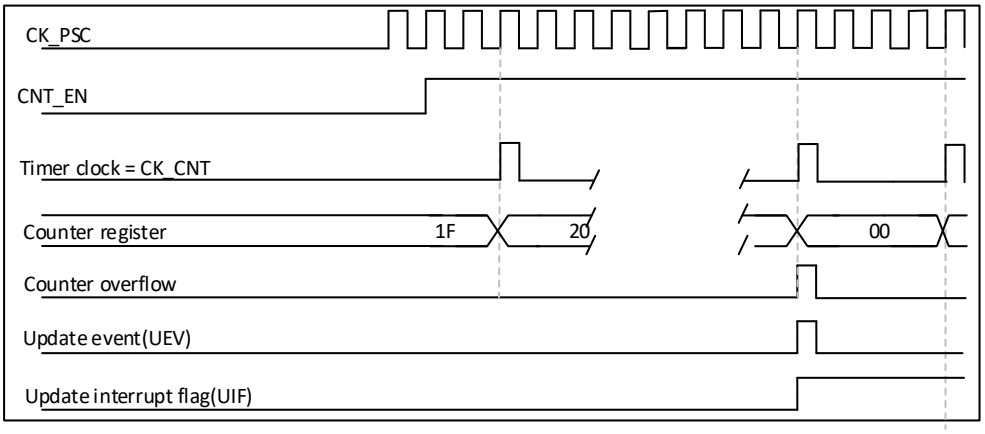


Figure 23-6 Counter timing diagram, internal clock divided by N

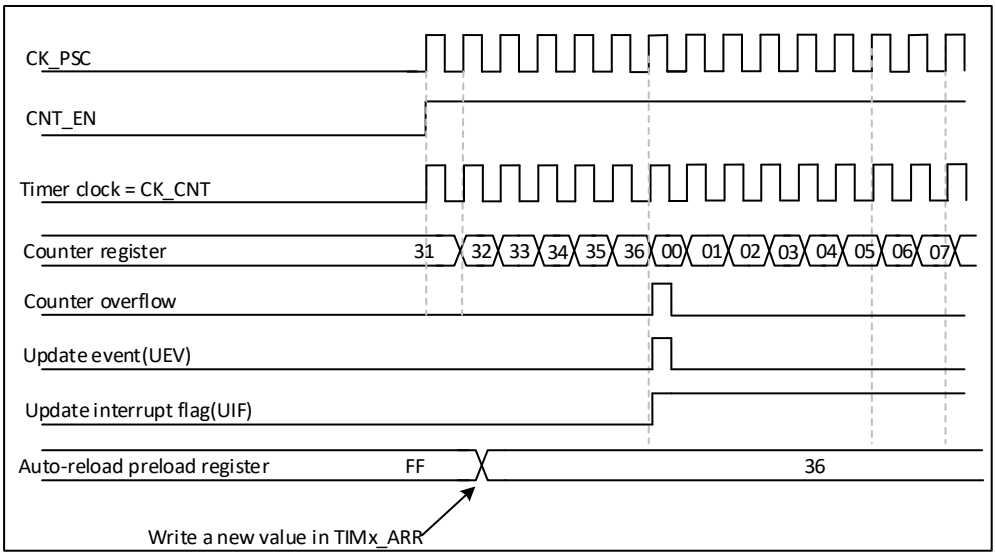


Figure 23-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

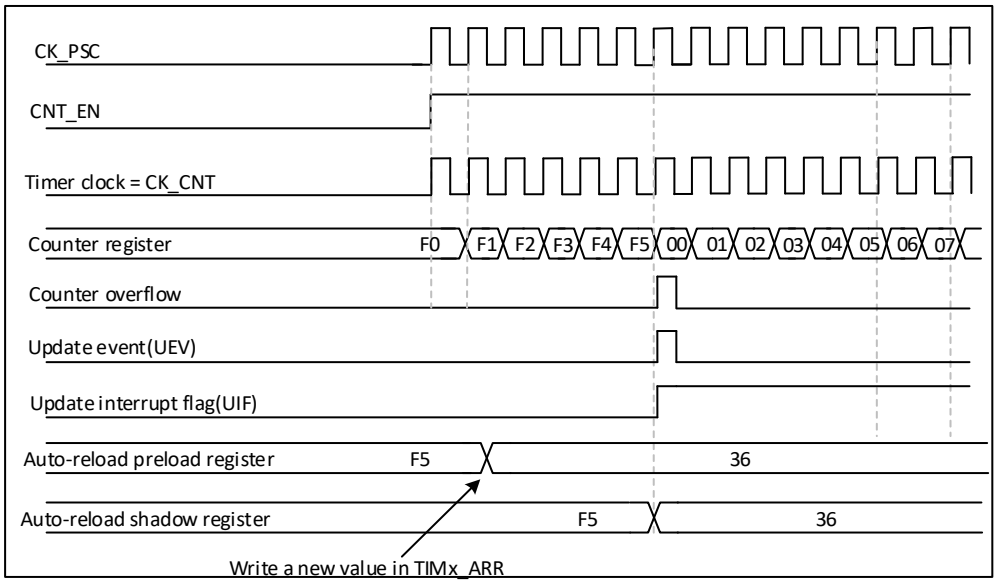


Figure 23-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

23.2.3 Repetition counter

The Time Base Unit explains how an update event (UEV) is generated when the counter overflows, in fact it can only be generated when the repetition counter count reaches 0. This feature is very useful for generating PWM signals.

This means thatdata is transferred from the preload register to the shadow register (TIMx_ARR auto-reload register, TIMx_PSC preload register, and capture/compare register TIMx_CCRx in comparison mode) onlyevery N+1 count overflow, with N being the value in the TIMx_RCR repeat count register. The repeat counter is decremented when the counter overflows.

The repetition counter is automatically reloaded and the repetition rate is defined by the value of the TIMx_RCR register. When the update event is generated by software (by setting the UG bit in

TIMx_EGR) or by the slave mode controller of hardware, the update event occurs immediately regardless of the value of the repetition counter, and the contents of the TIMx_RCR register are reloaded to the repetition counter.

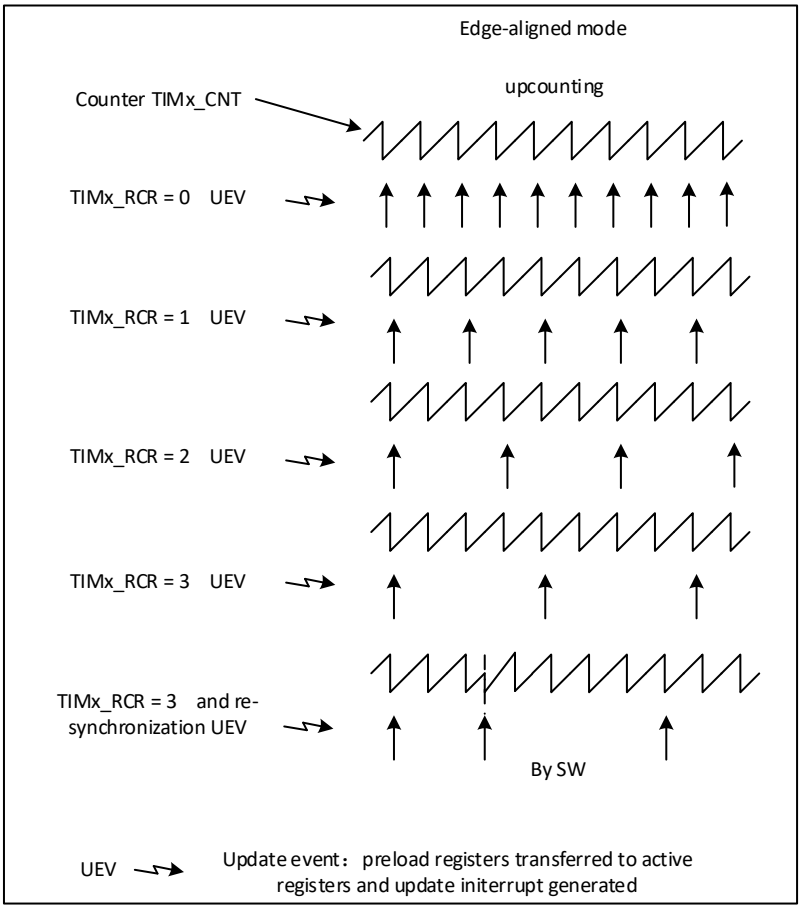


Figure 23-9. Examples of update rates in different modes, and register settings for TIMx_RCR

23.2.4 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pins (TI1 and TI2)
- Internal Trigger Input (ITRx) (tim15 only): Uses one timer as a prescaler for the other.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 0000), the CEN and UG bits (TIMx_EGR register) are de facto control bits and can only be modified by software (the UG bits are still automatically cleared). As long as the CEN bit is written as' 1 ', the clock of the prescaler is provided by the internal clock CK_INT.

The diagram below shows the operation of the control circuit and up counter in general mode, without the prescaler.

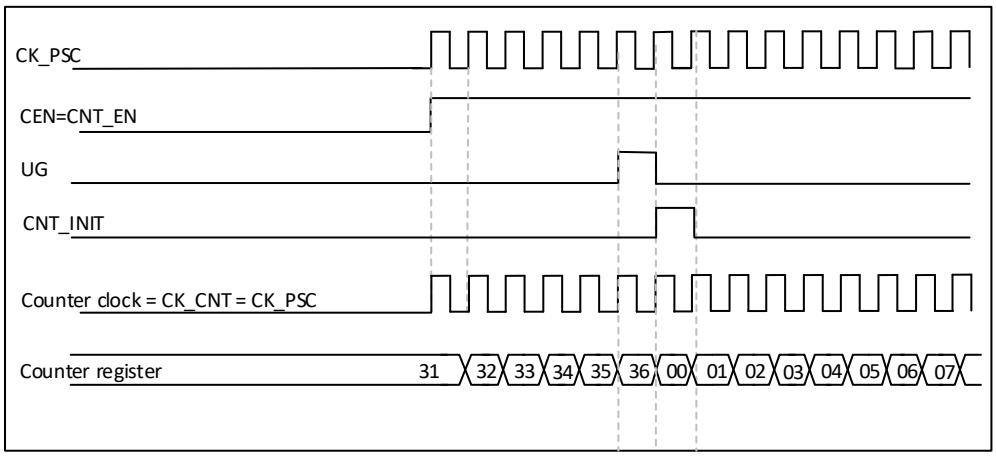


Figure 23-10 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

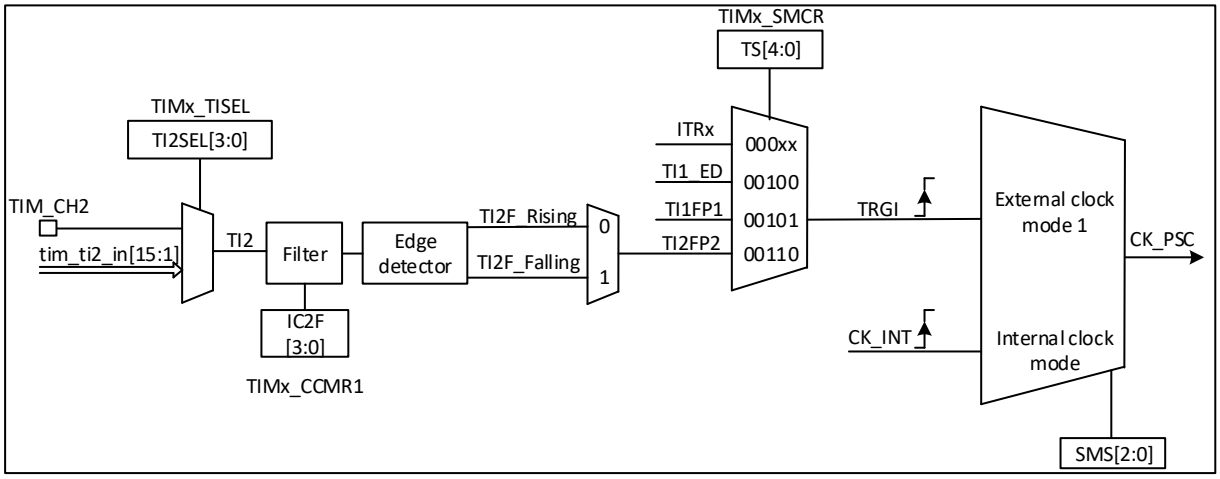


Figure 23-11 TI2 Example of external clock connection

For example, to configure the counter to count up on the rising edge of the TI2 input, use the following steps:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = '01' in the TIMx_CCMR1 register.

2. Configure IC2F [3: 0] of the TIMx_CCMR1 register and select the input filter bandwidth (if no filter is needed, keep IC2F = 0000);

3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.

4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=00110 in the TIMx_SMCR register;

6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the synchronization circuit at the input of TI2.

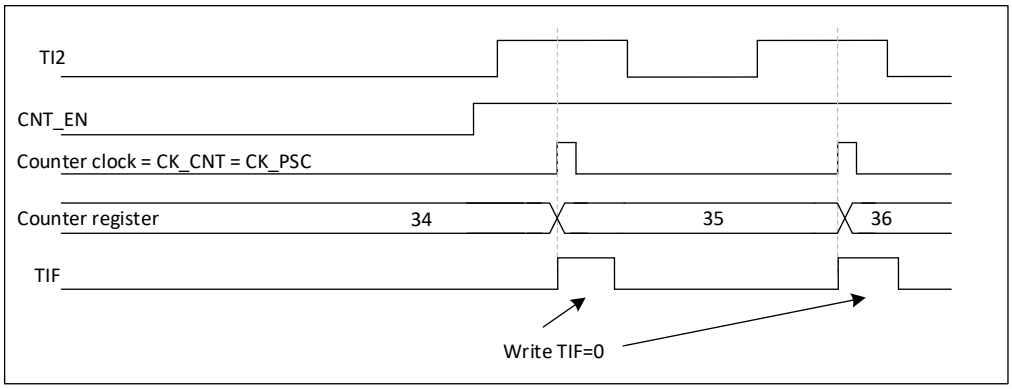


Figure 23-12 Control circuit in external clock mode 1

23.2.5 Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The next few pictures are an overview of the capture/compare channel.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge monitor with polarity selection then generates a signal (TIxFPx) that can be triggered as an input from the slave mode controller or as a capture control. It is prescaled before the capture register (ICxPS).

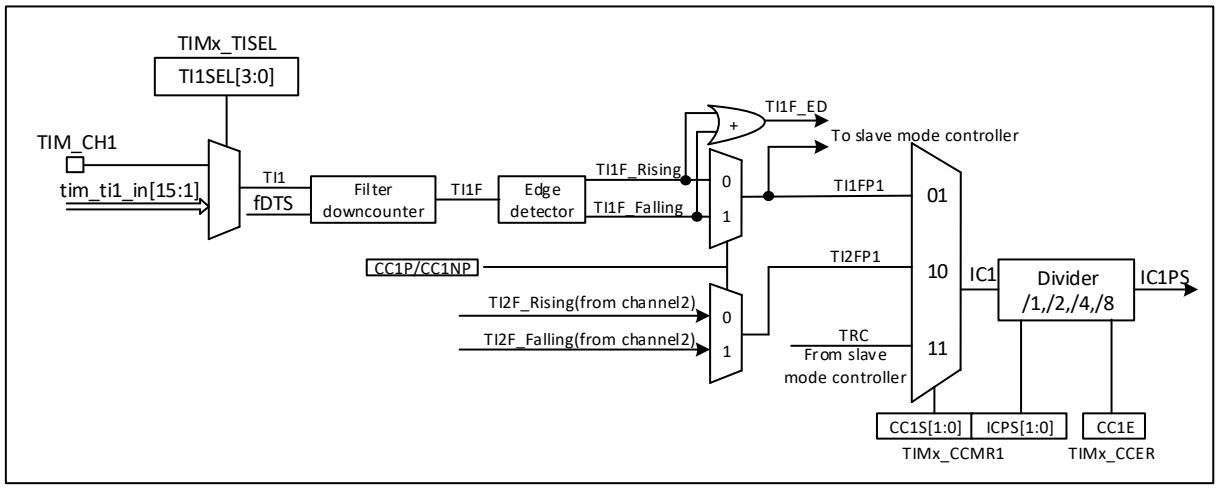


Figure 23-13 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

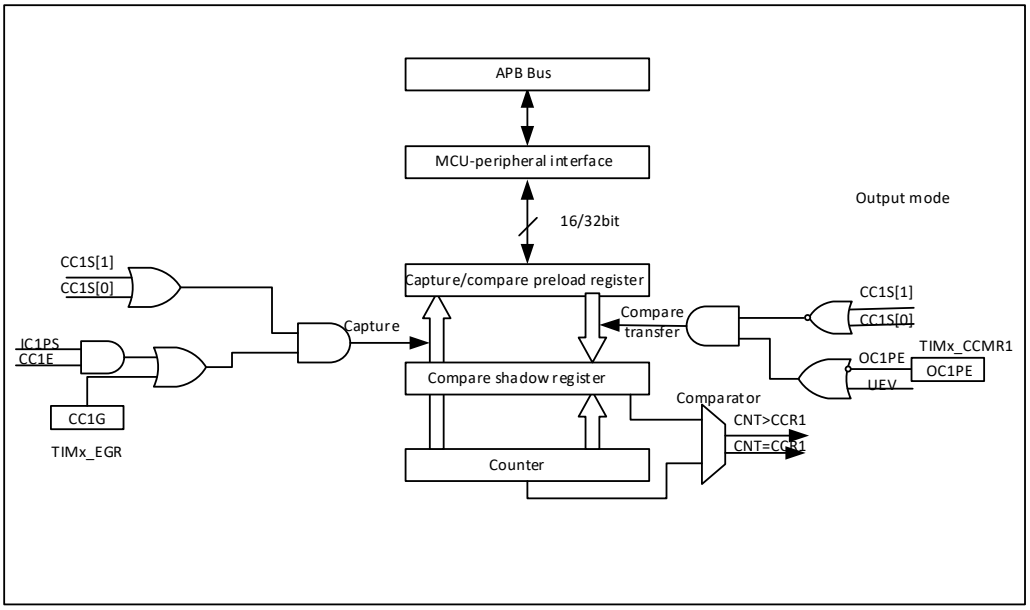


Figure 23-14 Capture/Compare the main circuit of channel 1

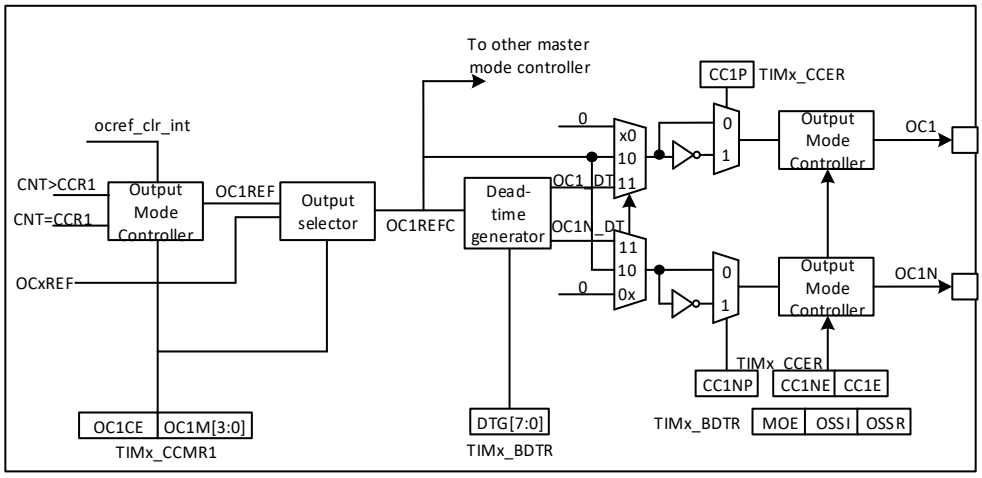


Figure 23-15 Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

23.2.6 Input capture mode:

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched into the capture/compare register (TIMx_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx_SR register) is set to 1, and if an interrupt or DMA operation is enabled, an interrupt or DMA request will be generated. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (TIMx_SR register) is set to 1. The CCxIF can be cleared by writing CCxIF = 0, or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select valid input: TIMx_CCMR1 must be connected to the TI1 input, so CC1S = 01 written in the TIMx_CCMR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (set to rising edge) in the TIMx_CCER register.
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If desired, the associated interrupt request may be allowed by setting the CC1IE bit in the TIMx_DIER register, or the DMA request may be allowed by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- The CC1IF flag bit is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

23.2.7 PWM input mode (only for tim15)

This mode is a special case of the input capture mode, and the rest of the operation is the same as the input capture mode except for the following differences:

- Both ICx signals are mapped to the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, the user can measure the cycle (TIMx_CCR1 register) and the duty cycle (TIMx_CCR2 register) of the PWM signal input to TI1 as follows

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

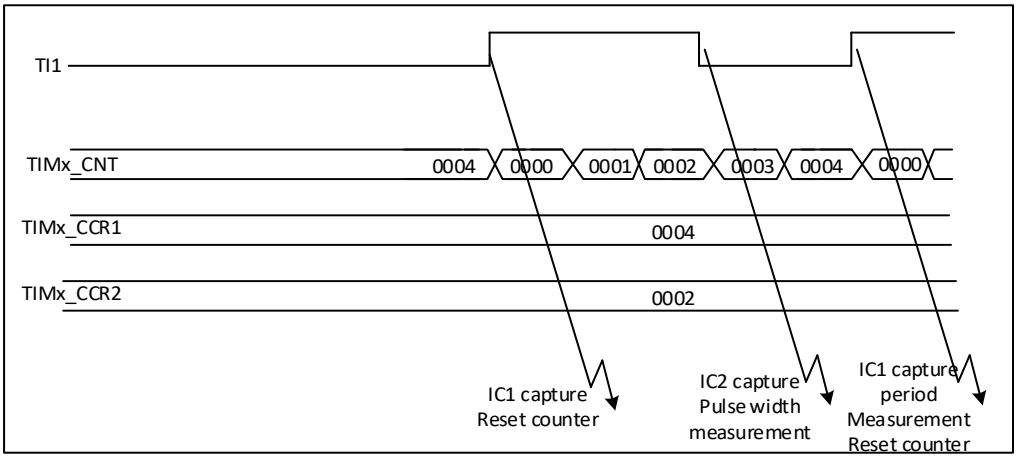


Figure 23-16 PWM input mode timing

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use the TIMx_CH1/TIMx_CH2 signal.

23.2.8 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

Set the corresponding OCxM = 0101 in the TIMx_CCMRx register to force the output comparison signal (OCxREF/OCx) to an active state. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 0100 in the TIMx_CCMRx register. Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

23.2.9 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has expired.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=0000), be set active (OCxM=0001), be set inactive (OCxM=0010) or can toggle (OCxM=0011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- If the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register selects the DMA request function), a DMA request is generated.

You can select whether the TIMx_CCRx register needs to use a preload register by configuring the OCxPE bit in TIMx_CCMRx.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

— Write OCxM = 0011 to toggle OCx output pin when CNT matches CCRx

— Write OCxPE = 0 to disable preload register

— Write CCxP = 0 to select active high polarity

— Write CCxE = 1 to enable the output

5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated by software at any time to control the output waveform, provided the preload register is not used (OCxPE = '0', otherwise the shadow register of TIMx_CCRx can only be updated when the next update event occurs). An example is given in the figure below.

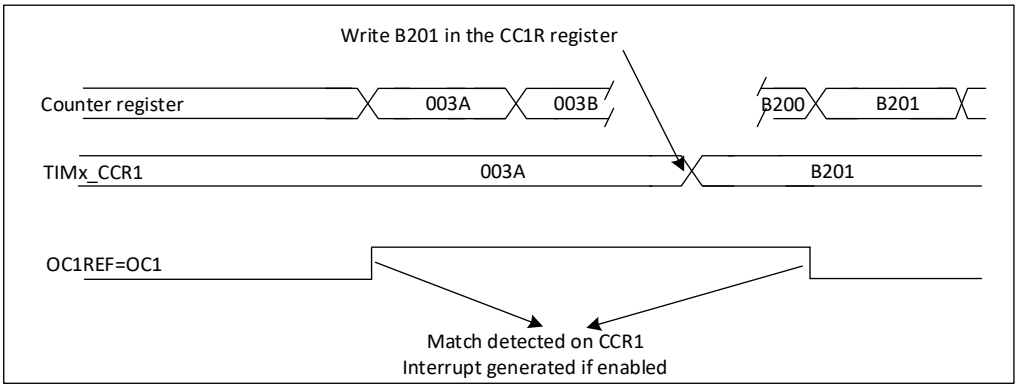


Figure 23-17 Output comparison mode, flip OC1

23.2.10 PWM mode

The pulse width modulation mode may produce a signal whose frequency is determined by the TIMx_ARR register and whose duty cycle is determined by the TIMx_CCRx register.

The OCxM bit in the TIMx_CCMRx register is written to '0110' (PWM mode 1) or '0111' (PWM mode 2), and each OCx output channel can be independently set to generate a PWM. The corresponding preload register must be enabled by setting the OCxPE bit of the TIMx_CCMRx register, and finally setting the ARPE bit of the TIMx_CR1 register to enable the preload register that can be automatically reloaded.

The preload register can only be transferred to the shadow register when an update event occurs, so the user must initialize all registers by setting the UG bit in the TIMx_EGR register before the counter starts counting.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

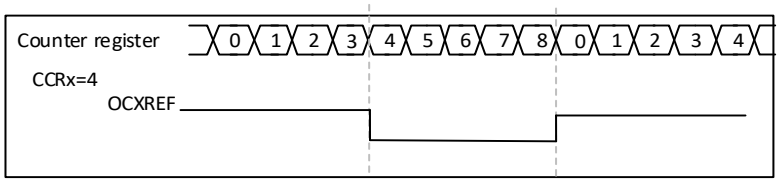


Figure 23-18 Edge-aligned PWM waveforms (ARR=8)

Jitter mode

The DITHEN bit of the TIMx_CR1 register may be enabled to turn on the jitter mode to improve the effective resolution of the PWM mode. Can be applied to CCR (improved duty cycle resolution) and ARR (improved resolution of PWM frequency)

The principle of operation is to make the actual CCR (or ARR) value change slightly (with or without increasing a timer cycle) over 16 consecutive PWM cycles, and how the change can be set by predefined settings. When calculating the average duty cycle, you can achieve a 16-fold increase in resolution. The figure below shows the jitter principle of four consecutive PWM periods

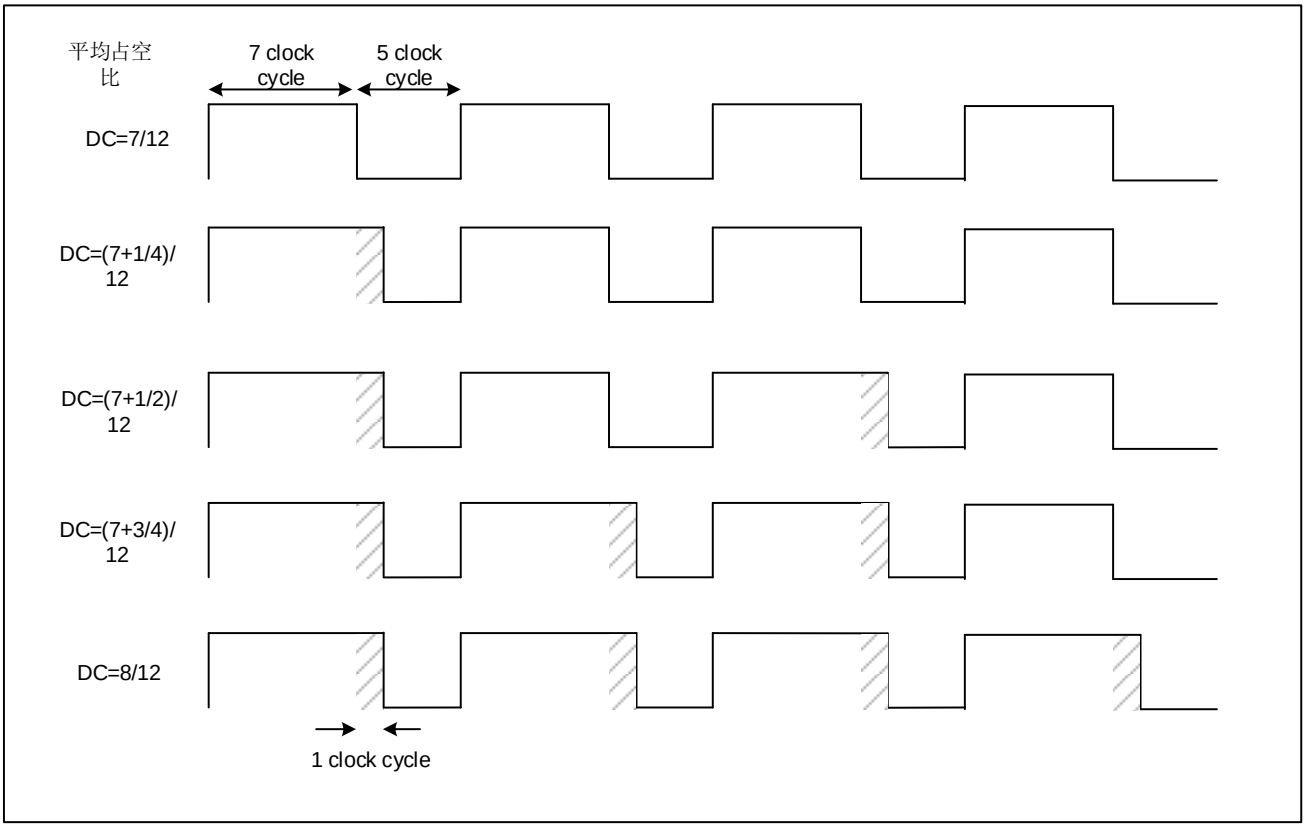


Figure 23-19 jitter principle

When jitter mode is enabled, the value of the register will automatically change as follows:

- The lowest 4 bits is the encoding for improving resolution (fractional part)
- The high bit is shifted left to 19: 4 for encoding as the base value (integer part)

Note: When dither mode is on or off, the values of ARR and CCR are automatically updated (for example, ARR = 0x05 when DITHEN = 0, then ARR = 0x50 when DITHEN = 1), the following steps must be followed when resetting the DITHEN bit

1. CEN and ARPE bits must be reset
2. The ARR [3: 0] bit must be reset
3. The DITHEN bit must be reset
4. The CCIF flag must be cleared
5. CEN bit can be set (ARPE = 1 can be set)

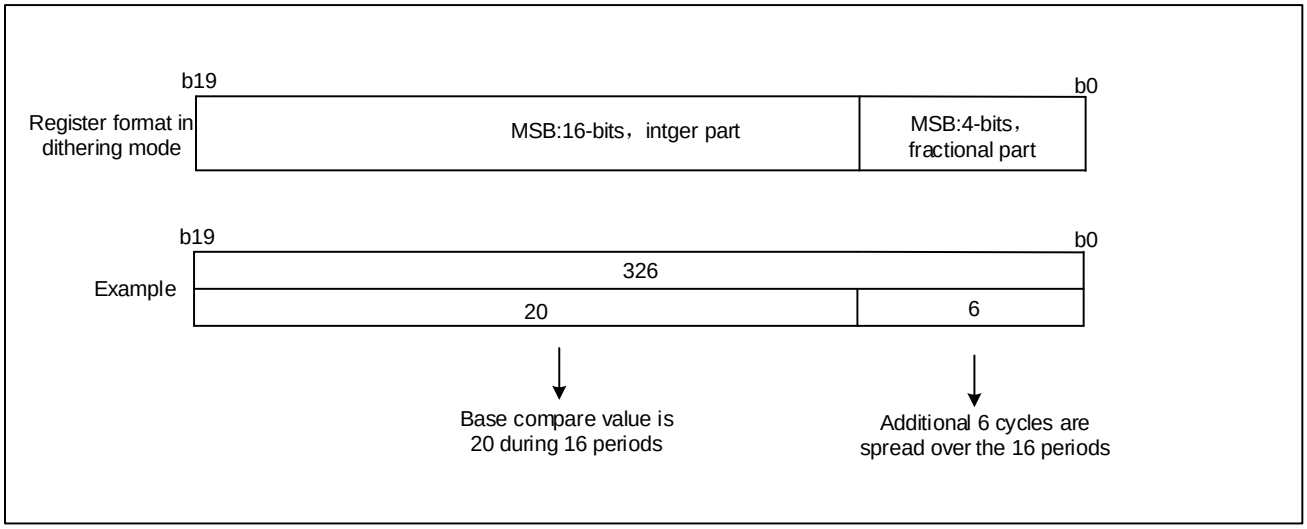


Figure 23-20 Data mapping and register coding in dither mode

The minimum frequency is calculated as follows:

According to resolution = $\frac{\text{定时器频率}}{\text{PWM 频率}}$ push out: minimum PWM frequency = $\frac{\text{定时器频率}}{\text{最大分辨率}}$

In jitter-free mode: Minimum PWM frequency = $\frac{\text{定时器频率}}{65536}$

When there is jitter mode: minimum PWM frequency = $\frac{\text{定时器频率}}{65535+\frac{15}{16}}$

Note: The maximum values of TIMx_arr and TIMxCCRy in jitter mode are limited to 0xFFFEF (corresponding to 65534 in the integer part and 15 in the jitter part). Exceeding 0xFFFEF, there will be an overflow situation, that is, arr = 0XFFFF+1 = 0.

The figure below shows that in jitter mode, the resolution of the PWM can be improved regardless of the PWM frequency

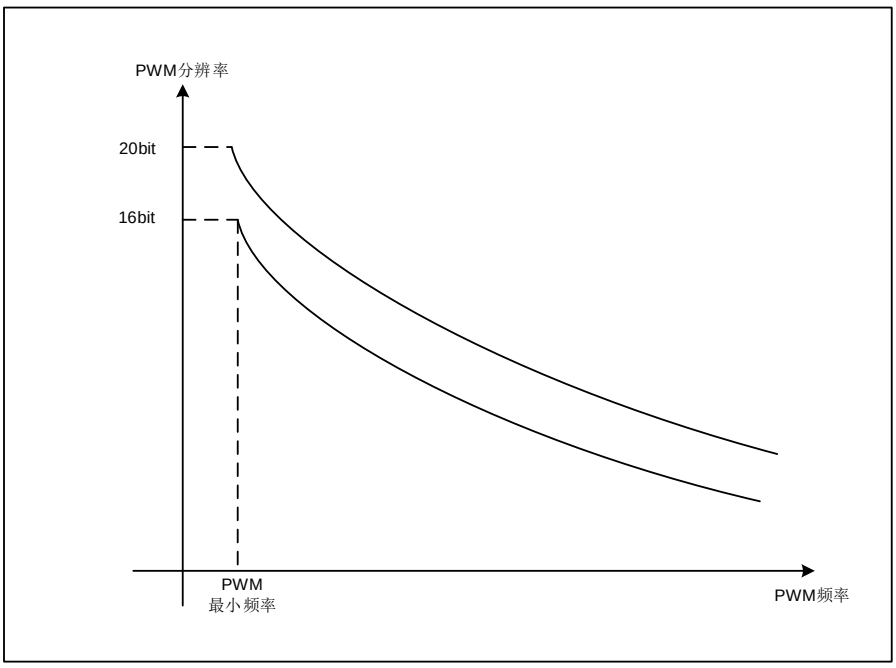


Figure 23-21 PWM Resolution VS Frequency

The duty cycle and cycle changes in 16 consecutive cycles are as follows:

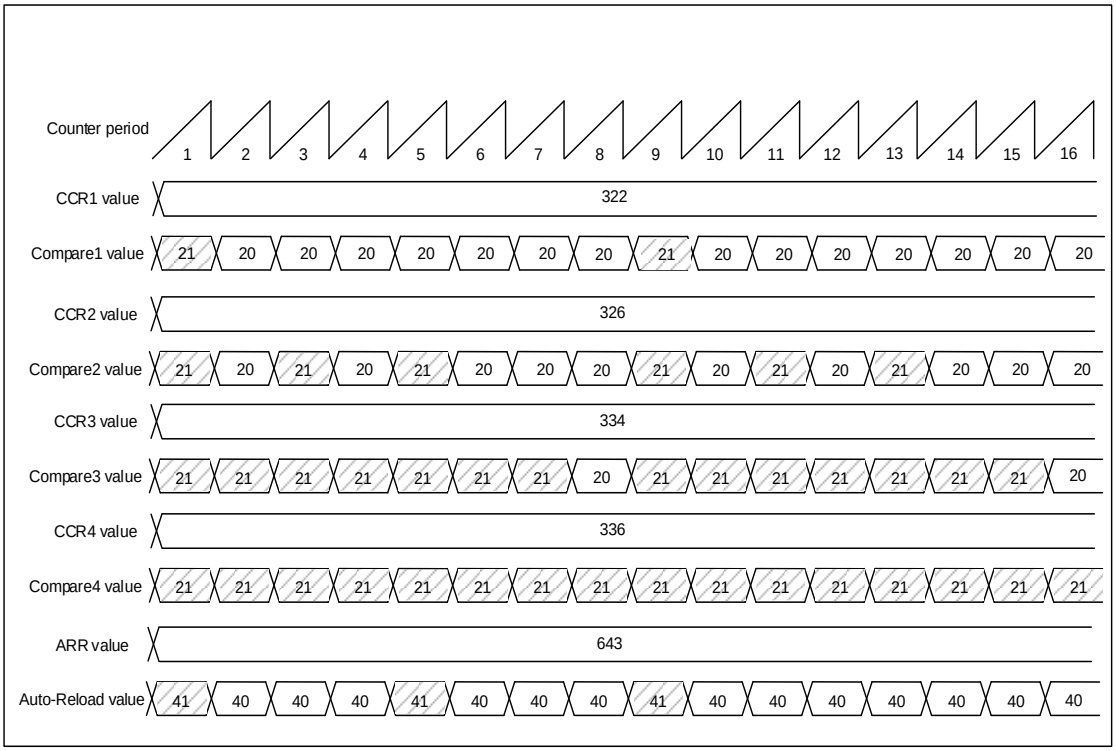


Figure 23-22 PWM Jitter Mode

The auto-reload and compare value increments are distributed in a specific pattern as described in the following table, with the incremental distribution of the jitter sequence as uniformly as possible, minimizing the overall ripple

Table 23-1 CCR and ARR Register Jitter Changes

LSB value	PWM Cycle															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000																
0001	+1															
0010	+1								+1							
0011	+1				+1				+1							
0100	+1				+1				+1				+1			
0101	+1		+1		+1				+1				+1			
0110	+1		+1		+1				+1		+1		+1			
0111	+1		+1		+1		+1		+1		+1		+1			
1000	+1		+1		+1		+1		+1		+1		+1		+1	
1001	+1	+1	+1		+1		+1		+1		+1		+1		+1	
1010	+1	+1	+1		+1		+1		+1	+1	+1		+1		+1	
1011	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1		+1	
1100	+1	+1	+1		+1	+1	+1		+1	+1	+1		+1	+1	+1	
1101	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1		+1	+1	+1	
1110	+1	+1	+1	+1	+1	+1	+1		+1	+1	+1	+1	+1	+1	+1	
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	

23.2.11 Combined PWM mode (tim15 only)

The combined PWM mode allows the PWM signal to be generated by programmable delay and phase shift between pulses. The frequency is determined by the ARR and the duty cycle and delay are determined by the two CCRs. The resulting signal ocxrefc is generated by AND/OR logic between two reference PWMs

- The OC1REFC(orOC2REFC) is controlled by TIMx_CCR1 and TIMx_CCR2

Asymmetric PWM modes can be independently selected on dual channels (each pair of CCR registers controls one OCx output) by writing to OCxM 1100 (combined PWM mode 1, logic OR output) and OCxM 1101 (combined PWM mode 2, logic AND output

When a given channel is used as a combined PWM channel, its complementary channels must be configured in opposite PWM modes (e.g., one for combined PWM mode 1 and one for combined PWM mode 2).

The following figure shows an example of a combined mode generation signal, including

- Channel 1 is combined PWM mode 2
- Channel 2 is PWM Mode 1

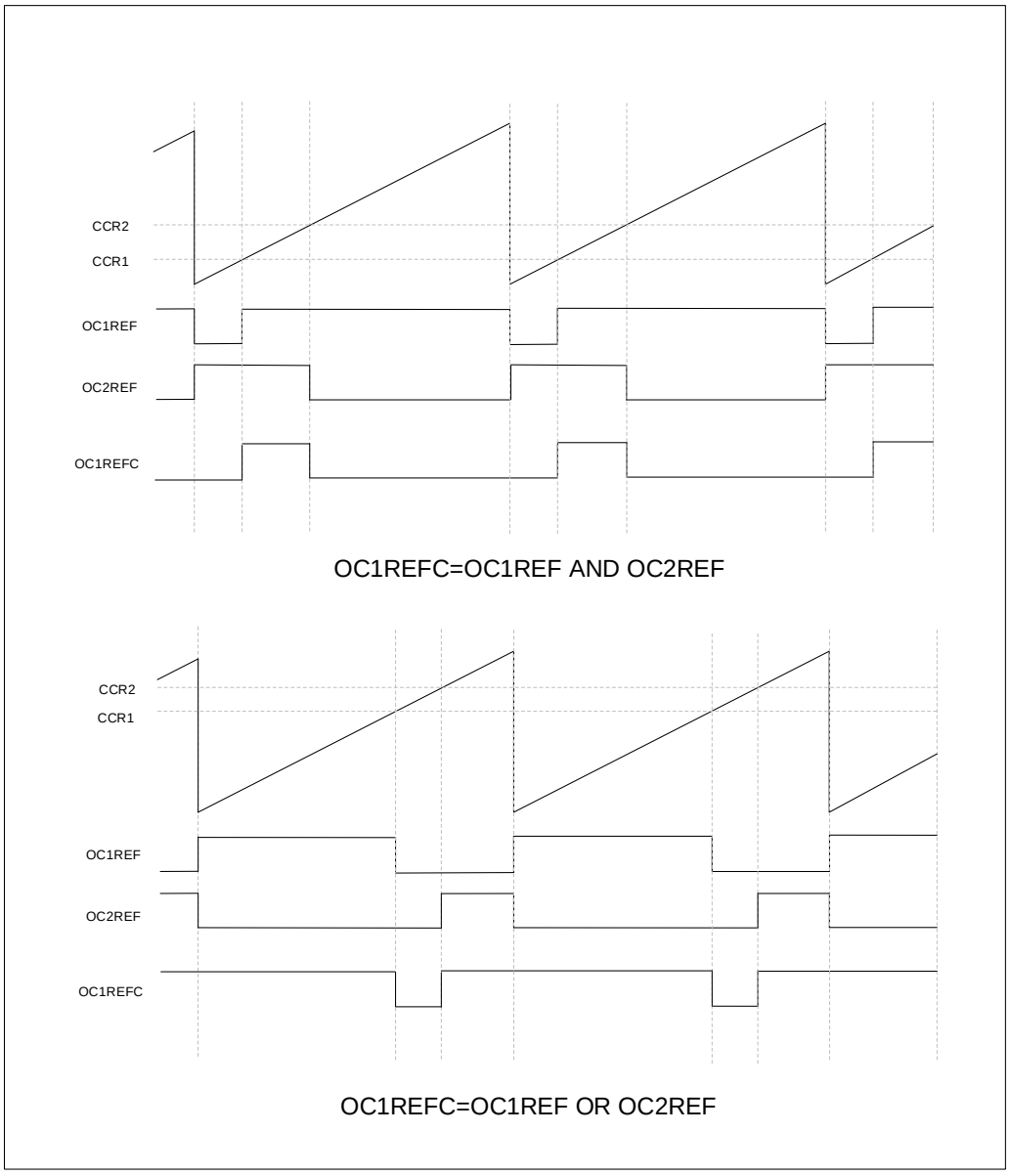


Figure 23-23 Combined PWM patterns for channel 1 and channel 2

23.2.12 Complementary outputs and dead-time insertion

The universal control timer (TIM15_16_17) can output two complementary signals and can manage the instantaneous turn-off and turn-on of the output.

This time is generally known as dead-time, and you have to adjust it depending on the devices you have connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

You can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: the CCxE and CCxNE bits of the TIMx_CCER register, the MOE, OISx, OISxN, OSSI and OSSR bits of the TIMx_BDTR and TIMx_CR2 registers, as detailed in the control bits of the complementary output channels OCx and OCxN with brake function. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. By configuring the DTG [7: 0] bit in the TIMx_BDTR register, you can control the dead time generator for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP=0, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

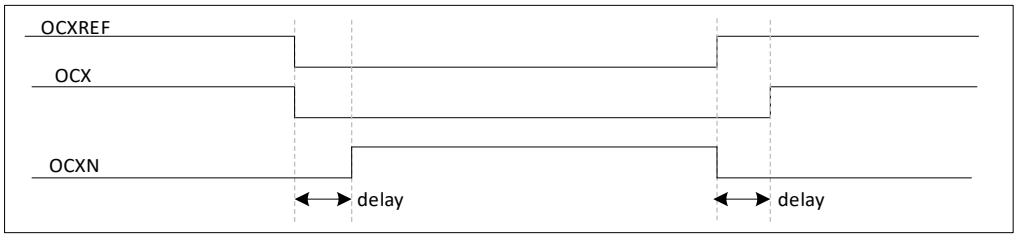


Figure 23-24 Complementary output with dead-time insertion

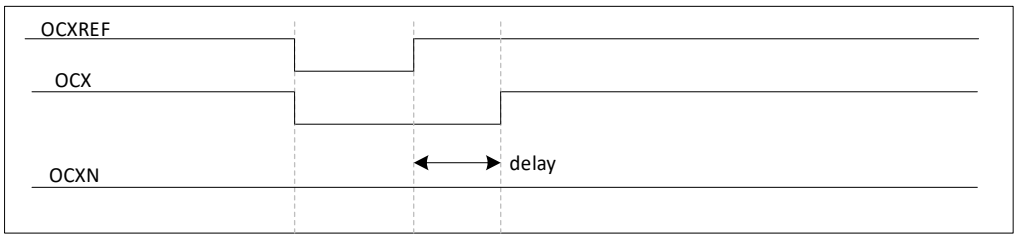


Figure 23-25 Dead-time waveforms with delay greater than the negative pulse

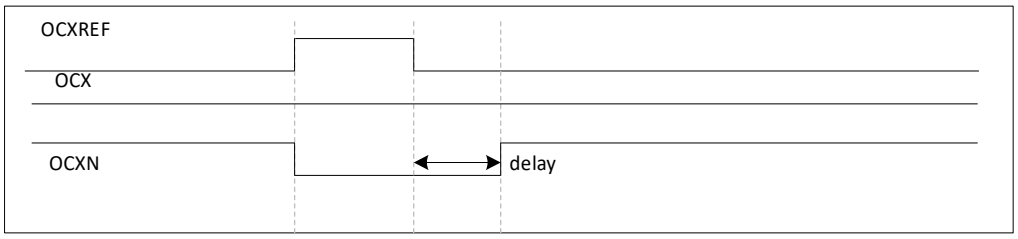


Figure 23-26 Dead-time waveforms with delay greater than the positive pulse.

When DTAE = 1, the rising edge dead time is configured by the DTG [7: 0] bit of TIMx_BDTR, and the falling edge dead time is configured by the DTGF [7: 0] bit of TIMx_DTR2. The writing of the DTAE needs to be before the counter is enabled. When CEN = 1, it is not allowed to modify the DTAE.

The dead time can be modified during the PWM operation through the mechanism of preloading. When the DTPE of TIMX_DTR2 is set, the dead time configuration registers DTG [7: 0] and DTGF [7: 0] are preloaded, and the preloaded data is loaded intothe cache register at the next update event.

Note: If the DTPE bit is set while the counter is enabled, new data written since the last update is discarded and the previous value is used

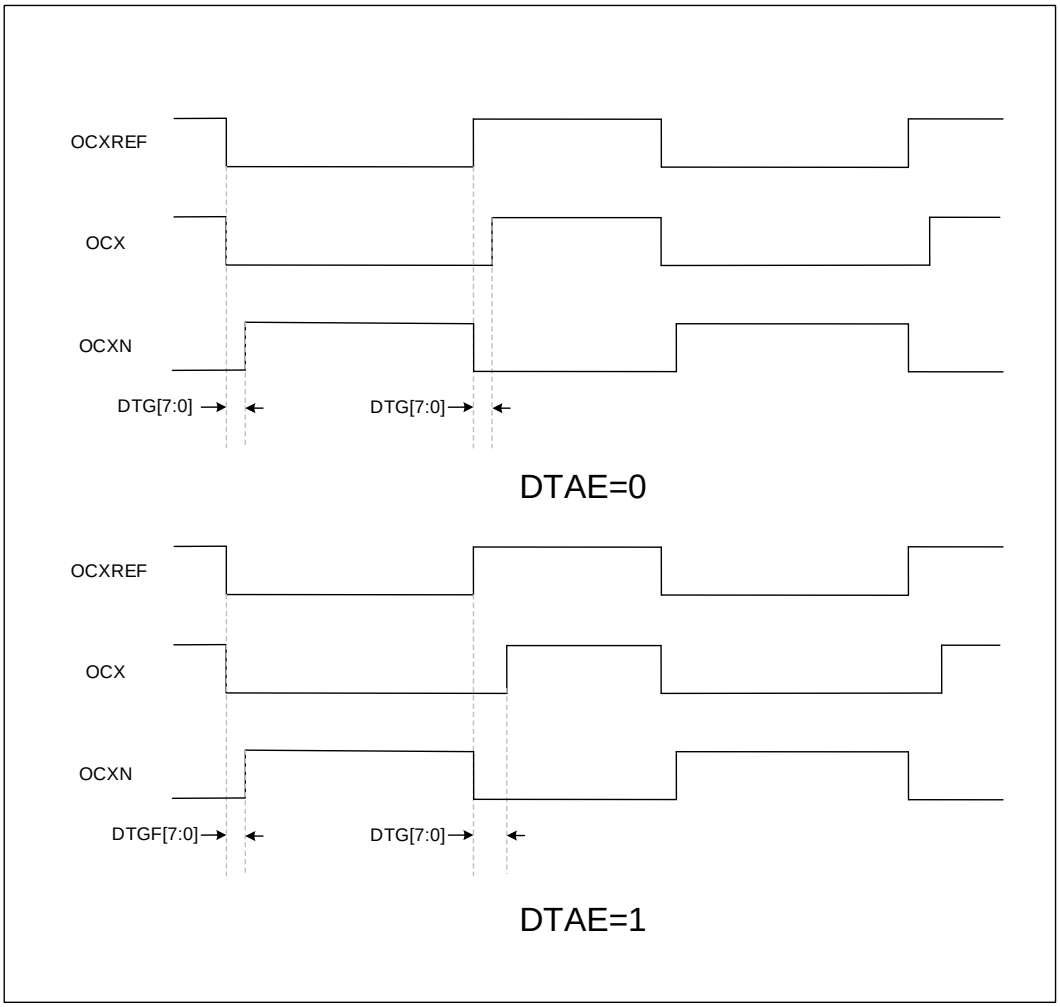


Figure 23-27 asymmetric dead time

23.2.12.1 Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register.

This allows you to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other alternative possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it does not invert and becomes high immediately when OCxREF is active. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

23.2.13 Using the break function

The purpose of the brake function is to protect the power switch driven by the PWM output from the TIMER. The input to the two-way brakes is usually the fault output connected to the three-phase inverter and power stage chip. When the brake signal is active, it will immediately turn off all outputs of the PWM and force them to a preset safe level state. Some internal events of the MCU can also be used as trigger signals to turn off the output.

There are two channels of braking events. Channel 1 includes system-level faults (clock failure, ECC, parity, etc.) and application-level faults (through input pins or built-in comparators), and can be forced to output to a preset level (whether working or idle) after a dead zone. Channel 2 contains only some application layer faults and forces the output to an invalid state.

The output enable signal and output level during braking are controlled by the following bits

- The MOE bit of TIMx_BDTR can be enabled and turned off by software or 2-way brake event reset
- The OSSI bit of TIMx_BDTR is used to define whether the timer also controls the output when idle, or releases control of the GPIO controller (high impedance state)
- The OISx and OISxN bits of TIMx_CR2 are used to set the output to an invalid level regardless of the working or idle state, andthe OCxandOCxN outputscannot be set to an active level at the same time at the same time, regardless of the values of OISx and OISxN.

When exiting from reset, the brake function is turned off and the MOE is 0. The brake function can be turned on by enabling the BKE bit of TIMxBDTR. The polarity of the brake inputs can be configured via BKP. BKEx and BKPx can be modified simultaneously. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1APB clock period to correctly read back the bit after the write operation.

MOE falling edge can be asynchronous, thus a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

A source of the brake channel;

- An external source can be connected to one of the TIMx_BKIN pins (select GPIO and configuration registers), plus polarity selection and filtering
- The internal source contains 2 parts

- Signal from brake comparator (tim_brk_cmpx)
- From system brake request

The brake event can also be generated by software, and the BG bit of TIMx_EGR can be set, with all brake sources OR up as the tim_brk brake input. Figure below:

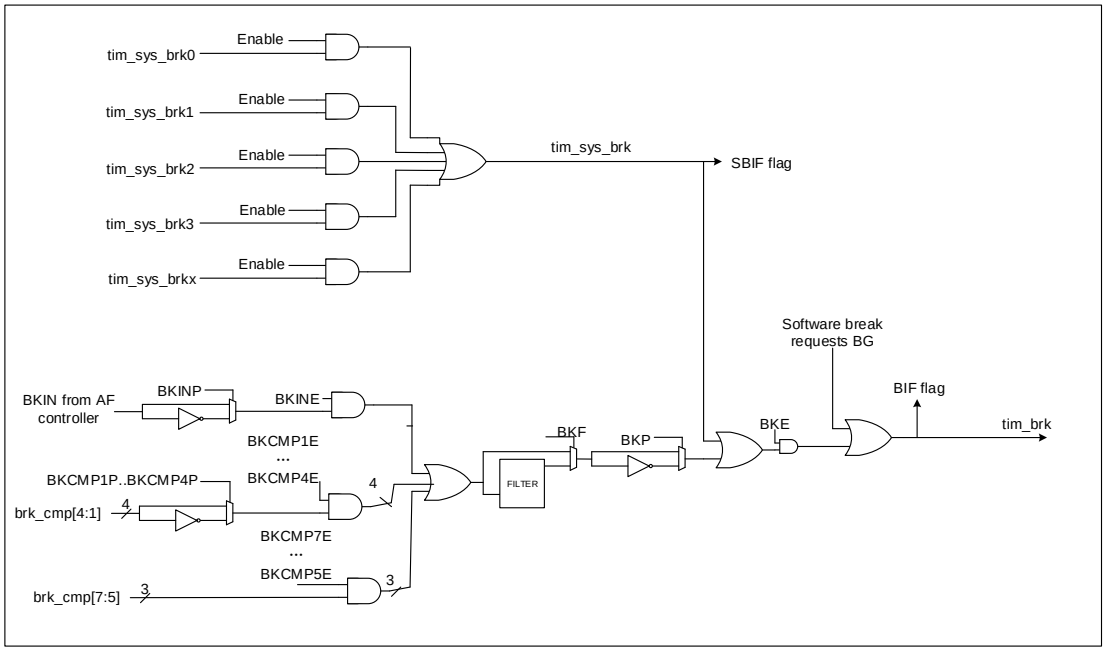


Figure 23-28 Brake circuit

Note: Asynchronous operation can only be guaranteed when there is no filtering. If filtering is turned on, a failsafe clock mode must be used to ensure that brake interrupt events are handled (e.g. using internal PLL or CSS, no filtering).

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, placing the output in an invalid state, an idle state, or releasing control of the GPIO controller (selected by the OSSI bit). This feature functions even if the MCU oscillator is off.
- Each output channel is driven with the level programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSSI = 0, the timer releases output control, otherwise the enable output is always high.
- When complementary outputs are used:

  -The output is first put into a reset state, i.e. an invalid state (depending on polarity). This is done asynchronously so that it works even if no clock is provided to the timer.

  ─ If the clock of the timer still exists, the dead time generator will re-take effect, driving the output port according to the level indicated by the OISx and OISxN bits after the dead time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that the dead time is longer than usual (approximately 2 clock cycles) because of the resynchronization of the MOE.

  -if OSSI = 0, the timer releases the output, otherwise keeps the output enabled; Or once one of CCxE and CCxNE becomes high, the enable output becomes high.
- The brake status flag is pulled up (BIF of TIMx_SR). If the BIE bit in the TIMx_DIER register is set, an interrupt is generated when the brake status flag (BIF bit in the TIMx_SR register) is' 1 '.

If the TDE bit in the TIMx_DIER register is set, a DMA request is generated.

- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Otherwise, the MOE remains low until it is set to '1' again; At this time, this feature can be used in safety. You can connect the brake input to the power-driven alarm output, thermal sensor or other safety devices.

Note: When AOE is 1, if MOE is reset by the CPU, the output will enter the idle state and be forced to an invalid level or high impedance (depending on the value of OSSI). If both MOE and AOE are reset by the CPU, the output will enter the invalid state and the output level is driven by the OISx bit.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF can not be cleared.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows the user to freeze several configuration parameters (dead time duration, OCx/OCxN polarity and disabled state, OCxM configuration, brake enable and polarity). The user can select one of the three levels of protection by setting the LOCK bit in the TIMx_BDTR register, see Brake and Dead Time Register (TIMx_BDTR). The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

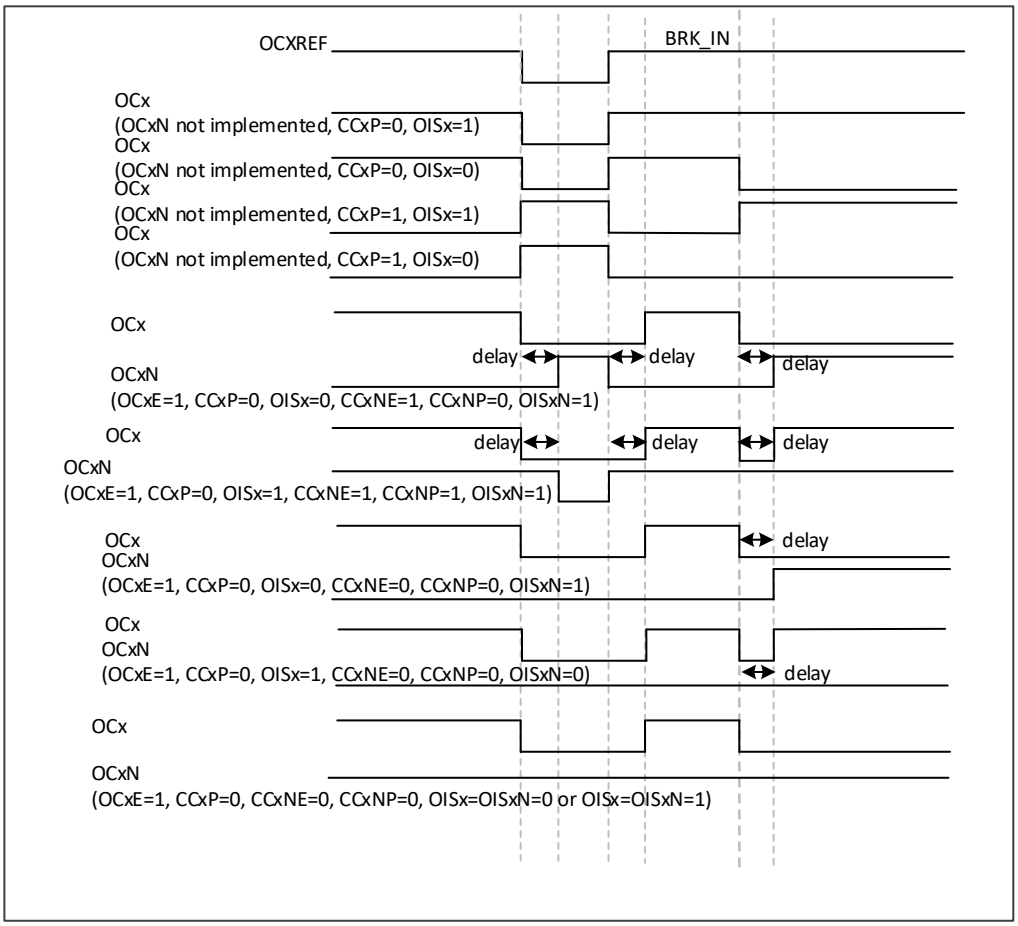


Figure 23-29 Output behavior in response to a break

23.2.14 Bidirectional brake input

TIM15/16/17 has two-way brake I/O, as shown in the figure below:

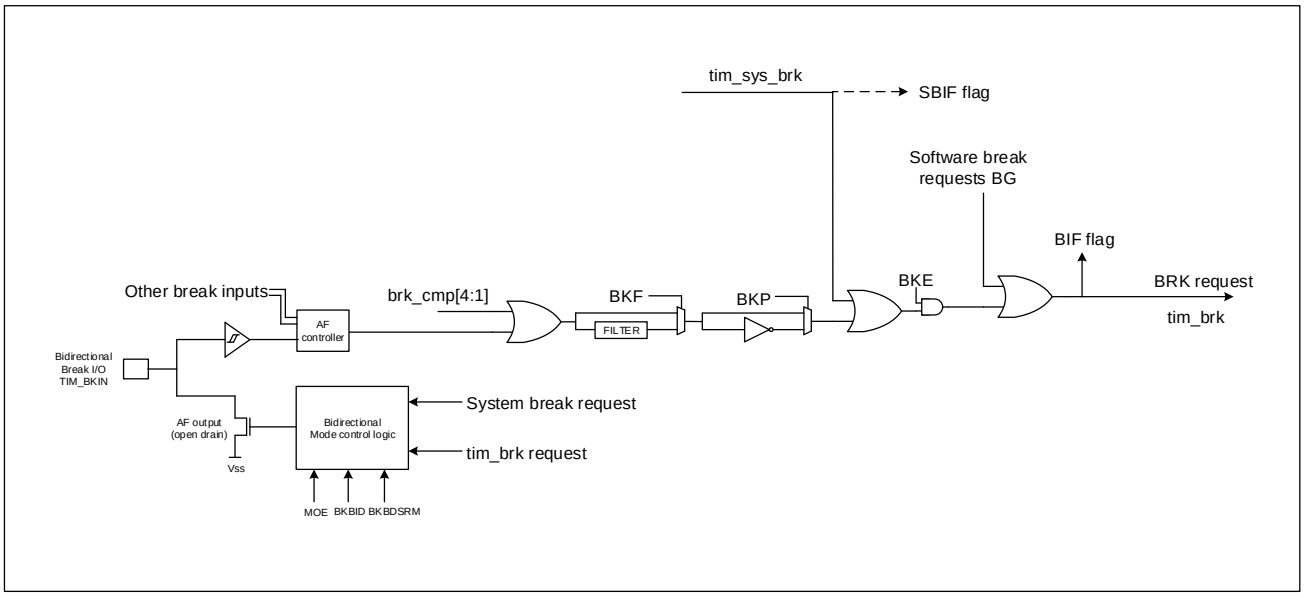


Figure 23-30 Output Redirection

I/O support features:

- Provides a board-level global brake signal that sends a fault signal to the external MCU and gate driver through a unique pin that can be used as both input and output
- When it is necessary to combine a plurality of internal and external brake sources, the internal brake source and the plurality of external open-drain sources are OR calculated as the unique trigger signal of the braking event

The input of tim_brk is controlled to be bidirectional by the BKBID bit of TIMxBDTR. The BKBID can be set to 1 using the LOCK bit (LOCK level 1) of TIMxBDTR, and the implementation is locked into read-only mode.

The tim_brk input may use bidirectional modeand requires I/O to be configured in open-drain mode with low activepolarity (via BKINP, BKP control bits). Whether from the system (e.g. CCS), from the board-level peripheral, or the brake input forcing a low level indicates a brake input failure event, it can be used as a brake request. However, for safety reasons, when the polarity bit is not effectively configured (when configured to be active high), the bidirectional mode is disabled.

The software braking event (BG) will also cause the brake I/O to be pulled down, which is used to show external devices that this timer has entered the braking state. Of course, it can only be possible when BKE is 1. When BKE is 0, a software brake event occurs, the output is set to a safe state and a brake flag signal is generated, but it has no effect on TIMx_BKIN I/O

A safe release mechanism prevents the system from being restrictively locked (a low level at the brake input triggers a brake, which in turn forces the input to low level)

When the BKDSRM bit is set to 1, the brake output control can be released, clearing the fault flag signal and providing the possibility to reconfigure the system.

Under no circumstances should the brake protection circuit be turned off:

- The brake input path is always active: even if the BKDSRM bit is set to 1 and the open-drain control is released, the brake event is still active. This prevents that PWM output from being re-started while still in the brake state
- When the output is enabled (MOE = 1), the BKDSRM bit cannot release brake protection

Table 23-2 Brake protection status

MOE	BKBID	BKDSRM	Brake protection status
0	0	X	Enable
0	1	0	Enable
0	1	1	1 1 Disarmed
1	X	X	Enable

23.2.14.1 Enable and re-enable the brake circuit

By default, the brake circuit (in input or bi-directional mode) is enabled

Follow the following procedure to restart the protection after braking occurs:

- The BKDSRM bit must be set to 1 to release output control
- The software must wait until the system brake status disappears and then clear the SBIF status flag (or systematically clear it before re-enabling)
- The software must poll the BKDSRM bit until it is cleared by the hardware (when the application brake state disappears)

In this way, the brake circuit is activated and the MOE bit can be re-enabled for the PWM output

23.2.15 Clearing the OCxREF signal on an external event

For a given channel, setting the corresponding OCxCE bit in the TIMx_CCMRx register to '1' enables the OCxREF signal to be pulled low with a high level at the OCREF_CLR_INT input, and the OCxREF signal will remain low until the update event UEV generated by the next count overflow occurs. This function can only be used in output compare and PWM modes. It does not work in Forced mode. The OCREF_CLR_INT input can be selected among several inputs as shown in the figure below.

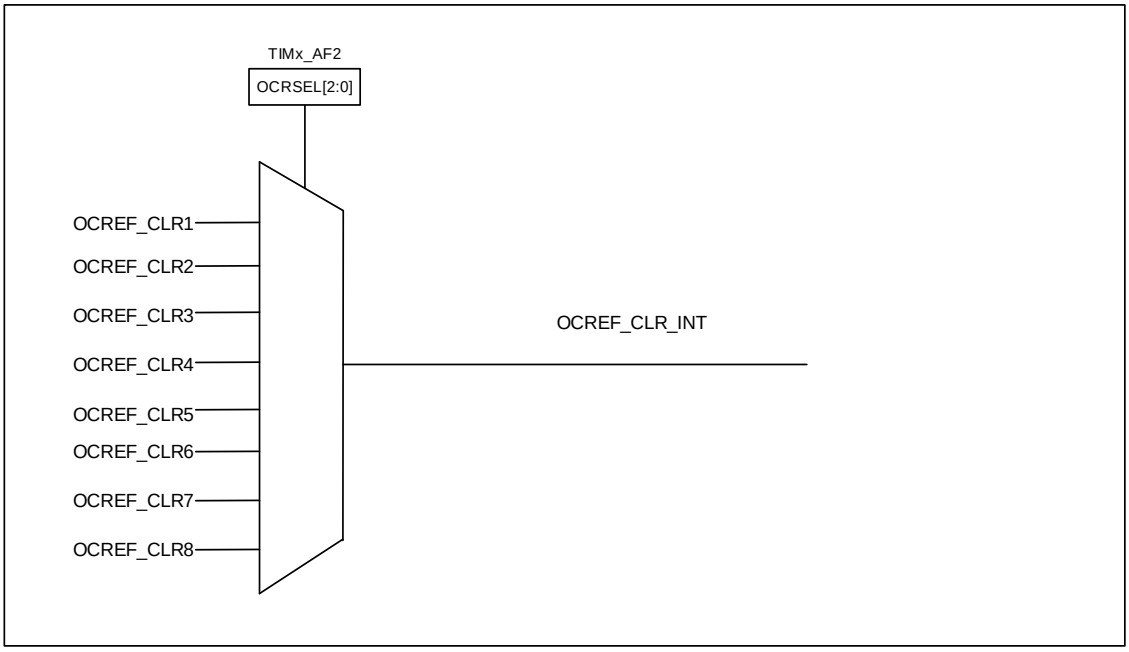


Figure 23-31 OCREF_CLR input selection

23.2.16 Generate a six-step PWM output

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. These preload bits are transferred to the shadow register when a COM commutation event occurs. In this way, you can pre-set the configuration of the next step and change the configuration of all channels at the same time. COM can be generated by software by setting the COMG bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

When a COM event occurs, a flag bit (COMIF bit in the TIMx_SR register) will be set. At this time, if the COMIE bit of the TIMx_DIER register has been set, an interrupt will be generated; If the COMDE bit of the TIMx_DIER register has been set, a DMA request is generated.

The figure below describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

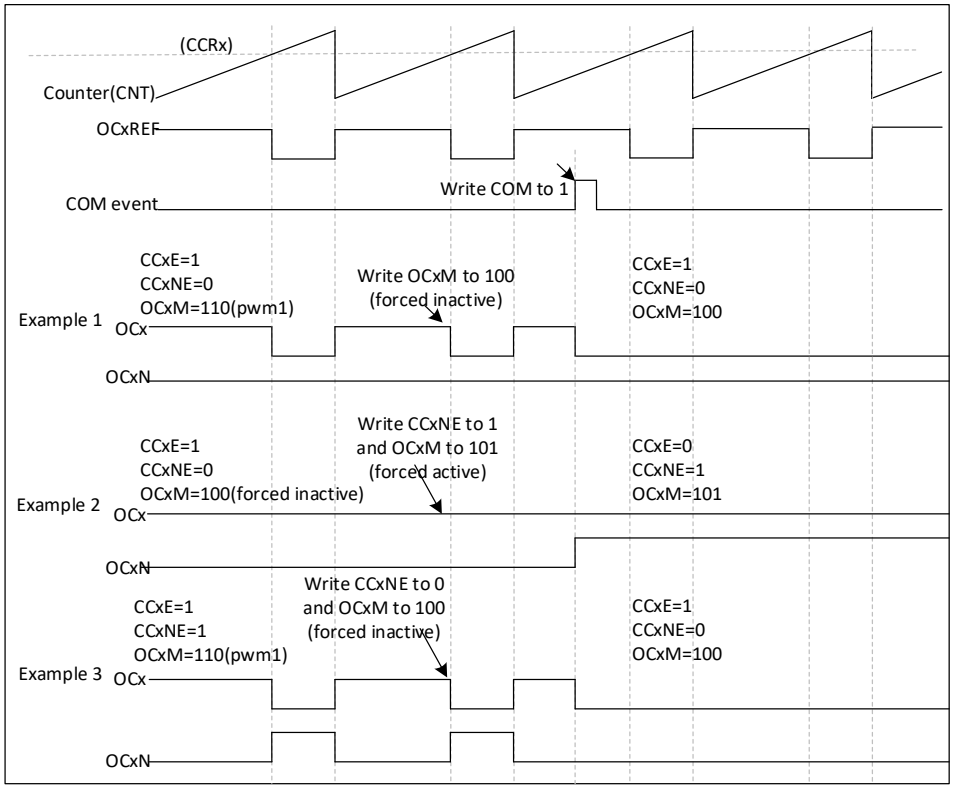


Figure 23-32 generates a six-step PWM, using COM example (OSSR = 1)

23.2.17 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV. A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- Up count mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx),
- Down counting mode: Counter CNT > CCRx.

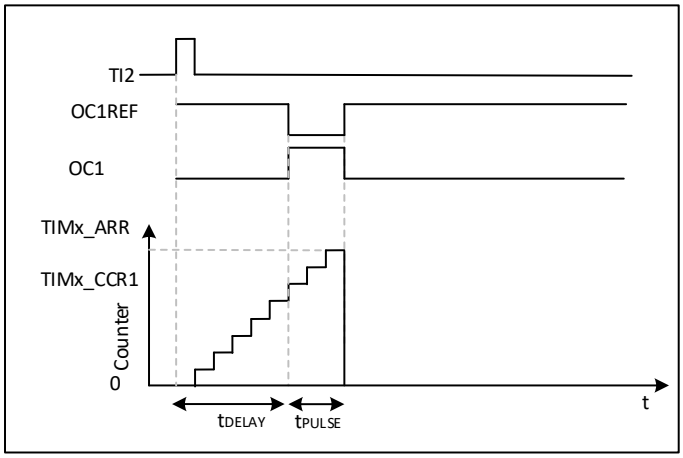


Figure 23-33 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Assume TI2FP2 as the trigger:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let’s say one want to build a waveform with a transition from ‘0’ to ‘1’ when a compare match occurs and a transition from ‘0’ to ‘1’ when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to ‘0’ in this example.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0) When OPM bit in the TIMx_CR1 register is set to '0', so the repetitive mode is selected.

23.2.17.1 Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay tDELAY min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF (and OCx) is forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

23.2.18 Retriggerable single pulse mode (tim15 only)

This mode allows the counter to start in response to excitation and generate pulses of programmable length, which differs from the non-retriggerable single pulse mode described in the previous section as follows:

- The pulse starts as soon as the trigger occurs (no programmable delay).
- If a new trigger occurs before the pulse generated by the previous trigger is completed, the pulse is extended.

To use the retriggerable monopulse mode, the timer must be in slave mode, the bit SMS [3: 0] = "1000" in the TIMx_SMCR register (combined mode-reset + trigger), and the OCxM [3: 0] bit set to "1000" or "1001" (retriggerable OPM mode 1 or 2).

Note: For compatibility reasons, the OCxM [3: 0] and SMS [3: 0] bit fields are split into two parts, with the most significant bit not contiguous with the 3 least significant bit positions.

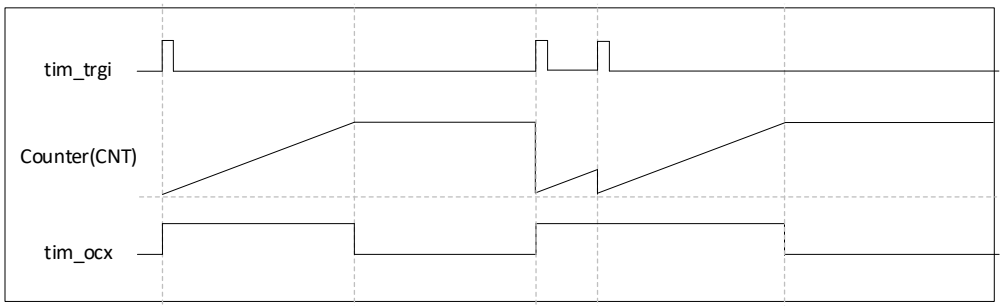


Figure 23-34 Example of a retriggerable monopulse mode

23.2.19 UIF bit remapping

The UIFREMAP bit of the TIMx_CR1 register can force a continuous copy of the UIF to the 31st bit of the timer counter (TIMxCNT [31]). This allows both the value of the counter and a potential rolling state exhibited by the UIFCPY flag signal to be read out. In special cases, this can simplify the calculation by avoiding race conditions, for example by simultaneous processing of counter readout and update interrupts.

The UIF and UIFCPY flags are generated without delay.

23.2.20 Timer input XOR function (tim15 only)

The TI1S bit in the TIMx_CR2 register allows the input filter of channel 1 to be connected to the output of an exclusive OR gate. The two inputs of the exclusive OR gate are TIMx_CH1 and TIMx_CH2.

The XOR output can be used with all the timer input functions such as trigger or input capture. It is useful for measuring the spacing between the edges of two input signals.

23.2.21 Synchronization of TIMx timer and externally triggered

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gated mode, trigger, reset + trigger and gated + reset mode.

23.2.21.1 Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; Meanwhile, if the UDIS bit of the TIMx_CR1 register is low, an update event UEV is also generated; All preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=0100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=00101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. At the same time, a trigger flag (TIF bit in the TIMx_SR register) is set, and an interrupt request or a DMA request is generated according to the setting of the TIE (Interrupt Enable) bit and the TDE (DMA Enable) bit in the TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

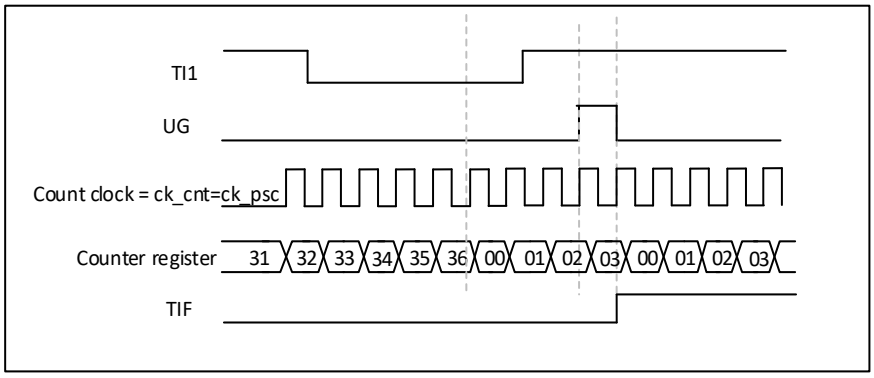


Figure 23-35 Control circuit in reset mode

23.2.21.2 Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. SelectTI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter

doesn't start if CEN=0, whatever is the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

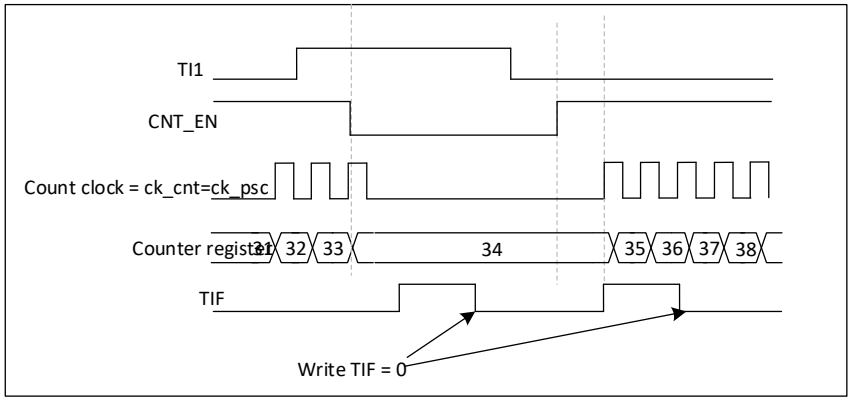


Figure 23-36 Control circuit in Gated mode

23.2.21.3 Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMx_CCMR1 register. Write CC2P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

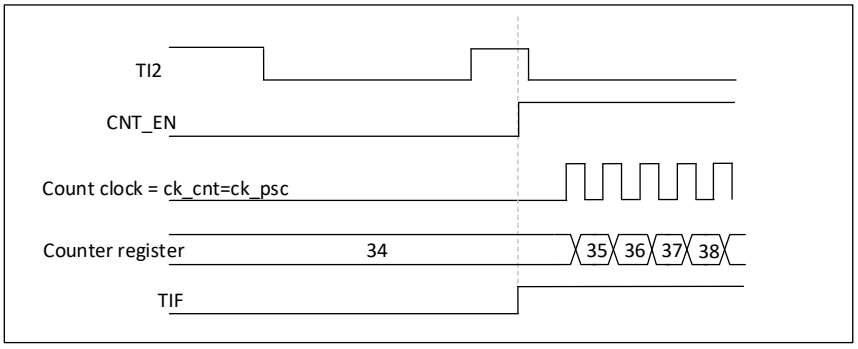


Figure 23-37 Control circuit in trigger mode

23.2.21.4 Slave mode: reset plus trigger combination mode (tim15 only)

In this mode, the rising edge of the trigger input initializes the counter and generates an update signal for the register, turning on the counting

This mode is used for a single pulse mode

23.2.21.5 Slave mode: gated plus reset combination mode (tim15 only)

The clock of the counter is enabled only when the trigger input is high. The counter stops when the trigger signal is pulled low. The start and stop of the counter can be controlled

This mode allows detection of out-of-range PWM signals (duty cycle exceeding maximum expected value)

23.2.22 Timer synchronization (tim15)

The TIMx timers are linked together internally for timer synchronization or chaining. See Section 20.2.22: Timer Synchronization for details.

23.2.23 DMA Burst Mode

The TIMx timer has the ability to generate multiple DMA requests when a single event occurs. The main purpose is to be able to reprogram some functions of the timer multiple times without the overhead of software. But it is also possible to read several registers in a row at fixed intervals.

The destination of the DMA controller is unique and must point to the virtual register TIMx_DMAR. When a given timer event occurs, the timer initiates a series of DMA requests. Each write to the TIMx_DMAR register actually redirects a timer register.

The DBL [4: 0] bit in the TIMx_DCR register sets the DMA burst length. When a read or write access is made to a TIMx_DMR address, the timer recognizes burst transmissions, that is, the number of transmissions (in half a word or byte)

The DBA [4: 0] bit of TIMx_DCR defines the DMA base address for DMA transmission (when read and write operations are completed through the TIMx_DMAR address). The DBA is defined as the offset from the TIMx_CR1 register address.

Example:

00000: TIMx_CR1

00001: TIMx_CR2

00010: TIMx_SMCR

As an example, the timer DMA burst characteristic is used to update the contents in the CCRx register when an update event occurs, transmitting a half-word to CCRx by DMA.

This is done through the following steps:

1. Configure the corresponding DMA channel as follows:
   a) The DMA channel peripheral address is the address of the DMAR register
   b) The DMA channel storage address is the BUFF address in RAM, which contains the data transmitted by DMA to CCRx
   c) Number of data transferred = 3 (see note)
   d) Polling Mode Off
2. Configure DCR registers by configuring DBA and DBL: DBL = 3, DBA = 0XE
3. Enable timer update DMA request (UDE = 1)

4. Enable timer
5. Enable the DMA channel

This example applies to the case where each CCRx register is updated once. If each CCRx is updated twice, the number of data transfers needs to be set to 4. For example, suppose that the buffer of RAM contains data1, data2, data3, and data4. Data is sequentially transmitted to CCRx: the first update DMA request, data1 is transmitted to CCR1, and data2 is transmitted to CCR2; For the second update of the DMA request, data3 is transmitted to CCR1 and data4 is transmitted to CCR2.

Note: Null values can be written to the reserved register

23.2.24 DMA Request

TIM15/TIM16/TIM17 can generate DMA requests as shown in the following table:

Table 23-3 DMA Request

DMA request source	DMA enable control bit
update	UDE
Compare/Capture 1	CC1DE
COM(1)	COMDE
Trigger (1)	TDE

Of which 1 is for TIM15 only.

23.2.25 Debug Mode

When the microcontroller enters debug mode (Cortex-M4F core stops), according to the setting of DBG_TIMx_STOP in the DBG module,

The TIMx counter may either continue normal operation or stop. See subsequent DBG sections for details.

23.2.26 TIM15/TIM16/TIM17 Interrupt

Interrupt event	Event flag	Interrupt enable control bit
update	UIF	UIE
Compare/Capture 1	CC1IF	CC1IE
Compare/Capture 2	CC2IF	CC2IE
COM	COMIF	COMIE
Trigger	TIF	TIE
Brake	BIF	BIE

23.3 TIM15 registers

23.3.1 TIM15 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DITHEN	UIFREMAP	Res	CKD [1: 0]		ARPE	Res			OPM	URS	UDIS	CEN
			RW	RW		RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 13	Reserved	-	-	-
12	DITHEN	RW	0	Jitter enable (Dithering enbale) 0: Jitter off 1: Jitter on Note: This bit can only be modified when the CEN bit is reset

11	UIFREMAP	RW	0	UIF status bit remapping enable 0: No remapping, UIF status bit is not copied to bit 31 of TIMx_CNT register 1: Remap, UIF status bit copied to bit 31 of TIMx_CNT register
10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the frequency division ratio between the timer clock (CK_INT) frequency and dead time and the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 4	Reserved	-	-	-
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

23.3.2 TIM15 control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					OIS2	OIS1N	OIS1	TI1S	MMS [2: 0]			CCDS	CCUS	Res	CCPC
					RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10	OIS2	RW	0	Output idle state 2 (OC2 output) Refer to OIS1 bit.
9	OIS1N	RW	0	Output Idle state 1 0: OC1N = 0 after dead time when MOE = 0; 1: OC1N = 1 after dead zone when MOE = 0. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
8	OIS1	RW	0	Output Idle state 1 0: when MOE = 0, OC1 = 0 after dead time if OC1N is achieved; 1: When MOE = 0, OC1 = 1 after dead time if OC1N is implemented. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
7	TI1S	RW	0	TI1 selection 0: TIMx_CH1 pin connected to TI1 input; 1: The TIMx_CH1, TIMx_CH2, and TIMx_CH3 pins are connected to the TI1 input via XOR.
6: 4	MMS	RW	3'h0	Main mode selection (Master mode selection) For selecting synchronization information (TRGO) to be sent to the slave timer in master mode. The combination is as follows: 000: Reset-The UG bit of the TIMx_EGR register is used as a trigger output (TRGO). If the reset is generated by the trigger input (the slave mode controller is in reset mode), the signal on the TRGO will have a delay relative to the actual reset. 001: Enable-Counter enable signal CNT_EN may be used as a trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timeris enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 010: Update-The update event is selected as Trigger Output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Comparison Pulse-When a capture occurs or a comparison is successful, when the CC1IF flag is to be set (even if it is already high), the trigger output sends a positive pulse (TRGO). 100: Compare - OC1REFC signal is used as trigger output (TRGO) 101: Compare - OC2REFC signal is used as trigger output (TRGO) Other: Reserved Note: 1. The clock of the slave timer or ADC must be enabled prior to receive events from the master timer, and must not be changed. 2. If the master and slave timers are not on the same bus, the master mode should be configured to the width that can be picked by the slave timer.
3	CCDS	RW	0	Capture/compare DMA selection 0: When a CCx event occurs, send a DMA request for CCx; 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/Compare Control Update Selection (Capture/compare control update selection) 0: If the capture/compare control bits are preloaded (CCPC = 1), they can only be updated by setting the COM bit; 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	-
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE, and OCxM bits are not preloaded; 1: CCxE, CCxNE, and OCxM bits are preloaded; When this bit is set, they are only updated after the COM bit is set (COMG bit setting or rising edge detected on tim_trgi, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

23.3.3 TIM15 slave mode control register (TIMx_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										TS [4: 3]		Res			SMS[3]
										RW	RW				RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								MSM	TS [2: 0]			Res	SMS [2: 0]		
								RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 22	Re-served	-	-	-
21: 20	TS [4: 3]	RW	2'h0	See TS [2: 0] description for details
19: 17	Re-served	-	-	-
16	SMS[3]	RW	0	See SMS [2: 0] description for details
15: 8	Re-served	-	-	-
7	MSM	RW	0	Master/slave mode 0: No action; 1: The event on the trigger input (TRGI) is delayed to allow perfect synchronization between the current timer (via TRGO) and its slave timers. It is useful if we want to synchronize several timers on a single external event.
6: 4	TS	RW	3'h0	Trigger selection Trigger selection Select the trigger input used to synchronize the counter. 00000: Internal Trigger 0 (ITR0) 00001: Internal Trigger 1 (ITR1) 00010: Internal Trigger 2 (ITR2) 00011: Internal Trigger 3 (ITR3) 00100: TI1 edge detection (TI1F_ED) 00101: Filtered timer input 1 (TI1FP1) 00110: Filtered timer input 2 (TI1FP2) 00111 external trigger input (ETRF) 01000: Internal Trigger 4 (ITR4) 01001: Internal Trigger 5 (ITR5) 01010: Internal Trigger 6 (ITR6) 01011: Internal Trigger 7 (ITR7) 01100: Internal Trigger 8 (ITR8) 01101: Internal Trigger 9 (ITR9) 01110: Internal Trigger 10 (ITR10) 01111: Internal Trigger 11 (ITR11) 10000: Internal Trigger 12 (ITR12) 10001: Internal Trigger 13 (ITR13) 10010: Internal Trigger 14 (ITR14) 10011: Internal Trigger 15 (ITR15) 10100: Internal Trigger 16 (ITR16) 10101: Internal Trigger 17 (ITR17) 10110: Internal Trigger 18 (ITR18) Others: Reserved For more details about ITRx, see the System Cascade Table Note: These bits can only be changed when they are not used (e.g. SMS = 000) to avoid erroneous edge detection when changing.
3	Re-served	-	-	-
2: 0	SMS	RW	3'h0	Select from Mode (Slave mode selection) When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 0000: Slave mode disabled - if CEN = 1 then the prescaler is clocked directly by the internal clock. 0001: Reserved.

				0010: Reserved. 0011: Reserved. 0100: Reset Mode-The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a signal to update the register. 0101: Gated mode-When the trigger input (TRGI) is high, the clock of the counter is turned on. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger mode-The counter starts (but does not reset) on the rising edge of the trigger input TRGI, only the start of the counter is controlled. 0111: External clock mode 1-The rising edge of the selected trigger input (TRGI) drives the counter. 1000: Reset + Trigger Combination Mode:-Rising edge of Trigger Input (TRGI) initializes counter, generates register update and starts counter 1001: Gate + Reset Combination Mode:-Trigger input (TRGI) is high when counter clock enabled. The counter stops and resets once the trigger signal is pulled low. The start and stop of the counter are controllable. Other: Reserved Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS=00100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. Note: The slave device (timer, ADC) must be enabled to receive TRGO before receiving the master timer event, and the clock frequency can no longer be changed during reception

23.3.4 TIM15 DMA/interrupt enable register (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TDE	COMDE	Res			CC1DE	UDE	BIE	TIE	COMIE	Res		CC2IE	CC1IE	UIE
	RW	RW				RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 15	Reserved	-	-	-
14	TDE	RW	0	Trigger DMA request enable 0: prohibit triggering of DMA request; 1: Allow DMA request to be triggered.
13	COMDE	RW	0	COM DMA request enable 0: DMA request of COM is prohibited; 1: Allow DMA requests for COM.
12: 10	Reserved	-	-	-
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: DMA request to capture/compare 1 is prohibited; 1: DMA request allowed to capture/compare 1. 。
8	UDE	RW	0	Update DMA request enable 0: DMA request to prohibit update; 1: Allow updated DMA requests.
7	BIE	RW	0	Allow brake interruption (Break interrupt enable) 0: Brake interruption is prohibited; 1: Allow the brake to be interrupted.
6	TIE	RW	0	Trigger interrupt enable (Trigger interrupt enable) 0: prohibit triggering interrupt; 1: Enable trigger interrupt.
5	COMIE	RW	0	COM interrupt enable 0: Disable COM interrupt; 1: COM interrupt enabled
4: 3	Reserved	-	-	-
2	CC2IE	RW	0	Allow capture/compare 2 interrupts (Capture/Compare 2 interrupt enable) 0: Disable capture/compare 2 interrupt; 1: Allow capture/compare 2 interrupts.
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable)

				0: Disable capture/compare 1 interrupt; 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

23.3.5 TIM15 status register (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										IC2IF	IC1IF	Res		IC2IR	IC1IR
										RC_W0	RC_W0			RC_W0	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res					CC2OF	CC1OF	Res	BIF	TIF	COMIF	Res		CC2IF	CC1IF	UIF
					RC_W0	RC_W0		RC_W0	RC_W0	RC_W0			RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21	IC2IF	RC_W0	0	Falling edge capture 2 flag Refer to IC1IF description
20	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by falling edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A falling edge capture event occurs.
19: 18	Reserved	-	-	-
17	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IR description
16	IC1IR	RC_W0	0	Rising edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by rising edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A rising edge capture event occurs.
15: 11	Reserved	-	-	-
10	CC2OF	RC_W0	0	Capture/Compare 2 Repeat Capture Markers (Capture/Compare 2 overcapture flag) Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 Repeat Capture Marker (Capture/Compare 1 over-capture flag) This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been generated. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	Reserved	-	-	-
7	BIF	RC_W0	0	Brake interruption mark (Break interrupt flag) Once the brake input is active, the position '1' is applied by the hard-ware. If the brake input is invalid, the bit may be cleared '0' by the soft-ware. 0: No break event occurred. 1: An active level has been detected on the break input. Generate an interrupt if BIE = 1 for TIM_DIER
6	TIF	RC_W0	0	Trigger interrupt flag (Trigger interrupt flag) This position is' 1 'by hardware when a trigger event occurs (a valid edge is detected at the TRGI input when the slave mode controller is in a mode other than gated mode, or either edge in gated mode). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.

5	COMIF	RC_W0	0	COM interrupt flag Once a COM event is generated (when the capture/compare control bits: CCxE, CCxNE, OCxM have been updated) this bit is set '1' by the hardware. It is cleared by software. 0: No update occurred. 1: COM interrupt waiting for response.
4: 3	Reserved	-	-	-
2	CC2IF	RC_W0	0	Capture/Compare 2 Interrupt Markers (Capture/Compare 2 interrupt flag) Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes high under the counter overflow condition in the up or up/down count mode, or under the counter underflow condition in the down count mode If channel TI1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0 and the repetition counter = 0, an update event is generated. − If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

23.3.6 TIM15 event generation register (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BG	TG	COMG	Res		CC2G	CC1G	UG
								W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	BG	W	0	Break generation This bit is set '1' by software to generate a braking event, and '0' is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	W	0	Trigger generation This bit is set '1' by the software to generate a trigger event, and '0' is automatically cleared by the hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	COMG	W	0	Capture/compare events, generate control updates Capture/Compare control update generation

				This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit is only valid for channels with complementary outputs.
4: 3	Reserved	-	-	-
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action 1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged). The counter is cleared '0' if in centrosymmetric mode or DIR = 0 (count up); If DIR = 1 (count down), the counter takes the value of TIMx_ARR.

23.3.7 TIM15 Capture/Compare Mode Control Register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							OC2M[3]	Res.							OC1M[3]
							RW								RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M [2: 0]			OC2PE	OC2FE	CC2S [1: 0]		OC1CE	OC1M [2: 0]			OC1PE	OC1FE	CC1S [1: 0]	
IC2F [3: 0]				IC2PSC [1: 0]				ICIF [3: 0]				ICIPSC [1: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

23.3.7.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	-
24	OC2M[3]	RW	0	See OC2M [2: 0] description for details
23:17	Reserved	-	-	-
16	OC1M[3]	RW	0	See OC1M [2: 0] description for details
15	OC2CE	RW	0	Output Compare 2 Clear Enable (Output Compare 2 clear enable)
14:12	OC2M	RW	3'h0	Output Comparison 2 Mode Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 Preload Enable (Output Compare 2 preload enable)
10	OC2FE	RW	0	Output Compare 2 Fast Enable (Output Compare 2 fast enable)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection Capture/Compare 2 selection This bit defines the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable

				0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	OutputCompare1mode These bits define the action of outputting the reference signal OC1REF, and OC1REF determines the values of OC1, OC1N. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 0001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high. 0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 0011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 0100: Force inactive level OC1REF is forced low. 0101: Force active level OC1REF is forced high. 0110: PWM Mode 1-On up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 0111: PWM Mode 2-On up count, channel 1 is invalid level once TIMx_CNT < TIMx_CCR1, active level otherwise; When counting down, channel 1 is active once TIMx_CNT > TIMx_CCR1, otherwise it is invalid. 1000: Retriggerable OPM Mode 1-In incremental count mode, the channel is active until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 1, and the channel becomes valid again at the next update. In decrement count mode, the channel is invalid until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 1, and the channel becomes invalid again at the next update 1001: Retriggerable OPM Mode 2-In incremental count mode, the channel is in an invalid state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM mode 2, and the channel becomes invalid again at the next update. In the decrement count mode, the channel is in an active state until a trigger event (tim_trgi signal) is detected. Then, the comparison is performed as in PWM Mode 2, and the channel becomes active again at the next update 1010: Reserved 1011: Reserved 1100: Combined PWM Mode 1-OC1REF has the same behavior as in PWM Mode 1, OC1REFC is composed of OC1REF and OC2REF or logic 1101: Combined PWM Mode 2-OC1REF has the same behavior as in PWM Mode 2, OC1REFC is composed of OC1REF and OC2REF AND logic 1110: Reserved 1111: Reserved Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: In PWM mode, the OC1REF level is changed only when the comparison result is changed or when the output comparison mode is switched from freeze mode to PWM mode. Note 3: This bit segment can be preloaded when the channel has complementary outputs. If the CCPC position of TIMx_CR2 is 1, then the OC1M valid bit is updated with a new value from the preload only when the COM event is generated
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output).

				Note 2: Only in monopulse mode (OPM = 1 of TIMx_CR1 register), PWM mode can be used without confirming the preload register, otherwise its operation is uncertain.
2	OC1FE	RW	0	Output Compare 1 Fast Enable (Output Compare 1 fast enable) This bit is used to speed up the output response to the triggering input event. Must be used in single pulse mode (OPM position 1 of the TIMx_CR1 register) to achieve fast pulse output when the trigger comes. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. Only available for PWM Mode 1 and PWM Mode 2
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

23.3.7.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 12	IC2F	RW	4'h0	Input Capture 2 Filter (Input capture 2 filter)
11: 10	IC2PSC	RW	2'h0	Input/Capture 2 Prescaler (Input capture 2 prescaler)
9: 8	CC2S	RW	2'h0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC2S is writable only when the channel is closed (CC2E = 0 in the TIMx_CCER register).
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input;

				01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

23.3.8 TIM15 Capture/Compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								CC2NP	Res	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
								RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	CC2NP	RW	0	Input/capture 2 complementary output polarity (Capture/Compare 2 output polarity) Refer to CC1NP description.
6	Reserved	-	-	-
5	CC2P	RW	0	Input/capture 2 output polarity (Capture/Compare 2 output polarity) Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 Output Enable (Capture/Compare 2 output enable) Refer to CC1E description.
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configuration as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/capture 1 complementary output enable (Capture/Compare 1 implementary output enable) 0: OFF-OC1N disables output, so the level of OC1N depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. 1: ON-The OC1N signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 inverted (gated mode).

				10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output, so the output level of OC1 depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: ON-The OC1 signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Notes: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 23-4 Control bits for complementary output channels OCx and OCxN with brake function

Control bits					**Output states**	
MOE bit	**OSSI bit**	**OSSR bit**	**CCxE bit**	**CCxNE bit**	**OCx output status**	**OCxN output state**
1	x	0	0	0	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	0	1	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
		1	0	0	Output disabled (not driven by the timer anymore). OCx = CCxP, OCx = 0 OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = CCxNP, OCx = 0 OCxN_EN = 0
		1	0	1	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
0	0	x	x	x	Output disabled (not driven by the timer anymore).	

	1		0	0	
	1		0	1	Off state (output enabled but invalid level) Asynchronous: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCx_EN = 1; if brake or brake 2 is triggered Then if the clock exists (only valid when brake 1 is triggered, brake 2 is not needed): OCx = OISx, OCxN = OISxN after a dead time, assuming that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN Note: Brake 2 is only for OSSI = OSSR = 1
	1		1	0	
	1		1	1	

In particular, when the dead band is active, the output is always output according to the rules of dead band output, although moe may have been set to 1 by software or hardware at this time.

If neither of the two outputs of a channel is used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP, and CCxNP must all be cleared.

23.3.9 TIM15 Counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIFCPY	Res														
R															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	UIFCPY	R	0	This bit is a read-only register, and the UIF bit of the TIMx_ISR register is copied. If the UIF remap bit in TIMx_CR1 is reset, the bit is reserved and read out as 0
30:16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

23.3.10 TIM15 Prescaler (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes a counter being The UG bit of TIM_EGR is cleared '0' or is cleared '0' by a slave controller operating in reset mode.

23.3.11 TIM15 Automatic Reload Register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												ARR [19:16]			
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Re-served	-	-	-
19: 0	ARR	RW	20'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null. No jitter mode (DITHEN = 0): This register holds that auto-load value Dither Mode (DITHEN = 1): The register holds an integer portion ARR [19: 4], and ARR [3: 0] contains a dither portion

23.3.12 TIM15 Repeat Count Register (TIMx_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								REP							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Re-served	-	-	-
7: 0	REP	RW	8'h0	Value of repeat counter Repetition counter value When the preload function is turned on, these bits allow the user to set the update rate of the comparison register (i.e., periodically transfer from the preload register to the current register); If an update interrupt is allowed, the rate at which the update interrupt is generated will also be affected. Each time the down counter reaches 0, an update event is generated and the repeat counter resumes counting from the REP value. The new value written to the TIMx_RCR register only takes effect when the next update event occurs.

23.3.13 TIM15 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR1 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Re-served	-	-	-
19: 0	CCR1	RW/RO	20'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs.

				The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR1 [15: 0] and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR1 [19: 4], and CCR1 [3: 0] contains a dither portion If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR1 [15: 0], and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR1 [19: 4], and the CCR1 [3: 0] bit is reset

23.3.14 TIM15 capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR2 [19:16]			
												RW/RO			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	CCR2	RW/RO	20'h0	Capture/Compare 2 value for channel 2 If channel CC2 is configured as output: CCR2 contains the value loaded into the current capture/compare 2 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC2PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR2 [15: 0], and the CCR2 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR2 [19: 4], and CCR2 [3: 0] contains a dither portion If channel CC2 is configured as input: CCR2 contains the counter value transmitted by the last input capture 2 event (IC2). TIMx_CCR2 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR2 [15: 0], and the CCR2 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR2 [19: 4], and the CCR2 [3: 0] bit is reset

23.3.15 TIM15 Brake and Dead Time Register (TIMx_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			BKBID	Res	BKDSRM	Res						BKF [3: 0]			
			RW		RW							RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1: 0]		DTG [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28	BKBID	RW	0	Break bidirectional 0: Brake input BRK in input mode 1: Brake input BRK in bi-directional mode In the bi-directional mode (BKBID is set to 1), the brake input is configured as both an input mode and an open-drain output mode. Any valid braking event will pull the brake input level down, indicating to external devices that a braking event has occurred internally Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified. Note: Any write operation to this bit will delay the APB clock from taking effect.
27	Reserved	-	-	-
26	BKDSRM	RW	0	Brake release (Break disarm) 0: Brake input BRK activated 1: Brake input BRK released This bit is cleared by hardware when there is no brake source The BKDSRM bit must be set by software to release the bidirectional output control (open drain output high impedance state) and then poll it until it is reset by hardware, indicating that the fault state has disappeared. Note: Any write operation to this bit will delay the APB clock from taking effect.
25: 20	Reserved	-	-	-
19: 16	BKF [3: 0]	RW	4'h0	Brake filter (Break filter) These bits define the frequency at which the BRK signal is sampled and the bandwidth at which the BRK is digital filtered. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK2 is asynchronous 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8 Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified.
15	MOE	RW	0	Main output enabled (Main output enable) Once the brake input is valid, the bit is asynchronously cleared '0' by the hardware. Depending on the setting value of the AOE bit, this bit can be cleared '0' by software or automatically set to 1. It is only valid for channels configured as output. 0: OC and OCN outputs are disabled or forced to idle state. 1: Turn on the OC and OCN outputs if the corresponding enable bits (CCxE, CCxNE bits of the TIMx_CCER register) are set. See TIMx Capture/Compare Enable Register (TIMx_CCER) for details of OC/OCN enabling.
14	AOE	RW	0	Automatic output enable (Automatic output enable) 0: MOE can only be set '1' by software; 1: The MOE can be set '1' by software or automatically set '1' at the next update event (if the brake input is invalid).

				Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified.
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
12	BKE	RW	0	Break enable 0: Disable brake input (BRK and CCS clock failure events); 1: Turn on the brake input (BRK and CCS clock failure events). Note: When LOCK level 1 is set (the LOCK bit in the TIMx_BDTR register), this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
11	OSSR	RW	0	"Off state" selection in running mode (Off-state selection for Run mode) This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0); 1: When the timer is not working, once CCxE = 1 or CCxNE = 1, first turn on OC/OCN and output an invalid level, and then set OC/OCN enable output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
10	OSSI	RW	0	"Off state" selection in idle mode (Off-state selection for Idle mode) This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0); 1: When the timer is not working, once CCxE = 1 or CCxNE = 1, the OC/OCN first outputs its idle level and then the OC/OCN enables the output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
9: 8	LOCK	RW	2'h0	Lock settings (Lock configuration) These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot be written to DTG, BKBID, BK2BID, BKE, BKP, AOE bits of TIMx_BDTR register and OISx/OISxN bits of TIMx_CR2 register; 10: Lock level 2, cannot write bits in lock level 1, nor can you write CC polarity bits (once the relevant channel is set to output by CCxS bits, the CC polarity bits are CCxP/CCNxP bits of the TIMx_CCER register) and OSSR/OSSI bits; 11: Lock level 3, cannot write bits in lock level 2, and cannot write CC control bits (CC control bits are OCxM/OCxPE bits of TIMx_CCMRx register once the relevant channel is set as output through CCxS bits); Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7: 0	DTG	RW	8'h0	Dead Generator Settings (Dead-time generator setup) This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG [7: 5] = 0xx = > DT = DTG [7: 0] × Tdtg, Tdtg = TDTS; DTG [7: 5] = 10x = > DT = (64 + DTG [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTG [7: 5] = 110 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTG [7: 5] = 111 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

23.3.16 TIM15 dead time register 2 (TIMx_DTR2)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														DTPE	DTAE
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								DTGF [7: 0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	DTPE	RW	0	Deadtime preload enable 0: The value of dead time cannot be preloaded 1: The value of the dead time can be preloaded Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
16	DTAE	RW	0	Deadtime asymmetric enable 0: The rising and falling edges have the same dead time, both configured by DTG [7: 0] 1: Rising edge dead time is configured by DTG [7: 0] and falling edge dead time is configured by DTGF [7: 0] Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
15: 8	Reserved	-	-	-
7: 0	DTGF [7: 0]	RW	8'h0	Falling edge dead time generator Dead-time falling edge generator setup This bit segment defines the dead time of the falling edge when the complementary output is inserted, assuming DT denotes its duration: DTGF [7: 5] = 0xx = > DTF = DTGF [7: 0] × Tdtg, Tdtg = TDTS; DTGF [7: 5] = 10x = > DTF = (64 + DTGF [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTGF [7: 5] = 110 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTGF [7: 5] = 111 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified

23.3.17 TIM15 input select register (TIMx_TISEL)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				TI2SEL [3: 0]				Res				TI1SEL [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11: 8	TI2SEL	RW	4'h0	TI2 input selection (tim_TI2 [15: 0] input) 0000: TIMx_CH2 0001: ti_in1 0010: ti_in2 Other: Reserved
7: 4	Reserved	-	-	-

3: 0	TI1SEL	RW	4'h0	TI1 input selection (tim_TI1 [15: 0] input) 0000: TIMx_CH1 0001:tim _ti1_in1 0010:tim _ti1_in2 0011:tim _ti1_in3 Other: Reserved

23.3.18 TIM15 Spare Function Option Register 1 (TIMx_AF1)

Address offset: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	Res				BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
		RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13	BKCMP4P	RW	0	Tim_brk_cmp4 input polarity This bit select that input polarity of tim_brk_cmp4 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp4 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp4 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
12	BKCMP3P	RW	0	Tim_brk_cmp3 input polarity This bit select that input polarity of tim_brk_cmp3 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp3 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp3 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
11	BKCMP2P	RW	0	Tim_brk_cmp2 input polarity This bit selects that brk_cmp2 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp2 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp2 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
10	BKCMP1P	RW	0	Tim_brk_cmp1 input polarity This bit selects that brk_cmp1 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp1 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp1 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
9	BKINP	RW	0	TIMx_BKIN input polarity This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
8: 5	Reserved	-	-	-

4	BKCMP4E	RW	0	Tim_brk_cmp4 input enable This bit is used to enable tim_brk_cmp4 on brake input. The tim_brk_cmp4 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp4 input off 1: tim_brk_cmp4 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
3	BKCMP3E	RW	0	Tim_brk_cmp3 input enable This bit is used to enable tim_brk_cmp3 on brake input. The tim_brk_cmp3 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp3 input off 1: tim_brk_cmp3 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
2	BKCMP2E	RW	0	Tim_brk_cmp2 input enable This bit is used to enable tim_brk_cmp2 on brake input. The tim_brk_cmp2 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp2 input off 1: tim_brk_cmp2 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
1	BKCMP1E	RW	0	Tim_brk_cmp1 input enable This bit is used to enable tim_brk_cmp1 on brake input. The tim_brk_cmp1 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp1 input off 1: tim_brk_cmp1 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
0	BKINE	RW	1	TIMx_BKIN input enable This bit is used to enable the standby function of BKIN when the brake is input. The BKIN input is OR logic with other brake sources as a brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.

23.3.19 TIM15 Alternate Function Option Register 2 (TIMx_AF2)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res													OCRSEL [2: 0]		
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															

Bit	Name	R/W	Reset Value	Function
31: 19	Reserved	-	-	-
18: 16	OCRSEL	RW	3'h0	OCREF Reset source selection (OCREF_clr source selection) This bit segment is used to select that ocref_clr input source 0000: ocref_clr0 0001: ocref_clr1 0010 ocref_clr2 0011 ocref_clr3 Others: Reserved Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
15: 0	Reserved	-	-	-

23.3.20 TIM15 DMA Control Register (TIMx_DCR)

Address offset: 0x3DC

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DBL [4: 0]					Res			DBA [4: 0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	-
12: 8	DBL	RW	5'h0	DMA burst length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register, the timer makes a continuous transfer), i.e. defines the number of transfers, which can be half words (double bytes) or bytes: 00000: 1 transfer 00001: 2 transfers 00010: 3 transmissions..... ..... 11001: 26 transfers Example: We consider such a transmission: DBL = 7 bytes, DBA = TIMx_CR1 If DBL = 7 bytes and DBA = TIMx_CR1 represents the address of the data to be transmitted, then the transmitted address is given by: (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL Where (address of TIMx_CR1) + DBA plus 7 gives the address where data will be written or read out, so that the transfer of data will occur in 7 registers starting from address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following can occur: -If the data is set to a half word (16 bits), then the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, for the timer, the user must specify the data width to be transmitted by the DMA.
7: 5	Reserved	-	-	-
4: 0	DBA	RW	5'h0	DMA base address These bits define the base address of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register), and the DBA is defined as the offset from the address where the TIMx_CR1 register is located: 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ……..

23.3.21 DMA address for TIM15 continuous mode (TIMx_DMAR)

Address offset: 0x3E0

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DMAB	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA index) x4 where:

				The "TIMx_CR1 address" is an address where the control register 1 (TIMx_CR1) is located; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

23.4 TIM16_TIM17 registers

23.4.1 TIM16_TIM17 Control Register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DITHEN	UIFREMAP	Res	CKD [1: 0]		ARPE	Res			OPM	URS	UDIS	CEN
			RW	RW		RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 13	Reserved	-	-	-
12	DITHEN	RW	0	Jitter enable (Dithering enbale) 0: Jitter off 1: Jitter on Note: This bit can only be modified when the CEN bit is reset
11	UIFREMAP	RW	0	UIF status bit remapping enable 0: No remapping, UIF status bit is not copied to bit 31 of TIMx_CNT register 1: Remap, UIF status bit copied to bit 31 of TIMx_CNT register
10	Reserved	-	-	-
9: 8	CKD	RW	2'h0	Clock division factor These 2 bits define the frequency division ratio between the timer clock (CK_INT) frequency and dead time and the sampling clock (tDTS) used by the dead generator and the digital filter (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 4	Reserved	-	-	-
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter

				and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

23.4.2 TIM16_TIM17 Control Register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						OIS1N	OIS1	Res				CCDS	CCUS	Res	CCPC
						RW	RW					RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	OIS1N	RW	0	Output Idle state 1 0: OC1N = 0 after dead time when MOE = 0; 1: OC1N = 1 after dead zone when MOE = 0. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
8	OIS1	RW	0	Output Idle state 1 0: when MOE = 0, OC1 = 0 after dead time if OC1N is achieved; 1: When MOE = 0, OC1 = 1 after dead time if OC1N is implemented. Note: This bit cannot be modified after the LOCK (TIMx_BKR register) level 1, 2, or 3 has been set.
7: 4	Reserved	-	-	-
3	CCDS	RW	0	Capture/compare DMA selection 0: When a CCx event occurs, send a DMA request for CCx; 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/Compare Control Update Selection (Capture/compare control update selection) 0: If the capture/compare control bits are preloaded (CCPC = 1), they can only be updated by setting the COM bit; 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	-
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE, and OCxM bits are not preloaded; 1: CCxE, CCxNE, and OCxM bits are preloaded; When this bit is set, they are only updated after the COM bit is set (COMG bit setting or rising edge detected on tim_trgi, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

23.4.3 TIM16_TIM17 DMA/interrupt enable register (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CC1DE	UDE	BIE	Res	COMIE	Res			CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	CC1DE	RW	0	Capture/Compare 1 DMA request enable 0: DMA request to capture/compare 1 is prohibited;

				1: DMA request allowed to capture/compare 1. 。
8	UDE	RW	0	Update DMA request enable 0: DMA request to prohibit update; 1: Allow updated DMA requests.
7	BIE	RW	0	Allow brake interruption (Break interrupt enable) 0: Brake interruption is prohibited; 1: Allow the brake to be interrupted.
6	Reserved	-	-	-
5	COMIE	RW	0	COM interrupt enable 0: Disable COM interrupt; 1: COM interrupt enabled
4: 2	Reserved	-	-	-
1	CC1IE	RW	0	Allow capture/compare 1 interrupt (Capture/Compare 1 interrupt enable) 0: Disable capture/compare 1 interrupt; 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

23.4.4 TIM16_TIM17 status register (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CC1OF	Res	BIF	Res	COMIF	Res			CC1IF	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 10	Re-served	-	-	-
9	CC1OF	RC_W0	0	Capture/Compare 1 Repeat Capture Marker (Capture/Compare 1 overcapture flag) This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been generated. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	Re-served	-	-	-
7	BIF	RC_W0	0	Brake interruption mark (Break interrupt flag) Once the brake input is active, the position '1' is applied by the hardware. If the brake input is invalid, the bit may be cleared '0' by the software. 0: No break event occurred. 1: An active level has been detected on the break input. Generate an interrupt if BIE = 1 for TIM_DIER
6	Re-served	-	-	-
5	COMIF	RC_W0	0	COM interrupt flag Once a COM event is generated (when the capture/compare control bits: CCxE, CCxNE, OCxM have been updated) this bit is set '1' by the hardware. It is cleared by software. 0: No update occurred. 1: COM interrupt waiting for response.
4: 2	Re-served	-	-	-
1	CC1IF	RC_W0	0	Capture/Compare 1 Interrupt Flag (Capture/Compare 1 interrupt flag) If channel CC1 is configured as output: 0: No match; 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the content of TIMx_CCR1 is larger than the content of TIMx_APR, the CC1IF bit becomes highat the timeof overflow If channel TI1 is configured as input:

				This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to TIMx_CCR1 (an edge of the same polarity as the selected polarity is detected on IC1).
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: If the UDIS of the TIMx_CR1 register = 0, an update event is generated when the repetition counter = 0). − If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

23.4.5 TIM16_TIM17 event generation register (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BG	Res	COMG	Res			CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	BG	W	0	Break generation This bit is set '1' by software to generate a braking event, and '0' is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	Reserved	-	-	-
5	COMG	W	0	Capture/compare events, generate control updates Capture/Compare control update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit is only valid for channels with complementary outputs.
4: 2	Reserved	-	-	-
1	CC1G	W	0	Capture/Compare 1 generation This bit is set '1' by software to generate a capture/compare event, and '0' is automatically cleared by hardware. 0: No action 1: produce a capture/compare event on channel 1: If channel 1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel 1 is configured as input: The current counter value is captured to the TIMx_CCR1 register; Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged). The counter is cleared '0' if in centrosymmetric

				mode or DIR = 0 (count up); If DIR = 1 (count down), the counter takes the value of TIMx_ARR.

23.4.6 TIM16_TIM17 Capture/Compare Mode Control Register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value: 0x0000 000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															OC1M[3]
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								OC1CE	OC1M [2: 0]			OC1PE	OC1FE	CC1S [1: 0]	
								ICIF [3: 0]				ICIPSC [1: 0]			
								RW	RW	RW	RW	RW	RW	RW	RW

23.4.6.1 Output compare mode

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	OC1M[3]	RW	0	See OC1M [2: 0] description for details
15: 8	Reserved	-	-	-
7	OC1CE	RW	0	Output Compare 1 clear '0' enable Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6: 4	OC1M	RW	3'h0	Output Comparison 1 Mode Output Compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. When the value of counter TIMx_CNT is the same as that of capture/compare register 1 (TIMx_CCR1), OC1REF is forced to be high. 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110: PWM mode 1-on up count, channel 1 is active once TIMx_CNT < TIMx_CCR1, otherwise it is invalid; On counting down, channel 1 is an inactive level (OC1REF = 0) once TIMx_CNT > TIMx_CCR1, otherwise it is an active level (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 else inactive. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Comparison 1 Preload Enable Output Compare 1 preload enable 0: The preload function of the TIMx_CCR1 register is disabled, the TIMx_CCR1 register can be written at any time, and the newly written value takes effect immediately. 1: Turn on the preload function of the TIMx_CCR1 register. Read and write operations only operate on the preload register. The preload value of TIMx_CCR1 is loaded into the current register when the update event arrives. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: Only in monopulse mode (OPM = 1 of TIMx_CR1 register), PWM mode can be used without confirming the preload register, otherwise its operation is uncertain.
2	OC1FE	RW	0	Output Compare 1 Fast Enable (Output Compare 1 fast enable)

				This bit is used to speed up the output response to the triggering input event. Must be used in single pulse mode (OPM position 1 of the TIMx_CR1 register) to achieve fast pulse output when the trigger comes. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. Only available for PWM Mode 1 and PWM Mode 2
1: 0	CC1S	RW	2'h0	Capture/Compare 1 selection Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. Other: Reserved Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

23.4.6.2 Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 4	IC1F	RW	4'h0	Input Capture 1 Filter (Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter consists of an event counter, which records N events and produces an output jump: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
3: 2	IC1PSC	RW	2'h0	Input/Capture 1 Prescaler (Input capture 1 prescaler These 2 bits define the prescalation coefficient of the CC1 input (IC1). Once CC1S = 00, the prescaler is reset. 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	RW	2'h0	Capture/Compare 1 Selection These 2 bits define the direction of the channel (input/output), and the selection of the input pin: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode only operates when the internal flip-flop input is selected (selected by the TS bit of the TIMx_SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx_CCER register).

23.4.7 TIM16_TIM17 Capture/Compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3	CC1NP	RW	0	Input/capture 1 complementary output polarity (Capture/Compare 1 implementary output polarity) 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configuration as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/capture 1 complementary output enable (Capture/Compare 1 implementary output enable) 0: OFF-OC1N disables output, so the level of OC1N depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits. 1: ON-The OC1N signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 Output Polarity (Capture/Compare 1 output polarity) CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: Non-inverted/rising edge: The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Inverted/falling edge: The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: non-inverted/double edge The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	RW	0	Input/Capture 1 Output Enable (Capture/Compare 1 output enable) CC1 channel configured as output: 0: OFF-OC1 disables output, so the output level of OC1 depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: ON-The OC1 signal is output to the corresponding output pin, and its output level depends on the values of the MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Notes: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 23-5 Control bits for complementary output channels OCx and OCxN with brake function

Control bits					Output states	
MOE bit	**OSSI bit**	**OSSR bit**	**CCxE bit**	**CCxNE bit**	**OCx output status**	**OCxN output state**
1	x	0	0	0	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	0	1	Output disabled (not driven by the timer anymore). OCx = 0, OCx_EN = 0	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Output disabled (not driven by the timer anymore). OCxN = 0, OCxN_EN = 0
		0	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
		1	0	0	Output disabled (not driven by the timer anymore). OCx = CCxP, OCx = 0 OCx_EN = 0	Output disabled (not driven by the timer anymore). OCxN = CCxNP, OCx = 0 OCxN_EN = 0
		1	0	1	Off-State (output enabled with in-active state) OCx = CCxP, OCx_EN = 1	OCxREF + polarity, OCxN = OCREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + polarity, OCx = OCREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + polarity + dead zone, OCxN_EN = 1
0	0	x	x	x	Output disabled (not driven by the timer anymore).	
	1		0	0		
	1		0	1	Off state (output enabled but invalid level) Asynchronous: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCx_EN = 1; if brake or brake 2 is triggered Then if the clock exists (only valid when brake 1 is triggered, brake 2 is not needed): OCx = OISx, OCxN = OISxN after a dead time, assuming that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN Note: Brake 2 is only for OSSI = OSSR = 1	
	1		1	0		
	1		1	1		

In particular, when the dead band is active, the output is always output according to the rules of dead band output, although moe may have been set to 1 by software or hardware at this time.

If neither of the two outputs of a channel is used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP, and CCxNP must all be cleared.

23.4.8 TIM16_TIM17 Counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UIFCPY	Res														
R															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	UIFCPY	R	0	This bit is a read-only register, and the UIF bit of the TIMx_ISR register is copied. If the UIF remap bit in TIMx_CR1 is reset, the bit is reserved and read out as 0
30: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

23.4.9 TIM16_TIM17 Prescaler (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Re-served	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes the counter being cleared '0' by the UG bit of TIM_EGR or cleared '0' by the slave controller operating in reset mode.

23.4.10 TIM16_TIM17 Automatic Reload Register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												ARR [19:16]			
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Re-served	-	-	-
19: 0	ARR	RW	20'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null. No jitter mode (DITHEN = 0): This register holds that auto-load value Dither Mode (DITHEN = 1): The register holds an integer portion ARR [19: 4], and ARR [3: 0] contains a dither portion

23.4.11 TIM16_TIM17 Repeat Count Register (TIMx_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								REP							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Re-served	-	-	-
7: 0	REP	RW	8'h0	Value of repeat counter Repetition counter value When the preload function is turned on, these bits allow the user to set the update rate of the comparison register (i.e., periodically transfer from the preload register to the current register); If an update interrupt is allowed, the rate at which the update interrupt is generated will also be affected. Each time the down counter reaches 0, an update event is generated and the repeat counter resumes counting from the REP value. The new value written to the TIMx_RCR register only takes effect when the next update event occurs.

23.4.12 TIM16_TIM17 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CCR1 [19:16]			
												RW/RO			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1 [15: 0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31: 20	Re-served	-	-	-
19: 0	CCR1	RW/RO	20'h0	Capture/Compare 1 value for channel 1 If channel CC1 is configured as output: CCR1 contains the value loaded into the current capture/compare 1 register (preload value). If the preload function is not selected in the TIMx_CCMR1 register (OC1PE bit), the written value is immediately transferred to the current register. Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. No jitter mode (DITHEN = 0): This register holds the comparison value of CCR1 [15: 0] and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): The register holds an integer portion CCR1 [19: 4], and CCR1 [3: 0] contains a dither portion If channel CC1 is configured as input: CCR1 contains the counter value transmitted by the last input capture 1 event (IC1). TIMx_CCR1 Register Read Only No jitter mode (DITHEN = 0): This register holds the capture value to CCR1 [15: 0], and the CCR1 [19: 16] bit is reset Dither Mode (DITHEN = 1): This register holds the capture value to CCR1 [19: 4], and the CCR1 [3: 0] bit is reset

23.4.13 TIM16_TIM17 Dead time register 2 (TIMx_DTR2)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														DTPE	DTAE
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								DTGF [7: 0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	DTPE	RW	0	Deadtime preload enable 0: The value of dead time cannot be preloaded 1: The value of the dead time can be preloaded Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
16	DTAE	RW	0	Deadtime asymmetric enable 0: The rising and falling edges have the same dead time, both configured by DTG [7: 0] 1: Rising edge dead time is configured by DTG [7: 0] and falling edge dead time is configured by DTGF [7: 0] Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, 3, this bit cannot be modified.
15: 8	Reserved	-	-	-
7: 0	DTGF [7: 0]	RW	8'h0	Falling edge dead time generator Dead-time falling edge generator setup This bit segment defines the dead time of the falling edge when the complementary output is inserted, assuming DT denotes its duration: DTGF [7: 5] = 0xx = > DTF = DTGF [7: 0] × Tdtg, Tdtg = TDTS; DTGF [7: 5] = 10x = > DTF = (64 + DTGF [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTGF [7: 5] = 110 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTGF [7: 5] = 111 = > DTF = (32 + DTGF [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified

23.4.14 TIM16_TIM17 Brake and Dead Time Register (TIMx_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			BKBID	Res	BKDSRM	Res						BKF [3: 0]			
			RW		RW							RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1: 0]		DTG [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28	BKBID	RW	0	Break bidirectional 0: Brake input BRK in input mode 1: Brake input BRK in bi-directional mode In the bi-directional mode (BKBID is set to 1), the brake input is configured as both an input mode and an open-drain output mode. Any valid braking event will pull the brake input level down, indicating to external devices that a braking event has occurred internally Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified. Note: Any write operation to this bit will delay the APB clock from taking effect.
27	Reserved	-	-	-
26	BKDSRM	RW	0	Brake release (Break disarm) 0: Brake input BRK activated 1: Brake input BRK released

				This bit is cleared by hardware when there is no brake source The BKDSRM bit must be set by software to release the bidirectional output control (open drain output high impedance state) and then poll it until it is reset by hardware, indicating that the fault state has disappeared. Note: Any write operation to this bit will delay the APB clock from taking effect.
25: 20	Reserved	-	-	-
19: 16	BKF [3: 0]	RW	4'h0	Brake filter (Break filter) These bits define the frequency at which the BRK signal is sampled and the bandwidth at which the BRK is digital filtered. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK2 is asynchronous 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8 Note: Once LOCK level 1 (LOCK bit in the TIMx_BDTR register) is set, this bit cannot be modified.
15	MOE	RW	0	Main output enabled (Main output enable) Once the brake input is valid, the bit is asynchronously cleared '0' by the hardware. Depending on the setting value of the AOE bit, this bit can be cleared '0' by software or automatically set to 1. It is only valid for channels configured as output. 0: OC and OCN outputs are disabled or forced to idle state. 1: Turn on the OC and OCN outputs if the corresponding enable bits (CCxE, CCxNE bits of the TIMx_CCER register) are set. See TIMx Capture/Compare Enable Register (TIMx_CCER) for details of OC/OCN enabling.
14	AOE	RW	0	Automatic output enable (Automatic output enable) 0: MOE can only be set '1' by software; 1: The MOE can be set '1' by software or automatically set '1' at the next update event (if the brake input is invalid). Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified.
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to '1', this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
12	BKE	RW	0	Break enable 0: Disable brake input (BRK and CCS clock failure events); 1: Turn on the brake input (BRK and CCS clock failure events). Note: When LOCK level 1 is set (the LOCK bit in the TIMx_BDTR register), this bit cannot be modified. Note: Any write operation to this bit requires an APB clock delay before it can work.
11	OSSR	RW	0	"Off state" selection in running mode (Off-state selection for Run mode) This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0);

				1: When the timer is not working, once CCxE = 1 or CCxNE = 1, first turn on OC/OCN and output an invalid level, and then set OC/OCN enable output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
10	OSSI	RW	0	"Off state" selection in idle mode (Off-state selection for Idle mode) This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enable (TIMx Capture/Compare Enable Register (TIMx_CCER)). 0: OC/OCN output is disabled when the timer is not working (OC/OCN enable output signal = 0); 1: When the timer is not working, once CCxE = 1 or CCxNE = 1, the OC/OCN first outputs its idle level and then the OC/OCN enables the output signal = 1. Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 2, this bit cannot be modified.
9: 8	LOCK	RW	2'h0	Lock settings (Lock configuration) These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot be written to DTG, BKBID, BK2BID, BKE, BKP, AOE bits of TIMx_BDTR register and OISx/OISxN bits of TIMx_CR2 register; 10: Lock level 2, cannot write bits in lock level 1, nor can you write CC polarity bits (once the relevant channel is set to output by CCxS bits, the CC polarity bits are CCxP/CCNxP bits of the TIMx_CCER register) and OSSR/OSSI bits; 11: Lock level 3, cannot write bits in lock level 2, and cannot write CC control bits (CC control bits are OCxM/OCxPE bits of TIMx_CCMRx register once the relevant channel is set as output through CCxS bits); Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.
7: 0	DTG	RW	8'h0	Dead Generator Settings (Dead-time generator setup) This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG [7: 5] = 0xx = > DT = DTG [7: 0] × Tdtg, Tdtg = TDTS; DTG [7: 5] = 10x = > DT = (64 + DTG [5: 0]) × Tdtg, Tdtg = 2 × TDTS; DTG [7: 5] = 110 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 8 × TDTS; DTG [7: 5] = 111 = > DT = (32 + DTG [4: 0]) × Tdtg, Tdtg = 16 × TDTS; Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16 us to 31750 ns by 250 ns steps; 32 us to 63us by 1 us steps; 64 us to 126 us by 2 us steps Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, 2, or 3, these bits cannot be modified.

23.4.15 TIM16_TIM17 Input select register (TIMx_TISEL)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												TI1SEL [3: 0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	TI1SEL	RW	4'h0	TI1 input selection (tim_TI1 [15: 0] input) 0000: TIMx_CH1 0001: Reserved 0010:tim _ti1_in2 0011:tim _ti1_in3 0100:tim _ti1_in4 0101:tim _ti1_in5 Other: Reserved

23.4.16 TIM16_TIM17 Alternate Function Option Register 1 (TIMx_AF1)

Address offset: 0x60

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	Res				BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
		RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13	BKCMP4P	RW	0	Tim_brk_cmp4 input polarity This bit select that input polarity of tim_brk_cmp4 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp4 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp4 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
12	BKCMP3P	RW	0	Tim_brk_cmp3 input polarity This bit select that input polarity of tim_brk_cmp3 and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp3 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp3 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
11	BKCMP2P	RW	0	Tim_brk_cmp2 input polarity This bit selects that brk_cmp2 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp2 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp2 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
10	BKCMP1P	RW	0	Tim_brk_cmp1 input polarity This bit selects that brk_cmp1 input polarity and must be configured at the same time as the BKP polarity bit 0: tim_brk_cmp1 input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: tim_brk_cmp1 input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
9	BKINP	RW	0	TIMx_BKIN input polarity This bit selects the alternate function of the BKIN input polarity and must be configured at the same time as the BKP polarity bit 0: BKIN input polarity does not flip (low active when BKP = 0, high active when BKP = 1) 1: BKIN input polarity flip (high active when BKP = 0, low active when BKP = 1) Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
8: 5	Reserved	-	-	-
4	BKCMP4E	RW	0	Tim_brk_cmp4 input enable This bit is used to enable tim_brk_cmp4 on brake input. The tim_brk_cmp4 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp4 input off

				1: tim_brk_cmp4 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
3	BKCMP3E	RW	0	Tim_brk_cmp3 input enable This bit is used to enable tim_brk_cmp3 on brake input. The tim_brk_cmp3 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp3 input off 1: tim_brk_cmp3 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
2	BKCMP2E	RW	0	Tim_brk_cmp2 input enable This bit is used to enable tim_brk_cmp2 on brake input. The tim_brk_cmp2 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp2 input off 1: tim_brk_cmp2 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
1	BKCMP1E	RW	0	Tim_brk_cmp1 input enable This bit is used to enable tim_brk_cmp1 on brake input. The tim_brk_cmp1 input is OR logic with other brake sources as a brake input. 0: tim_brk_cmp1 input off 1: tim_brk_cmp1 input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
0	BKINE	RW	1	TIMx_BKIN input enable This bit is used to enable the standby function of BKIN when the brake is input. The BKIN input is OR logic with other brake sources as a brake input. 0: BKIN input off 1: BKIN input on Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified.

23.4.17 TIM16_TIM17 Alternate Function Option Register 2 (TIMx_AF2)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res													OCRSEL [2: 0]		
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															

Bit	Name	R/W	Reset Value	Function
31: 19	Reserved	-	-	-
18: 16	OCRSEL	RW	3'h0	OCREF Reset source selection (OCREF_clr source selection) This bit segment is used to select that ocref_clr input source 0000: ocref_clr0 0001: ocref_clr1 0010 ocref_clr2 0011 ocref_clr3 Others: Reserved Note: Once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 1, this bit cannot be modified
15: 0	Re-served	-	-	-

23.4.18 TIM16_TIM17option register (TIMx_OR1)

Address offset: 0x68

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															HSE32EN
															RW

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	HSE32EN	RW	0	HSE32crossover connection tim_ti1_in3enabled 0: Disabled 1: Enabled

23.4.19 TIM16_TIM17 DMA Control Register (TIMx_DCR)

Address offset: 0x3DC

Reset value: 0x0000 000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DBL [4: 0]					Res			DBA [4: 0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12: 8	DBL	RW	5'h0	DMA burst length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register, the timer makes a continuous transfer), i.e. defines the number of transfers, which can be half words (double bytes) or bytes: 00000: 1 transfer 00001: 2 transfers 00010: 3 transmissions..... ..... 11010: 27 transfers Example: We consider such a transmission: DBL = 7 bytes, DBA = TIMx_CR1 If DBL = 7 bytes and DBA = TIMx_CR1 represents the address of the data to be transmitted, then the transmitted address is given by: (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL Where (address of TIMx_CR1) + DBA plus 7 gives the address where data will be written or read out, so that the transfer of data will occur in 7 registers starting from address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following can occur: -If the data is set to a half word (16 bits), then the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, for the timer, the user must specify the data width to be transmitted by the DMA.
7: 5	Reserved	-	-	-
4: 0	DBA	RW	5'h0	DMA base address These bits define the base address of the DMA in continuous mode (when reading or writing to the TIMx_DMAR register), and the DBA is defined as the offset from the address where the TIMx_CR1 register is located: 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR …….

23.4.20 TIM16_TIM17 DMA address for continuous mode (TIMx_DMAR)

Address offset: 0x3E0

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DMAB	RW	0	DMA register for burst accesses A read or write operation to the DMAR register accesses the register located at the address: TIMx_CR1 address + (DBA + DMA index) x4 where: The "TIMx_CR1 address" is an address where the control register 1 (TIMx_CR1) is located; DBA is the DMA baseaddress configured in TIMx_DCR register; DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

24. Basic Timer (TIM6 and TIM7)

24.1 Introduction

The base timers TIM6 and TIM7 each contain a 16-bit autoload counter driven by their respective programmable prescalers.

They can provide a time reference as a general purpose timer, and in particular can provide a clock for a digital-to-analog converter (DAC). In fact, the timers are internally connected to the DAC and are able to drive it through their trigger outputs.

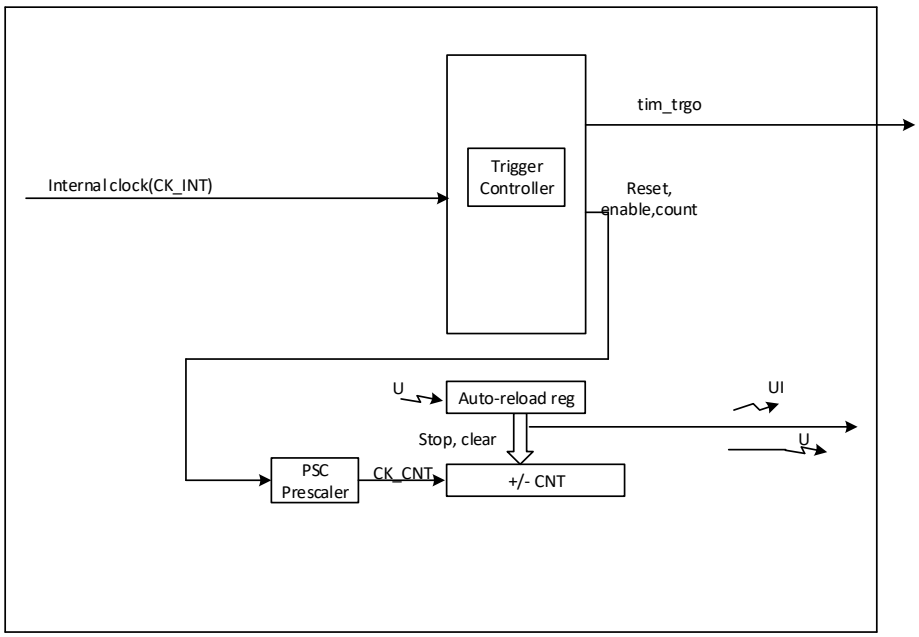
TIM6 and TIM7 are completely independent, they do not share any resources.

24.1.1 TIM6 and TIM7 Main Features

The main functions of the TIM6 and TIM7 timers include:

- 16-bit auto-load counter
- 16-bit programmable (can be modified in real time) prescaler, the frequency division coefficient of the counter clock frequency is any value between 1 and 65535
- DAC synchronous circuit is triggered.
- Generate interrupt/DMA when an update event (counter overflow) occurs

24.1.2 CAN block diagram



24.2 TIMx functional description

24.2.1 Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. This counter can count up. The clock of the counter may be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time base unit comprises:

- Counter Register (TIMx_CNT)
- Prescaler Register (TIMx_PSC)
- Automoad Register (TIMx_ARR)

An autoload register is preloaded, and a write or read autoload register will access its preload register. The contents of the preload register are transferred to the shadow register either immediately or at each update event (UEV), depending on the setting of the Automoad preload enable bit (ARPE) in the TIMx_CR1 register. An update event occurs when the counter reaches an overflow condition and when the UDIS bit in the TIMx_CR1 register is equal to 0. The update event may also be generated by software and other conditions. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK_CNT divided by the prescaler, which is valid for the counter only when the counter enable bit (CEN) in the TIMx_CR1 register is set. (See the description of the slave mode controller for more details on enabling counters).

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the clock frequency of the counter by any value between 1 and 65536. It is a 16-bit counter controlled based on a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescalation parameters will be adopted when the next update event comes.

The following figures give examples of changing counter parameters when the prescaler is running.

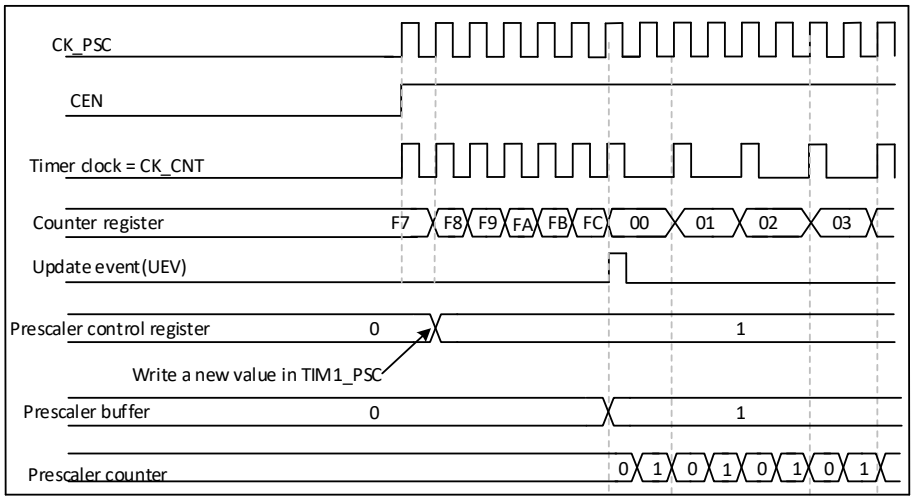


Figure 24-1 Timing diagram of the counter when the parameters of the prescaler change from 1 to 2

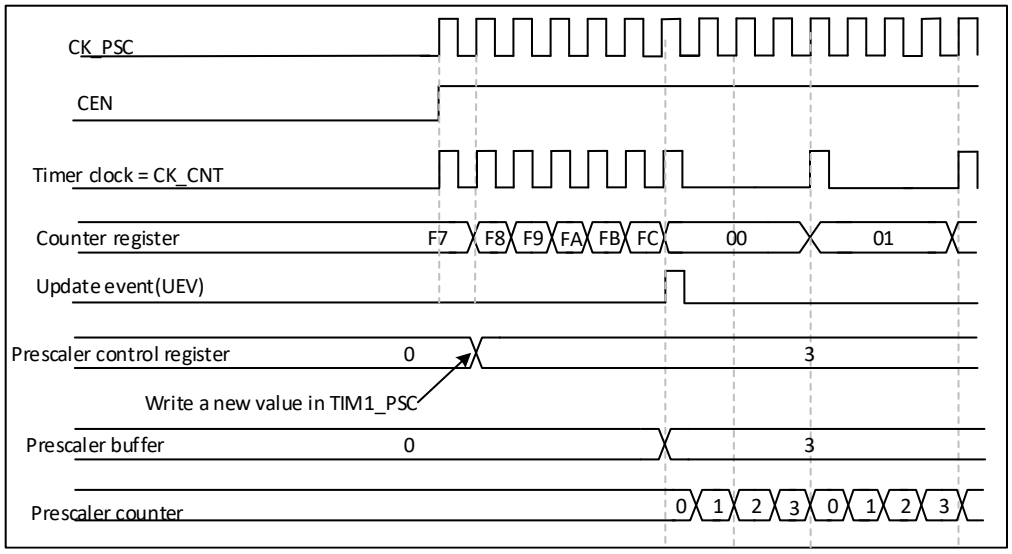


Figure 24-2 Timing diagram of the counter when the parameter of the prescaler is changed from 1 to 4

24.2.2 Counter mode

Upcounting mode

In the count-up mode, the counter counts from 0 to the auto-load value (the content of TIMx_ARR), then counts from 0 again and generates a count overflow event.

Setting the UG bit in the TIMx_EGR register (either by software or using a slave mode controller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. No update event will be generated until the UDIS bit is cleared '0'. Even so, when an update event should occur, the counter is still cleared '0', and the counter inside the prescaler is also cleared '0' (but the value of the prescaler remains unchanged).

In addition, if the URS bit in the TIMx_CR1 register is set (select the update request source), an update event UEV can be generated by setting the UG bit, but the UIF flag bit will not be set (i.e., no interrupt request will be generated). This is to avoid generating both update andcapture interrupts when clearing the counter on the capture event.

When an update event occurs, all of the following registers are updated, and the hardware sets an update flag bit (UIF bit in the TIMx_SR register) at the same time (according to the URS bit):

- The auto-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

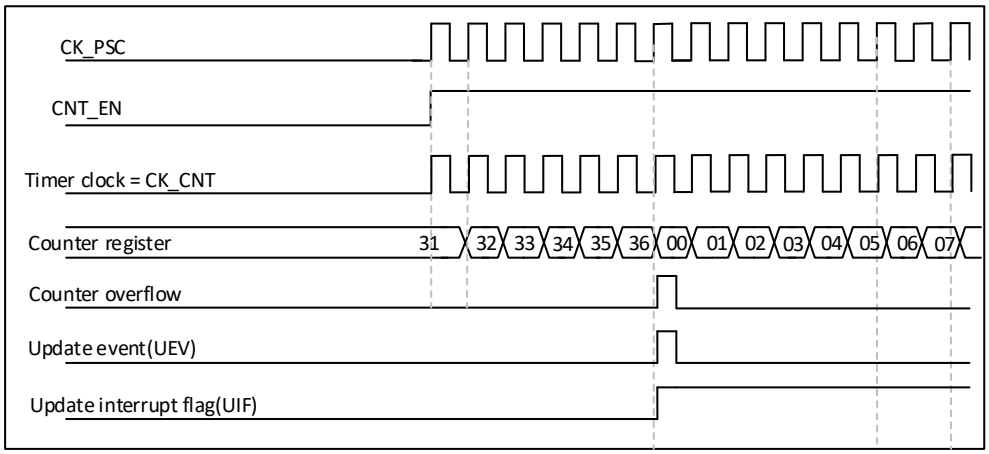


Figure 24-3 counter timing diagram with internal clock division factor of 1

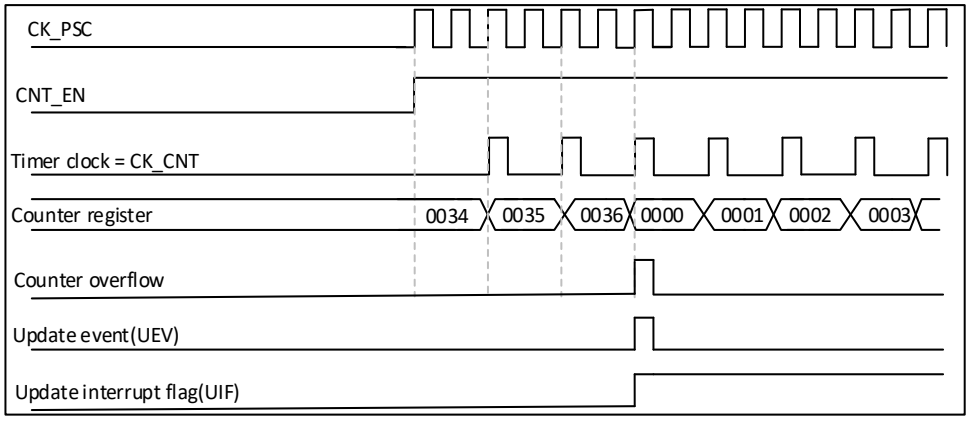


Figure 24-4 counter timing diagram with internal clock division factor of 2

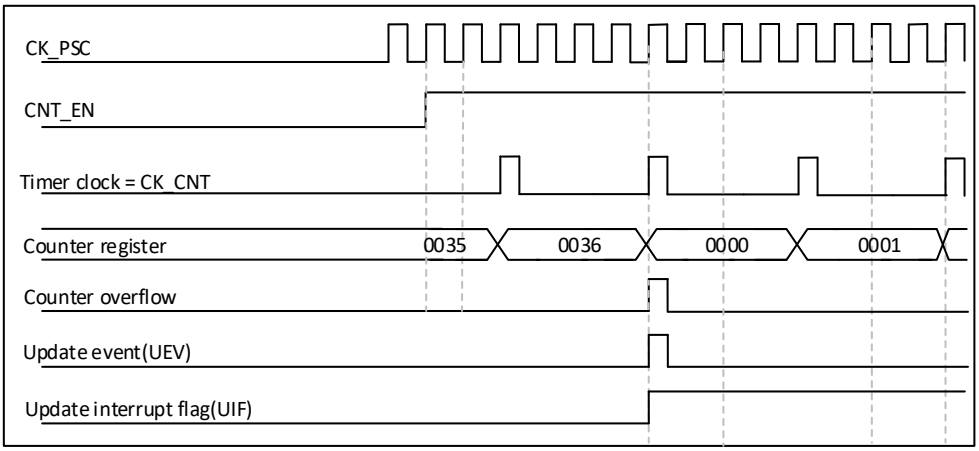


Figure 24-5 Counter timing diagram, internal clock divided by 4

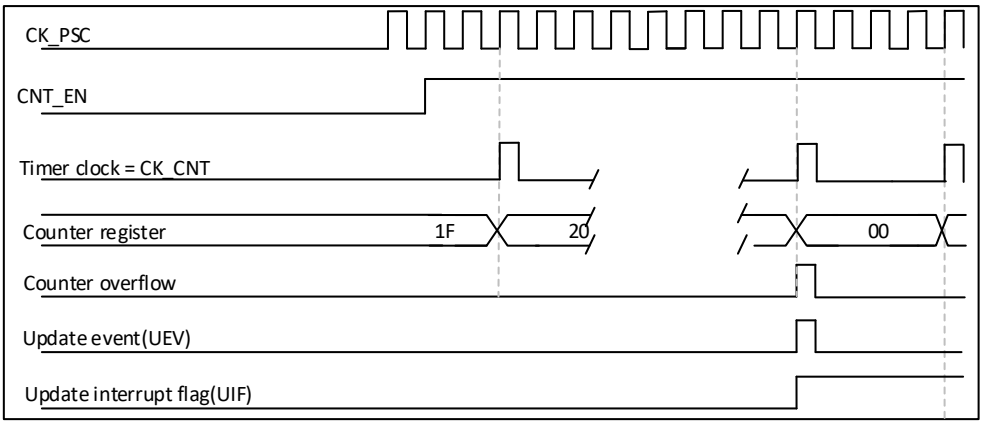


Figure 24-6 Counter timing diagram, internal clock divided by N

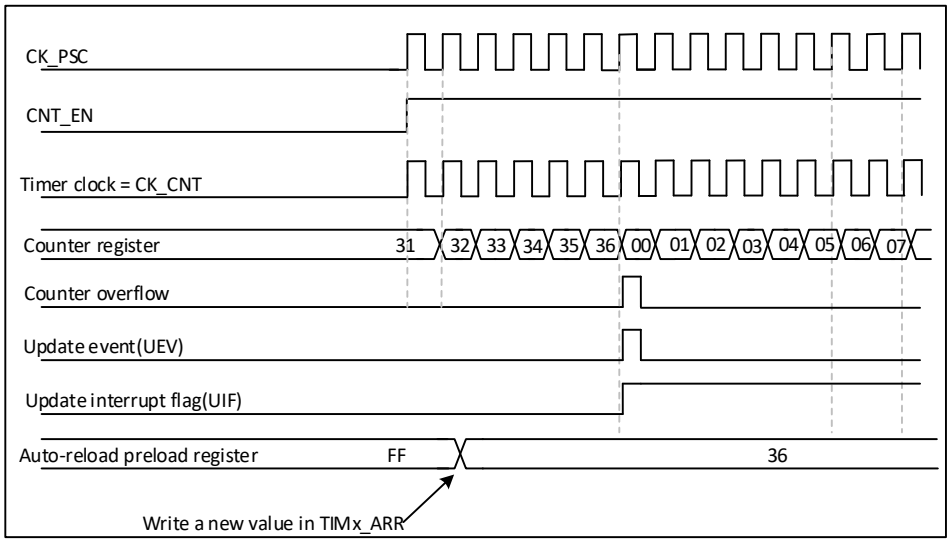


Figure 24-7 Counter timing diagram, update event when ARPE=0 (no TIMx_ARR preloaded)

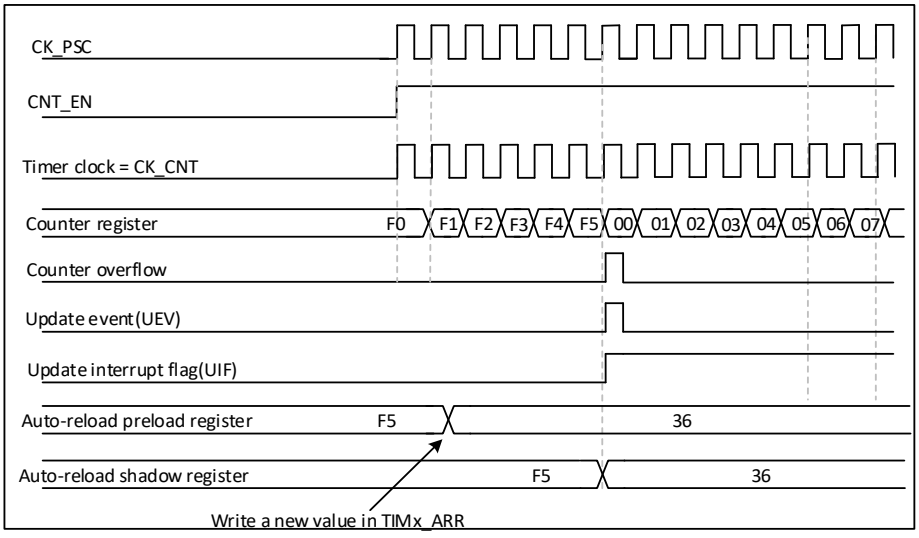


Figure 24-8 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

24.2.3 Debug mode

When the microcontroller enters the debug mode (Cortex-M4F core stops), the TIMx counter may either continue normal operation or stop according to the setting of DBG_TIMx_STOP in the DBG module. See subsequent DBG sections for details.

24.3 TIMx registers

24.3.1 TIM6 and TIM7 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								ARPE	Res			OPM	URS	UDIS	CEN
								RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered; 1: TIMx_ARR register is loaded into buffer.
6: 4	Reserved	-	-	-
3	OPM	RW	0	Single pulse mode (One pulse mode) 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source Update request source This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is enabled, either of the following events generates an update interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: If an update interrupt or DMA request is enabled, only a counter overflow/underflow generates an update interrupt or DMA request.
1	UDIS	RW	0	Prohibit updates Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller Buffered registers are then loaded with their preload values. (Translation: Update shadow register) 1: UEV disabled. No update events are generated, and the shadow registers (ARR, PSC, CCRx) hold their values. However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	RW	0	Counter enable 0: Disable counter; 1: Enable counter. Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

24.3.2 TIM6 and TIM7 control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									MMS [2: 0]			Res			
									RW						

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6: 4	MMS	RW	0	Main mode selection (Master mode selection) These 3 bits are used to select synchronization information (TRGO) to be sent to the slave timer in master mode. The combination is as follows: 000: Reset-The UG bit of the TIMx_EGR register is used as a trigger output (TRGO). If the reset is generated by the trigger input (the slave mode controller is in reset mode), the signal on the TRGO will have a delay relative to the actual reset. 001: Enable-Counter enable signal CNT_EN is used as a trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timeris enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. Note: The clock of the slave timer or ADC must be enabled prior to receive events from the master timer, and must not be changed.
3: 0	Reserved	-	-	-

24.3.3 TIM6and TIM7DMA/Interrupt Enable Registers (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							UDE	Res							UIE
							RW								RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	UDE	RW	0	Update DMA request enable 0: DMA request to prohibit update; 1: Allow updated DMA requests.
7: 1	Reserved	-	-	-
0	UIE	RW	0	Allow update interrupts Update interrupt enable 0: Prohibit update interrupt; 1: Allow update interrupt.

24.3.4 TIM6and TIM7status registers (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31: 1	Re-served	-	-	-
0	UIF	RC_W0	0	Update interrupt flag Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set '1' by the hardware when the register is updated: − If the UDIS of the TIMx_CR1 register = 0, when the repetition counter value overflows or underflows (an update event occurs when the repetition counter = 0).

				− If URS = 0 and UDIS = 0 of the TIMx_CR1 register, an update event is generated when UG = 1 of the TIMx_EGR register is set, and the counter CNT is re-initialized by software. − If UDIS = 0 and URS = 0 of the TIMx_CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Refer to Slave Mode Control Register (TIMx_SMCR)).

24.3.5 TIM6 and TIM7 Event Generation Register (TIMx_EGR)

Address offset: 0x14

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															UG
															W

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	UG	W	0	Update generation This bit is set '1' by the software and automatically cleared '0' by the hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the counter of the prescaler is also cleared '0' (but the prescaler coefficient remains unchanged).

24.3.6 TIM6 and TIM7 counters (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT	RW	16'h0	Counter value

24.3.7 TIM6 and TIM7 prescalers (TIMx_PSC)

Address offset: 0x28

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PSC	RW	16'h0	Prescaler value The clock frequency (CK_CNT) of the counter is equal to fCK_PSC/(PSC [15: 0] +1). The PSC contains the value loaded into the current prescaler register each time an update event occurs; The update event includes a counter being The UG bit of TIM_EGR is cleared '0' or is cleared '0' by a slave controller operating in reset mode.

24.3.8 TIM6 and TIM7 Automatic Reload Registers (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ARR	RW	16'hFFFF	Automatic Reload Value The counter does not work when the value of auto-reload is null.

25. Low-power timer (LPTIM)

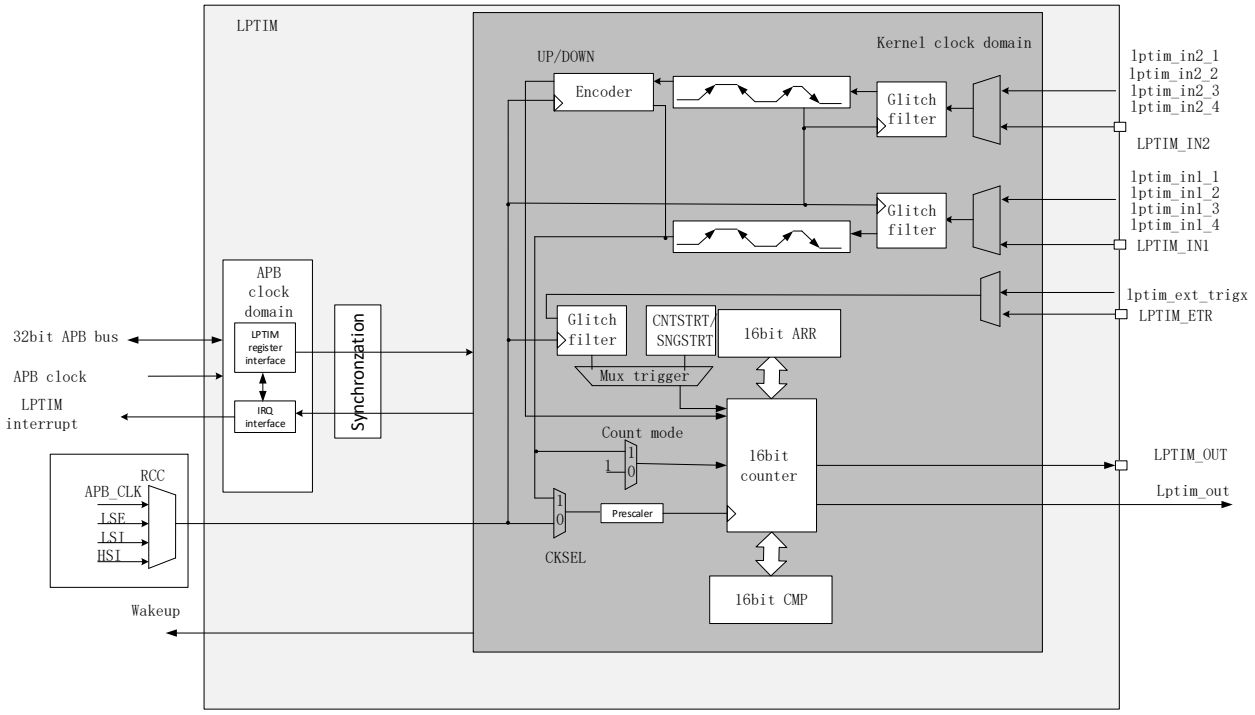
25.1 Introduction

PTIM is a 16-bit timer. Due to its diversity of clock sources, LPTIM is able to maintain operation in all power consumption modes except standby mode. Considering that LPTIM can operate even without an internal clock source, it can be used as a pulse counter, which is useful in some applications. LPTIM introduces a flexible clocking scheme that delivers the required functionality and performance while minimizing power consumption.

25.1.1 Main features

- 16 bit upcounter
- 3-bit prescaler with 8 possible division factors (1, 2, 4, 8, 16, 32, 64, 128)
- Selectable clock
  - Internal clock source: APB clock or any other oscillator (see RCC section)
  - External clock source via LPTIM input (works without running an embedded oscillator, used by the pulse counter application).
- 16-bit ARR reloadable register
- 16-bit comparison register
- Continuous/One-shot mode
- Optional software/hardware input trigger
- Configurable digital glitch filter
- Configurable outputs: Pulse, PWM
- Configurable IO polarity
- Encoder Mode

25.1.2 CAN block diagram



25.2 External Signal Description

25.2.1 LPTIM input and trigger mapping

Table 25-1 LPTIM external trigger connection

TRIGSEL	External trigger
lptim_ext_trig0	GPIO
lptim_ext_trig1	RTC_ALARM
lptim_ext_trig2	COMP1_OUT
lptim_ext_trig3	COMP2_OUT
lptim_ext_trig4	COMP3_OUT
lptim_ext_trig5	COMP4_OUT

Table 25-2 LPTIM Input 1 Connection

lptim_in1_mux	LPTIM input 1 connection
lptim_in1_0	GPIO
lptim_in1_1	COMP1
lptim_in1_2	COMP3

Table 25-3 LPTIM Input 2 Connection

lptim_in2_mux	LPTIM input 1 connection
lptim_in2_0	GPIO
lptim_in2_1	COMP2
lptim_in2_2	COMP4

25.3 LPTIM functional description

25.3.1 Reset and Clock

Through the RCC module, the LPTIM can be clocked using an internal clock signal. Multiple clock sources may be used to provide clocks for the LPTIM. It can be timed using an internal clock signal, which can be a PCLK (APB clock) or any other oscillator selectable via RCC. Furthermore, the LPTIM may be clocked by an external clock source Input1. When counting with an external clock source, LPTIM may run in one of two configurations:

- The first configuration is that the LPTIM is clocked by an external signal, but at the same time an internal clock signal is supplied to the LPTIM from the PCLK or any other oscillator.
- The second configuration is when the LPTIM is separately clocked by an external clock source through its external Input1. This configuration is used to implement the timeout function or pulse counter function when all oscillators are turned off after entering the low power mode.

Programming the CKSEL bit controls whether the LPTIM uses an external or internal clock source. When configured to use an external clock source, the CKPOL bit is used to select an external clock signal active edge.

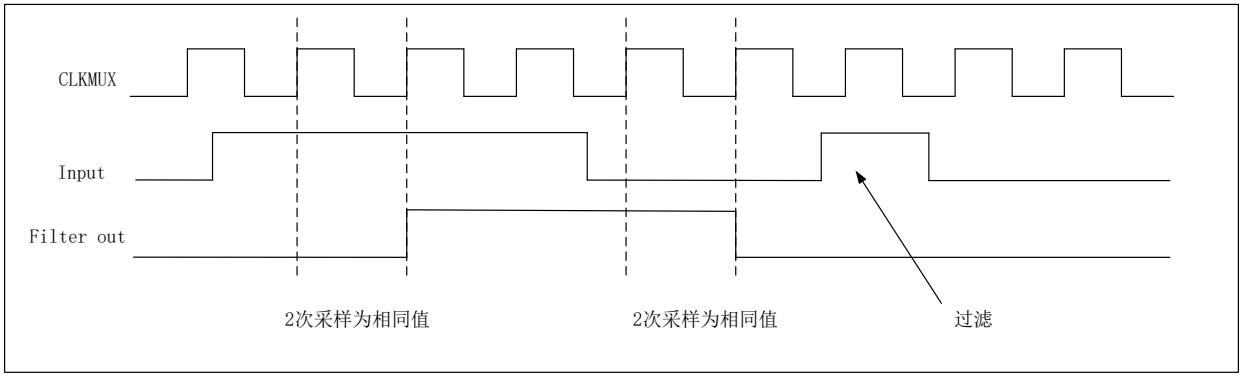
25.3.2 Glitch filter

The LPTIM inputs, whether external (mapped to the GPIO) or internal (mapped to other embedded peripherals at the chip level), are protected by digital filters that prevent any glitches and noise disturbances from propagating inside the LPTIM. This is to prevent false counts or triggers. Before activating the digital filter, an internal clock source should first be provided to the LPTIM. This is necessary to guarantee the proper functioning of the filter. The digital filters are divided into two groups:

- The first set of digital filters protects the LPTIM external input. The sensitivity of the digital filter is controlled by CKFLT bit
- A second set of digital filters protects the LPTIM internal trigger inputs. The digital filter sensitivity is controlled by the TRGFLT bit.

Note: Digital filter sensitivity is controlled by banks. It is not possible to configure each digital filter sensitivity individually within the same group.

The filter sensitivity acts on the number of successive equal samples detected on an input of the LPTIM to treat the signal level change as a valid transition. The figure below shows an example of the glitch filter behavior in the case of programming 2 consecutive samples.



If no internal clock signal is provided, the digital filter must be deactivated by setting the CKFLT and TRGFLT bits to "0". In this case, an external analog filter can be used to protect the LPTIM external input from glitch interference.

25.3.3 Prescaler

The LPTIM 16-bit counter is preceded by a configurable power-of-2 prescaler. The prescaler division ratio is controlled by LPTIM_CFGR.PRESC [2: 0].

The table below lists all the possible division ratios:

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

25.3.4 Trigger selection

The LPTIM counter can be started by software or after detecting a valid edge of one of the 6 (optional maximum number of inputs) trigger inputs.

TRIGEN [1: 0] is used to determine the LPTIM trigger source:

When TRIGEN [1: 0] equals "00", the LPTIM counter starts once one of the CNTSTRT or SNGSTRT bits is set by the software. The three remaining possible values of TRIGEN [1: 0] are used to configure the valid edge used by the trigger input. Once a valid edge is detected, the LPTIM counter starts.

When TRIGEN [1: 0] is not '00', TRIGSEL [2: 0] is used to select which of the trigger inputs is used to start the counter.

External triggers are treated as asynchronous signals for LPTIM. Therefore, after the trigger detection, due to the synchronization, a delay of two counter clock cycles is required before the timer starts running.

If a new trigger event occurs when the timer has already started, it will be ignored (unless the timeout feature is enabled).

The timer must be enabled before setting the SNGSTRT/CNTSTRT bit. When the timer is disabled, any writes to these bits will be discarded by the hardware.

When the counter is started via software (TRIGEN [1: 0] = 00), there is a delay of 3 count clock cycles between the LPTIM_CR register update (setting one of the SNGSTRT or CNTSTRT bits) and the effective start of the counter.

25.3.5 Mode of operation

LPTIM has two operating modes as follows:

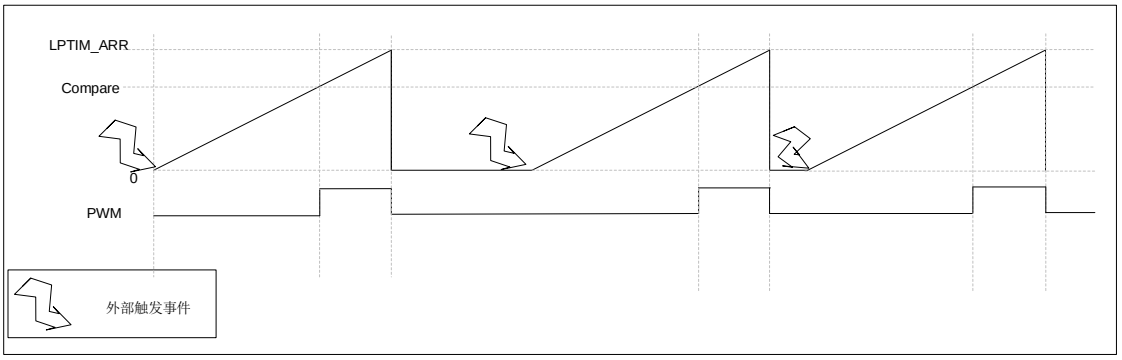
- Continuous counting mode: The timer runs freely, starts from the trigger event, and does not stop until the timer is disabled.
- Single count mode: The timer starts with a trigger event and stops when the ARR value is reached.

25.3.5.1 Single count mode

To enable single count, the LPTIM_CR.SNGSTRT bit must be set to 1.

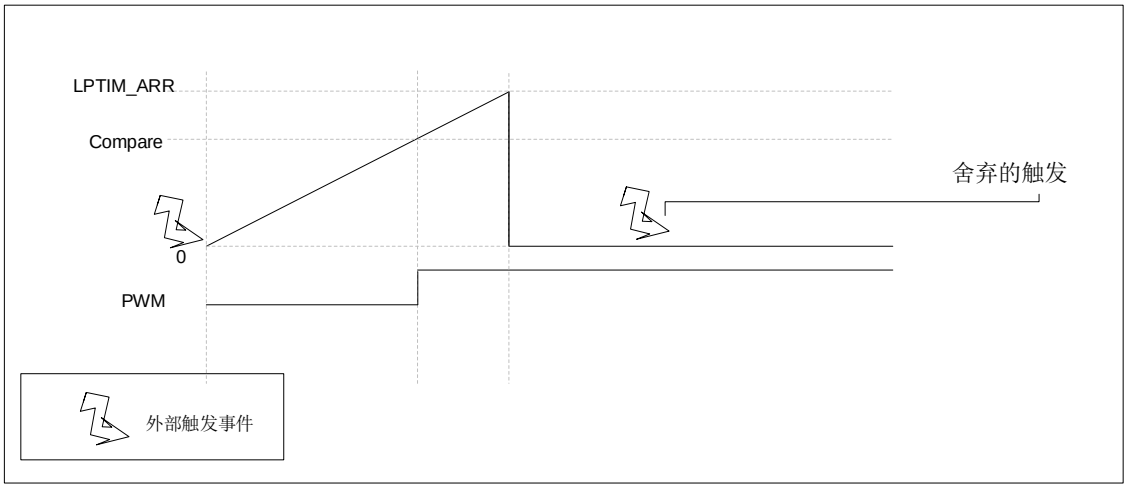
If External Trigger is selected, an external event that arrives after the SNGSTRT bit is set and the counter stops (containing zero values), starts the counter for a new single counting cycle, as shown in the following figure.

A new trigger event will re-start the timer. Any trigger events are ignored after the counter is started and until the ARR is reached.



-Activate setting mode once

When LPTIM_CFGR.WAVE position 1, the one-time settingmode is activated. In this case, the counter is only started once after the first trigger, and any subsequent trigger events are ignored. As shown in the following image:



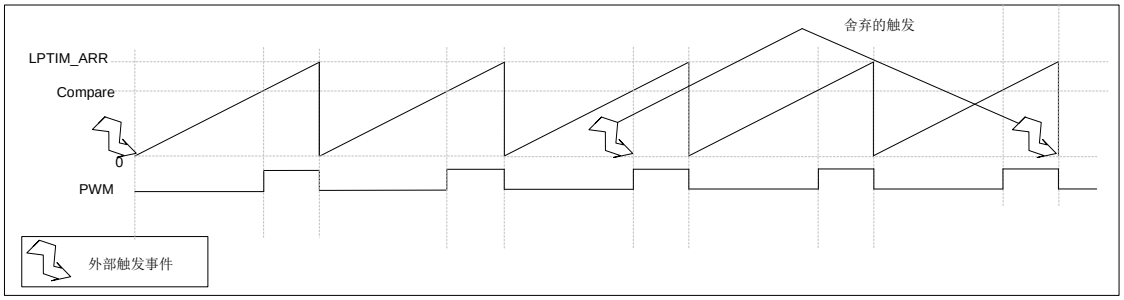
In case of software start (TRIGEN [1: 0] = '00'), the SNGSTRT set starts the counter for a single count.

25.3.5.2 Continuous counting mode

To enable the continuous counting, the LPTIM_CR.CNTSTRT bit must be set.

If external trigger is selected, external trigger events that arrive after CNTSTRT is set will start the counter for continuous counting. Any subsequent externally triggered events will be discarded as shown in the image below.

In case of software startup (TRIGEN [1: 0] = '00'), setting CNTSTRT starts the counter for continuous counting.



The LPTIM_CR.SNGSTRT and LPTIM_CR.CNTSTRT bits can only be set when the timer is enabled (LPTIM_CR.ENABLE is' 1 '). You can quickly change from single mode to continuous mode:

- If the Continuous mode was previously selected, setting SNGSTRT will switch the LPTIM to the One-shot mode. The counter (if it starts counting) stops as soon as it reaches ARR.
- If the One-shot mode was previously selected, setting LPTIM_CR.CNTSTRT will switch the LPTIM to the Continuous mode. The counter (if it starts counting) will restart as soon as it reaches ARR.

25.3.5.3 Waveform generation

Two 16-bit registers, LPTIM_ARR (Automatic Reload Register) and LPTIM_CMP (Comparison Register), are used to generate several different waveforms on the LPTIM output.

The timer can generate the following waveforms:

- PWM mode: The LPTIM output is set when the counter value in LPTIM_CNT exceeds the comparison value in LPTIM_CMP. Once a match occurs between the LPTIM_ARR and LPTIM_CNT registers, the LPTIM output is reset.
- Single pulse mode: The output waveform is similar to the PWM mode of the first pulse, and then the output is permanently reset.
- Once set mode: The output waveform is similar to monopulse mode, except that the output will remain at the last signal level (depending on the polarity of the output configuration).

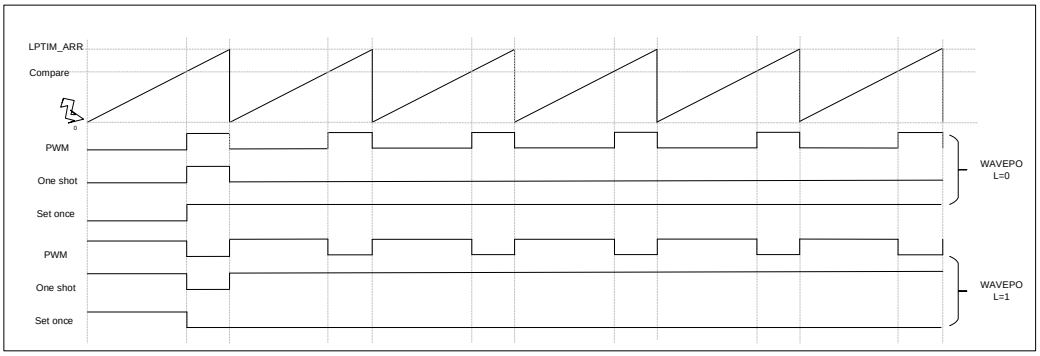
The above mode requires that the LPTIM_ARR register value be strictly greater than the LPTIM_CMP register value.

The LPTIM output waveform can be configured via the WAVE bit as follows:

- Resetting the WAVE bit to "0" forces LPTIM to generate either a PWM waveform or a monopulse waveform, depending on which bit is set: CNTSTRT or SNGSTRT.
- Setting the WAVE bit to "1" forces LPTIM to generate a set mode waveform once.

The WAVPOL bit controls the LPTIM output polarity. Changes take effect immediately, so the output defaults will change immediately after reconfiguring the polarity, even before enabling the timer.

A signal can be generated with a frequency up to the LPTIM clock frequency divided by 2. The figure below shows three possible waveforms that can be generated on the LPTIM output. Furthermore, it shows the effect of changing polarity using WAVPOL bits.



25.3.6 Register update

The LPTIM_ARR register and the LPTIM_CMP register are updated immediately after the APB bus write operation, or at the end of the current counting period if the timer has been started. The PRELOAD bit controls how the LPTIM_ARR register is updated:

When the PRELOAD bit is reset to "0", the LPTIM_ARR register is updated immediately after any write access.

When the PRELOAD bit is set to "1", the LPTIM_ARR register will be updated at the end of the current counting period if the timer has been started.

The LPTIM APB interface and the LPTIM kernel logic use different clocks, so there is some delay between the moment when the APB is written and the moment when these values are available for the counter comparator. Any additional writes to these registers must be avoided during this delay period.

The ARROK flag and the CMPOK flag in the LPTIM_ISR register indicate a write operation completion flag to the LPTIM_ARR register and the LPTIM_CMP register, respectively.

After writing to the LPTIM_ARR register or the LPTIM_CMP register, a new write operation can be performed to the same register only after the previous write operation is completed. Any successive writes before the ARROK or CMPOK flag position bit will result in unpredictable results.

25.3.7 Counter mode

The LPTIM counter can be used to count external events on LPTIM Input1 and can also be used to count internal clock cycles. The CKSEL bit controls which source will be used to update the counter.

If LPTIM is configured to count external events on Input1, the counter can be updated on the rising, falling, or double edges depending on the value written to the CKPOL [1: 0] bit.

Depending on the value of CKSEL, the following counting modes can be selected:

CKSEL = 0: LPTIM is timed by internal clock source

The LPTIM clock is provided by an internal clock source, and the LPTIM counter is configured to update after each internal clock pulse.

CKSEL = 1: LPTIM is clocked by an external clock source.

In this configuration, LPTIM does not require an internal clock source (clock glitch filtering is not supported in this mode). The signal injected on Input1 external to the LPTIM is used as the clock of the LPTIM. This configuration is suitable for operating modes where the embedded oscillator is not enabled.

For this configuration, the LPTIM counter can be updated at the rising or falling edge of the input1 clock signal, but not at the same time. Since the signal injected on Input1 outside the LPTIM is also used to clock the LPTIM kernel logic, there is some initial delay before the counter is incremented (after LPTIM is enabled).

25.3.8 Timer enable

The LPTIM_CR.ENABLE bit is used to enable/disable LPTIM kernel logic. After the ENABLE bit is set, two counter clocks need to be delayed to ENABLE LPTIM.

The LPTIM_CFGR and LPTIM_IER registers must be modified only when the LPTIM is disabled.

25.3.8.1 Encoder Mode

This mode allows processing of signals from the quadrature encoder used to detect the angular position of the rotating element. The encoder interface mode simply acts as an external clock with directional selection. This means that the counter just counts continuously between 0 and ARR values (0 to ARR or ARR to 0, depending on the direction). Therefore, LPTIM_ARR must be configured before startup. A clock signal is generated from two external input signals Input1 and Input2 to clock the LPTIM counter. The phase between these two signals determines the counting direction.

The encoder mode is only available when the LPTIM is counted by the internal clock source. The signal frequency on the Input1 and Input2 inputs must not exceed 1/4 of the LPTIM internal clock. This is a mandatory requirement to ensure the proper operation of LPTIM.

The direction change is indicated by two down and up flags in the LPTIM_ISR register. In addition, if the DOWNIE/UPIE bit is enabled, an interrupt can be generated for both direction change events.

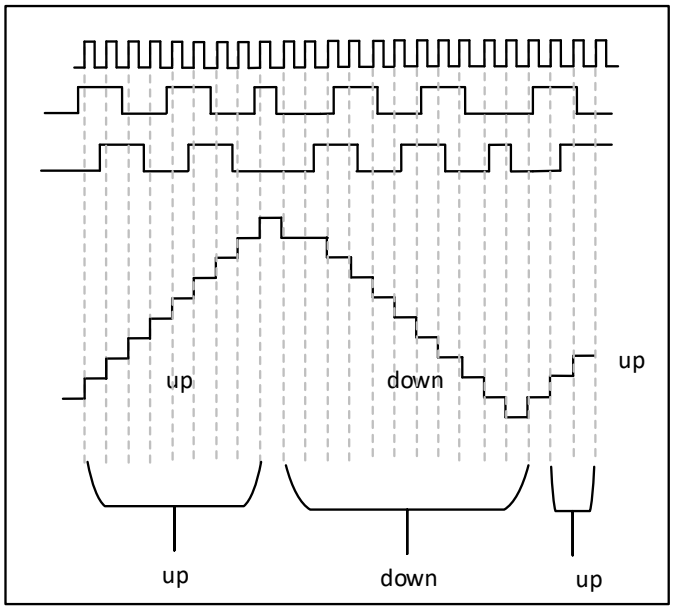
To activate encoder mode, the ENC bit must be set to "1". The LPTIM must first be configured in Continuous mode.

When the encoder mode is active, the LPTIM counter is automatically modified according to the speed and direction of the incremental encoder. Therefore, its contents always represent the position of the encoder. The counting direction indicated by the UP and DOWN flags corresponds to the rotational direction of the encoder rotor.

Different counting cases may occur depending on the edge sensitivity of the CKPOL [1: 0] bit configuration. The table below summarizes the possible combinations, assuming that Input1 and Input2 do not switch at the same time.

Effective edge	**Reverse signal level (Input1 for Input2, Input2 for Input1)**	**Input1 signal**		**Input2 signal**	
		Rising	**Falling**	**Rising**	**Falling**
rising edge	High	Minus	-	Add	-
	Low	Add	-	Minus	-
falling edge	High	-	Add	-	Minus
	Low	-	Minus	-	Add
Double edge	High	Minus	Add	Add	Minus
	Low	Add	Minus	Minus	Add

The figure below shows the count sequence for an encoder mode configured with dual-edge sensitivity. In this mode, the LPTIM must be clocked by an internal clock source, so the CKSEL bit must maintain its reset value of 0. In addition, the prescaler division ratio must be equal to its reset value of 1 (the PRESC [2: 0] bit must be "000").



25.3.9 Debug mode

When the microcontroller enters debug mode (core halted), the LPTIM counter either continues to work normally or stops, depending on the DBG_LPTIM_STOP configuration bit in the DBG module.

25.3.10 LPTIM low-power modes

Mode	Description
Sleep	No impact. LPTIM interrupts cause the device to exit Sleep mode.
Low-power run	No impact.
Low-power sleep	No impact. The LPTIM interrupt causes the device to exit low power sleep mode.
Stop0/Stop1	There is no effect when LPTIM is timed by LSE or LSI. The LPTIM interrupt causes the device to exit Stop 0 and Stop 1.
Standby	The LPTIM peripheral is powered down and must be reinitialized after exiting standby or shut-down mode.
Shutdown	

Note: When using lptim wakeup, it is necessary to ensure that the interval between two wakeups is greater than 6 PCLK cycles, otherwise it will lead to unpredictable results.

25.3.11 LPTIM interrupts

The following events generate an interrupt/wake-up event, if they are enabled through the LPTIM_IER register:

- Compare Match
- Automatic reload match (regardless of orientation if encoder mode)
- External trigger event
- Automatic reload register write complete
- Compare register write complete
- Direction change (encoder mode), programmable (up/down/both).

Note: If the corresponding bit in the LPTIM_IER register (interrupt enable register) is set to 1 after the corresponding flag in the LPTIM_ISR register (status register) is set to 1, no interrupt will occur.

Interrupt event	Description
Compare Match	An interrupt flag is generatedwhen the contents of the counter register (LPTIM_CNT) match the contents of the comparison register (LPTIM_CMP).
Auto-reload match	Interrupt flag is raised when the content of the Counter register (LPTIM_CNT) matches the content of the Auto-reload register (LPTIM_ARR).
External trigger event	When an external trigger event is detected, the interrupt flag is set
Automatic reload register write complete	An interrupt flag is generatedwhen a write operationto the LPTIM_ARR registeris completed.

Compare register write complete	An interrupt flag is generated when a write operationto the LPTIM_CMPregister is completed.
Change of direction	Used in encoder mode. Two interrupt flags are embedded to indicate a change of direction: — The UP flag indicates a change in increment count direction — The DOWN flag indicates that the count DOWN direction has changed

25.4 TIMx registers

25.4.1 LPTIM interrupt and status register, Address = 0x00 (LPTIM_ISR)

Address offset: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									DOWN	UP	ARROK	CMPOK	EXTTRIG	ARRM	CMPM
									R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	DOWN	R	0	The counter changes from up count to down count In encoder mode, the DOWN bit is set by the hardware to inform the application that the counter has changed from counting up to counting DOWN. The DOWN flag may be cleared by writing 1 to the DOWNCF bit in the LPTIM_ICR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UP	R	0	The counter changes from down count to up count In encoder mode, the UP bit is set by the hardware to inform the application that the counter has changed from count down to count UP. The UP flag may be cleared by writing 1 to the UPCF bit in the LPTIM_ICR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROK	R	0	Auto-reload register update OK ARROK is set by hardware to notify the application that the APB bus write to LPTIM_ARR has successfully completed. ARROK flag can be cleared by writing 1 to the ARROKCF bit in the LPTIM_ICR register.
3	CMPOK	R	0	Compare Register Update OK CMPOK is set by hardware to notify the application that an APB bus write operation to the LPTIM_CMP register has successfully completed.
2	EXTTRIG	R	0	External Trigger Edge Event EXTTRIG is set by hardware to notify the application that a valid edge has appeared on the selected external trigger input. This flag is not set if the trigger is ignored because the timer has already started. The EXTTRIG flag can be cleared by writing 1 to the EXTTRIGCF bit in the LPTIM_ICR register.
1	ARRM	R	0	Automatic overload matching. ARRM is set by hardware to inform the application that the value of the LPTIM_CNT register matches the value of the LPTIM_ARR register. Writing 1 to the LPTIM_ICR.ARRMCF bit clears the ARRM flag.
0	CMPM	R	0	Compare Match The CMPM bit is set by hardware to inform the application that the LPTIM_CNT register value has reached the value of the LPTIM_CMP register. Writing 1 to the LPTIM_ICR.CMPMCF bit clears the ARRM flag.

25.4.2 LPTIM interrupt clearing register, Address = 0x04 (LPTIM_ICR)

Address offset: 0x004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									DOWNCF	UPCF	ARROKCF	CMPOKCF	EXTTRIGCF	ARRMCF	CMPMCF
									W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	DOWNCF	W	0	Count Direction Changes to Down Clear Flag Writing 1 to this bit clears the DOWN flag in the LPTIM_ISR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UPCF	W	0	Direction changed to up clear sign Writing 1 to this bit clears the UP flag in the LPTIM_ISR register. Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROKCF	W	0	Auto-reload register update OK clear flag Writing 1 to this bit clears the ARROK flag in the LPTIM_ISR register
3	CMPOKCF	W	0	Compare Register Update OK Clear Flag Writing 1 to this bit clears the CMPOK flag in the LPTIM_ISR register
2	EXTTRIGCF	W	0	External trigger valid edge clear flag Writing 1 to this bit clears the EXTTRIG flag in the LPTIM_ISR register
1	ARRMCF	W	0	Auto-reload match clear flag Writing 1 to this bit clears the ARRM flag in the LPTIM_ISR register
0	CMPMCF	W	0	Compare Match Clear Flag Writing 1 to this bit clears the CMPM flag in the LPTIM_ISR register

25.4.3 LPTIM interrupt enable register, Address = 0x08 (LPTIM_IER)

Address offset: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									DOWNIE	UPIE	ARROKIE	CMPOKIE	EXTTRIGIE	ARRMIE	CMPMIE
									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	DOWNIE	RW	0	The counter changes from up count to down count interrupt enable 0: Disable DOWN interrupt 1: Enable DOWN interrupt Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
5	UPIE	RW	0	The counter changes from down count to up count interrupt enable 0: Disable UP interrupt 1: Enable UP interrupt Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
4	ARROKIE	RW	0	Automatic reload register update OK interrupt enabled 0: Disable ARROK interrupt 1: Enable ARROK interrupt
3	CMPOKIE	RW	0	Compare Register Update OK Interrupt Enable 0: Disable CMPOK interrupt 1: Enable CMPOK interrupt
2	EXTTRIGIE	RW	0	Externally triggered effective edge interrupt enable 0: Disable EXTTRIG interrupt 1: Enable EXTTRIG interrupt

1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: Disable ARRM interrupt 1: Enable ARRM interrupt
0	CMPMIE	RW	0	Compare Match Interrupt Enable 0: Disable CMPM interrupt 1: Enable CMPM interrupt

25.4.4 LPTIM configuration register, Address = 0x0C (LPTIM_CFGR)

Address offset: 0x00C

Reset value: 0x0000 0000

When configuring this register, you need to set the required registers at once.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							ENC	Res	PRELOAD	WAVPOL	WAVE	Res	TRIGEN [1: 0]		Res.
							RW		RW	RW	RW		RW	RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Trigsell [2: 0]			Res.	PRESC [2:0]			Res.	TRGFLT [1: 0]		Res.	CKFLT [1: 0]		CKPOL [1: 0]		CKSEL
RW	RW	RW		RW	RW	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	ENC	RW	0	Encoder Mode Enable The ENC bit controls the Encoder mode 0: Disable encoder mode 1: Enable encoder mode Note: This bit is reserved if the encoder mode function is not supported by LPTIM.
23	Reserved	-	-	-
22	PRELOAD	RW	0	Register update mode. The preload bit controls the LPTIM_ARR register update mode. 0: Update registers after each APB bus write access; 1: The register is updated at the end of the current LPTIM counting period;
21	WAVPOL	RW	0	Waveform polarity WAVPOL bit control output polarity 0: The LPTIM output reflects the comparison between the LPTIM_CNT and LPTIM_CMP registers 1: The LPTIM output reflects the inverse comparison result between the LPTIM_CNT and LPTIM_CMP registers
20	WAVE	RW	0	Wave shape WAVE bit control output shape 0: Disable the setting mode once, PWM or monopulse waveform depends on how the timer is started, CNTSTRT for PWM or SNGSTRT for monopulse waveform. 1: Activate the setting mode once
19	Reserved	-	-	-
18: 17	TRIGEN [1: 0]	RW	2'h0	Trigger enabling and polarity The TRIGEN bit controls whether the LPTIM counter is started by an external trigger. If the external trigger option is selected, there are three configurations to trigger valid edges: 00: Software trigger (count start is initiated by software) 01: Rising edge is valid edge 10: The falling edge is an effective edge 111: Reserved
16	Reserved	-	-	-
15: 13	Trigsell [2: 0]	RW	3'h0	Trigger selector The TRIGSEL bit selects among the following 6 available sources that will be the trigger source for the LPTIM trigger event: 000: lptim_ext_trig0 001: lptim_ext_trig1 010: lptim_ext_trig2 011: lptim_ext_trig3 100: lptim_ext_trig4 101 lptim_ext_trig5

12	Reserved	-	-	-
11: 9	PRESC [2:0]	RW	3'h0	Clock prescaler. The PRESC bits configure the prescaler division factor. It can be a factor in the following segments: 000: /1; 001: /2; 010: /4; 011: /8; 100: /16; 101: /32; 110: /64; 111: /128;
8	Reserved	-	-	-
7: 6	TRGFLT [1: 0]	RW	2'h0	Triggered configurable digital filter The TRGFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an internal flip-flop before it is considered a valid level transition. An internal clock source must be present to use this feature 00: Any trigger activity level change is considered a valid trigger 01: The trigger effective level change must be stable for at least 2 clock cycles before it is considered a valid trigger. 10: The trigger effective level change must be stable for at least 4 clock cycles before it is considered an effective trigger. 11: The trigger effective level change must be stable for at least 8 clock cycles before it is considered a valid trigger.
5	Reserved	-	-	-
4: 3	CKFLT [1: 0]	RW	2'h0	Configurable digital filter for external clock The CKFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on the external clock signal before it is considered a valid level transition. An internal clock source must be present to use this feature 00: Any external clock signal level change is considered an effective jump 01: The external clock signal level change must be stable for at least 2 clock cycles before it is considered an effective jump. 10: The level change of external clock signal must be stable for at least 4 clock cycles before it is considered an effective jump. 11: The external clock signal level change must be stable for at least 8 clock cycles before it is considered an effective jump.
2: 1	CKPOL [1: 0]	RW	2'h0	Clock polarity If LPTIM is clocked by an external clock source: When LPTIM is clocked by an external clock source, the CKPOL bit is used to configure one or more valid edges used by the counter: 00: The rising edge is the count valid edge If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 1 is active. 01: The falling edge is the count valid edge If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 2 is active. 10: Both edges are effective edges. When both external clock signal edges are considered valid edges, the LPTIM must also be clocked by an internal clock source at a frequency equal to at least four times the external clock frequency. If LPTIM is configured to encoder mode (set ENC bit), encoder sub-mode 3 is active. 11: Not allowed
0	CKSEL	RW	0	Clock selector The CKSEL bit selects which clock source the LPTIM will use: 0: LPTIM is clocked by an internal clock source (APB clock or any embedded oscillator 1: LPTIM is clocked by an external clock source via LPTIM Input1

25.4.5 LPTIM Control Register, Address = 0x10 (LPTIM_CR)

Address offset: 0x010

Reset value: 0x0000 0000

After configuring LPTIM1EN/LPTIM2EN of the RCC module, program LPTIM_CFGR.TRIGEN immediately, and then wait for 5 kernal clocks before configuring SNGSTRT or CNTSTRTof this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res													CNTSTRT	SNGSTRT	ENABLE
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	CNTSTRT	RW	0	Timer start in Continuous mode This bit is set by software and cleared by hardware. In case of software start (TRIGEN [1: 0] = '00'), this position 1 will start LPTIM in continuous mode. If software start is disabled (TRIGEN [1: 0]), set this bit to start the timer in continuous mode once an external trigger is detected. If this bit is set to 1 when a single count mode count is performed, the timer does not stop when the next LPTIM_ARR and LPTIM_CNT registers match. The LPTIM counter keeps counting in continuous mode. Note: This bit can be set only when the LPTIM is enabled. It will be zeroed automatically by the hardware.
1	SNGSTRT	RW	0	LPTIM start in Single mode This bit is set by software and cleared by hardware. Setting this bit starts the LPTIM in single pulse mode Note: This bit can be set only when the LPTIM is enabled. It will be automatically reset by hardware.
0	ENABLE	RW	0	LPTIM enable The enable bit is set 1 and cleared by the software. 0: LPTIM disabled. 1: LPTIM enabled.

25.4.6 LPTIM comparison register, Address = 0x14 (LPTIM_CMP)

Address offset: 0x014

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CMP	RW	16'h0	Automatically reload values. ARR is the comparison value of LPTIM. Note: This register cannot be updated until LPTIM is enabled.

25.4.7 LPTIM Auto Reload Register, Address = 0x18 (LPTIM_ARR)

Address offset: 0x018

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ARR	RW	16'h1	Automatically reload values. ARR is the auto-reload value of LPTIM. This value must be strictly greater than the CMP [15: 0] value. Note: This register cannot be updated until LPTIM is enabled.

25.4.8 LPTIM Count Register, Address = 0x1C (LPTIM_CNT)

Address offset: 0x01C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT	R	16'h0	Counter value. When the LPTIM is running with an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical. It should be noted that for a reliable LPTIM_CNT register read access, two consecutive read operations need to be performed and compared. When the values of consecutive read accesses are equal, a read access can be considered reliable.

25.4.9 LPTIM select register, Address = 0x20 (LPTIM_OR)

Address offset: 0x020

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res										IN2 [2: 1]		IN1 [2: 1]		IN2[0]	IN1[0]
										RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 4	IN2 [2: 1]	RW	2'h0	LPTIM input 2 remapping extension 00: COMP2 01: COMP4
3: 2	IN1 [2: 1]	RW	2'h0	LPTIM input 1 remapping extension 00: COMP1 01: COMP3
1	IN2[0]	RW	0	LPTIM input 2 remapping 1: Connect to COMP output indicated by IN2 [2: 1] 0: Connect to GPIO
0	IN1[0]	RW	0	LPTIM input 1 remap 1: Connect to COMP output indicated by IN1 [2: 1] 0: Connect to GPIO

26. Infrared interface (IRTIM)

An infrared interface (IRTIM) for remote control is integrated into the chip. It can realize the function of remote control together with an infrared LED. It uses the internal connection of TIM16 and TIM17 as shown in the figure below.

In order to generate the infrared remote control signal, the Infrared interface must be turned on and channel 1 (TIM16_OC1) of TIM16 and channel 1 (TIM17_OC1) of TIM17 must be properly configured to generate the correct waveform.

The infrared receiver can be easily implemented through a basic input capture mode.

All standard infrared pulse modulation modes can be obtained by programming the output comparison channels of the two timers.

TIM 17 is used to generate a high frequency carrier signal, while TIM 16 may generate a modulation envelope.

The infrared function is output to the IR_OUT pin, and the activation of this function is achieved by enabling the associated multiplexing function bit of the GPIO_AFRx register.

The LED requires a large sink current driving capability (only at the PB9/PA13pin), which can be activated through the I2C1_FMP bit of the SYSCFG_CFGR1 register, which is enough to support the direct control of the large sink current of the infrared LED.

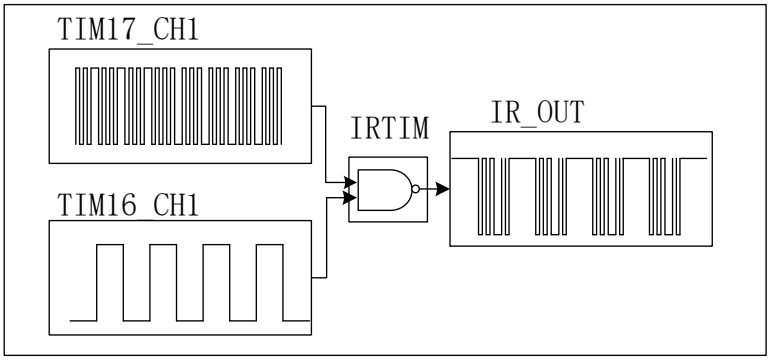


Figure 26-1 TRTIM internal connection diagram

27. Real-time clock (RTC)

27.1 Overview

The RTC is an independent counter. The RTC provides a set of continuously running counters which can be used, with suitable software, to provide a clock-calendar function. The counter values can be written to set the current time/date of the system.

The RTC module and the clock configuration system (RCC_BDCR register) are in the backup domain, i.e. the setting and time of the RTC remain unchanged after the system resets or wakes up from standby mode.

After the system reset, access to the backup registers and RTC is disabled to prevent accidental write operations to the backup domain (BKP). Doing the following will enable access to the backup registers and RTC:

- Set the PWREN and BKPEN bits of register RCC_APB1ENR to enable power and backup interface clocks
- Sets the DBP bit of the register PWR_CR to enable access to the backup register and RTC.

27.2 Main features

The DMA supports:

- Programmable pre-division coefficient: The division coefficient is up to 220.
- 32-bit programmable counter, which can be used for measurement of longer time periods.
- 2 separate clocks: PCLK1 and RTC clocks for the APB1 interface (the frequency of the RTC clock must be less than a quarter of the PCLK1 clock frequency).
- The RTC clock source could be any of the following ones:
  - 128 division of the HSE clock;
  - LSE oscillator clock;
  - LSI oscillator clock
- Two separate reset types:
  - The APB1 interface is reset by the system;
  - RTC cores (prescalers, alarms, counters, and dividers) can only be reset by the backup domain.
- 2 dedicated maskable interrupts:
  - Alarm interrupt, used to generate a software-programmable alarm interrupt.
  - RTC global interrupts, including second interrupts, alarm clock interrupts, and overflow interrupts.

27.3 Functional description

27.3.1 Overview

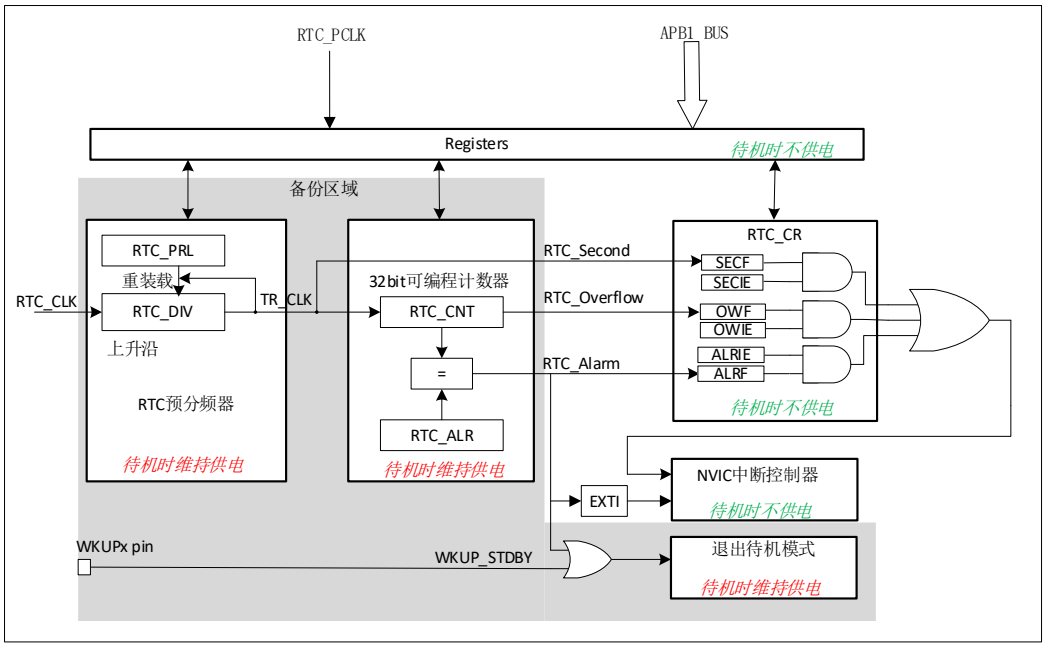


Figure 27-1 RTC functional block diagram

The RTC consists of two main parts (see figure above). The first part (APB1 interface) is used to connect to the APB1 bus. This unit also contains a set of 32-bit registers (RTC_CR, registers actually scattered at two addresses), which can be read and written through the APB1 bus. The APB1 interface is driven by the APB1 bus clock and is used to connect to the APB1 bus.

The other unit (RTC Core) consists of a chain of programmable counters made of two main blocks.

The first module is the prescaler module of the RTC, which is programmable to produce an RTC time reference TR_CLK of up to 1 second. It includes a 20-bit programmable divider (RTC prescaler). If the corresponding allow bit is set in the RTC_CR register, RTC generates an interrupt (second interrupt) in each TR_CLK cycle.

The second block is a 32-bit programmable counter that can be initialized to the current system time. The system time is accumulated in TR_CLK cycles and compared to the programmable time stored in the RTC_ALR register. If the corresponding allow bit is set in the RTC_CR control register, an alarm interrupt will be generated when the comparison matches.

The alarm interrupt serves as a wake-up signal for the standby mode and the stop mode. The picture above shows the situation in standby mode, after waking up from standby mode. In stop mode, the RTC alarm clock needs to be configured as a wake-up EXTI line17 corresponding register.

27.3.2 Register reset

The RTC_PRL, RTC_ALR, RTC_CNT, and RTC_DIV registers can only be reset by the backup domain reset signal. Other system registers (RTC_CR) are reset asynchronously by system reset or power reset.

27.3.3 Reading RTC registers

The RTC core is completely independent of the RTC APB1 interface.

The software accesses the prescalation values, counter values and alarm clock values of RTC through APB1 interface. However, the associated readable register is updated only when the signal after synchronization with the rising edge of the RTC clock to the RTC APB1 clock is valid. This is also true for the RTC flags.

This means that if the APB1 interface has been closed and the read operation is just after APB1 is returned, the RTC register value read from APB1 may be corrupted (usually read to 0) before the first internal register update. This can occur if:

- A system reset or power reset has occurred
- The system just woke up from standby mode

The system just woke up from shutdown mode (in this case the count value just won't update because the CPU doesn't work and the system clock stops but the RTC counts normally and the count value won't sync to the VDDD zone)

In all the above cases, the RTC core remains running when the APB1 interface is disabled (reset, no clock, or power down).

Therefore, if the APB1 interface of the RTC has been disabled when reading the RTC register, the software must first wait for the RSF bit (register synchronization flag) in the RTC_CR register to be set '1' by the hardware.

Note:

1) CPU readable registers include RTC_CR, RTC_CNT, RTC_ALR, and RTC_DIV;
2) The RTC_CR register is the RTC_PCLK field, and the CPU can read a stable value at any time;
3) RTC_CNT and RTC_DIV are derived from the RTC_CLK domain. After RTC operation, the RTC_DIV register is updated at the rising edge of each RTC_CLK; The RTC_CNT and the flag bits originating from the RTC_CLK clock domain also use the same update signal as the RTC_DIV register, although the value of RTC_CNT does not change every time it is updated;
4) In the RTC_PCLK domain, the RSF is set when the pulse signal after the RTC_CLK is synchronized to the RTC_PCLK is valid;
5) RSF only controls the read timing of RTC_CNT and RTC_DIV (hardware does not control it)

27.3.4 Configuring RTC registers

The CNF bit in the RTC_CR register must be set so that RTC enters configuration mode before writing to the RTC_PRL, RTC_CNT, and RTC_ALR registers.

In addition, writing to any RTC register is only enabled if the previous write operation is finished. It is possible to determine whether the RTC register is being updated by querying the RTOFF status bit in

the RTC_CR register. A new value can be written to the RTC registers only when the RTOFF status bit value is ’1’.

Configuration procedure:

- Querying the RTOFF bit until the value of RTOFF becomes' 1 '; (indicates that the previous configuration has been completed)
- Set the CNF value to 1 to enter the configuration mode; (At this time, the RTOFF bit is still 1, ensuring that the CPU RTOFF = 1 when writing to the register)
- Performing a write operation to one or more RTC registers; (RTOFF is still 1 in this process, and the RTC_CLK field register is written to the buffer register in this step)
- Clear the CNF flag bit and exit the configuration mode; (After the hardware detects that CNF is cleared, it starts to perform the register write operation in the previous step and starts to write to the registers in the RTC_CLK field; At the same time, RTOFF is cleared)
- Poll RTOFF, wait until its value goes to ‘1’ to check the end of the write operation. (The buffer register is written to the RTC_CLK field after setting RTOFF)

Note: The write operation only executes when the CNF bit is cleared; it takes at least three RTC_CLK cycles to complete. (3 RTC_CLK cannot restart the configuration after the CNF flag is cleared, otherwise a configuration error will occur (in this case, it is controlled by RTOFF = 0))

Note:

1) In this process, RTOFF = 1 when the CPU writes to the register;
2) The write cycle of the CPU is from CNF = 1 to CNF = 0, and other registers are configured between these two operations;
3) First write CNF to 1 and then clear CNF. This operation clears RTOFF; RTOFF will not be cleared after only writing CNF = 0 or not clearing after writing CNF = 1;
4) First write the CNF to 1 and then clear the CNF. This operation starts the process of writing the buffer register to the RTC_CLK field register;
5) RTOFF is implemented in the RTC_PCLK domain.

27.3.5 RTC Flag Settings

The RTC Second Flag (SECF) is set before changing the RTC counter during each RTC core clock cycle.

In the last RTC clock cycle before the counter reaches 0x0000, the RTC overflow flag (OWF) is set. (OWF is set at the maximum value of detected RTC_CNT before reaching 0)

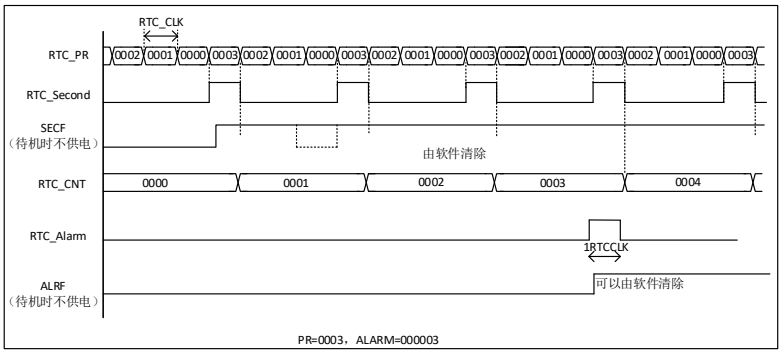
The RTC_Alarm and the RTC alarm flag (ALRF) are set in the RTC clock cycle before the value of the counter reaches the value of the alarm register plus 1 (RTC_ALR+1). (RTC_CNT count to RTC_ALR detected and ALRF set at last RTC_CLK)

Writes to the RTC alarm register (RTC_ALR) must be synchronized with the RTC second flag using one of the following procedures:

- Use the RTC alarm interrupt and modify the RTC alarm register (RTC_ALR) and/or the RTC counter register (RTC_CNT) in the interrupt handler.
- Wait for the SECF bit in the RTC control register to be set before changing the RTC alarm register (RTC_ALR) and/or the RTC counter register (RTC_CNT).

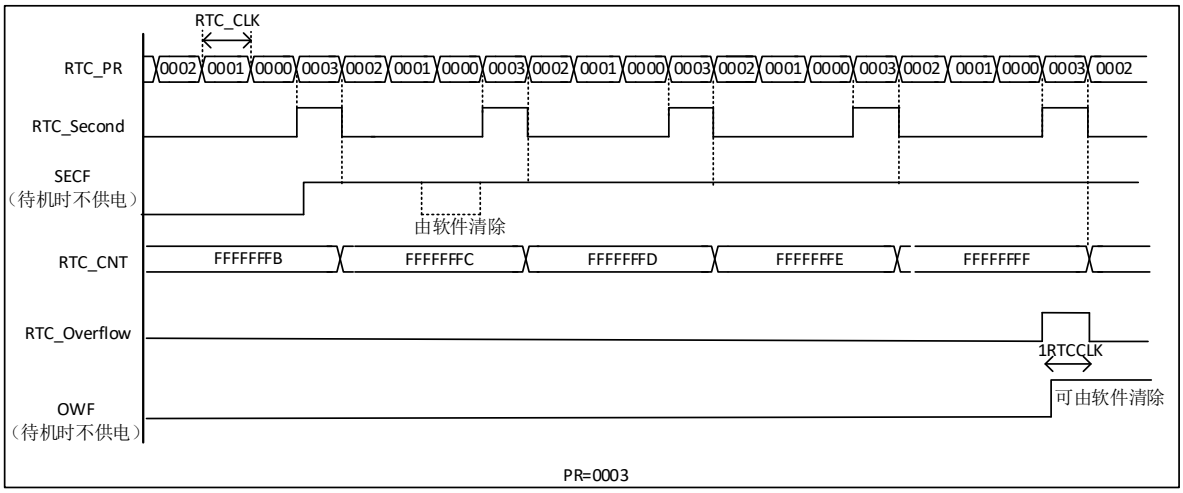
27.3.6 RTC timing

(1) RTC seconds and alarm timing



SECF and ALRF are RTC_PCLK domain signals, and are generated by sampling RTC_Sectond and RTC_Alarm signals in RTC_CLK domain, respectively.

(2) RTC overflow timing



SECF and OWF are RTC_PCLK domain signals, which are generated by sampling RTC_CLK domain RTC_Sectond and RTC_Overflow numbers, respectively.

27.4 TIMx registers

The register width is 32 bits and only allows word operations.

27.4.1 RTC Control Register High Bit (RTC_CRH) (0x00)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OWIE	ALRIE	SECIE
-													RW	RW	RW

This register is reset by a system reset.

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	OWIE	RW	0	Overflow interrupt enable 0: overflow interrupt disabled 1: overflow interrupt enabled
1	ALRIE	RW	0	Alarm interrupt enable

				0: Alarm interrupt disabled 1: Alarm interrupt enabled
0	SECIE	RW	0	Second interrupt enable 0: second interrupt disabled 1: second interrupt enabled

These bits are used to mask interrupt requests. Note that at reset all interrupts are disabled, so it is possible to write to the RTC registers to ensure that no interrupt requests are pending after initialization. It is not possible to write to the RTC_CRH register when the peripheral is completing a previous write operation.

The RTC functions are controlled by this control register. Some bits must be written using a specific configuration procedure.

27.4.2 RTC Control Register Low (RTC_CRL) (0x04)

Address offset: 0x04

Reset value: 0x0000 0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	SECF
-										R	RW	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	-
5	RTOFF	R	1	RTC operation OFF, this bit is read-only. With this bit the RTC reports the status of the last write operation performed on its registers, indicating if it has been completed or not. If its value is ‘0’ then it is not possible to write to any of the RTC registers. 0: Last write operation on RTC registers is still ongoing. 1: Last write operation on RTC registers terminated.
4	CNF	RW	0	Configuration flags This bit must be set '1' by the software to enter the configuration mode to allow new values to be written to the RTC_CNT, RTC_ALR or RTC_PRL registers. The write operation is only executed when the CNF bit is reset by software after has been set. 0: Exit configuration mode (start updating RTC registers) 1: Enter configuration mode
3	RSF	RC_W0	0	Registers synchronized flag. This register is set by hardware and cleared by software. When the RTC_CNT register and the RTC_DIV register are updated, the hardware sets the bit '1'. After the APB1 is reset, or after the APB1 clock stops, this bit must be cleared '0' by the software. Before any read operation is to be performed, the user program must wait for the bit to be set '1' by the hardware to ensure that RTC_CNT, RTC_ALR, or RTC_PRL have been synchronized. 0: Registers not yet synchronized. 1: Registers synchronized.
2	OWF	RC_W0	0	Overflow flag When the 32-bit programmable counter overflows, this bit is set '1' by the hardware. An interrupt is generated if OWIE=1 in the RTC_CRH register. It can be cleared only by software. Writing ‘1’ has no effect. 0: no overflow; 1: 32-bit programmable counter overflow
1	ALRF	RC_W0	0	Alarm flag When the 32-bit programmable counter reaches a predetermined value set by the RTC_ALR register, this bit is set '1' by the hardware. An interrupt is generated if ALRIE=1 in the RTC_CRH register. It can be cleared only by software. Writing ‘1’ has no effect. 0: No alarm; 1: There is an alarm clock.
0	SECF	RC_W0	0	Second flag

				When the 32-bit programmable prescaler overflows, this bit is set '1' by the hardware and the RTC counter is incremented by 1. Therefore, this flag provides a periodic signal (typically 1 second) to the resolution programmable RTC counter. Aninterrupt is generated if SECIE=1 in the RTC_CRH register. It can be cleared only by software. Writing '1' has no effect. 0: Second flag condition does not hold 1: The second flag condition is established

The functions of the RTC are controlled by this control register. When the peripheral is continuing the last write operation (RTOFF = 0), the RTC_CRL register cannot be written.

Notes:

- Any flag bit will remain suspended until the appropriate RTC_CR request bit is reset by the software, indicating that the requested interrupt has been received.
- All interrupts are disabled during reset, there are no pending interrupt requests, and the RTC register can be written
- When the APB1 clock is not running, the OWF, ALRF, SECF and RSF bits are not updated (cannot be synchronized).
- The OWF, ALRF, SECF and RSF bits can only be set by hardware and cleared by software.
- If ALRF = 1 and ALRIE = 1, an RTC global interrupt is allowed to be generated. If an EXTI line 17 interrupt is allowed to be generated in the EXTI controller, an RTC global interrupt and an RTC alarm interrupt are allowed to be generated.
- If ALRF = 1, an RTC alarm interrupt is allowed if the interrupt mode of the EXTI line 17 is set in the EXTI controller; If the event mode of EXTI line 17 is set in the EXTI controller, a pulse is generated on this line (no RTC alarm interrupt is generated).

27.4.3 RTC Prescale Load Register High Bit (RTC_PRLH) (0x08)

The PRL register is used to hold the count value of the periodicity of the RTC prescaler. This register is write-protected by the RTOFF bit of the RTC_CR register. Only when RTOFF = 1 is allowed to write the CPU.

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL [19:16]			
-												W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	PRL [19:16]	W	4'h0	RTC prescaler reload value high (RTC prescaler reload value high) These bits define the clock frequency of the counter according to the following formula: FTR_CLK= fRTCCLK/(PRL [19: 0] +1) Note: 0 value is not recommended, otherwise RTC interrupt and flag bits cannot be generated correctly

27.4.4 RTC Prescale Load Register Low Bit (RTC_PRLL) (0x0C)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

Address offset: 0x0C

Reset value: 0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRL [15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	PRL [15: 0]	W	16'h8000	RTC prescaler reload value high. These bits are used to define the clock frequency of the counter according to the following formula: FTR_CLK= fRTCCLK/(PRL [19: 0] +1) Note: The value of 0 is not recommended, otherwise RTC interrupt and flag bits cannot be generated correctly.

27.4.5 RTC Prescalation Remainder Register High Bit (RTC_DIVH) (0x10)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

At each TR_CLK cycle, the value of the RTC_PRL register is reloaded into the RTC prescalation counter. The user can obtain an accurate time measurement by reading the RTC_DIV register to obtain the current value of the prescalation counter without stopping the operation of the frequency division counter.

This register is a read-only property, and when there is any change in the value of the RTC_PRL or RTC_CNT register, the register value will be reloaded by the hardware.

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV [19:16]			
-												R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	DIV [19:16]	R	4'h0	RTC clock divider remainder high bit.

27.4.6 RTC Prescalation Remainder Register Low (RTC_DIVL) (0x14)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

Address offset: 0x14

Reset value: 0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	DIV [15: 0]	R	16'h8000	RTC clock divider

27.4.7 RTC Count Register High (RTC_CNTH) (0x18)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

The RTC module has a 32-bit programmable counter, which is accessed through two registers. The count is based on the TR_CLK time reference generated by the prescaler.

RTC_CNT register is used to store the count value of the counter. The register is write-protected, and the CPU can only write when RTOFF = 1. Write to the upper 16-bit RTC_ CNTH or lower 16-bit RTC_CNTL register, load it directly into the corresponding programmable counter, and reload the RTC prescaler. When reading, the current value in the counter (system date) is returned.

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT [31: 16]	RW	16'h0000	RTC core counter 16 bits higher. When the RTC_CNTH register is read, the upper 16 bits of the current value of the RTC counter register are returned. To write to this register it is necessary to enter configuration mode.

27.4.8 RTC Count Register Low (RTC_CNTL) (0x1C)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	CNT [15: 0]	RW	16'h0000	RTC core counter Lower 16 bits When the RTC_CNTL register is read, the lower 16 bits of the current value of the RTC counter register are returned. To write to this register it is necessary to enter configuration mode.

27.4.9 RTC alarm register high (RTC_ALRH) (0x20)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

When the programmable counter (count) reaches the 32-bit value stored in the RTC_ALR register, an alarm interrupt request is generated. This register is write-protected by the RTOFF bit, and a write operation is allowed if the RTOFF value is '1'.

Address offset: 0x20

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALR [31: 16]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ALR [31: 16]	RW	16'hFFFF	RTC alarm clock value is 16 bits higher. This register is used to hold the upper 16 bits of the alarm clock time written by software. To write to this register it is necessary to enter configuration mode.

27.4.10 RTC alarm register low (RTC_ALRL) (0x24)

This register is located in the VBAT field, and the data does not change after the microcontroller is powered down.

Address offset: 0x24

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	ALR [15: 0]	RW	16'hFFFF	RTC alarm clock value is 16 bits lower. This register is used to hold the upper 16 bits of the alarm clock time written by software. To write to this register it is necessary to enter configuration mode.

28. Independent watchdog (IWDG)

28.1 Introduction

Independent watchdog register (IWDG for short), which has the characteristics of high security level, accurate timing and flexible use. IWDG can be used to detect and resolve faults caused by software errors and trigger a system reset when the counter reaches a specified TIMEOUT value (TIMEOUT).

The Independent Watchdog (IWDG) is driven by a dedicated low-speed internal clock (LSI), which is still active even if the master clock fails.

The IWDG is best suited for applications that require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.

28.2 Main features

- Free running decrementing counter
- The clock is provided by an independent RC oscillator (can operate in stop mode and standby mode)
- When the watchdog is activated, a reset is generated when the counter is decremented to 0x000
- With low power freezing function, it can keep working state in stop mode and standby mode, and the counter keeps decrementing count

28.3 Function description

28.3.1 CAN block diagram

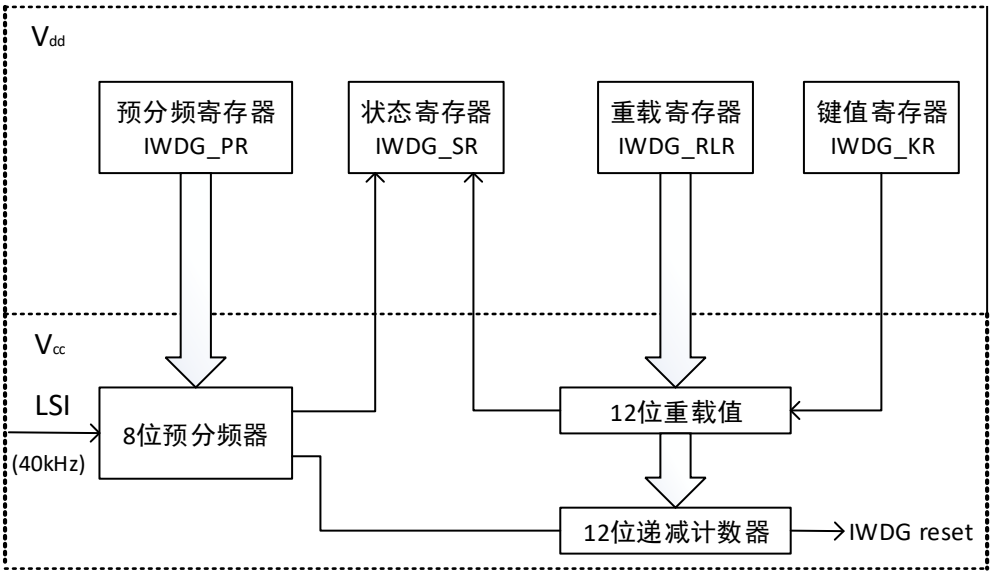


Figure 28-1 IWDG module block diagram

Note: The watchdog function is in the Vcc power supply zone, that is, it can still work normally in shutdown and standby modes.

Write 0xCCCC in the key register (IWDG_KR) to start enabling the independent watchdog; At this point the counter starts to count down from its reset value of 0xFFF. When the counter counts to the end of 0x000, a reset signal (IWDG_RESET) is generated.

Whenever 0xAAAA is written to the key register IWDG_KR, the value in IWDG_RLR is reloaded into the counter to avoid an IWDG reset.

Table 28-1 Watchdog Timeout 40 kHz Input Clock (LSI)

Prescaler division ratios	PR [2: 0] bit	Minimum Time (ms) PR [11: 0] = 0x000	Maximum time (ms) PR [11: 0] = 0xFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	6	6.4	26214.4

Note: These times are given as per the 40 kHz clock. In fact, the RC frequency inside the microcontroller will vary between 30 kHz and 60 kHz. Furthermore, even if the frequency of the RC oscillator is accurate, the exact timing still depends on the phase difference between the APB interface clock and the RC oscillator clock, so there will always be a complete RC cycle that is uncertain. A relatively accurate watchdog timeout can be obtained by calibrating the LSI.

28.3.2 Hardware watchdog

If the user enables the "hardware watchdog" function in the selection byte, the watchdog will automatically start running after the system is powered on and reset; If the software does not write the corresponding value to the key register before the counter count ends, the system will generate a reset

28.3.3 Register access protection

Write access to the IWDG_PR and IWDG_RLR registers is protected.To modify the values of these two registers, IWDG_KR must first be written to 0x5555. Writes to other numbers of these registers will re-turn write protection, such as write 0xAAAA, and IWDG_PR and IWDG_RLR will be re-protected. IWDG_SR indicates whether the value of IWDG_PR or IWDG_RLR is being updated.

28.3.4 Debug mode

If the CPU enters debug mode,the counter of IWDG continues to work or stopsdepending on the status of the DBG_IWDG_STOP configuration bit in the DBG module in the debug module

28.3.5 Low power freeze

Depending on the IWDG_STOP and IWDG_STBY bits in the option byte, it is determinedwhether the counting is continued in the IWDG stop modeorstandbymode. If the IWDG remains operating in stop or standby mode, the IWDG can wake up the microcontroller from the current low power mode.

28.4 TIMx registers

You can only operate the following registers in a word manner.

28.4.1 Key Value Register (IWDG_KR)

Address offset: 0x00

Reset value: 0x0000 0000 (reset by standby mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY [15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	KEY [15: 0]	W	16'h0	Key Value.

				These bits must be written by software at regular intervals with the key value 0xAAAA, otherwise the watchdog generates a reset when the counter reaches 0. Write 0x5555: Allow access to IWDG_PR and IWDG_RLR; Write 0xCCCC: Activates IWDG (not controlled by this command if hardware watchdog is selected).

28.4.2 IWDG prescaler register (IWDG_PR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR [2: 0]		
-													RW		

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2: 0	PR [2: 0]	RW	3'h0	Prescaler divider They are written by software to select the prescaler divider feeding the counter clock. PVU bit of the IWDG status register (IWDG_SR) must be reset in order to be able to change the prescaler divider. This register is write protected. Writing 0x5555 to IWDG_KR releases the write protection. 000: divider /4; 001: divider /8; 010: divider /16; 011: divider /32; 100: divider /64; 101: divider /128; 110: divider /256; 111: divider /256;

28.4.3 IWDG reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF (Reset by standby mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	RL [11: 0]											
-				RW											

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	RL [11: 0]	RW	12'h0	IWDG counter reload value They are written by software to define the value to be loaded in the watchdog counter each time the value0xAAAA is written in the IWDG key register (IWDG_KR). The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. This register is write protected. Writing 0x5555 to IWDG_KR releases the write protection. The RVU bit in the IWDG status register (IWDG_SR) must be reset to be able to change the reload value.

28.4.4 IWDG status register (IWDG_SR)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
-														R	R

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	RVU	R	0	Watchdog counter reload value update This bit is set to 1 by the hardware to indicate that the reload value is being updated. It is reset by hardware when the reload value update operation is completed.
0	PVU	R	0	Watchdog prescaler value update This bit is set to 1 by the hardware, indicating that the prescalation value is being updated. It is reset by hardware when the prescaler update operation is completed.

Note: Before updating IWDG_PR and IWDG_SR.RLR, wait for IWDG_PVU and IWDG_SR.RVU to be 0, respectively. However, after updating IWDG_PR and IWDG_RLR, you do not have to wait for IWDG_SR.PVU and IWDG_SR.RVU to be 0, and you can continue to executesubsequent code.

29. Window watchdog (WWDG)

29.1 Introduction

Window watchdogs are usually used to monitor software failures caused by application running away caused by external interference or unforeseen logical conditions. Unless the value of the decrement counter is refreshed before the T [6] bit becomes 0, the watchdog circuit generates a reset when the preset time period is reached. If the 7-bit decrement counter value (in the control register) is refreshed before the decrement counter reaches the window register value, then a reset will also occur. This indicates that the decrement counter needs to be refreshed in a limited time window.

29.2 Main features

- Programmable free-running downcounter
- If the watchdog is activated, a system reset occurs when any of the following conditions are met:
  - The decrement counter has a value less than 0x40
  - The decrement counter is reloaded outside the window
- If the watchdog is activated and the early wake-up interrupt (EWI) is turned on, the EWI is generated when the decrement counter is equal to 0x40, and the software may reload the counter within the EWI interrupt program to avoid WWDG reset.

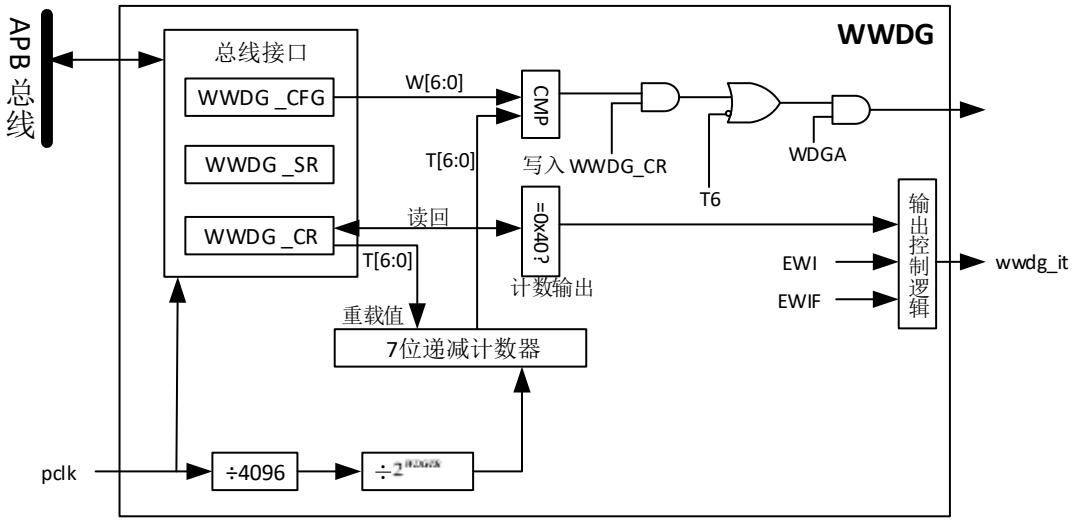


Figure 29-1 WWDG module block diagram

29.3 WWDG functional description

If the watchdog is enabled (the WDGA bit in the WWDG_CR register is set to "1"), and when the 7-bit (T [6: 0]) decrement counter is flipped from 0x40 to 0x3F (T [6] bit cleared), a reset is generated. If the software reloads the counter when the counter value is greater than the value in the window register, a reset will be generated.

The softwaremust be written to the WWDG_CR register periodically during normal operation to prevent the microcontroller from resetting. A write operation can only be performed if the counter value is less than the value of the window register. The value stored in the WWDG_CR register must be between 0xFF and 0xC0:

- Enabling the watchdog

When WWDG_SW in the option byte selects the software to start WWDG (WWDG_SW = 1), WWDG is turned off after the system is reset,and WWDG is enabled by setting WDGA in WWDG_CR to 1. Once WWDG is started by the software, it cannot be shut down unless a system reset occurs.

When WWDG_SW in the option byte selects hardware start WWDG (WWDG_SW = 0), the WWDG clock starts automatically after the system is reset, and WWDG is in the ON state and cannot be turned off.

- Controlling the downcounter

This downcounter is free-running, counting down even if the watchdog is disabled. When the watchdog is enabled, the T [6] bit must be set to 1 to prevent a reset from occurring immediately.

The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset. The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WWDG_CR register.

The configuration register (WWDG_CFR) contains the upper limit value of the window: to avoid a reset, the decrement counter must be reloaded when its value is less than the value of the window register and greater than 0x3F.

Another way to reload the counter is to utilize an early wake-up interrupt (EWI). Setting the EWI bit in the WWDG_CFR register turns on the interrupt. This interrupt is generated when the decrement counter reaches 0x40, and the corresponding interrupt service routine (ISR) can be used to load the counter to prevent the WWDG from resetting. This interrupt can be cleared by writing '0' in the WWDG_SR register.

Note: You can use the T [6] bit to generate a software reset (set the WDGA bit to "1" and the T [6] bit to "0").

29.4 How to program the watchdog timeout

You can calculate the timeout of the window watchdog using the formula provided in the figure below.

Warning: When writing to the WWDG_CR register, always set the T [6] bit to '1' to avoid immediately generating a reset.

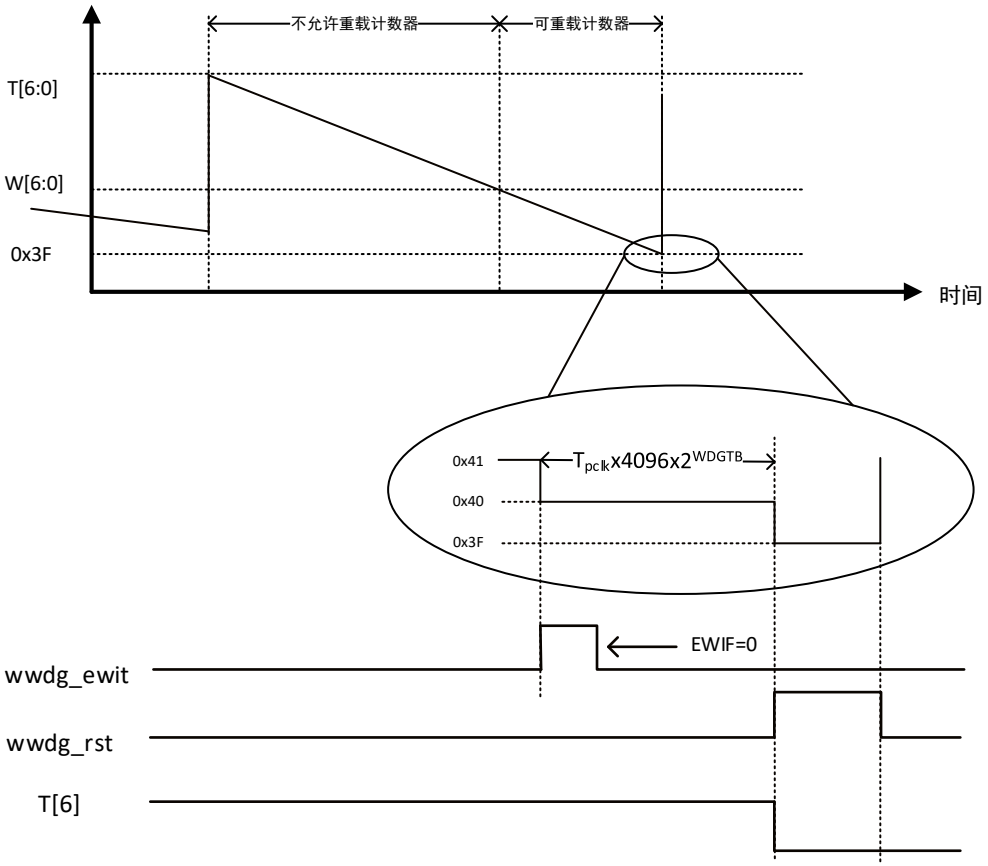


Figure 29-2 Window Watchdog Timing Diagram

The formula for calculating the WWDG timeout value is as follows:

tWWDG = tPCLK x 4096 x 2WWDGTB [1: 0] x (T [5: 0] + 1) (ms)

29.5 Debug mode

When the microcontroller enters the debug mode (Cortex-M4F core stops), it isdetermined whetherthe counter of WWDG continues to work or stops according to the state of the DBG_WWDG_STOP configuration bit in the debug module. See the Debug support for details.

29.6 TIMx registers

29.6.1 WWDG control register (WWDG_CR)

Address offset: 0x00

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								WDGA	T [6: 0]						
-								RS	RW						

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7	WDGA	RS	0	Activation bit Activation bit This bit is set to "1" by software, but can only be cleared to "0" by hardware after reset. When WDGA = 1, the watchdog can generate a reset. 0: No Watch Dogs 1: Enable Watchdog
6: 0	T [6: 0]	RW	7'h7F	7-bit counter (MSB to LSB)

				These bits are used to store the counter value of the watchdog. Minus 1 per (4096 × 2 ^ WDGTB) PCLK1 cycle. When the counter value changes from 0x40 When it is 0x3F (T [6] becomes 0), a watchdog reset is generated.

29.6.2 WWDG configuration register (WWDG_CFR)

Address offset: 0x04

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB [1: 0]		W [6: 0]						
-						RS	RW		RW						

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
9	EWI	RS	0	Early wake-up interrupt Early wakeup interrupt If this bit is set to '1', an interrupt will be generated when the counter value reaches 0x40. This interrupt can only be cleared by the hardware after reset.
8: 7	WDGTB [1: 0]	RW	2'h0	Timer base The time base of the prescaler may be set as follows: 00: CK timer clock (PCLK1 divided by 4096) divided by 1 01: CK timer clock (PCLK1 divided by 4096) divided by 2 10: CK timer clock (PCLK1 divided by 4096) divided by 4 11: CK timer clock (PCLK1 divided by 4096) divided by 8
6: 0	W [6: 0]	RW	7'h7F	7-bit window value These bits contain the window values used for comparison with the decrement counter.

29.6.3 WWDG status register (WWDG_SR)

Address offset: 0x08

Reset value: 0x0000 0000

21	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															EWIF
-															RC_W0

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	EWIF	RC_W0	0	Early wake-up interrupt flag Early wakeup interrupt flag When the counter value reaches 0x40, this bit is set to "1" by the hardware. It must be cleared by software writing "0". Writing "1" to this bit is not valid. If the interrupt is not enabled, this bit will also be set to "1".

30. External serial memory controller (ESMC)

30.1 Introduction

ESMC (External serial memory controller) is a dedicated communication interface for single, dual, quad,dual-quad and octal channels SPI interface memory (NOR Flash, PSRAM, etc.). It can run in either of the following two modes:

- Indirect mode: All operations are performed using ESMC registers
- Memory mapped (XIP) mode: External flash memory is mapped to the device address space and the system treats it as internal memory

Using dual memory mode, two Qual SPI memories are accessed simultaneously, similar to a single Octal SPI memory, doubling throughput and capacity.

30.2 Main Features

- Supports two functional modes: indirect and memory mapped
- Can transmit/receive 8 bits simultaneously
  - Dual flash mode, which allows simultaneous transmission/reception of 8 bits by accessing two flashes in parallel
  - Octal SPI
- Supports SDR and DDR
- Has receive and transmit FIFO
- Allows 8-bit, 16-bit, and 32-bit data access (indirect mode only supports 32-bit read)
- In indirect mode, DMA operation is supported
- Interrupt indicating FIFO status and completion of operation

30.3 ESMC functional description

30.3.1 ESMC external memory connection diagram

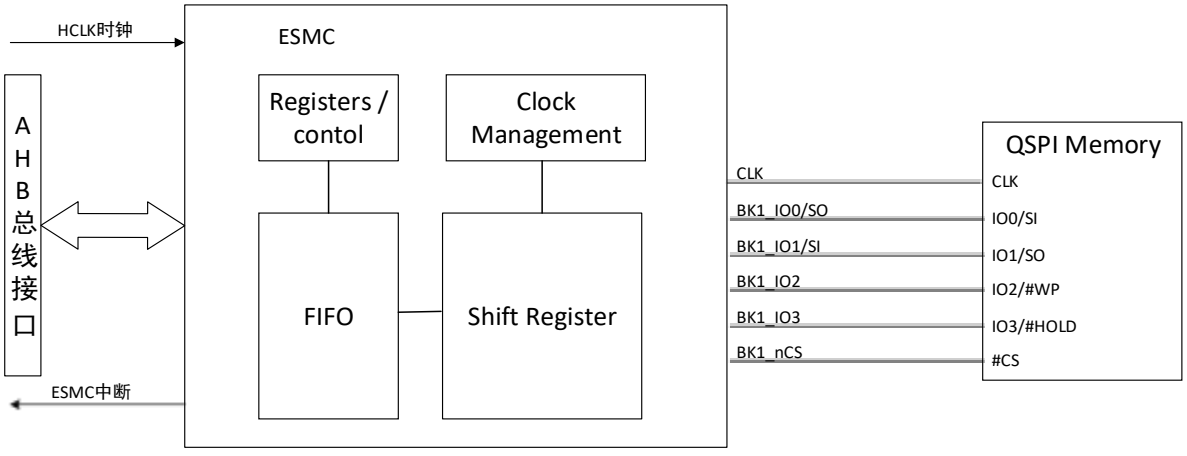


Figure 30-1 ESMC connection diagram (when Dual QSPI Memory mode is disabled)

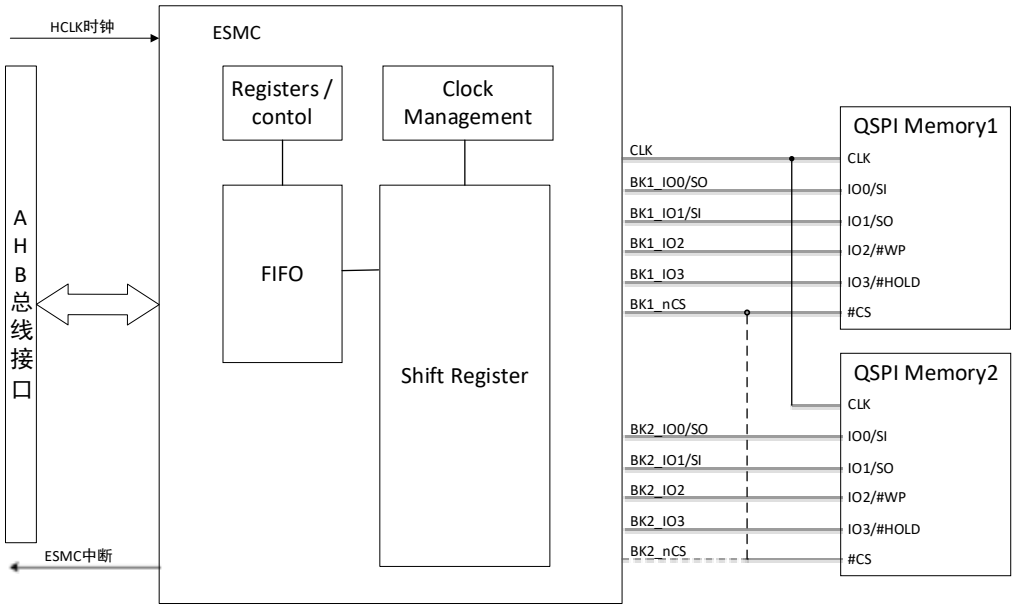


Figure 30-2 ESMC connection diagram (when Dual QSPI Memory mode is set)

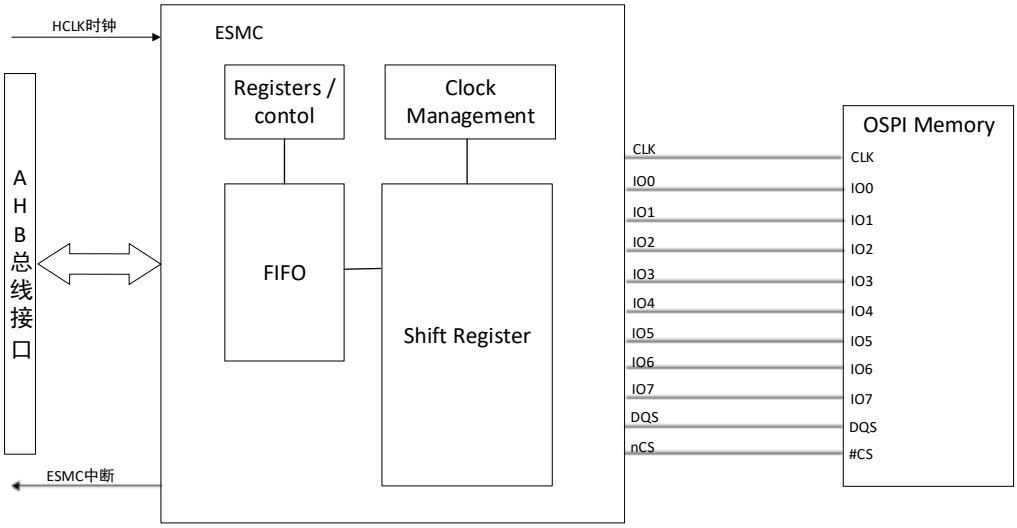


Figure 30-3 ESMC connection diagram (when OSPI Memory mode is set)

30.3.2 ESMC Command Sequence

The ESMC uses commands to communicate with external memory. Each command can include 5 stages: Instruction, Address, Spare Byte, Dummy, Data. Any configuration of these phases can be skipped, but at least one of the instruction, address, spare byte, or data phases must be present.

The nCS pulls low before the start of each command and pulls high again after the completion of each command.

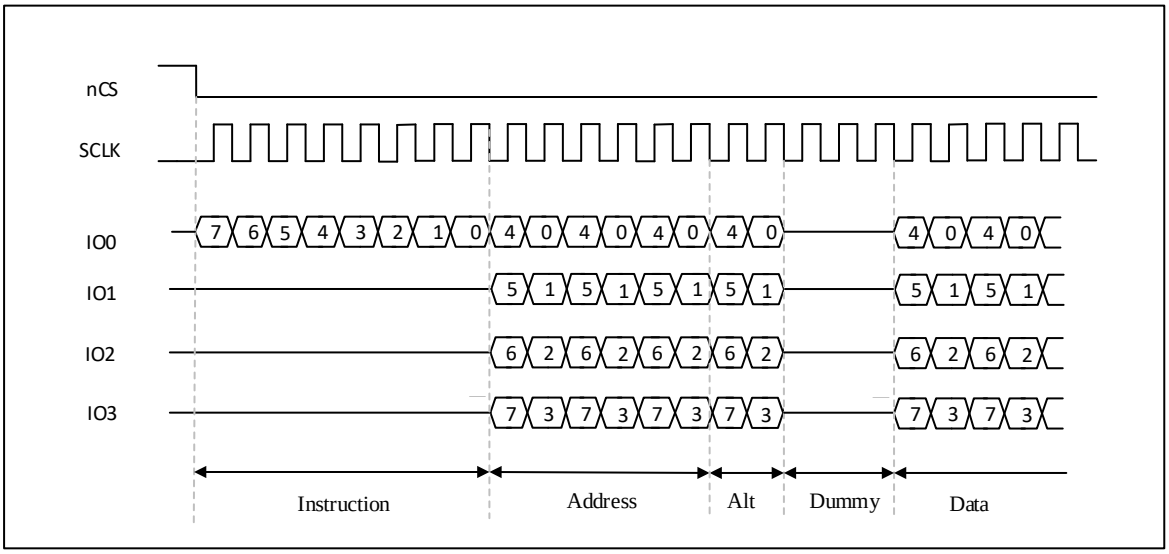


Figure 30-4 QSPI instruction example

Instruction phase

At this stage, an 8-bit instruction configured in the instruction field of the ESMC_SFCR [7: 0] or ESMC_ADDR24 [7: 0] register is sent to the flash memory, specifying the type of operation to be performed.

Although most flash memories can only receive one bit of instruction at a time (per cycle) from the IO pin (single SPI mode), the instruction stage can choose to send 2 bits at a time (IO0/IO1 in dual SPI mode), 4 bits at a time (IO0/IO1/IO2/IO3 in quad SPI mode), or 8 bits at a time (IO0/IO1/IO2/IO3/IO4/IO5/IO6/IO7 in eight SPI mode).

The length of the instruction may be set to no instruction length, 1 byte instruction and 2 byte instruction.

The size of the instruction length is implemented by the following registers:

Size of instruction length	Configured registers
No instruction code	Enable bit: ESMC_DCR/ESMC_XDCR/ESMC_XDCR_WE.NO_CMD = 1
1-byte instruction code	Enable bit: none, bit 1 byte instruction by default 1st byte instruction: ESMC_ADDR24.INS1 [7: 0] or ESMC_SFCR (send instruction only)
2-byte instruction code	Enable bit: ESMC_CR3/ESMC_XCR3/ESMC_XCR3_WE.CMD_EXT = 1 1st byte instruction: ESMC_ADDR24.INS1 [7: 0] 2byte instruction:write bit ESMC_ADDR32. MREG [7: 0] or ESMC_XMODE or ESMC_XMODE_WE

Address phase

In the address phase, 1-4 bytes are sent to the flash memory to indicate the address of the operation. The number of address bytes to be sent is configured in the 32 BIT_ADDR, 16 BIT_ADDR, 8 BIT_ADDR bits of the ESMC_CR3 register (the default is a 24-bit address). In indirect mode, the address byte to be sent is specified in the address [31: 0] field of the ESMC_ADDR register, while in memory-mapped mode, the address is givendirectly over the AHB bus.

Addresses can be sent 1 bit at a time (through IO0 in single SPI mode), 2 bits at a time (through IO0/IO1 in dual SPI mode), 4 bits at a time (through IO0/IO1/IO2/IO3 in quad SPI mode) or 8 bits at a time (IO0/IO1/IO2/IO3/IO4/IO5/IO6/IO7 in eight SPI mode).

When the ESMC_SOCR register NO_ADDR bit is set, the address phase will be skipped and the command sequence will proceed directly to the next phase.

The ESMC contains four 8-bit address registers ADDR3, ADDR2, ADDR1, ADDR0, allowing addressing flash memory that supports 24-bit and 32-bit address fields. By default, ESMC is configured as a 24-bit address.

A 32-bit address is sent from the ESMC during access to the external SPI flash memory. By default, only the 24 LSB bits of the address register are sent. When ADDR32BIT (CR3 [2]) is set, the 32-bit address will be transferred out of the ESMC. The address register values are transmitted in the currently selected SPI mode (extended SPI, single SPI, dual SPI, quad SPI, or eight SPI) channel. The address register should be written before the ESMC loads the SFCR register.

Alternate byte phase

In the spare byte phase, 1 byte is sent to the flash memory and is typically used to control the mode of operation. The spare byte data to be transmitted is configured in the MREG [7: 0] field of the ESMC_ADDR32 [23:16] register.

The spare byte stage can send 2 bits at a time (through IO0/IO1 in dual SPI mode), or 4 bits at a time (through IO0/IO1/IO2/IO3 in quad SPI mode).

When ESMC_SOCR.SEND_M = 0, the spare byte phase will be skipped and the command sequence will proceed directly to the next phase.

Dummy stage

In the DUMMY phase, 1-63 cycles are given without sending or receiving any data, so that the flash memory has time to prepare data when using a higher clock frequency. The number of cycles given by this phase is specified in the DUMMY [5: 0] field of the ESMC_DCR register. In both SDR and DDR modes, the duration is specified as the number of full SCLK cycles.

When DUMMY is zero, the DUMMY cycle phase is skipped and the command sequence goes directly to the data phase.

Data Phase

In the data phase, any number of bytes may be sent to or received from the flash memory.

In indirect mode, the number of bytes transmitted/received is specified in the ESMC_BCR register.

In the indirect write mode, data sent to the flash memory mustfirst bewritten to the ESMC_DATA register, while in the indirect read mode, data received from the flash memory is obtained by reading the ESMC_DATA register.

In memory-mapped mode, the data read/written is directly through the AHB bus.

The data phase can send/receive 1 bit (through IO0/IO1 in single SPI mode), 2 bits (through IO0/IO1 in dual SPI mode), 4 bits (through IO0/IO1/IO2/IO3 in quad SPI mode) or eight bits (IO0/IO1/IO2/IO3/IO4/IO5/IO6/IO7 in eight SPI mode) at a time.

30.3.3 ESMC Interface Protocol Mode

Single SPI mode

Legacy SPI mode allows only a single bit to be transmitted/received serially. In this mode, data is sent to the flash memory via the SO pin (this I/O is shared with IO0) and received from the flash memory via the SI pin (this I/O is shared with IO1).

Single SPI mode is used by setting ESMC_SOCR [1: 0] to 00.

Dual SPI mode

In dual SPI mode, two bits of data are transmitted/received simultaneously via the IO0/IO1 pin.

The dual SPI mode is used by setting ESMC_SOCR [1: 0] to 01.

Quad SPI Mode

In Quad SPI mode, four bits of data are sent/received simultaneously through the IO0/IO1/IO2/IO3 pins.

The quad SPI mode is used by setting ESMC_SOCR [1: 0] to 10.

Dual Flash mode

When 2QUAD bit (ESMC_CR) is 1 and ESMC_SOCR [1: 0] is set to 11, ESMC is in dual flash mode where two external QUAD SPI flashes (Flash1 and Flash 2) are used to send/receive 8 bits of data per cycle (or 16 bits of data in DDR mode), doubling throughput and capacity.

Each flash uses the same SCLK pin and optionally the same nCS pin, but each has separate IO0, IO1, IO2, and IO3 pins.

The dual flash mode can be combined with the SDR or DDR mode.

Each byte of [7: 4] bits of data is stored in flash memory 1, and each byte of [3: 0] bits of data is stored in flash memory 2.

When reading a Flash status register in dual Flash mode, twice as many bytes should be read compared to reading the same byte in single Flash mode. This means that if the status bit of each Flash is 8 bits, QUAD SPI must be configured to be 2 bytes, and QUADSPI receives one byte from each Flash. If the state of each Flash is 16 bits, QUAD SPI must be configured to be 4 bytes, and QUADSPI receives two bytes from each Flash.

The first byte of the data register is the first byte of Flash1, and the second byte is the first byte of Flash2. Then, the third byte of the data register is the Flash1 second byte, and the fourth byte is the Flash2 second byte (in the case of Flash having a 16-bit status register).

In dual flash mode, an even number of bytes must always be accessed. Therefore, ESMC_BCR needs to be set to an even number.

Note: In dual flash mode, the address written to the external memory is half of the actual address. For example, the written address is 0x8, and the actual address sent to the external memory is 0x4. The read address is consistent with the actual address. For example, the read address is 0x8, and the actual address sent to the external memory is also 0x8.

Octal SPI Mode

In eight SPI mode, eight bits of data are sent/received simultaneously through the IO0/IO1/IO2/IO3/IO4/IO5/IO6/IO7 pins.

The eightSPI modeis used by setting ESMC_SOCR [1: 0] to 11.

SDR mode

By default, the DDRDATA, DDRADDR, DDRCMD bits (ESMC_SOCR) are 0, and ESMC operates in single data rate (SDR) mode.

In SDR mode, when ESMC drives the IO0, IO1, IO2, IO3 pins, these pins only transition with the SCLK falling edge.

When receiving data in SDR mode, it is assumed that the flash memory transmits data using the falling edge of the SCLK, and the ESMC samples the signal using the upper and falling edge of the SCLK.

DDR mode

When the DDR bitis set(data, address, instructioncan be set respectively by setting the DDRDATA, DDRADDR, DDRCMD bitsof the ESMC_SOCR register), the ESMC operates in dual data rate (DDR) mode.

In DDR mode, when the ESMC drives the IO0, IO1, IO2, IO3, IO4, IO5, IO6, IO7 pins in the address/spare byte/data phase, one bit of data is sent on each falling and rising edge of the SCLK.

When receiving data in DDR mode, assuming that the flash memory also transmits data using rising and falling SCLK edges, the ESMC samples the signal after half of the SCLK period (at the next opposite edge).

Usage of DQS (/RWDS)

The DQS pin can be used to read data strobe.

In eight SPI mode,the XSPI_RWDS enable of the configuration register (ESMC_CR3) is set to 1 to enable DQS management.

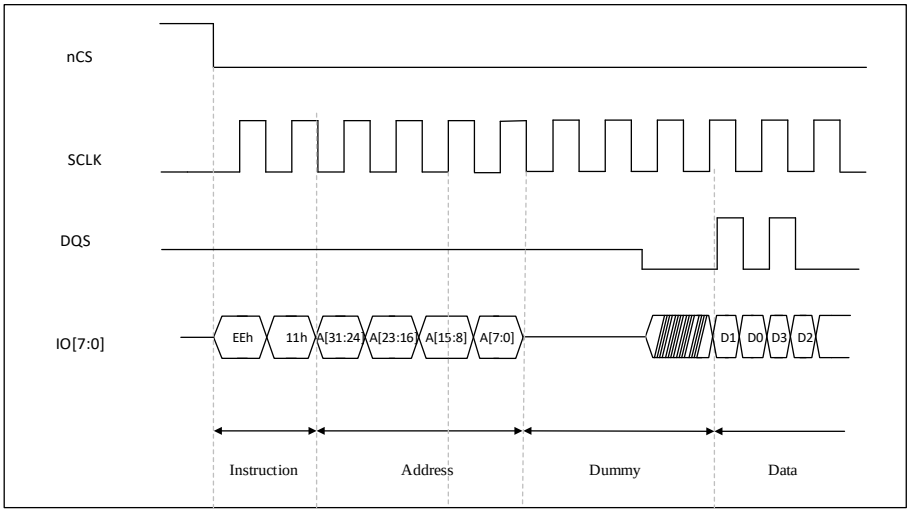


Figure 30-5 8-DDR read timing of DQS in line mode

Table 30-1 The memory types that ESMC can support are as follows

Device Type	Interface protocol mode		Data direction
NOR Flash	1 line	SDR	Reception
			Transmission
		DDR	Reception
	2 lines	SDR	Reception
			Transmission
		DDR	Reception
	4 lines	SDR	Reception

			Transmission
		DDR	Reception
	Double 4 wire	SDR	Reception
			Transmission
		DDR	Reception
	8 line	SDR	Transmission
			Reception
		DDR	Transmission
			Reception
PSRAM	1 line	SDR	Reception
			Transmission
	4 lines	SDR	Reception
			Transmission
	Double 4 wire	SDR	Reception
			Transmission

30.3.4 ESMC Working Mode

ESMC has two modes of operation:

- Indirect mode (register control)
- Memory Mapping Mode

The working mode is set by the XIP_EN bit (6th bit of ESMC_CR) (this module supports XIP read and XIP write functions, but the XIP write function only supports PSRAM use except 8 lines):

When set to 1, it is memory mapped mode.

When set to 0, it is indirect mode.

30.3.4.1 ESMC indirect mode

In indirect mode, commands are initiated by writing to the relevant register and data is transferred by writing or reading the data register.

When REC = 0 (7th bit of ESMC_DCR), ESMC is in indirect write mode,sending bytes to Flash during the data phase bywriting to the data register (ESMC_DATA).

When REC = 1, ESMC is in indirect read mode, receiving bytes from Flash during the data phase by reading ESMC_DATA.

30.3.4.2 ESMC memory mapping mode

In memory mapped mode, ESMC allows data to be read or written directly from SPI Flash. In this mode, the ESMC operates as a serial-to-parallel converter between the SPI flash and the CPU. Different from indirect mode, memory mapping (XIP) mode has dedicated registers, and the general registers are XSTRR, XSSOCR and XSSOCR_WE; Read operations include XMODE, XSFCR, XSOCR, XDCR and XCR3; The write operation registers are XMODE_WE, XSFCR_WE, XSOCR_WE, XDCR_WE, and XCR3_WE.

The instructions sent during the XIP access are stored in the ESMC XIP instruction registers (XSFCRand XSFCR_ WE). These two registers function similarly to the SFCR registers, but are only used in XIP mode, and writing to the registers does not initiate instruction code transfer. The transfer of instructions is completed in the selected SPI mode.

To operate in XIP mode, you need to configure SPI Flash first. Once configured correctly, the XIP operation of the ESMC is ready after powering on and no further action is required.

The ESMC controller allows FLASH reads using the preset READ Command after the system is reset. Other transmission configurations need to be implemented by configuring XIP related registers through software.

In the XIP mode, the CPU starts the read operation according to the command of the XSFCR configuration of the ESMC or starts the transmission operation according to the command of the XSFCR_WE configuration of the ESMC through the AHB bus, and the address comes from the AHB bus. In the process of using XIP, if the user wants to send a new FLASH command, he does not need to prohibit XIP, but only avoid any access to the XIP memory space during the operation, otherwise the operation will terminate and start XIP reading.

To disable XIP mode in ESMC, the XIP_EN bit (ESMC_ CR[6]) needs to be cleared in the software. Once this bit is 0, access to the XIP bus has no effect on the ESMC. XIP operations are enabled by default, so READ can be started immediately after system reset. It can be disabled in the software by clearing the XIP enable bit (ESMC_ CR [6]). If the XIP_EN bit (ESMC_ CR [6]) is cleared, the XIP bus is set to busy to indicate an access error.

Note: When XIP is written in mode0, it needs to be set to more than two-division frequency to work.

30.3.5 Programmable options for ESMC

30.3.5.1 Byte count

ESMC uses a 32-bit register (BCR register) to control the transfer of data, including read operations and write operations;

ESMC supports fixed length data transfer, and by default ESMC will read/write data from external memory as long as there is at least one free space/at least one data in RXFIFO/TXFIFO. Once the storage space of the RXFIFO/TXFIFO is full/empty, the ESMC will enter the WAIT state and WAIT for the host to read data from the RXFIFO/TXFIFO or write data. The RXFIFO must be completely empty, or the TXFIFO must be written to full, before the next transmission can be started.

This mode is only available in indirect mode. After reading the specified number of bytes, the SSO pin is deactivated and the ESMC enters the IDLE state, waiting for the next command to be executed.

Table 30-2 The number of read and write bytes that can be supported in different modes

Size	Indirect mode reading	Indirect mode write	XIP Mode Read	XIP Mode Write	DMA Mode Read	DMA Mode Write
Bytes	x	√	√	√	x	√
Half word	x	√	√	√	x	√
Word	√	√	√	√	√	√

30.3.5.2 Polarity and Phase

ESMC only supports SPI protocol operation in Mode 0 and Mode 3.

This is achieved by configuring the ESMC_CR2 [1: 0] register.

When ESMC_CR2. CPOL and ESMC_CR2. CPHA are both set to 0, mode 0 is indicated. If both ESMC_CR2. CPOL and ESMC_CR2. CPHA are set to 1, it is indicated as SPI mode 3.

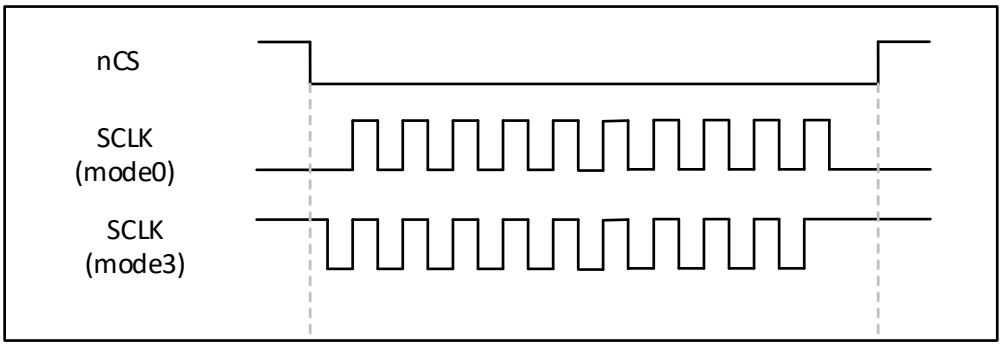


Figure 30-6 Timing of Mode 0 and Mode 3 in SDR mode

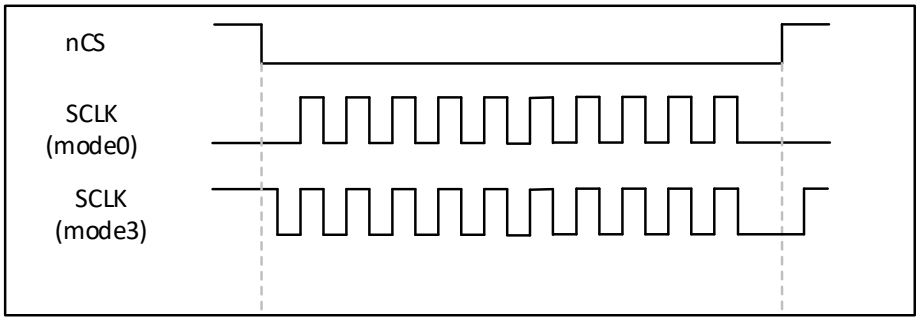


Figure 30-7 Timing of Mode 0 and Mode 3 in DDR Mode

30.3.5.3 Baud rate

The BAUD rate register BAUD (0x3) is used to define the SCLK frequency of the ESMC, divided based on the system clock. When BAUD is set to 0x0 and 0x1, the frequency is divided into two; That is, the fastest SCLK clock is the two-division frequency of the system clock. Other values are calculated as:

SCLK = HCLK/BAUD;

30.3.5.4 Timing control

The timing control register TCR (0x2) is used to define and control timing:

-Set the SETUP time of the nCS pin output before the transmission starts.

-Set the HOLD time of the nCS pin output after the end of transmission.

-Set the HIGH time of the Ncs pin output between successive commands.

Where the SETUP and HOLD times are configured in the ESMC_TCR.HOLD [3: 0] register and the HIGH time is configured in the ESMC_TCR.HIGH [3: 0] register. The time of SETUP and HOLD is the number of cycles in which the corresponding HCLK is inserted in units of HCLK. The time of HIGH is measured in units of SCLK, and the set value indicates the number of SCL cycles that the nCS pin needs to be held when it is HIGH.

Note: If the HIGH level time of the nCS pin exceeds the set HIGH time number, no new instructions are written and the nCS pin will not be pulled down.

30.3.5.5 ESMC command sending

By default, commands are sent in SINGLE mode at the beginning of each sequence. ESMC can also send specific commands in DUAL, QUAD, or OCTAL mode.

The ESMC has two command registers SFCR0 and SFCR1. The differences in the use of commands are as follows:

1. SFCR0 is used for single byte Flash command transmission, such as: RESET ENABLE (66h), RESET MEMORY (99h), etc. Write the command into the SFCR0 register, the command will be issued according to the current SPI mode setting, and once the command transfer is complete, the nCS pin will become inactive in order to execute the next command.

2. SFCR1 is used for non-single-byte Flash command transmission (including sequences such as ADDRESS/M [7: 0]/DUMMY/DATA).

30.3.5.6 ESMC command sequence configuration

A specific ESMC sequencecan be transmittedby configuringthe ESMC_SOCRregister. The following figure shows several Flash read/write operations supported by ESMC and their formats. As described below, configuring the control registers allows any combination of flash commands, including multi-SPI commands, addresses and data, SDR/DDR, and any number of DUMMY cycles.

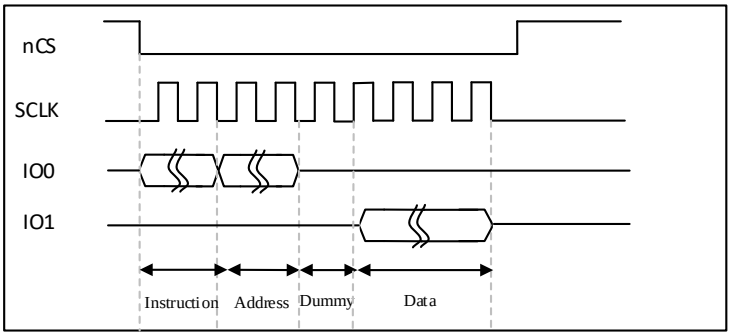


Figure 30-8 single line SPI readcommand: MULTI ADDR = 0; MULTI COMM = 0; SPI_MODE = 2 'b00; DUMMY! =0

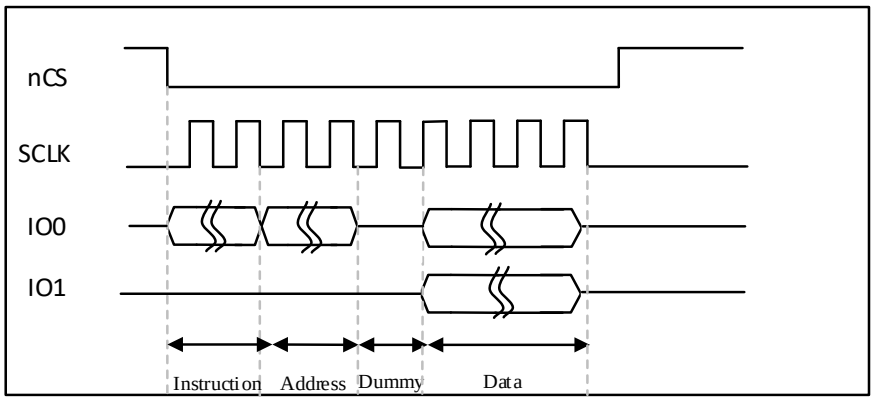


Figure 30-9 Two-line SPI read command: MULTI ADDR = 0; MULTI COMM = 0; SPI_MODE = 2 'b01; DUMMY! =0

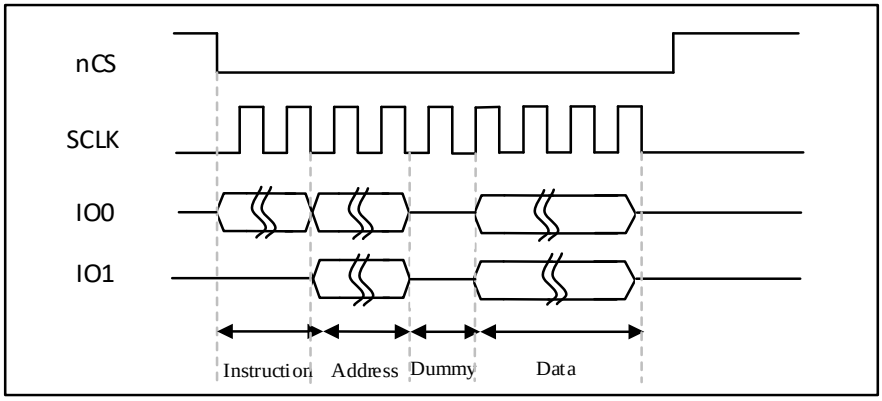


Figure 30-10 2 IO readcommand: MULTI COMM = 0; MULTI ADDR = 1; SPI_MODE = 2 'b01; DUMMY! = 0

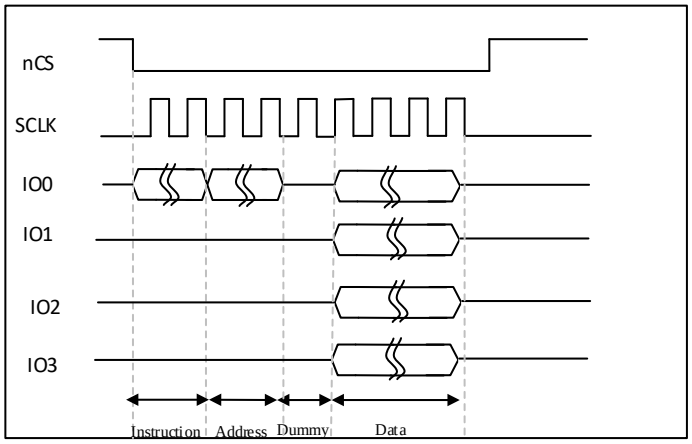


Figure 30-11 4-wire readcommand: MULTI ADDR = 0; MULTI COMM = 0; SPI_MODE = 2 'b10

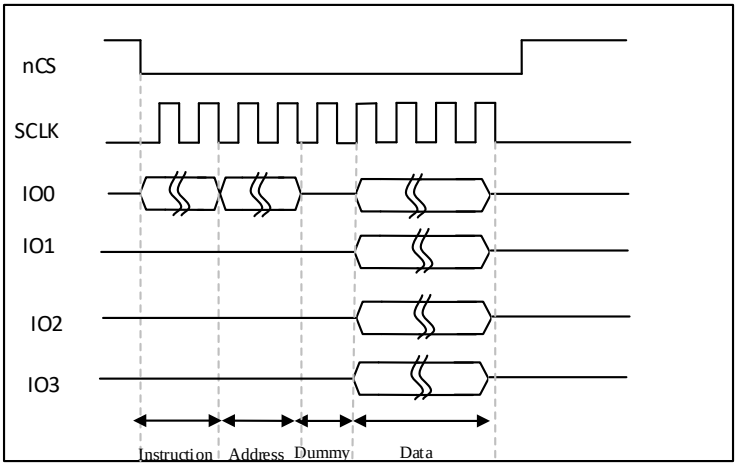


Figure 30-12 4 IO read command: MULTI COMM = 0; MULTI ADDR = 1; SPI_MODE = 2 'b10

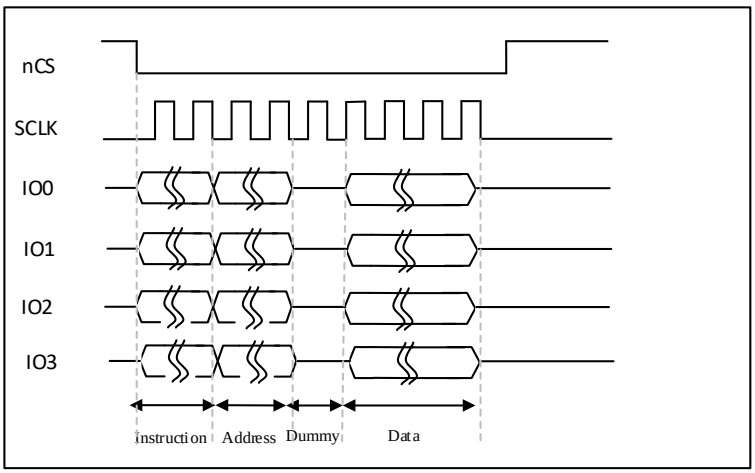


Figure 30-13 4 IO QPI mode read command: MULTI COMM = 1; MULTI ADDR = 1; SPI_MODE = 2 'b10

The difference between 8-line mode and DDR mode and the above is transmission on more IO lines and double edge sampling. Other sequence contents are similar to the above commands.

30.3.5.7 Dummy Cycle Control

Setthe number of Dummy cycles(bits [5: 0]) by ESMC _DCR.Dummy[5: 0], andthe maximum valueallowed to set is 63. During the transmit Dummy cycle, the IO pin is set to a high impedance state and no data is stored in the RX FIFO or read out.

The REC bit (7th bit of the DCR) is used to specify the type of operation. When REC = 1, ESMC performs a read operation on FLASH. When REC = 0, ESMC writes to FLASH. If the DATA has been received (or sent), the ESMC enters the WAIT state and waits for the DATA to be read or written from the FIFO.

NO_CMD (6th bit of DCR) is used to enable/disable sending instruction bytes to SPI Flash, andNO_ADDR(5thbit ofCR3) isused to enable/disable sending addresses. When the NO_CMD bit is set, the ESMC sends only the address and the set Dummy cycle, and then sends the data bytes.

In addition,the value of the Dummy cycle settingis also related tothe XSPI_DUAL_ LAT_OFF bit (the 7th bit of CR 2). When XSPI_DUAL_LAT_OFF = 1, the ESMC turns off double delay. When double delay is enabled when XSPI_DUAL_LAT_OFF = 0. This feature is only available in xSPI (setting CR3 [4] = 1, SOCR [1: 0] = 2 'h3, SOCR [7] = 1) mode.

Table 30-3 Number of DUMMY cycles in different modes

DCR [5: 0]	DUMMY Cycles		
	non-xSPI (CR3 [4] = 0)	xSPI OCTAL (CR2 [7] = 1, SOCR [7] = 0, CR3 [5] = 1, SOCR [1: 0] = 3)	xSPI OCTAL Double delay (CR2 [7]) = 0, SOCR [7] = 0, CR3 [5] = 1, SOCR [1: 0] = 3)
N	N	N-1	2N-1
0	0	0	0
1	1	-	-
2	2	1	3
3	3	2	5
4	4	3	7
5	5	4	9

63	63	62	125

30.3.5.8 Slave selection output control

The SSOCR control register can be read or written to at any time, and it is used to select which slave to drive during transfer. The contents of the SSOCR register are automatically assigned to SS5O-SS0O. This register allows SPI operations with up to 6 different SPI flash memories.

30.3.5.9 Size of FIFO

ESMC has 3 FIFOs: TXFIFO for sending data in indirect mode, RXFIFO for receiving data in indirect mode, and XIPFIFO for receiving data in XIP mode. The depths are 16 bytes, 16 bytes, and 8 bytes respectively.

30.3.5.10 Sampling extension

Throughthe ESMC_STR and ESMC_XSTR registers, it is possible to configure the number ofHCLKs that thesampling points of thedata phase ESMCextendrelative to the SCLK clock edge. By default, the value of these two registers is 0, and there is no delay between the sampling point and the SCLK. This feature is used with the following restrictions:

1. This function can only be used in SDR mode, not in DDR mode

2. The maximum value of the sample extension setting needs to be less than or equal to half of the SCLK period. For example, when ESMC_BAUD is set to 2, the maximum value of ESMC_STR and ESMC_XSTR registers can only be set to 0x1.

30.3.6 DMA mode

DMA transmission is possibleby enabling DMA_EN(ESMC_CR [3]). When ESMC is enabled, the hardware automatically generates a DMA send request or a DMA receive request signal. Before DMA operation, the ESMC should be initialized and the corresponding registers should be configured. In DMA mode, the ESMC generates a DMA send/receive request signal indicating that the ESMC expects data transfer, which will start once the ESMC pulls the nCS pin low and transmits the instruction and address to SPI Flash.

ESMC supports DMA burst read function (DMA burst write function is not supported). ESMC needs to set ESMC_IER [2] = 1 first. Aftersetting, ESMC will only send DMA requests after RXFIFO is full.

In addition, ESMC needs to pay attention to the following points when using DMA mode:

1. In DMA mode, the DMA module needs to be configured before transmission can begin.
2. In DMA mode, DMA_EN(ESMC_ CR [3])must be configuredto1 beforewriting the instruction.
3. In DMA receiving mode, it can only accept word reading, but does not support byte and half-word reading. In DMA sending mode, byte, half-word, and word writing are supported.
4. DMA operations can only be operated in register mode (not in XIP mode).
5. In DMA burst read mode, the amount of data transferred must be configured to be an integer multiple of a single burst data.

30.3.7 ESMC interrupts

The ESMC has an interrupt enabling GIE (ESMC_CR[4]) that enables/disables all interrupt sources in the ESMC.

The ESMC has four interrupt-related control registersESMC_ IER1,ESMC_ IER2,ESMC_ IFR1, and ESMC_ IFR2. Several interrupt sources of ESMC have their own masking bits regardless of the global

IE bits. The IE mask bits all have corresponding flag bits. When a specific event occurs, the flag bit is automatically set regardless of whether the corresponding IE bit is set. If you want togenerate an interrupt request, you mustset boththe global GIE (ESMC_ CR [4]) bit andthe correspondingIE (ESMC_ IER1 andESMC_ IER2)mask bits.

Interrupts can be generated in the following events:

Interrupt Event	Event Flag	Enable Control Bit
ADDR DONE	ADDRDONEIF	ADDRDONEIE
RX FIFO Word	RXFIFOWIF	RXFIFOWIE
DATA WAIT	DATAWAITIF	DATAWAITIE
IDLE	IDLE_DONEIF	Idle_DONEIE
RX FIFO FULL	RXFIFFIF	RXFIFFIE
RX FIFO HALF FULL	RXFIFOHFIF	RXFIFOHFIE
FIFO not Empty	RXFIFONEIF	RXFIFONEIE
COMMAND DONE	COMMANDDONEIF	COMMANDDONEIE
RXTIMEOUT	RXTOUTIF	-
XIP WE Instr set	XIPWEInstrsetIF	-
TX WORD OVER	TXWORDOVIF	TXWORDOVIE
TX FIFO OVER	TXFIFOOVIF	TXFIFOOVIE
TX FIFO full	TXFIFFIF	TXFIFFIE
TX FIFO HALF FULL	TXFIFOHFIF	TXFIFOHFIE
TX FIFO Empty	TXFIFOEIF	TXFIFOEIE
XIP Rd Instr Set	XIPRDInstrSetIF	-

Note: The following are the differences between registers with similar functions in ESMC_SR and ESMC_IFR1/2.

ESMC_SR. Difference between IDLE_STATE and ESMC_IFR.IDLE_DONE

The ESMC_SR.IDLE_STATE status register indicates that the ESMC's state machine has entered the IDLE state. While ESMC_IFR.IDLE_DONE represents an idle interrupt request, which refers to a transition from a non-idle state to an idle state. For example, at high frequency division, ESMC is idle. If a new commandis written, ESMC_IFR.IDLE_DONE will immediately change from 1 to 0. ESMC_SR.IDLE_STATE needs to change from 1 to 0 after the nCS pin ispulledup(after ESMC is not idle).

When using ESMC_SR.IDLE_STATE at high frequency division, it is necessary toread the ESMC_ IFR.CMDIF flagbit first aftersending the instruction, and then read the register after ESMC_ IFR.CMDIF is set.

ESMC_IFR2. TXFIFOOV and ESMC_SR.TXFIFOOV have the same function. ESMC_IFR.DATAWAIT and ESMC_SR. DATAWAIT has the same functionality.

The difference between these two sets of signals is: ESMC_IFR.DATAWAIT and ESMC_IFR2. ESMC_IFR2. TXFIFOOV is an interrupt request signal, ESMC_SR. DATAWAIT and ESMC_SR. TXFIFOOV is a status bit signal.

30.4 ESMC register description (byte access only)

ESMC register base address: 0xA000 1000

30.4.1 Configuration Register (ESMC_CR)

Address offset: 0x00

Reset value: 0xD2

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
CR	**0x00**	SPIEN	XIPEN	Res	GIE	DMAEN	2xQUAD	Res	SOFTRST	0xD2

Bit	Name	R/W	Reset Value	Function
7	SPIEN	RW	1	SPI enabled 0: SPI disabled, all internal registers and counters stopped 1: SPI enabled
6	XIPEN	RW	1	Enable/Disable XIP Enable Any change to this bit terminates the SPI operation, clears the FIFO, and drives the ESMC to an idle state. After clearing the XIPEN bit, you can only program commands and operations through indirect mode 0: Disable XIP function 1: Enable XIP function
5	Reserved	-	0	-
4	GIE	RW	1	Global interrupt enable 0: Turn off global interrupt enable 1: Turn on the global interrupt enable (the position 1 needs to be turned on while turning on the interrupt enable of each bit)
3	DMAEN	RW	0	DMA enabled, 0: DMA prohibition 1: DMA enabled
2	2xQUAD	RW	0	Dual Flash Mode 0: Single Flash mode 1: Dual Flash mode
1	Reserved	-	1	-
0	SOFTRST	RW	0	Software reset 0: Turn off software reset 1: Enable software reset function Note: When the software is reset, the ESMC_DATA register is the clear pointer (the data points to the first data in the RXFIFO), and the remaining registers are reset to their default values.

30.4.2 Configuration Register 2 (ESMC_CR2)

Address offset: 0x01

Reset value: 0x40

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
CR2	**0x01**	XSPI_Dual LAT_OFF	Res.	Res.	RXFIFO_CLR	TXFIFO_CLR	Res	CPOL	CPHA	0x40

Bit	Name	R/W	Reset Value	Function
7	XSPI_Dual LAT_OFF	RW	0	Double delay for xSPI mode DUMMY of RWDSI is turned off. 1: Disable the double delay function. In xSPI mode, the number of DUMMY CYCLE sent by ESMC is set by DCR [5: 0], and the number of cycles is N-1 0: Double delay function is enabled. In xSPI mode, the number of DUMMY CYCLE sent by ESMC is 2N-1
6	Reserved	-	1	-
5	Reserved	-	0	-
4	RXFIFO_CLR	RW	0	Clear RX FIFO Pointers
3	TXFIFO_CLR	RW	0	Clear TX FIFO Pointer
2	Reserved	-	0	-
1	CPOL	RW	0	Clock polarity 0: In idle state, SCLK remains low 1: SCLK maintains high level in idle state Note: CPOL and CPHA need to be used together, currently only 0x0 and 0x3 are supported When Mode 3 is enabled, it is recommended to enable the CPOL and CPHA registers first, and then configure the baud rate.
0	CPHA	RW	0	Clock phase 0: Sampling data from the first clock edge

				1: Start sampling data from second clock edge Note: CPOL and CPHA need to be used together, currently only 0x0 and 0x3 are supported When Mode 3 is enabled, it is recommended to enable the CPOL and CPHA registers first, and then configure the baud rate.

30.4.3 Timing Control Register (ESMC_TCR)

Address offset: 0x02

Reset value: 0x11

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
TCR	**0x02**	HIGH [3: 0]				HOLD [3: 0]				0x11

Bit	Name	R/W	Reset Value	Function
7: 4	HIGH [3: 0]	RW	4'h1	Number of cycles of HIGH time (in SCLK cycles): 1111: The number of cycles is 15 1110: The number of cycles is 14 …… 0010: The number of cycles is 2 0001: The number of cycles is 1 0000: The number of cycles is 1
3: 0	HOLD [3: 0]	RW	4'h1	Setup and hold time (in HCLK cycles): Build time: STR mode Mode0, DDR mode Mode0 and Mode3: 1111: 16 system clock cycles inserted 1110: 15 system clock cycles inserted …… 0010: Insert 3 system clock cycles 0001: Insert 2 system clock cycles 0000: Disabled STR mode Mode3: 1111: 15 system clock cycles inserted 1110: 14 system clock cycles inserted …… 0010: Insert 2 system clock cycles 0001: Insert 1 system clock cycle 0000: Disabled Hold time: (regardless of STR, DDR, Mode0, Mode3) 1111: 16 system clock cycles inserted 1110: 15 system clock cycles inserted …… 0010: Insert 3 system clock cycles 0001: Insert 2 system clock cycles 0000: Disabled

30.4.4 Baud rate register (ESMC_BAUD)

Address offset: 0x03

Reset value: 0x04

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
BAUD	**0x03**	BAUD [7: 0]								0x04

Bit	Name	R/W	Reset Value	Function
7: 0	BAUD [7: 0]	RW	8'h04	The baud rate register is used to define the SCLK frequency of the ESMC, divided based on the system clock. When BAUD is set to 0x0 and 0x1, the frequency is divided into two; That is, the fastest SCLK clock is the two-division frequency of the system clock. Other values are calculated as : SCLK = HCLK/BAUD;

30.4.5 SPI Flash Command Register (ESMC_SFCR)

Address offset: 0x04

Reset value: 0x0B

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
SFCR0 COMMAND ONLY	**0x04**	SFCR0 [7: 0]								0x0B

Bit	Name	R/W	Reset Value	Function
7: 0	SFCR0 [7: 0]	RW	8'h0B	SPI instructions (send instructions only) The instruction format contains only instruction bytes. Address, backup, Dummy, data bytes are not sent.

30.4.6 SPI output control register (ESMC_SOCR)

Address offset: 0x05

Reset value: 0x02

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
SOCR	**0x05**	DDR DATA	DDR ADDR	DDR COMM	MULTI ADDR	MULTI COMM	Send_M	SPI_MODE [1: 0]		0x02

Bit	Name	R/W	Reset Value	Function
7	DDRDATA	RW	0	Dual data rate operation is enabled during data reception/transmission. When this bit is set, data is moved in/out on both SCLK clock edges.
6	DDRADDR	RW	0	Dual data rate operation is enabled during flash address byte transmission. When this bit is set, the address shifts out on both SCLK clock edges.
5	DDRCMD	RW	0	Dual data rate operation is enabled during flash instruction byte transmission. When this bit is set, the instruction moves out on both SCLK clock edges.
4	MULTIADDR	RW	0	Enable multi-bit addresses When this bit is set, the address bytes and data bytes are sent in the same format. Where the address bits are sent in double, quadruple or octal format respectively, if MULTIADDR is clear, it means that the address is sent in single SPI mode.
3	MULTICMD	RW	0	Enable multi-bit instructions When this bit is set, the instruction byte and the data byte are sent in the same format. Where the instruction bits are sent in double, quad or octal format, respectively, and if MULTICMD is cleared, it means that the instruction is sent in single SPI mode.
2	SENM	RW	0	An additional M [7: 0] is transmitted after ADDR (8-bit, 24-bit, or 32-bit).
1: 0	SPIMODE [1: 0]	RW	2'h2	Set the SPI transport format: 2 'b00 SINGLE SPI 2 'b01 DUAL SPI 2 'b10 QUAD SPI 2 'b11 OCTAL SPI

30.4.7 Dummy Control Register (ESMC_DCR)

Address offset: 0x06

Reset value: 0x80

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
DCR	**0x06**	REC	NO_CMD	DUMMY [5: 0]						0x80

Bit	Name	R/W	Reset Value	Function
7	REC	RW	1	Continuous receive enable Once set, the ESMC should execute a flash read command and after sending the command, address, and DUMMY bytes, the ESMC starts receiving data bytes. If this bit is set, the ESMC receives data (read mode), otherwise the ESMC sends data (write mode).
6	NO_CMD	RW	0	When this bit is set, the instruction will be omitted and the transfer will begin with the address.
5: 0	DUMMY [5: 0]	RW	6'h0	Defines the number of DUMMY cycles generated after address transfer. The setting of DUMMY CYCLE is also related to the setting of some registers (ESMC_CR2 [7], ESMC_SOCR [7], ESMC_CR3 [5], ESMC_SOCR [1:

				0]). There will be three cases: non-xSPI mode, xSPI mode turns off double delay and xSPI enables double delay.

30.4.8 Configuration register 3 (ESMC_CR3)

Address offset: 0x07

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
CR3	**0x07**	No_DATA	Res	CMD_Ext	XSPI_RWDS	NO_ADDR	ADDR32BIT	ADDR16BIT	ADDR8BIT	0x00

Bit	Name	R/W	Reset Value	Function
7	No_DATA	RW	0	When the setting is enabled, the address will not enter the data wait state after sending. This bit can be used to erase the command, sending only the instruction and address.
6	Reserved	-	0	-
5	CMD_Ext	RW	0	When setting enabled, the ESMC sends extra bytes (contents of the M register) after sending the command.
4	XSPI_RWDS	RW	0	Supports the xSPI mode RWDS pin when setting enabled.
3	NO_ADDR	RW	0	When set to enable, ESMC does not send address bytes, instruction bytes followed by DUMMY cycles (if any) and data bytes.
2	ADDR32BIT	RW	0	Enable 32-bit addresses
1	ADDR16BIT	RW	0	Enable 16-bit addresses
0	ADDR8BIT	RW	0	Enable 8-bit addresses

30.4.9 Status Register (ESMC_SR)

Address offset: 0x14

Reset value: 0x10

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
SR	**0x14**	SPIF	DATAWAIT	TXFIFOOV	Idle_STATE	XIPBUSY	TXBUSY	RXBUSY	SSACTIVE	0x10

Bit	Name	R/W	Reset Value	Function
7	SPIF	R	0	SPI Interrupt Request Flag 0: No interrupt request pending 1: At least one interrupt request is pending. The flag will remain high until all interrupt requests are cleared.
6	DATAWAIT	R	0	The ESMC state machine is in a waiting state. If it is in a receiving state, there is data to be read. If it is in a sending state, there is data waiting to be sent 0: ESMC state machine is in non-waiting state 1: ESMC state machine is in waiting state
5	TXFIFOOV	R	0	TX_FIFOoverflowflag bit
4	Idle_STATE	R	1	The ESMC is at the IDLE status flag bit.
3	XIPBUSY	R	0	XIP operation BUSY flag bit
2	TXBUSY	R	0	Indirect mode write operation BUSY flag bit
1	RXBUSY	R	0	Indirect mode read operation BUSY flag bit
0	SSACTIVE	R	0	Slave selects valid flag If any nCS output is valid, this position is 1

30.4.10 Status register 2 (ESMC_SR2)

Address offset: 0x15

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
SR1	**0x15**	Res.	Res.	Res.	Res.	Res.	Res.	xSPI DUAL LATENCY	RWDSI_ND	0x00

Bit	Name	R/W	Reset Value	Function
7: 2	Reserved	-	-	-
1	XSPI DUAL LATENCY	RC_W1	0	This bit indicates that double latency for xSPI is enabled. The bit can be cleared by writing 1 to the bit.

0	RWDSI_ND	R	0	Indicates that the Rwdsi clock was not detected. This flag is set if the rwdsi clock is not detected in the xSPI read operation, or rwdsi is not detected for more than 1 SCLK cycle.

30.4.11 RX_FIFO status register (ESMC_RXSTAT)

Address offset: 0x16

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
RXSTAT	**0x16**	Res.	Res.	Res.	Res.	RXSTAT [3: 0]				0x00

Bit	Name	R/W	Reset Value	Function
7: 4	Reserved	-	-	-
0: 3	RXSTAT	R	0	This register shows the number of bytes of data stored in the RX FIFO 1111: 15 numbers available for reading in RXFIFO 1110: 14 numbers available for reading in RXFIFO ..... 0002: There are 2 numbers available for reading in RXFIFO 0001: There is 1 number available for reading in RXFIFO 0000: No data available to read in RXFIFO

30.4.12 Interrupt Flag Register (ESMC_IFR1)

Address offset: 0x18

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
IFR	**0x18**	ADDRDONEIF	RXFIFOWIF	DATAWAITIF	IDLEDONEIF	RXFIFFIF	RXFIFOHFIF	RXFIFONEIF	CMDIF	0x00

Bit	Name	R/W	Reset Value	Function
7	ADDRDONEIF	RC_W1	0	Address Send Complete Flag. When the address field is issued, the flag position bit. This flag bit is automatically cleared by writing a new command to the SFCR register or writing 1.
6	RXFIFOWIF	R	0	This bit indicates that the RX FIFO contains at least 1 word of data. This means that there are at least 4 bytes of data stored in the RX FIFO. Note: Users should avoid reading data when there is less than one word in the RXFIFO. This causes incorrect data to be read.
5	DATAWAITIF	R	0	This bit indicates that the ESMC is currently performing flash read and write operations, and the ESMC state machine is in a waiting state, which means: During the write operation, the ESMC is waiting to send the next portion of data bytes, and the FIFO is empty. During the read operation, the ESMC is waiting to receive the next portion of data bytes, and the FIFO is full. After the FIFO is written or emptied, this bit is cleared so that the ESMC can resume the current flash read/write operation. 0: The data waiting phase has not been entered, and the current transmission has not ended 1: Enter the data waiting phase, and the current transmission has ended
4	IDLEDONEIF	R	0	Idle interrupt flag. 0: Instruction, Address, Dummy or Data Transfer is in progress. 1: ESMC completes the operation and enters the IDLE state.
3	RXFIFFIF	R	0	RX_FIFO Full 0: There is at least 1 space available for writing in the RXFIFO. 1: When the RXFIFO is fully full, there is no free space in the FIFO memory.
2	RXFIFOHFF	R	0	RX_FIFO half full This interrupt is triggered when the RX FIFO is half full (the data of the RXFIFO is 8).
1	RXFIFONEIF	R	0	FIFO is not empty 0: No data in RXFIFO

				1: RXFIFO contains at least 1 byte of data
0	CMDIF	RC_W1	0	Instruction Send Complete Flag 0: Clear when the SFCR register loads a new instruction or writes a 1 to the bit 1: Set when the instruction byte is sent completely.

30.4.13 Interrupt enable register (ESMC_IER1)

Address offset: 0x19

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
IER	**0x19**	ADDRDONEIE	RXFIFOWORDIE	DATAWAITIE	IDLEDONEIE	RXFIFFIE	RXFIFOHFIE	RXFIFONEIE	CMDIE	0x00

Bit	Name	R/W	Reset Value	Function
7	ADDRDONEIE	RW	0	Address Send Complete Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
6	RXFIFOWIE	RW	0	RX FIFO Receive Word Unit Data Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
5	DATAWAITIE	RW	0	DATAWAIT interrupt enable 0: Interrupt prohibition 1: Interrupt enable
4	IDLEDONEIE	RW	0	Idle interrupt enable 0: Interrupt prohibition 1: Interrupt enable
3	RXFIFFIE	RW	0	RX FIFO Full Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
2	RXFIFOHFIE	RW	0	RX FIFO Half Full Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
1	RXFIFONEIE	RW	0	FIFO non-null interrupt enable 0: Interrupt prohibition 1: Interrupt enable
0	CMDIE	RW	0	Instruction Send Complete Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable

30.4.14 Interrupt Flag Register (ESMC_IFR2)

Address offset: 0x1A

Reset value: 0x06

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
IFR2	**0x1A**	RXTOUTIF	XIPWEINSTRSETIF	TXWORDOVIF	TXFIFOOVIF	TXFIFOFULLIF	TXFIFOHEIF	TXFIFOEIF	XIPRDINSTRSETIF	0x06

Bit	Name	R/W	Reset Value	Function
7	RXTOUTIF	R	0	Receive data timeout flag If the RXFIFOF or RXFIFOHF interrupt is enabled, the RX TIMEOUT interrupt is also enabled. If the RXFIFO is not empty and no new data is written to the RXFIFO and no words are read within the time specified in the RXTOUT register, a timeout interrupt is generated.
6	XIPWEINSTRSETIF	R	0	Indicates that the bus is in XIP write access.
5	TXWORDOVIF	RC_W1	0	This bit indicates that the data to be written to the TX FIFO is not word-aligned. This bit is set when software attempts to write more than 4 bytes of data into 4 bytes of memory space. (For example, write a single byte first and then write 4 bytes of data.) Equivalent to writing 5 bytes to 4 bytes of memory space)

				Notes: 1. Data loaded into the TX FIFO should always be aligned with word. 2. The method of cross-writing of words and bytes is not supported. 3. This bit is only set when the TXFIFO is not full. The rebit can be cleared by writing a 1 to the bit.
4	TXFIFOOVIF	RC_W1	0	TX FIFO Overflow flag This bit is set to 1 when the software attempts to write a new data to a full TX FIFO memory. When the bit is set to 1, it indicates that the TXFIFO has over-flowed, and the bit can be cleared by writing 1 to the bit.
3	TXFIFFIF	R	0	TXFIFO Full Flag 0: There is at least 1 empty space in the FIFO. 1: There is no free space in FIFO memory.
2	TXFIFOHEIF	R	1	TXFIFO Half-Empty Flag This interrupt will be triggered when TX FIFO is half empty (when the data of TXFIFO is 8).
1	TXFIFOEIF	R	1	TXFIFO empty 0: At least one byte of data in TXFIFO 1: TXFIFO is completely empty. Data can be written to fill TXFIFO.
0	XIPRDINSTRSETIF	R	0	Indicates that the bus is in XIP read access.

30.4.15 Interrupt enable register (ESMC_IER2)

Address offset: 0x1B

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
IER2	**0x1B**	Res	Res	TXWORDOVIE	TXFIFOOVIE	TXFIFFIE	TXFIFOHEIE	TXFIFOEIE	Res	0x00

Bit	Name	R/W	Reset Value	Function
7: 6	Reserved	-	-	-
5	TXWORDOVIE	RW	0	Send word overflow interrupt enable 0: Interrupt prohibition 1: Interrupt enable
4	TXFIFOOVIE	RW	0	TX FIFO Overflow Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
3	TXFIFFIE	RW	0	TX FIFO Full Interrupt Enable 0: Interrupt prohibition 1: Interrupt enable
2	TXFIFOHEIE	RW	0	TX FIFO mid-air interrupt enable 0: Interrupt prohibition 1: Interrupt enable
1	TXFIFOEIE	RW	0	TX FIFO null interrupt enable 0: Interrupt prohibition 1: Interrupt enable
0	Reserved	-	-	-

30.4.16 Sample extension register (ESMC_STRR)

Address offset: 0x1C

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
STR	**0x1C**	Res	Res	SMP [5: 0]						0x00

Bit	Name	R/W	Reset Value	Function
7: 6	Reserved	-	-	-
5: 0	SMP [5: 0]	RW	6'h0	This bit defines the number of HCLKs that the sampling points of the configurable data phase ESMC extend relative to the SCLK clock edge. 111111: The number of cycles is 63 111110: The number of cycles is 62 …… 011111: The number of cycles is 31 ……

				000010: The number of cycles is 2 000001: The number of cycles is 1 000000: No sampling point delay. The data is sampled at the SCLK clock edge

30.4.17 Receive timeout register (ESMC_RXTOUT)

Address offset: 0x30

Reset value: 0x3F

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
RXTOUT	**0x30**	RXTOUT [7: 0]								0x3F

Bit	Name	R/W	Reset Value	Function
7: 0	RXTOUT	RW	8'h3F	This register is used to set the time period of RX TIMEOUT (in SCLK cycles).

30.5 XIP Mode Control Register-Read Operation

30.5.1 XIP SPI Flash Command Register (ESMC_XSFCR)

Address offset: 0x20

Reset value: 0x0B

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSFCR	**0x20**	XSFCR [7: 0]								0x0B

Bit	Name	R/W	Reset Value	Function
7: 0	XINS [7: 0]	RW	8 'hB	Instructions in XIP mode

30.5.2 XIP SPI output control register (ESMC_XSOCR)

Address offset: 0x21

Reset value: 0x02

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Re-set
XSOCR	**0x21**	DDR DATA	DDR ADDR	DDR COMM	MULTI ADDR	MULTI COMM	Send_M	SPI_MODE [1: 0]		0x02

Bit	Name	R/W	Reset Value	Function
7	DDRDATA	RW	0	Dual data rate operation is enabled during data reception/transmission. When this bit is set, data is moved in/out on both SCLK clock edges.
6	DDRADDR	RW	0	Dual data rate operation is enabled during flash address byte transmission. When this bit is set, the address shifts out on both SCLK clock edges.
5	DDRCMD	RW	0	Dual data rate operation is enabled during flash instruction byte transmission. When this bit is set, the instruction moves out on both SCLK clock edges.
4	MULTIADDR	RW	0	Enable multi-bit addresses When this bit is set, the address bytes and data bytes are sent in the same format. Where the address bits are sent in double, quadruple or octal format respectively, if MULTIADDR is clear, it means that the address is sent in single SPI mode.
3	MULTICMD	RW	0	Enable multi-bit instructions

				When this bit is set, the instruction byte and the data byte are sent in the same format. Where the instruction bits are sent in double, quad or octal format, respectively, and if MULTICMD is cleared, it means that the instruction is sent in single SPI mode.
2	SENM	RW	0	An additional M [7: 0] is transmitted after ADDR (8-bit, 24-bit, or 32-bit).
1: 0	SPIMODE [1: 0]	RW	2'h2	Set the SPI transport format: 2 'b00 SINGLE SPI 2 'b01 DUAL SPI 2 'b10 QUAD SPI 2 'b11 OCTAL SPI

30.5.3 XIPReadDummy Control Register (ESMC_XDCR)

Address offset: 0x22

Reset value: 0x80

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XDCR	**0x22**	RES	NO_CMD	DUMMY [5: 0]						0x80

Bit	Name	R/W	Reset Value	Function
7	Reserved	-	1	-
6	NO_CMD	RW	0	When this bit is set, the instruction will be omitted and the transfer will begin with the address.
5: 0	DUMMY [5: 0]	RW	6'h0	Defines the number of DUMMY cycles generated after address transfer. The setting of DUMMY CYCLE is also related to the setting of some registers (ESMC_XCR2 [7], ESMC_XSOCR [7], ESMC_XCR3 [5], ESMC_XSOCR [1: 0]). There will be three cases: non-xSPI mode, xSPI mode turns off double delay and xSPI enables double delay.

30.5.4 XIP configuration register 3 (ESMC_XCR3)

Address offset: 0x23

Reset value: 0x00

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XCR3	**0x23**	No_DATA	Res	CMD_Ext	XSPI_RWDS	NO_ADDR	ADDR32BIT	ADDR16BIT	ADDR8BIT	0x00

Bit	Name	R/W	Reset Value	Function
7	No_DATA	RW	0	When the setting is enabled, the address will not enter the data wait state after sending. This bit can be used to erase the command, sending only the instruction and address.
6	Reserved	-	0	-
5	CMD_Ext	RW	0	When setting enabled, the ESMC sends extra bytes (contents of the M register) after sending the command.
4	XSPI_RWDS	RW	0	Supports the xSPI mode RWDS pin when setting enabled.
3	NO_ADDR	RW	0	When set to enable, ESMC does not send address bytes, instruction bytes followed by DUMMY cycles (if any) and data bytes.
2	ADDR32BIT	RW	0	Enable 32-bit addresses
1	ADDR16BIT	RW	0	Enable 16-bit addresses
0	ADDR8BIT	RW	0	Enable 8-bit addresses

30.6 XIP mode control general purpose register

30.6.1 XIP Mode Register (ESMC_XMODE)

Address offset: 0x24

Reset value: 0x00

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XMODE	**0x24**	M [7: 0]								0x00

Bit	Name	R/W	Reset Value	Function
7: 0	M [7: 0]	RW	8'h0	The contents of the ESMC_ XCR3 register are sent after the address byte only ifthe SEND_M bit is set in this register.

30.6.2 XIP write mode register (ESMC_XMODE_WE)

Address offset: 0x25

Reset value: 0x00

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XMODE_We	**0x25**	M [7: 0]								0x00

Bit	Name	R/W	Reset Value	Function
7: 0	M [7: 0]	RW	8'h0	The contents of the ESMC_XCR3_WE register are sent after the address byte only ifthe SEND_M bit is set in this register.

30.6.3 XIP slave select output control register (ESMC_XSSOCR)

Address offset: 0x26

Reset value: 0x01

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSSOCR	**0x26**	Res	SS_CLR_RQ	SS [5: 0]						0x01

Bit	Name	R/W	Reset Value	Function
7	Reserved	-	-	-
6	SS_CLR_RQ	RW	0	Clear slave select output request Writing a 1 to this bit will change the slave select output to an inactive high state. The current operation will be terminated immediately, the ESMC will enter the IDLE state, and the nCS pin will be deactivated. This bit will be automatically cleared after the operation is completed.
5: 0	SS [5: 0]	RW	6'h1	Slave Select Enable/Disable 0: The corresponding SPI slave is disabled and the nCS pin is forced inactive. 1: Enable the corresponding SPI slave, perform SPI operation, and activate the nCS pin during SPI transfer

30.6.4 XIP slave select output control register (ESMC_XSSOCR_WE)

Address offset: 0x27

Reset value: 0x01

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSSOCR	**0x27**	Res	SS_CLR_RQ	SS [5: 0]						0x01

Bit	Name	R/W	Reset Value	Function
7	Reserved	-	-	-
6	SS_CLR_RQ	RW	0	Clear slave select output request

				Writing a 1 to this bit will change the slave select output to an inactive high state. The current operation will be terminated immediately, the ESMC will enter the IDLE state, and the nCS pin will be deactivated. This bit will be automatically cleared after the operation is completed.
5: 0	SS [5: 0]	RW	6'h1	Slave Select Enable/Disable 0: The corresponding SPI slave is disabled and the nCS pin is forced inactive. 1: Enable the corresponding SPI slave, perform SPI operation, and activate the nCS pin during SPI transfer

30.6.5 XIP mode sample stretch register (ESMC_XSTRR)

Address offset: 0x31

Reset value: 0x00

This register supports soft_rst reset

Address offset: 0x1C

Reset value: 0x00

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSTRR	**0x31**	Res	Res	SMP [5: 0]						0x00

Bit	Name	R/W	Reset Value	Function
7: 6	Reserved	-	-	-
5: 0	SMP [5: 0]	RW	6'h0	This bit defines the number of HCLKs that the sampling points of the configurable data phase ESMC extend relative to the SCLK clock edge. 111111: The number of cycles is 63 111110: The number of cycles is 62 …… 011111: The number of cycles is 31 …… 000010: The number of cycles is 2 000001: The number of cycles is 1 000000: No sampling point delay. The data is sampled at the SCLK clock edge

30.7 XIP Mode Control Register-Write operation

30.7.1 XIP Mode Write SPI flash command register (ESMC_XSFCR_WE)

Address offset: 0x28

Reset value: 0x32

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSFCR_we	**0x28**	XSFCR [7: 0]								0x32

Bit	Name	R/W	Reset Value	Function
7: 0	XSFCR_WE [7: 0]	RW	8'h32	Instructions in XIP read mode

30.7.2 XIP Mode Read SPI Output Control Register (ESMC_XSOCR_WE)

Address offset: 0x29

Reset value: 0x02

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XSOCR_We	**0x29**	DDR DATA	DDR ADDR	DDR COMM	MULTI ADDR	MULTI COMM	Send_M	SPI_MODE [1: 0]		0x02

Bit	Name	R/W	Reset Value	Function
7	DDRDATA	RW	0	Dual data rate operation is enabled during data reception/transmission. When this bit is set, data is moved in/out on both SCLK clock edges.
6	DDRADDR	RW	0	Dual data rate operation is enabled during flash address byte transmission. When this bit is set, the address shifts out on both SCLK clock edges.
5	DDRCMD	RW	0	Dual data rate operation is enabled during flash instruction byte transmission. When this bit is set, the instruction moves out on both SCLK clock edges.
4	MULTIADDR	RW	0	Enable multi-bit addresses When this bit is set, the address bytes and data bytes are sent in the same format. Where the address bits are sent in double, quadruple or octal format respectively, if MULTIADDR is clear, it means that the address is sent in single SPI mode.
3	MULTICMD	RW	0	Enable multi-bit instructions When this bit is set, the instruction byte and the data byte are sent in the same format. Where the instruction bits are sent in double, quad or octal format, respectively, and if MULTICMD is cleared, it means that the instruction is sent in single SPI mode.
2	SENM	RW	0	An additional M [7: 0] is transmitted after ADDR (8-bit, 24-bit, or 32-bit).
1: 0	SPIMODE [1: 0]	RW	2'h2	Set the SPI transport format: 2 'b00 SINGLE SPI 2 'b01 DUAL SPI 2 'b10 QUAD SPI 2 'b11 OCTAL SPI

30.7.3 XIPwriteDummy cycle control register (ESMC_XDCR_WE)

Address offset: 0x2A

Reset value: 0x00

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XDCR_we	**0x2A**	RES	NO_CMD	DUMMY [5: 0]						0x00

Bit	Name	R/W	Reset Value	Function
7	Reserved	-	0	-
6	NO_CMD	RW	0	When this bit is set, the instruction will be omitted and the transfer will begin with the address.
5: 0	DUMMY [5: 0]	RW	6'h0	Defines the number of DUMMY cycles generated after address transfer. The setting of DUMMY CYCLE is also related to the setting of some registers (ESMC_XCR2_WE [7], ESMC_XSOCR_WE [7], ESMC_XCR3_WE [5], ESMC_XSOCR_WE [1: 0]). There will be three cases: non-xSPI mode, xSPI mode turns off double delay and xSPI enables double delay.

30.7.4 XIP write configuration register 3 (ESMC_XCR3_WE)

Address offset: 0x2B

Reset value: 0x00

This register supports soft_rst reset

Register	ADDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Reset
XCR3_WE	**0x2B**	No_DATA	Res	CMD_Ext	XSPI_RWDS	NO_ADDR	ADDR32BIT	ADDR16BIT	ADDR8BIT	0x00

Bit	Name	R/W	Reset Value	Function
7	No_DATA	RW	0	When the setting is enabled, the address will not enter the data wait state after sending.

				This bit can be used to erase the command, sending only the instruction and address.
6	Res		0	
5	CMD_Ext	RW	0	When setting enabled, the ESMC sends extra bytes (contents of the M register) after sending the command.
4	XSPI_RWDS	RW	0	Supports the xSPI mode RWDS pin when setting enabled.
3	NO_ADDR	RW	0	When set to enable, ESMC does not send address bytes, instruction bytes followed by DUMMY cycles (if any) and data bytes.
2	ADDR32BIT	RW	0	Enable 32-bit addresses
1	ADDR16BIT	RW	0	Enable 16-bit addresses
0	ADDR8BIT	RW	0	Enable 8-bit addresses

30.8 ESMC Register Description

ESMC register base address: 0xA000 1000

30.8.1 24-bit address register (ESMC_ADDR24)

Address offset: 0x08

Reset value: 0x0000000B

This register can only be accessed word-wise (32-bit)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR [23: 8]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR [7: 0]								SFCR [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	ADDR [23: 0]	RW	24'h0	24-bit address Use a 24-bit address when ESMC_CR3 [2] = 0 (default 24-bit address mode) The address register values are transmitted in the currently selected SPI mode (extended SPI, single SPI, dual SPI, quad SPI, or eight SPI) channel. The address register should be written before the ESMC loads the ESMC _SFCR register.
7: 0	SFCR [7: 0]	RW	8'h0B	SPI instruction settings

30.8.2 32-bit address register (ESMC_ADDR32)

Address offset: 0x0C

Reset value: 0x0001 0000

This register can only be accessed word-wise (32-bit)

SS0 ~ 1 is used to configure which slave select output should be driven during the SPI master transfer. The contents of the registers are automatically assigned to SS1O-SS0O. This register allows easy SPI operation with up to 2 different SPI flash memories.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.								SSSET	SS CLR	SS [5: 0]					
-								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MREG [7: 0]								ADDR [31: 24]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	SSSET	RW	0	Slave Select Output Setup Request Writing 1 to this bit causes the SS [5: 0] selected slave select output (nCS pin) to be driven. This bit will be cleared automatically once the nCS pin output is active.
22	SSCLR	RW	0	Clear slave select output request

				Writing a 1 to this bit will change the slave select output to an inactive high state. The current operation will be terminated immediately, the ESMC will enter the IDLE state, and the nCS pin will be deactivated. This bit will be automatically cleared after the operation is completed.
21: 16	SS [5: 0]	RW	6'h1	Slave Select Enable/Disable 0: The corresponding SPI slave is disabled and the nCS pin is forced inactive. 1: Enable the corresponding SPI slave, perform SPI operation, and activate the nCS pin during SPI transfer
15: 8	MREG	RW	8'h0	Only when the SEND_M bit is set in the CR3 register will the contents of this register be sent after the address byte.
7: 0	ADDR [31: 24]	RW	8'h0	32-bit address When ESMC_CR3 [2] =1(32-bit address enabled), 32-bit address is used

30.8.3 Data Register (ESMC_DATA)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DATA [31: 0]	RW	32'h0	Data to be received/transmitted by ESMC A data buffer for data transfer during a flash write operation, a data buffer for data storage during a read command. The data can be used in both directions. When the data buffer memory is enabled, these registers all point to the top of the FIFO in the read-write direction.

30.8.4 Transfer Count Register (ESMC_BCR)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BCR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BCR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	BCR [31: 0]	RW	32'h0	Number of data read and written ESMC supports fixed-length data transfer, and by default ESMC will read/write data from external memory as long as the RXFIFO/TXFIFO has at least one free space/at least one data. Once the storage space of the RXFIFO/TXFIFO is full/empty, the ESMC will enter the WAIT state and WAIT for the host to read data from the RXFIFO/TXFIFO or write data. The RXFIFO must be completely empty or new data must be written to the TXFIFO before the next transfer can be resumed.After the ESMC reads the specified number of bytes, the nCS pin is deactivated and the ESMC goes idle, waiting for the next command to be executed.This register allows user commands to receive 232-1 bytes of data.

31. SDIO Interface (SDIO)

31.1 Introduction

The SD/SDIO MMC card host module provides an operation interface between the AHB peripheral bus and the multimedia card (MMC), the SD memory card, the SDIO card and the CE-ATA device.

The MMC system specification sheet is published by the MMCA Technical Committee and is available on the Multimedia Card Association's website.

CE-ATA System Specifications are available on the CE-ATA Working Group's website.

The SDIO features include the following:

- Full compliance with SD Memory Card Specifications Version 2.0
- Full compliance with SD I/O Card Specification Version 2.0
- Full compliance with MultiMediaCard System Specification Version 4.2
- Fully compliance with the CE-ATA digital protocol Rev1.1
- Supports sending commands and requesting interrupts from the host processor

Notes:

SDIO does not support communication mode of SPI mode

It only supports the I/O part of SD card or composite card in I/O mode, and does not support all required commands in SD storage devices. There are some commands here that do not work in the SD I/O card, such as the erase command, so they are not supported by SDIO. In addition, several commands are different between SD memory cards and SD I/O cards and thus are not supported in the SDIO.

SDIO consists of 2 parts:

- SDIO adapter module: realizes all related functions of MMC/SD/SD I/O card, such as clock generation, command and data transmission.
- AHB bus interface: operates the registers in the SDIO adapter module and generates interrupt and DMA request signals.

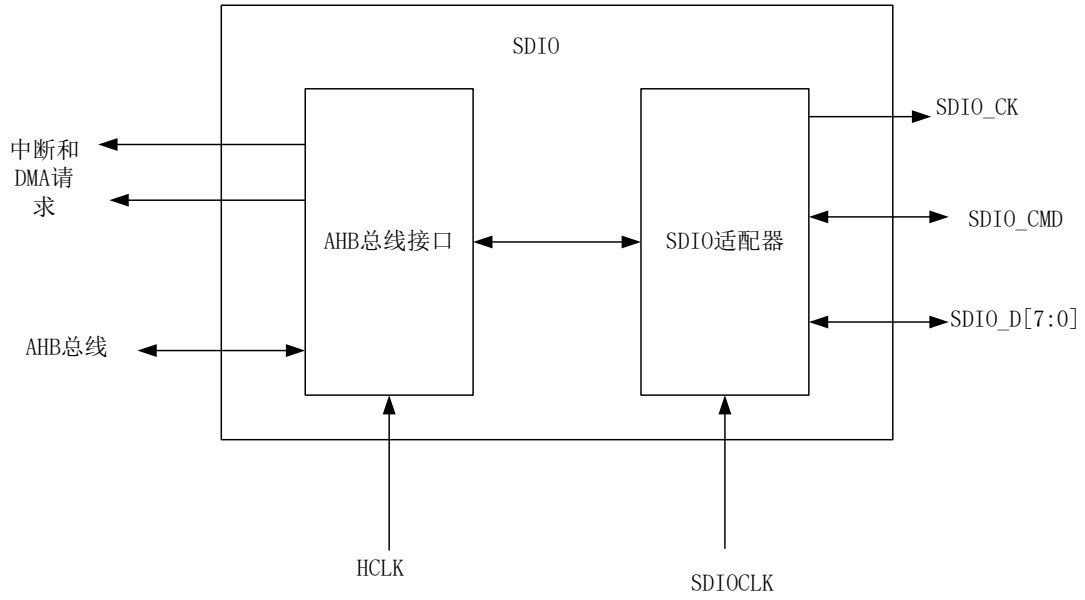


Figure 31-1 SDIO block diagram

SDIO_D [0] is used for data transfer by default after reset. After initialization, the host can change the width of the data bus. If an MMC is connected to the bus, SDIO_D [0], SDIO_D [3: 0], or SDIO_D [7: 0] may be used for data transmission.

If an SD or SD I/O card is connected to the bus, SDIO_D [0] or SDIO_D [3: 0] can be used for data transmission through the host configuration. All data lines operate in push-pull mode.

SDIO_CMD has two modes of operation:

- Open mode for initialization (only for MMC version V3.31 or earlier)
- Push-pull mode for command transmission (SD/SD I/O cards and MMC V4.2 also use push-pull drive at initialization).

The following table applies to the MMC/SD/SD I/O card bus:

Table 31-1 SDIO Pin Definition

Pin	Direction	Description
SDIO_CK	Output	MMC/SD/SDIO card clock. This is the clock line from the host to the card
SDIO_CMD	bidirectional	MMC/SD/SDIO card commands. This is a bidirectional command/response signal line
SDIO_D [7: 0]	bidirectional	MMC/SD/SDIO card data. These are bidirectional data buses.

31.2 SDIO adapter

The SDIO adapter includes a control unit, a command unit, and a data unit that can generate signals to the card. A specific description of these signals is as follows:

SDIO_CLK: The clock supplied to the card by the SDIO controller. Each clock cycle sends one bit of command or data directly on the command line (SDIO_CMD) and all data lines (SDIO_D). The SDIO_CLK frequency may be between 0 MHz and 20 MHz for MMC card version V3.31, between 0 MHz and 48 MHz for MMC card version V4.2, and between 0 MHz and 25 MHz for SD or SD I/O cards

SDIO_CMD: This signal is a bidirectional command channel used for initialization of the card and transmission of commands. Commands are sent from the SDIO controller to the card and responses are sent from the card to the host.

SDIO_D [7: 0]: These signal lines are all bidirectional data channels. The data signal line operates in push-pull mode. Only the card or host drives these signals at a time. By default, only D0 is used for data transmission after power on or reset. The SDIO adapter can be configured with a wider data bus for data transfer, using D0-D3 or D0-D7 (only for MMCV4.2). The SDIO has an internal pull-up to the data signal lines D1-D7. After entering 4-bit mode, the card disconnects the internal pull-ups of D1 and D2 (the internal pull-up of D3 remains unchanged due to the use of CS slice selection in SPI mode). Accordingly, after entering 8-bit mode, the internal pull-ups of D1, D2, and D4-D7 are disconnected (the internal pull-up of D3 remains unchanged due to the use of CS slice selection in SPI mode).

The SDIO adapter is the interface of SD/SD I/O/MMC/CE-ATA and consists of 3 sub-units:

31.2.1 Control unit

The control unit includes a power management function and a clock management function for outputting a clock. Power management is controlled by the SDIO_PWRCTL register to realize power-down and power-on of the power supply. The power saving mode is configured by setting the PWRSAV bit of SDIO_CLKCR, so that when the bus is idle, SDIO_CLK is turned off. Clock management generates an SDIO_CLK clock signal to the card.

31.2.2 Command unit

The command unit implements sending and receiving commands to the card. The data transmission flow is controlled by a command state machine (CSM). After a write operation is performed to the SDIO_CMD register and the start_cmd bit of the register is set to 1, the command transfer begins. First, a command is sent to the card. This command contains 48 bits and is issued through the SDIO_CMD line. Each SDIO_CLK sends one bit of data. This 48-bit command contains a 1-bit start bit, a 1-bit transfer bit, a 6-bit command index (defined by the cmd_index bit of the SDIO_CMD register), a 32-bit argument (defined by the SDIO_CMDARG), a 7-bit CRC, and a 1-bit stop bit. The response from the card is then received, which is divided into a 48-bit short response and a 136-bit long response, both of which are stored in the SDIO_RESP0-SDIO_RESP3 registers.

- **Command channel state machine (CPSM)**

When the command register is written and the enable bit is set, the command is sent. When the command is sent, the Command Channel State Machine (CPSM) sets the status flag and enters the idle state when no response is required (see figure below). When the response is received, the received CRC code will be compared with the internally generated CRC code, and then the corresponding status flag will be set.

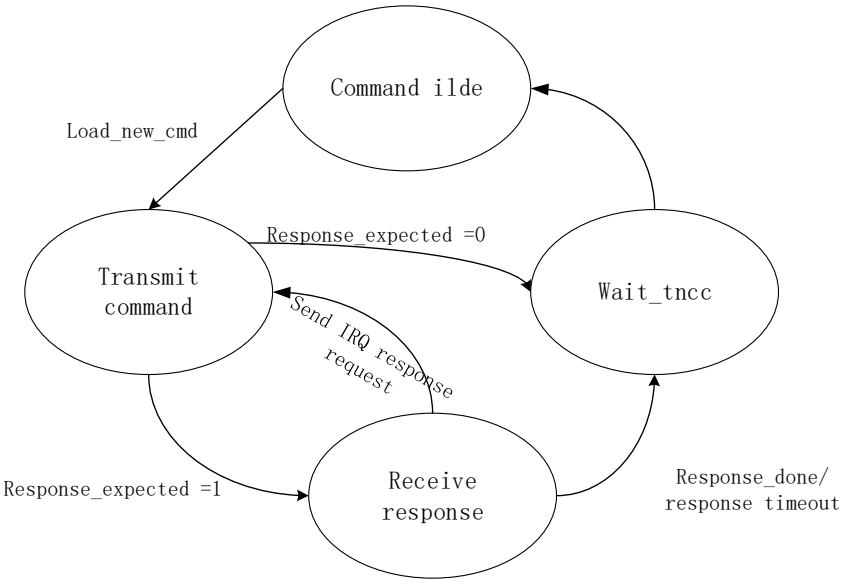


Figure 31-2 SD_MMC command channel state machine (CPSM)

The command channel state machine performs the following functions based on the CMD register bit values:

1. send_initialization-Sends an initialization sequence of 80 clocks before sending a command.

2. response_expected-The expected response to the command. After the command is issued, the command channel state machine receives a 48-bit or 136-bit response and sends it to the SDIO adapter. If the start bit of the card response is not received within the number of clocks programmed in the timeout register, the response timeout and command completion bits are set in the interrupt status register. If the response expected bit is not set, the command complete bit is set in the interrupt status register after the command is issued.

3. response_length-if this bit is set, a 136-bit response is received; If not set, a 48-bit response is received.

4. check_response_crc-If this bit is set, the command channel compares the CRC7 received in the response with the internally generated CRC7. If the two do not match, the response CRC error signal is sent to the bus interface; Set the response CRC error bit in the interrupt status register.

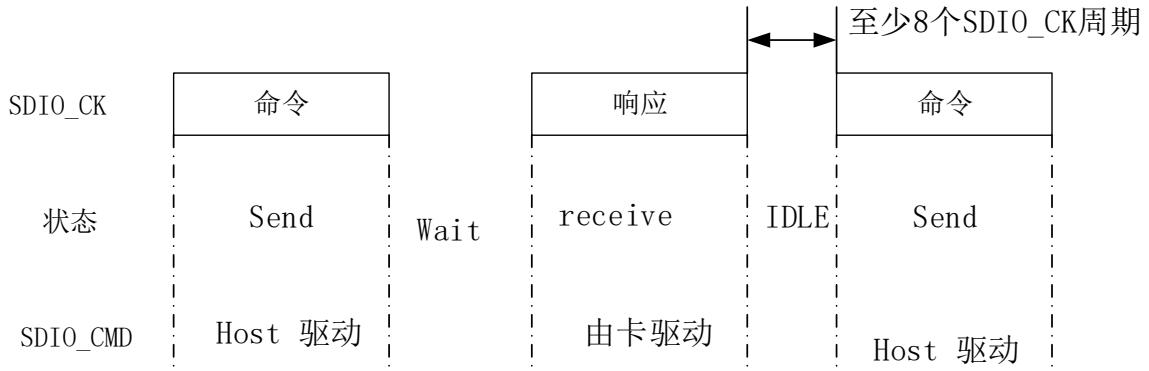


Figure 31-3 SDIO command transfer

- **Command format**

Command: A command is used to start an operation. The host issues addressed commands or broadcast commands to a specified card or all cards (broadcast commands are only suitable for MMC V3.31 or earlier). The commands are transmitted serially on the CMD line. All commands are fixed to 48 bits in length, and the following table gives the general command formats on MMC, SD memory cards, and SDIO cards.

The CE-ATA command is an extension of the MMC V4.2 command, so it has the same format.

The command channel is in half-duplex mode so that commands and responses can be sent and received. If the CPSM is not in the transmit state, the SDIO_CMD output is in the high impedance state. The data on SDIO_CMD is synchronized with the rising edge of SDIO_CK.

Table 31-2 command format

Bit	Width	Value	Description
47	1	0	Start bit
46	1	1	Transfer bit
[45: 0]	6	-	Command Index
[39: 8]	32	-	Parameter
[7: 1]	7	-	CRC7
0	1	1	End bit

Response: A response is sent to the host by a card with an assigned address. The response is a response to a previously received command. The response is transmitted serially on the CMD line.

SDIO supports 2 response types, both of which have CRC error detection:

- 48-bit short response
- 136-bit long response

If the response does not contain a CRC (such as the response of CMD1), the device driver should ignore the CRC error status.

Table 31-3 short response format

Bit	Width	Value	Description
47	1	0	Start bit
46	1	1	Transfer bit
[45: 0]	6	-	Command Index
[39: 8]	32	-	Parameter
[7: 1]	7	-	CRC7
0	1	1	End bit

Table 31-4 length response format

Bit	Width	Value	Description
135	1	0	Start bit
134	1	0	Transfer bit
[133: 128]	6	111111	Reserved
[127: 1]	127	-	CID or CSD (including internal CRC7)
0	1	1	End bit

The command register contains the command index (6 bits issued to the card) and the command type; The command itself determines whether a response is required and the type of response: 48-bit or 136-bit. The status flags in the command channel are shown in the following table:

Table 31-5 command channel status flags

mark	Description
CMDREND	Correct CRC for response
CCRCFAIL	CRC error in response
CMDSENT	Commands (commands that do not require a response) have been sent
CTIMEOUT	Response timeout
CMDACT	Sending commands

The CRC generator calculates the CRC checksum of all bits prior to the CRC code, including the start bit, the transmit bit, the command index, and the command parameter (or card status). For the long response format, the CRC check calculates the first 120 bits of the CID or CSD.

Note that the start bit, the transmit bit, and the 6 reserved bits in the long response format do not participate in the CRC calculation. The CRC check value is a 7-bit value.

31.2.3 Data unit

The data unit realizes data transmission between the host and the card. When the data width is 8 bits, the data transmission uses the SDIO_D [7: 0] signal line; When the data width is 4 bits, the data transmission uses the SDIO_D [3: 0] signal line; When the data width is 1 bit, the data transmission uses the SDIO_D [0] signal line. The data transmission flow is controlled by a data state machine (DSM).

- Data Sending State Machine (DTSM)

The DTSM operates at the SDIO_CK frequency, and the card bus signal is synchronized with the rising edge of SDIO_CK.

As shown in the figure, the data sending state machine starts data sending two clocks after receiving the response of the data write command; This happens even if the command channel state machine detects an error or responds to a CRC error. If no response is received from the card due to response timeout, no data is transmitted. Depending on the value of the transfer_mode bit in the command register, the data sending state machine places data on the card data bus in streams or blocks.

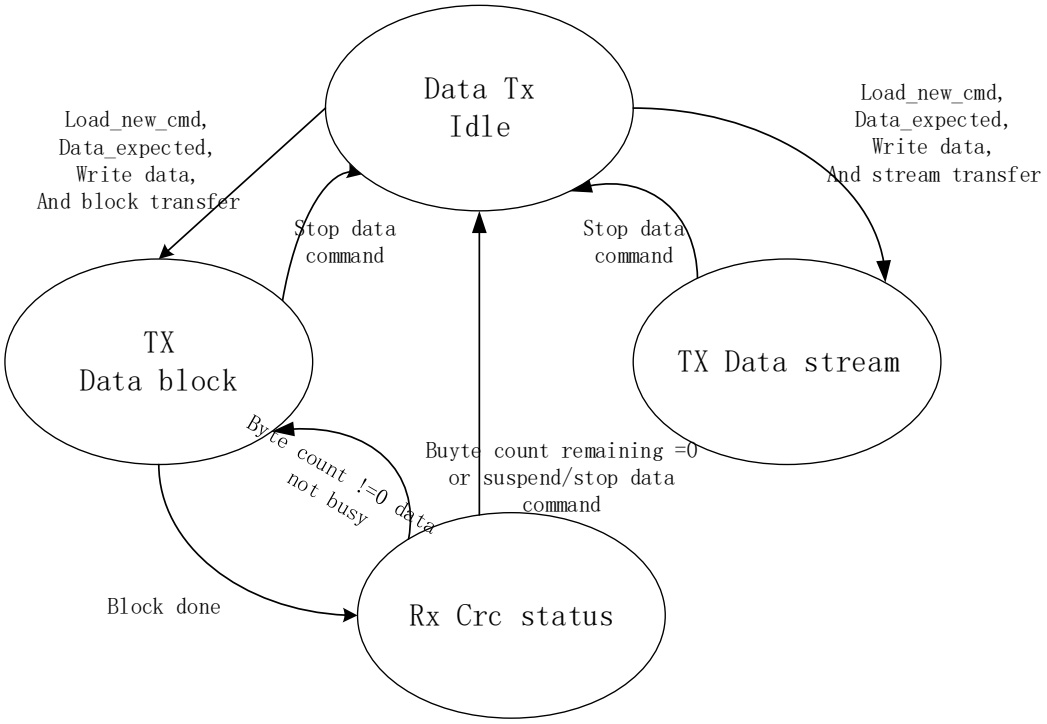


Figure 31-4 Data Send State Machine (DTSM)

- **Data Reception State Machine (DRSM)**

As shown, the data receiving state machine receives data two clock cycles after the end bit of the data read command, even if the command channel detects a response error or a response CRC error. If the response from the card is not received because the response timed out, the SDIO adapter will not receive the signal that the data transmission is complete; This happens if the command sent is an illegal operation on the card, which prevents the card from starting to read the data transfer. If data is not received before the data timeout, the data receiving state machine signals the data timeout to the SDIO adapter and ends the data transmission. Depending on the value of the transfer_mode bit in the command register, the data receiving state machine fetches data from the card data bus as a stream or a block.

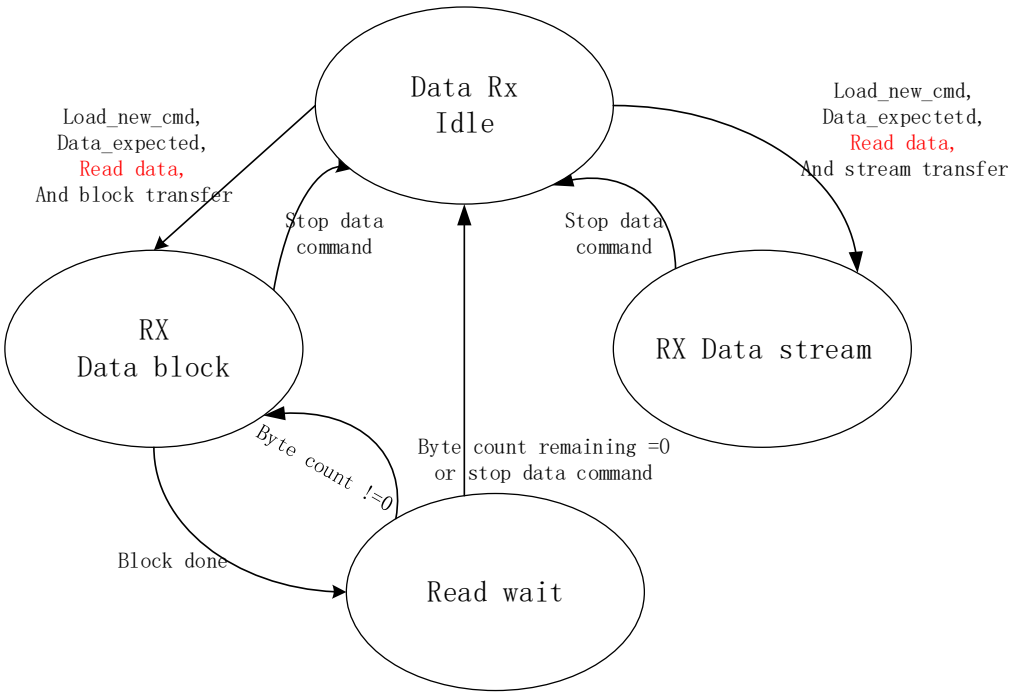


Figure 31-5 Data receiving state machine (DRSM)

31.2.4 SDIO clock control

SDIO mainly has two clock controls, one is the system clock clk and the other is clk_in. The specific clock description is as follows:

1. clk, rising edge valid: system clock,
2. Cclk_in rising edge valid: card clock, which needs to satisfy "clk > = cclk_in/10"
3. Cclk_in_drv: The delayed clk_in drives the clock of the card, which is used to control data_out and cmd_out.
4. Cclk_in_sample: The input of the delayed clk_in sample card clock, which needs to consider the delay on the PAD, and is used to control the sampling of ccmd_in and cdata_in
5. The falling edge of cclk_in is valid: the falling edge of clk_in drives the clock for stop_clk_neg and fifo reading and writing

There are also three register bits that can be used for clock processing, namely the CKSEL, SMPEN, SMPCLKSEL bits of clk:

1. CKSEL: Used to control output commands, data selection
2. SMPEN: pre-sampling enable,
3. SMPCLKSEL: Clock phase selection, you can select the specific phase of the clock

31.3 SDIO AHB interface

The AHB interface generates interrupts and DMA requests and accesses the SDIO interface registers and the data FIFO. It contains a data channel, register decoder, interrupts, and DMA control logic.

31.3.1 SDIO interrupts

When at least one status flag is high, the interrupt control logic generates an interrupt request. There is a mask register for selecting conditions that can generate interrupts, and if the corresponding mask flag is set, the corresponding status flag can generate interrupts.

31.3.2 SDIO/DMA interface: the process of data transfer between SDIO and memory

SDIO/DMA: The process of data transfer between SDIO and memory.

In the following example, the host controller uses CMD 24 (WRITE_BLOCK) to transfer 512 bytes from the host to the MMC card, and the DMA controller is used to fill data from memory to the FIFO of the SDIO.

1. Perform the card identification process
2. Increase SDIO_CK frequency
3. Send CMD7 command to select card
4. Configure DMA2 as follows:
   a) Enable the DMA2 controller and clear all interrupt flags
   b) Set the source address register of DMA2 channel 4 as the base address of the memory buffer, and the destination address register of DMA2 channel 4 as the address of the SDIO_FIFO register
   c) Set DMA2 channel 4 control register (memory increment, non-peripheral increment, data width of peripheral and source is word width)
   d) Enable DMA2 Channel 4
5. Send CMD24 (WRITE_BLOCK) as follows:
   a) Set the SDIO data length register (the SDIO data clock register should be set before performing the card identification process)
   b) Set the SDIO parameter register to the address in the card to which data needs to be transferred
   c) Set the SDIO command register: CmdIndex is set to 24 (WRITE_BLOCK); WaitRest is set to 1 (SDIO card host waits for response); Start_cmd is set to 1 (enables the SDIO card host to send commands), keeping other fields at their reset values.
   d) Wait for the CD (end of command) to interrupt, and then set the SDIO data register: DTEN is set to 1 (enable the SDIO card host to send data); DTDIR is set to 0 (controller to card direction); DTMODE is set to 0 (block data transfer); DMAEN is set to 1 (enable DMA); DBLOCKSIZE is set to 9 (512 bytes); Other fields do not need to be set.
   e) Wait for SDIO_STA [10] = 1.
6. Query the enable status register of the DMA channel to confirm that no channel is still enabled.

31.4 Card Function Description

31.4.1 Card recognition mode

In card recognition mode, the host resets all the cards, detects the operating voltage range, identifies the cards and sets a relative address (RCA) for each card on the bus. In the card recognition mode, all data communications use only the command signal line (CMD).

31.4.2 Card reset

The GO_IDLE_STATE command (CMD0) is a software reset command that puts the multimedia card and SD memory into an idle state. The IO_RW_DIRECT command (CMD52) resets the SD I/O card. After power-on or after CMD0 is executed, the outputs of all cards are in a high impedance state, and

all cards are initialized to a default relative card address (RCA = 0x0001) and default driver register setting (lowest speed, maximum current driving capacity).

31.4.3 Operating voltage range validation

All cards can communicate with the SDIO card host using any specified range of voltages, and the minimum and maximum supportable voltage VCC values are defined by the operating condition register (OCR) on the card.

A card whose internal memory stores a card identification number (CID) and card specific data (CSD), which can only be transmitted under data transmission VCC conditions. When the SDIO card host module is inconsistent with the VCC rangeof the card, the card will not be able to complete the identification cycle or send CSD data; Therefore, when the VCC range does not match, the SDIO card host may use the following special commands to identify and reject the card: SEND_OP_COND (CMD1), SD_APP_OP_COND (ACMD41 of the SD memory card), and IO_SEND_OP_COND (CMD5 of the SD I/O card). The SDIO card host will generate the required VCC voltage when executing these commands. Cards that cannot transmit data in the specified voltage range will be disconnected from the bus and entered into an inactive state.

Using these commands that do not contain a voltage range as an operand, the SDIO card host can query each card and put cards that are not within this range into an inactive state before determining a common voltage range. The SDIO card host can make such a query when the SDIO card host can select a common voltage range or when the user needs to know if the card is usable.

31.4.4 Card identification process

There is a difference in the card recognition process between multimedia cards and SD cards; For multimedia cards, the card identification process starts at the clock frequency Fod, and all SDIO_CMD outputs are driven open circuit, allowing parallel connection of cards in this process. The identification process is as follows:

1. The bus is activated
2. The SDIO card host broadcast transmits the SEND_OP_COND (CMD1) command and receives the operating conditions
3. The resulting response is a "line AND" of the operating condition register contents of all cards
4. Incompatible cards are placed inactive
5. SDIO card host broadcast sends ALL_SEND_CID (CMD2) to all active cards
6. All active cards simultaneously send their CID numbers serially, and those cards that detect that the output CID bits do not match the data on the command line must stop sending and wait for the next identification cycle. In the end, only one card can successfully transmit the complete CID to the SDIO card host and enter the identification state.
7. The SDIO card host sends a SET_RELATIVE_ADDR (CMD3) command to this card. This new address is called the relative card address (RCA), which is shorter than the CID and is used to address the card. At this point, the card goes to standby and no longer responds to the new identification process, and its output drive changes from open circuit to push-pull mode.
8. The SDIO card host repeats steps 5 through 7 above until a timeout condition is received.

For SD cards, the card identification process starts at the clock frequency Fod, and all SDIO_CMD outputs are push-pull drive instead of open circuit drive. The identification process is as follows:

1. The bus is activated
2. The SDIO card host broadcasts to send the SEND_APP_OP_COND (ACMD41) command
3. The resulting response is the contents of the operating condition registers for all cards
4. Incompatible cards are placed inactive
5. SDIO card host broadcast sends ALL_SEND_CID (CMD2) to all active cards
6. All activated cards send back their unique card identification number (CID) and enter the identification status.
7. The SDIO card host sends the SET_RELATIVE_ADDR (CMD3) command and an address to an active card. This new address is called the relative card address (RCA), which is shorter than the CID and is used to address the card. At this point, the card goes to standby mode. The SDIO card host can send this command again to change the RCA, and the card's RCA will be the last assignment.
8. The SDIO card host repeats steps 5 through 7 above for all activated cards.

31.4.5 Write data block

When the write block command (CMD24-27) is executed, the host transfers one or more data blocks from the host to the card, and a CRC code is transferred at the end of each data block. A card that supports write block commands should always be able to receive blocks defined by WRITE_BL_LEN. If the CRC check is erroneous, the card indicates the error through the SDIO_D signal line, the transferred data is discarded and not written, and all subsequent (in multi-block write mode) transferred data blocks are ignored.

If the host transmits partial data and the accumulated data length is not aligned with the data block, the card will detect a block misalignment error before the first misaligned block (setting the ADDRESS ERROR error bit in the status register) when block misalignment is not allowed (without setting the parameter WRITE_BLK_MISALIGN of the CSD). When the host attempts to write a write-protected area, the write operation will also be aborted, at which time the card sets the WP_VIOLATION bit.

Setting the CID and CSD registers does not need to set the block length in advance, and the transmitted data is also protected by CRC. If part of the CSD or CID register is stored in the ROM, this part that cannot be changed must be consistent with the corresponding part of the receive buffer. If there is an inconsistency, the card will report an error without modifying the contents of any register. Some cards take a long or even unpredictable time to complete writing a data block. After receiving a data block and completing the CRC check, the card starts writing operations. If its write buffer is full and it can no longer receive new data from the new WRITE_BLOCK command, it will pull the SDIO_D signal line down. The host can check the status of the card at any time using SEND_STATUS (CMD13), and the card will return to its current status. The READY_FOR_DATA status bit indicates whether the card can receive new data or whether a write operation is still in progress. The host can use CMD7 (Select Another Card) to unselect a card and put the card in the disconnected state, which can release the SDIO_D signal line without interrupting the outstanding write operation; When a card is re-selected, if the write operation is

still in progress and the write buffer is still unavailable, it will re-indicate the busy state by pulling down the SDIO_D signal line.

31.4.6 Read data block

In read block mode, the basic unit of data transmission is a data block, the size of which is defined in the CSD (READ_BL_LEN). If READ_BL_PARTIAL is set, a smaller data block can also be transferred. A smaller data block means that the start and end addresses are completely contained in a physical block. READ_BL_LEN defines the size of the physical block. To ensure the correctness of data transmission, there is a CRC check code after each data block. CMD 17 (READ_SINGLE_BLOCK) starts a read block operation, and the card returns to the transmission state after the transmission is completed. CMD 18 (READ_MULTIPLE_BLOCK) initiates a read operation of multiple consecutive data blocks at a time.

The host may abort the multi-block read operation at any time, regardless of the type of operation. Send a stop transmission command to abort the operation.

If the card detects an error (e.g. out of bounds, address misalignment, or internal error) during a multi-block read operation (either type), it will stop data transmission and remain in the data state; At this time, the host must send a stop transmission command to abort the operation. Report a read error in the response to the stop transmission command.

If when the host sends the stop transmission command, the card has finished transmitting the last data block of a certain number of data block operations, because the card is no longer in the data state at this time, the host will receive a response with an illegal command. If the host transmits part of the data block and the accumulated data length cannot be aligned with the physical block and block misalignment is not allowed, the card detects a block alignment error when the first misaligned block occurs and sets the ADDRESS_ERROR error flag in the status register.

31.4.7 Erasure: Group and Sector Erasure

The erase unit of the multimedia card is the erase group, which is calculated by write data block, and the write data block is the basic write unit of the card. The size of the erase group is a card-specific parameter, defined in the CSD.

The host can erase a continuous range of erase groups, and there are three steps to start the erase operation.

First, the host uses the ERASE_GROUP_START (CMD35) command to define the start address of the continuous range, then uses the ERASE_GROUP_END (CMD36) command to define the end address of the continuous range, and finally sends the ERASE command ERASE (CMD38) to start the ERASE operation. The address field of the erase command is the erase group address in bytes. The card will discard the portion that is not aligned with the size of the erase group and align the address boundary to the boundary of the erase group.

If the erase command is not received as described above, the card sets the ERASE_SEQ_ERROR bit in the status register and waits for the first step again.

If a command other than the SEND_STATUS and erase command is received, the card sets the ERASE_RESET bit in the status register, cancels the erase sequence and executes a new command.

31.4.8 Wide bus selection and de-selection

The wide bus (4-bit bus width) operation mode can be selected or not selected by the SET_BUS_WIDTH (ACMD6) command, and the default bus width is 1 bit after power-up or after the GO_IDLE_STATE (CMD0) command. The SET_BUS_WIDTH (ACMD6) command is only valid in the transfer state, i.e. the bus width can only be changed after the card has been selected using the SELECT/DESELECT_CARD (CMD7) command.

31.4.9 Protection management

The SDIO card host module supports three protection methods:

1. Internal card protection (in-card management)
2. Mechanical write protection switch (only managed by SDIO card host module)
3. Card locking operation of password management

31.4.10 Card status register

Response format R1 contains a 32-bit card status field, which is used to send card status information to the card host (this information may be stored in a local status register). Unless otherwise specified, the status returned by the card is always related to the previous command.

The abbreviations in the table for types and purge condition fields are defined as follows:

Type:

● E: Error bit

● S: Status bit

● R: detection bit and set according to the actual command response

● X: detection bit, set during the execution of the command. The SDIO card host checks the status of the card by sending a status command to read out these bits.

Clear condition:

● A: According to the current status of the card

● B: Always related to the previous command. The correct command is received to clear (with a delay of one command).

● C: Read to clear

Table 31-6 card status

Bit	Pin name	Type	Value	Description	Clear condition
31	ADDRESS_OUT_OF_RANGE	E R X	'0' = no error '1' = error	A multi-block or stream read/write operation (even starting from a legitimate address) attempts to read or write a portion that exceeds the capacity of the card.	C
30	ADDRESS_ISALIGN		'0' = no error '1' = error	The first data block defined by the address parameter in the command (compared to the current data block length) is not aligned with the physical block of the card. A multi-block or data stream read/write operation (even starting from a legitimate address) attempts to read or write a block of data that is not aligned with the physical block.	C
29	BLOCK_LEN_ERROR		'0' = no error '1' = error	The parameters of the SET_BLOCKLEN command exceed the maximum allowable range of the card, or the previously defined data block length is illegal for the current command (for example, the host issues a write command, the current block length is less than the minimum allowed length of the card,	C

				and at the same time, it is not allowed to write some data blocks).	
28	ERase_SEQ_ERRO R		'0' = no error '1' = error	The erase commands were sent in the wrong order.	C
27	ERase_PARAM	EX	'0' = no error '1' = error	An illegal erase group was selected during the erase.	C
26	WP_VIOLATION	E X	'0' = no error '1' = error	Trying to program a write-protected block of data.	C
25	CARD_IS_Locked	S R	'0' = card unlocked '1' = card locked	When this bit is set, it means that the card has been locked.	A
24	LOCK_Unlock_FAIL ED	E X	'0' = no error '1' = error	There is a wrong order of commands in locking/unlocking or a wrong password detected	C
23	COM_CRC_ERROR	E R	'0' = no error '1' = error	CRC verification error in previous command.	B
22	ILLEGAL_ COMMAND	E R	'0' = no error '1' = error	For the current card status, the command is illegal.	B
21	CARD_ECC_FAILE D	E X	'0' = Success '1' = failure	ECC checks were implemented inside the card but failed to correct the data.	C
20	CC_ERROR	E R	'0' = no error '1' = error	(Not defined in the standard) An error occurred inside the card, independent of the command of the host.	C
19	ERROR	E X	'0' = no error '1' = error	Generated related to the execution of the previous host command (not defined in the standard) Errors inside the card	C
18: 17	Reserved	-	-	-	-
16	CID/CSD_OverWRIT E	E X	'0' = no error '1' = error	It can be any of the following errors: The CID register has been written and cannot be overwritten The read-only part of the CSD does not match the contents of the card An attempt to perform a reverse operation of copy or permanent write protection, that is, to restore or de-write protection.	C
15	WP_ERASE_SKIP	E X	'0' = unprotected '1' = protected	Encountered an already existing write-protected block and only part of the address space was erased	C
14	CARD_ECC_DISABl ed	S X	'0' = not allowed '1' = allowed	The command was executed without using internal ECC.	A
13	ERASE_RESET		'0' = unprotected '1' = protected	The sequence entering the erase process is aborted because a command outside the erase sequence is received (non-CMD35, CMD36, CMD38, or CMD13 command).	C
12: 9	Current_STATE	S R	0 = idle 1 = Ready 2 = Identification 3 = Standby 4 = Send 5 = Data 6 = Receive 7 = Programming 8 = Disconnect	The state of the state machine in the card when the command is received. If the execution of the command leads to B Causes a change in state, this change will be reversed in the response to the next command Reflect it. These four bits are interpreted as decimal numbers 0 to 15.	B

			9 = busy test 10 ~ 15 = Reserved		
8	READY_FAR_DATA	S R	'0' = not ready '1' = Ready	Corresponds to the signal that the buffer on the bus is empty.	
7	SWITCH_ ERROR	E X	'0' = no error '1' = error	The card did not transition to the desired mode as required by the SWITCH command.	B
6	Reserved				
5	APP_CMD	S R	'0' = not allowed '1' = allowed	The card expects an ACMD, or indicates that the command has been interpreted as an ACMD command	C
4	Reserved for SD I/O cards				
3	AKE_SEQ_ERROR	E R	'0' = no error '1' = error	There is a wrong order of validation	C
2	Reserved for app-related commands.				
1: 0	Test mode reserved for manufacturer				

31.4.11 SD status register

The SD state contains status bits related to specific functions of the SD memory card and some status bits related to future applications. The length of the SD state is a 512-bit data block. Upon receipt of the ACMD13 command (CMD55, then CMD13), the contents of this register are transferred to the SDIO card host. ACMD13 commands can only be sent when the card is in the transmission state (the card has been selected).

X The following table defines different SD status register information. The abbreviations for types and clear condition fields in the table define the following types:

- E: Error bit
- S: status bit
- R: detection bit and set according to the actual command response
- X: detection bit, set during the execution of the command. The SDIO card host checks the status of the card by sending a status command to read out these bits.

Clear condition:

- A: According to the current status of the card
- B: Always related to the previous command. The correct command is received to clear (with a delay of one command).
- C: Clear after reading

Table 31-7 SD Status

Bit	Pin name	Type	Value	Description	Clear condition
511: 510	DAT_BUS_WIDTH	SR	'00' = 1 (default) '01' = reserved '10' = 4 bits wide '11' = reserved	The current data bus width defined by the SET_BUS_WIDTH command.	A
509	SECURED_MODE	SR	'0' = not in confidential mode '1' = in secret mode	The card is in a secure mode of operation '0' = not in confidential mode '1' = in secret mode	A
508: 496	Reserved				

495: 480	SD_CARD_TYPE	SR	00xxh '= in physical specification version 1.01 A ~ 2.00 SD memory card ('x 'indicates any value). The defined cards are: '0000' = universal SD read/write card '0001' = SD ROM Card	The lower 8 bits of this field can define different variants of the SD memory card in the future (each bit can be used to define a different SD type). The upper 8 bits can be used to define those SD cards that do not comply with current SD physical layer specifi-cations	A
479: 448	SIZE_OF_PROTECTED_AREA	SR	Protected area size (see description below)	(See description below)	A
447: 440	Speed_CLASS	SR	Speed type of card (see description below)	(See description below)	A
439: 432	PERFORMANCE_MOVE	SR	Transmission perfor-mance in 1MB/sec (see description below)	(See description below)	A
431: 428	AU_Size	SR	Size of AU (see de-scription below)	(See description below)	A
427: 424	Reserved				
423: 408	ERASE_SIZE	SR	Number of AUs that can be erased at a time	(See description below)	A
407: 402	ERase_TIMEOUT	SR	Wipe the timeout value for the range specified by UNIT_OF_ERASE_A	(See description below)	A
401: 400	ERase_OFFSET	SR	Fixed offset value added on erase	(See description below)	A
399: 312	Reserved				
311: 0	Reserved for manufacturers				

SIZE_OF_PROTECTED_AREA

Standard capacity cards and high capacity cards set this bit differently. For a standard capacity card, the capacity of the protected area is calculated by:

Protected area = SIZE_OF_PROTECTED_AREA * MULT * BLOCK_LEN

The unit of SIZE_OF_PROTECTED_AREA is MULT * BLOCK_LEN.

For high capacity cards, the capacity of the protected area is calculated by:

Protected area = SIZE_OF_PROTECTED_AREA

The unit of SIZE_OF_PROTECTED_AREA is bytes.

Speed_CLASS

These 8 bits indicate the type of speed and the value that can be calculated by PW/2 (PW is the performance of writing).

Table 31-8 Speed Type Code

Speed_CLASS	Numerical Definition
00h	Type 0
01h	Type 2
02h	Type 4
03h	Type 6
04h-FFH	Reserved

PERFORMANCE_MOVE

These 8 bits indicate mobility performance (Pm) in units of 1 MB/sec. If the card does not move data using RU (unit of record), Pm should be considered to be infinite. Setting this field to FFh means infinity.

Table 31-9 mobile performance codes

Speed_CLASS	Numerical Definition
00h	Undefined
01h	1 MB/S
02h	2 MB/S
……	……
Feh	254 MB/sec
FFh	Infinity

AU_Size

These 4 bits indicate the length of the AU, and the value is a multiple of the power of 2 of 16K bytes.

Table 31-10 AU_SIZE Code

Speed_CLASS	Numerical Definition
00h	Undefined
01h	16 KB
02h	32 KB
03h	64 KB
04h	128 KB
05h	256 KB
06h	512 KB
07h	1 MB
08h	2 MB
09h	4 MB
AH-Fh	Reserved

Depending on the card capacity, the maximum AU length is defined by the table below. The card may set an arbitrary AU length between the RU length and the maximum AU length.

Table 31-11 Maximum AU Length

Size	16-64 MB	128-256 MB	512 MB	1-32 GB
Maximum AU length	512 KB	1 MB	2 MB	4 MB

ERASE_SIZE

This 16-bit field gives NERASE, and ERASE_TIMEOUT defines the timeout when NERASE AUs are erased. The host should determine the appropriate number of AUs to erase in one operation so that the host can display the progress of the erase operation. If the domain is 0, the timeout calculation of erasure is not supported.

Table 31-12 ERASE_SIZE Code

ERase_CLASS	Numerical Definition
0000h	Timeout operation for erasure is not supported
0001h	1 AU
0002h	2 AU
0003h	3 AU
……	……
FFFFh	65535 AU

ERase_TIMEOUT

These 6 bits give TERASE, ERASE_SIZE indicates that a plurality of AUs are erased, and this value gives the erase timeout from the offset. The range of ERASE_TIMEOUT can be defined to a maximum of 63 seconds, and the manufacturer of the card can select an appropriate combination of ERASE_SIZE and ERASE_TIMEOUT according to the specific implementation, determining ERASE_TIMEOUT first and then ERASE_SIZE.

Table 31-13 erase timeout code

ERase_TIMEOUT	Numerical Definition
00	Timeout operation not supported for erasure
01	1 sec
02	2 seconds
03	3 seconds
……	……
63	63 seconds

ERase_OFFSET

These two bits give TOFFSET, which is meaningless when ERASE_SIZE and ERASE_TIMEOUT are both 0.

Table 31-14 Erase Offset Code

ERase_OFFSET	Numerical Definition
0	0 seconds
1	1 sec
2	2 seconds
3	3 seconds

31.4.12 I/O Mode of SD

I/O interrupt for SD

In order to allow the SD I/O card to interrupt the MMC/SD module, there is a pin with interrupt function on the SD interface-pin 8. In 4-bit SD mode, this pin is SDIO_D1, and the card uses it to submit an interrupt application to the MMC/SD module. The interrupt function is optional for each card or function within the card. The interrupt of SD I/O is active level, that is, the interrupt signal line must remain active level (low) before being recognized and receiving a response from the MMC/SD module, and inactive level (high) after the end of the interrupt process. After the MMC/SD module serves the interrupt request, the interrupt status bit can be cleared by writing the appropriate bit to the internal register of the SD I/O card through an I/O write operation. The interrupt output of all SD I/O cards is active low, and pull-up resistors are provided on all data lines (SDIO/D [3: 0]) of the MMC/SD module. The MMC/SD module samples pin 8 (SDIO_D/IRQ) during the interrupt phase and performs interrupt detection, and the value on this signal line will be ignored at other times.

Both memory operations and I/O operations have interrupt phases, and the definition of the interrupt phase of a single data block operation is different from that of a plurality of data block transfer operations.

I/O suspension and resumption of SD

In a multifunctional SD I/O card or a card having both I/O and memory functions, a plurality of devices (I/O and memory) share the MMC/SD bus. In order to enable multiple devices in the MMC/SD module to share the bus, the SD I/O card and the composite card can selectively implement the concept of pause/resume; If a card supports pause/resume, the MMC/SD module can temporarily stop the data transfer operation (pause) of one function or memory, thereby giving up the bus to another function or memory having a higher priority, and resuming the previously suspended transmission after this higher priority transmission is complete. Actions that support pause/resume are optional. There are the following steps to perform a pause/resume operation on the MMC/SD bus:

1. Determine the current function of the SDIO_D [3: 0] signal line
2. Request a low-priority or slow operation pause

3. Wait for the pause operation to complete and confirm that the device has been paused
4. Start high-priority transmission
5. Wait for high-priority transmission to end
6. Resume suspended operations

SD I/O ReadWait

The optional read wait (RW) operation is only available in the 1-bit or 4-bit mode of the SD card. The read wait operation allows the MMC/SD module to ask a card to temporarily stop data transfer while it is reading multiple registers (IO_RW_EXTENDED, CMD53), while allowing the MMC/SD module to send commands to other functions in the SD I/O device. To determine whether a card supports the read wait protocol, the MMC/SD module should check the internal registers of the card. The read wait time is related to the interrupt phase

31.4.13 Commands and responses

31.4.13.1 Apply dependent and generic commands

SD card host module system is used to provide a standard interface suitable for various application types, but at the same time, it should take into account the functions of specific users and applications. Therefore, two types of general commands are defined in the standard: application-related commands (ACMD) and general commands (GEN_CMD).

When the card receives the APP_CMD (CMD55) command, the card expects the next command to apply the relevant command. The application-related command (ACMD) has the same format structure as a normal MMC and can use the same CMD number because it appears after the APP_CMD (CMD55), so the card recognizes it as an ACMD command. If APP_CMD (CMD55) is not followed by an already defined application-related command, it is considered a standard command; Example: There is an SD_STATUS (ACMD13) application related command. If CMD13 is received immediately after APP_CMD (CMD55), it is interpreted as SD_STATUS (ACMD13); However, if the card receives CMD7 immediately after APP_CMD (CMD55) and this card does not define ACMD7, it will be interpreted as a standard CMD7 (SELECT/DESELECT_CARD) command.

If you want to use the manufacturer-defined ACMD, the SD card host needs to do the following operations:

1. Send the APP_CMD (CMD55) command

The card sends back a response to the multimedia/SD card module indicating that the APP_CMD bit is set and waits for the ACMD command.

2. Send the specified ACMD

The card sends back a response to the multimedia/SD card module indicating that the APP_CMD bit is set and that the received command has been correctly parsed according to the ACMD command; If a non-ACMD command is sent, the card will follow the normal MMC command and clear the APP_CMD bit of the status register in the card.

If an illegal command (whether ACMD or CMD) is sent, it will be error handled according to the standard illegal MMC command.

The bus operation procedure of the GEN_CMD command is the same as that of the single data block read/write command (WRITE_BLOCK, CMD24 or READ_SINGLE_BLOCK, CMD17); At this time, the parameters of the command represent the direction of data transmission instead of the address, and the data block has a user-defined format and meaning.

Before sending the GEN_CMD (CMD56) command, the card must be selected (the state machine is in the transmission state) and the length of the data block is defined by SET_BLOCKLEN (CMD16). The response to the GEN_CMD (CMD56) command is in R1b format.

31.4.13.2 Command Type

There are four different types of application-dependent and generic commands:

1. Broadcast Command (BC): Sent to all cards, no response returned.

2. Broadcast command with response (BCR): Send to all cards and receive responses returned from all cards at the same time.

3. Command (AC) with addressing (point-to-point): sent to the selected card, data transmission is not included on the SDIO_D signal line.

4. Data Transmission Command (AC) with Addressing (Point-to-Point): Send to the selected card, including data transmission on the SDIO_D signal line.

31.4.13.3 Command format

See Command Format Table

Commands for MMC/SD Card Module

Table 31-15 Block Transfer Based Write Commands

CMD Index	Type	Parameter	Response format	Written	Description
CMD23	ac	[31: 16] = 0 [15: 0] = number of data blocks	R1	SET_BLOCK_Count	Define the number of transport blocks needed in subsequent multi-block read or write commands
CMD24	adtc	[31: 0] = data address	R1	Write_BLOCK	Write a block of the length selected by the SET_BLOCKLEN command
CMD25	adtc	[31: 0] = data address	R1	Write_MULTIPLE_BLOCK	Continuously write blocks of data until a STOP_TRANSMISSION command is received or the specified number of blocks is reached
CMD26	adtc	[31: 0] = padding bit	R1	PROGRAM_CID	Program the identification register of the card. You can only send this command once per card. There are hardware mechanisms in the card to prevent multiple programming operations. Usually this command is reserved for the manufacturer
CMD27	adtc	[31: 0] = padding bit	R1	PROGRAM_CSD	Program the programmable bits in the CSD of the card.
CMD28	ac	[31: 0] = data address	R1b	SET_WritE_PROT	If the card has write protection, this command sets the write protection bits for the specified group. The write protection feature is set in the

					special data area of the card (WP_GRP_SIZE).
CMD29	ac	[31: 0] = data address	R1b	CLR_Write_PROT	If the card has write protection, this command clears the specified set of write protection bits
CMD30	adtc	[31: 0] = write protected data address	R1	Send_WRITE_PROT	If the card has write protection, this command requires the card to send the status of the write protection bit
CMD31	Reserved				

Table 31-16 Write Protection Commands Based on Block Transfer

CMD Index	Type	Parameter	Response format	Written	Description
CMD28	ac	[31: 0] = data block address	R1b	SET_WritE_PORT	Define the number of transport blocks needed in subsequent multi-block read or write commands
CMD29	ac	[31: 0] = data address	R1b	CLR_Write_PORT	Write a block of the length selected by the SET_BLOCKLEN command
CMD30	adtc	[31: 0] = write protected data address	R1	Send_WRITE_PROT	If the card has write protection, this command requires the card to send the status of the write protection bit
CMD31	Reserved				

Table 31-17 erase command

CMD Index	Type	Parameter	Response format	Written	Description
CMD32 … CMD34	Reserve. These command codes cannot be used for backward compatibility with older versions of the Media Card Protocol.				
CMD35	ac	[31: 0] = data address	R1b	CLR_Write_PORT	Write a block of the length selected by the SET_BLOCKLEN command
CMD36	adtc	[31: 0] = write protected data address	R1	Send_WRITE_PROT	If the card has write protection, this command requires the card to send the status of the write protection bit
CMD37	Reserve. For backward compatibility with older versions of the media card protocol, this command code cannot be used				
CMD38	ac	[31: 0] = filler	R1	ERASE	Erase the previously selected data block

Table 31-18 I/O Mode Commands

CMD Index	Type	Parameter	Response format	Written	Description
CMD39	ac	[31: 16] = RCA [15] = register write flag [14: 8] = register address [7: 0] = register data	R4	FAST_IO	For writing and reading 8-bit (register) data fields. This command specifies a card and register, and also provides data to be written if the write flag is set. The R4 response contains data read from the specified register. This command accesses application-related registers that are not defined in the MMC standard.
CMD40	bcr	[31: 0] = write protected data address	R5	GO_IRO_STATE	Set the system in interrupt mode.
CMD41	Reserved				

Table 31-19 lock command

CMD Index	Type	Parameter	Response format	Written	Description
CMD42	adtc	[31: 0] = padding bit	R1b	LOCK_Unlock	Set/clear the password or lock/unlock the card. The length of the data block is set by the SET_BLOCKLEN command.
CMD43 …… CMD54	Reserved				

Table 31-20 related commands

CMD Index	Type	Parameter	Response format	Written	Description
CMD55	ac	[31: 16] = RCA [15: 0] = padding bit	R1	APP_CMD	Indicates that the next command of the card is to apply the related command instead of the standard command
CMD56	adtc	[31: 1] = padding bit [0] = RD/WR			In general purpose or application-related commands, either for transferring a data block into or reading a data block from the card. The length of the data block is set by the SET_BLOCKLEN command.
CMD57 … CMD59	Reserved				
CMD60 …… CMD63	Reserved for manufacturers				

31.5 Response format

All responses are transmitted over the SDIO_CMD signal line via the MCCMD command. The transmission of a response always starts at the far left of the bit string corresponding to the response word, the length of which depends on the type of response.

A response always has a start bit (always 0) followed by the direction bit of transmission (card = 0). The numerical values denoted x in the table below represent a variable portion. All responses are CRC protected except for the R3 response type. Each command codeword has an end bit (always 1).

There are 7 response types, and their formats are defined as follows:

31.5.1 R1 (normal response command)

Code length = 48 bits. Bit 45: 40 indicates the command index to be responded to, and its value ranges from 0 to 63. The status of the card is encoded by 32 bits.

Table 31-21 R1 Response

Bit	Domain width	Value	Description
47	1	0	Start bit
46	1	0	Transfer bit
[45: 40]	6	X	Command Index
[39: 8]	32	X	Card status
[7: 1]	7	X	CRC7
0	1	1	End bit

31.5.2 R1b

Same format as R1, but with the option to send a busy signal on the data line. Upon receipt of these commands, the card may become busy depending on the status before the command was received.

31.5.3 R2 (CSD, CSD register)

Code length = 136 bits. The contents of the CID register will be issued as a response to CMD2 and CMD10. The contents of the CSD register will be issued as a response to CMD9. The card only sends out the bits of CID and CSD [127 … 1], and bit 0 of these registers is replaced by the end bit of the response at the receiving end. The card indicates that it is performing an erase operation by pulling MCDAT low; The actual erase operation can take very long, and the host can send a CMD7 command to unselect this card.

Table 31-22 R2 Response

Bit	Domain width	Value	Description
135	1	0	Start bit
134	1	0	Transfer bit
[133: 128]	6	'111111'	Command Index
[127: 1]	127	X	Card status
0	1	1	End bit

31.5.4 R3 (OCR register)

Code length = 48 bits. The contents of the OCR register will be issued as a response to CMD1. The definition of the level code is: limited voltage window = low, card busy = low.

Table 31-23 R3 Response

Bit	Domain width	Value	Description
47	1	0	Start bit
46	1	0	Transfer bit
[45: 40]	6	'111111'	Reserved
[39: 8]	32	X	OCR registers
[7: 1]	7	'111111'	Reserved
0	1	1	End bit

31.5.5 R4 (Fast I/O)

Code length = 48 bits. The parameter field contains the RCA of the specified card, the address to be read out or written to the register, and its contents.

Table 31-24 R4 Response

Bit		Domain width	Value	Description
47		1	0	Start bit
46		1	0	Transfer bit
[45: 40]		6	'111111'	Reserved
[39: 8] Parameter Domain	[31: 16]	16	X	RCA
	[15: 8]	8	X	Register Address
	[7: 0]	8	X	Read the contents of the register

[7: 1]	7	‘111111’	CRC7
0	1	1	End bit

31.5.6 R4b

Suitable for SD I/O cards only: An SDIO card will return a unique SDIO response R4 after receiving CMD5.

Table 31-25 R4b Response

Bit		Domain width	Value	Description
47		1	0	Start bit
46		1	0	Transfer bit
[45: 40]		6	X	Reserved
[39: 8] Parameter Domain	[39]	1	X	Card ready
	[38: 36]	3	X	Number of I/O functions
	[35]	1	X	Current memory
	[34: 32]	3	X	Padding bit
	[31: 8	24	X	I/O ORC
[7: 1]		7	X	Reserved
0		1	1	End bit

When an SD I/O card receives the command CMD5, the I/O portion of the card is enabled and can normally respond to all subsequent commands. The enabled state of the I/O card will remain until the next reset, power down, or receipt of the CMD52 command for I/O reset. Note that an SD card containing only memory functions can respond to the CMD5 command, and its correct response can be: current memory = 1, number of I/O functions = 0. An SD card designed according to the SD memory card specification version 1.0 containing only memory functions can detect the CMD5 command as an illegal command and do not respond to it. A host that can handle the I/O card will send a CMD5 command, and if the card returns a response R4, the host will determine the configuration of the card based on the data in the R4 response.

31.5.7 R5 (Interrupt Request)

For MMC only. Code length = 48 bits. If this response is generated by the host, the RCA domain in the parameter is 0x0.

Table 31-26 R5 Response

Bit		Domain width	Value	Description
47		1	0	Start bit
46		1	0	Transfer bit
[45: 40]		6	‘111111’	CMD40
[39: 8] Parameter Domain	[31: 16]	16	X	RCA of successful card or host [31: 16]
	[15: 0]	16	X	Undefined. Can be used as interrupt
[7: 1]		7	X	CRC7
0		1	1	End bit

31.5.8 R6 (Interrupt Request)

Works with SD I/O cards only. This is a normal response of a memory device to a CMD3 command.

Table 31-27 R6 Response

Bit		Domain width	Value	Description
47		1	0	Start bit
46		1	0	Transfer bit
[45: 40]		6	'101000'	CMD40
[39: 8] Parameter Domain	[31: 16]	16	X	RCA of successful card or host [31: 16]
	[15: 0]	16	X	Undefined. Can be used as interrupt data
[7: 1]		7	X	CRC7
0		1	1	End bit

When a CMD3 command is sent to a card with only I/O function, the status bit [23: 8] of the card changes; At this point, the 16 bits in the response will be the value in the SD card with only I/O function:

- Bit 15 = COM_CRC_ERROR
- Bit 14 = ILLEGAL_COMMAND
- Bit 13 = ERROR
- Bit [12: 0] = reserved

31.6 Programming sequence

31.6.1 Card identification

After the host is reset, it enters card recognition mode and looks for a new card on the bus. In card identification mode, the host resets all cards, verifies the operating voltage range, identifies the cards and asks for the relative card address (RCA) of each card. This operation is done separately on each card's own command signal line CMD. All data communications in card recognition mode use only the command signal line (CMD).

During card identification, the card should operate with a clock frequency of the clock rate FOD (400 kHz).

31.6.1.1 Card reset

The command GO_IDLE_STATE (CMD0) is a software reset command and sets the MMC and SD memory cards into an Idle State regardless of the current card status. The reset command (CMD0) is only used for the memory or memory portion of the combination card. To reset only the I/O portion of the I/O card or combination card, write 1 to the RES bit of the CCCR using CMD52. Cards in Inactive State are not affected by this command.

After the host is powered on, all cards are in Idle State (Idle State), including cards that have been in Inactive State before. After power-on or CMD0, the CMD line of all cards is in input mode, waiting for the start bit of the next command. These cards are initialized with the default relative card address (RCA) and driven with the default clock frequency of 400 kHz.

31.6.1.2 Operating voltage range verification

When communication begins between the host and the card, the host may not know the voltage the card supports, and the card may not know whether the host can provide the voltage it supports. To verify the voltage, the following commands are all defined in the relevant specifications. Commands

defined in the protocol specification include: SEND_OP_COND (CMD1 for MMC), SD_SEND_OP_COND (ACMD41 for SD memory cards), IO_SEND_OP_COND (CMD5 for SD I/O cards), which provide a mechanism for the host to identify and reject cards that do not match the VCC range required by the host. This is achieved by the host sending the required VCC voltage window as an operand of this command. If the card cannot transmit data within the specified range, it must be disconnected from the bus and entered into an Inactive State. Otherwise, the card will return its VCC range in response.

If the card can operate at the supplied voltage, the response will return the supply voltage and the check mode set in the command parameters. If the card does not work at the voltage supplied, it does not return a response and remains idle. When initializing the SDHC card, it is mandatory to send CMD8 before the ACMD41 command. Receiving CMD8 is to let the card know that the host supports the physical layer 2.00 protocol and the card supports higher version functions.

31.6.1.3 Card identification process

The identification process of the card is different for different cards. These cards include MMC, CE-ATA, SD, or SD I/O cards. All types of SD I/O cards are supported, namely, SDIO_IO_ONLY, SDIO_MEM_ONLY, and SDIO COMBO. The card identification process steps are as follows:

1. Check whether the card is connected.

2. Type of identification card: SD card, MMC (CE-ATA) or SD I/O card.

-Send a CMD5 command. If the host receives a response, it is the SD I/O card;

-If no response, send ACMD 41. If the host receives a response, it is the SD card;

-Otherwise, it is an MMC or CE-ATA device.

3. Initialize the card according to the card type.

Use FOD (400 KHz) as the clock source and send commands in the following order:

-SD card-transmit CMD0, ACMD41, CMD2, CMD3;

-SDHC card-transmit CMD0, CMD8, ACMD41, CMD2, CMD3;

-SD I/O card-if the card has no memory port, send CMD52, CMD0, CMD5, CMD3; Otherwise, transmit CMD52, CMD0, CMD5, ACMD41, CMD11 (optional), CMD2, CMD3;

-MMC/CE-ATA-transmit CMD0, CMD1, CMD2, CMD3.

Identify the MMC/CE-ATA device.

-The CPU should query 504 bytes of the EXT_CSD register (S_CMD_SET) by sending CMD8. If the 4th bit is set to 1, the device supports ATA mode;

-If ATA mode is supported, the CPU should select ATA mode by setting the (4th bit) ATA bit of the 191 bytes (CMD_SET) of the EXT_CSD register to activate the use of the ATA command set. The CPU selects the command set using the SWITCH (CMD6) command;

-If a CE-ATA device exists, the FAST_IO (CMD39) and RW_MULTIPLE_REGISTER (CMD60) commands will succeed and the data returned will be the CE-ATA reset signature.

31.6.2 No data command

When sending any data-less commands, the software needs to set the SDIO_CMD register and SDIO_ARG register with the appropriate parameters. Through these two registers, the host forms

commands and sends them onto the command bus. The host reflects the error of the command response through the error flag of the SDIO_STA register.

When the response is received, the host sets the SDIO_INTSTS register RCRC (response error) bit to 1. The short response is copied to SDIO_RESP0, while the long response is copied to all four response registers. The 31st bit of the SDIO_RESP3 register represents the highest bit of the long response, while the 0th bit of the SDIO_RESP0 register represents the lowest bit of the long response

31.6.3 Write data command

The single block or multi-block writing operation steps are as follows:

1. Set the data size (in bytes) in the SDIO_DLEN register.

2. Set the data block size in the SDIO_BLKSIZE register; The host sends a BLKSIZE sized block each time.

3. Set the address at which data should be written in the SDIO_CMDARG register.

4. Set the SDIO_CMD register. For SD memory cards and MMC cards, use the CMD24 command to write for a single block and the CMD25 command to write for multiple blocks. For SD I/O cards, the CMD53 command is used for single and multi-block transfers. For CEATA, first write the ATA task file with CMD60, and then write the data with CMD61 command. After writing to the CMD register, the host starts executing a command, and when the command is sent to the bus, the CD (Command Complete) flag is set.

5. Write data to SDIO_FIFO

6. The software should query the data error interrupt. If desired, the software can terminate the data transmission by sending a stop command (CMD12).

7. When a DLEN interrupt is received, the data transfer ends. For open block transfers, if the byte count is 0, the software must send the STOP command. If the byte count is not 0, the host should send a stop command at the end of the given byte count transfer

31.6.4 Single block or multiple block reads

The single block or multi-block reading operation steps are as follows:

1. Set the data size (in bytes) in the SDIO_DLEN register.

2. Set the data block size in the SDIO_BLKSIZE register; The host sends a BLKSIZE sized block each time.

3. Set the start address where the data needs to be read in the SDIO_CMDARG register.

4. Set the SDIO_CMD register. For SD and MMC cards, use CMD17 for single-block reading and CMD18 for multi-block reading. For SD I/O cards, use CMD53 for single-block and multi-block transfers. For CE-ATA, first write the ATA task file with CMD60, and then use CMD61 to read the data. After setting the CMD register, the host starts executing the command, and when the command is sent to the bus, the CD flag is set.

5. The software should query the data error interrupt. If desired, the software can terminate the data transmission by sending a stop command (CMD12).

6. The software should read data from the FIFO and free up space in the FIFO to receive more data.

7. When a DTEND interrupt is received, the software should read out the remaining data in the FIFO.

31.7 TIMx registers

31.7.1 SDIO power control register (SDIO_POWER)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PWRCTRL
-															RW

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	PWRCTRL	RW	0	PWRCTRL: power supply control bit Used to define the current functional status of the card clock: 0: The power is off and the clock of the card stops. 1: Reserved power-on state.

Note: This register cannot be written within 7 HCLK clock cycles after data is written.

31.7.2 SDIO Clock Control Register (SDIO_CLKCR)

Address offset: 0x04

Reset value: 0x0000 7000

The SDIO_CLKCR register controls the SDIO_CK output clock.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CKSEL	SMPEN	SMPCLKSEL	WIDBUS [1: 0]		PWRSAV	CLKEN	CLKDIV [7: 0]							
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 15	Reserved	-	-	-
14	CKSEL	RW	1	CMD & DAT Output clock selection: 0: SD_CLK clock offset 90 degrees output; 1: SD_CLK clock offset 180 degrees output;
13	SMPEN	RW	1	Presampling enable: 0: No pre-sampling clock is used; 1: Use a pre-sampling clock;
12	SMPCLKSEL	RW	1	CMD & DAT pre-sample clock selection (both rising edges): 1 SD_CLK clock 0: SD_CLK clock offset 270 degrees;
11: 10	WIDBUS	RW	2'h0	WIDBUS: wide bus mode enable bit 00: Default bus mode, using SDIO_D [0]. 01: 4-bit bus mode using SDIO_D [3: 0]. 10/11: 8-bit bus mode using SDIO_D [7: 0].
9	PWRSAV	RW	0	PWRSAV: power saving configuration bit To save power, setting the PWRSAV bit turns off the SDIO_CK clock output when the bus is idle. 0: SDIO_CK is always output. 1: Output SDIO_CK only when there is bus activity.
8	CLKEN	RW	0	CLKEN: clock enable bit 0: SDIO_CK is off.

				1: SDIO_CK enabled.
7: 0	CLKDIV	RW	8'h0	CLKDIV: clock frequency division coefficient, defines the frequency division coefficient between the input clock (SDIOCLK) and the output clock (SDIO_CK): SDIO_CK frequency = SDIOCLK/[CLKDIV * 2]

Note:

1 When the SD/SDIO card or MMC is in recognition mode, the frequency of SDIO_CK must be lower than 400KHz.

2 When all cards are assigned corresponding addresses, the clock frequency can be changed to the maximum frequency allowed by the card bus.

3 This register cannot be written within 7 HCLK clock cycles after data is written. For SD I/O cards, SDIO_CK may be stopped during read wait, at which time the SDIO_CLKCR register does not control SDIO_CK.

31.7.3 SDIO parameter register (SDIO_ARG)

Address offset: 0x08

Reset value: 0x0000 0000

The SDIO_ARG register contains 32-bit command parameters that will be sent to the card as part of the command.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMDARG															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMDARG															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	CMDARG	RW	32'h0	CMDARG: command parameters A command parameter is a part of a command sent to the card. If a command contains an argument, this register must be loaded before the command is written to the command register.

31.7.4 SDIO Command Register (SDIO_CMD)

Address offset: 0x0C

Reset value: 0x2000 0000

The SDIO_CMD register contains the command index and command type bits. The command index is sent to the card as part of the command. Command Type Bit Control Command Channel State Machine (CPSM)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
START CMD	Res			BOOTMODE	BOOTDIS	BOOTACK	BOOTEN	IEN	ATACMD	REGSYNC	Res				
RW	-			RW	RW	RW	RW	RW	RW	RW	-				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AUTIINIT	ABORTCMD	WAITPEND	AUTOSTOP	DTMODE	DIR	DEXPECT	CHECKRESPCRC	RESPLEN	WAITRESP	CMDINDEX [5: 0]					
RW															

Bit	Name	R/W	Reset Value	Function
31	STARTCMD	RW	0	Start command. Once the command is received by the CIU, the bit is cleared. When this bit is set, the host cannot write to the command register. If a write is attempted, the hardware lock error is set in the interrupt status register. Once the command is sent and a response is received from the SD_MMC_CEATA card, the command complete bit is set in the interrupt status register.
30: 28	Reserved	-	-	-
27	BOOTMODE	RW	0	Start Mode 0: Forced start operation 1: Standby start operation
26	BOOTDIS	RW	0	Disable_boot Disable startup. "BOOTDIS" and "BOOTEN" cannot be set at the same time.
25	BOOTACK	RW	0	Start response expected. When this bit is set together with BOOTEN, the CIU is given an expected response.
24	BOOTEN	RW	0	Enable startup This bit should only be set to forced start mode. When the software sets this bit and start_cmd together, the CIU starts by determining whether the boot line of the corresponding card is pulled low. "BOOTDIS" and "BOOTEN" cannot be set at the same time.
23	IEN	RW	0	the interrupt enable 0: No interrupt enabled in CE-ATA device 1: Enable interrupt on CE-ATA device (nIEN = 0),
22	ATACMD	RW	0	ATACMD 0: The host did not perform read access to the CE-ATA device 1: The host performs read access to the CE-ATA device.
21	REGSYNC	RW	0	0: Normal command sequence 1: Don't send commands, just update clock register values to card clock domain. Change clock frequency or stop clock without sending commands to the card
20: 16	Reserved	-	0	-
15	AUTOINIT	RW	0	0: No initialization sequence is sent before this command is sent (80 clocks) 1: Send initialization sequence before sending this command
14	ABORTCMD	RW	0	0: Do not end the command that is currently performing data transfer. 1: The command to end the data transmission currently in progress.
13	WAITPEND	RW	0	If this position is bit, the command state machine needs to wait for the data transmission to end before starting to send commands. 0: No effect 1: Waiting for data transmission to end
12	AUTOSTOP	RW	0	0: No stop command is sent at the end of data transmission 1: Send stop command at the end of data transmission
11	DTMODE	RW	0	0: Data block transfer command 1: Data stream transmission command
10	DIR	RW	0	0: card reading 1: Write card

9	DEXPECT	RW	0	0: No expected data transfer (read/write) 1: There is expected data transfer (read/write)
8	CHECKRESPCRC	RW	0	0: Response CRC not checked 1: Check response CRC Some command responses do not return valid CRC bits. The software should disable CRC checking for these commands to disable CRC checking for the controller.
7	RESPLEN	RW	0	0: Short response 1: Long response
6	WAITRESP	RW	0	0: Not waiting for response 1: Waiting for response
5: 0	CMDINDEX	RW	6'h0	Command Index

Note: 1: This register cannot be written within 7 HCLK clock cycles after data is written.

2: The MMC can send two kinds of responses: a short response with a length of 48 bits or a long response with a length of 136 bits. SD cards and SD I/O cards can only send short responses, the parameters can vary according to the type of response, and the software will distinguish the type of response according to the command sent. CE-ATA devices send only short responses.

31.7.5 SDIO command response register (SDIO_RESPCMD)

Address offset: 0x10

Reset value: 0x0000 0000

The SDIO_RESPCMD register contains the command index in the last received command response. If the transmitted command response does not contain the command index (long response or OCR response), the content of the RESPCMD field is unknown, although it should contain 111111b (reserved field value in response).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESPCMD [5: 0]					
-										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5: 0	RESPCMD	R	6'h0	RESPCMD: Command index of response The command index in the last received command response.

31.7.6 SDIO command response register (SDIO_RESPX)

Address offset: 0x14 + 4 * (x-1), where x = 1.. 4

Reset value: 0x0000 0000

The SDIO_RESP1/2/3/4 register contains the status of the card, i.e. part of the received response.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CARDSTATUSx [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CARDSTATUSx [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	CARDSTATUSx	R	32'h0	CARDSTATUSx: See table below

Depending on the response state, the state length of the card is 32 bits or 127 bits.

Table 31-28 response type and SDIO_RESPx register

Register	Short response	Long response
SDIO_RESP1	Card status [31: 0]	Card Status [127: 96]
SDIO_RESP2	need not	Card Status [95: 64]
SDIO_RESP3	need not	Card Status [63: 32]
SDIO_RESP4	need not	Card status [31: 1]

The highest bit of the card status is always received first, and the lowest bit of the SDIO_RESP3 register is always 0.

31.7.7 SDIO Data Timer Register (SDIO_TMOUT)

Address offset: 0x24

Reset value: 0xFFFFFF40

The register contains the data timeout in units of card bus clock cycles.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATATIME [23: 8															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATATIME [7: 0								RESPTIME [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

When DPSM is in these states, if the counter decreases to 0, a timeout flag is set.

Bit	Name	R/W	Reset Value	Function
31: 8	DATATIME [23: 0	RW	24 'hFFFFFF	DATATIME: Data timeout Data timeout in units of card bus clock cycles.
7: 0	RESPTIME [7: 0]	RW	8'h40	RESPTIME: Command timeout time The command timeout in card bus clock cycles.

Note: The data timer register and the data length register must be written before writing to the data control register for data transfer.

31.7.8 SDIO block length register (SDIO_BLKSIZE)

Address offset: 0x28

Reset value: 0x0000 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBLOCKSIZE [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	DBLOCKSIZE	RW	16'h200	DBLOCKSIZE: data block length When the block data transmission mode is selected, the field defines the data block length

31.7.9 SDIO data length register (SDIO_DLEN)

Address offset: 0x2C

Reset value: 0x0000 0200

The SDIO_DLEN register contains the length of data bytes to be transferred. When the data transfer starts, this value is loaded into the data counter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATALENGTH															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATALENGTH [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	DATALENGTH	RW	32'h200	Data length

Note: For block data transfer, the value in the data length register must be a multiple of the data block length (SDIO_BLKSIZ). Before writing to the data control register for data transfer, you must first write the data timer register and the data length register.

31.7.10 SDIO Control Register (SDIO_CTRL)

Address offset: 0x30

Reset value: 0x01000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							ODPUEN	Res							
-							RW	-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	CEATAINTEN	AUTOSTOPCCSD	CCSDEN	ABORTRD	AUTOIRQRESP	READWAIT	DMAEN	INTEN	Res.		FIFORST	SDIORST
-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24	ODPUEN	RW	1	CMD online pull-up: 0: No pull-up 1: pull-up
23: 12	Reserved	-	-	-
11	CEATAINTEN	RW	0	0: In CE-ATA device, interrupt is not enabled; 1: In CE-ATA device, interrupt is enabled;
10	AUTOSTOPCCSD	RW	0	0: No STOP is transmitted after CCSD is transmitted to CE-ATA; 1: After sending CCSD to CE-ATA, automatically send STOP; After sending, the bit is automatically cleared;
9	CCSDEN	RW	0	0: CCSD is not transmitted to CE-ATA 1: Send CCSD to CE-ATA;
8	ABORTRD	RW	0	0: No change 1: Issue a suspend command during data reading, reset the data state machine and wait for the next data block.
7	AUTOIRQRESP	RW	0	0: No change 1: Send IRQ response Once the response is sent, this bit is automatically cleared.

6	READWAIT	RW	0	0: No read waiting 1: There is read waiting Used to send a read wait to the SDIO card.
5	DMAEN	RW	0	0: Disable DMA transfer mode 1: Enable DMA transfer mode
4	INTEN	RW	0	Global interrupt enable/disable bit: 0: Disable interrupt 1: Enable interrupt The global interrupt port is 1 only if the bit is 1 and there are one or more unmasked interrupt bits set
3: 2	Reserved	-	-	-
1	FIFORST	RW	0	0: No change 1: Reset FIFO To reset the FIFO, this bit should be set to 1. Clears automatically after completing the reset operation
0	SDIORST	RW	0	0: No change 1: Reset controller To reset the controller, the bit should be set to 1. This bit is cleared automatically.

31.7.11 SDIO status register (SDIO_STA)

Address offset: 0x34

Reset value: 0x0000 0006

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		FIFOCNT [12: 0]													Res
-		RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.						CARDBSY	CARDPRESENT	CMDFSM [3: 0]				FIFOF	FIFOE	TXWMARK	RXWMARK
-						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	-	-
29: 17	FIFOCNT	R	13'h0	FIFO Count
16: 10	Reserved	-	-	-
9	CARDBSY	R	0	Card status 0: Idle 1: Busy
8	CARDPRESENT	R	0	Check if the card exists 0: Card does not exist 1: Card presence

7: 4	CMDFSM	R	4'h0	Command FSM status: 0: Idle 1: Send init sequence 2: Tx cmd Start bit 3: Tx cmd Tx bit 4: Tx cmd Index + Parameters 5: Tx cmd crc7 6: Tx cmd End bit 7: Rx Response start bit 8: Rx response IRQ response 9: Rx response tx bit 10: Rx response cmd index 11: Rx Response data 12: Rx corresponds to crc7 13 Rx End of response bit 14: Cmd path waiting NCC 15: Wait
3	FIFOF	R	0	FIFO full
2	FIFOE	R	1	FIFO empty
1	TXWMARK	R	1	TXFIFO reaches output water mark
0	RXWMARK	R	0	RXFIFO reaches receive water line

31.7.12 SDIO Interrupt Status Register (SDIO_INTSTS)

Address offset: 0x38

Reset value: 0x0000 0000

At the corresponding position '1', the bit of SDIO_INTSTS is cleared.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															SDIOINT
-															RC_W1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EBE	ACD	SBE	HLE	FRUN	HTO	DRTO_BDS	RTO_BAR	DCRC	RCRC	RXDR	TXDR	DTO	CD	RE	CAD
RC_W1															

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	SDIOINT	RC_W1	0	0: No SDIO interrupts from the card 1: There is an SDIO interrupt from the card
15	EBE	RC_W1	0	End bit error/no CRC error (EBE)
14	ACD	RC_W1	0	Automatic Command Completion (ACD)
13	SBE	RC_W1	0	Start bit error (SBE)
12	HLE	RC_W1	0	Hardware Lock Write Error (HLE)
11	FRUN	RC_W1	0	FIFO overflow error (FRUN)
10	HTO	RC_W1	0	Host Timeout Error (HTO)
9	DRTO_BDS	RC_W1	0	Data Read Timeout (DRTO_BDS)
8	RTO_BAR	RC_W1	0	Response timeout (RTO_BAR)
7	DCRC	RC_W1	0	Data CRC Error (DCRC)
6	RCRC	RC_W1	0	Response CRC Error (RCRC)
5	RXDR	RC_W1	0	Receive a FIFO Data Request (RXDR)
4	TXDR	RC_W1	0	Send FIFO Data Request (TXDR)
3	DTO	RC_W1	0	Data Transfer (DTO)
2	CD	RC_W1	0	Command Complete (CD)
1	RE	RC_W1	0	Response Error (RE)
0	CAD	RC_W1	0	Card Detection (CAD)

31.7.13 SDIO Interrupt Mask Register (SDIO_INTMASK)

Address offset: 0x3C

Reset value: 0x0000 0000

At the corresponding position '1', the interrupt mask register determines which state bit interrupt can be generated.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SDIOINTIE
-															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEIE	ACDIE	SBEI	HLEIE	FRUNIE	HTOIE	DRTOIE	RTOIE	DCRCIE	RCRCIE	RXDRIE	TXDRIE	DTOIE	CDIE	REIE	CADIE
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	SDIOINTIE	RW	0	SDIO interrupt enable 0: Disabled 1: Enabled
15	BEIE	RW	0	End bit error/no CRC error (EBE) interrupt enable 0: Disabled 1: Enabled
14	ACDIE	RW	0	Automatic Command Complete (ACD) Interrupt Enable 0: Disabled 1: Enabled
13	SBEI	RW	0	Start bit error (SBE) interrupt enable 0: Disabled 1: Enabled
12	HLEIE	RW	0	Hardware locked write error (HLE) interrupt enable 0: Disabled 1: Enabled
11	FRUNIE	RW	0	FIFO overflow error (FRUN) interrupt enable 0: Disabled 1: Enabled
10	HTOIE	RW	0	Host Timeout Error (HTO) Interrupt Enable 0: Disabled 1: Enabled
9	DRTO_BDSIE	RW	0	Data Read Timeout (DRTO_BDS) Interrupt Enable 0: Disabled 1: Enabled
8	RTO_BARIE	RW	0	Response Timeout (RTO_BAR) Interrupt Enable 0: Disabled 1: Enabled
7	DCRCIE	RW	0	Data CRC Error (DCRC) Interrupt Enable 0: Disabled 1: Enabled
6	RCRCIE	RW	0	Response CRC Error (RCRC) Interrupt Enable 0: Disabled 1: Enabled
5	RXDRIE	RW	0	Receive FIFO Data Request (RXDR) interrupt enable 0: Disabled 1: Enabled
4	TXDRIE	RW	0	Send FIFO Data Request (TXDR) interrupt enable 0: Disabled 1: Enabled
3	DTOIE	RW	0	Data Transfer (DTO) Interrupt Enable 0: Disabled 1: Enabled
2	CDIE	RW	0	Command Complete (CD) Interrupt Enable 0: Disabled 1: Enabled
1	REIE	RW	0	Response Error (RE) Interrupt Enable

				0: Disabled 1: Enabled
0	CADIE	RW	0	Card Detection (CAD) Interrupt Enable 0: Disabled 1: Enabled

31.7.14 SDIO FIFO Threshold Register (SDIO_FIFOTH)

Address offset: 0x40

Reset value: 0x000F 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				RXWMARK [11: 0]											
-				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				TXWMARK [11: 0]											
-				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 16	RXWMARK [11: 0]	RW	12 'hf	Receive the water mark setting of the FIFO. When the FIFO data count exceeds this number, a read data request is generated. When packet reception is completed, a request is generated to complete the remaining data reading regardless of the water mark setting.
15: 12	Reserved	-	-	-
11/0	TXWMARK [11: 0]	RW	12 'hf	Send the water mark setting of the FIFO. When the FIFO data count is less than or equal to this number, write data requests are issued.

31.7.15 SDIO sent to card counter (SDIO_TCBCNT)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TCBCNT [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCBCNT [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	TCBCNT [31: 0]	R	32'h0	The number of bytes transferred to the card.

31.7.16 SDIO sent to FIFO counter (SDIO_TBBCNT)

Address offset: 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBBCNT [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBBCNT [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	TCBCNT [31: 0]	R	0	The number of bytes of the FIFO received or sent.

31.7.17 SDIO Data FIFO Register (SDIO_FIFODATA)

Address offset: 0x200

Reset value: 0x0000 0000

A receive and send FIFO is a 32-bit wide read or write set of registers that contains 32 registers on 32 consecutive addresses.

Note: The data fifo register includes sending and receiving fifos, so these 32 addresses should be divided into groups of 16, with sending and receiving 50%. At each time of reading and writing, data of the size of the receiving FIFO or the writing transmitting FIFO is read at most.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIFODATA [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FIFODATA [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	FIFODATA [31: 0]	RW	32'h0	FIFODATA: Receive or send FIFO data FIFO does not support simultaneous reading and writing from one port.

32. USB OTG full speed (OTG_FS)

32.1 Overview

OTG_FS is a Dual Role Device (DRD) controller that supports both device and host functionality and is fully compliant with the On-The-Go supplementary standard of the USB 2.0 specification. Additionally, the controller can also be configured in "Host Only" mode or "Device Only" mode, fully compliant with USB 2.0 specifications. In host mode, OTG_FS supports full-speed (FS, 12 Mb/s) and low-speed (LS, 1.5 Mb/s) transceivers, while in device mode only full-speed (FS, 12 Mb/s) transceivers are supported. OTG_FS supports both HNP (Host Negotiation Protocol) and SRP (Session Request Protocol). The only external device needed in host mode is a charge pump that provides VBUS.

32.1.1 Main features

The following will introduce the characteristics of OTG_FS controller from three aspects: general functions, host mode functions and device mode functions.

32.1.1. 1 General Features

- Certified by USB-IF, Universal Serial Bus Specification, Revision2.0 standard
- The OTG protocol defined in the (physical layer of the OTG_FS controller, i.e. PHY) USB On-The-Go Supplement, Revision 1.3 specification is fully supported.
  - Identification of inserted Class A-B devices (ID lines)
  - Supports Host Negotiation Protocol (HNP) and Session Request Protocol (SRP)
  - In OTG applications, allow the host to turn off VBUS to save power consumption
  - The OTG controller monitors the VBUS level using an internal comparator
  - Host/device roles can be dynamically switched
- You can configure the following roles through software:
  - USB full-speed devices supporting SRP protocol (Class B devices)
  - USB full-speed/low-speed host supporting SRP protocol (Class A device)
  - USB OTG full-speed dual-role device
- SOF signal supporting full-speed communication and keep-active signal supporting low-speed communication
  - The pulse of the SOF can be output to pin
  - The SOF is internally connected to Timer 2 (TIM2)
  - Configurable frame period
  - Configurable end of frame interrupt
- Has power saving functions, such as stopping the system during USB suspension, turning off the digital module clock, and managing PHY and DFIFO power.
- 1.25 KB of dedicated RAM with advanced FIFO control
  - Configure different RAM areas for different FIFOs through software for flexible and effective use of RAM
  - Each FIFO can store multiple data packets
  - Allow dynamic allocation of storage areas
  - It is not limited that the length of the FIFO must be a power of 2 value to allow flexible and

efficient use of RAM

- The maximum data flow of one frame (1 ms) can be guaranteed without the intervention of the system

32.1.1. 2 Host mode functions

The OTG_FS interface has the following main features and requirements in host mode:

- Requires an external charge pump to power the VBUS
- Supports up to 8 host channels, each channel can be dynamically configured for any transmission type
- Built-in hardware scheduling controller:
  - Supports up to 8 interrupt and synchronous transfer requests in a periodic hardware transfer request queue
  - Supports up to 8 control and bulk transfer requests in the aperiodic hardware transfer request queue.
- In order to efficiently use the RAM space, the data RAM area of the USB is divided into a shared receive FIFO, a periodic transmit FIFO, and an aperiodic transmit FIFO.

32.1.1. 3 Device mode functionality

The OTG_FS interface has the following characteristics in device mode:

- 1 two-way control endpoint 0
- 3 IN endpoints for batch, interrupt, or synchronous transfer
- 3 OUT endpoints for batch, interrupt, or synchronous transfer
- Manage a shared receive FIFO and a send OUT FIFO for efficient use of USB data RAM
- Manage up to 4 dedicated sending IN FIFOs (one FIFO for each IN endpoint) to reduce application load
- Software disconnect function is supported.

32.1.2 Structural block diagram

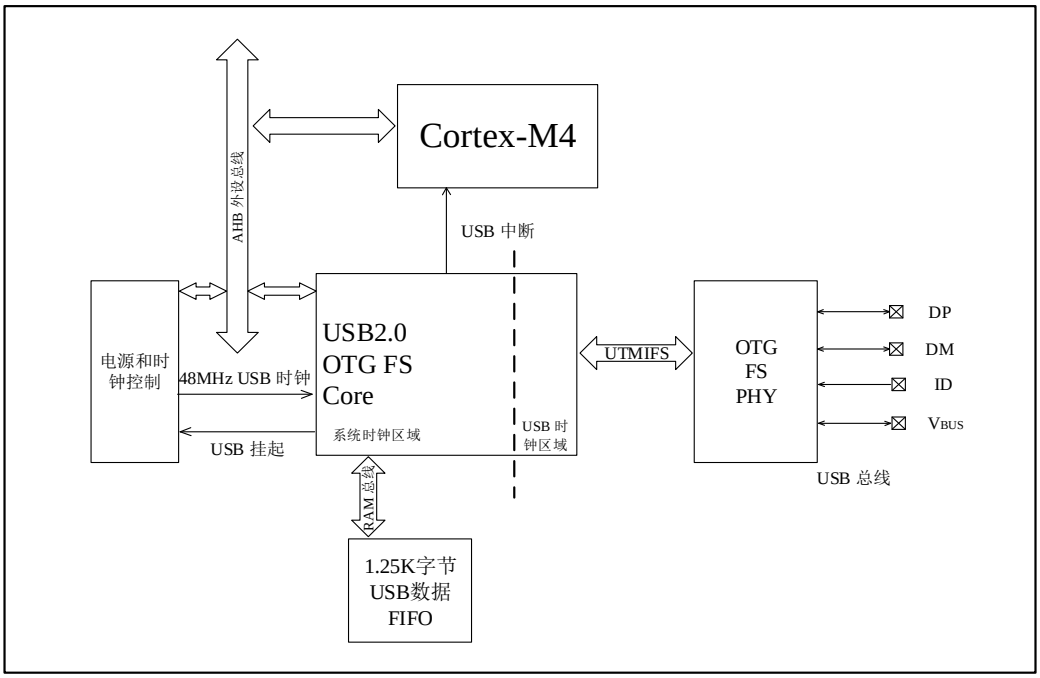


Figure 32-1 OTG_FS structural block diagram

32.2 Functional description

32.2.1 OTG Full Speed Controller

The USB OTG full speed controller obtains a 48 MHz ± 0. 25% clock from the Reset and Clock Control Module (RCC). This clock is used to drive the 48 MHz control clock of the USB full speed module (12 Mbit/s). This control clock must be enabled before the OTG full speed controller is configured.

The microcontroller core (CPU) accesses the registers of the OTG full speed controller through the AHB peripheral bus, and USB events are managed by a separate USB OTG interrupt control line.

The microcontroller transmits data to the USB controller by writing 32 bits of data to the OTG_FS dedicated address (PUSH register), which will be automatically stored in the USB configured send FIFO. Each transmitting FIFO is configured with such a PUSH register to serve each IN endpoint (device mode) or OUT channel (host mode).

The microcontroller obtains 32-bit data from the USB bus by reading the dedicated address (POP register) of the OTG_FS, which will be automatically loaded from the shared receive FIFO. The receive FIFO is located in a total 1.25 Kbyte USB data RAM area. Each OUT endpoint or IN channel is served by such a POP register.

The USB protocol layer is driven by the Serial Interface Engine (SIE) and connected to a USB full/low speed transceiver module supported by the built-in physical layer (PHY).

32.2.2 Full Speed OTG PHY (Physical Interface)

The built-in full-speed OTG PHY is controlled by the OTG full-speed controller and sends and receives control and data signals through the full-speed subset of the UTMI + bus (UTMIFS). PHY provides physical layer support for USB communication.

The OTG PHY at full speed includes the following parts:

- Full/low speed transceiver module for use in host and device modes. This module drives sending and receiving directly on a single-ended USB cable.
- Built-in pull-up resistor for ID line is used to distinguish class A/B devices.
- The DP/DM line has built-in pull-up and pull-down resistors, which are controlled by the OTG_FS controller to meet the needs of current equipment types. In device mode, when an active level appears on the VBUS line (Class B active), the controller enables the pull-up resistor of the DP line to notify the host to access a USB full-speed device. In host mode, the controller enables the pull-down resistors of both DP and DM lines. The pull-up and pull-down resistors can be dynamically switched when the controller switches role types through the Host Negotiation Protocol (HNP).
- ECN circuit for pull-up/pull-down resistors. The DP line has two pull-up resistors that are separately controlled by the OTG_FS controller, which meets the ECN requirements of the USB 2.0 version (Engineering Change Notice). Supports dynamic adjustment of the pull-up strength of DP lines, which can better resist noise and improve the quality of transmitted and received signals.
- Built-in VBUS detection comparator with hysteresis function is used to detect the active level, A-B session active level and session end level of VBUS. These levels are used to drive the Session Request Protocol (SRP), detect valid session start and end conditions, and continuously detect VBUS line status in USB communication.

- The VBUS pulse circuit is used for charge/discharge operation (weak drive) of the VBUS through the resistor during SRP.

Note: To ensure proper operation of the USB OTG FS module, the frequency of the AHB should be greater than or equal to 30 MHz.

32.2.3 OTG Dual Role Device (DRD)

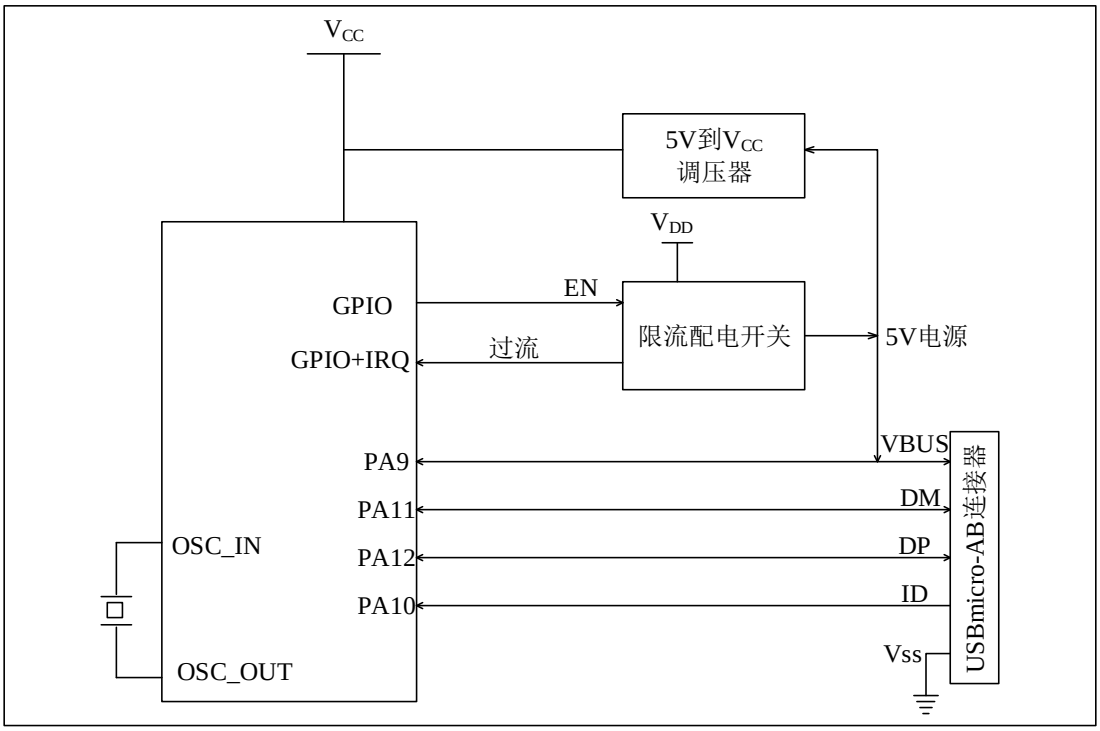


Figure 32-2 A-B Device Connections for OTG_FS

32.2.3.1 ID signal detection

The roles of the host and device (default) are defined by the status of the ID line. When you plug in the USB, you can determine the ID cable status based on which end of the USB cable is connected to the micro-AB socket.

- If the B end of the USB cable is plugged in and the ID line is floating, the built-in pull-up resistor will detect the high level of the ID line, at which time the controller is in the default device mode. In this mode, the OTG_FS controller complies with the description of the standard FSM in Section 6.8. 2: OTG Class B Devices of the OTG 1.3 specification, a supplement to the USB 2.0 specification.
- If the A end of the USB cable is plugged in and the ID cable is grounded. At this point, the OTG_FS controller will execute an ID line state change interrupt (CIDSCHG bit of the OTG_FS_GINTSTS register), automatically switch to host mode, and require software to initialize host mode. In this mode, the OTG_FS controller complies with the description of standard FSMs in Section 6.8. 1: OTG Class A Devices of the OTG 1.3 specification, which supplements the USB 2.0 specification.

32.2.3.2 HNP dual-role device

The HNP enable bit of the USB global configuration register (the HNPCAP bit of the OTG_FS_GUSBCFG register) allows the OTG_FS controller to dynamically switch between a class A host and a class A device, and between a class B host and a class B device through a host negotiation

protocol (HNP). The status of the current device can be synthesized by reading the ID status bit of the OTG global control and status register (CIDSTS bit of the OTG_FS_GOTGCTL register) and the current operation mode bit of the global interrupt and status register (CMOD bit of the OTG_FS_GINTSTS register).

32.2.3.3 SRP dual-role device

The SRP enable bit of the USB global configuration register (the SRPCAP bit of the OTG_FS_GUSBCFG register) allows the OTG_FS controller to turnoffVBUS power, savingpower for-ClassA devices. Note: Class A devices need to control the power supply of the management VBUS whether in host mode or device mode.

32.2.4 USB device mode

This chapter describes the functionality of the OTG_FS controller in USB device mode.

The OTG_FS controller operates in device mode when:

- OTG B Class equipment
  - When the B end of the USB cable is plugged in, the default is OTG Class B device mode
- OTG A Class equipment
  - The OTG Class A device switches to the device role via the HNP protocol
- Class B equipment
  - When the ID line exists, the B end of the USB cable is plugged in, and the HNP bit of the USB global configuration register
- When (HNPCAP bit of OTG_FS_GUSBCFG register) is 0 (refer to Section 6.8. 3 of OTG version 1.3)
- As a USB device only
  - The force device mode bit of the USB global configuration register (the FDMOD bit of the OTG_FS_GUSBCFG register) of 1 will force the OTG_FS controller to operate in USB device mode (refer to Section 6.8. 3 of OTG version 1.3). In this case, the status of the ID line is ignored regardless of whether it exists or not.

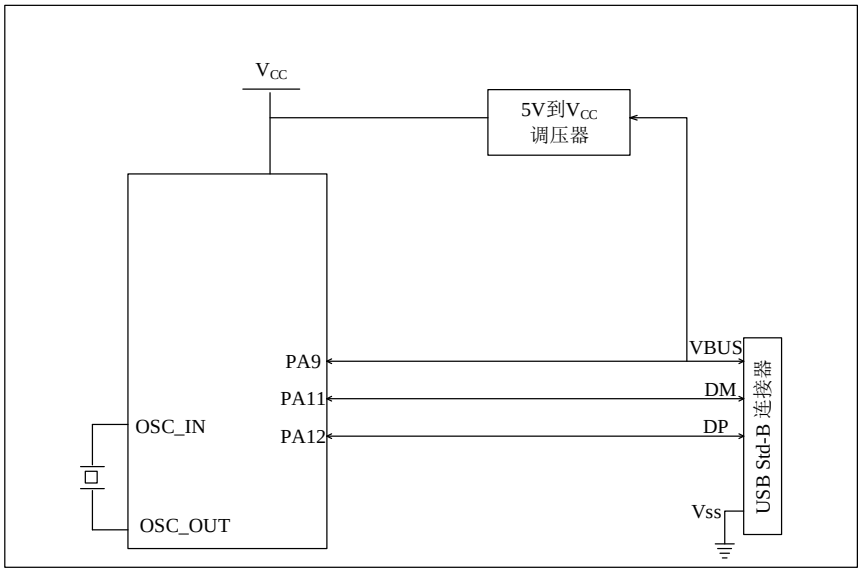


Figure 32-3 USB is used for peripheral connection only

32.2.4.1 SRP-capable devices

The SRP enable bit of the USB global configuration register (the SRPCAP bit of the OTG_FS_GUSBCFG register) allows the OTG_FS controller to implement the Session Request Protocol (SRP). In this case, the Class A device can stop supplying power to the VBUS while the USB session is suspended.

32.2.4.2 Device Status

Power-on state

After the valid level of the Class B device is detected on the VBUS line, the USB device enters the power-on state. The OTG_FS controller will automatically enable the pull-up resistor on the DP line to inform the host that a full-speed device is connected, and generate a session request interrupt (SRQINT bit of the OTG_FS_GINTSTS register) to mark the power-on state.

The host needs to provide a valid VBUS level throughout the USB communication phase. If it is detected that the level on the VBUS line is below the Class B active level (e.g., a power supply disturbance occurs, or the host power is turned off), the OTG_FS controller will automatically disconnect the USB connection and generate an end-of-session interrupt (SEDET bit of the OTG_FS_GOTGINT register), marking the controller to exit the power-on state.

In the power-on state, the OTG_FS controller expects to receive a reset signal from the host, and other USB operations are invalid. When the reset signal is received, a reset interrupt (USBRST bit of the OTG_FS_GINTSTS register) is generated. After the reset is complete, an enumeration interrupt (ENUMDNE bit of the OTG_FS_GINTSTS register) is generated, indicating that the OTG_FS controller has entered the default state.

Software disconnect

You can use software to configure the exit power-on state. Setting the software disconnect bit of the device control register (the SDIS bit of the OTG_FS_DCTL register) removes the pull-up resistor on the DP line, allowing the host to recognize a device disconnect event even when the USB cable is still connected.

Default status

By default, the OTG_FS controller expects to receive the SET_ADDRESS command, at which time all other operations are invalid. When a valid SET_ADDRESS command is received, the application needs to write the relevant address into the device address bit of the device configuration register (the DAD bit of the OTG_FS_DCFG register). The OTG_FS controller enters the address state and is ready to respond to host transmissions to this address.

Pending status

The OTG_FS controller continuously detects activity on the USB cable. After detecting the USB idle state for 3 ms, an early suspension interrupt (ESUSP bit of the OTG_FS_GINTSTS register) is generated, and a suspension interrupt (USBSUSP bit of the OTG_FS_GINTSTS register) is generated after 3 ms to acknowledge the suspension. The device suspend bit of the device status register (the SUSPSTS bit of the OTG_FS_DSTS register) is automatically set and the OTG_FS controller enters the suspend state.

The device can spontaneouslyexitthe suspended state by setting the remote wake-up signal bit of the device control register (the WKUPINT bit of the OTG_FS_DCTL register) and clearing this bit between 1 ms and 15 ms.

When it is detected that the host sends a recovery signal,arecovery interrupt (RWUSIG bit of the OTG_FS_GINTSTS register) will be generated, and the suspend bit of the devicewill be automatically cleared.

32.2.4.3 Device endpoint

The OTG_FS controller supports the following USB endpoints:

- **Control Endpoint 0**
  - Bidirectional endpoint, used only for processing control information
  - For both IN and OUT directions, there is an independent set of registers to control
  - Includes a control (OTG_FS_DIEPCTLO/OTG_FS_DOEPCTLO) register, a transfer configuration (OTG_FS_DIEPTSIZO/OTG_FS_DOEPTSIZO) register, and a status interrupt (OTG_FS_DIEPINTO/OTG_FS_DOEPINTO) register. Certain bits in the control and transfer length registers differ from the registers of other endpoints.
- **3 IN endpoints**
  - Each endpoint can be configured for synchronous, bulk, or interrupt transmission
  - Each endpoint has a control (OTG_FS_DIEPCTLx) register, a transfer configuration (OTG_FS_DIEPTSIZx) register, and a status interrupt (OTG_FS_DIEPINTx) register
  - The device IN endpoint universal interrupt mask register (OTG_FS_DIEPMSK) is used to enable/disable any type of endpoint interrupt source for all IN endpoints, including endpoint 0
  - Supports an outstanding synchronous IN transmission interrupt (IISOIXFR bit of the OTG_FS_GINTSTS register), which occurs when at least one synchronous IN transmission is outstanding IN the current frame. This interrupt occurs along with a periodic frame interrupt (EOPF of the OTG_FS_GINTSTS register)
- **3 OUT endpoints**
  - Each endpoint can be configured for synchronous, bulk, or interrupt transmission
  - Each endpoint has a control (OTG_FS_DOEPCTLx) register, a transfer configuration (OTG_FS_DOEPTSIZx) register, and a status interrupt (OTG_FS_DOEPINTx) register
  - The device OUT endpoint universal interrupt mask register (OTG_FS_DOEPMSK) is used to enable/disable any type of endpoint interrupt source for all OUT endpoints, including endpoint 0
  - Supports an outstanding synchronous OUT transmission interrupt (INCOMPISOOUT bit of the OTG_FS_GINTSTS register), which occurs when at least one synchronous OUT transmission is outstanding in the current frame. This interrupt occurs along with a periodic frame interrupt (EOPF of the OTG_FS_GINTSTS register)

Endpoint configuration

- The following configuration of endpoints is achieved by setting the endpoint IN/OUT control register (OTG_FS_DIEPCTLx/OTG_FS_DOEPCTLx):

- Endpoint enable/disable
- Activate endpoints in the current configuration
- Configure USB transfer type (synchronous, batch, interrupt)
- Configure USB packet length
- Configure the transmit FIFO number associated with the IN endpoint
- Configure the PID packets DATA0/DATA1 expected to be received or sent (valid for bulk and interrupt transmissions only)
- Configure odd/even frames for transactions to be sent or received (only valid for synchronous transmission)
- Optional configuration: Set the NAK bit to respond to the host NAK regardless of the FIFO status
- Optional configuration: Set the STALL bit to respond to STALL for any command sent by the host
- Optional configuration: Set the OUT endpoint to listen (SNOOP) mode, and do not verify CRC for received data

Endpoint Transport

The device endpoint transfer length register (OTG_FS_DIEPTSIZx/OTG_FS_DOEPTSIZx) is used to configure the transfer length and read the transfer state. This register must be configured before setting the enable bit of the port control register. Once the endpoint is enabled, the register is in a read-only state, and only the OTG_FS controller can update the transfer status bit of the register.

- The transmission parameters that need to be configured are as follows:
  - Length of a single transfer in bytes
  - Number of USB packets completing the entire transmission

Endpoint Status/Interrupt

The device endpoint interrupt register (OTG_FS_DIEPINTx/OTG_FS_DOEPINTx) indicates the event status of the endpoint related to USB and AHB. When the OUT endpoint interrupt bit or IN endpoint interrupt bit of the controller interrupt register (OEPINT bit and IEPINT bit of the OTG_FS_GINTSTS register) is set, the application program needs to first read all endpoint interrupt registers of the device (OTG_FS_DAINT), determine the endpoint number where the interrupt occurred, and then read the endpoint interrupt register of the device to obtain detailed information of the interrupt. The application must immediately clear the corresponding interrupt bits of the registers to clear the corresponding interrupt bits of the OTG_FS_DAINT registers and the OTG_FS_GINTSTS registers.

- The device controller provides the following status bits and interrupt events:
  - The transmission completion interrupt indicates that the data transmission in both AHB and USB is complete.
  - SETUP phase completed (valid for control endpoints only)
  - The relevant transmission FIFO is already half empty or fully empty (IN endpoint)
  - NAK response sent to host (valid for synchronous IN transmission only)
  - When the send FIFO is empty, the host makes an IN request (only valid for bulk IN transmission and interrupt IN transmission)

- The host makes an OUT request when the endpoint is not enabled
- Crosstalk error detected
- The application closed endpoint
- The application sets the endpoint state to NAK (only valid for synchronous IN transmissions)
- Received more than 3 consecutive SETUP packets (valid for control OUT transmission only)
- Timeout error detected (valid for control IN transmission only)
- Synchronous transmission type packets are lost without interruption

32.2.5 USB host

This chapter describes the functions of the OTG_FS controller when operating in USB host mode. The OTG_FS controller operates in host mode when:

- OTG A Class host
  - The A end of the USB cable is plugged in, and the default is OTG class A host mode
- OTG B Class host
  - The OTG Class B device switches to the host role via the HNP protocol
- Class A equipment
  - When the ID line is present, the A end of the USB cable is plugged in, and the HNP bit of the USB global configuration register is 0
- (HNPCAP bit of OTG_FS_GUSBCFG register). At this time, the pull-up resistor on the built-in DP/DM line is automatically active.
- As a host only
  - The force host mode bit of the USB global configuration register (the FHMOD bit of the OTG_FS_GUSBCFG register) enables the USB host mode in which the OTG_FS controller operates. At this time, even if the ID cable on the USB connector exists, it is still invalid. The pull-up resistor on the built-in DP/DM line is automatically active.

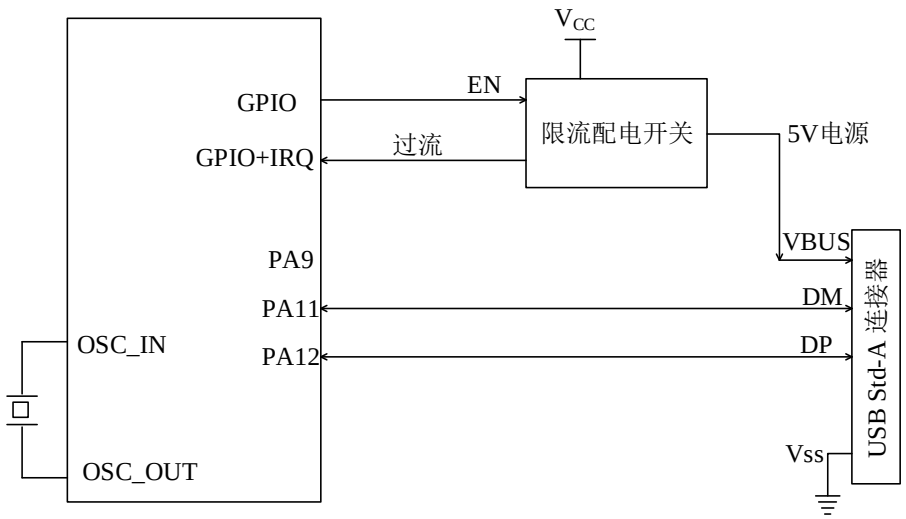


Figure 32-4 USB is only used for host connection

32.2.5.1 SRP-enabled hosts

The SRP function can be supported by configuring the SRP enable bit of the USB global configuration register (the SRPCAP bit of the OTG_FS_GUSBCFG register). When the SRP function is enabled, the host can save power by turning off the power supply of VBUS when the USB session is suspended.

32.2.5.2 USB host status

Host port power supply

The controller does not provide 5 V power supply to VBUS internally, so an external charge pump needs to be used to power the VBUS line. If 5 V power supply can be provided on the application board, an ordinary power switch can also be used. This external charge pump can be controlled through ordinary I/O ports. When the application program needs to set this common I/O port to power the VBUS line, it also needs to set the power supply bit of the host port control and status register (PPWR bit of the OTG_FS_HPRT register).

VBUS enabled

During the entire USB communication process, the VBUS line needs to be guaranteed to be at an active level at all times. Any event that causes the level of the VBUS line to drop below the threshold (4.25 V) will cause an OTG interrupt (SEDET bit of the OTG_FS_GOTGINT register) triggered by the session abort bit. At this point, the application needs to turn off power to the VBUS and clear the port power level. The overcurrent signal of the charge pump can also be used to prevent electrical damage. Connect this overcurrent signal to the I/O port set as the interrupt input. Once an overcurrent event occurs, the application needs to immediately turn off the power supply of VBUS during the interrupt and clear the port power supply potential.

Monitoring of device access by host

If the VBUS line has no active level (4.75 V) at all times, the OTG_FS controller cannot detect the device plugging in whenever a USB device or Class B device is plugged in.

When the VBUS line is within the active level range, once a Class B device is inserted, the OTG_FS controller generates a host port interrupt triggered by the device insertion bit (PCDET bit of the OTG_FS_HPRT register).

Monitoring of device disconnection by host

Disconnecting the inserted device will generate a disconnect interrupt (DISCINT bit of the OTG_FS_GINTSTS register) triggered by the device disconnect event.

Host enumeration

Once the host detects that a device is plugged in and needs to enter the enumeration process, it sends USB reset and configuration commands to the newly plugged device. Because the inserted device has a pull-up resistor on the DP (full speed device) or DM (low speed device) line, it will cause bus instability. When the bus returns to a stable state again, it will trigger an OTG interrupt (DBCDNE bit of the OTG_FS_GOTGINT register) for the end of the rebound, and then the host can send a USB reset command to the device.

The application sends a USB reset signal (single-ended 0) to the USB bus by setting the port reset bit of the host port control and status register (PRST bit of the OTG_FS_HPRT register). The application program needs to ensure that the reset signal is maintained between 10ms and 20ms, and clear the port reset bit in time.

Once the USB reset sequence is complete, a host port interrupt (PENCHNG bit of the OTG_FS_HPRT register) is generated that is triggered by the port enable/disable change bit. This interrupt notifies the application that it can obtain the speed information of the enumerated device (PSPD bit of the OTG_FS_HPRT register) from the host port control and status register, whereupon the host will send a SOF (full speed device) signal or remain active (low speed state). At this time, the host can send a configuration command to the device to complete the device enumeration.

Host suspended

The application can suspend USB activity by setting the port suspend bits of the host control and status registers (the PSUSP bit of the OTG_FS_HPRT register). At this time, the OTG_FS controller will stop sending the SOF signal and enter the suspended state.

The controller may optionally support the remote device to wake up the host to exit the suspended state. At this time, after the controller detects the remote wake-up signal, it will generate a remote wake-up interrupt (WKUPINT bit of the OTG_FS_GINTSTS register), and then, the wake-up bits of the host port control and status register (PRES bit of the OTG_FS_HPRT register) will be automatically set, and the controller will automatically send a wake-up signal to the USB bus. The application needs to control the maintenance time of the wake-up signal, and clear the port wake-up bit in time to exit the suspended state and restart sending the SOF signal.

If the host needs to spontaneously exit the suspended state, the application program can set the port wake-up bit. At this time, the wake-up signal will be sent to the USB bus. The application program needs to control the time for which this wake-up signal is maintained and clear the port wake-up bit in time.

32.2.5.3 Host channel

The OTG_FS controller provides 8 host channels, each corresponding to a USB host transfer (USB PIPE). The host does not support initiating more than 8 transmission requests at the same time. If the application has more than eight unprocessed transfer requests, the host controller driver (HCD) must reschedule the channel after the channel is re-valid, that is, after receiving the interrupt of transfer completion and channel abort.

Each channel can be configured as input/output mode or as any periodic/aperiodic transmission type. Each channel has a control register (OTG_FS_HCCHARx), a transfer configuration register (OTG_FS_HCTSIZx) and a status/interrupt register (OTG_FS_HCINTx) and a corresponding mask register (OTG_FS_HCINTMSKx).

Host channel control

- The channel control register (OTG_FS_HCCCHARx) may provide the following operations:
  - Channel enable/disable
  - Configure transfer speed for connected USB devices: Full/Low

- Configure addresses for connected USB devices
- Configure endpoint numbers for connected USB devices
- Configure transmission direction: input/output
- Configure transmission type: control, batch, interrupt, synchronous
- Configure maximum packet length (MPS)
- Configure odd/even frames for periodic transmission

Host channel transmission

The application configures the transfer length through the host channel transfer length register (OTG_FS_HCTSIZx) and reads back the transfer state. You must configure the transmission length before enabling a transmission channel. Once the channel is enabled, the transfer length register becomes read-only, and only the OTG_FS controller can update this register with the current transfer state.

- The following transmission parameters can be configured:
  - Transmission length of a single packet in bytes
  - The number of all packets in the entire transmission process
  - Starting data PID

Host channel status/interrupt

The host channel x interrupt register (OTG_FS_HCINTx) indicates the channel's USB and AHB related events. When the host channel interrupt bit of the host controller interrupt register (HCINT bit of the OTG_FS_GINTSTS register) is set, the application program needs to first read all channel interrupt registers of the host (OTG_FS_HCAINT) to obtain the channel number that generated the host channel interrupt, and then read the host channel x interrupt register (OTG_FS_HCINTx) according to the channel number to obtain interrupt information. The application needs to clear the corresponding bits of the host channel x interrupt register (OTG_FS_HCINTx) in order to clear the corresponding bits of the OTG_FS_HAINT and OTG_FS_GINTSTS registers. The interrupt source for each channel can be masked by the OTG_FS_HCINTMSKx register.

- The host controller provides the following status queries and interrupts:
  - Transfer completion interrupt, indicating that data transfer has been completed on both the application side (AHB) and USB side.
  - Channel abort due to transfer completion, USB communication error, or application prohibit command
  - The corresponding transmission FIFO is already half empty or fully empty (valid for IN channel only)
  - ACK response received
  - NAK response received
  - Received STALL response
  - USB transfer error caused by CRC error, timeout, bit padding error, or erroneous EOP
  - Crosstalk error
  - Frame overflow
  - Data PID (DATA0/DATA1) flip error

32.2.5.4 Host scheduler

The host controller has a built-in hardware scheduler, which is used to spontaneously manage and drive application-initiated transmission requests. At the beginning of each frame, the host will process periodic (synchronous and interrupted transmissions) data transmissions before aperiodic (control and bulk transmissions) data transmissions in order to comply with the high priority guarantee of synchronous and interrupted transmissions of the USB specification.

The host manages the USB transmission channel by request queues (one periodic queue and one aperiodic queue). Each request queue can hold 8 requests, and each request represents an unprocessed transfer made by the application. Each request IN the request queue carries the IN/OUT direction, channel number and other information to perform a USB transfer. The order in which USB transaction requests are written in the queue determines the order in which they are executed on the USB bus. At the beginning of each frame, the host will process the periodic request queue first, and then the aperiodic request queue. If, at the end of a frame, there is still an unprocessed synchronization or interrupt transmission request, an incomplete periodic transmission interrupt (IPXFR bit of the OTG_FS_GINTSTS register) will be generated.

The management of periodic queues and aperiodic queues is completely done by the OTG_FS controller. The application can obtain the status information of each request queue through read-only registers.

- The periodic transfer FIFO and queue status register (OTG_FS_HPTXSTS) and the aperiodic transfer FIFO and queue status register (OTG_FS_GNPTXSTS) include:
  - Number of requests that can also be inserted in periodic and aperiodic queues (up to 8)
  - Periodically (aperiodically) transmit the remaining free space in the FIFO (for OUT transmission)
  - IN/OUT commands, number of host channels, and other status information

Since each request queue can hold up to 8 requests, applications can make the best use of this feature and submit up to 8 periodic transmission requests and 8 aperiodic transmission requests to improve transmission efficiency. For example, for block transmission OUT/IN data, the application can configure to transmit up to 64 (maximum number of bytes per packet) × 8 (maximum number of requests) = 512 bytes of data, achieving the highest data transfer rate for a full-speed device. This data will be automatically scheduled by the host without requiring application insertion management.

- The application initiates a periodic (aperiodic) OUT transmission request to the host scheduler through the following steps:
  - Configure transport parameters for host channels
  - Enable configured channels
  - Read the HPTXSTS bit of the OTG_FS_GNPTXSTS register to ensure that the periodic (aperiodic) queue can still hold at least 1 request.
  - The OTG_FS_HPTXSTS (OTG_FS_GNPTXSTS) register is read to ensure that sufficient space remains in the periodic (aperiodic) transmit FIFO. This step may be omitted if the application submits the transmission request after receiving a periodic (aperiodic) transmission

of the FIFO mid-air or full-air break.

— Data is loaded into the corresponding FIFO (PUSH register). Each channel is configured with a PUSH register. Data will be automatically loaded into the corresponding periodic or aperiodic transmit FIFO according to the EPTYPE bit of the OTG_FS_HCCHARx register. When the last 32 bits of data is written to the FIFO, a request will be inserted at the end of the request queue, waiting for scheduled execution.

- The application initiates a periodic (aperiodic) IN transmission request to the host schedule through the following steps:
  — Configure transport parameters for host channels
  — Set the channel enable bit of the host channel control register (the CHENA bit of the OTG_FS_HCCHARx register) to enable the configured channel. This operation will insert a transport request at the end of the periodic (aperiodic) request queue and wait for scheduled execution.

32.2.6 SOF Trigger

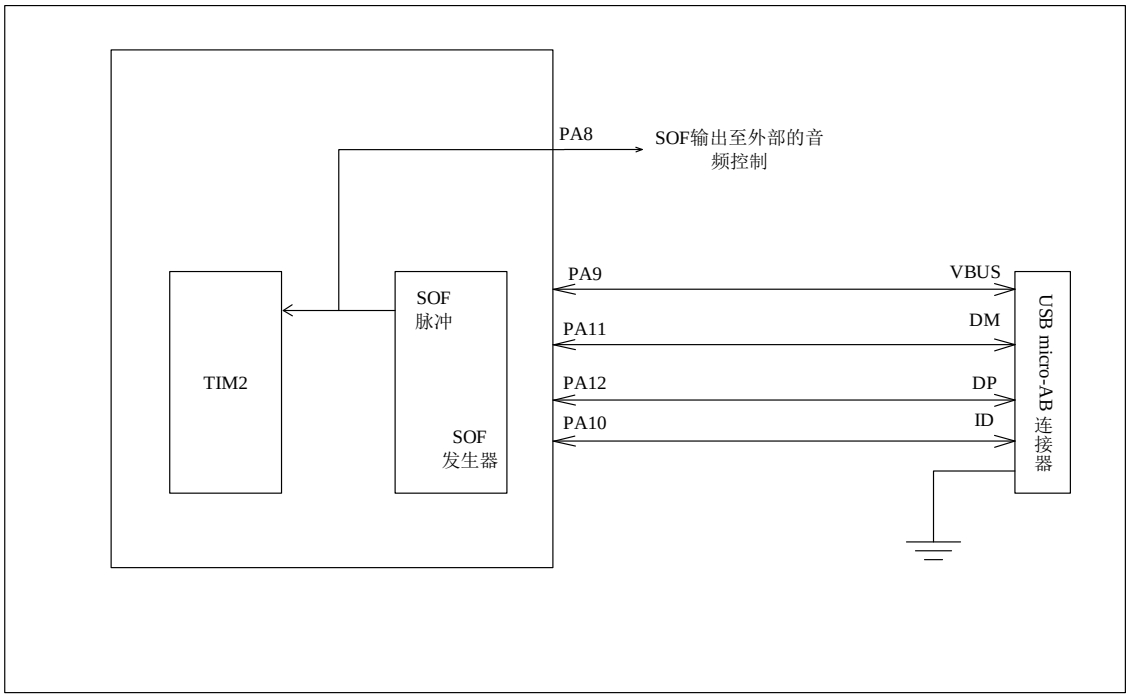


Figure 32-5 SOF Connection

The OTG_FS controller provides:

- Monitoring, tracking and configuration of SOF
- SOF pulse output

These features are useful for the clock output of audio applications because the audio device needs to be synchronized with the data stream of the PC, or because the host needs to adjust the frame rate according to the needs of the audio device

32.2.6.1 Host SOF

In host mode, the number of clocks between two consecutive SOFs (full speed devices) or hold active (low speed devices) can be set by configuring the host frame interval register (OTG_FS_HFIR), so the application can use it to control the SOF frame period. An interrupt (SOF bit of the OTG_FS_GINTSTS register) may be generated in each frame capital. Both the current frame number and time remain unchanged until the next frame number appears in the host frame number register (OTG_FS_HFNUM).

Each SOF frame will generate a SOF pulse signal with a width of 12 system clock cycles. Through the SOFOEN bit of the global control and configuration register, this pulse can be configured to be output on the SOF pin. The SOF pulse is also internally connected to the trigger input of timer 2 (TIM2), so input capture, output comparison, and timer can all be triggered by the SOF pulse.

32.2.6.2 Device SOF

In device mode, a frame head interrupt (SOF bit of the OTG_FS_GINTSTS register) will be generated each time a SOF is received on the USB cable. The current frame number (FNSOF bit of the OTG_FS_DSTS register) may be read from the device status register. At this time, a SOF pulse signal with a width of 12 system clock cycles will be generated. This pulse signal can be output on the SOF pin by enabling the SOF output enable bit of the global control and configuration register (SOFOEN bit of the OTG_FS_GCCFG register). The SOF pulse signal is also internally connected to the trigger input of timer 2 (TIM2), at which time input capture, output comparison and timer can all be triggered by the SOF pulse. The periodic frame end interrupt (EOPF bit of the OTG_FS_GINTSTS register) is used to inform the application that 80%, 85%, 90%, or 95% of the frame has been completed, depending on the frame interval bit of the device configuration register (PFIVL bit of the OTG_FS_DCFG register). This function is used to determine whether all synchronous transmissions in a frame have been completed.

32.2.7 Power supply options

The power consumption of the OTG PHY is determined by the following 3 bits of the general controller configuration register:

- PHY supply bit (PWRON bit of OTG_FS_GCCFG register)
  - Switch on/off of the PHY full speed transceiver module. Pre-setting is required to allow USB communication.
- Class A VBUS monitoring enable bit (VBUSACEN bit of OTG_FS_GCCFG register)
  - Toggle on/off the VBUS comparator for Class A devices. In Class A device mode (USB host), it is necessary to enable
  - Can deserve it.
- Class B VBUS monitoring enable bit (VBUSBCEN bit of OTG_FS_GCCFG register)
  - Switch that toggles the VBUS comparator for Class B devices. In Class B device mode (USB device), it is necessary to enable
  - Can deserve it.

When USB has no communication or no USB device connection, suspend USB, which can save power.

- Stop PHY clock (STPPCLK bit of OTG_FS_PCGCCTL register)
  - When the stop PHY clock bit of the clock gating control register is set, most of the 48MHz clock internally connected to the OTG full speed controller is turned off by gating. At this time, even though the application still enables the 48MHz input clock, the dynamic power consumption caused by the USB clock will be greatly reduced.
  - Most transceivers will be disabled, but the circuitry for detecting asynchronous reset signals and remote wake-up events will remain at
  - Activation status.
- Gated HCLK (GATEHCLK bit of OTG_FS_PCGCCTL register)
  - When the gated HCLK bit of the clock gating register is set, most of the system clock internally connected to the OTG_FS controller is turned off by the gating circuit. Only the access interface of the register remains valid. At this point, even though the application still enables the system clock, the dynamic power consumption caused by the USB clock will be greatly reduced.
- USB System Stop
  - When the OTG_FS controller is in the USB suspended state, the application needs to completely shut down all system clock sources to completely reduce all power consumption. Set the PHY clock stop bit first, and then set the Power Supply Control System Module (PWR) to enter the system deep sleep mode, which will stop the USB system.
  - Upon detection of a remote wake-up (as a host) or a wake-up signal from the USB bus (as a device), the OTG_FS controller will automatically resume system activity and the USB clock.

To reduce power consumption, the USB data FIFO is clocked only when the OTG_FS controller needs to access the FIFO.

32.2.8 USB Data FIFO

The following diagram shows a block diagram of the OTG_FS controller and the functions of each part.

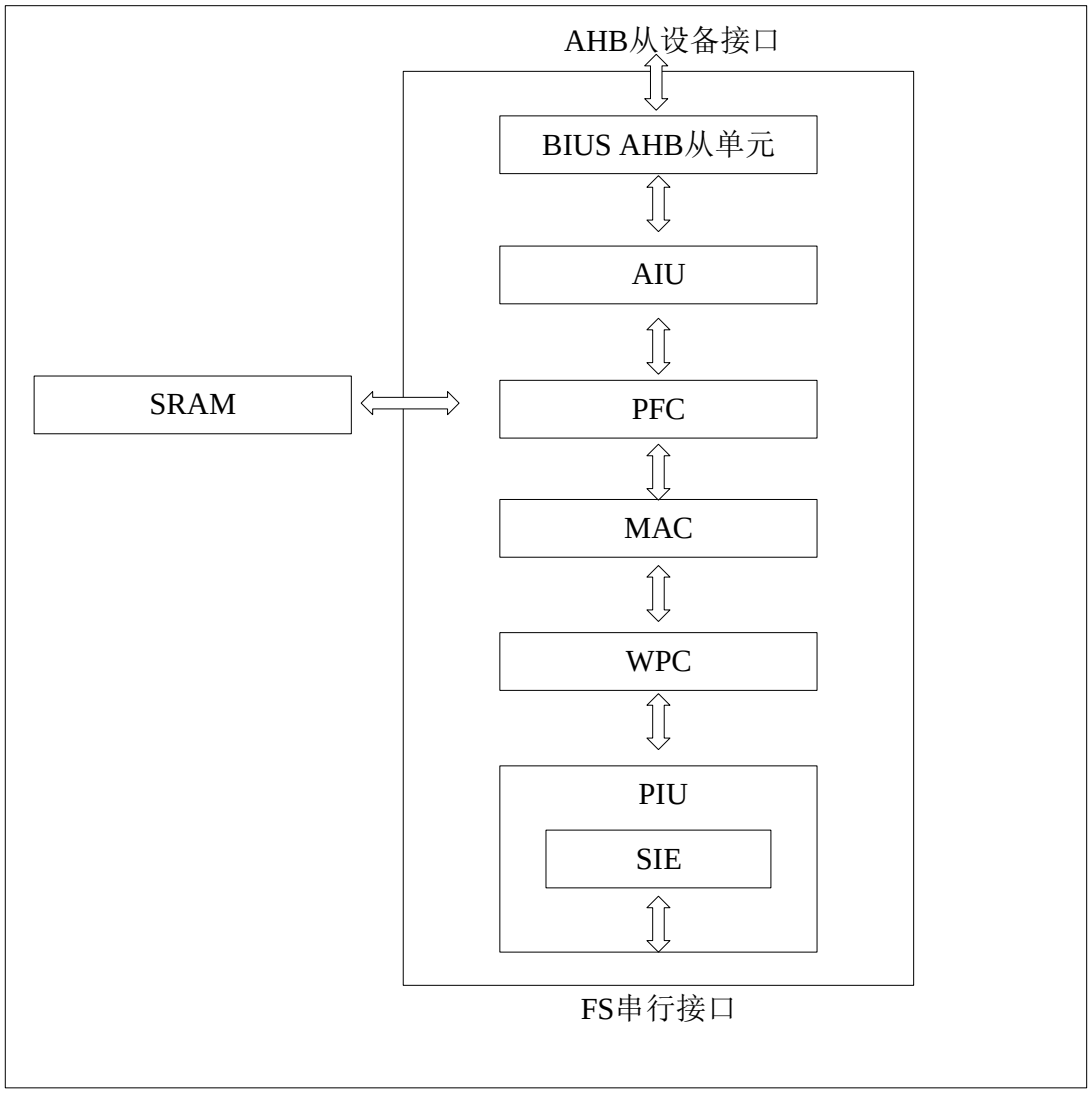


Figure 32-6 OTG_FS Controller Block Diagram

BIUS: bus interface unit; AIU: Application Program Interface Unit; PFC: package FIFO controller; MAC: media access controller; WPC: wake-up and power-up controller; PIU: PHY interface unit; SIE: serial interface controller

The USB system plans 1.25 K bytes of dedicated RAM for FIFO management. The packet FIFO controller (PFC) module of the OTG_FS controller divides this RAM into a sending FIFO area and a receiving FIFO area. The application temporarily caches data into the sending FIFO and waits to be sent, while the data received from the USB bus is temporarily cached in the receiving FIFO and waits to be read by the application. The actual number of FIFOs and how these FIFOs are planned in dedicated RAM are determined by the actual application. For example, IN device mode, each IN endpoint will be configured with a sending FIFO, and the size of these FIFOs can be specified by the software. IN this way, the dedicated RAM can optimally meet the needs of the application.

32.2.9 FIFO Structure in Device Mode

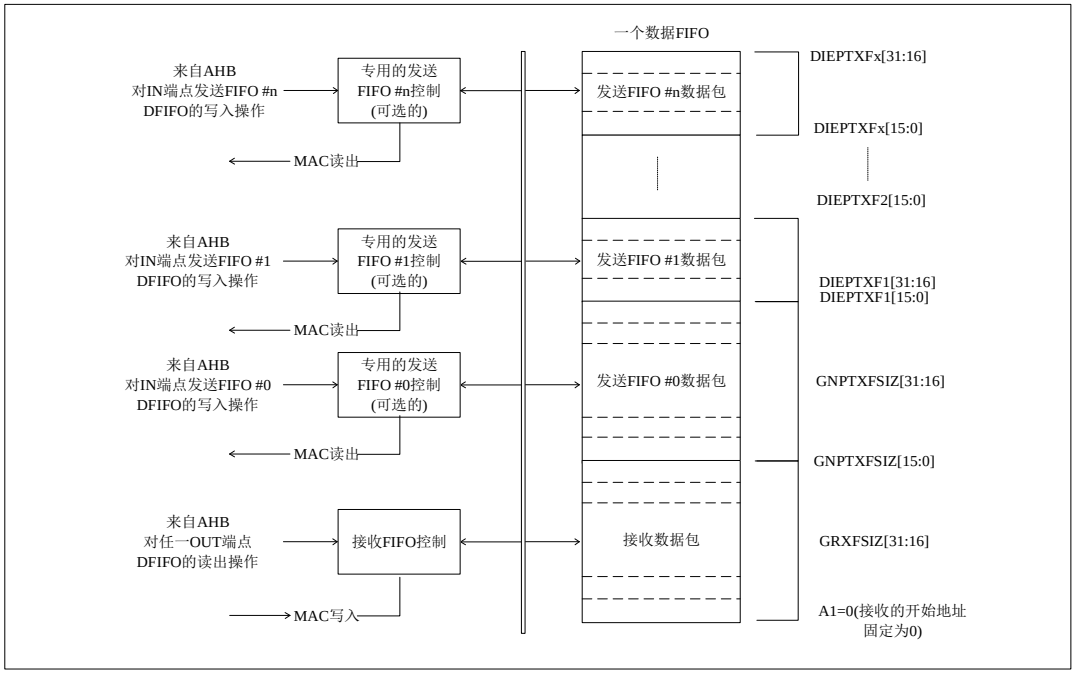


Figure 32-7 FIFO address image in device mode and FIFO operation image of AHB

32.2.9.1 Receive FIFO in device mode

In device mode, the FS_OTG controller uses a separate FIFO to cache data for all OUT endpoints. The received packets are in turn cached in the available space of the RX FIFO. The status of the received data packet (including the OUT endpoint number, byte number, data PID number and validity of the received data) is written by the PFC module before the received data. If there is no space left in the receiving FIFO, the controller will respond to the host with a NACK and generate an interrupt at the corresponding endpoint. The size of the receive FIFO may be configured by the receive FIFO length register (OTG_FS_GRXFSIZ).

The architecture of using a single RX FIFO enables USB devices to make more effective use of the entire receiving RAM space.

- All OUT endpoints share the entire RAM space (shared FIFO)
- The OTG_FS controllercan utilize the receive FIFOto the maximum extentin the order in whichthe host sends OUTdata

As long as the receiving FIFO has at least one packet waiting to be fetched by the application, the application will continue to receive the receiving FIFO non-null interrupt (RXFLVL bit of the OTG_FS_GINTSTS register). The application program obtains packet information by reading the receive status and eject registers (OTG_FS_GRXSTSP), and can obtain packets from the receive FIFO by reading the POP register of the corresponding endpoint.

32.2.9.2 Send FIFO in device mode

The shared FIFO mode is not suitable for IN transmission. Only by knowing the order of host requests IN advance or predicting the processing procedure of the process can all endpoint packets be transmitted to the same sending FIFO IN order. So IN device mode, the controller configures a dedi-

cated FIFO for each IN endpoint. The application program may configure the FIFO length of IN endpoint 0 by aperiodic transfer FIFO length register (OTG_FS_GNPTXFSIZ), and configure the FIFO length of IN endpoint x by device IN endpoint transfer FIFO register (OTG_FS_DIEPTXFx).

The architecture of the dedicated FIFO is very flexible, which greatly reduces the load of the application. These FIFOs have no request sequence, and there is no need to predict the order of access when the USB host accesses the aperiodic endpoint.

According to the value of the aperiodic transmit FIFO null level bit (TXFELVL bit of the OTG_FS_GAHBCFG register) configured IN the AHB configuration register, the OTG_FS controller generates a transmit FIFO null interrupt (NPTXFE bit of the OTG_FS_GINTSTS register), indicating that the transmit FIFO corresponding to the IN endpoint is half empty or completely empty. The application first obtains the IN endpoint number to be served by reading the interrupt register of all endpoints of the device (OTG_FS_DAINT), then checks whether the FIFO has enough remaining space by reading the sending FIFO status register of the IN endpoint x of the device (OTG_FS_DTXFSTSx), and finally transmits data to the sending FIFOx by writing the PUSH register of the corresponding endpoint.

32.2.10 FIFO Structure in Host Mode

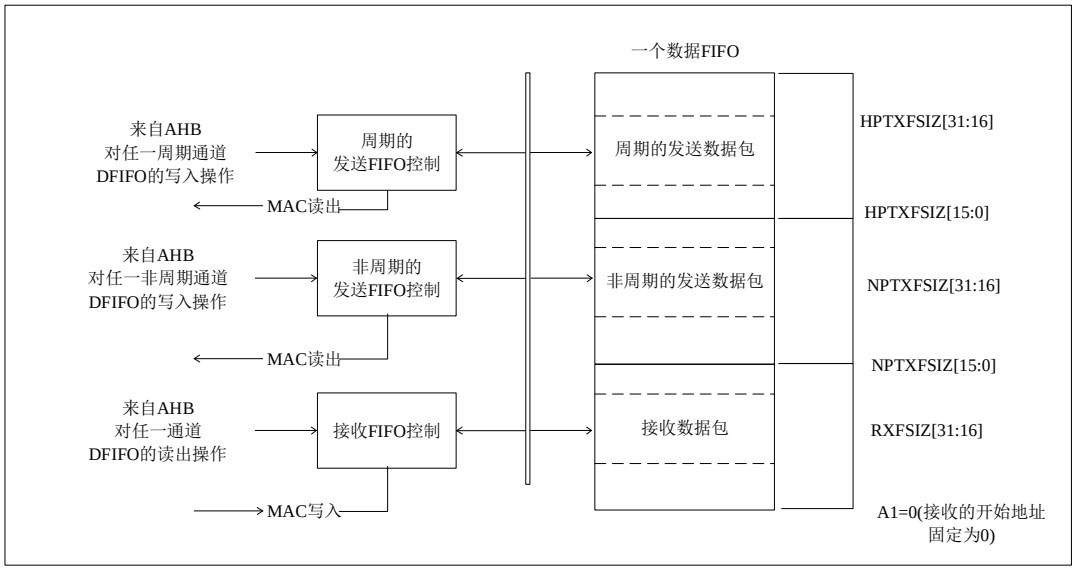


Figure 32-8 Host Mode FIFO Address Image and FIFO Operation Image of AHB

32.2.10.1 Receive FIFO in Host Mode

In host mode, all periodic and aperiodic transmissions are managed using a receive FIFO, which is used to temporarily store data (received data packets) received from the USB bus but not yet transferred to the system storage area. Packets from any remote IN endpoint are temporarily stored IN the FIFO IN order. The status of the received packet (including the host channel number, byte number, data PID and validity of the received data) is also stored in the FIFO. The application may configure the length of this received FIFO through the received FIFO length register (OTG_FS_GRXFSIZ).

The architecture of using a single receiving FIFO enables the USB host to make more effective use of the entire receiving RAM space.

● All configured IN channels share the entire RAM space (shared FIFO)

● The OTG_FS controller can utilize that receive FIFO to the maximum extent according to the sequence of IN commands sent by the host

The application receives a receive FIFO non-null interrupt as long as the receive FIFO has at least one packet waiting to be fetched by the application. The application program can obtain packet information by reading the receive status and eject registers, and obtain packets from the receiving FIFO by reading the POP register of the corresponding endpoint.

32.2.10.2 Send FIFO in Host Mode

In host mode, the controller uses one transmit FIFO to manage all aperiodic (control and block transmission) OUT transmissions and the other transmit FIFO to manage all periodic (sync and interrupt) OUT transmissions. These two FIFOs are used to temporarily store data (transmitted packets) to be transmitted to the USB bus. The length of the periodic (aperiodic) transmission FIFO may be configured by the host periodic (aperiodic) transmission FIFO length register (OTG_FS_HPTXFSIZ/OTG_FS_GNPTXFSIZ).

Two transmitting FIFOs are used because it is necessary to ensure the high priority of periodic transmissions in a USB frame. At the beginning of each frame, the built-in host scheduler processes the periodic request queue first and then the aperiodic request queue.

The use of two sending FIFOs makes it convenient for the USB host to optimize the management of periodic and aperiodic data caches separately.

- All host channels used for periodic (aperiodic) OUT transmissions share the same RAM buffer area (shared FIFO)
- The OTG_FS controllercan maximizethe use of periodic(aperiodic)transmission FIFOaccordingto the scheduling of the OUT commandby the host software.

According to the value of the periodic transmission FIFO null flag bit (PTXFELVL bit of the OTG_FS_GAHBCFG register) configured in the AHB configuration register, the OTG_FS controller generates a periodic transmission FIFO null interrupt (PTXFE bit of the OTG_FS_GINTSTS register) to indicate that the periodic transmission FIFO has been half empty or fully empty. The application needs to read the periodic send FIFO and queue status register (OTG_FS_HPTXSTS) first to obtain the information of whether the periodic send FIFO and the periodic request queue have remaining space. Only when both the send FIFO and the request queue have remaining space, the application can write data into the FIFO.

According to the value of the aperiodic transmission FIFO null level bit (TXFELVL bit of the OTG_FS_GAHBCFG register) configured in the AHB configuration register, the OTG_FS controller generates an aperiodic transmission FIFO null interrupt (NPTXFE bit of the OTG_FS_GINTSTS register), indicating that the aperiodic transmission FIFO has been half empty or fully empty. The application needs to read the aperiodic send FIFO and queue status register (OTG_FS_GNPTXSTS) first to obtain the information of whether there is room left in the aperiodic send FIFO and aperiodic request queue. Only when there is room left in both the send FIFO and the request queue, the application can write data into the FIFO.

32.2.11 USB System Performance

By configuring large-capacity RAM buffer area, high-degree-of-freedom FIFO length configuration, 32-bit fast access of AHB PUSH/POP registers, and advanced FIFO control mechanism, the OTG_FS controller can maximize the utilization of RAM space without worrying about the order of USB data, and the USB system achieves the best performance.

These advantages are as follows:

- Reduced application load, no interference with USB transmission, optimized CPU bandwidth usage

— When data is efficiently transmitted through the USB system, the application program can prepare a large amount of transmission data in advance.

— The application gets a lot of time to get data from a single receiving FIFO

- The USB module can keep working at full speed, that is, provide the maximum full speed bandwidth (as much hardware as possible runs automatically and as little software as possible participates).

— The USB controller manages a large data cache area and can autonomously manage the transmission of data on the USB bus

— The USB controller manages a large data receiving cache area, which can autonomously receive data from the USB bus and fill the cache area

Since the OTG_FS controller can use the 1.25 K bytes of RAM buffer very efficiently, and 1.25 K bytes is sufficient for managing the transmitted/received data for a full speed frame, the USB system can provide the maximum full speed data transfer rate within each USB frame (1 ms).

32.2.12 OTG_FS Interrupt

Regardless of whether the OTG_FS controller operates in device mode or host mode, applications cannot access register banks in the other mode. If the application has illegal access, a pattern mismatch interrupt will be generated and the corresponding bit of the controller interrupt register (the MMIS bit of the OTG_FS_GINTSTS register) will be affected. When the controller switches from one mode to another, the register banks in the new mode need to be re-initialized in the same way as during power-on reset.

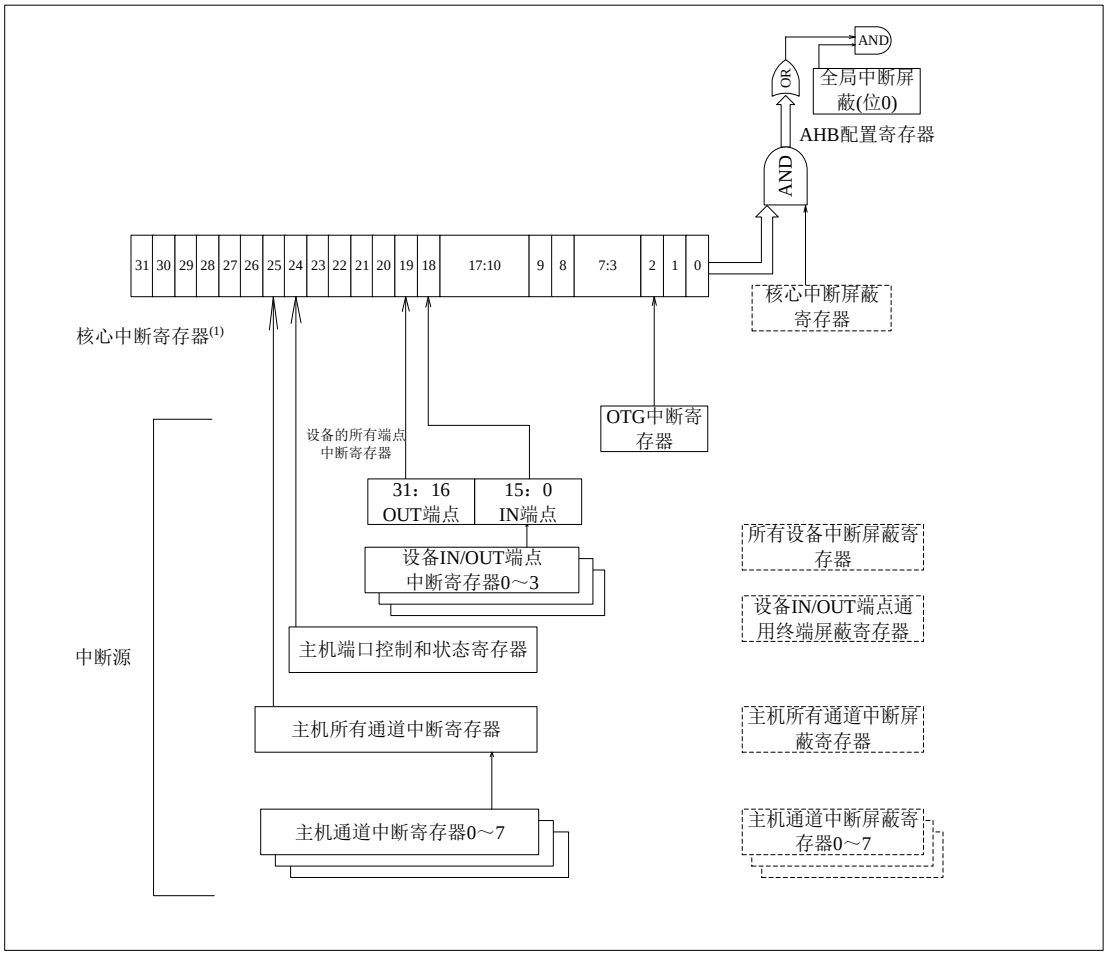


Figure 32-9 Interrupt Architecture

(1) Please refer to the relevant bits of the controller interrupt register

32.3 USBFS registers

The register width is 32 bits.

The application program reads and writes the control and status register (CSRx) through the AHB slave interface, thus realizing the control of the OTG_FS controller. These registers are all 32-bit accessed and 32-bit address aligned. Includes the following register sets:

- Controller Global Register Group
- Host Mode Register Group
- Host Global Register Group
- Host Port CSR Register Group
- Host channel related register group
- Device Mode Register Group
- Device Global Register Group
- Device endpoint dependent register set
- Power supply and clock control register bank
- Data FIFO (DFIFO) access register bank

Only the controller global registers, power supply and clock control registers, data FIFO access registers, and host port control and status registers are valid in both host mode and device mode. Regardless of whether the OTG_FS controller operates in host mode or device mode, applications cannot access register banks in the other mode. If the application has an illegal access, a pattern mismatch interrupt will be generated and the corresponding bit of the controller interrupt register (the MMIS bit of the OTG_FS_GINTSTS register) will be affected. When the controller switches from one mode to another, the register banks in the new mode need to be re-initialized in the same way as during power-on reset.

These peripheral registers must be operated in a word (32-bit) manner.

32.3.1 CSR memory image

The host mode register and the device mode register occupy different addresses. All registers are driven by the AHB clock.

Figure 32-10 CSR memory image

x is 3 in device mode and 7 in host mode

Global CSR address image

These registers are valid in both host mode and device mode.

Table 32-1 controller global control and status registers (CSRs)

Register code	Offset address	Register name
OTG_FS_OTGCTL	0x000	OTG_FS control and status registers
OTG_FS_GOTGINT	0x004	OTG_FS Interrupt Register
OTG_FS_GAHBCFG	0x008	OTG_FS AHB configuration register
OTG_FS_GUSBCFG	0x00C	OTG_FS USB configuration register
OTG_FS_GRSTCTL	0x010	OTG_FS Reset Register
OTG_FS_GINTSTS	0x014	OTG_FS controller interrupt register

OTG_FS_GINTMSK	0x018	OTG_FS Interrupt Mask Register
OTG_FS_GRXSTSR	0x01C	OTG_FS Receive Status Debug Read/OTG Status Read and Eject Register
OTG_FS_GRXSTSP	0x020	
OTG_FS_GRXFSIZ	0x024	OTG_FS receive FIFO length register
OTG_FS_GNPTXFSIZ	0x028	OTG_FS aperiodic transmission FIFO length register
OTG_FS_GNPTXSTS	0x02C	OTG_FS aperiodic send FIFO/queue status register
OTG_FS_GCCFG	0x038	OTG_FS Universal Control Configuration Register
OTG_FS_CID	0x03C	OTG_FS Controller ID register
OTG_FS_HPTXFSIZ	0x100	OTG_FS host mode periodically sends FIFO length register
OTG_FS_DIEPTXFx	0x104 0x124 … 0x13C	OTG_FS Device Mode IN Endpoint TX FIFO Length Register (x = 1... 3, indicating FIFO number)

Host mode CSR address image

These registers need to be configured each time you switch to host mode.

Table 32-2 Control and Status Registers (CSRs) in Master Mode

Register code	**Offset address**	**Register name**
OTG_FS_HCFG	0x400	OTG_FS host mode configuration register
OTG_FS_HFIR	0x404	OTG_FS Host Mode Frame Interval Register
OTG_FS_HFNUM	0x408	OTG_FS Host Mode Frame Number/Remaining Frame Time Register
OTG_FS_HPTXSTS	0x410	OTG_FS host mode periodic TX FIFO/queue status register
OTG_FS_HAINT	0x414	OTG_FS Host Mode All Channel Interrupt Register
OTG_FS_HAINTMSK	0x418	OTG_FS Host Mode All Channel Interrupt Mask Register
OTG_FS_HPRT	0x440	OTG_FS Host Mode Port Control and Status Register
OTG_FS_HCCHARx	0x500 0x520 … 0x6E0	OTG_FS Host Mode Channel x Characteristic Register (x = 0... 7, x indicates channel number)
OTG_FS_HCINTx	0x508	OTG_FS host mode channel x interrupt register (x = 0... 7, x indicates channel number)
OTG_FS_HCINTMSKx	0x50C	OTG_FS Host Mode Channel x Interrupt Mask Register (x = 0... 7, x indicates channel number)
OTG_FS_HCTSIZx	0x510	OTG_FS Host Mode Channel x Transfer Length Register (x = 0... 7, x indicates channel number)

Device Mode CSR Address Map

These registers must be configured each time you switch to device mode.

Table 32-3 device mode control and status register

Register code	**Offset address**	**Register name**
OTG_FS_DCFG	0x800	OTG_FS device mode configuration register
OTG_FS_DCTL	0x804	OTG_FS device mode control register
OTG_FS_DSTS	0x808	OTG_FS device mode status register
OTG_FS_DIEPMSK	0x810	OTG_FS Device Mode IN Endpoint Common Interrupt Mask Register
OTG_FS_DOEPMSK	0x814	OTG_FS Device Mode OUT Endpoint Common Interrupt Mask Register
OTG_FS_DAINT	0x818	OTG_FS Device Mode All endpoint interrupt registers
OTG_FS_DAINTMSK	0x81C	OTG_FS Device Mode All Endpoint Interrupt Mask Register
OTG_FS_DVBUSDIS	0x828	OTG_FS device mode VBUS power down time register
OTG_FS_DVBUSPulse	0x82C	OTG_FS device mode VBUS pulse time register
OTG_FS_DIEPEMPMSK	0x834	OTG_FS Device Mode IN Endpoint FIFO Null Interrupt Mask Register
OTG_FS_DIEPCTL0	0x900	OTG_FS Device Mode IN Control Endpoint 0 Control Register
OTG_FS_DIEPCTLx	0x920 0x940 … 0xAE0	OTG_FS device mode IN endpoint x control register (x = 1... 3, x indicates endpoint number)
OTG_FS_DIEPINTx	0x908	OTG_FS device mode IN endpoint x interrupt register (x = 1... 3, x indicates endpoint number)
OTG_FS_DIEPTSIZ0	0x910	OTG_FS Device Mode IN Endpoint 0 Transfer Length Register
OTG_FS_DTXFSTSx	0x918	OTG_FS Device Mode IN Endpoint TX FIFO status register (x = 1... 3, x indicates endpoint number)

OTG_FS_DIEPTSIZx	0x930 0x950 … 0xAF0	OTG_FS Device Mode IN endpoint x Transfer Length Register (x = 1... 3, x indicates endpoint number)
OTG_FS_DOEPCTL0	0xB00	OTG_FS Device Mode OUT Control Endpoint 0 Control Register
OTG_FS_DOEPCTLx	0xB20 0xB40 … 0xCC0 0xCE0 0xCFD	OTG_FS device mode OUT endpoint x control register (x = 1... 3, x indicates endpoint number)
OTG_FS_DOEPINTx	0xB08	OTG_FS device mode OUT endpoint x interrupt register (x = 1... 3, x indicates endpoint number)
OTG_FS_DOEPTSIZx	0xB10	OTG_FS Device Mode OUT endpoint x Transfer Length Register (x = 1... 3, x indicates endpoint number)

Data FIFO (DFIFO) access register address mapping

This set of register columns is valid in both host mode and device mode and is used to read and write the FIFO of a special endpoint or channel in a specified direction. If a channel IN host mode is of type IN, the corresponding FIFO can only perform read operations. Similarly, if a channel in host mode is of OUT type, the corresponding FIFO can only perform write operations.

Table 32-4 Data FIFO (DFIFO) Access Register

Data FIFO (DFIFO) access register segment	Address range	Access method
IN endpoint 0 IN device mode/OUT channel 0 IN host mode: DFIFO write only	Write operation	0x1000~0x1FFC
OUT endpoint 0 IN device mode/IN channel 0 IN host mode: DFIFO read only	Read operation	
IN endpoint 1 IN device mode/OUT channel 1 IN host mode: DFIFO write only	Write operation	0x2000~0x2FFC
OUT endpoint 1 IN device mode/IN channel 1 IN host mode: DFIFO read only	Read operation	
……	……	……
IN endpoint x IN device mode/OUT channel x IN host mode: DFIFO write only	Write operation	0xX000~0xXFFC
OUT endpoint x IN device mode/IN channel x IN host mode: DFIFO read only	Read operation	

The x in the table is 3 in device mode and 7 in host mode

Power and Clock Control CSR Register Image

There is only one register used to control power supply and clock control, and this register is valid in both device mode and host mode.

Table 32-5 Supply and Gate Control and Status Registers

Register name	abbreviation	Offset address
Powered and gated clock control register	PCGCR	0xE00~0xE04
Reserved	0xE05~0xFFF	

32.4 OTG_FS Global Register

These registers are valid in both device mode and host mode, and do not require re-initialization during mode switching. Unless otherwise specified, the value of each of the registers is represented by a binary number.

32.4.1 OTG_FS control and status register (OTG_FS_GOTGCTL)

Address: 0x0000

Reset value: 0x0001 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												BSVLD	ASVLD	DBCT	CIDSTS
												R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				DHNPEN	HSHNPEN	HNPRQ	HNGSCS	Res.						SRQ	SRQSCS
				RW	RW	RW	R							RW	R

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19	BSVLD	R	0	BSVLD: B-session valid Indicates the status of the device mode transceiver 0: Invalid class B session; 1: Class B session is valid. In OTG mode, the user can use this bit to determine whether the device is connected to the host Note: Only valid in device mode.
18	ASVLD	R	0	ASVLD: A-session valid Indicates the status of the host mode transceiver 0: Invalid class A session; 1: Class A session is valid. Note: Only valid in host mode.
17	DBCT	R	0	DBCT: Long/short debounce time Indicates reflection time of the detected connection 0: long reflection time for physical connection (100ms + 2.5 us); 1: Short reflection time for software connection (2.5 us). Note: Only valid in host mode
16	CIDSTS	R	1	CIDSTS: Connector ID status Indicates ID wire status on connector 0: OTG_FS is in class A device mode; 1: OTG_FS is in Class B device mode. Note: Works in both device mode and host mode
15: 12	Reserved	-	-	-
11	DHNPEN	RW	0	DHNPEN: Device HNP enabled The application sets this bit when it successfully receives a SetFeature.SetHNPEnable command from the USB host. 0: Disabled device mode HNP 1: Device mode HNP is enabled. Note: Only valid in device mode.
10	HSHNPEN	RW	0	HSHNPEN: Host set HNP enable The application sets this bit when it successfully enables the HNP of the connected device (via the SetFeature.SetHNPEnable command). 0: Disable host mode HNP; 1: Enable host mode HNP. Note: Only valid in host mode
9	HNPRQ	RW	0	HNPRQ: HNP request The application sets this bit when it needs to send an HNP request to the USB host. When the host negotiation success state transition bit of the OTG interrupt register (the HNSSCHG bit of the OTG_FS_GOTGINT register) is set, the application may clear this bit by writing '0'. The controller clears this bit when clearing the HNSSCHG bit. 0: No NHP request; 1: HNP request. Note: Only valid in device mode.
8	HNGSCS	R	0	HNGSCS: Host negotiation success The controller sets this bit when the host negotiation is successful. When the application sets the HNP request bit (HNPRQ), the controller clears this bit. 0: Host negotiation failed; 1: The host negotiation is successful. Note: Only valid in device mode
7: 2	Reserved	-	-	-
1	SRQ	RW	0	SRQ: Session request The application initiates a session request on the USB bus by setting this bit. When the host negotiation success state change bit of the OTG interrupt register (the HNSSCHG bit of the OTG_FS_GOTGINT register) is set, the application may clear this bit by writing '0'. The controller clears this bit when clearing the HNSSCHG bit. If a user initiates a session request using the USB1.1 full speed serial transceiver interface, the application must wait for the VBUS line to drop to 0.2 V after the Class B session valid bit (BSVLD bit of the

				OTG_FS_GOTGCTL register) is cleared. Different PHYs use different waiting times, which are controlled by the supplier of the PHY. 0: No session request; 1: Session request. Note: Only valid in device mode.
0	SRQSCS	R	0	SRQSCS: Session request success The controller sets this bit after the session request is successful. 0: session request failed; 1: Session request successful. Note: Only valid in device mode.

32.4.2 OTG_FS interrupt register (OTG_FS_GOTGINT)

Address: 0x0004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												DBCDNE	ADTOCHG	HNGDET	Res
												RC_W1	RC_W1	RC_W1	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.						HNSSCHG	SRSSCHG	Res.					SEDET	Res.	
						RC_W1	RC_W1						RC_W1		

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19	DBCDNE	RC_W1	0	DBCDNE: Debounce done The controller sets this bit after the device connection line reflection ends. The application can initiate a USB reset after detecting this interrupt Signal. This bit is only valid in the HNP or SRP valid bit of the controller USB configuration register (OTG_FS_GUSBCFG register HNPCAP bit or SRPCAP bit) is set. Note: Only valid in host mode.
18	ADTOCHG	RC_W1	0	ADTOCHG: A-device timeout change The controller sets this bit while a Class A device waits for a Class B device to insert a timeout. Note: Works in both host mode and device mode
17	HNGDET	RC_W1	0	HNGDET: Host negotiation detected The controller sets this bit when it detects a host negotiation request on the USB bus. Note: Works in both host mode and device mode.
16: 10	Reserved	-	-	-
9	HNSSCHG	RC_W1	0	HNSSCHG: Host negotiation success status change The controller sets this bit when the USB host negotiation request succeeds or fails. The application needs to read OTG control and status register The host of the device negotiates a success bit (the HNGSCS bit of the OTG_FS_GOTGCTL register) to obtain success or failure information. Note: Works in both host mode and device mode.
8	SRSSCHG	RC_W1	0	SRSSCHG: Session request success status change The controller sets this bit when the session request succeeds or fails. The application needs to pass the session of reading OTG control and status registers The success bit (the SRQSCS bit of the OTG_FS_GOTGCTL register) is requested to obtain success or failure information. Note: Works in both host mode and device mode.
7: 3	Reserved	-	-	-
2	SEDET	RC_W1	0	SEDET: Session end detected When the controller detects VBUS < 0.8 V, it sets this bit to indicate that the level of the VBUS line of the class B device is invalid.
1: 0	Reserved	-	-	-

32.4.3 OTG_FS AHB configuration register (OTG_FS_GAHBCFG)

Address: 0x0008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.							PTXFELVL	TXFELVL	Res.						GINT
							RW	RW							RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	PTXFELVL	RW	0	PTXFELVL: Periodic TxFIFO empty level Indicates under which circumstances it is necessary to trigger the periodic sending of the controller interrupt register FIFO null interrupt (OTG_FS_GINTSTS send PTXFE bit of the memory): 0: PTXFE (in OTG_FS_GINTSTS register) interrupt indicates that the periodic transmission FIFO has been half empty; 1: The PTXFE (in the OTG_FS_GINTSTS register) interrupt indicates that the periodic transmission FIFO has been completely empty. Note: Only valid in host mode.
7	TXFELVL	RW	0	TXFELVL: TxFIFO empty level IN device mode, this bit indicates the circumstances under which the IN endpoint needs to be triggered to send a FIFO null interrupt (OTG_FS_DIEPINTx send TXFE bit of the memory): 0: TXFE (IN the OTG_FS_DIEPINTx register) interrupt indicates that the IN endpoint TX FIFO has been half empty; 1: The TXFE (IN the OTG_FS_DIEPINTx register) interrupt indicates that the IN endpoint TX FIFO has been fully empty. In host mode, this bit indicates under which circumstances it is necessary to trigger an aperiodic send FIFO null interrupt (OTG_FS_GINTSTS NPTXFE bit of the register): 0: NPTXFE (in the OTG_FS_GINTSTS register) interrupt indicates that the aperiodic transmission FIFO has been half empty; 1: The NPTXFE (in the OTG_FS_GINTSTS register) interrupt indicates that the aperiodic transmission FIFO has been completely empty.
6: 1	Reserved	-	-	-
0	GINT	RW	0	GINT: Global interrupt mask The application can choose to mask or not mask the interrupt through this bit. But whether this bit is set or not, the controller still updates the interrupt Status Register. 0: Mask interrupt; 1: Do not mask interrupts. Note: Works in both device mode and host mode

32.4.4 OTG_FS configuration register (OTG_FS_GUSBCFG)

Address: 0x000C

Reset value: 0x10001440

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	FDMOD	FHMOD	Res.												
	RW	RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.		TRDT [3: 0]				HNPCAP	SRPCAP	Res.					TOCAL [2: 0]		
		RW				R/RW	R/RW						RW		

Bit	Name	R/W	Reset Value	Function

31	Reserved	-	-	-
30	FDMOD	RW	0	FDMOD Force device mode Regardless of the status of the OTG_FS_ID input pin, setting this bit to '1' will force the controller to enter device mode. 0: normal mode; 1: Force device mode. After this bit is set, the application needs to wait at least 25ms for the setting to take effect. Note: Works in both host mode and device mode.
29	FHMOD	RW	0	FHMOD Force host mode Regardless of the status of the OTG_FS_ID input pin, setting this bit to '1' will force the controller to enter host mode. 0: Normal mode 1: Force host mode After this bit is set, the application needs to wait at least 25ms for the setting to take effect. Note: Works in both host mode and device mode.
28: 14	Reserved	-	-	-
13: 10	TRDT [3: 0]	RW	4'h5	TRDT: USB turnaround time Set the turnaround time in units of clock cycles of the PHY layer. Sets the response time from a MAC request to the packet FIFO controller (PFC) requesting data from the DFIFO (SPRAM). These bits must be set to: 4 'h5: when the MAC interface is 16-bit UTMIFS; 4 'h9: When the MAC interface is 8-bit UTMIFS. Note: Only valid in device mode.
9	HNPCAP	R/RW	0	HNPCAP: HNP-capable The application uses this bit to enable the HNP function of the OTG_FS controller. 0: Disable HNP function; 1: Enable HNP function. Note: Works in both device mode and host mode.
8	SRPCAP	R/RW	0	SRPCAP: The SRP-capable application uses this bit to enable the SRP function of the OTG_FS controller. If the controller is operating in Class B device mode without SRP function, the connected Class A device (host) cannot be required to enable the VBUS line and initiate a session. 0: Disable SRP function; 1: Enable SRP function. Note: Works in both device mode and host mode.
7: 3	Reserved	-	-	-
2: 0	TOCAL [2: 0]	RW	3'h0	TOCAL: FS timeout calibration (FS timeout calibration) Considering the additional delay brought by PHY, the application program can adjust the full-speed packet timeout determination delay of the controller. These bits give the length of the adjustment delay in terms of the number of PHY clocks. Different PHYs bring different additional delays when controlling the bus state, and these bits can adapt to different bus delays. The timeout judgment of full-speed packets specified by the USB standard is 16 to 18 BIT times. The application needs to configure this bit based on the speed of the enumeration. The increased BIT time per PHY clock is 0.25 BIT time.

32.4.5 OTG_FS Reset Register (OTG_FS_GRSTCTL)

Address: 0x0010

Reset value: 0x8000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AHBIDL	Res	CSftRstDone	Res												
R		RC_W1													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.					TXFNUM [4: 0]					TXFFLSH	RXFFLSH	Res	FCRST	HSRST	CSRST
					RW					RS	RS		RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	AHBIDL	R	1	AHBIDL: AHB master idle Indicates whether the AHB master module is idle. Note: Works in both device mode and host mode.
30	Reserved	-	-	-

29	CSftRstDone	RC_W1	0	Core Software Reset Ended Core sets this bit when all necessary logic is reset in Core. This bit and CSRST (bit0) in USBFS_GRSTCTL are cleared by the application 0: No reset 1: Core software reset ended Note: Works in both device mode and host mode.
28: 11	Reserved	-	-	-
10: 6	TXFNUM [4: 0]	RW	5'h0	TXFNUM: Transmission FIFO number (TxFIFO number) This bit indicates which FIFO needs to be refreshed by sending a FIFO refresh bit. Only when the controller clears the send FIFO refresh bit, The application can modify this bit. ● 5 'b00000: In host mode, it means that FIFO is transmitted aperiodically In device mode, it means to send FIFO 0 ● 5'b 00001: In host mode, it means that FIFO is sent periodically In device mode indicates that FIFO 1 is sent ● 5 'b 00010: In device mode indicates that FIFO 2 is sent …… ● 5'b 00101: In device mode indicates that FIFO 5 is sent ● 5'b 10000: Refresh all send FIFOs in host mode or device mode Note: Works in both host mode and device mode.
5	TXFFLSH	RS	0	TXFFLSH: Transmit FIFO flush (TxFIFO flush) This bit is used to refresh individual or all transmit FIFOs. This bit cannot be set when the controller is transmitting. The application can only set this bit if it detects that the controller is neither writing data to nor reading data from the sending FIFO. It can be detected by: Read: The NAK active interrupt ensures that the controller is not reading data from the FIFO. Write: The AHBIDL bit of the OTG_FS_GRSTCTL register ensures that the controller is not writing the FIFO. Note: Works in both device mode and host mode.
4	RXFFLSH	RS	0	RXFFLSH: RxFIFO flush The application can refresh the entire receive FIFO with this bit, but must ensure that the controller is not transmitting data. The application can only set this bit if it detects that the controller has neither written nor read receive FIFO. The application can only perform other operations after waiting for this bit to be cleared again by the controller. This bit takes 8 clock cycles (whichever is slower of the PHY or AHB clock) to clear. Note: Works in both host mode and device mode.
3	Reserved	-	-	-
2	FCRST	RS	0	FCRST: Host frame counter reset The application resets the frame counter inside the controller by writing this bit. After the frame counter is reset, the frame number of the first SOF sent by the controller is 0. Note: Only valid in host mode.
1	HSRST	RS	0	HSRST: The HCLK soft reset application can use this bit to refresh the control logic of the AHB clock domain. Only modules driven by the AHB clock domain are reset. FIFO is not affected by this bit. According to the protocol, all state machines driven by the AHB clock domain will be reset to the idle state after completing the current transmission of the AHB. Clear the CSR control bit in the AHB clock domain driven state machine. To clear this interrupt, the state mask bits used to control the interrupt state and generated by the state machine driven by the AHB clock domain will be cleared. Since the interrupt status bit is not cleared, the application can still obtain the status of any controller events generated after setting this bit. The controller automatically clears this bit after all necessary logic modules are reset. This operation will be maintained for several clock cycles, which are determined by the current state of the controller. Note: Works in both device mode and host mode.
0	CSRST	RW	0	CSRST: Controller Software Reset (Core soft reset) Reset HCLK and PCLK Driver Field: Clear all interrupt and CSR registers except the following bits: RSTPDMODL bit of OTG_FS_PCGCCTL

				register GAYEHCLK bit of OTG_FS_PCGCCTL register PWRCLMP bit of OTG_FS_PCGCCTL register STPPCLK bit of OTG_FS_HCFG register FSLSPCS bit of OTG_FS_HCFG register DSPD bit of OTG_FS_HCFG register All modules state machines (except AHB slave unit) will be reset to idle state, and all send FIFOs and receive FIFOs are flushed. All data transmission on the AHB master module ends immediately after the final data phase of the AHB transmission is completed. All transfers over USB end immediately. The application can set this bit any time it needs to reset the controller. The controller will automatically clear this bit after all necessary logic modules are reset. This operation will be maintained for several clock cycles, depending on the current state of the controller. Once this bit is cleared, the software must wait for at least 3 PHY clock cycles (delay of synchronous operation) before activating the PHY domain. The application must ensure that bit 31 of this register is' 1 '(i.e., the AHB master is idle) before starting other operations. Generally, it is only necessary to enter a software reset operation during the software development phase and when the PHY select bit is dynamically modified in the USB configuration registers listed above. If the user changes the PHY configuration, the corresponding clock is selected and applied to the PHY domain. Once a new clock is selected, the PHY domain needs to be reset for further operation. Note: Works in both device mode and host mode

32.4.6 OTG_FS Controller Interrupt Register (OTG_FS_GINTSTS)

Address: 0x0014

Reset value: 0x0400 0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WKUPINT	SRQINT	DISCINT	CIDSCHG	Res.	PTXFE	HCINT	HPRTINT	Res.		IPXFR/INCOMPISOUT	IISOIXFR	OEPINT	IEPINT	Res.	
RC_W1					R	R	R			RC_W1	RC_W1	R	R		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EOPF	ISOODRP	ENUMDNE	USBRST	USBSUSP	ESUSP	Res.		GONAKEFF	GINAKEFF	NPTXFE	RXFLVL	SOF	OTGINT	MMIS	CMOD
RC_W1								R	R	R	R	RC_W1	R	RC_W1	R

Bit	Name	R/W	Reset Value	Function
31	WKUPINT	RC_W1	0	WKUPINT: Resume/remote wakeup detected interrupt (Resume/remote wakeup detected interrupt) This interrupt is generated when a wake-up signal on the USB bus is detected in device mode; This interrupt is generated when a remote wakeup signal on the USB bus is detected in host mode. Note: Works in both device mode and host mode.
30	SRQINT	RC_W1	0	SRQINT: Session request/new session detected interrupt In host mode, this interrupt occurs when a session request from the device is detected; In device mode, this interrupt occurs when VBUS is within the active level range of Class B devices. Note: Works in both device mode and host mode.
29	DISCINT	RC_W1	0	DISCINT: Disconnect detected interrupt (Disconnect detected interrupt) This interrupt is generated when the device is detected to be disconnected. Note: Only valid in host mode.
28	CIDSCHG	RC_W1	0	CIDSCHG: Connector ID status change The controller sets this bit when it detects a change in the ID line status on the connection. Note: Works in both host mode and device mode
27	Reserved	-	-	-
26	PTXFE	R	1	PTXFE: Periodic TxFIFO empty (Periodic TxFIFO empty) This interrupt occurs when half-empty or full-empty FIFO is periodically sent, and at least one request can be written in the periodic request queue. The state

				of half empty or full empty is determined by the periodic transmission of the FIFO empty level bit of the controller AHB configuration register (PTXFELVL bit of the OTG_FS_GAHBCFG register). Note: Only valid in host mode.
25	HCINT	R	0	HCINT: Host channels interrupt The controller sets this bit to indicate that there is an unhandled host channel event (in host mode). The application needs to read all channel interrupt registers of the host (OTG_FS_HAINT registers) to obtain the channel number that generated the interrupt, and then read the corresponding host channel x interrupt register (OTG_FS_HCINTx register) to obtain the specific information of the interrupt. The application needs to clear this bit by clearing the corresponding bit of the OTG_FS_HCINTx register. Note: Only valid in host mode.
24	HPRTINT	R	0	HPRTINT: Host port interrupt controller setting This bit indicates a change in the state of a port of the OTG_FS controller. The application needs to read the host port control and status register (OTG_FS_HPRT) to get the specific information that caused this interrupt. The application must clear this bit by clearing the corresponding bits in the host port control and status registers. Note: Only valid in host mode.
23: 22	Reserved	-	-	-
21	IPXFR	RC_W1	0	IPXFR: Incomplete periodic transfer (Incomplete periodic transfer) In host mode, the controller generates this interrupt when there are still incomplete periodic transfers belonging to the current frame at the end of the frame. Note: Only valid in host mode.
	INCOMPISOOUT			INCOMPISOOUT: Incomplete isochronous OUT transfer (Incomplete isochronous OUT transfer) In device mode, the controller generates this interrupt when there is still at least one outstanding synchronous OUT transfer belonging to the current frame at the end of the frame. This interrupt is generated along with a periodic end-of-frame interrupt (EOPF) in this register. Note: Only valid in device mode.
20	IISOIXFR	RC_W1	0	IISOIXFR: Incomplete isochronous IN transfer (Incomplete isochronous IN transfer) IN device mode, the controller generates this interrupt when there is still at least one Incomplete isochronous IN transfer belonging to the current frame at the end of the frame. This interrupt is generated along with a periodic end-of-frame interrupt (EOPF) in this register. Note: Only valid in device mode.
19	OEPINT	R	0	OEPINT: OUT endpoint interrupt (OUT endpoint interrupt) In device mode, the controller sets this bit to indicate that there is an OUT endpoint event not handled by the controller. The application needs to read all endpoint interrupt registers of the device (OTG_FS_DAINT) to obtain the endpoint number that generated the interrupt event, and then read the corresponding device OUT endpoint x interrupt register (OTG_FS_DOEPINTx) to obtain the specific information that generated the interrupt. The application needs to clear this bit by clearing the corresponding bit of the OTG_FS_DOEPINTx register. Note: Only valid in device mode.
18	IEPINT	R	0	IEPINT: IN endpoint interrupt IN device mode, the controller sets this bit to indicate that there is an IN endpoint event that is not handled by the controller. The application needs to read all endpoint interrupt registers of the device (OTG_FS_DAINT) to obtain the endpoint number that generated the interrupt event, and then read the corresponding device IN endpoint x interrupt register (OTG_FS_DIEPINTx) to obtain the specific information that generated the interrupt. The application needs to clear this bit by clearing the corresponding bit of the OTG_FS_DIEPINTx register. Note: Only valid in device mode.

17: 16	Reserved	-	-	-
15	EOPF	RC_W1	0	EOPF: End of periodic frame interrupt This interrupt indicates that in the current frame, the periodic frame interval time (PFIVL bit of the OTG_FS_DCFG register) set in the device configuration register has arrived. Note: Only valid in device mode.
14	ISOODRP	RC_W1	0	ISOODRP: Isochronous OUT packet dropped interrupt (Isochronous OUT packet dropped interrupt) When the receiving FIFO does not have enough space to store a data packet of maximum length for a synchronous OUT endpoint, it will cause the controller to fail to write the synchronous OUT packet and generate this interrupt. Note: Only valid in device mode.
13	ENUMDNE	RC_W1	0	ENUMDNE: Enumerationdone The controller sets this bit to indicate that the speed enumeration information has been obtained. The application obtains the speed information for enumeration by reading the device status register (OTG_FS_DSTS). Note: Only valid in device mode.
12	USBRST	RC_W1	0	USBRST: The USB reset controller sets this bit when a USB reset signal is detected. Note: Only valid in device mode.
11	USBSUSP	RC_W1	0	USBSUSP: USB suspend The controller sets this bit when it detects a suspend signal on the USB cable. The controller will enter a suspended state when the data line is inactive for more than 3ms. Note: Only valid in device mode.
10	ESUSP	RC_W1	0	ESUSP: Early suspend The controller sets this bit when it detects that the USB bus is idle for 3ms. Note: Only valid in device mode.
9: 8	Reserved	-	-	-
7	GONAKEFF	R	0	GONAKEFF: Global OUT NAK effect indicates the application that the global OUT NAK bit of the device control register (SGONAK bit of the OTG_FS_DCTL register) is active. The application clears this bit by writing the clear global OUT NAK bit of the device control register (the CGONAK bit of the OTG_FS_DCTL register). Note: Only valid in device mode.
6	GINAKEFF	R	0	GINAKEFF: Global IN non-periodic NAK effect (Global IN non-periodic NAK effect) indicates that the setting of the global aperiodic IN NAK bit (SGINAK bit of the OTG_FS_DCTL register) of the device control register has taken effect. That is, the controller has responded to the application's operation of setting the global IN NAK bit. The application clears this bit by clearing the clear global aperiodic IN NAK bit of the device control register (the CGINAK bit of the OTG_FS_DCTL register). This interrupt does not mean that a NAK handshake signal is sent to the USB bus, and the STALL bit has a priority over the NAK bit. Note: Only valid in device mode.
5	NPTXFE	R	1	NPTXFE: Non-periodic TxFIFO empty (Non-periodic TxFIFO empty) When the aperiodic FIFO is fully empty or half empty, and at least one request can be written in the aperiodic request queue, the controller will generate this interrupt. The FIFO half-empty or full-empty state depends on the aperiodic transmission of the FIFO null level bit (TXFELVL bit of the OTG_FS_GAHBCFG register) by the controller AHB configuration register. Note: Only valid in host mode.
4	RXFLVL	R	0	RXFLVL: RxFIFO non-empty indicates that at least one packet in the receiving FIFO is waiting to be processed. Note: Works in both device mode and host mode.
3	SOF	RC_W1	0	SOF: Start of frame In host mode, the controller sets this bit to indicate that a SOF (full speed device) or hold active (low speed device) signal has been sent to the USB bus. The application must write '1' to this bit to clear this interrupt. In device mode, the controller sets this bit to indicate that a SOF is detected on the USB bus. The ap-

				plication needs to read the device status register to obtain the frame number. This interrupt occurs only when the controller is at full speed. Note: Works in both device mode and host mode.
2	OTGINT	R	0	OTGINT: The OTG interrupt controller sets this bit to indicate that an OTG protocol event has occurred. The application must read the OTG interrupt register (OTG_FS_GOTGINT) to obtain the specific information that generated this interrupt. The application must clear this bit by clearing the corresponding bit of the OTG_FS_GOTGINT register. Note: Works in both device mode and host mode
1	MMIS	RC_W1	0	MMIS: Mode mismatch interrupt The controller sets this bit when an application attempts to access a register in host mode while the controller is running in device mode. When the controller is running in host mode, it attempts to access registers in device mode. When operating the register, the correct response will be received on the AHB side, but such operations will be ignored internally in the controller and will not have any impact on the operation of the controller. Note: Works in both host mode and device mode
0	CMOD	R	0	CMOD: Current mode of operation indicates the current mode. 0: device mode; 1: Host mode. Note: Works in both device mode and host mode

32.4.7 OTG_FS Interrupt Mask Register (OTG_FS_GINTMSK)

Address: 0x0018

Reset value: 0x0000 0000

This register is used in conjunction with the controller interrupt register to generate interrupts. When an interrupt bit is masked, the corresponding interrupt will not be generated, but the corresponding bit of the controller interrupt register (OTG_FS_GINTSTS) will still be set.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WUIM	SRQIM	DISCINT	CIDSCHGM	Res.	PTXFEM	HCIM	PRTIM	Res.		IPXFRM/IISOOXFRM	IISOIXFRM	OEPINT	IEPINT	Res.	
RW	RW	RW	RW		RW	RW	RW			RW	RW	RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOPFM	ISOODRPM	ENUMDNEM	USBRST	USBSUSPM	ESUSPM	Res.		GONAKEFFM	GINAKEFFM	NPTXFEM	RXFLVLM	SOFM	OTGINT	MMISM	Res.
RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31	WUIM	RW	0	WUIM: Resume/remote wakeup detected interrupt mask (Resume/remote wakeup detected interrupt mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Works in both device mode and host mode
30	SRQIM	RW	0	SRQIM: Session request/new session detected interrupt mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Works in both device mode and host mode.
29	DISCINT	RW	0	DISCINT: Disconnect detected interrupt mask (Disconnect detected interrupt mask) 0: Mask the interrupt; 1: The interrupt is not masked. Note: Only valid in host mode.
28	CIDSCHGM	RW	0	CIDSCHGM: Connector ID status change mask on connection 0: Mask the interrupt; 1: The interrupt is not masked. Note: Works in both host mode and device mode
27	Reserved	-	-	-
26	PTXFEM	RW	0	PTXFEM: Periodic TxFIFO empty mask (Periodic TxFIFO empty mask) 0: mask this interrupt; 1: The interrupt is not masked. Note: Only valid in host mode

25	HCIM	RW	0	HCIM: Host channels interrupt mask 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in host mode.
24	PRTIM	RW	0	PRTIM: Host port interrupt mask 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in host mode
23: 22	Reserved	-	-	-
21	IPXFRM	RW	0	IPXFRM: Incomplete periodic transfer mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Only valid in host mode.
	IISOOXFRM			IISOOXFRM: Incomplete isochronous OUT transfer mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
20	IISOIXFRM	RW	0	IISOIXFRM: Incomplete isochronous IN transfer mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
19	OEPINT	RW	0	OEPINT: OUT endpoints interrupt mask (OUT endpoints interrupt mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode.
18	IEPINT	RW	0	IEPINT: IN endpoints interrupt mask (IN endpoints interrupt mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
17: 16	Reserved	-	-	-
15	EOPFM	RW	0	EOPFM: End of periodic frame interrupt mask 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
14	ISOODRPM	RW	0	ISOODRPM: Isochronous OUT packet dropped interrupt mask (Isochronous OUT packet dropped interrupt mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
13	ENUMDNEM	RW	0	ENUMDNEM: Enumeration done mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
12	USBRST	RW	0	USBRST: USB reset mask (USB reset mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
11	USBSUSPM	RW	0	USBSUSPM: USB suspend mask (USB suspend mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode.
10	ESUSPM	RW	0	ESUSPM: Early suspend mask (Early suspend mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
9: 8	Reserved	-	-	-
7	GONAKEFFM	RW	0	GONAKEFFM: Global OUT NAK status interrupt mask (Global OUT NAK effect mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode
6	GINAKEFFM	RW	0	GINAKEFFM: Global non-periodic IN NAK effect mask 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in device mode.
5	NPTXFEM	RW	0	NPTXFEM: Non-periodic TxFIFO empty mask 0: mask the interrupt; 1: The interrupt is not masked. Note: Only valid in host mode.
4	RXFLVLM	RW	0	RXFLVLM: Receive FIFO non-empty mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Works in both device mode and host mode
3	SOFM	RW	0	SOFM: Start of frame mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Works in both device mode and host mode.
2	OTGINT	RW	0	OTGINT: OTG interrupt mask (OTG interrupt mask) 0: mask the interrupt; 1: The interrupt is not masked. Note: Works in both device mode and host mode
1	MMISM	RW	0	MMISM: Mode mismatch interrupt mask 0: Mask the interrupt; 1: The interrupt is not masked. Note: Works in both host mode and device mode.
0	Reserved	-	-	-

32.4.8 OTG_FS Receive Status Debug Read/OTG Status Read and POP Register (OTG_FS_GRXSTSR/OTG_FS_GRXSTSP)

Read address offset: 0x001C

Ejected address offset: 0x0020

Reset value: 0x0000 0000

A read operation on the receive state debug read register will return the top data in the receive FIFO. A read operation on the receive state debug read register and the POP register will eject the top data item in the receive FIFO. The interpretation of received state data is not the same in host mode and device mode. If the receive FIFO is empty, the controller will ignore the read/POP operation on the receive state and return 0x0000 0000. The application can POP out the receive state FIFO only when the received FIFO non-empty bit of the controller interrupt register (the RXFLVL bit of the OTG_FS_GINTSTS register) is' 1 '.

Host Mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.											PKTSTS				DPID
											R				R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID	BCNT [10: 0]											CHNUM [3: 0]			
R	R											R			

Bit	Name	R/W	Reset Value	Function
31: 21	Reserved	-	-	-
20: 17	PKTSTS [3: 0]	R	4’h0	PKTSTS: Packet status indicates the status of the received data packet 4'b0010: IN packet received; 4 'b0011: IN transmission complete (trigger interrupt); 4 'b0101: data flip bit error (trigger interrupt); 4 'b0111: channel abort (trigger interrupt); Other: Reserved.
16: 15	DPID [1: 0]	R	2’h0	DPID: Data PID (Data PID) indicates the Data PID of the received data packet 2'b00: DATAO; 2 'b10: DATA1; 2 'b01: retained; 2 'b11: Reserved.
14: 4	BCNT [10: 0]	R	11’h0	BCNT: Byte count Indicates the number of bytes of the received packet
3: 0	CHNUM [3: 0]	R	4’h0	CHNUM: Channel number indicates which channel the currently received data packet belongs to

Device Mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							FRMNUM [3: 0]				PKTSTS [3: 0]				DPID
							R				R				R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID	BCNT [10: 0]											EPNUM [3: 0]			
R	R											R			

Bit	Name	R/W	Reset Value	Function
31: 25	Reserved	-	-	-
24: 21	FRMNUM [3: 0]	R	4’h0	FRMNUM: Frame number (Frame number) These 4 bits are the last 4 bits of the Frame number to which the packet received from USB belongs, and are only valid for synchronous OUT transmission
20: 17	PKTSTS [3: 0]	R	4’h0	PKTSTS: Packet status indicates the status of the received data packet 4'b0001: global OUT NAK (trigger interrupt); 4'b0010: OUT packet received; 4 'b0011: OUT transmission complete (trigger interrupt); 4 'b0100: SETUP transmission complete (trigger interrupt); 4 'b0110: SETUP packet received; Other: Reserved.
16: 15	DPID [1: 0]	R	2’h0	DPID: Data PID indicates the PID of the OUT packet received 2'b00: DATAO; 2 'b10: DATA1; 2 'b01: retained;

				2 'b11: Reserved.
14: 4	BCNT [10: 0]	R	11'h0	BCNT: Byte count Indicates the number of bytes of the received packet
3: 0	EPNUM [3: 0]	R	4'h0	EPNUM: The endpoint number indicates the endpoint number to which the currently received data packet belongs

32.4.9 OTG_FS receive FIFO length register (OTG_FS_GRXFSIZ)

Address: 0x0024

Reset value: 0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					RXFD [11: 0]										
					RW										

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10: 0	RXFD [10: 0]	RW	12'h100	RXFD: Received FIFO depth (RxFIFO depth) The unit of this value is a 32-bit word Minimum is 16 and maximum is 256 The value of the power-on reset is the maximum depth value of the received FIFO.

32.4.10 OTG_FS Aperiodic TX FIFO Length Register (OTG_FS_GNPTXFSIZ)

Address: 0x0028

Reset value: 0x0100 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					NPTXFD / TX0FD										
					RW										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					NPTXFSA/TX0FSA										
					RW										

Master mode

Bit	Name	R/W	Reset Value	Function
31: 27	Reserved	-	-	-
26: 16	NPTXFD	RW	11'h100	NPTXFD: Non-periodic TxFIFO depth The unit of this value is a 32-bit word. The minimum value is 16 and the maximum value is 256
15: 11	Reserved	-	-	-
10: 0	NPTXFSA	RW	11'h100	NPTXFSA: Non-periodic transmit RAM start address of FIFO in RAM The values of these bits represent the start address of the aperiodic receive FIFO in RAM

Device Mode

Bit	Name	R/W	Reset Value	Function
31: 27	Reserved	-	-	-
26: 16	TX0FD	RW	11'h100	TX0FD: EP0 TxFIFO depth (Endpoint 0 TxFIFO depth) The unit of this value is 32-bit word. The minimum value is 16 and the maximum value is 256
15: 11	Reserved	-	-	-
10: 0	TX0FSA	RW	11'h100	TXOFSA: EP0 Non-periodic transmit RAM start address The values of these bits represent the start address of the EP0 transfer FIFO RAM

32.4.11 OTG_FS Aperiodic TX FIFO/Request Queue Status Register (OTG_FS_GNPTXSTS)

Address: 0x002C

Reset value: 0x0008 0100

Note: In device mode, this register is invalid.

This register is a read-only register and stores the remaining space information of the aperiodic transmission FIFO and the aperiodic transmission request queue

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	NPTXQTOP [6: 0]							NPTQXSAV [7: 0]							
	R							R							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NPTXFSAV [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 24	NPTXQTOP [6: 0]	R	7’h0	NPTXQTOP: The Top of the non-periodic transmit request queue (Top of the non-periodic transmit request queue) is being processed by the MAC module: Bit [30: 27]: channel/endpoint number; Bit [26:25]: 2 'b00: IN/OUT command; 2 'b01: 0 length transmission packet (device IN/host OUT); 2 'b11: channel abort command; Bit 24: End (last request for the channel/endpoint selected).
23: 16	NPTQXSAV [7: 0]	R	8’h8	NPTQXSAV: Non-periodic transmit request queue space available Indicates the remaining space of the aperiodic transmission request queue. IN host mode, this queue holds both transmission requests for IN and transmission requests for OUT, and IN device mode, only transmission requests for IN are saved. 8 'h0: aperiodic transmission request queue is full; 8 'h1: 1 request space remaining; 8 'h2: 2 request spaces remaining; 8'hn: n remaining request spaces (0 ≤ n ≤ 8); Other: Reserved.
15: 0	NPTXFSAV [15: 0]	R	16’h100	NPTXFSAV: Non-periodic TxFIFO space available indicates the remaining space of the aperiodic transmission FIFO, and this value is 32 bits. 16'h0: aperiodic transmission FIFO full; 16 'h1: space with 1 word remaining; 16 'h2: space for 2 words remaining; 16'hn: space of the remaining n words (0 ≤ n ≤ 256); Other: Reserved.

32.4.12 OTG_FS Universal Controller Configuration Register (OTG_FS_GCCFG)

Address: 0x0038

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.										VBUSIG	SOFOUTEN	VBUSBSEN	VBUSASEN	Res.	PWRON
										RW	RW	RW	RW		RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21	VBUSIG	RW	0	VBUS ignore When this control bit is set, USBFS does not monitor the VBUS pin voltage and considers that the VBUS voltage is always active in host and device

				modes, and then the VBUS pin can be released for other uses. 0: VBUS is not ignored 1: VBUS is ignored and the VBUS voltage is considered to be valid all the time
20	SOFOEN	RW	0	SOFOEN: SOF output enable 0: No SOF pulse is output; 1: Output SOF pulse to pin.
19	VBUSBCEN	RW	0	VBUSBSEN: Enable the VBUS compare "B" device to monitor the Class B active level 0: VBUS does not monitor Class B active level; 1: VBUS monitors Class B active level.
18	VBUSACEN	RW	0	VBUSASEN: Enable the VBUS compare "A" device for Class A active level monitoring 0: VBUS does not monitor Class A active level; 1: VBUS monitors Class A active level.
17	Reserved	-	-	-
16	PWRON	RW	0	PWRON: Power on is used to activate the transceiver during transmission and reception 0: enable power down; 1: Prohibit power-down (transceiver activation)
15: 0	Reserved	-	-	-

32.4.13 OTG_FS Controller ID Register (OTG_FS_CID)

Address: 0x003C

Reset value: 0x0000 1000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRODUCT_ID [31: 16]															
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRODUCT_ID [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 0	PRODUCT_ID	RW	32'h1000	PRODUCT_ID: Product ID field The application can write this ID bit.

32.4.14 OTG_FS host periodically sends FIFO length register (OTG_FS_HPTXFSIZ)

Address: 0x0100

Reset value: 0x0100 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTXFSIZ [15: 0]															
R/RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PTXSA [15: 0]															
R/RW															

Bit	Name	R/W	Reset Value	Function
31: 16	PTXFSIZ [15: 0]	R/RW	16'h100	PTXFSIZ: Host periodic TxFIFO depth (Host periodic TxFIFO depth) This value is a 32-bit word. The minimum value is 16 and the maximum value is 512
15: 0	PTXSA [15: 0]	R/RW	16'h200	PTXSA: Host periodic TxFIFO start address (Host periodic TxFIFO start address) The reset value of this register is the sum of the maximum received FIFO depth and the maximum aperiodic transmitted FIFO depth.

32.4.15 The OTG_FS device IN endpoint sends a FIFO length register (OTG_FS_DIEPTXFx) (where x is the number of the FIFO, x = 1... 3)

Address: 0x0104 + (FIFO_number-1) * 0x04

Reset value: 0x0100 0200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INEPTXFD [15: 0]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPTXSA [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	INEPTXFD [15: 0]	RW	16'h100	INEPTXFD: The depth at which the IN endpoint transmits the FIFO (IN endpoint TxFIFO depth) Words with this value of 32 bits The minimum value is 16 The maximum value is 512 The reset value is the depth at which the maximum possible IN endpoint sends the FIFO
15: 0	INEPTXSA [15: 0]	RW	16'h200	INEPTXSA: IN endpoint FIFOx transmit RAM startaddress This value is the starting address IN RAM from which the IN endpoint sends the FIFO.

32.5 Registers in host mode

Unless otherwise specified, the values in the registers are expressed in binary form.

Registers in host mode only take effect in host mode and cannot be accessed in device mode. The result of illegal access is undefined

Of. The registers in host mode are as follows:

32.5.1 OTG_FS host mode configuration register (OTG_FS_HCFG)

Address: 0x0400

Reset value: 0x0000 0200

This register configures the operation of the controller after power-on. Do not modify this register after initialization.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.													FSLSS	FSLSPCS	
													R	RW	

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	FSLSS	R	0	FSLSS: FS-and LS-only support The application uses this bit to configure how quickly the controller can enumerate devices. The application can use this bit to cause the controller to use full speed Pattern to enumerate a device that supports high-speed communication. Do not modify this value after initialization. 1: Only support full-speed/low-speed devices, even if the plugged-in device supports high-speed communication (read-only)
1: 0	FSLSPCS	RW	2'h0	FSLSPCS: FS/LS PHY clock select When the controller is in full speed host mode: 2 'b01: PHY clock running at 48 MHz; Other values: reserved. When the controller is in low-speed host mode: 2 'b00: reserved; 2 'b01: PHY clock running at 48 MHz; 2 'b10: The PHY clock runs at 6MHz, according to the USB1.1 full speed mode definition, when the UTMIFS PHY low power mode is selected When the PHY supports a 6MHz clock, the 6MHz clock is used in low speed mode. Once the user is in low speed mode The 6MHz clock is selected, and a software reset operation needs to be performed;

				2 'b11: Reserved.

32.5.2 OTG_FS Host Frame Interval Register (OTG_FS_HFIR)

Address: 0x0404

Reset value: 0x0000 EA60

This register sets the frame interval time for the speed selected by the OTG_FS controller during the enumeration.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRIVL [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	FRIVL [15: 0]	RW	16'hEA60	FRIVL: Frame interval The application uses this bit to configure the time interval between two consecutive SOFs (full speed) or hold active (low speed). This register represents the frame interval with the number of PHY clocks. The application can only set this register after it has set the port enable bit of the host port control and status register (the PENA bit of the OTG_FS_HPRT register). If this register is not set, the controller calculates the value according to the PHY clock defined by the full/low speed PHY clock select bit of the host configuration register (FSLSPCS bit of the OTG_FS_HCFG register). Do not modify this register after initialization. 1 ms × (full speed/low speed PHY clock frequency)

32.5.3 OTG_FS Host Frame Number/Frame Time Remaining Register (OTG_FS_HFNUM)

Address: 0x0408

Reset value: 0x0000 03FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FTREM [15: 0]															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRNUM [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 16	FTREM [15: 0]	R	16'h0	FTREM: Frame time remaining Indicates how much time remains in the current frame, expressed in the number of PHY clocks. Each PHY clock, this value decreases by 1. When this value decreases Is 0, the value defined in the frame interval register will automatically load this register and send a new SOF to the USB bus.
15: 0	FRNUM [15: 0]	R	16'h3FFF	FRNUM Frame number Every time a SOF signal is sent to the USB bus, this field is automatically incremented by 1. When it reaches 0x3FFF, it automatically returns to zero and starts accumulation again.

32.5.4 Primary OTG_FS host periodically sends FIFO/request queue registers (OTG_FS_HPTXSTS)

Address: 0x0410

Reset value: 0x0008 0100

This register is a read-only register, which holds the information of the remaining space of periodically sending FIFO and request queue.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTXQTOP [8: 0]									PTXQSAV [6: 0]						
R									R						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTXFSAVL [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 23	PTXQTOP [8: 0]	R	9’h0	PTXQTOP: Top of the periodic transfer request queue Indicates a periodic transmit request item being processed by the MAC module. This register is for module debugging only. Bit 31: odd/even frames 0: Even frame transmission 1: Odd frame transmission. Bit [30: 27]: channel/endpoint number; Bit [26:25]: type 2 'b00: IN/OUT; 2 'b01: zero length packet; 2 'b11: Abort channel command. Bit 24: End (last request of selected channel/end-point) Bit 23: Terminate (last entry of selected chan-nel/endpoint)
22: 16	PTXQSAV [6: 0]	R	7’h8	PTXQSAV: Periodic transmit request queue space available Indicates the remaining space of the periodic transmission request queue, which includes re-quests for IN and OUT. 7 'h0: Periodic transmission request queue is full; 7'h1: 1 requested position remaining; 7 'h2: 2 requested positions remaining; 7'hn: n remaining requested positions (0 ≤ n ≤ 8); Other: Reserved
15: 0	PTXFSAVL [15: 0]	R	16’h100	PTXFSAVL: Periodic transmit data FIFO space available Indicates the remaining space for periodically transmitting FIFO This value is a 32 bit word. 16'h0: Periodic transmission FIFO full; 16 'h1: 1 word remaining; 16 'h2: 2 words remaining; 16'hn: n words remaining (0 ≤ n ≤ 256); 16 'h100: 256 words remaining; Other: Reserved.

32.5.5 OTG_FS host all channel interrupt register (OTG_FS_HAINT)

Address: 0x0414

Reset value: 0x0000 0000

When a signature event occurs on a certain channel, all channel interrupts of the host will interrupt the application program through the host channel interrupt bit of the controller interrupt register (HCINT bit of the OTG_FS_GINTSTS register). Each channel has a corresponding channel interrupt bit, with a total of 8 control bits. The application sets and clears this bit by setting and clearing the corresponding bit of the corresponding host channel x interrupt register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								HAINT [7: 0]							
								R							

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	HAINT [7: 0]	R	8'h0	HAINT: Channel interrupts Each bit represents a channel: bit 0 represents channel 0, bit 7 represents channel 7 0: No host channel interrupt occurred 1: Generate host channel interrupt

32.5.6 OTG_FS Host All Channel Interrupt Mask Register (OTG_FS_HAINTMSK)

Address: 0x0418

Reset value: 0x0000 0000

The host all channel interrupt mask register and the host all channel interrupt register are configured to interrupt the response when an event is generated

Use the program. Each channel has a corresponding interrupt mask bit, with a total of 8 bits.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW								HAINTM [7: 0]							
								RW							

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	HAINTM	RW	8'h0	HAINTM: Channel interrupt mask 0: Disable channel n interrupt 1: Enable channel n interrupt Each bit represents a channel: bit 0 represents channel 0 and bit 7 represents channel 7

32.5.7 OTG_FS host port control and status register (OTG_FS_HPRT)

Address: 0x0440

Reset value: 0x0000 0000

This register is only valid in host mode. An OTG host controller only supports one port.

This register holds information related to host ports, including USB reset, enable, suspend, wake up, connection status for each port

State and test mode information, etc. The bit designated rc_w1 can be passed through the host port of the host interrupt register

The interrupt bit (the HPRTINT bit of the OTG_FS_GINTSTS register) is used to trigger an interrupt to interrupt the application program. Interrupt service at port

In the service program, the application program must read this register and clear the bit that caused the interrupt. For the bit designated rc_w1, the application needs

Clear the interrupt by writing '1'.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.													PSPD [1: 0]		PTCTL
													R		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

PTCTL [3: 0]	PPWR	PLSTS [1: 0]	Res.	PRST	PSUSP	PRES	POCCHNG	POCA	PENCHNG	PENA	PCDET	PCSTS
	RW	R		RW	RS	RW	RC_W1	R	RC_W1	RC_W1	RC_W1	R

Bit	Name	R/W	Reset Value	Function
31: 19	Reserved	-	-	-
18: 17	PSPD	R	2'h0	PSPD: Port speed Indicates the device speed of connected ports 2 'b01: full speed device; 2 'b10: low speed equipment; Other: Reserved.
16: 13	PTCTL	RW	4'h0	PTCTL: Port test control The application enters test mode by writing a non-zero value to this bit, and the corresponding signal will appear at the port. 4 'b0000: non-test mode; 4 'b0001: Test_J mode; 4'b0010: Test_K mode; 4'b0011: Test_SEO_NAK mode; 4 'b0100: Test_Packet mode; 4 'b0101: Test_Force_Enable Other: Reserved
12	PPWR	RW	0	PPWR: Port power The application supplies power to the port by setting this bit, and the controller clears this bit when a current overflow event occurs. 0: no power supply; 1: Power supply.
11: 10	PLSTS	R	2'b0	PLSTS: Port line status Indicates the current logical state of the USB data cable Bit 10: logical state of OTG_FS_FS_DP line; Bit 11: logic state of OTG_FS_FS_DM line; Other: Reserved.
9	Reserved	-	-	-
8	PRST	RW	0	PRST: Port reset When the application sets this bit, a reset sequence is generated on the port. The application needs to control the reset time, and Clear this bit after bit completion. 0: Port not reset; 1: Port reset. The application needs to maintain the '1' state of this bit for at least 10ms in order to initiate the reset sequence of the port. The application can also clear this The '1' state is added 10ms before the bit. The USB specification does not define the maximum duration of the reset signal.
7	PSUSP	RS	0	PSUSP: Port suspend The application sets this bit to put the port into a suspended state. In this state, the controller only stops transmitting the SOF signal. Applications The PHY clock needs to be stopped by setting the port clock stop bit. A read of this bit will return the suspended state of the current port when the application sets the port reset bit or port wake for this register Bit, or the controller detects a remote wake-up signal, or the controller interrupt register's wake-up/remote wake-up detects interrupt bit or port The detection interrupt bit (either the WKUINT bit or the DISCINT bit of the OTG_FS_GINTSTS register) is set, and the controller will clear this power-on and power-off reset for the chip. 0: Port is not in suspend mode; 1: The port is in suspended mode.
6	PRES	RW	0	PRES: Port resume The application sets this bit to drive the port to wake up. The controller will output a drive wake-up signal until the application clears this bit. When the controller detects the USB remote wake-up sequence, according to the port wake-up/remote wake-up detection interrupt bit of the controller interrupt register

				As shown, the controller automatically outputs the wake-up sequence without application intervention. If the controller detects that the device is disconnected, it will automatically clear Except this bit. A read on this bit will return information whether the controller is outputting a wake-up signal. 0: no wake-up; 1: Output wake-up signal.
5	POCCHNG	RC_W1	0	POCCHNG Port overcurrent change The controller sets this bit when the port overcurrent bit (bit 4) of this register changes
4	POCA	R	0	POCA: Port overcurrent active Indicates whether overcurrent has occurred at the port. 0: No overcurrent; 1: overcurrent
3	PENCHNG	RC_W1	0	PENCHNG: Port enable/disable change The controller sets this bit when the port enable bit (bit 2) of this register changes
2	PENA	RC_W1	0	PENA: Port enable The port can only be enabled by the controller after the reset sequence, and the terminal is disabled when overcurrent occurs, the device is disconnected and the controller clears this bit Mouth. The program should not be able to set this bit by register write operation. This bit does not trigger an interrupt. 0: Port disabled; 1: Port enable
1	PCDET	RC_W1	0	PCDET: Port connect detected When the controller detects that a device is connected to a port, the host port interrupt bit of the controller interrupt register is triggered (HPRTINT bit of the OTG_FS_GINTSTS register) to generate an interrupt and set this bit. The application must write '1' Clear the interrupt.
0	PCSTS	R	0	PCSTS: Port connect status 0: The device is not connected to the port 1: The device is connected to this port

32.5.8 OTG_FS host channel x characteristic register (OTG_FS_HCCCHARx) (here x code channel number, x = 0... 7)

Address: 0x0500 + (channel number * 0x20)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHENA	CHDIS	ODDFRM	DAD [6: 0]							MCNT [1: 0]		EPTYP [1: 0]		LSDEV	Res.
RS	RS	RW	RW							RW		RW		RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPDIR	EPNUM [3: 0]				MPSIZ [10: 0]										
RW	RW				RW										

Bit	Name	R/W	Reset Value	Function
31	CHENA	RS	0	CHENA: Channel enable This bit is set by the application and cleared by the FS_OTG host controller. 0: channel prohibited; 1: Channel enable
30	CHDIS	RS	0	CHDIS: Channel disable By setting this bit, the application can immediately stop receiving/sending data on the channel, even if the transmission of data on the channel is not finished Cheng. An application can only consider a channel disabled until a channel disabling interrupt occurs.
29	ODDFRM	RW	0	ODDFRM: Odd frame The application tells the OTG host controller that the transmission should occur at odd/even frames by setting/clearing this bit. This bit is only valid for periodic transmissions (synchronous or interrupted). 0: even frame;

				1: Odd frame.
28: 22	DAD	RW	7'h0	DAD: Device address The application selects the desired device through this address.
21: 20	MCNT	RW	2'h0	MCNT: device address (Multicount) This field indicates the number of transmissions that the host must perform per frame on this periodic endpoint. Aperiodic transmissions do not use this parameter. 2 'b00: Reserved. Setting this value will produce an undefined result. 2 'b01: 1 transmission. 2 'b10: At this endpoint, 2 transmissions per frame need to be generated. 2 'b11: At this endpoint, 3 transmissions per frame need to be generated. Note: The value of this field should be set to at least '01'
19: 18	EPTYP	RW	2'b0	EPTYP: Endpoint Type Indicates the type of transfer selected 2'b00: control transmission 2 'b01: synchronous transmission 2'b10: block transmission 2 'b11: interrupt transmission
17	LSDEV	RW	0	LSDEV: Low speed device The application sets this bit to indicate that the device it is communicating with is a low speed device.
16	Reserved	-	-	-
15	EPDIR	RW	0	EPDIR: Endpoint direction Indicates whether the transmission is IN or OUT direction. 0: OUT 1: IN
14: 11	EPNUM	RW	4'h0	EPNUM: Endpoint number Indicates the number of the endpoint to communicate with
10: 0	MPSIZ	RW	11'h0	MPSIZ: Maximum packet size Indicates the maximum packet length of the selected endpoint

32.5.9 OTG_FS host channel x interrupt register (OTG_FS_HCINTx) (where x represents channel number, x = 0... 7)

Address: 0x0508 + (channel number * 0x20)

Reset value: 0x0000 0000

This register indicates the status of the channel in relation to the USB and AHB time. When the host channel interrupt bit of the controller interrupt register (HCINT bit of the OTG_FS_GINTSTS register) is set, the application program needs to first read all channel interrupt registers of the host (OTG_FS_HAINT) to obtain the channel number where the host channel interrupt occurred, and then read the interrupt register of the corresponding channel to obtain detailed information of the interrupt. The application clears the corresponding bits of the OTG_FS_HAINT register and the OTG_FS_GINTSTS register by clearing the corresponding bits of this register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					DTERR	FRMOR	BBERR	TXERR	Res.	ACK	NAK	STALL	Res.	CHH	XFRC
					RC_W1	RC_W1	RC_W1	RC_W1		RC_W1	RC_W1	RC_W1		RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10	DTERR	RC_W1	0	DTERR: Data toggle error
9	FRMOR	RC_W1	0	FRMOR: Frame overflow
8	BBERR	RC_W1	0	BBERR: Babble error
7	TXERR	RC_W1	0	TXERR: Transaction error

				Indicates that the following error occurred: CRC check failed Timeouts Bit padding error EOP failure
6	Reserved	-	-	-
5	ACK	RC_W1	0	ACK: ACK response received/transmitted interrupt
4	NAK	RC_W1	0	NAK: NAK response received interrupt
3	STALL	RC_W1	0	STALL: STALL response received interrupt
2	Reserved	-	-	-
1	CHH	RC_W1	0	CHH Channel halted Indicates that the transfer ended abnormally due to a USB transfer error, or an application abort request.
0	XFRC	RC_W1	0	XFRC: Transfer completed The transfer completed normally with no errors.

32.5.10 OTG_FS host channel x interrupt mask register (OTG_FS_HCINTMSKx) (where x is the channel number, x = 0... 7)

Address: 0x050C + (channel number * 0x20)

Reset value: 0x0000 0000

This register contains the interrupt enable bit of the interrupt flag bit in the USBFS_HCHxINTF register. If a bit of this register is set by software, the corresponding bit in the USBFS_HCHxINTF register can trigger a channel interrupt. The bits in this register can be set and cleared by software.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.					DTERRM	FRMORM	BBERRM	TXERRM	Res.	ACKM	NAKM	STALML	Res.	CHHM	XFRCM
					RW	RW	RW	RW		RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-
10	DTERRM	RW	0	DTERRM: Data toggle error mask 0: Mask interrupt; 1: Do not mask interrupts.
9	FRMORM	RW	0	FRMORM: Frame overrun mask 0: Mask interrupt; 1: Unmasked interrupt
8	BBERRM	RW	0	BBERRM: Babble error mask 0: Mask interrupt; 1: Unmasked interrupt
7	TXERRM	RW	0	TXERRM: Transaction error mask 0: Mask interrupt; 1: Do not mask interrupts.
6	Reserved	-	-	-
5	ACKM	RW	0	ACKM: ACK response received/transmitted interrupt mask 0: Mask interrupt; 1: Unmasked interrupt
4	NAKM	RW	0	NAKM: NAK response received interrupt mask 0: Mask interrupt; 1: Unmasked interrupt
3	STALML	RW	0	STALML: STALL response received interrupt mask 0: Mask interrupt; 1: Unmasked interrupt
2	Reserved	-	-	-
1	CHHM	RW	0	CHHM: Channel halted mask 0: Mask interrupt; 1: Unmasked interrupt
0	XFRCM	RW	0	XFRCM: Transfer completed mask

				0: Mask interrupt; 1: Unmasked interrupt

32.5.11 OTG_FS Host Channel x Transfer Length Register (OTG_FS_HCTSIZx) (where x is the channel number, x = 0... 7)

Address: 0x0510 + (channel number * 0x20)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DPID [1: 0]		PKTCNT [9: 0]										XFRSIZ [18:16]		
	RW		RW										RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XFRSIZ [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 29	DPID [1: 0]	RW	2'h0	DPID: Data PID The application uses this bit to inform the controller of the type of PID used for the first transfer. The controller will automatically control the type of PID for subsequent transmissions. 2'b00: DATAO; 2 'b01: retained; 2 'b10: DATA1; 2 'b11: SETUP (control)
28: 19	PKTCNT [9: 0]	RW	10'h0	PKTCNT: Packet count Through this bit, the application informs the controller of the number of packets it expects to receive (IN) or the number of packets it expects to send (OUT). The controller will automatically decrement this parameter after each successful OUT/IN packet is sent or received. Once the field is 0, the application The sequence will receive an interrupt, indicating that the transmission has ended normally.
18: 0	XFRSIZ [18: 0]	RW	19'h0	XFRSIZ Transfer size For OUT transmissions, these bits indicate the number of data bytes sent by the host during the transmission. For IN transmissions, these bits indicate the buffer length reserved by the application. For IN transmissions (periodic and aperiodic), the application needs to predefine this parameter as an integer multiple of the largest packet.

32.6 Registers in device mode

32.6.1 OTG_FS device configuration register (OTG_FS_DCFG)

Address: 0x0800

Reset value: 0x0820 0000

This register is used to configure the operating characteristics of the controller in device mode after a reset, or special control command, or enumeration. In

Do not modify this register after initialization.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			PFIVL [1: 0]		DAD [6: 0]							Res.	NZLSOHSK	DSPD [1: 0]	
			RW		RW								RW	RW	

Bit	Name	R/W	Reset Value	Function

31: 13	Reserved	-	-	-
12: 11	PFIVL [1: 0]	RW	2'h0	PFIVL: Periodic frame interval Indicates the percentage of the entire frame of the time at which periodic frame end interrupts were generated. The application can use this interrupt to determine the Whether the synchronous transfers have all been transferred. 2 'h00: 80% frame time; 2 'h01: 85% frame time; 2 'h10: 90% frame time; 2 'h11: 95% frame time.
10: 4	DAD [6: 0]	RW	7'h0	DAD: Device address After receiving the control command of SetAddress, the application populates this bit according to the parameters
3	Reserved	-	-	-
2	NZLSOHSK	RW	0	NZLSOHSK: Non-zero-length status OUT handshake signal During the state phase of control transmission, if a packet of non-zero length is received, the application can select to send a A handshake signal. 1: Send a STALL handshake to a non-zero length state OUT transmission, and do not transmit the received OUT packet to the application. 0: Select to send a handshake signal according to the NAK bit and STALL bit status of the device port control register, and send the received OUT packet Delivered to the application (zero length and non-zero length).
1: 0	DSPD [1: 0]	RW	2'h0	DSPD: Device speed Indicates the speed at which the application requires the controller to perform enumeration operations, or the maximum speed that the application can support. However, the actual The international bus speed can only be determined according to the speed of the connected USB host after the entire sequence is completed. 2 'b00: reserved; 2 'b01: retained; 2 'b10: retained; 2 'b11: Full speed (USB 1.1 transceiver clocked at 48 MHz)

32.6.2 OTG_FS device control register (OTG_FS_DCTL)

Address: 0x0804

Reset value: 0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				POPRGDNE	CGONAK	SGONAK	CGINAK	SGINAK	TCTL [2: 0]			GONSTS	GINSTS	SDIS	RWUSIG
				RW	W	W	W	W	RW			R	R	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11	POPRGDNE	RW	0	POPRGDNE: Power-on programming done With this bit, the application indicates that the configuration of the registers has been completed after waking up from the power-up state.
10	CGONAK	W	0	CGONAK: Clear global outnak A write to this bit will clear the global OUT NAK state.
9	SGONAK	W	0	SGONAK: Set global OUT NAK A write to this bit sets the global OUT NAK state.

				Through this bit, the application tells the controller to send NAK signals to all OUT endpoints. Before setting this bit, the program must confirm that the global OUT NAK valid bit of the controller interrupt register (the GONAKEFF bit of the OTG-FS_GINTSTS register) has been cleared.
8	CGINAK	W	0	CGINAK: Clear global IN NAK Writing this bit will clear the global IN NAK state
7	SGINAK	W	0	SGINAK: Set global IN NAK A write to this bit will set the NAK state of the global aperiodic IN. The program uses this bit to tell the controller to control all aperiodic IN Both endpoints send NAK handshake signals. The program must confirm that the global IN NAK valid bit of the controller interrupt register (the GINAKEFF bit of the OTG_FS_GINTSTS register) has been cleared before setting this bit
6: 4	TCTL	RW	3'h0	TCTL: Test control 3 'b000: test mode disabled; 3 'b001: Test_J mode; 3 'b010: Test_K mode; 3 'b011: Test_SEO_NAK mode; 3 'b100: Test_Packet mode; 3 'b101: Test_Force_Enable; Other: Reserved.
3	GONSTS	R	0	GONSTS Global OUT NAK status 0: A handshake signal is transmitted according to the state of the FIFO and the states of the NAK bit and the STALL bit. 1: No data is written to the receiving FIFO regardless of whether the storage area is empty. A NAK handshake signal is sent to all transmissions except SETUP, and all synchronous OUT packets are discarded. 。
2	GINSTS	R	0	GINSTS: Global IN NAK status 0: The handshake signal is transmitted according to the state of transmitting data in the FIFO. 1: NAK handshake signal is transmitted to all aperiodic IN endpoints regardless of the state of data IN the transmission FIFO
1	SDIS	RW	1	SDIS: Soft disconnect Through this bit, the application tells the OTG controller to perform a software disconnect operation. When this bit is set, the host will see that the device has been disconnected, and the device will not receive any signal from the USB bus. The controller will remain in the OFF state until the program clears this bit. 0: Normal operation. When this bit is cleared after the device software is disconnected, the controller will send a device connection event to the host, which will re-execute the enumeration operation. 1: The controller performs device software disconnection operation
0	RWUSIG	RW	0	RWUSIG: Remote wakeup signaling When the application sets this bit, the controller will send a remote wake-up signal to wake the USB host. The application must set this bit to exit the controller from the suspended state. According to the USB 2.0 specification, the program must clear this bit again between 1 and 15ms after setting it

In order for the USB host to recognize the device disconnect operation, the software disconnect (SDIS) bit needs to be held for a period of time. The following table lists

Minimum duration (depending on different states of the device). In order to accommodate the jitter of the clock, it is recommended that the application operate at a defined minimum time

An additional period of delay is reserved.

Table 32-6 Minimum duration of software disconnect

Operating speed	Device Status	Minimum duration
full speed	Suspend	1ms + 2.5 us
full speed	Idle	2.5us
full speed	Not idle or suspended (transmission operation being performed)	2.5us

32.6.3 OTG_FS Device Status Register (OTG_FS_DSTS)

Address: 0x0808

Reset value: 0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.										FNSOF [13: 8]					
										R					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FNSOF [7: 0]								Res.				EERR	ENUMSPD [1: 0]		SUSPSTS
R													R		R

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	-	-
21: 8	FNSOF [13: 0]	R	14'h0	The Frame number of the received SOF (Frame number of the received SOF).
7: 4	Reserved	-	-	-
3	EERR	R	0	EERR: Erratic error The controller sets this bit when some strange error occurs. If a strange error occurs, the OTG_FS controller will enter a suspend state and set the early suspend bit of the controller interrupt register (the ESUSP bit of the OTG_FS_GINTSTS register) to generate an interrupt. If the early suspend interrupt is caused by this bit, the application can only resolve it by performing a software disconnect.
2: 1	ENUMSPD [1: 0]	R	2'h1	ENUMSPD: Enumerated speed Indicates the execution speed selected by the OTG_FS controller through the sequence. 2 'b01: retained; 2 'b10: retained; 2 'b11: full speed (PHY clock running at 48 MHz); Other: Reserved
0	SUSPSTS	R	0	SUSPSTS Suspend status In device mode, this bit will be set if a suspend condition is detected on the USB bus. The controller will detect that USB data The line goes into a suspended state with no activity within 3ms. The controller exits suspend mode when: When there is activity on the USB data cable When the application program sets the remote wake signal bit of the device control register (RWUSIG bit of the OTG_FS_DCTL register)

32.6.4 OTG_FS Device IN Endpoint Universal Interrupt Mask Register (OTG_FS_DIEPMSK)

Address: 0x0810

Reset value: 0x0000 0000

This register cooperates with the device IN endpoint interrupt register (OTG_FS_DIEPINTx) to generate the IN endpoint interrupt. The corresponding IN endpoint interrupt corresponding to the OTG_FS_DIEPINTx register can be masked by configuring this register. All interrupts are masked by default.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.									INEPNEM	INEPNMM	ITTXFEMSK	TOM	Res.	EPDM	XFRCM
									RWE	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	INEPNEM	RW	0	INEPNEM: IN endpoint NAK effective mask 0: interrupt mask; 1: Interrupt is not masked
5	INEPNMM	RW	0	INEPNMM: Endpoint receives IN command mismatch mask (IN token received with EP mismatch mask)

				0: interrupt mask; 1: Interrupt is not masked
4	ITTXFEMSK	RW	0	ITTXFEMSK: IN token received when TxFIFO empty mask 0: interrupt mask; 1: Interrupt is not masked
3	TOM	RW	0	TOM: Timeout condition mask (Non-isochronous endpoints) 0: interrupt mask; 1: Interrupt is not masked
2	Reserved	-	-	-
1	EPDM	RW	0	EPDM: Endpoint disabled interrupt mask 0: interrupt mask; 1: The interrupt is not masked.
0	XFRCM	RW	0	XFRCM: Transfer completed interrupt mask 0: interrupt mask; 1: Interrupt is not masked

32.6.5 OTG_FS Device OUT Endpoint General Interrupt Mask Register (OTG_FS_DOEPMSK)

Address: 0x0814

Reset value: 0x0000 0000

This register is used in conjunction with the device OUT endpoint interrupt register (OTG_FS_DOEPINTx) to generate an OUT endpoint interrupt.

Each bit of the OTG_FS_DOEPINTx register can be masked by writing the corresponding bit of this register. By default, All interrupts are masked.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									B2BSTUP	Res.	OTEPDM	STUPM	Res.	EPDM	XFRCM
									RW		RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	B2BSTUP	RW	0	B2BSTUP: Back-to-back SETUP packets received mask 0: interrupt mask; 1: Interrupt is not masked
5	Reserved	-	-	-
4	OTEPDM	RW	0	OTEPDM: OUT token received when endpoint is disabled disabled mask) 0: interrupt mask; 1: Interrupt is not masked
3	STUPM	RW	0	STUPM: SETUP phase done mask Only valid for control endpoints 0: interrupt mask; 1: Interrupt is not masked
2	Reserved	-	-	-
1	EPDM	RW	0	EPDM: Endpoint disabled interrupt mask 0: interrupt mask; 1: The interrupt is not masked.
0	XFRCM	RW	0	XFRCM: Transfer completed interrupt mask 0: interrupt mask; 1: Interrupt is not masked

32.6.6 OTG_FS device all endpoint interrupt registers (OTG_FS_DAINT)

Address: 0x0818

Reset value: 0x0000 0000

When an event occurs at each endpoint, all endpoint interrupt registers of the device pass through the device OUT end of the controller interrupt register respectively

Point interrupt bit or device IN endpoint interrupt bit (OEPINT or IEPINT bit of OTG_FS_GINTSTS register) to generate an interrupt

Disconnect the application. Each endpoint has a corresponding bit, the OUT endpoint has 16 bits, and the IN endpoint has 16 bits. For the bidirectional end

Point, using both the IN and OUT bits. The corresponding bit of this register, when the application sets and clears the corresponding device endpoint x interrupts the send

The corresponding bits of the registers (OTG_FS_DIEPINTx and OTG_FS_DOEPINTx registers) are automatically set and cleared. 。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												OEPINT [3: 0]			
												R			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.												IEPINT [3: 0]			
												R			

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 16	OEPINT [3: 0]	R	4'h0	OEPINT: OUT endpoint interrupt bits Each bit corresponds to an OUT endpoint. Bit 16 corresponds to OUT endpoint 0 and bit 18 corresponds to OUT endpoint 3.
15: 4	Reserved	-	-	-
3: 0	IEPINT [3: 0]	R	4'h0	IEPINT: IN endpoint interrupt bits Each bit corresponds to an IN endpoint. Bit 0 corresponds to IN endpoint 0 and bit 3 corresponds to IN endpoint 3

32.6.7 OTG_FS All endpoint interrupt mask register (OTG_FS_DAINTMSK)

Address: 0x081C

Reset value: 0x0000 0000

The Device All Endpoint Interrupt Mask register is used in conjunction with the Device Endpoint Interrupt register to generate an interrupt to interrupt the application when a device endpoint event occurs. However, the corresponding bits of the device's all endpoint interrupt registers (OTG_FS_DAINT) are still set when masked.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												OEPM [3: 0]			
												RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.												IEPM [3: 0]			
												RWE			

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 16	OEPM [3: 0]	RW	4'h0	OEPM: OUT EP interrupt mask bits Each bit corresponds to an OUT endpoint. Bit 16 corresponds to OUT endpoint 0 and bit 19 corresponds to OUT endpoint 3. 0: Mask interrupt; 1: Unmasked interrupt
15: 4	Reserved	-	-	-
3: 0	IEPM [3: 0]	RW	4'h0	IEPM: IN EP interrupt mask bits Each bit corresponds to an IN endpoint. Bit 0 corresponds to IN endpoint 0, and bit 3 corresponds to IN endpoint 3. 0: Mask interrupt;

				1: Unmasked interrupt

32.6.8 OTG_FS Device VBUS Discharge Time Register (OTG_FS_DVBUSDIS)

Address: 0x0828

Reset value: 0x0000 17D7

This register defines the VBUS discharge time after the VBUS pulse during the SRP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBUSDT [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	VBUSDT [15: 0]	RW	16'h17D7	VBUSDT: Device VBUS discharge time Defines the VBUS discharge time after the VBUS pulse during the SRP. This value is equal to the VBUS discharge time/1024 calculated as a PHY clock. This value can be adjusted appropriately according to different VBUS loads

32.6.9 OTG_FS device VBUS pulse time register (OTG_FS_DVBUSPULSE)

Address: 0x082C

Reset value: 0x0000 05B8

This register defines the time of the VBUS pulse during the SRP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				DVBUSP [11: 0]											
				RW											

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11/0	DVBUSP [11: 0]	RW	12'h5B8	DVBUSP Device VBUS pulsing time Define the VBUS pulse time during SRP. This value is equal to the VBUS pulse time/1024 calculated as a PHY clock.

32.6.10 OTG_FS Device IN Endpoint FIFO Null Interrupt Mask Register (OTG_FS_DIEPEMMPSK)

Address: 0x0834

Reset value: 0x0000 0000

This register cooperates with the IN endpoint FIFO empty interrupt register (TXFE_OTG_FS_DIEPINTx) to generate an interrupt

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.												INEPTXFEM [3: 0]			
												RW			

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	INEPTXFEM [3: 0]	RW	4'h0	INEPTXFEM: IN EP Tx FIFO empty interrupt mask bits

				This bit is used to mask the corresponding bit of the OTG_FS_DIEPINTx register. One bit of the TXFE interrupt corresponds to an IN endpoint: bit 0 corresponds to IN endpoint 0, and bit 3 corresponds to IN endpoint 3. 0: Mask interrupt; 1: Unmasked interrupt

32.6.11 OTG_FS device control IN endpoint 0 control register (OTG_FS_DIEPCTL0)

Address: 0x0900

Reset value: 0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	Res.		SNAK	CNAK	TXFNUM [3: 0]				STALL	Res.	EPTYP [1: 0]		NAKSTS	Res.
RS	RS			W	W	RW				RS		R		R	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBAEP	Res.													MPSIZ [1: 0]	
R														RW	

Bit	Name	R/W	Reset Value	Function
31	EPENA	RS	0	EPENA: Endpoint enable The application sets this bit to initiate data transfer on endpoint 0. The controller clears this bit before generating the following endpoint interrupts: Endpoint prohibition; Transmission complete.
30	EPDIS	RS	0	EPDIS: Endpoint disable By setting this bit, the application can immediately stop the data transfer on the endpoint, even if the transfer has not been completed. The application needs to wait until Endpoint disabled interrupts to be sure that this endpoint has been disabled. The controller clears this bit when setting the endpoint interrupt interrupt. The application has only This bit can only be set when the endpoint is already enabled.
29: 28	Reserved	-	-	-
27	SNAK	W	0	SNAK: Set NAK Writing this bit sets the endpoint's NAK status By setting this bit, the program can tell the controller to send a NAK handshake signal. The controller also sets this bit when it receives a SETUP packet.
26	CNAK	W	0	CNAK: Clear NAK Writing this bit clears the endpoint's NAK state
25: 22	TXFNUM [3: 0]	RW	4'h0	TXFNUM: Transmission FIFO number (TxFIFO number) This bit sets the FIFO number assigned to IN endpoint 0
21	STALL	RS	0	STALL: STALL handshake The application can only set this bit, and the controller clears it when it receives the SETUP command. Even if the NAK bit is set at the same time, or The global IN NAK bit, or the global OUT NAK bit, and the STALL bit still have high priority
20	Reserved	-	-	-
19: 18	EPTYP [1: 0]	R	2'h0	EPTYP: Endpoint type For the control endpoint, set to '00' by the hardware.
17	NAKSTS	R	0	NAKSTS: NAK status Indicates the following: 0: According to the FIFO state, the controller sends a non-NAK handshake signal; 1: The controller sends a NAK handshake signal. As long as this bit is set, whether the application sets this bit or the controller sets this bit, the controller will stop data transfer, and

				Regardless of whether the data in the transmit FIFO is valid or not. Regardless of this bit setting, the controller always sends an ACK handshake response by the number of SETUP According to package
16	Reserved	-	-	-
15	USBAEP	R	1	USBAEP: USB active endpoint This bit is always' 1 ', indicating that control endpoint 0 is active in any configuration situation
14: 2	Reserved	-	-	-
1: 0	MPSIZ [1: 0]	RW	2'h0	MPSIZ: Maximum packet size The application uses this bit to configure the maximum packet length for this endpoint 2 'b00: 64 bytes; 2 'b01: 32 bytes; 2 'b10: 16 bytes; 2 'b11: 8 bytes.

32.6.12 OTG device endpoint x control register (OTG_FS_DIEPCTLx) (where x is the endpoint number, x = 1... 3)

Address: 0x0900 + (x * 0x20)

Reset value: 0x0000 0000

The application controls the operating characteristics of non-zero endpoints through these registers.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	SODDFRM	SD0PID/SEVNFRM	SNAK	CNAK	TXFNUM [3: 0]				STALL	Res.	EPTYP [1: 0]		NAKSTS	EONUM/DPID
RS	RS	W	W	W	W	RW				RW/RS		RW		R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBAEP	Res.					MPSIZ [10: 0]									
RW						RW									

Bit	Name	R/W	Reset Value	Function
31	EPENA	RS	0	EPENA: Endpoint enable The application sets this bit to initiate data transfer on the endpoint. The controller clears this bit before generating the following endpoint interrupts: SETUP phase completed Endpoint prohibition Transfer complete
30	EPDIS	RS	0	EPDIS: Endpoint disable By setting this bit, the application can immediately stop the data transfer on the endpoint, even if the transfer has not been completed. The application needs to wait until Endpoint disabled interrupts to be sure that this endpoint has been disabled. The controller clears this bit when setting the endpoint interrupt interrupt. The application has only You can only set this bit when the endpoint is already enabled
29	SODDFRM	W	0	SODDFRM: Set odd frame Only valid for synchronized IN and OUT endpoints. Write this bit to select odd frames in EONUM
28	SD0PID	W	0	SD0PID: Set DATA0 PID Only valid for interrupt/block transmission IN endpoints. Writing this bit will set the data PID bit to DATA0
	SEVNFRM	W	0	SEVNFRM: Set even frames Valid for Sync IN endpoints only. Write this bit to select even frames in EONUM

27	SNAK	W	0	SNAK: Set NAK Writing this bit sets the endpoint's NAK status By setting this bit, the program can tell the controller to send a NAK handshake signal. The controller will also receive a SETUP packet or an OUT endpoint Set this bit on end-of-transfer interrupt
26	CNAK	W	0	CNAK: Clear NAK Writing this bit clears the endpoint's NAK state
25: 22	TXFNUM [3: 0]	RW	4'h0	TXFNUM: Transmission FIFO number (TxFIFO number) This bit sets the FIFO number assigned to the endpoint, and each valid IN endpoint must be assigned a separate FIFO number. This bit is only valid for the IN endpoint.
21	STALL	RS	0	STALL: STALL handshake Valid only for non-control, non-synchronous IN endpoints (operation type is rw) The application sets this bit and the endpoint responds to all host commands with STALL. Even if the NAK bit is set at the same time, or The global IN NAK bit, or the global OUT NAK bit, the STALL bit still has the highest priority. Only the application can clear this bit, not the controller. Valid only for control endpoints (operation type is rs) The application can only set this bit, and the controller clears it when it receives the SETUP command. Even if the NAK bit, or the global IN NAK bit, or the global OUT NAK bit are set at the same time, the STALL bit still has a high priority. However that controller will always respond to the SETUP packet with an ACK handshake
20	Reserved	-	-	-
19: 18	EPTYP [1: 0]	RW	2'h0	EPTYP: Endpoint type This bit indicates the transmission type of the endpoint 2 'b00: control 2 'b01: synchronization; 2'b10: block transmission; 2 'b11: interrupted;
17	NAKSTS	R	0	NAKSTS: NAK status Indicates the following status: 0: According to the FIFO state, the controller sends a non-NAK handshake; 1: The controller sends a NAK handshake signal. As long as this bit is set, whether it is an application setting or a controller setting: For asynchronous IN endpoints, the controller will stop data transmission regardless of whether the data IN the sending FIFO is valid or not. For the synchronous IN endpoint, the controller will send a zero-length packet even if the data IN the sending FIFO is valid. Regardless of this bit setting, the controller always sends an ACK handshake in response to the SETUP packet
16	EONUM	R	0	EONUM: Even/odd frame Valid for Sync IN endpoints only: The frame number indicating that the endpoint performs synchronous data transmission and reception. The program must apply to that SEVNFRM and SODDFRM bit in this register The even/odd frame numbers of the desired frames are set according to the state. 0: even frame; 1: Odd frame
	DPID	R	0	DPID: Endpoint data PID

				Only valid for interrupt/block transfer IN endpoints: This bit holds the PID of packets transmitted through this endpoint. The application must configure this bit after enabling the endpoint, select The PID of the first packet sent or received. The application configures either DATA0 or DATA1 through the SD0PID register bit. 0: DATA0; 1: DATA1
15	USBAEP	RW	0	USBAEP: USB active endpoint Indicates whether this endpoint is an active endpoint under the current configuration. The controller clears this bit for all endpoints after detecting a USB reset (except endpoint 0), the application must set this bit as needed after receiving the SetConfiguration and SetInterface commands.
14: 11	Reserved	-	-	-
10: 0	MPSIZ [10: 0]	RW	11'h0	MPSIZ: Maximum packet size The application uses this bit to configure the maximum packet length for this endpoint The value of this bit is in bytes

32.6.13 OTG_FS device control OUT endpoint 0 control register (OTG_FS_DOEPCTL0)

Address: 0x0B00

Reset value: 0x0000 8000

This section describes the control register for device control OUT endpoint 0. Non-0 control endpoints use the corresponding endpoint 1-3 control registers.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	Res.		SNAK	CNAK	Res.				STALL	SNPM	EPTYP [1: 0]		NAKSTS	Res.
RS	R			W	W					RS	RW	R		R	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBAEP	Res.													MPSIZ	
R														R	

Bit	Name	R/W	Reset Value	Function
31	EPENA	RS	0	EPENA: Endpoint enable The application sets this bit to initiate data transfer on endpoint 0. The controller clears this bit before generating the following endpoint interrupts: SETUP phase completed; Endpoint prohibition; Transfer complete
30	EPDIS	R	0	EPDIS: Endpoint disable The application cannot disable control of OUT endpoint 0
29: 28	Reserved	-	-	-
27	SNAK	W	0	SNAK: Set NAK Writing this bit sets the endpoint's NAK status. By setting this bit, the application can tell the controller to send a NAK handshake signal. The controller will also receive a SETUP packet or transmit Set this bit when you end the interrupt
26	CNAK	W	0	CNAK: Clear NAK Setting this bit clears the endpoint's NAK status
25: 22	Reserved	-	-	-
21	STALL	RS	0	STALL: STALL handshake The application can only set this bit; Cleared by the controller when a SETUP command is received. Even if the NAK bit is set at the same time, Or the global OUT NAK bit, the STALL bit still has high priority. However, regardless of how this bit is set, that controller always handshake with ACK

				To respond to that SETUP packet
20	SNPM	RW	0	SNPM: Snoop mode Setting this bit will put the endpoint into listening mode, in which the controller will The correctness of the OUT package is not detected.
19: 18	EPTYP [1: 0]	R	2'h0	EPTYP: Endpoint type Set to '00' by hardware.
17	NAKSTS	R	0	NAKSTS: NAK status Indicates the following status: 0: According to the FIFO state, the controller sends a non-NAK handshake signal. 1: The controller sends a NAK handshake signal. As long as this bit is set, whether it is an application setting or a controller setting: The controller will stop data transmission regardless of whether the data in the receiving FIFO is valid or not. Regardless of this bit setting, the controller Always send ACK handshake signal in response to SETUP packet
16	Reserved	-	-	-
15	EPACT	R	1	USBAEP: USB active endpoint Always' 1 ', indicating that the control endpoint 0 is active under any configuration.
14: 2	Reserved	-	-	-
1: 0	MPSIZ [1: 0]	R	2'h0	MPSIZ: Maximum packet size The maximum packet length of control OUT endpoint 0 must coincide with the maximum packet length of control IN endpoint 0. 2 'b00: 64 bytes; 2 'b01: 32 bytes; 2 'b10: 16 bytes; 2 'b11: 8 bytes.

32.6.14 OTG_FS device OUT endpoint x control register (OTG_FS_DOEPCTLx) (where x is the endpoint number, x = 1... 3)

Address: 0x0B00 + (x * 0x20)

Reset value: 0x0000 0000

The application uses this register to control the operating characteristics of endpoints other than endpoint 0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPENA	EPDIS	SODDFRM/SD1PID	SD0PID/SEVNFRM	SNAK	CNAK	Res.				STALL	SNPM.	EPTYP		NAKSTS	EONUM/DPID
RS	RS	W	W	W	W					RW/RS	RW	RW		R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBAEP	Res.					MPSIZ [10: 0]									
RW						RW									

Bit	Name	R/W	Reset Value	Function
31	EPENA	RS	0	EPENA: Endpoint enable Valid for both IN and OUT endpoints. The application sets this bit to initiate data transfer on the endpoint. The controller clears this bit before generating the following endpoint interrupts: SETUP phase completed; Endpoint prohibition; Transfer complete
30	EPDIS	RS	0	EPDIS: Endpoint disable By setting this bit, the application can immediately stop the data transfer on the endpoint, even if the transfer has not been completed. The application needs to wait until

				Endpoint disabled interrupts to confirm that this endpoint has been disabled. The controller clears this bit when setting an endpoint interrupt. The application is only available in This bit can only be set when the endpoint is already enabled. 。
29	SD1PID	W	0	SD1PID: Set DATA1 PID Valid only for interrupt/block transmission IN/OUT endpoints. Writing this bit will set the data PID bit to DATA1
	SODDFRM	W	0	SODDFRM: Set odd frame Only valid for synchronous OUT endpoints When this bit is set, odd frames are selected in the parity frame field (EONUM)
28	SD0PID	W	0	SD0PID: Set DATA0 PID Valid only for interrupt/block transmission OUT endpoints. Writing this bit will set the data PID bit to DATA0
	SEVNFRM	W	0	SEVNFRM: Set even frames Valid for Synchronous OUT endpoints only. When this bit is set, even frames are selected in the parity frame field (EONUM)
27	SNAK	W	0	SNAK: Set NAK Writing this bit sets the endpoint's NAK status. By setting this bit, the application can control the controller to send the NAK handshake signal. The controller will also receive the SETUP package or OUT Set this bit when the endpoint transmission end interrupts
26	CNAK	W	0	CNAK: Clear NAK Writing this bit clears the endpoint's NAK state
25: 22	Reserved	-	-	-
21	STALL	RS	0	STALL: STALL handshake Valid only for non-control, non-synchronous OUT endpoints (operation type is rw) The application sets this bit and the endpoint responds to all commands from the host with STALL. Even if NAK is set at the same time Bit, or global IN NAK bit, or global OUT NAK bit, the STALL bit still has the highest priority. Only applications can clear Except this bit, the controller cannot. Valid only for control endpoints (operation type is rs) The application can only set this bit, and the controller clears it when it receives the SETUP command. Even if the NAK bit, or the global IN NAK bit, or the global OUT NAK bit are set at the same time, the STALL bit still has a high priority. However that controller will always respond to the SETUP packet with an ACK handshake
20	SNPM	RW	0	SNPM: Snoop mode Setting this bit will put the endpoint into listening mode. In listening mode, before the controller writes the OUT packet to the application buffer Data is not checked for correctness
19: 18	EPTYP [1: 0]	RW	2’h0	EPTYP: Endpoint type This bit indicates the transmission type of the endpoint: 2 'b00: control 2 'b01: Synchronization 2'b10: block transmission 2 'b11: Interrupt
17	NAKSTS	R	0	NAKSTS: NAK status Indicates the following status: 0: According to the FIFO state, the controller sends a non-NAK handshake signal.

				1: The controller sends a NAK handshake signal. As long as this bit is set, whether it is an application setting or a controller setting: The controller will stop data transmission regardless of whether the receiving FIFO can receive data. Regardless of this bit setting, the controller always sends an ACK handshake response SETUP packet
16	EONUM	R	0	EONUM: Even/odd frame Only valid for synchronous OUT endpoints: The frame number indicating that the endpoint performs synchronous data transmission and reception. The program must be based on that SEVNFRM and SODDFRM bit in this register To set even/odd frame numbers. 0: even frame; 1: Odd frame
	DPID	R	0	DPID: Endpoint data PID Only valid for interrupt/block transmission OUT endpoints: This bit holds the PID of packets transmitted through this endpoint. The program must configure this bit after enabling the endpoint, select the first send The PID of the packet sent or received. The application configures either DATA0 or DATA1 through the SD0PID register bit. 0: DATA0; 1: DATA1
15	USBAEP	RW	0	USBAEP: USB active endpoint Indicates whether this endpoint is an active endpoint under the current configuration. The controller clears this bit for all endpoints (except endpoint 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the program must set this bit as needed
14: 11	Reserved	-	-	-
10: 0	MPSIZ [10: 0]	RW	11'h0	MPSIZ: Maximum packet size The application uses this bit to configure the maximum packet length for this endpoint Length Value in bytes

32.6.15 OTG_FS device endpoint x interrupt register (OTG_FS_DIEPINTx) (where x is the endpoint number, x = 0... 3)

Address: 0x0908 + (x * 0x20)

Reset value: 0x0000 0080

This register indicates the event status of the respective endpoint related to USB and AHB. When the IN endpoint interrupt bit of the controller interrupt register (IEPINT bit of the OTG_FS_GINTSTS register) is' 1 ', the program must first read all endpoint interrupt registers of the device (OTG_FS_DAINT) to obtain the endpoint number where the event occurred, and then read the interrupt register of the corresponding endpoint to obtain detailed information. The application must clear the corresponding bits of the OTG_FS_DAINT and OTG_FS_GINTST registers by clearing this bit.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								TXFE	INEPNE	Res.	ITTXFE	TOC	Res.	EPDISD	XFRC
								R	RC_W1		RC_W1	RC_W1		RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-

7	TXFE	R	1	TXFE: Transmit FIFO empty This interrupt occurs when the transmit FIFO corresponding to this endpoint is empty or half-empty. The TxFIFO null level bit of the controller AHB configuration register (the TXFELVL bit of the OTG_FS_GAHBCFG register) will determine whether the transmit FIFO generates an interrupt in the null or half-null state.
6	INEPNE	RC_W1	0	INEPNE: IN endpoint NAK effective Writing to the CNAK bit of the OTG_FS_DIEPCTLx register will clear the NAK status of the IN endpoint, as well as this bit. This interrupt means that the controller detects that the NAK has been set (either by the application or by the controller), and this interrupt indicates that the NAK bit set by the application has taken effect. This interrupt does not guarantee that a NAK signal has been sent on the USB cable. The STALL bit has priority over the NAK bit.
5	Reserved	-	-	-
4	ITTXFE	RC_W1	0	ITTXFE: IN token received when TxFIFO is empty Valid only for aperiodic IN endpoints. A command for IN is received indicating that the transmit FIFO (periodic/aperiodic) associated with this endpoint is empty. This interrupt is only when an IN life is received Let the endpoint of generate.
3	TOC	RC_W1	0	TOC Timeout condition Valid only for control IN endpoints. Indicates that the controller has detected a timeout state since the last IN command was received by the endpoint
2	Reserved	-	-	-
1	EPDISD	RC_W1	0	EPDISD: Endpoint disabled interrupt This interrupt indicates that this endpoint has been disabled at the request of the application
0	XFRC	RC_W1	0	XFRC: Transfer completed interrupt This interrupt indicates that the configured transmission on the endpoint, whether on the AHB or USB side, has completed

32.6.16 OTG_FS device endpoint x interrupt register (OTG_FS_DOEPINTx) (where x is the endpoint number, x = 0... 3)

Address: 0x0B08 + (x * 0x20)

Reset value: 0x0000 0000

This register indicates the event status of the respective endpoint related to USB and AHB. When the OUT endpoint interrupt bit of the controller interrupt register (the OEPINT bit of the OTG_FS_GINTSTS register) is' 1 ', the application must first read all endpoint interrupt registers of the device (OTG_FS_DAINT) to obtain the endpoint number of the event, and then read the interrupt register of the corresponding endpoint to obtain detailed information. The application must clear the corresponding bits of the OTG_FS_DAINT and OTG_FS_GINTST registers by clearing this bit.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									B2BSTUP	Res	OTEPDIS	STUP	Res.	EPDISD	XFRC
									RC_W1		RC_W1	RC_W1		RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	B2BSTUP	RC_W1	0	B2BSTUP: Back-to-back SETUP packets received Valid only for control OUT endpoints.

				Indicates that the endpoint has received more than 3 consecutive SETUP packets
5	Reserved	-	-	-
4	OTEPDIS	RC_W1	0	OTEPDIS: OUT token received when endpoint disabled Valid only for control OUT endpoints. Indicates that an OUT command was received when the endpoint was disabled, and this interrupt is only generated on the endpoint that received the OUT command.
3	STUP	RC_W1	0	STUP SETUP phase done Valid only for control OUT endpoints. Indicates that the SETUP phase associated with this control endpoint has completed, and there will be no more consecutive SETUP packets in the current transmission. After this interrupt is generated, the application can begin processing received SETUP packets.
2	Reserved	-	-	-
1	EPDISD	RC_W1	0	EPDISD: Endpoint disabled interrupt This interrupt indicates that this endpoint has been disabled at the request of the application
0	XFRC	RC_W1	0	XFRC: Transfer completed interrupt This interrupt indicates that the configured transmission on the endpoint, whether on the AHB side or on the USB side, has completed

32.6.17 OTG_FS Device IN Endpoint 0 Transfer Length Register (OTG_FS_DIEPTSIZ0)

Address: 0x0910

Reset value: 0x0000 0000

The application must configure this register before enabling endpoint 0. Once the endpoint of the control register is enabled through the device control endpoint 0

Bit (EPENA bit of the OTG_FS_DIEPCTL0 register) enables endpoint 0, and only the controller can modify this register.

The application can only read this register until the endpoint enable bit is cleared.

Non-zero endpoints use registers from endpoint 1 to endpoint 15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.										PKTCNT [1: 0]		Res.			
										RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.									XFRSIZ [6: 0]						
									RW						

Bit	Name	R/W	Reset Value	Function
31: 21	Reserved	-	-	-
20: 19	PKTCNT [1: 0]	RW	2'h0	PKTCNT: Packet count Indicates the number of USB packets needed to transmit "transmission length" data on endpoint 0. Every time the controller reads a packet (maximum length packet and short packet) from the sending FIFO, the value of this register is automatically decreased by 1
18: 7	Reserved	-	-	-
6: 0	XFRSIZ [6: 0]	RW	7'h0	XFRSIZ Transfer size Indicates the number of bytes to be transmitted on endpoint 0. When the number of bytes transferred is reduced to 0, the controller generates an interrupt and notifies the application. can After the end of each packet, set this register value to the maximum transmission length of the endpoint. The controller will automatically decrement this field when sending FIFO write data from the storage area.

32.6.18 OTG_FS Device OUT Endpoint 0 Transfer Length Register (OTG_FS_DOEPTSIZ0)

Address: 0x0B10

Reset value: 0x0000 0000

The application must configure this register before enabling endpoint 0. Once endpoint 0 controls the endpoint of the register by device control endpoint 0

When the enable bit (EPENA bit of the OTG_FS_DOEPCTL0 register) is enabled, only the controller can modify this register. control

The application can only read this register until the endpoint enable bit is cleared.

Endpoints other than 0 use registers for endpoint 1 through endpoint 3.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	STUPCNT		Res.									PKTCNT	Res.		
	RW											RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									XFRSIZ [6: 0]						
									RW						

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 29	STUPCNT	RW	2'h0	STUPCNT: SETUP packet count This bit indicates the number of consecutive SETUP packets that the endpoint can receive 2 'b01: 1 pack; 2 'b10: 2 packs; 2 'b11: 3 packs
28: 20	Reserved	-	-	-
19	PKTCNT	RW	0	PKTCNT: Packet count When a packet is written to the receive FIFO, this bit is decremented until it decreases to 0.
18: 7	Reserved	-	-	-
6: 0	XFRSIZ	RW	7'h0	XFRSIZ Transfer size Indicates the number of bytes transferred on endpoint 0. The controller generates an interrupt when the transmission length is 0 and notifies the application. You can set this register value to the maximum transmission length of the endpoint at the end of each packet and generate an interrupt at the end of each packet. The controller automatically decrements this field when a packet is written to the storage area from the receiving FIFO

32.6.19 OTG_FS Device Endpoint x Transfer Length Register (OTG_FS_DIEPTSIZx) (where x is the endpoint number, x = 1... 3)

Address: 0x0910 + (x * 0x20)

Reset value: 0x0000 0000

The application must configure this register before enabling the endpoint. Once the endpoint passes the endpoint enable bit of the device endpoint x control register

(EPENA bit of OTG_FS_DIEPCTLx register) is enabled, only the controller can modify this register. Controller cleared

The application can only read this register until the endpoint enable bit.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	MCNT [1: 0]		PKTCNT [9: 0]										XFRSIZ [18:16]		
	RW		RW										RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XFRSIZ [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 29	MCNT [1: 0]	RW	2'h0	MCNT: Multi count

				For periodic IN endpoints, this bit indicates the number of packets to be transmitted during one frame of USB. Based on this bit, the controller calculates the data PID number sent by the synchronous IN endpoint. 01: 1 pack 10: 2 packs 11: 3 packs Other: Reserved
28: 19	PKTCNT [9: 0]	RW	10'h0	PKTCNT: Packet count Indicates the total number of packets to be sent on the endpoint. This field is automatically decremented every time a packet is read from the sending FIFO
18: 0	XFRSIZ [18: 0]	RW	19'h0	XFRSIZ: Transmission Length This bit indicates the transmission length of the current endpoint. The controller generates an interrupt when this bit is 0 and notifies the application. This register can be configured to be the maximum transmission length of the endpoint and generate an interrupt at the end of each packet. The controller automatically decrements this field each time a FIFO write data is sent from the storage area

32.6.20 OTG_FS device endpoint x transfer length register (OTG_FS_DOEPTSIZx) (where x is the endpoint number, x = 1... 3)

Address: 0x0B10 + (x * 0x20)

Reset value: 0x0000 0000

The application must configure this register before enabling the endpoint. Once the endpoint passes the endpoint enable bit of the device endpoint x control register

(EPENA bit of the OTG_FS_DOEPCTLx register) is enabled, only the controller can modify this register. When the controller clears

The application can only read this register until the endpoint enable bit.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	RXDPID/STUPCNT		PKTCNT [9: 0]										XFRSIZ [18:16]		
	R/RW		RW										RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XFRSIZ [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30: 29	RXDPID	R/RW	2'h0	RXDPID: Received data PID This bit is only valid for synchronous OUT endpoints. Data PID indicating the last packet received: 00: DATA0; 01: DATA2; 10: DATA1 11: MDATA
	STUPCNT	R/RW	2'h0	STUPCNT: SETUP packet count This bit is only valid for the control OUT endpoint. Indicates the number of consecutive SETUP packets received by the endpoint: 01: 1 pack; 10: 2 packs; 11: 3 packs
28: 19	PKTCNT [9: 0]	RW	10'h0	PKTCNT: Packet count Indicates the number of USB packets transmitted on this endpoint. The controller automatically decrements this field each time a packet is written from the storage area to the receiving FIFO.
18: 0	XFRSIZ [18: 0]	RW	19'h0	XFRSIZ Transfer size This bit indicates the number of bytes to be transmitted by this endpoint. The controller generates an interrupt notification application when this domain is 0. You can configure this domain as

				the maximum transmission length of the endpoint and generate an interrupt at the end of each packet. The controller automatically decrements this value each time data is written from the receiving FIFO to the memory area.

32.6.21 OTG_FS device IN endpoint transfer FIFO status register (OTG-FS_DTXFSTSx) (where x is the endpoint number, x = 0... 3)

Address: 0x0918 + (x * 0x20)

Reset value: 0x0000 0100

This register is a read-only register, which holds the remaining space information of the sending FIFO of the IN endpoint of the device.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPTFSAV [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	INEPTFSAV [15: 0]	R	16'h100	INEPTFSAV: IN endpoint TxFIFO space available Indicates the remaining space information of the transmit FIFO of the IN endpoint, and the register value is IN 32-bit words. 16 'h0: transmit FIFO full; 16 'h1: 1 word remaining; 16 'h2: 2 words remaining; 16'hn: n words remaining (where 0 < n < 256); 16 'h100: 256 words remaining; Other: Reserved.

32.7 OTG_FS Power and Clock Gating Register (OTG_FS_PCGCCTL)

Address: 0x0E00

Reset value: 0x0000 0000

This register is valid in both device mode and host mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.														GATEHCLK	STPPCLK
														RW	RW

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	GATEHCLK	RW	0	GATEHCLK: Gate HCLK By setting this bit, the application can control HCLK to wake up the logical module when the USB is suspended or the session is invalid. The application is in Clears this bit when USB wakes up or when a new session is initiated.
0	STPPCLK	RW	0	STPPCLK: Stop PHY clock The application can stop the clock drive of the PHY by setting this bit when the USB is suspended, or the session is invalid, or the device is disconnected Move. The application can clear this bit when the USB wakes up or a new session is initiated

33. Ethernet (ENET)

33.1 Introduction to Ethernet

Ethernet peripherals can realize Ethernet transmission and data reception according to IEEE 802.3-2002 standard

Ethernet provides configurable, flexible peripherals to meet a variety of applications and customer needs. It supports two industry standard interfaces to the external physical layer: Media Independent Interface (MII) and Reduced Media Independent Interface (RMII) defined in the specification. It can be used in switches, network interface cards and other applications.

Ethernet complies with the following standards:

- IEEE 802.3-2002 standard Ethernet MAC
- Accurate Network Clock Synchronization for IEEE 1588-2002 Standard
- AMBA 2.0 for AHB Master/Slave Port
- RMII specifications as defined by the RMII Association

33.2 Ethernet features

Ethernet peripherals include the following features:

- Supports 10/100 Mbit/s data transmission rate via external PHY interface
- IEEE 802.3 compatible MII interface for communication with external Fast Ethernet PHY interface
- Supports full-duplex and half-duplex work
  - Support half-duplex work of CSMA/CD protocol
  - Supports full-duplex operation of IEEE 802.3 x processes
  - In the full duplex operation mode, the received pause control frame can be selectively sent to the user application program
  - Half-duplex operation supporting back pressure flow control
  - When the input flow control signal fails in full duplex mode, the PAUSE frame will be automatically sent
- The preamble and start frame data (SFD) are inserted in the transmitted packet and these fields are deleted in the received packet
- Automatically calculates CRC and generates controllable padding bits on a frame basis
- Automatic removal of padding bits/CRC is optional when receiving a frame
- Programmable frame length to support standard frames up to 16K bytes
- Programmable frame gaps (40 ~ 96 bits, varying in 8 bits)
- Supports multiple flexible address filtering modes
  - Up to 4 48-bit (DA) address filters with a mask per byte
  - Do more than 3 48-bit SA address comparison detection, each byte has a mask
  - 64-bit hash filter for multicast and unicast (DA) addresses
  - Optionally let all multicast address frames pass
  - Promiscuous mode, which supports no filtering during network monitoring, allowing all frames to pass directly
- Returns independent 32-bit status information for packets sent and received

- IEEE 802.1 Q VLAN tags that support detection of received frames
- The application program has independent sending, receiving and controlling interfaces
- Supports mandatory network statistics using RMON/MIB counters (RFC2819/RFC2665)
- Use the MDIO interface to configure and manage PHY
- Detect LAN wake-up frames and AMD's Magic PacketTM frames
- Receiving function: checksum and load shedding of IPv4 and TCP received packets under Ethernet frame framework
- Advanced receive capabilities: IPv4 header checksum and TCP, UDP or ICMP checksum checks in IPv4 and IPv6 datagrams
- Ethernet frame timestamps described in IEEE 1588-2002 are supported. Use a 64-bit timestamp for each frame transmit or receive state
- 2 FIFOs: 2 KB transmit FIFO with programmable depth, 2KB receive FIFO with programmable depth
- The receive state after the EOF is transmitted is inserted into the receive FIFO at the time of receive, and multi-frame storage is enabled in the receive FIFO without requiring another FIFO to store the receive state of these frames
- Optionally filter error frames when received and do not forward them to the application in store-and-forward mode
- Optional forward valid frames with insufficient size
- Supports generating pulses to count the number of frames lost and corrupted (due to overflow) in the received FIFO
- Supports the use of store-and-forward mechanism when sending
- Automatically generate pause frame control or back pressure signal to MAC controller according to FIFO filling level (threshold configurable)
- Handling automatic retransmission of conflicting frames when transmitting
- Drop frames under late collision, over-collision, over-delay, and under-load conditions
- Software Control Empty Send FIFO
- Calculate and insert IPv4 header checksums and TCP, UDP, or ICMP checksums into transport frames in store and forward mode
- Supports internal loopback MII interface for debugging

33.2.1 DMA features

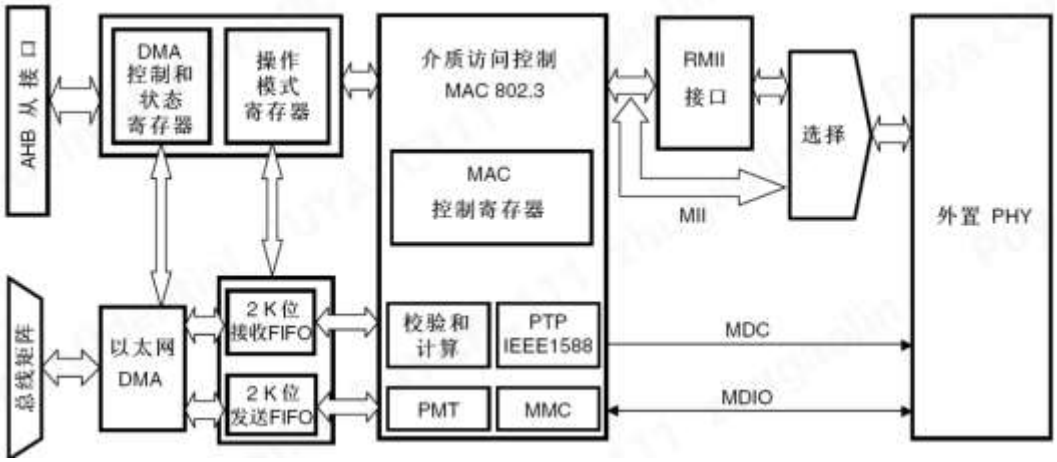
- Under the AHB slave interface, all types of AHB burst transmission are supported
- Under the AHB main interface, the software can select the type of AHB burst transmission (burst of fixed or unfixed length)
- Optional address alignment burst from the AHB master port
- Optimizing packet-oriented DMA transmission using frame delimiters
- Supports byte alignment to address data buffer areas
- Double buffer (ring) and linked list (chain) descriptor list
- Descriptor architecture, allowing large data block transfer with minimal CPU intervention
- Each descriptor can transmit up to 2KB of data

- Comprehensive status reporting for normal operation and error transmission
- Individually programmable burst sizes for sending and receiving DMA, optimizing bus usage
- Programmable interrupt options for different operating conditions
- Interrupt control after transmission/reception of each frame
- Receive and transmit engines two-point loop or fixed priority arbitration
- Start/Stop Mode
- Current Tx/Rx buffer area pointer as status register
- Current Tx/Rx descriptor pointer as status register

33.2.2 PTP Characteristics

- Timestamps of received and transmitted frames
- Coarse and fine correction methods
- Trigger interrupt when system time is greater than target time
- Pulse output per second (product replacement function output)

33.2.3 CAN block diagram



33.3 Description of Ethernet functions: SMI, MII, and RMII

The Ethernet peripheral consists of MAC 802.3 (Media Access Control) and a dedicated DMA controller. Default Media Independent Interface (MII) and Simplified Media Independent Interface (RMII) are supported, both of which can be selected by a 1-bit control bit.

The DMA controller accesses the MAC controller and memory through the AHB master-slave interface. The AHB master interface controls the transfer of data, while the AHB slave interface is used for access control and status register (CSR) space.

Send FIFO (TX FIFO) caches data in system memory through DMA before sending by the MAC controller. Similarly, the receive FIFO (RX FIFO) stores Ethernet frames received from the line until they are transferred to system memory via DMA.

Ethernet peripherals also contain SMI interfaces to communicate with external PHY chips, and users can configure MAC and DMA controllers through a set of configuration registers to select desired operating modes and characteristics.

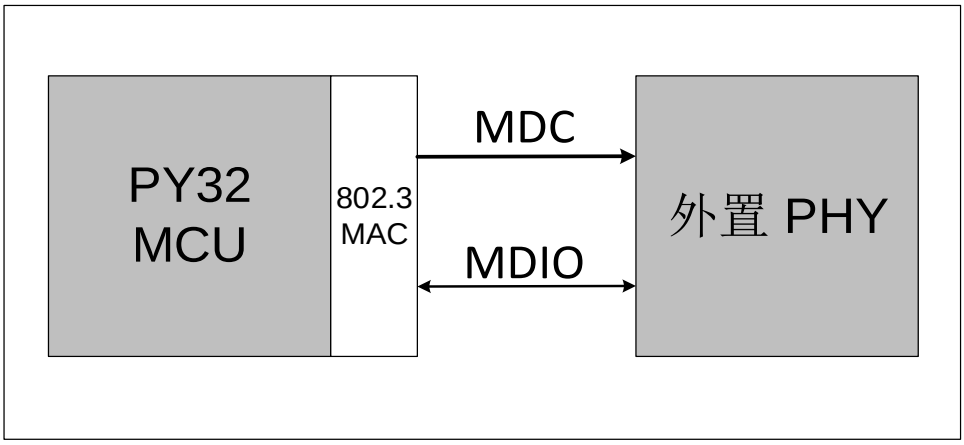
Note: When using Ethernet, the AHB clock frequency must be greater than or equal to 25Mhz

33.3.1 State Management Interface SMI

SMI allows access to PHY registers by implementing the application layer through a 2-wire clock and data line. This interface supports access to up to 32 PHYs. The application layer may select any one of the 32 PHYs and any one of the 32 registers in the PHY to transmit control data and read status signals. Only one register can be addressed in a PHY at any given time

Both the MDC clock line and the MDIO data line are implemented as I/O of the multiplexing function in the MCU

- MDC: A periodic clock provides a time base for data transmission, up to 2.5 MHz. Both the high level and the low level minimum are 160ns, and the period minimum is 400ns. In the idle state, the SMI management interface drives the MDC clock to low.
- MDIO: Data input and output lines, the MDC clock signal synchronously sends status information to or reads from the PHY



33.3.1.1 SMI frame formats

The frame format of read and write operations is as follows:

	帧内容							
	前导符(32 位)	起始符	操作符	PADDR	RADDR	TA	数据(16 位)	空闲位
读	1…1	01	10	ppppp	rrrrr	Z0	dddddddddddddddd	Z
写	1…1	01	01	ppppp	rrrrr	10	dddddddddddddddd	Z

The management frame consists of the following 8 parts:

● Preamble: Each transmission (read or write) must start with a preamble, which is 32 consecutive logical '1' signals on the MDIO line, and 32 clock signals on the corresponding MDC line. This part of the signal is used to establish synchronization with the PHY device.

● Starter: The starter of the frame is defined as' 01 ', that is, the MDIO line drops from logical' 1 'to' 0 'and back to' 1 'to mark the beginning of the transmission.

● Operator: Used to define the type of operation: read or write.

● PADDR: The address of PHY has 5 bits and can distinguish 32 PHYs. High bits are sent and received first.

● RADDR: The address of the register has 5 bits and can address 32 independent registers. High

bits are sent and received first.

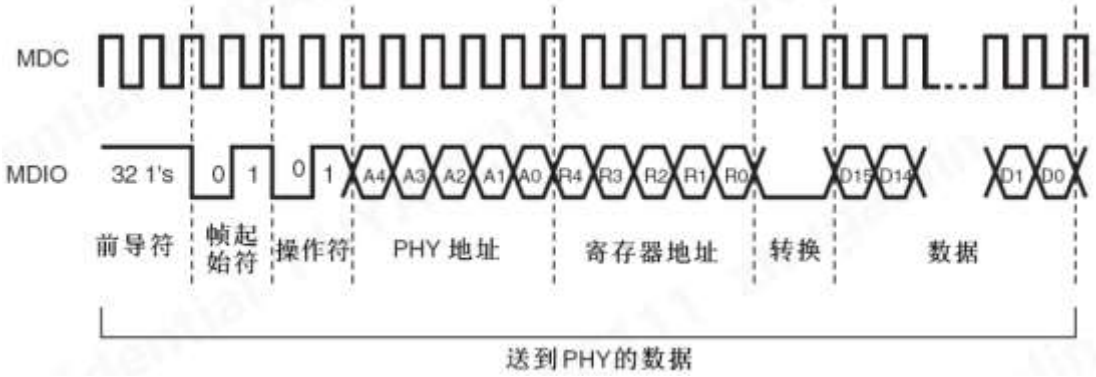
● TA: A 2-digit steering character inserted between RADDR and DATA (DATA) to avoid collisions during read operations. During the read operation, during the 2-bit time of TA, the MAC controller maintains the high impedance state of the MDIO line, while the PHY device maintains the high impedance state of 1 bit first, and outputs a '0' signal at the second bit. During the write operation, during this 2-bit time of TA, the MAC controller drives the MDIO line to output the '10' signal, while the PHY setting remains high impedance.

● DATA: 16 The data field of bits. The first to send and receive is the15thbitof the ETH_MIID register.

● Idle bit: The MDIO line remains in a high impedance state. All tri-state drives are cancelled, and the MDIO line is guaranteed to be at logic '1' by the PHY's pull-up resistor.

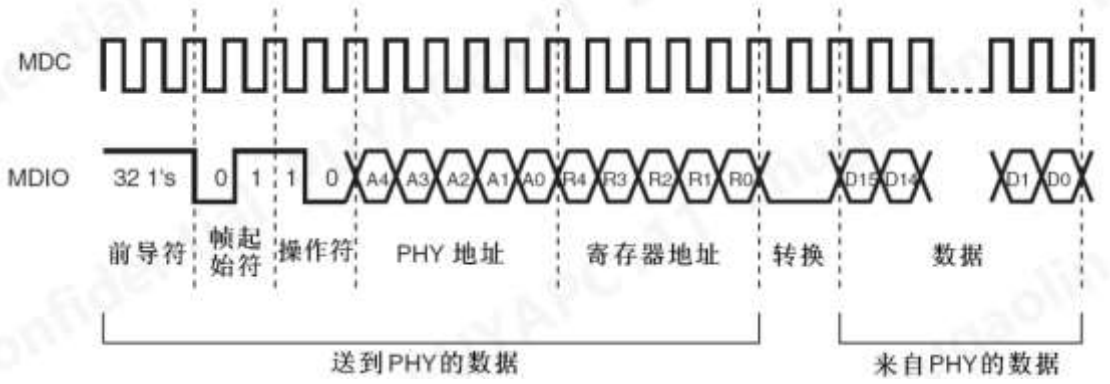
33.3.1.2 SMI write operation

When the application program setsthe MWandMB bits(ETH_MACMIIAR),SMIwill perform a write operation to the PHYregister throughthe address ofthe PHYand writethe data in the ETH_MACMIIDRregister. The MII address and data register cannot be modified during the operation. Even if they are modified, they will be ignored (the MB bit is always high), and no error will be reported after sending. When the write operation is finished, the SMI interface will clear the MB bits.



33.3.1.3 SMI Read Operation

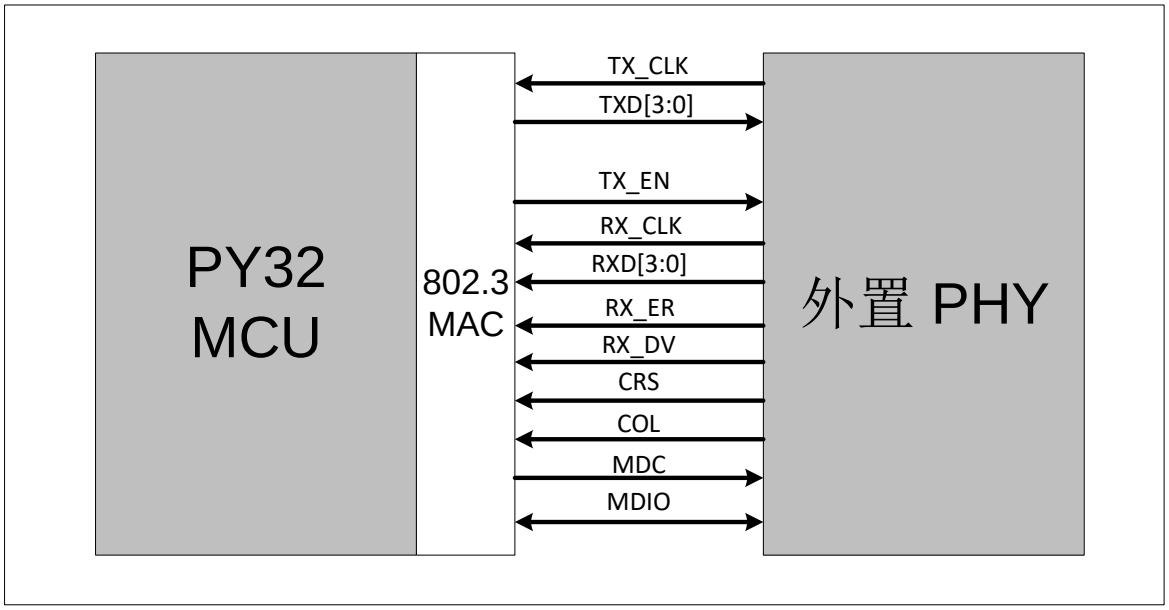
Whenthe MW bit ofthe ETH_MACMIIAR registeris0, the user setsthe MB bitand performs a read operation. The MII address and data register cannot be modified during the operation. Even if they are modified, they will be ignored (the MB bit is always high), and no error will be reported after sending. When the read operation is finished, the MB is pulled low and the MII data register is updated.



33.3.1.4 SMI clock selection

The clock of the SMI interface is divided by the application clock (AHB), and the selection of frequency division coefficient depends on the range of the clock. See the bit [5: 2]of the ETH_MACMIIARregister forthe specific frequency division coefficient.

33.3.2 Independent Media Interface MII



- MII_TX_CLK: The clock signal used to transmit data. The clock is 2.5 MHz for data transmission of 10 Mbit/
  s and 25 MHz for data transmission of 100 Mbit/s.
- MII_RX_CLK: The clock signal used to receive data, the clock is 2.5 MHz for 10 Mbit/s data transmission, and the clock is
  25 MHz for 100 Mbit/s data transmission.
- MII_TX_EN: Send an enable signal, which must be valid when the start bit of the data preamble appears, and needs to
  remain valid until the transmission is completed.
- MII_TXD [3: 0]: Transmit data line, transmit 4 bits of data at a time, and the data is valid when the MII_TX_EN signal is valid. When
  the MII_TX_EN signal is invalid, the PHY will ignore the transmitted data.
- MII_CRS: Carrier sense signal, operating only in half-duplex mode, controlled by PHY. This signal does not need to be synchronized with MII_TX_CLK
  and MII_RX_CLK. When it is in an active state, it means that the transmitting or receiving medium is not in an idle state.
  The MII_CRS signal remains active until both the transmission and reception medium are idle.
- MII_COL: Collision detection signal, operating only in half-duplex mode, controlled by PHY. This signal does not need to be synchronized with MII_TX_CLK
  and MII_RX_CLK. This signal is valid when a medium conflict is detected and remains
  valid for the entire duration of the conflict.
- MII_RXD [3: 0]: Receive data line, receives 4 bits of data each time, and the data is valid when the MII_RX_DV signal is valid. Depending on
  the state of the MII_RX_DV and MII_RX_ER signals, the MII_RXD [3: 0] data value may be used to convey some specific information
- MII_RX_DV: Receive data enable signal, controlled by the PHY, which is active when the PHY

prepares data for the MAC to receive.

- The signal must be valid when the first 4 bits of the data frame appear, and needs to remain valid until the transmission is complete. This signal must become invalid before the first clock after the last 4 bits of data are transferred. To ensure that the frame is received correctly,
  the MII_RX_DV signal should be valid until the SFD field appears.
- MII_RX_ER: Received error signal. To indicate that the MAC detected an error during reception, the MII_RX_ER signal must
  remain valid for one or more clock cycles (MII_RX_CLK). The specific error reason should be combined with the status of MII_RX_DV
  and the data value of MII_RXD [3: 0]. See the following table for details: Receiving interface signal encoding

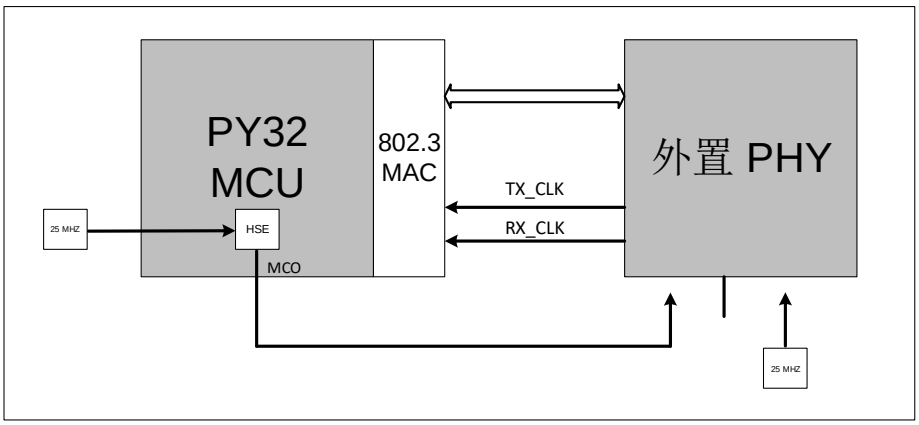
Table 33-1 transmit interface signal encoding

MII_TX_EN	MII_TXD [3: 0]	Description
0	0000 to 1111	Normal frame interval
1	0000 to 1111	Normal data transfer

Table 33-2 receive interface signal coding

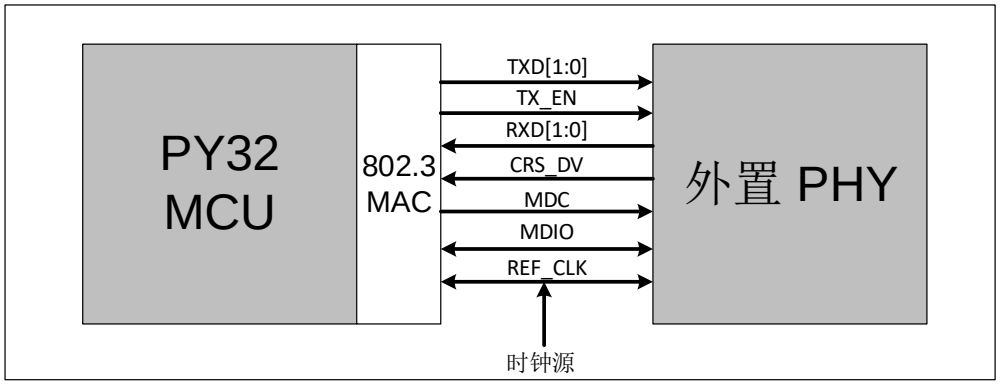
MII_RX_DV	MII_RX_ER	MII_RXD	Description
0	0	0000 to 1111	Normal frame interval
0	1	0000	Normal frame interval
0	1	0001 to 1101	Reserved
0	1	1110	Failed carrier indication
0	1	1111	Reserved
1	0	0000 to 1111	Normal data reception
1	1	0000 to 1111	Data reception error

33.3.2.1 MII clock scheme



The user needs to provide an external 25MHz clock to the external PHYto generate the TX_CLK and RX_CLK clock signals. This clock does not need to be the same as the MAC clock. This clock can be provided using an external 25 MHz crystal oscillator or the clock output pin MCO of the microcontroller. When the clock source is the MCO pin, it is necessary to configure a suitable PLL to ensure that the clock output by the MCO pin is 25 MHz.

33.3.3 Simplified Independent Media Interface RMII

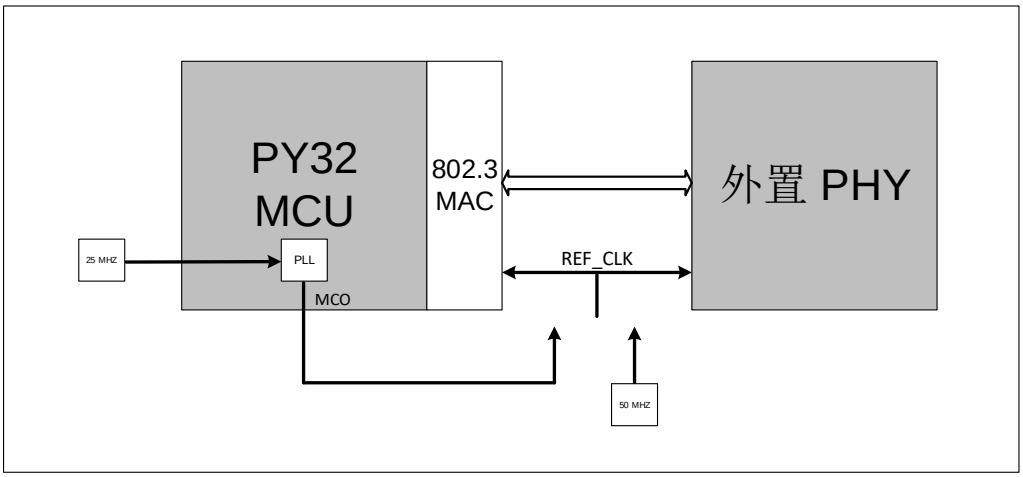


The Reduced Media Independent Interface (RMII) specification reduces the number of pins required for Ethernet communication. According to the IEEE 802.3 standard, the MII interface requires 16 pins for data and control signals, while the RMII standard reduces the number of pins to 7.

RMII features:

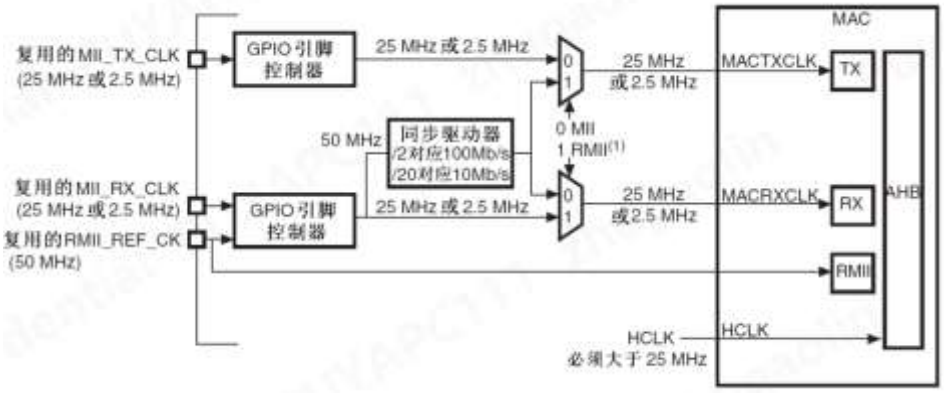
- Supports 10M and 100Mbit/s rates
- There is only one clock signal, and the clock signal needs to be raised to 50 MHz;
- MAC and external Ethernet PHY need to use the same clock source;
- Use 2-bit width data transmission and reception.

33.3.3.1 RMII clock scheme



Synchronization of both clock sources is ensured by connecting the same clock source to the REF_CLK pin of the MAC and Ethernet PHY. This clock can be provided via an external 50MHz signal or the MCO pin of the microcontroller. When the clock comes from the MCO pin, a suitable PLL needs to be configured to ensure that the clock output by the MCO pin is 50MHz.

33.3.4 MII/RMII selection



The selection of MII and RMII is through bit2 of the offset address 0x0000 register in SYSCFG.

33.4 Ethernet function description: MAC 802.3

The IEEE 802.3 Local Area Network (LAN) international standard adopts CSMA/CD (Carrier Sense Multiple Access with Collision Detection) as the access method.

The Ethernet peripheral consists of MAC802.3 controller, media independent interface and DMA controller

The MAC block implements the LAN CSMA/CD sublayer of the following systems: 10 Mbit/s and 100 Mbit/s data rates for baseband and width systems; Full-duplex and half-duplex working modes; The collision detection access mode is only used in half-duplex operation mode; Support MAC control frame sublayer

The MAC sublayer performs the following functions associated with the data link control procedure:

- Data encapsulation (sending and receiving)
  - Generate frames (frame boundary delimitation, frame synchronization)
  - Generate addresses (process source and destination addresses)
  - Error detection
- Media access management
  - Media allocation (collision avoidance)
  - Conflict Resolution (handling conflicts)

The MAC sublayer contains two modes of operation

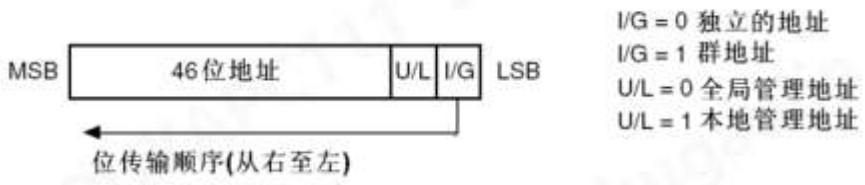
- Half-duplex mode: sites compete for the right to use physical media through CSMA/CD algorithm
- Full-duplex mode: can send and receive simultaneously without contention resolution when the following states are sent
  - Physical media capabilities to support simultaneous transmission and reception
  - 2 sites connected to LAN
  - 2 sites are configured in full duplex mode simultaneously

33.4.1 MAC 802.3 Frame Format

Two frame formats are defined through CSMA/CDMAC for data transmission:

- Basic MAC frame format
- Label MAC frame format (extension of base MAC frame format)

- Preamble: 7 bytes for synchronization (PLS circuit).
  Hexadecimal value: 55-55-55-55-55-55-55-55
  Binary: 01010101 01010101 01010101 01010101 01010101 01010101 01010101 (right-to-left transfer)
- Frame Header (SFD): 1 byte, used to indicate the frame head.
  Hexadecimal value: D5
  Binary: 11010101 (right-to-left transfer)
- Destination and Source Addresses: 6 bytes, indicating the destination and source addresses in the following way:
  - Each address is 48 bits.
  - The LSB bit (I/G) of the destination address is used to distinguish between individual addresses (I/G = 0) or group addresses (I/G = 1). A cluster address may point to 0, 1 or more, or all sites in the LAN. The first bit (LSB) bit in the source address is a reserved bit, fixed to '0'
  - The second bit (U/L) of the address is used to distinguish between a single (U/L = 1) address or a global (U/L = 0) address. For broadcast addresses, this bit is fixed to 1.
  - Each byte of the address bit must follow the low-order first transmission rule.
    Address assignments are of the following types:
- Individual address: This is the physical address that points to a specific site in the network.
- Group Address: This is the group address that points to one or more sites in a given network. There are two types of multiple addresses:
  - Multicast Group Address: An address pointing to a group of logically associated sites.
  - Broadcast address: It is a unique, pre-agreed multi-send address (destination address bits are all 1), which always points to all sites in a given LAN.



- QTag Prefix: A 4-byte identifier located between the source address domain and the MAC "Client Length/Type" domain. This identifier belongs to the tagged MAC frame and is an extension of the base frame (untagged). The base frame does not contain this identifier. This identifier includes:
  - The "length/type" field of 2 bytes as a constant, indicating the protocol type (greater than 0x0600), is equal to the value of the 802.1 Q tag protocol type (0x8100). This constant field is used to distinguish labeled and unlabeled MAC frames.
  - A 2 byte field, including control information for the tag. Contains the following parts: 3-bit user priority, 1-bit canonical format indication (CFI), and 12-bit VLAN identifier. The tagged MAC frame has 4 bytes more of the QTag prefix than the base frame.
- MAC client length/type: 2 bytes, which can be divided into two meanings (mutually exclusive) according to their different values:

- If the value of 2 bytes is equal to or less than maxValidFrame (1500), then these two bytes represent the number of data field MAC client data bytes (representing length) of the subsequent 802.3 frame.
- If the value of 2 bytes is equal to or greater than MinTypeValue (1536 or 0x0600), then these two bytes represent the MAC client protocol type (representative type) associated with the Ethernet frame.

  If the length of the data field is less than the minimum receivable length specified by the protocol, the data of the length/type field is ignored, and a padding (PAD) field is added after the data field and before the FCS (Frame Check Sequence) field. When data is transmitted and received in the length/type field, the high order comes first. The protocol does not define how the MAC sublayer handles length/type values between the maxValidLength and minTypeValue (range disjoint) values. Such data may or may not be received by the MAC sublayer.

- Data and padding fields: n bytes of data. The data fields are completely transparent, which means that any sequence of values can appear in the data segment. If there is a padding field, then the length of the padding is determined by the length of the data. The maximum and minimum lengths of the data and padding fields are as follows
  - The maximum length is 1500 bytes.
  - The minimum length of an unlabeled MAC frame is 46 bytes.
  - The minimum length of a labeled MAC frame is 42 bytes.

    If the length of the data is less than the minimum length required by the standard (42 bytes for labeled frames and 46 bytes for non-labeled frames), a padding field is added to meet the length required by the standard.
  - Frame check sequence: 4 bytes of cyclic redundancy check value (CRC). The CRC calculation includes numerical values for the following fields: source address, destination address, QTag prefix, length/type, LLC data, and padding (i.e. calculates all data except the preamble and SFD fields). Compute the polynomial as follows:

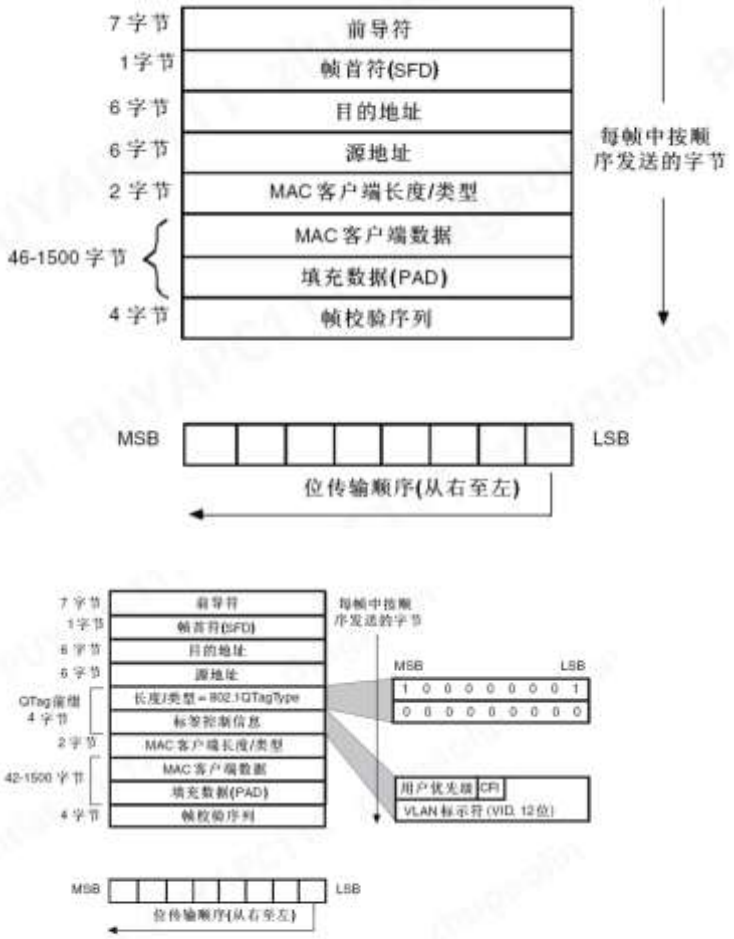
    G (x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1

    The CRC value of a frame is calculated as follows:

    The shortest length of an unlabeled MAC frame is 46 bytes.
  - The first two bits of the frame complement each other.
  - The n bits of the frame serve as coefficients of the polynomial M (x) of degree (n − 1). The first bit of the destination address is used as a coefficient of the xn-1 term, and the last bit of the data field

    is used as a coefficient of the x0 term.
  - The M (x) polynomial is multiplied by x32 and divided by G (x) to produce a remainder R (x) of degree ≤ 31.
  - The coefficients of the R (x) polynomial are taken as a 32-bit sequence.
  - The complement to this sequence is the CRC check value.
  - The frame check sequence transmits a 32-bit CRC check value. The coefficients of x32 are

transmitted first, and the coefficients of x0 are transmitted last.



Except for the FCS domain, the sequence of MAC frames is transmitted in low-bit first-out order. A MAC frame is

deemed invalid when:

- The frame length does not match the value defined in the length/type field. If the data of the length/type field is a type value, then the default frame length is consistent with the defined value (no invalid frames).
- The frame length is not an integer multiple of bytes (there are extra bits).
- The CRC value in the FCS domain does not match the CRC value calculated for the actual frame.

33.4.2 MAC frame transmission

DMA controls all transactions in the transmission path, by taking Ethernet frames out of system memory and storing them in FIFO. These frames are then sent to the MAC controller. When the EOF (end of frame) is transmitted, indicating the end of the transmission frame, the transmission status is obtained from the MAC controller and transmitted back to the DMA. The depth at which the FIFO is sent is 2 Kbyte. The fill level of the FIFO is indicated to the DMA so that it can use the AHB interface to extract the required burst data from the system memory.

When the SOF is detected, the MAC receives the data and sends it to the MII. The time at which the application layer turns on sending frame data to the MII interface is variable, depending on the delay

like IFG, the time at which the preamble/bounding code is transmitted, and the backoff (forced re-transmission after collision) delay in half-duplex mode. When the EOF is sent to the MAC controller, the controller will complete the regular transmission and then pass the transmission status to the DMA. If a regular collision (in half-duplex mode) occurs during the transmit phase, the MAC controller will make the transmit state valid, and then receive and discard the excess data until the next SOF is received. When a retry request from the MAC is observed, the same frame should be retransmitted from the SOF. If data is not continuously provided during transmission, the MAC issues an underflow state. During normal transmission of a frame, if the MAC receives the SOF without obtaining the EOF of the previous frame, the SOF is ignored and the new frame is considered a continuation of the previous frame.

There are two modes of operation for sending data to the MAC controller:

- Threshold mode: When the bytes in the FIFO exceed the configured threshold point (or write the EOF before the threshold is exceeded), the data is ready to be forwarded to the MAC controller. The threshold point may be configuredbythe TTC bitof the ETH_DMABMR.
- Store-and-forward mode: Only when a complete frame is stored after the FIFO, it will be forwarded to the MAC controller. If the depth of the TX FIFO is less than the length of the transmitted frame, the frame data will also begin to be forwarded to the MAC controller when the FIFO is nearly full

◆ Automatic CRC and byte padding generation

When the number of bytes received from the application layer is less than 60 (destination address + source address + length/type + data), 0 will be added after the data of the sent frame to ensure that the data length can reach 46 bytes to meet the minimum data length required by IEEE802.3. The MAC can also be configured without increasing the padding domain. The cyclic redundancy check (CRC) of the frame check sequence (FCS) field is calculated and appended to the data being transmitted. When the MAC is configured not to add a CRC value at the end of the frame, the calculated CRC is not transmitted. There is an exception to this rule that if the MAC is configured with additional padding bytes, the CRC will be added at the end of the frame.

- Send protocol

MAC controls the working process of sending Ethernet frames:

1. Generate preamble and delimiter
2. Generate blocking 01 sequence in half-duplex mode (jam pattern)
3. Control jabber timer
4. Control flow rate in half-duplex mode (back pressure)
5. Generate transmit frame status
6. Contains timestamp photo logic

When a new frame transmission is requested, the MAC transmits the preamble and delimiter followed by the data. The collision window is defined as 1 gap time (the time corresponding to 512bit), and the blocking mode is only used in the half-duplex mode.

In MII mode, if a collision is sent anywhere in a frame, the MAC sends a 32-bit blocking sequence (0x55555555) to the MII to inform all other stations that a collision has occurred. If the collision occurs

during the transmission of the preamble, the MAC will send the blocking sequence after completing the transmission of the preamble and the delimiter.

The jabber timer is used to cut off the sending process when more than 2048 bytes are sent (default). In half-duplex mode, MAC uses a delay mechanism for traffic control. When the application layer requests to stop receiving frames, as long as traffic control is enabled, it will send a 32-byte blocking sequence when the MAC senses the reception of the frame. This causes a conflict and the remote workstation will fall back. The application requests flow control through the BPA bitofthe ETH_MACFCR register. If the application requests to send a frame, it is scheduled and transmitted even when the activation is pressed. Note that if the backpressure remains active for a long period of time (and more than 16 consecutive collision events occur), the remote station aborts its transmission due to excessive collision. If IEEE 1588 timestamp is enabled for the transmit frame, this block takes a snapshot of system time when the delimiter is placed on the transmit MII bus

- Send scheduler

The MAC is responsible for scheduling frame transmission on the MII interface. It maintains the gap between two transmitted frames and follows a truncated binary exponential backoff algorithm in half-duplex mode. The MAC starts transmitting after satisfying the IFG and the backoff delay. It remains between two arbitrary transmit frames and configures an idle period (IFG bit in the ETH-MACCR register). If the frame arrives within the IFG time, the MII waits for an enable signal from the MAC before starting transmitting the frame. Once the carrier signal of the MII becomes inactive, the MAC starts its IFG counter. When the value of IFG is configured, the MAC enables transmission in full duplex mode. In the half-duplex mode, when the IFG is configured to correspond to 96 bits, the MAC follows the rules of IEEE 802.3 specification, section 4.2. 3.2. 1. If a carrier is detected in the first two-thirds of the IFG interval, the MAC resets the IFG counter. If it is detected in the last third, the MAC will let the IG counter continue to count and enable the transmitter after the IFG interval time has elapsed. MAC implements a truncated binary exponential backoff algorithm in half-duplex mode.

- Send flow control

When the transmit flow control enable (TFE bitofETH_MACFCR) bit is set to1, in full duplex mode,the MACgenerates a pause frame and transmits it. The calculated CRC is appended to the Pause frame and then sent. Pause frame generation can be initiated in two ways: 1.Whenthe FCB positionofETH_MACFCRis1 2.Whenthe receive FIFOis full

— If the application layer requests flow controlby placingthe FCB position1ofETH_MACFCR, the MACgenerates and transmits a single pause frame. The value of the pause time in the pause frame is configured by the ETH_MACFCR register. If the time set in the previously transmitted pause frame is to be delayed or ended, the application layer needs to request the transmission of another pause frame aftersetting a new pause time value (PTofERH_MACFCR).

— If the receiving FIFO is full, the application layer requests flow control, and the MAC will also generate and send a pause frame. If the receive FIFO remains full for the configurable slot time before the pause time is expired, a second pause frame is transmitted. The process repeats as long as the FIFO remains full. If this state is no longer satisfied, the MAC will send a pause frame with zero pause time to indicate

to the remote end that the receiving buffer is ready to receive new data.

■ Single packet sending operation

Regular transmission sequence:

— If the system has data to send, the DMA controller takes them out of memory through the AHB main interface and starts forwarding them to the FIFO. The FIFO continues to receive data until a complete frame is received.

— When the threshold is exceeded or a complete packet is received by the FIFO, the frame data is sent to the MAC controller. The DMA continuously sends data from the FIFO to the MAC. When a frame is transmitted, the DMA controller will receive a status notification from the MAC.

■ Two packet sending operations

— Because the DMA must update the descriptor status before releasing the descriptor to the host, it is possible that up to two frames are sent to the FIFO. Only when the OSF position is 1, the DMA will take out the second frame and put it into the FIFO. If OSF is not set to 1, the next frame is fetched from memory only after the MAC has completely processed one frame and the DMA has released the descriptor

— If OSF position 1, the DMA starts fetching the second frame immediately after completing the transmission of the first frame to the FIFO. It doesn't wait for an update of the status. At the same time, when the first frame is transmitted, the second frame is received by the FIFO. Once the first frame is sent and the MAC returns the status information, it is pushed to the DMA. If the DMA has completed sending the second data packet to the FIFO at this time, the second frame data transmission must wait for the state of the first frame transmission.

■ Retransmission Mechanism in Conflict

In half-duplex mode, when a frame is sent to the MAC, the MAC interface may generate a collision event. The MAC gives a state indicating retransmission even before receiving the end of the frame. Then the retransmission is enabled and the frame is ejected from the FIFO. When more than 96 bytes are forwarded to the MAC controller, the FIFO controller frees up that space and makes it available for DMA to push more data. This means that after transmitting more than 96 bytes or when the MAC controller indicates a delay collision event, it cannot be retransmitted.

■ Send FIFO empty operation

The software flush send FIFO can be performed using bit 20 of the operating mode register. The empty operation is an immediate behavior, even if the Tx FIFO is sending a frame to the MAC controller, it will be affected and the corresponding pointer is reset to the initial position. This causes an underflow event to occur in the MAC transmitter and terminates transmitting the current frame, whose status information will include the underflow and the occurrence of a frame empty event (bit13 and bit1 of TDES0). During the emptying process, DMA will not send data to FIFO. A send status word containing the number of emptied frames is passed to the application. A completely cleared frame will have a frame clearance status position 1 (TDES0). When the application program (DMA) receives all the status words of the clear frame, the clear operation is completed. Send FIFO Clear Control Register Clear.

From this point on, a new frame issued by the application layer is received. After the purge operation, all in-transit data will be discarded unless it starts with SOF.

■ Send Status Word

At the end of the frame sent to the MAC controller and when the controller finishes sending the frame, the sending status is returned to the application. The detailed description of the state is consistent with bit [17: 0] of TDES0. If the IEEE 1588 timestamp is enabled, both the transmit status and the 64-bit timestamp are also returned.

■ Send checksum load shedding

Communication protocols such as TCP and UDP have checksum fields, which can determine the integrity of data transmitted through the network. Because the most widespread use of Ethernet is to encapsulate TCP and UDP through IP fields. Ethernet controller has the feature of sending checksum load shedding, which is used to support checksum calculation and insert it into sending, and can detect errors when receiving. This section describes the checksum load shedding characteristics of the transmitted frame

Note: TCP, UDP, and ICMP checksums are calculated based on the complete frame, and then the calculation results are inserted into the corresponding header area. Based on this characteristic requirement, this function is valid only when the transmit FIFOis configured in a store-and-forward mode (the TSF bit ofthe ETH_DMAOMR register is1). If configured in threshold mode, the transmit checksum load shedding is bypassed.

It is necessary to ensure that the depth of the FIFO can store a complete frame. If the FIFO depth is not sufficient, the checksum insertion function of the payload (TCP/UDP/ICMP) will be bypassed. Even in the store-and-forward mode, only the IPv4 header checksum of the frame can be modified.

The transmit checksum load shedding supports two types of checksum calculation and insertion, and the checksum per frame can be controlled by setting the CIC bit (bit28: 27 of the transmit descriptor TDES1)

For IPv4, TCP, UDP, ICMP, IPv6, and ICMPv6 packet header specifications, refer to IETF specifications RFC 791, RFC 793, RFC 768, RFC 792, RFC 2460, and RFC 4443, respectively

— IP Header Checksum

In IPv4 fields, the integrity of the header field is represented by the 16-bit header checksum field (eleventh and twelfth bytes of the IPv4 datagram). When the value of the Ethernet frame type is 0x0800 and the IP field version is 0x4, the checksum load shedding detects the IPv4 field. The checksum field of the input frame is ignored during the calculation and replaced by the calculated value. The IPv6 header does not have a checksum field, so checksum load shedding does not modify the IPv6 header field. The result of the IP header checksum calculation is indicated by the IP header error status bit (bit16) in the transmit state, which will be set when the values of the Ethernet frame type field and the IP header version field are inconsistent, or when the Ethernet frame does not have enough data (displayed by the IP header length field). In other words, this bit will be set to 1 when an IP header error is asserted

1. IPv4 Field

a) The received Ethernet type is 0x0800, but the IP header version field is not equal to 0x4
b) The IPv4 header length field displays a value less than 0x5 (20 bytes)
c) The total length of the frame is less than the value of the IPv4 header length field

2. IPv6 Field
   a) Ethernet type is 0x86dd, but IP header version field is not equal to 0x6
   b) The frame ends before the IPv6 header (40 bytes) and the extended header (indicated by the corresponding header length field in the extended header) are fully received. If the Ethernet Type field shows a payload for IPv4, it inserts the IPv4 header checksum even if the checksum load shedding detects an IP header error.

— TCP/UDP/ICMP checksum

The TCP/UDP/ICMP checksum processes IPv4 and IPv6 headers (including extended headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP

Note:

1. For non-TCP/UDP/ICMP/ICMP v6 payloads, the checksum is bypassed and not modified in the frame
2. Segmented IP frames (IPv4 or IPv6), IP frames with security features such as authentication headers or encapsulated security payloads, and IPv6 frames with routing headers are bypassed and not processed by checksums.

The checksum calculates the TCP/UDP/ICMP payload and inserts it into the corresponding field in the header. There are two modes of operation:

1. First mode: TCP, UDP, or ICMP v6 false headers are not included in the checksum calculation and are assumed to be present in the checksum field of the input frame. The checksum field is included in the checksum calculation and is replaced by the final checksum calculation value.
2. The second mode: the checksum field is ignored, TCP, UDP or ICMP v6 false header data is included in the checksum calculation, and the checksum field is overwritten by the final calculated value.

Note that for packets for ICMP-over-IPv4, the checksum field in the ICMP packet must always be equal to 0x0000 in both modes, because there are no fake headers defined in such packets. If it is not equal to 0x0000, an incorrect checksum may be inserted into the packet.

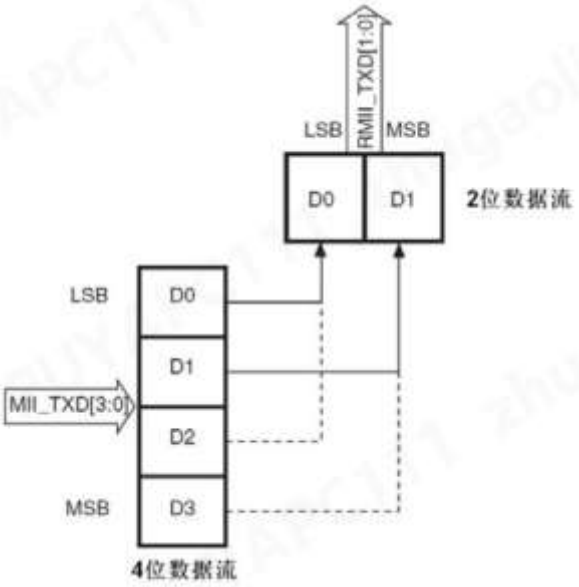
The checksum result is shown by the payload checksum error status bit, at bit 12 of the transmit state vector. Set when sent in any of the following cases:

1. In store-and-forward mode, the frame has begun to be forwarded to the MAC transmitter and the end of the frame is not written to the FIFO
2. In the received IP header, the packet has ended before the byte length corresponding to the payload length field.

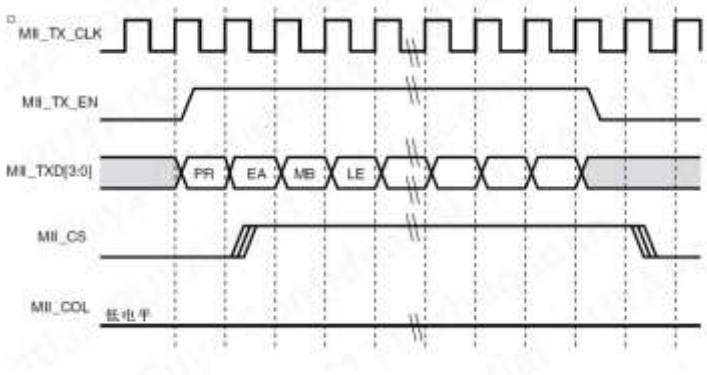
When the packet exceeds the byte length corresponding to the payload length field, the byte is ignored as a padding byte, and no error is reported. When the first type of error is detected, the header is not modified. For the second error type, the result of the checksum calculation is still inserted into the corresponding area of the header.

■ MII/RMII send bit sequence

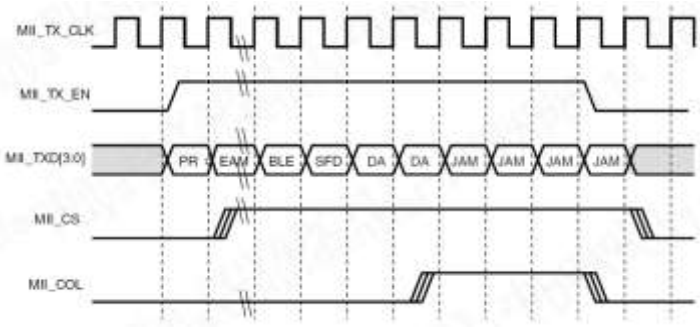
Send low bits first and then high bits



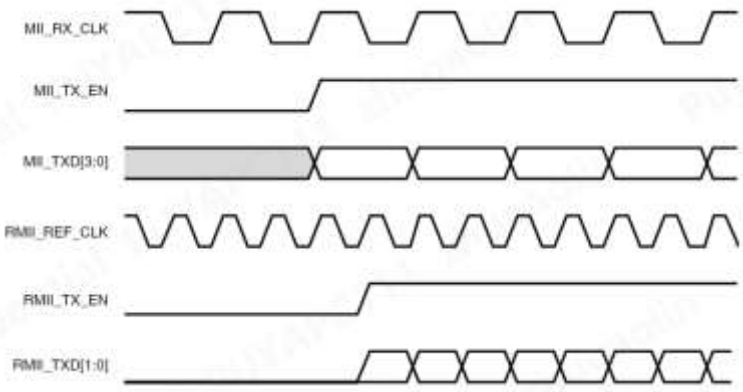
Transfer when there is no conflict:



Transfer when there is a conflict:



Frame transmission in MII and RMII modes:



33.4.3 MAC frame reception

The MACreceives the frame and then pushes it intothe Rx FIFO. Oncethe FIFOexceeds the configured reception threshold (RTC bitofthe ETH_DMAOMRregister), it will issuea status (padding level) to the DMA. So that the DMA can initiate a preconfigured burst transmission to the AHB interface.

In the default pass-through mode, when64bytes (the RTC bitofthe ETH_DMAOMR register) or a complete packet of data isreceived by the FIFO, the data is ejected andthe DMAis notified of its availability. When the DMA has initialized AHB interface transmission, the data will be continuously transmitted from the FIFO until a complete packet is sent. Once the EOF frame has been sent, the status word is ejected and sent to the DMA controller.

Inthe Rx FIFO store-and-forward mode (the RSF bit ofthe ETH_DMAOMR register), the frame can only be read out after it has been fully written into the receivingFIFO. In this mode, all error frames are discarded (if configured), and only valid frames can be read out and forwarded to the application. In the pass-through mode, part of the error frame is not discarded, because the error state is received at the end of the frame, when the beginning of the frame has been read by the FIFO.

When the MAC detects a delimiter on the MII interface, the receive operation is initialized. The MAC controller strips the preamble and delimiter before processing the frame. The header field is checked for filtering and its FCS field is CRC checked. If address filtering fails, the frame is discarded by the controller.

◆ Receiving protocol

Both the preamble and delimiter of the received frame will be stripped. Once the delimiter is detected, the MAC starts sending an Ethernet frame to the Rx FIFO, starting with the byte after the delimiter (destination address). If the IEEE 1588 timestamp is enabled, a photo of the system time is taken when the delimiter is detected. Unless the MAC filters and discards the frame, the timestamp is sent to the application.

If the length/type field of the received frame is less than 0x600, and if the MAC is configured for automatic CRC/PAD. The MAC sends the data of the frame to the RxFIFO until the count specified in the Length/Type field and then starts dropping bytes (including the delimiter field). If the length/type

field is greater than or equal to 0x600, the NAC sends all received frame data to the Rx FIFO, regardless of whether the automatic CRC function is configured. MAC watchdog is enabled by default, and all parts of frame data greater than 2048 bytes (destination address + source address + length/type + data + padding bytes + frame check sequence) are cut off. This feature is turned off when the watchdog function is turned off. Then even if the watchdog function is turned off, frames larger than 16KB in size will be cut off and a watchdog timeout state will be given.

◆ Receiving CRC: Autofill and CRC stripping

The received frame is checked for CRC errors by the MAC. It calculates the 32-bit CRC, ranging from the destination address to the field portion of the frame check sequence. Encoding Algorithm:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

The MAC performs a CRC check calculation on the received full frame, regardless of whether autofill and CRC are stripped.

◆ Receive checksum load shedding

In Ethernet receive frames, both IPv4 and IPv6 frames are detected and processed for data integrity. The reception checksum load sheddingmay be enabled by setting the IPCO bitof the ETH_MACCR register. The MAC receiver identifies IPv4 and IPv6 frames by determining whether the value of the frame type field is equal to 0x0800 and 0x86DD. The method can also be used to identify frames with VLAN tags. Received checksum load shedding calculates the IPv4 header and checks if it matches the received IPv4 header checksum. IP header errors are set if the payload type (Ethernet type field) and IP header version do not match, or if the received frame does not have enough bytes, indicated by the length field of the IPv4 header (or when there are fewer than 20 bytes available in the IPv4 or IPv6 header). Received checksum shedding is also used to identify and calculate the checksums of TCP, UDP, and ICMP payloads in IP data segments. It includes TCP/UDP/ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The match passes the payload checksum error bit in the receive status word
Indicate it. The error bit is also set if the TCP/UDP/ICMP payload length does not match the expected payload length given by the IP header. As mentioned in the TCP/UDP/ICMP verification chapter, receiving checksum load shedding bypasses the payloads of segmented IP datagrams, IP datagrams with security features, IPv6 routing headers, and payloads other than TCP, UDP, or ICMP. This information is reflected in the receive status, as detailed in the RDES0 (Receive Descriptor Word 0) section. In this configuration, the controller does not append any payload checksum bytes to the received Ethernet frame.

Bit 18: Ethernet frame	Bit 27: Header checksum error	Bit 28: Payload checksum error	Frame status
0	0	0	The frame is an IEEE 802.3 frame (Length field value is less than 0x0600).
1	0	0	IPv4/IPv6 Type frame in which no checksum error is detected.
1	0	1	IPv4/IPv6 Type frame in which a payload checksum error (as described for PCE) is detected
1	1	0	IPv4/IPv6 Type frame in which IP header checksum error (as described for IPCO HCE) is detected.
1	1	1	IPv4/IPv6 Type frame in which both PCE and IPCO HCE are detected.
0	0	1	IPv4/IPv6 Type frame in which there is no IP HCE and the payload check is bypassed due to unsupported payload.
0	1	1	Type frame which is neither IPv4 or IPv6 (checksum offload bypasses the checksum check completely)
0	1	0	Reserved

◆ Receive frame controller

If the RA bit in the MAC CSR frame filter register is reset, the MAC performs frame filtering based on the destination and source addresses (if the application decides not to receive any bad frames (e.g. runt, CRC error frames, etc.), it still needs to perform another level of filtering.). When a filtering failure is detected, the frame is discarded or sent to the application. When the filter address register is dynamically changed, if the filter fails (DA-SA), the rest of the frame is discarded, and the Rx status word is immediately updated (zero frame length, CRC error, and short error position bits) to show the filter failure. In Ethernet Low Power mode, all frames are discarded and are not forwarded to the application.

◆ Receiving flow control

The MAC detects the reception of the pause frame and pauses the transmission of the frame by the delay time specified in the pause frame (full-duplex mode only). The pause frame detection function may be enabled by the RFCE bitinthe ETH_MACFCR register. When receive flow control is enabled, the destination address of the received frame will always be monitored whether it matches the multicast address of the control frame (0x0180 C200 0001).If a match is detected (the destination address of the received frame matches the destination address of the reserved control frame),MACwill selectwhether to send thereceived control frame to the application program through the PCF bitofETH_MACFFR.

The MAC will also decode the type of the received control frame, the opcode, the pause timer time, if the byte counter of the state shows 64 bytes, and if there is no CRC error, the MAC transmitter will pause any data transmission within the decoded pause time (pause value multiplied by slot time, 1 slot time = transmission time corresponding to 64 bytes). Meanwhile, if another pause frame is detected and it is a zero pause time, the MAC resets the pause time and manages this new pause request. If the type of the received control frame is not 0x8808, the opcode is not 0x00001, and the byte length is not 64 bytes, the MAC does not generate a pause even if there is no CRC error.

If a pause frame is sent over a multicast destination address, MAC filters the frame by address matching.

If the pause frame is sent over the unicast destination address,MACwill choose to filter the frame based on whether the destination address matchesthe MAC address0registerandwhetherthe UPDFof ETH_MACFCRis set to1(detecting the pause frame even if it is transmitted over the unicast destination address). The PCF register bit(bit[7: 6]in ETH_MACFFR) controls filtering of the control frame in addition to address filtering.

◆ Receive Operation Multiframe Processing

Because the status is indicated immediately after the data is sent, the FIFO can store any number of frames as long as it is not full.

◆ Error handling

Ifthe Rx FIFOisfull before the end-of-frame data is received fromthe MAC, an overflow occurs and the entire frame is dropped, and the overflow counter in the ETH_DMAMFBOCR registeris accumulated. Due to overflow, the status indicates that this is a partial frame. If the FEFandFUGF bitsofETH_DMAOMRare enabled,the RX FIFOcan filter errors and small size frames.

If the receiving FIFO is configured in bit store and forward mode, all error frames are filtered and discarded.

In the pass-through mode, if the state and length of the frame are available when the SOF of the frame is read from the Rx FIFO, the entire error frame can be discarded. DMA can clear erroneous frames read from FIFO by enabling the clear bit of received frames. The data sent to the application is stopped and the rest of the frame is internally read and the MAC interrupt is discarded. You can then start that next frame transmission (if available)

◆ Receive Status Word

At the end of Ethernet frame reception, the MAC outputs the reception status to the application program (DMA). See RDES0 for detailed description of reception status

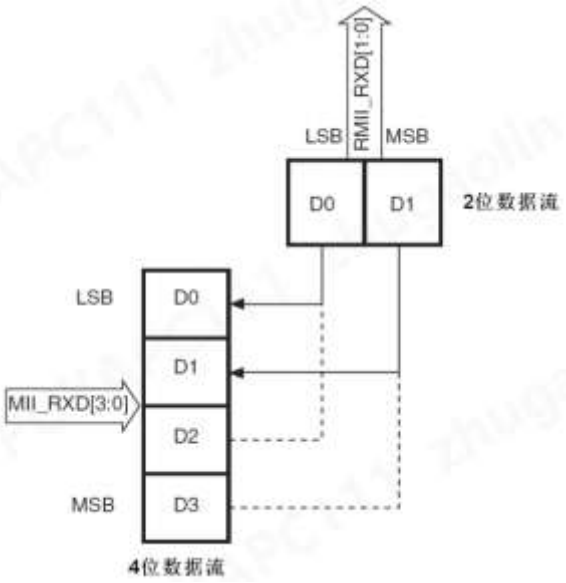
◆ Frame length interface

In the case of a switch application, data transmission and reception between the application and the MAC occurs as a complete frame transmission. The application layer should know the length of the frame received from the ingress port in order to transmit the frame to the egress port. The MAC controller places the frame length information in the state at the end of each frame reception.

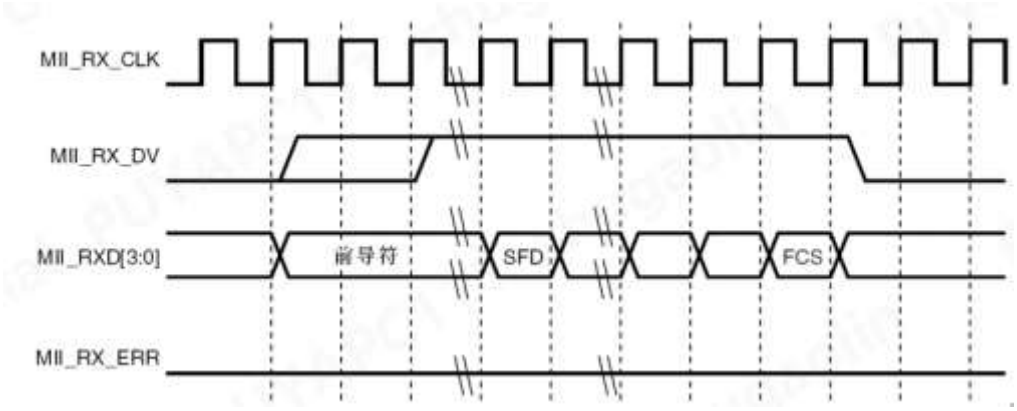
Note: The frame length value of part of the frames written to the Rx FIFO is 0 due to overflow

◆ MII/RMII receive bit sequence

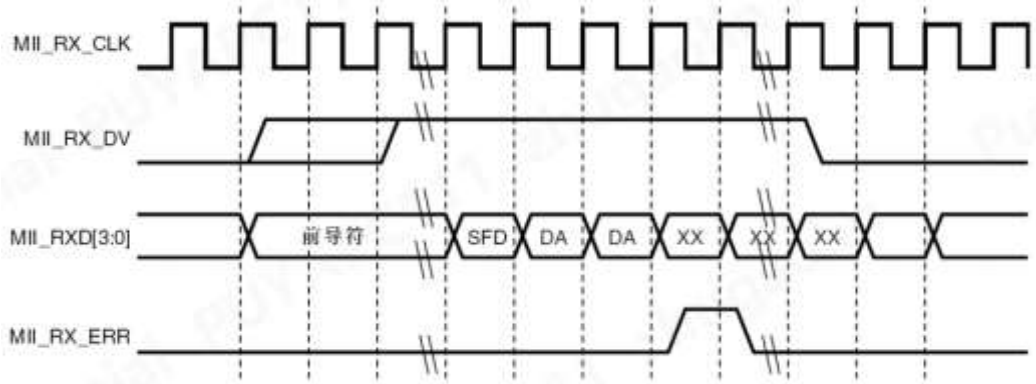
Receive the low bit first, then receive the high bit



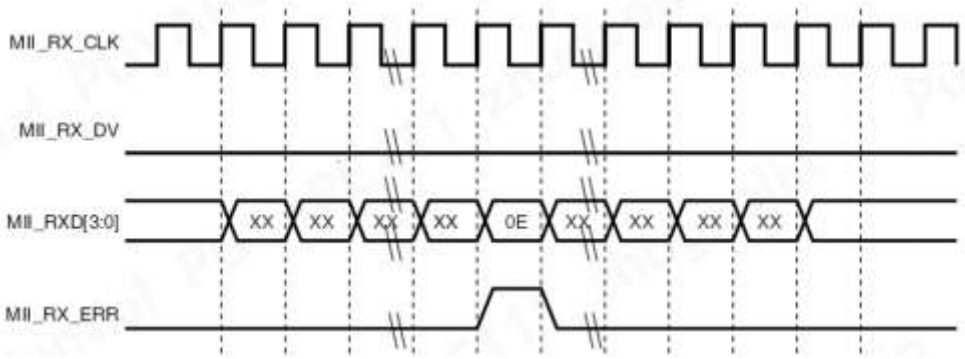
Received without error:



There is an error when receiving:



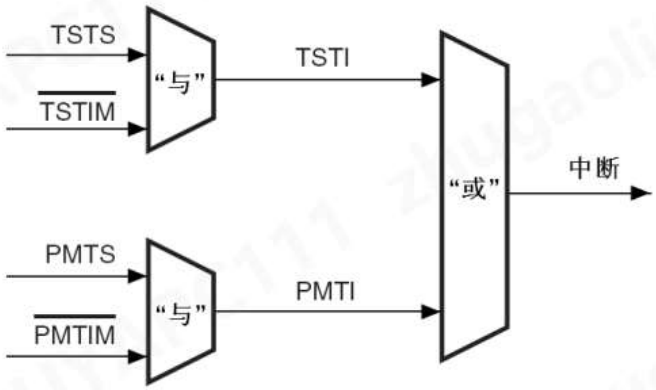
There is a false carrier on reception:



33.4.4 MAC interrupts

Due to various events, the MAC core may generate interruptions. The ETH_MACSRregister describes whena MAC interruptis caused. Each event can be prevented from asserting an interrupt by setting a corresponding mask bit in the interrupt mask register.

The interrupt register bit only indicates the set of event reports. The corresponding status register and other registers need to be read to clear the interrupt. For example, bit3 of the interrupt register is set to 1, indicating that a magic packet and a wake-up frame are received in the low power mode. The ETH_MACPMTCSRregister needs to be read to clear the interrupt event



33.4.5 MAC filtering

◆ Address filter

Address filtering checks destination and source addresses in all received frames and reports address filtering status. Address checking is based on different address parameters (frame filter registers) selected by the application layer. Filtered frames can also be identified: multicast or broadcast frames.

Address filtering performs address checking through a site's physical address and a multicast hash table

◆ Unicast destination address filtering

MAC supports unicast perfect filtering for up to 4 MAC addresses, and if perfect filtering is selected (the HU bit of the frame filter register is reset), MAC compares the 48-bit received unicast address to the configured MAC address for a match. The default MAC Addr0 is enabled, and the other addresses MAC Addr1-3 can be controlled by one enable bit, respectively. Each byte of MAC Addr1-3 can be masked for comparison when the destination address is compared, and only the corresponding byte

mask control bit needs to be set. This facilitates group address filtering of destination addresses. In hash filtering mode (HU set), MAC uses a 64-bit hash table to perform imperfect filtering of unicast addresses. For hash filtering, the MAC indexes the contents of the hash table using the upper CRC of 6 bits of the received destination address (see Note 1 below). bit0 in the select hash table register with value 000000 and bit63 in the select hash table register with value 111111. If the corresponding bit (determined by the 6-bit CRC) is set to 1, it means that the unicast frame has passed the hash filtering, otherwise it fails.

Note: CRC is an algorithm for 32-bit values

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

◆ Multicast destination address filtering

The MAC can pass all multicast frames by configuring the PAM bit in the frame filter register. If the PAM is not set, the MAC will filter the multicast address based on the HM bit. In perfect filtering mode, the address of the multicast frame matches the configured MAC destination address 1-3 register. Group address filtering is also supported. In hash filtering mode, the MAC uses a 64-bit hash table for imperfect filtering. For hash filtering, the MAC indexes the contents of the hash table using the upper CRC of 6 bits of the received destination address (see Note 1 below). bit0 in the select hash table register with value 000000 and bit63 in the select hash table register with value 111111. If the corresponding bit (determined by the 6-bit CRC) is set to 1, it means that the unicast frame has passed the hash filtering, otherwise it fails.

◆ Hash or perfect address filtering

By setting the HPF in the frame filter register, the frame can be considered to have passed when the destination address passes hash filtering or perfect filtering. If HPF = 0, then one type of filtering is supported. Set the corresponding HU and HM bits to realize which filtering mode to select in the filtering of unicast and multicast frames.

◆ Broadcast address filtering

The MAC does not filter broadcast addresses by default, however, if the MAC is configured to refuse to receive broadcast frames through the BFD bit, then all broadcast frames are dropped.

◆ Unicast source address filtering

The MAC also perfectly filters the source address of the received frame. By default, the MAC compares the values of the source address and the source address register. The MAC address registers 1-3 may be configured for source address comparison instead of destination address comparison, which is selected by bit 30 of each register. Group filtering of source addresses is also supported. When SAF position 1 in the frame filtering register, the received frame is discarded if it fails the source address filtering. Otherwise, the result of the source address filtering results in a status bit placed in the receive status word (refer to receive descriptor word 0, RDES0)

When SAD is set to 1, the filtering results of the source address and destination address will be combined with the logic, and then it is decided whether to perform frame forwarding. This means that as

soon as one fails, the frame is discarded. If both addresses are successfully filtered, the frame will be forwarded to the application.

◆ Reverse filtration operation

Filtering for destination and source addresses. One option is to reverse the filtered match result and then output it. This is done by configuring the DAIF and SAIF bits of the frame filter register. The DAIF bit is suitable for the destination address filtering reverse of unicast or multicast frames. SAIF is suitable for source address filtering reverse for unicast frames.

The following figure shows a summary of source address and destination address filtering when different register configurations are configured.

Frame type	PM	HPF	HU	DAIF	HM	PAM	DB	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all frames
	0	X	0	0	X	X	X	Pass on perfect/group filter match
	0	X	0	1	X	X	X	Fail on perfect/Group filter match
	0	0	1	0	X	X	X	Pass on hash filter match
	0	0	1	1	X	X	X	Fail on hash filter match
	0	1	1	0	X	X	X	Pass on hash or perfect/Group filter match
	0	1	1	1	X	X	X	Fail on hash or perfect/Group filter match
Multicast	1	X	X	X	X	X	X	Pass all frames
	X	X	X	X	X	1	X	Pass all frames
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop PAUSE control frames if PCF = 0x
	0	0	X	0	1	0	X	Pass on hash filter match and drop PAUSE control frames if PCF = 0x
	0	1	X	0	1	0	X	Pass on hash or perfect/Group filter match and drop PAUSE control frames if PCF = 0x
	0	X	X	1	0	0	X	Fail on perfect/Group filter match and drop PAUSE control frames if PCF = 0x
	0	0	X	1	1	0	X	Fail on hash filter match and drop PAUSE control frames if PCF = 0x
	0	1	X	1	1	0	X	Fail on hash or perfect/Group filter match and drop PAUSE control frames if PCF = 0x

Frame type	PM	SAIF	SAF	SA filter operation
Unicast	1	X	X	Pass all frames
	0	0	0	Pass status on perfect/Group filter match but do not drop frames that fail
	0	1	0	Fail status on perfect/group filter match but do not drop frame
	0	0	1	Pass on perfect/group filter match and drop frames that fail
	0	1	1	Fail on perfect/group filter match and drop frames that fail

33.4.6 MAC Loopback Mode

The MAC supports a loopback mode that sends frames to its own receiver. This mode is not enabled by default and can be enabled by the loopback control bit ofthe ETH_MACCR register.

33.4.7 MAC Management Counter: MMC

The MAC Management Counter (MMC) contains a set of registers to collect statistical information about received and transmitted frames. Includes control registers for controlling register behavior, two 32-bit registers containing generated interrupts, and two 32-bit registers containing corresponding interrupt masks. These registers are accessible through the application layer and are 32 bits wide

When the frame passes the address filtering, the receiving MMC counter will be updated, and the discarded frame statistics will not be updated unless the discarded frame is a short frame less than 6 bytes (DA bytes are not fully received)

◆ Good transmit and receive frames

The transmitted frame is considered good if the transmitter transmits successfully, in other words, if the transmission of the frame is not aborted because of the following errors, it is considered good.

- JABBER timer overflow
- No carrier/carrier loss
- Delay collision
- Short frame
- Excessive delay
- Excessive conflict

The received frame may be considered a good frame if the following error is not transmitted

- CRC error
- Short frames (less than 64 bytes)
- Alignment error (only at 10/100 Mbit, frame is not a full byte at the end)
- Wrong length (atypical frames only)
- Out of range (atypical frames only, over maximum width)
- MII_RXERinput error

The maximum frame width depends on the frame type:

- Non-labeled frame max size = 1518
- Label frame max size = 1522

33.4.8 Power Management: PMT

This section describes the PMT, a power management mechanism supported by MAC. PMT supports receiving wake-up frames and magic packet frames of remote networks. The PMT generates an interrupt when receiving both frames. When the remote wake-up frame is enabled and the magic packet is enabled, the PMT moduleis enabled with the enable control bits beingWFEandMPEinthe ETH_MACPMTCSRregister. When entering the low power mode, all received frames are discarded by the MAC and will not be forwarded to the application. Exit low power mode when a wake-up frame or magic packet of the remote network is detected.

◆ Remote wake-up frame filter register

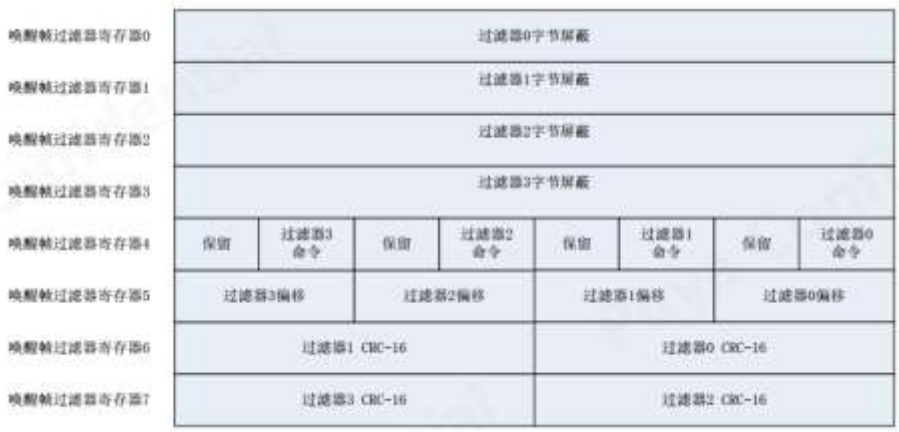
There are eight wake-up frame filter registers, but these registers share the same offset address. When reading or writing to a filter register is completed

, the internal pointer will automatically point to the next filter register. Whether it is a read or write

operation,

it needs to be performed 8 consecutive operations. That is to say, when setting it, the set values need to be divided into 8 times and written to the wake-up frame

filter register address one by one. When reading, it also needs to be read 8 times in a row to read all values.



■ Filter n-byte mask

This register defines which bytes of the frame are used by filter n (n = 0, 1, 2, 3) to check whether it is a wake-up frame. Its

31st bit must be '0' and bit [30: 0] is a byte mask bit. If the m-th bit (m = 0-30) of filter n (n = 0, 1, 2, 3) is' 1 ',

the CRC module that wakes up frame detection processes the [filter n offset + m] byte of the input frame, otherwise ignores it.

■ Filter n command

A total of 4 bits control the working mode of filter n. The highest bit 3 is the address type selection, and if this bit is' 1 ', only multicast frames are detected;

If the bit is' 0 ', only the unicast frame is detected. Bits 2 and 1 must remain 0. Bit 0 is the enable bit of filter n. When set, filter n is enabled, otherwise filter n is disabled.

■ Filter n-offset

Works with filter n-byte mask. This register defines the offset within the frame of the first byte to be checked by filter n. The minimum allowable value

is 12, which represents the 13th byte of the frame (an offset value of 0 represents the 1st byte of the frame).

■ Filter n CRC-16

This register contains a pre-written CRC-16 code for comparison with the CRC-16 value calculated from the frame data.

◆ Remote wake-up frame filtering detection

When the MAC is in a deep sleep state and the remote wake-up bit of the ETH_MACPMTCSR register is enabled, normal operation resumes after receiving the remote wake-up frame. The application will

write 8 wake-up filter registers, by writing the wake-up frame filter register address 8 times in succession. The application enables the remote wake-up function by writing bit2 of the ETH_MACPMTCSR register to 1. PMT supports four programmable filters, which provide different patterns of received frame sequences. If the received frame passes address filtering, and if the CRC-16 is consistent with the incoming detection sequence, then the wake-up frame will be received. The filter offset (minimum value 12, representing the 13th byte of the frame) determines the offset of the detected frame. The filtering byte mask determines which byte of the frame must be detected. The 31st bit of the byte mask must be 0. Wake up frames will only check for their length errors, FCS errors, bit loss errors, MII errors, collisions, and need to make sure it is not a short frame. Even if the wake-up frame exceeds 512 bytes in length, the frame is considered valid if it has a valid CRC value. Upon reception of each remote wake-up frame, the detection result of the wake-up frame is updated in the ETH_MACPMTCSR register. If enabled, an interrupt of the PMT is generated to indicate that a remote wake-up frame has been received.

◆ Magic Package Detection

Another wake-up method is to detect Magic Packet wake-up frames (see AMD's "Magic Packet Technology"). The MAC receives a special packet called a magic packet that addresses network nodes. Only magic packets addressed to a device or broadcast address are checked to determine if they meet the wake-up requirement. Magic packets that pass address filtering (unicast or broadcast) are checked to determine if they conform to the 6-byte remote wake-up LAN data format of all bytes, and then 16 occurrences of the MAC address. Detection is enabled by writing 1 to bit1 of ETH_MACPMTCSR. The PMT module continuously monitors a frame for a magic packet sequence, checking the 0xFFFF FFFF FFFF sequence of each received frame after the destination and source address fields. The PMT block then checks the frame for 16 repetitions of the MAC address without any interruption or interruption. If an interrupt occurs in 16 repetitions of the address, the 0xFFFF FFFF FFFF sequence is scanned again in the incoming frame. The 16 repetitions can be anywhere in the frame, but must be before the synchronization stream. The device will also receive multicast frames whenever 16 repetitions of the MAC address are detected. For example, if the node address of the MAC is 0x0011 2233 4455, then the MAC will next detect the following data sequence:

Destination Address Source Address … … … … ….. FFFF FFFF0011

2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455

0011 2233 44550011 2233 4455 0011 2233 4455 0011 2233 4455

0011 2233 4455 0011 2233 44550011 2233 4455 0011 2233 4455 0011

2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233

… CRC

When a magic packet is received, the detection result of the magic packet is updated in ETH_MACPMTSCR, and if enabled, a PMT interrupt is generated to indicate that the magic packet has been received.

■ System considerations during low power consumption

When the MCU is in deep sleep mode, if the external interrupt line 19 is enabled, the PMT module of Ethernet can still detect the frame.

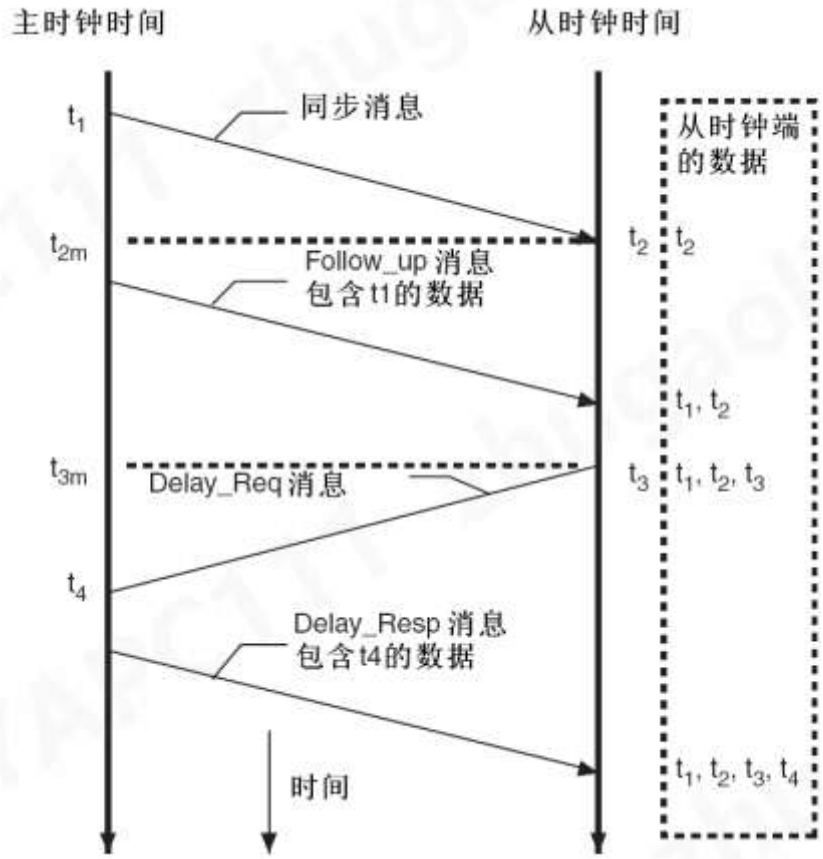
The state machine of the MAC receiverneeds to remain enabled at all times in low power mode, which means thatthe RE bitofthe ETH_MACCR registermust always be set because it involvesthe detection of magic packets/wake-up frames. However, during low power mode, the transmit state machineshould be turned off by clearing the TE bitin the ETH_MACCR register. And the DMA also needs to be turned off because there is no need to send the data of the magic packet/wake-up frame to the SRAM. In order to turn DMA off, the ST bitsandSR bitsinthe ETH_DMAOMR registerneed to bereset

The recommended steps to enter low power and wake up are as follows:

1. Turn offthe transmission DMAand wait for the complete transmission of the previous frame, which can bejudged by bit [0]ofETH_DMASR;
2. Turning offthe MAC transmitterand receiverby clearing the REandTE bitsinthe ETH_MACCRregister;
3. Waiting for the receive DMA to read out all frames in the Rx FIFO, and then turning off the receive DMA;
4. Configuration enables the external interrupt line19to generate an event or an interrupt. If the external interrupt line19is configured to generate an interrupt, it is also necessary to writean interrupt handler ETH_WKUP_IRQto clearthe interrupt flag bitof the external interrupt line 19;
5. Settingthe MFE/WFE bitinthe ETH_MACPMTCSRenablesdetection of the magic packet/wake-up frame;
6. Settingthe PD bitinETH_MACPMTCSRenables entry intothe MAC low power mode;
7. Settingthe RE bitin ETH_MACCRenablesthe MAC receiver;
8. Enter system STOP mode:
9. When a valid wake-up frame is received, the Ethernet peripheral exits the low power mode;
10. ReadETH_MACPMTCSRto clear the power management event criteria, enablethe MACsend state machine, and enable receive and sendDMA
11. Configure System Clock: Enable HSE and set the clock

33.4.9 Precise Time Protocol (IEEE1588 PTP)

The IEEE 1588 standard defines a protocol that allows accurate clock synchronization to be used in measurement and system control when using technologies such as network communications, local computing, and distributed objects. The protocol is suitable for systems that communicate over a local area network that supports multicast information, including (but not limited to) Ethernet. The protocol is used to synchronize heterogeneous systems, including clocks with different intrinsic accuracy, resolution, and stability. This protocol supports system-wide synchronization accuracy in the sub-microsecond range, with minimal network and local clock calculation data. Such message-based protocols are called time precision protocols. Systems and networks are classified as master and slave nodes for distributing timing/clock information. The technique of synchronizing master and slave nodes by exchanging PTP information is as follows:



1. The master node broadcasts the PTP synchronization message to all nodes. The synchronization message contains the time information of the master device. When the message leaves the master system, the time is t1. In the Ethernet port, this time can be captured by MII.
2. Receiving the synchronization message from the device will also capture the precise time, t2, through its time reference
3. The master device then sends a Follow_up message, which contains t1 information for later use
4. The slave device sends the Delay_Req information to the master device, and records the exact time t3 when the frame leaves the MII
5. When the information enters the system, the master receives the message and captures the exact time t4
6. The master device sends a Delay_Resp message containing t4 information to the slave device
7. The slave device synchronizes its local time reference to the time reference of the master device by four time values t1, t2, t3 and t4

The protocol is mostly implemented in software above the UDP layer, and then, as described above, hardware support is required to capture the exact time of a particular PTP packet as it enters and leaves the MII interface. This time information must be captured and returned to the software in order to implement PTP correctly and with high precision

33.4.9.1 Reference clock source

To get a snapshot of time, the controller requires a reference time in 64-bit format defined in the IEEE 1588 specification (divided into 2 32-bit channels, with the upper 32 bits providing time in seconds and the lower 32 bits indicating time in ns). The PTP reference clock input is used to internally generate

a reference time (also referred to as system time) and to capture the timestamp. The frequency of the reference clock must be greater than or equal to the resolution of the timestamp counter. The synchronization accuracy between master node and slave node is targeted to be 100ns

The generation, update and modification of system time are described in the next section

The accuracy depends on the input period of the PTP reference clock, the characteristics of the OSC, and the frequency of the synchronization process.

Due to the synchronization from the Tx and Rx clock input domains to the PTP reference clock domain, the uncertainty value of the timestamp is latched to 1 reference clock cycle. If we count the uncertainty value of the resolution, we will add half the cycle time of the timestamp

33.4.9.2 Transmit frame with PTP characteristic

When the MII interface outputs the SFD of the frame, a timestamp is captured. The frames that need to be captured timestamps are controllable. In other words, each transmit frame may be marked to indicate whether a timestamp necessarily needs to be captured. The transmit frame is not processed to identify the PTP frame. Control of the frame is performed by transmitting control bits in the descriptor. The captured timestamp will be returned to the application in the same way as the frame served state. The timestamp is sent back into the transmit descriptor of the response along with the transmit status of the frame, thereby automatically connecting the timestamp into a particular PTP frame. The 64-bit timestamp information is written back into TDES2 and TDES3, where TDES2 holds the low 32-bit significant bits of the timestamp

33.4.9.3 Receive frame with PTP characteristic

When the IEEE 1588 timestamp feature is enabled, the Ethernet MAC captures the timestamps of all frames received from the MII. When the frame is received, the MAC provides a timestamp. The captured timestamp will be returned to the application in the same way as the frame served state. The timestamp is sent back into the reception descriptor of the response along with the reception status of the frame. The 64-bit timestamp information is written back into RDES2 and RDES3, where RDES2 holds the low 32-bit significant bits of the timestamp

33.4.9.4 System time correction method

The 64-bit PTP time is updated by the PTP input reference clock (HCLK). The PTP time is used as a photographed (timestamp) source for MII receive and transmit frames. The system time count may be initialized or corrected by coarse or fine calibration

In the coarse correction method, the initialization value or the offset value is written to the timestamp update register. For initialization, the value of the system time counter is written in the timestamp update register. For system time correction, the offset value is added or subtracted from the system time.

In the fine correction method, the frequency drift of the slave device clock (reference clock) from the master device is corrected for a period of time. Unlike the coarse correction method, this correction method is not done in a single cycle. The longer correction time helps to maintain linear time and does not bring drastic jitter (or large jitter) to the reference time between PTP synchronization information intervals. In this method, an accumulator will accumulate the values in the ADDend register as shown

in the figure below, and the arithmetic result generated by the accumulator will be used as the increment pulse of the system time counter. The accumulator and ADDend are both 32-bit registers. Here, the accumulator acts as a high-precision frequency multiplier and divider.

The system time update logic requires a clock frequency of 50Mhz to achieve an accuracy of 20ns, and the frequency division is the ratio of the required clocks in the reference clock domain. Therefore, if the reference clock (HCLK) is 66 Mhz, then the ratio is 66 Mhz/50 Mhz = 1.32. Therefore. The default addend value should be set to 2 ^ 32/1.32 = 0xC1F0 7C1

If the reference clock offset is smaller, if 65 Mhz, then the ratio 65/50 = 1.3, and the value in the add register is set to 2 ^ 32/1.3 = 0xC 4EC 4EC 4. If the clock offset is larger, if it is 67Mhz, the value in the add register should be set to 0xBF0 B7672. If the offset is 0, the default value of add should still be set to 0xC1F0 7C1 (2 ^ 32/1. 32)

In the figure above, the constant value used to increment the Subsecond register is 0d43. This gives the system time an accuracy of 20ns (in other words, it is incremented in 20ns steps).

The software needs to calculate the offset of the frequency by synchronizing the message and then update the Addend register. Initially, set the slave device clock to a frequency uncompensated value in Addend, which value = 2 ^ 32/frequency division

If the master-to-slave delay is assumed to be the same for successive synchronization messages, then the algorithm described below needs to be applied. After several synchronization cycles, the frequency values are locked. The slave device clock can determine the delay value of a master device to a slave device and use the new value to resynchronize with the master device.

The algorithm is as follows:

- At MasterSyncTime, the master device sends a synchronization message to the slave device. When the message is received from the device, whether the local clock or SlaveClockTime is calculated MasterClockTime = MasterSyncTime (n) + MasterToSlaveDelay (n)
- Master device clock counts current synchronization period, MasterClockCount = MasterClockTime (n)-MasterClockTime (n-1) (assuming MasterToSlaveDelay) is the same as synchronization periods n and n-1)

- Count the current synchronization period from the device , SlaveClockCount (n) = SlaveClock-Time (n)-SlaveClockTime (n-1)
- The difference between the master and slave clock count values for the current synchronization cycle is ClockDiffCount (n) = MasterClockCount (n)-SlaveClockCount (n)
- Frequency scale factor of slave device clock FreqScaleFactor (n) = (MasterClockCount (n) + ClockDiffCount (n))/SlaveClockCount (n)
- Then the frequency compensation value of the Addend register FreqCompensationValue (n) = FreqScaleFactor (n) × FreqCompensationValue (n-1)

Theoretically, the algorithm realizes locking in a synchronization cycle; However, it may be delayed by several cycles due to network propagation delays and variations in operating conditions.

The algorithm is self-tuning: if the slave device clock is initially set to an incorrect value compared to the master device, the algorithm will correct it at the expense of more synchronization cycles.

33.4.9.5 Programming steps to initialize the generation system time

The timestamp feature can be enabled by configuringbit0of the ETH_PTPTSCRregister. After setting this bit, the timestamp counter needs to be initialized. Suitable steps are as follows:

1. Set bit9 of the MACIMR register to mask timestamp-triggered interrupts;
2. Enable timestamp;
3. Calculating the value based on the clock frequency of the PTP, and then configuring the sub-second accumulation register;
4. If using the fine correction method, configure the addend register of the timestamp and configure bit5 of the timestamp control register (addend register update);
5. Polling the timestamp control register until bit5 is cleared;
6. Setting the bit1 selection fine correction method of the timestamp control register;
7. Set the bit2 of the timestamp control register (timestamp time initialization);
8. The timestamp counter starts working after initialization
9. Enable MAC receivers and transmitters

Note: When thebit0ofETH_PTPTSCRis cleared to close the timestamp operation, the above operation needs to be restarted again.

33.4.9.6 Updating step of system time in coarse adjustment correction method

1. Write offset values (positive or negative) in timestamp update high and low registers
2. Set timestamp control register bit3 (TSSTU)
3. When the TSSTU bit is cleared, the value in the timestamp update register is added or subtracted from the system time

33.4.9.7 Update step of system time in fine adjustment correction method

1. Using the algorithm introduced in the aforementioned "System Time Calibration Method", the value of the addender register corresponding to the desired system clock frequency is calculated;
2. Update Timestamp
3. Waiting for the desired new value in the add register to take effect, may be by triggering a timestamp interrupt when the system reaches the target value;

4. Configure the required time in the target time high register and low registerto unmask the timestamp interrupt by clearing bit9in the ETH _MACIMR register;
5. Set the bit4 of the timestamp control register (TSARU);
6. When an interrupt is triggered, readthe ETH_MACSRregister;
7. Reprogram the timestamp addender register with the old value and setbit5for ETH_PTSCRagain

33.4.9.8 PTP Trigger Internal Connection to TIM2

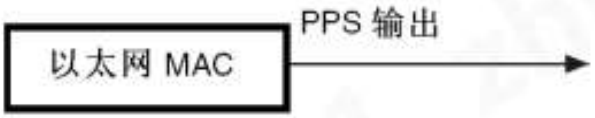
When the system time is faster than the target time, the MAC provides a trigger interrupt. Using interrupts introduces known delays and uncertainty in command execution times.

To avoid this uncertainty, the PTP outputs a trigger signal when the system time is faster than the target time. On the input trigger that is internally connected to TIM2. The analysis is carried out by the input capture characteristics of the timer, the output comparison characteristics and the waveform of the timer. If the timer (PCLK1: TIM2 APB1 clock) and the reference clock of the PTP are synchronized, there is no uncertain time left.

The PTP trigger signalmay be software connected totheITR1input of TIM2, enabled byregister controlinbit29 SYSCFGofthe AFIO_MAPR register

33.4.9.9 PTP PPS Output signal

The PTP pulse output is used to check the synchronization of different nodes in the network. In order to test the difference between the local slave device clock and the master reference clock, the PPS (second pulse) output of the master and slave devices can be connected to an oscilloscope to measure the difference between the two clocks. The enabling of the PPS output is controlled by a register in the SYSCFG

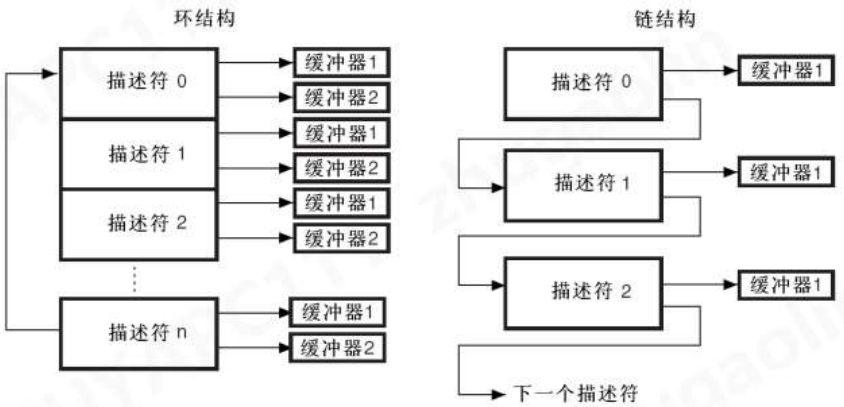


33.5 DMA Controller

DMA has separate send and receive engines, as well as CSR register space. The sending engine sends data from system memory to the Tx FIFO, and the receiving engine sends data from the Rx FIFO to system memory. The controller utilizes descriptors to efficiently move data from the source to the destination with minimal CPU intervention. DMA is designed for packet-oriented data transmission, such as frames in Ethernet. The controller may be programmed to implement interrupting the CPU when frame transmission and reception is complete and in the event of other normal/error conditions. The DMA communicates with the MCU controller through two data structures

- Control and Status Register (CSR)
- Descriptor List and Data Cache

The DMA sends the received frame data to the reception buffer in the system memory, and can also send the data in the reception buffer. The descriptors of system memory are stored in the form of pointers to the cache. There are two descriptor lists: one for receiving and one for sending. The base address of each list is written in the ETH_DMATDLAR register and the ETH_DMARDLAR register. The descriptor list is forward-linked (implicitly or explicitly). The last descriptor may point to the first entry to create a ring structure. The explicit linking of the descriptors is achieved by configuring the second address linked in the receive and send descriptors (RDES1 [24] and TDES1 [24]). The descriptor list is located in the physical memory space of the system. Each descriptor can point to a maximum of two cache areas. This allows two physically addressed caches to be used instead of two consecutive caches in memory. The data buffer area is located in the physical memory space of the system and consists of whole frames or partial frames, but cannot exceed a single frame. The cache contains only data. The status of the cache area is displayed in the descriptor. However, a single descriptor cannot span multiple frames, and when the end of a frame is detected, the DMA jumps to the next frame buffer. The data link may be enabled or disabled. The descriptor structure of ring and chain is as follows:



33.5.1 Initialization procedure using DMA transfer

The MAC initialization process is as follows:

1. Configuringparametersof system access to the ETH_DMABMR register;
2. Configuring the ETH_ DMAIERregister to mask necessary interrupts;
3. The software driver creates a list of send and receive descriptors, and then writes the start address of each list intothe ETH_DMARDLARandETH_DMATDLARregisters to provide tothe DMA;
4. Configuring the MAC registers 1, 2, 3 to select a desired filtering attribute;
5. Configurethe MAC ETH_MACCRregister to enable the receive and send working modes, andset the PSandDM bitsaccording to the automatic negotiation result (readfrom PHY);
6. Writebit13andbit1ofthe ETH_DMAOMR registerto1, and start transmitting and receiving;
7. The state machine of the sending and receiving engine starts to run. First, it tries to get descriptors from various descriptor lists, and then starts sending and receiving operations. The transmitting and receiving processes are independent of each other and can be started or stopped separately

33.5.2 Host Burst Access

Ifthe FB bitofETH_DMABMR isconfigured,the DMAattemptsto perform a fixed-length burst transmission at the AHB host interface. The maximum burst length is determined by the PBL. For the 16 bytes

to be read, the receive and send descriptors are always accessed at the largest possible burst size (subject to PBL restrictions).

Only when there is enough space in the transmitting FIFO for the configured burst or number of bytes, the transmitting DMA will initiate data transmission until the end of the frame (when it is less than the configured burst length). The DMA indicates the start address and the number of transmissions required for the AHB master interface. When the AHB interface is configured with a bit fixed length burst, it is transmitted in an optimal combination of INCR4, INCR8, INCR16 and SINGLE. Otherwise (no fixed length burst), INCR and SINGLE are used to transmit data.

The receive DMA will initiate data transmission only if there is enough data in the receive FIFO for the configured burst, or when the end of a frame is detected in the receive FIFO (when it is less than the configured burst length). The DMA indicates the start address and the number of transmissions required for the AHB master interface. When the AHB interface is configured with a bit fixed length burst, it is transmitted in an optimal combination of INCR4, INCR8, INCR16 and SINGLE. Otherwise (no fixed length burst), INCR and SINGLE are used to transmit data. When the AHB interface is configured for address alignment beats, both DMA engines ensure that the first burst transmission initiated by the AHB is less than or equal to the size of the configured PBL. Therefore, all subsequent beats start at addresses aligned with the configured PBL. DMA can only align the address of the beat to size 16 (for PBL > 16) because the AHB interface does not support more than INCR16.

33.5.3 Host data buffer alignment

The send and receive data caches do not have any restrictions on start address alignment. In 32-bit system memory, the start address of the cache can be any one of 4 bytes. However, DMA always initiates transmissions with address alignment, without the bus width of byte channels with dummy data required. This usually occurs during transmission at the beginning or end of an Ethernet frame.

- Example of read cache area:

If the address of the send cache is 0x0000 0FF2 and there are 15 bytes to transfer, then the DMA reads the full 5 words of data (20 bytes) starting at address 0x0000 0FF0, but when sending data to the Tx FIFO, the extra bytes (the first 2 bytes) are dropped. In the same way, the last 3 bytes will also be discarded. The DMA needs to ensure that it is 32-bit data that is transferred to the Tx FIFO, unless it is data at the end of the frame.

- Example of write cache area:

If the address of the receive cache is 0x0000 0FF2, and 16 bytes of data need to be received, then the DMA will write 5 full 32-bit data starting from the address 0x0000 0FF0 address. But the first 2 bytes of the first transmission and the last 2 bytes of the third transmission will have dummy data.

33.5.4 Buffer area size calculation

DMA does not update the size fields in the send and receive descriptors. The DMA will only update its status field. The drive needs the area to calculate the size. The transmission DMA transmits the exact number of bytes to the MAC core (indicated by the buffer size field in TDES1). If the FS bit in the descriptor TDES1 is set to 1, the DMA considers the first transmission from the buffer as the beginning of the frame. If the LS bit in the descriptor TDES1 is set to 1, the DMA treats this transmission from the buffer as the end of the frame. The receiving DMA will continue to send data until the buffer area

is full or a frame of data is sent out. If the descriptor is not marked as last (LS bit in RDES0), but when FS position 1, the buffer corresponding to the descriptor is full and the valid data in the buffer is equal to the value of the size field minus the offset pointed to by the buffer pointer. When the data buffer pointer is aligned with the data bit width, the offset is 0. If the descriptor is marked as last, the buffer may not be full (as shown by the buffer size in RDES1). To calculate the effective amount of data in this final buffer, the driver must read the frame length (the FL bit in RDES0 [29: 16]) and subtract the sum of the buffer sizes of the previous buffer areas in that frame. Typically the receiving DMA will send the start of the next frame at the time of a new description.

Note: Even when the start address of the receive buffer is misaligned with the system data bit width, the system should allocate a receive buffer area with the same size as the system bus width. For example: If the system allocates a 1024-byte receive buffer area with a starting address of 0x1000, the software can program an offset with the starting address of 0x1002 in the receive descriptor. The receiving DMA will supplement dummy data (0x1000 and 0x1001) at the first 2 positions when writing frame data to the buffer area. The actual frame is written from 0x1002. Therefore, due to the offset of the start address, the real effective space in the buffer is 1022 bytes, even though the buffer size is configured to be 1024 bytes.

33.5.5 DMA Arbitration

The arbiter inside the DMA is responsible for arbitrating between transmit and receive channel access to the AHB main interface. Two types of quorum methods are supported: round-round and fixed-priority. When the cyclic mode is selected (the DA bitofthe ETH_DMABMR registeris0), the data bus is proportionally allocated by settingthe PM bit arbiterwhen the transmitting and receivingDMArequest access at the same time. When the DA bit is 1, the receive DMA always takes precedence over the transmit DMA for data access.

33.5.6 DMA error status

Forany data transfer initiated by the DMA channel, if the slave device replies with an error response,the DMAstops all operations and updatesthe error bit and fatal bus error bit in the status register (ETH_DMASRregister). The DMA controller can only resume operation after a soft or hard reset of the peripheral and re-initialize the DMA.

33.5.7 Tx DMA Configuration of

33.5.7.1 Non-OSF mode (default mode)

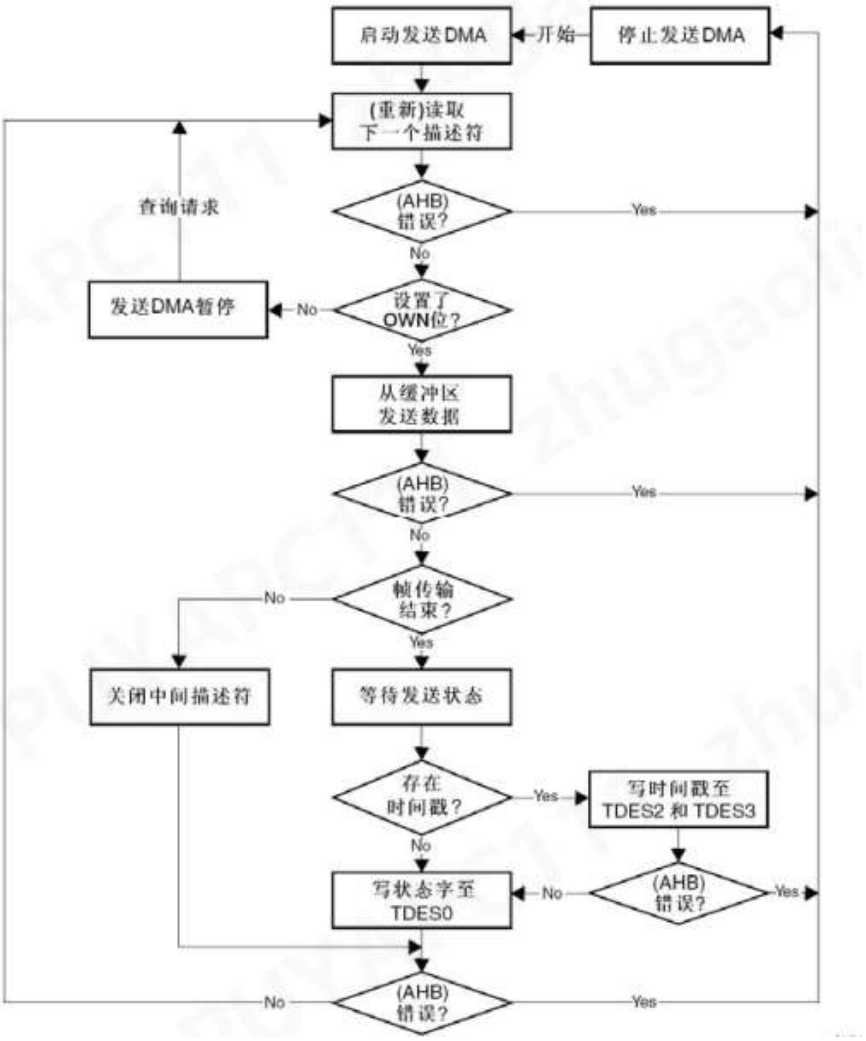
Default configuration for sending DMA engine:

Set the transmission descriptor (TDES0-TDES3), and set the OWN bit in TDES0 after establishing the corresponding data buffer area with Ethernet frame data;

1. Whenthe ST positionofthe ETH_DMAOMR registeris1,the DMAenters the running state;
2. When entering the running state, the DMA polls the transmit descriptor list. When the polling is complete, it continues in sequence descriptor ring order or chain order;
3. Ifthe DMAdetects a descriptor marked asCPUowned, or an error state occurs, the send buffer (bit2ofthe ETH_DMASR register) becomes unavailable and the regular interrupt statistics bit is set (bit16ofthe ETH_DMASR register), and the send engine flow jumps to step9;

4. If the acquired descriptor is marked as owned by DMA (TDES0 [31]) is set to 1), DMA will decode the transmit data buffer address from the acquired descriptor;
5. The DMA fetches the transmission data from the memory and transmits the data;
6. If an Ethernet frame is stored on a data buffer in multiple descriptors, DMA closes the intermediate descriptor and fetches the next descriptor. Steps 3, 4, and 5 are repeated until the end of the frame is transmitted;
7. When the frame is transmitted, if the IEEE 1588 timestamp function is enabled, the value of the timestamp is written to the transmit descriptors (TDES2 and TDES3) containing the frame end buffer. The status information is written to the transmission descriptor TDES0. Since the OWN bit is cleared in this step, the CPU now starts to occupy the descriptor. If the timestamp is not enabled, the DMA will not modify TDES2 and TDES3;
8. The transmission interrupt (ETH_DMASRregister [0]) is set after completion of the transmission of the framein which the "completion interrupt" (TDES1 [31]) is set in its last descriptor. The DMA engine goes back to step 3
9. In the suspended state, when the DMA receives a send poll request, the DMA attempts to refetch the descriptor (then returns to step 3) and the underflow interrupt status bit is cleared.

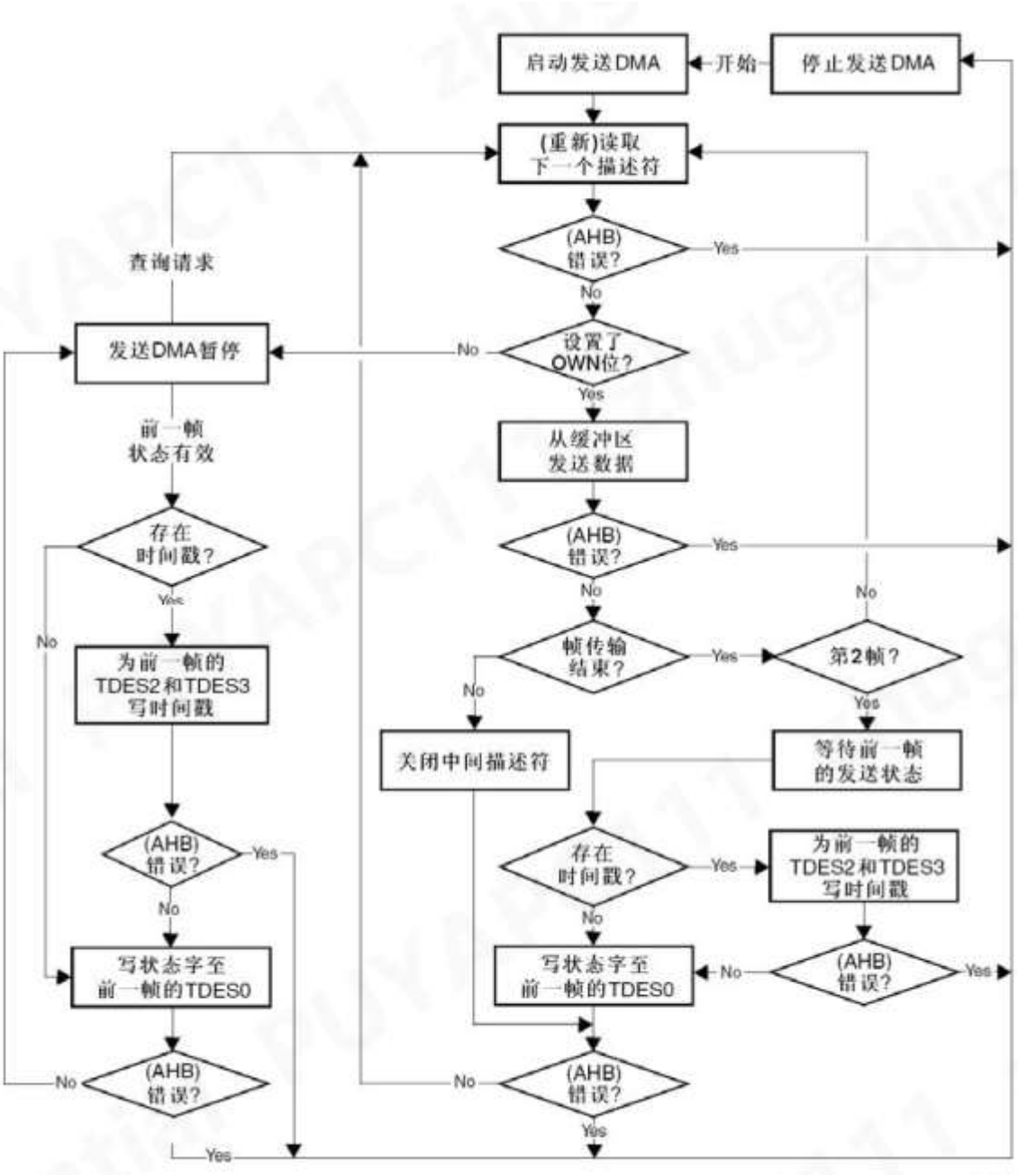
Tx DMA sending process in default mode:



33.5.7.2 OSF mode

When in the running state, the transmitting process may acquire both frames simultaneously without turning off the state descriptor of the first frame (ifthe OSF bitin ETH_DMAOMRisset). When the transmission of the first frame is completed, it immediately polls the transmission descriptor list of the second frame. If the second frame is valid, the transmission flow transmits the frame before writing the status information of the first frame. In OSF mode, DMA operates as follows in the running state:

1. The DMA operation is the same as steps 1-6 of TxDMA (default mode);
2. There is no need to close the last descriptor of the previous frame, DMA will directly take the next descriptor;
3. If the acquired descriptor is occupied by the DMA, the DMA decodes the address of the send buffer. If the DMA does not occupy the rights of the descriptor, the DMA will enter a suspended state and jump to step 7;
4. DMA takes frame data from memory and sends it, and the end of the frame is sent directly. If the frame is split into multiple descriptors, the intermediate descriptor is turned off;
5. The DMA waits for the transmission status and clock stamp of the previous frame, and when the status becomes certain, if the timestamp is captured, the DMA writes the timestamp value to TDES2 and TDES3. The MA then writes the status (cleared OWN bit) to the corresponding TDES0, thereby closing the descriptor. DMA does not modify TDES2 and TDES3 if the timestamp is not enabled in the previous frame;
6. If the send interrupt is enabled and juxtaposed, the DMA fetches the next descriptor and then performs step 3 (if the status is normal). If the status of the previous frame is an underflow error, the DMA enters suspend mode (step 7);
7. In suspended mode, if the suspended state and clock stamp are received by the DMA, the DMA will write the timestamp to TDES2 and TDES3, then write the state to the corresponding TDES0, then set the relevant interrupt and return to the suspended state.
8. Only when a send poll command (ETH_DMATPDRregister) isreceived, the DMAleaves the suspended state and then enters the running state (proceed to steps1and2)



33.5.7.3 The process of sending the frame

The transmitting DMA expects that the data buffer can contain the entire Ethernet frame, except for the preamble, padding field, and FCS field. The destination address, source address, and type/length fields contain some data. If the transmit descriptor shows that the MAC controller must turn off CRC and padding insertion, the buffer must have a full Ethernet frame (excluding the preamble), including the CRC bytes. A frame may be a data link and span multiple buffers. Frames must be separated by a first buffer descriptor (TDES1 [29]) and a last buffer descriptor (TDMES1 [30]). When transmission starts, TDES1 [29] needs to be set in the first descriptor. When it is set to 1, frame data starts to be transmitted from the buffer area to the Tx FIFO. If the last descriptor of the current frame, TDMES0 [29], is cleared, the sending flow attempts to get the next descriptor. The sending process expects TDES1 [29] to be cleared in this descriptor. If TDES1 [30] is cleared, it indicates that an intermediate buffer exists. If TDES1 [30] is set, it indicates the last buffer area of the frame. When the last buffer

area of the frame is sent, the DMA will write the last status information to the word of transmit descriptor 0 (TDES0), which sets the last segment in transmit descriptor 0 (TDES1 [30]). At the same time, if the completed interrupt (TDES1 [31]) is set, the send interrupt flag (bit0) inthe ETH_DMASR registerwill be set to1, and the next descriptor will be taken away and repeated. The transmission of the actual frame begins after the transmission FIFOreaches the programmable transmission threshold (ETH_DMAOMRregister[16:14]), or when a complete frame is saved tothe FIFO. And of course the store-and-forward mode. When DMA completes sending a frame, it will release the occupancy (the OWN bit of TDES0 [31] is cleared)

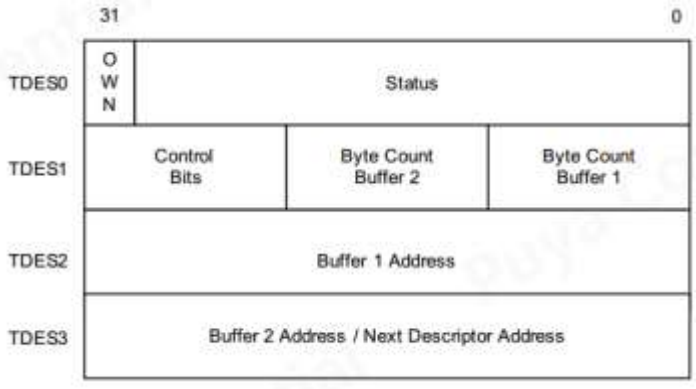
33.5.7.4 Send polling is suspended

The send polling process is suspended by the following conditions:

- The DMA detectsthat the descriptor isoccupied by the CPU, and the unavailability flag ofthe sending buffer area is set to 1(bit [2]ofthe ETH_DMASR register). If it wants to continue, the driver must grant ownership of the descriptor tothe DMA, and then issue a polling request command;
- When a transmission error due to an underflow is detected, the frame transmission is aborted. The appropriate Transport Descriptor 0 (TDES0) bit is set. If the second case is sent, both the total exception interrupt (in the ETH_DMASR register[15]) and the send underflow bit (inthe ETH_DMASR register[5]) are set to1, and this information is written to descriptor0, causing a pause and suspend. If the DMAenters the suspended state because of the first case, both the total normal interrupt bit (ETH_DMASRregister[16]) and the transmit buffer unavailable bit (ETH-DMASR register[2]) are set to1. In both cases, the location of the send list is saved, which is the next descriptor of the last descriptor closed by DMA. The driver must issue a send poll demand command after the correction pause.

33.5.7.5 Send DMA descriptor

At least one descriptor is required for each frame transmission



- TDES0: transmit descriptor word 0

TDES0 contains the status of the transmitted frame and the owner attribute of the descriptor. The status bit is only valid at LS position 1, so when a frame is transmitted through multiple descriptors, the status bit is only valid at the last descriptor.

31	**OWN**: Own bit 1: represents the DMA occupancy descriptor. 0: represents a CPU occupancy descriptor. After the DMA transmits the entire frame or reads out all the data in this buffer, the bit is cleared '0'. The occupancy bit of the first buffer descriptor of each frame must be set to '1' only after all the occupancy bits of the subsequent buffer descriptors are set to '1'.
30 : 18	Reserved
17	**TTSS:** Transmit time stamp status **The status bit indicates that a timestamp was captured in the transmit frame** When set to '1': TDES2 and TDES3 contain the captured timestamp value. This bit is only valid when the LS bit is 1
16	**IHE**: When IP header error is set to '1', it indicates that the checksum load shedding function has detected an IP header error. This bit is only valid when the send checksum load shedding function is enabled. Otherwise, when an IP header error is detected, the IPv4 header checksum will still be inserted if the Ethernet type field shows IPv4 payload
15	**ES**: Error Summary TDES0 [16]: IP header error TDES0 [14]: Jabber timer overflow TDES0 [13]: frame clear TDES0 [12]: payload checksum error TDES0 [11]: carrier loss TDES0 [10]: no carrier TDES0 [9]: late collision TDES0 [8]: too much collision TDES0 [2]: too much delay TDES0 [1]: underflow error
14	**JT**: When the Jabber timer overflow (Jabber timeout) is set to '1', it indicates that a verbose timeout has occurred at the MAC sender. This bit is set to '1' only if the JD bit of the MAC set register is not '1'.
13	**FF**: When Frame flushed is set to '1', it means that the DMA/MTL clears the frame from the FIFO due to a command issued by the CPU.
12	**PCE:** Payload Checksum Error When set to '1', it means that the checksum load shedding function detects an error and does not insert the checksum into TCP, UDP, and ICMP valid data. There are two kinds of errors: 1. The actual number of valid data bytes is less than the value of the valid data length field of the IP header; 2 In store-and-forward mode, before the checksum has been calculated, the transport layer starts forwarding frame data to the MAC transmitter. The second case occurs only if the depth of the transmission FIFO is less than the length of the Ethernet frame to be transmitted.
11	When **LOC**: Loss of carrier is set to '1', it indicates that a carrier loss occurred at the time of frame transmission (at the time of transmission, the MII_CSR signal is invalid in one or more transmission clock cycles). This bit is only valid in half-duplex mode when there is no collision in the transmitted frame.
10	**NC**: When No carrier is set to '1', it indicates that the carrier sense signal of the PHY is invalid at the time of frame transmission.
9	**LC**: When Late collision is set to '1', it indicates that transmission of the frame is stopped because a collision occurs after a collision window (64 bytes of time including the preamble in MII mode) at the time of transmission. If error position '1' is overflowed, this bit is invalid.
8	**EC**: When Excessive collision is set to '1', it indicates that transmission of the frame is stopped because 16 consecutive collisions occur during transmission. If the RD (no retry) bit of the MAC set register is' 1 ', then after a collision, the bit is set' 1 'and the transmission is aborted.
7	**VF**: When VLAN frame is set to '1', it indicates that the transmitted frame is a VLAN frame
6: 3	**CC**: Collision count This 4-bit value records the number of collisions that occur before the frame is sent out. When the EC bit (TDES0 [8]) is' 1 ', the bit is invalid
2	**ED**: When Excessive deferral is set to '1', it means that when the delay check bit of the MAC setting register is' 1 ', the transmission is terminated due to the delay of more than 24288 bits
1	**UF**: When data Underflow error (Underflow error) is set to '1', it means that data from RAM to MAC is too late, causing the MAC to stop sending the frame. A data underflow error indicates that the DMA encountered an empty buffer when sending the frame. The transmission process enters the suspended state, and the transmission data underflow bit (ETH_DMASR register bit 5) and the transmission status bit (ETH_DMASR register bit 0) are set to '1'

0	**DB**: When the Deferred bit is set to '1', it indicates that the MAC delays transmission because the carrier is occupied. This bit is only valid in half-duplex mode.

■ TDES1: transmit descriptor word 1

TDES1 contains the size of the buffer and the control bits of other control descriptor chains or rings when transmitting frames

31	**IC**: Interrupt on Completion When set to '1': When the current frame is sent, the transmission interrupt (ETH_DMASR) will be set. Note: This bit is valid only if LS position 1
30	**LS:** Last Segment When set to '1': it means that the buffer area contains the last block of the frame. At this time, the TBS1 and TBS2 bit segments should be non-zero values
29	**FS:** First Segment When set to '1', it means that the buffer area contains the first block of the frame
28: 27	**CIC**: Checksum insertion control These 2 bits control the calculation and insertion of the checksum: 00: no checksum is inserted; 01: IPv4 header checksum is inserted using this value when the frame encapsulates an IPv4 datagram 10: TCP/UDP/ICMP checksum is inserted, the checksum is calculated only on the TCP, UDP or ICMP segment, and the TCP, UDP and ICMP pseudo-header checksum is assumed to be present in the check-sum field of the corresponding input frame. If the encapsulated datagram is IPv4 compliant, an IPv4 header checksum is also inserted. 11: Fully computed TCP/UDP/ICMP checksum is inserted, i.e. the false header is also computed, and the corresponding checksum field of the input frame has an all-zero value. If the encapsulated datagram is IPv4 compliant, an IPv4 header checksum is also inserted. The checksum engine detects whether a TCP, UDP, or ICMP segment is encapsulated in IPv4 or IPv6 and processes its data accordingly. This bit is only valid if FS position 1
26	**DC**: When Disable CRC (Disable CRC) is set to '1', MAC will not send the additional CRC at the end of the frame This bit is only valid if FS position 1
25	**TER**: When the Transmit End of Ring is set to '1', it means that the descriptor list has reached the last descriptor and will return to the initial address of the list to create a descriptor ring
24	**TCH:** Address of the next descriptor chain (second address chain) When set to '1', the bit segment representing the next address in the descriptor represents the address of the next descriptor instead of the address of the next cache. When this bit is 1, the value in TBS2 has no effect. The function of TER takes precedence over the standard
23	**DP**: When Disable Padding is set '1', when the frame is less than 64 bytes, the MAC will not automatically increase the padding field. When this bit is reset, DMA automatically increments padding and CRC. This bit is valid only if the FS bit is 1
22	**TTSE**: enables hardware timestamp function when Transmit Timestamp Enable is set to '1' Valid only if the FS bit is 1
21: 11	**TBS2**: The size of Transmit Buffer 2 (Transmit Buffer 2 Size) indicates the size of the second buffer area. This bit is invalid when TCH is 1
10: 0	**TBS1**: Transmit Buffer 1 Size represents the size of the first buffer**.** If the value is 0, DMA ignores this buffer and uses buffer 2 or the next descriptor (depending on TCH)

■ TDES2: transmit descriptor word 2

TDES2 contains the address of the first buffer of the descriptor

31 : 0	Buffer 1 Address Pointer (Buffer 1 Address Pointer) **This bit segment represents the physical address of Buffer 1, and the alignment of buffer address is not limited**

■ TDES3: transmit descriptor word 3

TDES3 contains the second buffer of the descriptor or the address of the next descriptor

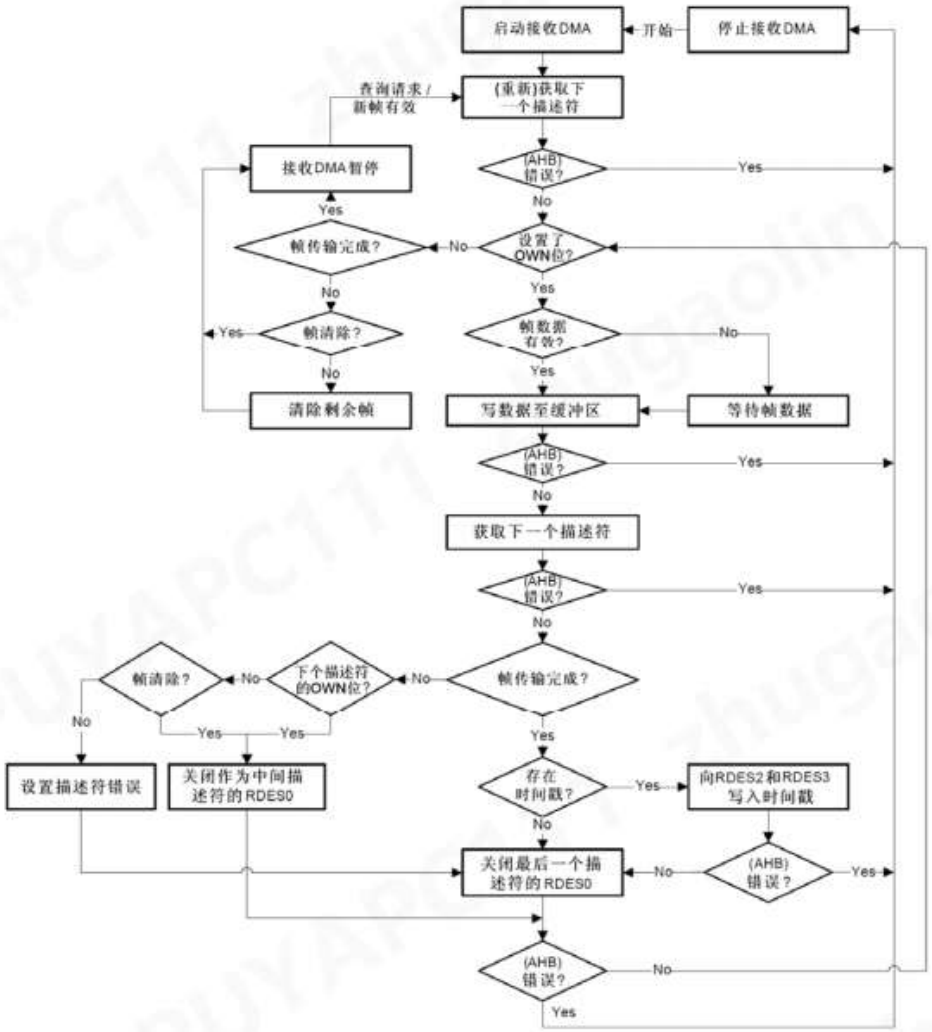
31 : 0	Buffer 2 Address Pointer/Next Descriptor Address (Buffer 2 Address Pointer/Next Descriptor Address) When the ring descriptor architecture is used, this bit segment represents the physical address of Buffer 2 When TCH in descriptor word 1 is set to 1, this bit segment represents the next descriptor physical address, and the cache address must be aligned with the bus width.

33.5.8 Rx DMA Configuration of

The receiving step of the receiving DMA engine is as follows:

1. The CPU sets the receive descriptors (RDES0-RDES3) and sets the OWN bit;
2. Oncethe SR positionofthe ETH_DMAOMR registeris1,the DMAenters the run mode. In running mode, the DMA polls the list of receive descriptors in an attempt to get the idle descriptor. If the fetched descriptor is not idle (CPU occupied), the DMA will enter a suspended state and jump to step 9;
3. Decoding the address of the received data buffer area by the acquired descriptor DMA;
4. The received frame is processed and placed in the data buffer area of the descriptor;
5. When the buffer area is full or a frame of data is transmitted, the receiving engine will exchange the next descriptor;
6. If the current frame has been transmitted, the DMA proceeds to step 7. If the DMA does not occupy the next acquired descriptor and the frame transmission has not been completed, the DMA will set the descriptor error position 1 in RDES0 (unless emptying is prohibited). DMA will close the current descriptor (by clearing the OWN bit) and mark it as an intermediate descriptor by clearing the LS bit in the RDES1 value (if clearing is not disabled, mark it as the last descriptor), and then perform step 8. If DMA occupies the next descriptor but the current frame has not been sent, DMA will close the current descriptor as an intermediate descriptor and return to step 4;
7. If the IEEE 1588 timestamp is enabled, the DMA writes timestamp data to RDES2 and RDES3 of the current descriptor. The status information of the received frame is then written into the status word (RDES0), the OWN bit is cleared, LS position 1.
8. The receiving engine checks the OWN bit of the latest descriptor. If the CPUowns the descriptor (OWN bitis0), the receive buffer area unavailable bit is set (inthe ETH_DMASR register[7]) andthe DMAreceive engine enters a suspended state (step9). If the DMA has a descriptor, it returns to Step 4 to wait for the next frame
9. When the receiving engine enters the suspended state, it will clear some frames in the receiving FIFO (the clear control bit can be set);
10. When a reception poll request is given or the start of the next frame is available from the reception FIFO, the reception DMA exits the suspended state. The engine goes back to step 2 and refetches the next descriptor

The DMA does not acknowledge the receive status until it completes the timestamp write-back and is ready to perform the status write-back to the descriptor. If the timestamp function is enabled by the software, the DMA writes all 1s to RDES2 and RDES3 when a valid timestamp value is invalid for the frame (e.g., the receive FIFO is full when the timestamp is written). Otherwise (timestamp is not enabled), RDES2 and RDES3 do not change.



33.5.8.1 Get Receive Descriptor

The receiving engine will always attempt to make additional descriptor acquisitions for incoming frames. Attempts to get a descriptor when any of the following conditions are met:

- Whenthe DMAis running, immediately set the receive start/receive bit (ETH_DMAOMR bit [1])
- The data buffer area of the current descriptor is full before the end of the received frame
- The frame has finished receiving, but the currently received descriptor is not closed
- The reception process is suspended because the CPU occupies the buffer area or a new frame is received
- Issue a receive poll request

33.5.8.2 Receive frame process

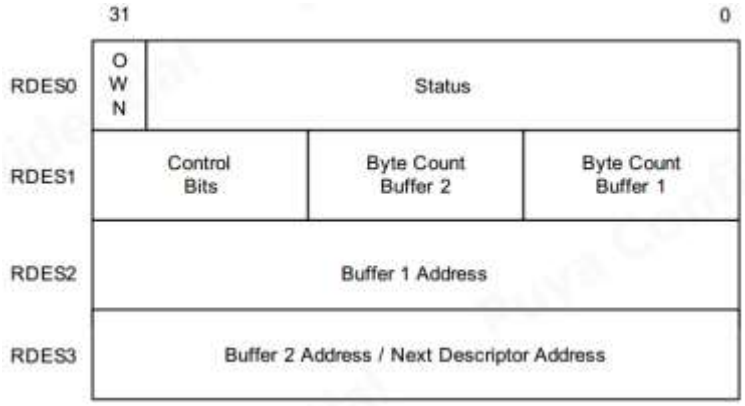
The MAC transmits and receives frames only if the frame has passed address filtering and the size of the frame is greater than or equal to the FIFO threshold configured by F, or uses store-and-forward mode. If frame address filtering fails, it will be discarded directly (unlessETH_MACFFR [31] is setto receive all frames). Frames shorter than 64 bytes (configurable) are cleared from the receive FIFO due to collision or early termination. The DMA block begins transmitting frame data to the receive buffer area pointed to by the current descriptor. The DMA sets the first descriptor (RDES0 [9]) after the AHB interface can receive data transmission. The descriptor is released when the OWN (RDES0 [31]) bit is reset to 0, either when the data buffer is filled or when the last segment of the frame is

transmitted to the receive buffer. If a frame is included in a descriptor, the first descriptor (RDES0 [9]) and the last descriptor (RDES0 [8]) identification bits are both set to 1. After the DMA sets 1 the last descriptor in the previous descriptor and releases the RDES0 status bit, it starts fetching the next descriptor. The DMA then sets the receive interrupt bit (bit [6]ofthe ETH_DMASR register) to1. The same process is repeated unless the DMA encounters a descriptor marked as being owned by the CPU. If this is sent, the receiving process will set the receive buffer invalid bit (ETH_DMASR bit [7]) to1and enter the suspended state. The position in the receive list will be preserved.

33.5.8.3 Receiving process suspended

If a new receive frame arrives while the receive process is suspended, the DMA refetches the descriptor currently in memory. If the descriptor is owned by the DMA, the receiving process will re-enter the running state to start the reception of the frame. If the descriptor is still owned by the CPU, the DMA discards the current frame at the top of the Rx FIFO by default and increments the lost frame counter. This process repeats if there is more than 1 completion frame in the Rx FIFO. Dropping or refreshing frames on top of the Rx FIFO can be avoided by setting the DMA mode of operation register bit 24 (DFRF). In this case, the receiving process sets the receiving buffer area unavailable status bit and returns to the suspended state.

33.5.8.4 DMA receive descriptor



- RDES0: Receive Descriptor Word 0

RDES0 includes information such as the state of the received frame, the length of the frame, and the assignment of the descriptor. These status bits are only valid when the LS position is 1.

31	**OWN**: Own bit 1: represents the DMA occupancy descriptor. 0: represents a CPU occupancy descriptor. The DMA clears the bit '0' after receiving the entire frame or the data in the buffer is completely filled.
30	**AFM** Destination Address Filter Fail When set to '1': MAC destination address filtering fails
29: 16	**FL**: Frame Length (Frame Length) This bit segment indicates the byte length of the received frame being sent to system memory. Valid only when LS (RDES0 [8]) is set to 1, descriptor error bit (RDES0 [14]), or overflow error bit is 0. The frame length also includes two bytes appended to the Ethernet frame when the IP checksum calculation is enabled and the received frame is not a MAC control frame. When the last descriptor bit is not set to 1, this bit segment indicates the cumulative number of bytes that have been transmitted in the current frame. When bit7 and bit25 of the ETH_MACCR register are set to 1, the frame length also includes length information of the CRC.
15	**ES**: Error Summary RDES0 [0]: Payload checksum error RDES0 [1]: CRC error RDES0 [3]: Receive error

	TDES0 [4]: Watchdog overflow TDES0 [6]: Late collision TDES0 [7]: IPC checksum error TDES0 [11]: Overflow error TDES0 [14]: Descriptor error This bit is valid only if the last descriptor is set to 1
14	**DE**: When Descriptor error is set to '1', this bit indicates that the frame is cut because the cache of the descriptor cannot hold the current frame, and the DMA does not occupy the next descriptor. This bit is only valid when the LS bit (RDES0 [8]) is' 1 '.
13	**SAF**: When the Source address filter fail is set to '1', this bit indicates that the frame did not pass the MAC controller's source address (SA) filter.
12	**LE**: When Length error is set to '1', it means that the actual length of the received frame does not match the value of the Ethernet frame header length/type field. This bit is only valid when the RDES0 [5] bit is' 0 '.
11	**OE**: When Overflow error is set to '1', it indicates that the received frame is corrupted due to the received FIFO overflow.
10	**VLAN**: When the VLAN tag is set to '1', it indicates that the frame pointed to by the descriptor is marked as a VLAN frame by the MAC controller.
9	**FS**: When the First descriptor is set to '1', it means that this descriptor contains the first buffer of the frame. If the length of the buffer is 0, the second buffer contains the beginning of the frame; if the length of the second buffer is also 0, the next buffer contains the beginning of the frame.
8	**LS**: When the Last descriptor (Last descriptor) is set to '1', it means that the buffer area pointed to by this descriptor is the last buffer of the frame.
7	When the IPC Checksum Error is set to '1', it indicates an IPv4 or IPv6 header error. The error may be caused by a mismatch between the Ethernet type domain and IP version domain values, an incorrect IPv4 header checksum, or an insufficient number of IP header bytes for an Ethernet frame.
6	**LCO**: When Late collision (Late collision) is set to '1', it indicates that a Late collision occurs when receiving a frame in half-duplex mode.
5	**FT**: When the Frame type is set to '1', it means that the received frame is an Ethernet frame (LT field is greater than or equal to 0x0600). When '0' is set, it indicates that the received frame is an IEEE 802.3 frame. This bit is invalid when the received frame is an too short frame of less than 14 bytes.
4	**RWT**: Receive watchdog timeout (Receive watchdog timeout) set to '1' indicates that when receiving the current frame, the watchdog timeout, after which the current received frame is truncated
3	**RE**: When Receive error is set to '1', it means that RX_ERR signal is valid when RX_DV is valid during receiving a frame
2	**DE**: Dribble bit error When set to '1', it indicates that the received frame length is not an integer multiple of bytes (odd number of 4-bit data). This bit is only valid in MII mode
1	**CE**: When CRC error is set to '1', it indicates that a cyclic redundancy detection (CRC) error occurs when the frame is received. This bit is only valid when the LS bit (RDES0 [8]) is' 1 '
0	**PCE**: When the data checksum error (Payload checksum error) is set to '1', it indicates that the TCP, UDP, or ICMP check calculated by the MAC controller does not match the TCP, UDP, or ICMP checksum field value of the received frame. This bit is also set to '1' when the data length of the received Ethernet frame does not match the value of the IPv4 or IPv6 packet length field

Bit 5: FrameType	**Bit 7: IPCChecksum Error**	**Bit 0: Payload-Checksum Error**	**Frame Status**
0	0	0	IEEE802.3 type frame (length field value less than 1536)
1	0	0	bit10 of ETH_MACCR is 1: IPv4/IPv6 type frame, no checksum error detected bit10 of ETH_MACCR is 0: IEEE802.3 type frame (frame length greater than or equal to 1536)
1	0	1	IPv4/IPv6 type frame, payload checksum error detected
1	1	0	IPv4/IPv6 type frame, IP header checksum error detected
1	1	1	IPv4/IPv6 type frame, payload checksum error and IP header checksum error detected

0	0	1	IPv4/IPv6 type frame, no payload checksum error detected, no IP header checksum error
0	1	1	Non-IPv4/IPv6 type frames
0	1	0	Reserved

- RDES1: Receive Descriptor Word 1

RDES1 contains the cache size and other control bits that control the ring or chain descriptor

31	**DIC**: When the reception completion interrupt (Disable interrupt on completion) is turned off and set to '1', the RS bit (bit 6) of the ETH_DMASR register is not set for the frame indicated by the current descriptor when frame reception is completed. This also prevents RS bit triggered interrupts from occur This bit is valid only if the last descriptor bit (RDES0 [8]) is set to 1
30: 26	Reserved
25	**RER**: When the Receive end of ring is set to '1', this bit indicates that the current descriptor is the last in the descriptor queue. The DMA will return to that base address of the descriptor queue to take the next descriptor, so that the descriptor queue forms a ring structure
24	**RCH**: When the Second address chained list (Second address chained) is set to '1', this bit indicates that the second address in the descriptor is the address of the next de-scriptor, not the second cached address. When this bit is' 1 ', the value of RBS2 (TDES1 [28: 16]) can be ignored. The role of RDES0 [15] takes precedence over the native (RDES0 [14]).
23: 22	**Reserved**
21: 11	**RBS2**: The size of Transmit Buffer 2 (Transmit Buffer 2 Size) represents the size of the second buffer area, and the buffer size must be a multiple of 4. This bit is invalid when RCH is 1
10: 0	**RBS1**: The size of Transmit Buffer 1 (Transmit Buffer 1 Size) represents the size of the first buffer area, and the buffer size must be a multiple of 4. If the value is 0, DMA ignores the buffer and uses buffer 2 or the next descriptor (depending on the RCH)

- RDES2: Receive Descriptor Word 2

RDES2 contains the address of the first buffer of the descriptor

31 : 0	Buffer 1 Address Pointer (Buffer 1 Address Pointer) This bit segment represents the physical address of buffer 1, and there is no restriction on the alignment of buffer address**.** When the RDES2 value is used to store the beginning of a frame, the DMA uses the configured value to generate its address. During the transmission at the beginning of the frame, the DMA performs a write operation with the RDES2 [1: 0] bit as 0, but the frame data is shifted according to the actual buffer address pointer.

- RDES3: transmit descriptor word 3

RDES3 contains the second buffer of the descriptor or the address of the next descriptor

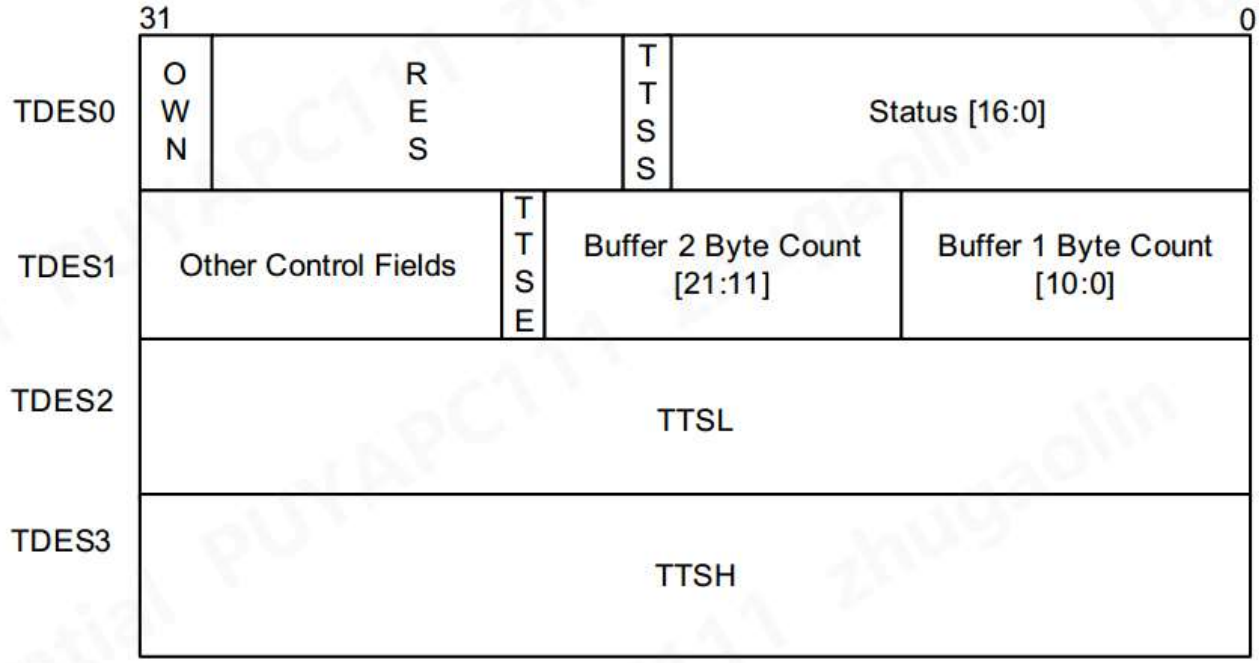
31 : 0	Buffer 2 Address Pointer/Next Descriptor Address (Buffer 2 Address Pointer/Next Descriptor Address) When the ring descriptor architecture is used, this bit segment represents the physical address of Buffer 2 When RCH in descriptor word 1 is set to 1, this bit segment represents the next descriptor physical ad-dress, and the cache address must be aligned with the bus width. When RCH is set to 0, there is no limit to the value of RDES3, except when the RDES3 value is used to store the beginning of the frame, the DMA uses the configured value to generate its address

33.5.9 Descriptor format to enable IEEE1588-2005 timestamps

If the software enables the IEEE1588-2005 feature, the contents of the DES2 and DES3 descriptor regions have different meanings when the DMA turns off the descriptor (clears the OWN bit of DES0). The timestamps are updated to DES2 and DES3 before DMA clears the OWN bit of DES0.

33.5.9.1 Send descriptor

The send descriptor has more control (TTSE and TTSS) to send the timestamp. As shown in the figure below, when the software sets TTSE (when OWN is 1), it means that the timestamp will be generated for sending Ethernet frames. DMA will update the timestamps to TDES2 and TDES3 before the de-scriptor is turned off.



- TDES0:

17	TTSS: Transmit Timestamp Status This field is a status bit indicating that a timestamp has been captured for the corresponding transport frame. When this bit is set, both TDES2 and TDES3 have timestamp values captured for the transmission frame. This field is valid only if the last segment control bit (TDES1 [30] in the descriptor) is set.

- TDES1:

22	TTSE: Transmit Timestamp Enable When set, this field enables the IEEE1588 hardware timestamp for the transmit frame described by the descriptor. This field is only valid if the first segment control bit (TDES1 [29] in the descriptor) is set.

The transport descriptor format and field description remain unchanged when created by software (OWM is 1), however, when the last descriptor is turned off by DMA and the IEEE1588 function is enabled, the TDES2 and TDES3 contents are timestamped updated.

- TDES2:

31: 0	TTSL: Transmit Frame Timestamp Low The DMA updates this field with the least significant 32 bits of the timestamp captured for the corresponding transport frame. This field has a timestamp only if the last segment control bit (LS) in the descriptor is set

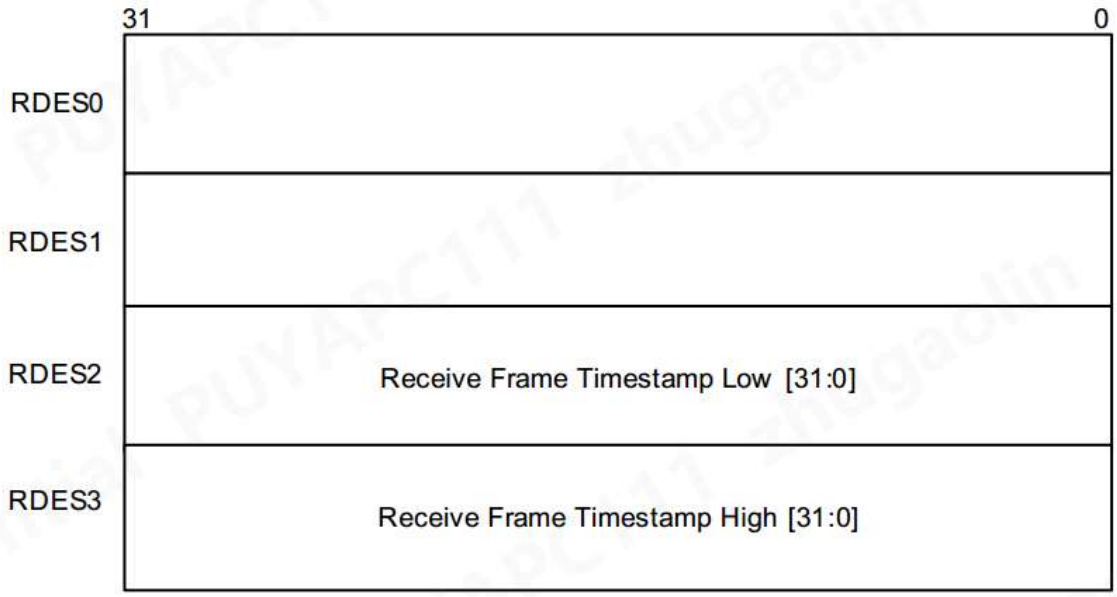
- TDES3:

31: 0	TTSL: Transmit Frame Timestamp High The DMA updates this field with the most significant 32 bits of the timestamp captured for the corresponding transport frame. This field has a timestamp only if the last segment control bit (LS) in the descriptor is set

33.5.9.2 Receive descriptor

When the timestamp is enabled, the receive descriptor format is as follows:

When the receive descriptor is off and the timestamp is enabled, RDES2 and RDES3 of the receive descriptor have fields of different meanings.



- RDES2:

31: 0	RTSL: Receive Frame Timestamp Low The DMA updates this field with the least significant 32 bits of the timestamp captured for the corresponding received frame. The DMA updates this field only for the last descriptor of the received frame indicated by the last descriptor status bit (RDES0 [8]). When this and RTSH fields in RDES3 show all 1 values, the timestamp must be considered corrupted

- RDES3

31: 0	RTSH: Receive Frame Timestamp High The DMA updates this field with the most significant 32 bits of the timestamp captured for the corresponding received frame. The DMA updates this field only for the last descriptor of the received frame indicated by the last descriptor status bit (RDES0 [8]). When this and RTSH fields in RDES3 show all 1 values, the timestamp must be considered corrupted

33.5.9.3 DMA interrupts

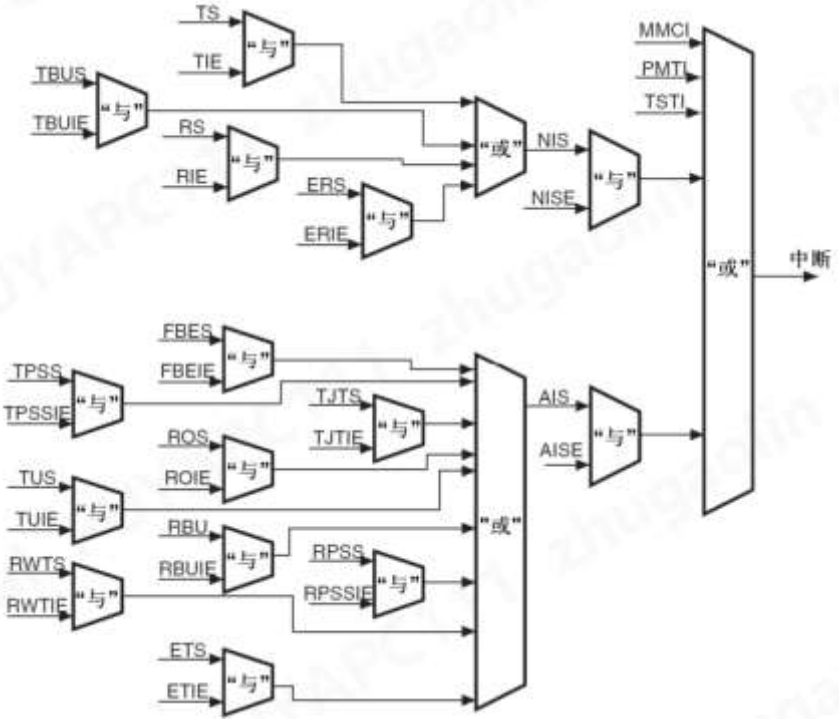
There are a variety of events that can generate an interrupt. The ETH_DMASRcontains all bits that can cause an interrupt, whilethe ETH_DMAIER registercontains every interrupt enable bit that can generate an interrupt event.

There are twogroups of interrupts, as described inthe ETH_DMASR register, which can be divided into normal group and abnormal group. The interrupt can be cleared by writing '1' to the corresponding bit. If all enabled interrupts in a group are cleared, the summary bits of that group are also cleared accordingly. If the interrupt is triggeredby the MAC controller, the TSTSbit orthe PMTS bitofthe ETH_DMASRis set high.

Interrupt events are not queued, that is, if an interrupt occurs, another interrupt will not occur before responding to it. For example, the receive interrupt bit (ETH_DMASR [6])indicates that one or more frames have been received in the buffer, and the interrupt handler needs to retrieveall DMAoccupied descriptors from beginning to end.

Multiple events that can cause interrupts at the same time can generate only one interrupt. The interrupt handler needs to check the ETH_DMASRregister to determine the source of the interrupt. After the corresponding bit of the ETH_DMASR registeris properly cleared, no interrupt will be generated unless another interrupt event occurs. For example, the controller generates a receive interrupt (ETH_DMASR [6]is' 1 '), and after the interrupt handler reads outthe ETH_DMASR register, the receive

buffer unavailable(ETH_DMASR [7]is' 1 ')event occurs; After the receive interrupt is cleared first, a new interrupt is generated because the receive cache unavailable interrupt is still valid and needs to be processed



33.6 Ethernet interrupt

The Ethernet controller has two kinds of interrupt vectors: one for regular operation-induced interrupts and one for Ethernet wake-up time (mapped at EXTI line 31)

The first is an interrupt generated for MAC and DMA, as described in the sections of MAC interrupts and DMA interrupts. The second is generated by the PMT for wake-up events. The wake-up event is an interrupt generated to exit the low power mode. When a wake-up event mapped to EXTI 31 occurs, if enable is turned on, it is detected that its rise is used to exit the low power mode.

When the Ethernet wake-up event is mapped to the EXTI line 31, once the wake-up event occurs, if the PMT interrupt of the MAC is enabled, and the interrupt is generated when the rising edge is detected by the EXTI line 31 is enabled, then two interrupts will occur simultaneously.

There is a watchdog timer outsidethe Ethernet network(seeETH_DMARSWTRregister), which can be used to flexibly control the setting ofthe RS bit(ETH_DMASR [6]). If a value other than zero is written to the watchdog and no receive interrupt is enabled in the descriptor of the received frame (RDES1 [31] = 0), the watchdog timer is activated when the receiving DMA receives the frame into system memory without setting the RS bit to '1'. The RS bit is set to '1' only when the watchdog timer is decremented to 0, and an interrupt occurs if the corresponding RIE interrupt is enabled. Before the watchdog timer is decremented to 0, if a frame is received, the frame descriptor enables the receive interrupt, and the RS bit is set to '1' while the watchdog timer is turned off.

Note: Reading the PMT control and status registers automatically clears the wake-up frame reception and Magic Packet reception interrupt flag bits. However, since the register in which these flags are located is located in the CLK_RX domain, there may be a considerable delay before the application

sees these flags cleared. This delay will be particularly long if the RX clock is slow (such as in 10M-bit mode) and the AHB bus clock frequency is high.

Since the interrupt request generated byPMTtothe CPUis also in the same register in the RX_CLK field, the CPUmay incorrectly call the interrupt programagain after readingthe PMT_CSR register. Therefore, the application needs to query the wake-up frame reception and Magic Packet reception flag bits in the interrupt until they become 0 before exiting the interrupt service routine.

33.7 TIMx registers

33.7.1 EthernetMACconfiguration register (ETH_MACCR)

Address offset: 0x0000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WD	JD.	Res.	Res.	IFG			CSD
-	-	-	-	-	-	-	-	RW	RW	-	-	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	FES	ROD	LM.	DM.	IPCO	RD	Res	APCS	BL		DC	TE	RE	Res.	Res.
-	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	-	-

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	WD	RW	0	Disable Watchdog Timer 1: Turn off the timer and can send up to 16384 bytes of frame data 0: When the transmitted data exceeds 2048 bytes, the data transmitted later is cut off
22	JD	RW	0	Disable Jabber Timer 1: Turn off the timer and can send up to 16384 bytes of frame data 0: When the transmitted data exceeds 2048 bytes, the data transmitted later is cut off
21: 20	Reserved	-	-	-
19: 17	IFG	RW	0	Inter-frame gap (Interframe gap) The minimum gap between two frames in the process of control frame transmission 000: 96bit corresponding time 000: 88bit corresponding time 000: 80bit corresponding time … 000: 40bit corresponding time Note: In half-duplex mode, the minimum IFG can only be configured for 64bit of time (IFG = 100)
16	CSD	RW	0	Disable carrier sense (Carrier sense disable) 1: The MAC transmitter does not respond to the signal of the MII CRS in half-duplex mode, and will not generate an error when there is no carrier or the carrier is lost. 0: Response, and can generate a transmission error or even abort transmission according to carrier sense
15	Reserved	-	-	-
14	FES	RW	0	Ethernet rate (Fast Ethernet speed) 1: 100 Mbit/s 0: 10 Mbit/s
13	ROD	RW	0	Prohibit Receipt (Receive own disable) 1: MAC does not receive frames in half-duplex mode 0: Receive all frames sent by PHY in half-duplex mode Note: Only available in half-duplex mode
12	LM	RW	0	Loopback mode (Loopback mode)

				0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
11	DM	RW	0	Full Duplex mode 1: Enabled 0: Disabled
10	IPCO	RW	0	IPv4 checksum offload Perform IPv4 checksum check on the TCP/UDP/ICMP header of the payload of the received frame (the sender will calculate the checksum for all fields and put it in the header) 1: Enabled 0: Disabled
9	RD	RW	0	Retry disable Transfer retry function 1: The MAC attempts only one transmission, when a collision occurs on the MII, the MAC ignores the current frame transmission and reports an error is reported in the transmission frame state. 0: multiple outputs, depending on the BL register, Note: For Half Duplex mode only.
8	Reserved	-	-	-
7	APCS	RW	0	Automatic pad/CRC stripping Automatic removal of padding and FCS fields functions 1: When the data of the frame is less than or equal to 1500 bytes, the MAC removes the padding and check fields, and when the data is greater than or equal to 1501, there is no operation. 0: Receiving all frame data without modification
6: 5	BL	RW	0	Backback Limit Back-off limit During the retry after the collision, the MAC needs to wait a random integer multiple of the slot time (the time corresponding to 512 bits) before reattempting to transmit. Note: For Half Duplex Mode Only 00: k = min (n, 10) 01: k = min (n, 8) 10: k = min (n, 4) 11: k = min (n, 1) n = send attempt, values of random integer multiples have an interval of [0, 2 ^ k]
4	DC	RW	0	Delay checking (Deferral check) 1: Enable the delay detection function when the sending state machine delays the sending by more than 24288 bits. The MAC issues a frame abort state and pulls up the excessive delay error flag bit in the transmitted frame. Delay counting will be turned on when the data is ready to be sent, and counting will be interrupted if the CRS signal is pulled up. The delay time does not accumulate. If the sender is delayed by 10000 bits, then sends, collides, bounces, and then must be delayed again after the fallback is complete, the delay is cleared and counted again Works in half-duplex mode only 0: The delay check function is turned off, and the MAC delays until the CRS signal becomes invalid. Note: For Half Duplex Mode Only
3	TE	RW	0	Transmitter enable 1: Enable MAC transmitter state machine 0: Disable the MAC transmitter state machine after completing the transmission of the current frame
2	RE	RW	0	Receiver enable 1: Enable MAC receiver state machine 0: Disable the MAC receiver state machine after completion of reception of the current frame
1: 0	Reserved	-	-	-

33.7.2 EthernetMACFrame Filter Register (ETH_MACFFR)

Address offset: 0x0004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RA.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			VTFE
RW	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res	Res	Res	Res	HPF	SAF	SAIF	PCF		BFD	PAM	DAIF	HM	HU.	PM
-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	RA	RW	0	Receive all 1. Receive all MAC frames regardless of address filtering. The results of SA/DA address filtering are updated (pass or fail) in the receive status word 0: Receive only frames that have passed SA/DA address filtering
30:17	Reserved	-	-	-
16	VTFE			VLAN Tag Filter Enable 1: Enable MAC to discard frames with VLAN tags that do not match the VLAN tags comparison 0: MAC forwards all frames regardless of the matching status of the VLAN tag
15:11	Reserved	-	-	-
10	HPF	RW	0	Hash or complete filtering (Hash or perfect filter) 1: If both HM and HU bits are 1, it means that the frame matching hash filter or perfect filter passed by address filtering 0: If both HM and HU bits are 1, it means that the frame only matches the hash filter
9	SAF	RW	0	Source address filtering (Source address filter) The source address filtering MAC compares the programmed values of the SA field and the SA register of the received frame. If it matches, the SAMatch bit in the receive status word is pulled up 1: If the source address match fails, the MAC drops the frame. 0: Receive the frame, but also perform address comparison judgment, and update and forward SAMatch
8	SAIF	RW	0	Source address reverse filter (Source address inverse filtering) 1: Frames whose source addresses match are considered filtering failures 0: Frames with mismatched source addresses are considered a failure
7: 6	PCF	RW	0	By Control Frame (Pass control frames) Control the forwarding of all control frames (unicast or multicast pause frames) 00 or 01: Do not forward control frame to application 10: Forward all control frames even if address filtering is not passed 11: Forwarding control frame subjected to address filter
5	BFD	RW	0	Disable broadcast frames (Broadcast frames disable) 1: Address filter filters all broadcast frames 0: By receiving broadcast frame
4	PAM	RW	0	Pass all multicast frames 1: through all broadcast frames 0: Filter the broadcast frame, the filtering method depends on the HM bit
3	DAIF	RW	0	Destination address reverse filtering (Destination address inverse filtering) 1: The address check module will reverse filter the DA address of unicast and multicast frames 0: No filtering operation
2	HM	RW	0	Hash multicast filtering 1: MAC performs target address filtering on the received multicast frame through hash list 0: MAC will perform perfect address filtering on the received multicast frame by comparing the DA field of the received frame with the configuration value in the DA register
1	HU	RW	0	Hash unicast frame filtering 1: MAC performs target address filtering on the received unicast frame through a hash list 0: MAC will perform perfect address filtering on the received unicast frame by comparing the DA field of the received frame with the configuration value in the DA register
0	PM	RW	0	Mixed Mode (Promiscuous mode)

				1: Regardless of the destination address and source address of the input frame, it is received, and the flag bit of SA/DA filtering failure will always be cleared

33.7.3 Ethernet MACHash Table High Order Register (ETH_MACHTHR)

Address offset: 0x0008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTH [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTH [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	HTH	RW	0	Hash table high (Hash table high) Contains the upper 32 bits of the multicast hash table

33.7.4 Ethernet MACHash Table Low Order Register (ETH_MACHTLR)

Address offset: 0x000C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	HTL	RW	0	Low Hash Table (Hash table low) Contains the lower 32 bits of the multicast hash table

33.7.5 EthernetMAC MIIaddress filter register (ETH_MACMIIAR)

Address offset: 0x0010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA					MR					Res.	CR			MW.	MB
RW	RW	-	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 11	PA	-	-	PHY address Select a PHY device, up to 32 PHY address devices are supported
10: 6	MR			MII interface register Select the MII register of the PHY device
5: 2	CR	RW	0	Clock Range (Clock range) Select MDC clocks based on different HCLK frequencies When bit [5] = 0: 0000: 60-100Mhz HCLK/42 0001: 100-150Mhz HCLK/62 0010: 20-35Mhz HCLK/16 0011: 35-60Mhz HCLK/26 0100: 150-250Mhz HCLK/102 0101: 250-300Mhz HCLK/104 0101, 0111: reserved When bit [5] = 1, you can configure a MAC clock frequency greater than 2.5 Mhz. 1000: HCLK/4 1001: HCLK/6

				1010: HCLK/8 1011: HCLK/10 1100: HCLK/12 1101: HCLK/14 1110: HCLK/16 1111: HCLK/18
1	MW.	RW	0	MII write 1: Tell PHY that a write operation will be performed through the MII data register 0: A read operation will be performed to store the data in the MII data register
0	MB	RW	0	MII busy This bit must be read to 0 before writing to the eth_macmiiar and eth_macmiidr registers, During writing to ETH_MACMIIAR, this bit must also be reset to 0. During a PHY register access, this bit can be set to 1 to show that a read and write operation is in progress. In a write operation, the data in ETH_MACMIIDR must remain valid until the bit is cleared by the MAC. In the process of reading, the data in ETH_MACMIIDR is not valid until the bit is cleared by the MAC. ETH_MACMIIAR cannot be written until this bit is cleared.

33.7.6 EthernetMAC MIIData Register (ETH_MACMIIDR)

Address offset: 0x0014

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MD															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	
15: 0	MD	RW	0	MII data (MII data) The MII data register stores 16-bit data written and read out of the PHY

33.7.7 EthernetMACFlow Register (ETH_MACFCR)

Address offset: 0x0018

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PT															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res	Res	Res	Res	Res	Res	Res	ZQPD	Res	PLT		UPFD	RFCE	TFCE	FCB/BPA
-	-	-	-	-	-	-	-	RW	-	RW	RW	RW	RW	RW	RW_W1/RW

Bit	Name	R/W	Reset Value	Function
31: 16	PT	RW	0	Pause Time (Pause time) Define the value corresponding to the pause time field of the transmitted control frame
15: 8	Reserved	-	-	-
7	ZQPD	RW		Prohibit zero quantum pause (Zero-quanta pause disable) 1: Prohibit the automatic generation of zero quantum pause control frame when the flow control signal of the FIFO layer is de-asserted 0: Enable automatic generation of zero quantum pause control frame
6	Reserved	-	-	-
5: 4	PLT	RW	0	Suspend Low Threshold (Pause low threshold) The lowest threshold used to configure the pause timer in the pause frame within which the pause frame will be automatically retransmitted. The value of the threshold needs to be smaller than the value configured in the PT. For example, PT = 0x100 (256 SLOT time values), PLT = 01, the pause frame will be retransmitted at 228slot (256-28) time 00: Pause time minus 4 slot time 01: The pause time is reduced by 28 slot time 10: Pause time minus 144 slot time

				11: The pause time is reduced by 256 slot time
3	UPFD	RW	0	Unicast Pause Frame Detection (Unicast pause frame detect) 1: MAC detects a pause frame of a unicast address defined in the ETH_MACAOHR and ETH_MACAOLR registers, or a pause frame of a special multicast address 0: Detect only pause frames of special multicast addresses specified in the 802.3 x standard
2	RFCE	RW	0	Receive flow control enabled (Receive flow control enable) 1: MAC decodes the received pause frame and pauses the receiver (PT) for a specified period of time 0: Disable decoding of paused frame
1	TFCE	RW	0	Send Flow Control Enable (Transmit flow control enable) 1: In full-duplex mode, MAC enables flow control when transmitting a pause frame; Enable back pressure mode 0 in half-duplex mode: in full-duplex mode, flow control is not enabled, and no pause frame is sent; Back pressure mode is not enabled in half duplex mode
0	FCB/BPA	RW	0	Flow control busy/back pressure active (Flow control busy/back pressure activate) This bit initializes a pause frame in full duplex mode; In half-duplex mode, the back pressure function is activated if the TFCE is 1. In full duplex mode, this bit should be read to 0 before writing to the flow control register, and in order to generate a pause frame, this bit needs to be written to 1. When the control frame is transmitted, this bit remains at 1 to indicate that the frame is being transmitted. When the transmission of the pause control frame is completed, the MAC resets this bit to 0. The flow control register cannot be written until this bit is cleared In half-duplex mode, when the bit is 1, the backpressure function is active. In backpressure, when the MAC receives a new frame, the transmitter starts transmitting the JAM sequence to create a collision. When the MAC is configured in full duplex mode, the BPA automatically disables

33.7.8 EthernetMAC VLANtag register (ETH_MACVLANTR)

Address offset: 0x001C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															VLANTC
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VLANTI															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

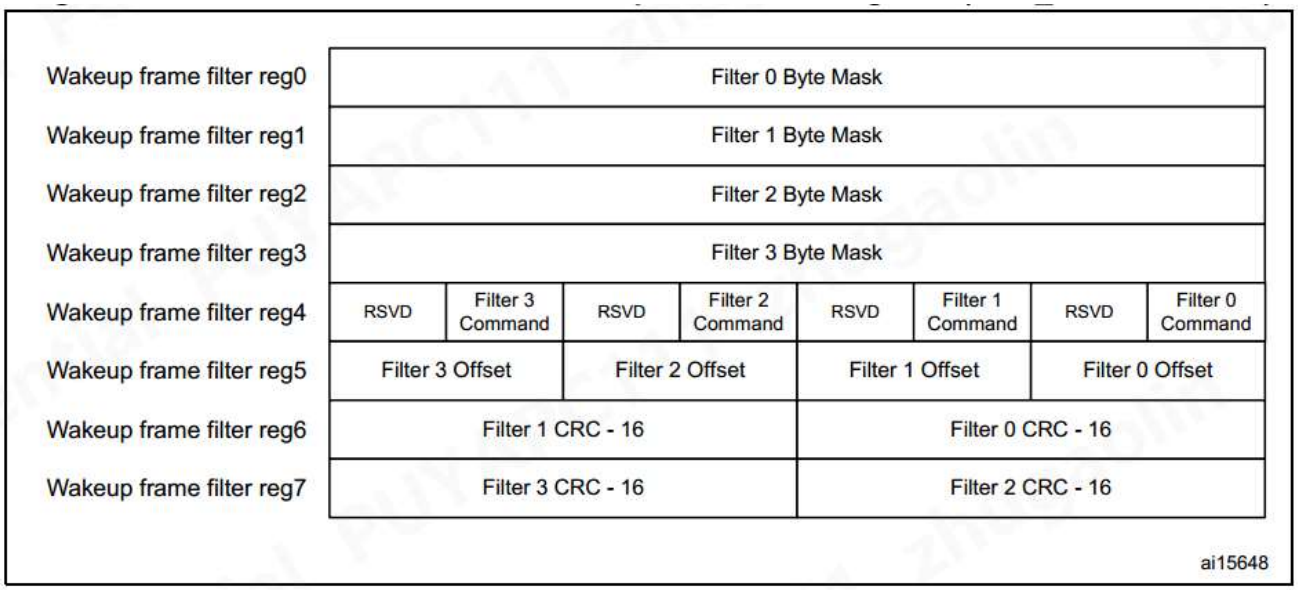
Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
16	VLANTC	RW		12-bit VLAN tag comparison 1: Use the 12-bit VLAN identifier for comparison and filtering instead of the 16-bit VLAN identifier bits [11: 0] to the corresponding field in the received VLAN tag frame 0: 16-bit data of the 15th and 16th bytes in the VLAN frame are used for comparison
15: 0	VLANTI	RW	0	VLAN tag identifier It contains an 802.1 Q VLAN tag to identify the VLAN frame and compares it to the 15th and 16th bytes of the received VLAN frame. bit [15:13] is the user priority, bit [12] is the canonical format indicator (CFI), and bit [11: 0] is the VLAN identifier field of the VLAN tag. When the VLANTC bit is set, only the VID (bit [11: 0]) is used for comparison

33.7.9 EthernetMACRemote Wake Frame Filter Register (ETH_MACRWUFFR)

Address offset: 0x0028

Reset value: 0x0000 0000

This register is essentially a pointer to 8 opaque wake-up frame filter registers (using the same offset address). 8 consecutive write operations to this register address (offset 0x0028) can write all 8 wake-up frame filter registers; Eight consecutive reads of this register address (offset 0x0028) can read all eight wake-up frame filter registers



33.7.10 EthernetMACcontrol and status register (ETH_MACPMTCSR)

Address offset: 0x002C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WFFRPR.	Res.	Res.	RWKPTR					Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	RO	RO	RO	RO	RO	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	GU	Res.	Res.	WFR	WPR	Res.	Res.	WFE	MPE	PD
-	-	-	-	-	-	RW	-	-	RW	RW	-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	WFFRPR.	RW	0	Wake-up frame filter register pointer reset (Wakeup frame filter register points reset) 1: Reset the pointer of the remote wake-up frame filter register to 0b000, and automatically pull it down after one cycle
30:29	Reserved	-	-	-
28:24	RWKPTR	RO		Remote wake-up FIFO pointer Represents the current value of the pointer to the remote wake-up frame filter register
23:10	Reserved	-	-	-
9	GU			Global unicast bal unicast) 1: Unicast packet filtered by MAC address is enabled to become wake-up frame
8: 7	Reserved	-	-	-
6	WFR	RW	0	The wake-up frame is received (Wakeup frame received) 1: Indicates that the power management event is generated by the received wake-up frame and can be read clearly
5	WPR	RW	0	Unicast Pause Frame Detection (Magic packet received) 1: Indicates that the power management event is generated by the received magic packet, which can be read clearly
4: 3	Reserved	-	-	-
2	WFE	RW	0	Wake up frame enable (Wakeup frame enable) 1: Enable PMT event generated by reception of wakeup frame
1	MPE	RW	0	Magic Pack Enabled (Magic Packet enable) 1: Enable PMT event generated by reception of magic packet
0	PD	RW	0	Power-off mode Power down 1: All received frames are thrown away, automatically cleared when magic frames and wake-up frames are received, and the power-off mode is turned

				off. The received frame after the bit is cleared is sent to the upper layer application. This bit must be set only if the magic packet is enabled or the wake-up frame is enabled

33.7.11 EthernetMACInterrupt State Register (ETH_MACSR)

Address offset: 0x0038

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSTS	Res.	Res.	MMCTS	MMCRS	MMCS	PMTS	Res.	Res.	Res.
-	-	-	-	-	-	RC_F	-	-	R	R	R	R	-	-	-

Bit	Name	R/W	Reset Value	Function
30: 10	Reserved	-	-	-
9	TSTS	RW	0	Timestamp trigger state Time stamp trigger status This bit is set to 1 when the system time is equal to or greater than the time specified in the target time register (high/low). This bit can be cleared by reading the ETH_PTPTSSR register
8: 7	Reserved	-	-	-
6	MMCTS	RW	0	MMC transmit status This bit is pulled up when an arbitrary interrupt occurs in the ETH_MMCTIR register, and is cleared when all bits in the interrupt register ETH_MMCTIR are cleared
5	MMCRS	RW	0	MMC receive status This bit is pulled up when an arbitrary interrupt occurs in the ETH_MMCRIR register, and is cleared when all bits in the interrupt register ETH_MMCRIR are cleared
4	MMCS	RW	0	MMC status Set 1 when either bit of MMCTS or MMCRS is high and 0 when both bits are low
3	PMTS	RW	0	PMT status When a magic packet or a wake-up LAN frame is received in power-down mode, this bit is pulled up. When bit5 and 6 of the ETH_MACPMTCSR register are read, this bit is cleared
2: 0	Reserved	-	-	-

33.7.12 EthernetMACInterrupt Mask Register (ETH_MACIMR)

Address offset: 0x003C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	TSTIM	Res.	Res.	Res.	Res.	Res.	PMTIM	Res.	Res.	Res.
-	-	-	-	-	-	RW	-	-	-	-	-	RW	-	-	-

Bit	Name	R/W	Reset Value	Function
30:10	Reserved	-	-	-
9	TSTIM	RW	0	Timestamp triggers interrupt masking Time stamp trigger interrupt mask If this position is 1, generating a timestamp interrupt is disabled
8: 7	Reserved	-	-	-
3	PMTIM	RW	0	PMT interrupt mask If this position 1,the PMT state position1 2 inthe ETH_MACSRis disabled from triggering an interrupt signal.
2: 0	Reserved	-	-	-

33.7.13 EthernetMACAddress0High Order Register (ETH_MACA0HR)

Address offset: 0x0040

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MO	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			Res.
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA0H															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	MO			Always 1
30:16	Reserved	-	-	-
15: 0	MACA0H	RW	FFFF	Upper 16 bits of MAC address 0 (MAC address0 high [47: 32]) Contains the upper 16 bits of 6-byte MAC address 0 for MAC address filtering of received frames and inserting MAC addresses when transmitting flow control (pause) frames

33.7.14 EthernetMACAddress0Low Order Register (ETH_MACA0LR)

Address offset: 0x0040

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MacAOL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACAOL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	MACAOL [31: 0]	RW	FFFFFF	Lower 32 bits of MAC address 0 (MAC address0 low [31: 0]) Lower 32 bits containing 6-byte MAC address 0 for MAC address filtering of received frames and inserting MAC addresses when sending flow control frames

33.7.15 EthernetMACAddress1High Order Register (ETH_MACA1HR)

Address offset: 0x0048

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA.	MBC.						Res.	Res.	Res.	Res.	Res.			Res.
RW	RW	RW	RW	RW	RW	RW	RW	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA1H															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	AE	RW	0	Address enable (Address enable) 1: Address perfect filtering by MAC address 1 0: The address filter does not perform address filtering
30	SA	RW	0	Source address (Source address) 1: MAC address 1 [47: 0] SA field comparison for received frame, 0: MAC Address 1 [47: 0] DA field comparison for received frame
29:24	MBC	RW	0	Mask byte control (Mask byte control) Mask control when comparing each MAC address byte. When set to 1, MAC does not compare the DA/SA field in address 1 Bit 29: Mask ETH_MACA1HR [15: 8] field Bit 28: Mask ETH_MACA1HR [15: 8] field Bit 27: Mask ETH_MACA1LR [31: 24] field … Bit 24: Mask ETH_MACA1LR [7: 0] field
23:16	Reserved	-	-	-
15: 0	MACA1H	RW	FFFF	The upper 16 bits of MAC address 1 (MAC address 1 high [47: 32])

				Contains the upper 16 bits of the 6-byte MAC address 1 for MAC address filtering of received frames and inserting MAC addresses when sending flow control (pause) frames

33.7.16 EthernetMACAddress1Low Order Register (ETH_MACA1LR)

Address offset: 0x004C

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MACA1L [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA1L [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	MACA1L	RW	FFFFFF	Lower 32 bits of MAC address 1 (MAC address1 low [31: 0]) Contains the lower 32 bits of 6-byte MAC address 1 for MAC address filtering of received frames and inserting MAC addresses when sending flow control frames

33.7.17 EthernetMACAddress2High Order Register (ETH_MACA2HR)

Address offset: 0x0050

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA.	MBC.						Res.	Res.	Res.	Res.	Res.			Res.
RW	RW	RW	RW	RW	RW	RW	RW	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA2H															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	AE	RW	0	Address enable (Address enable) 1: Address perfect filtering by MAC address 2 0: The address filter does not perform address filtering
30	SA	RW	0	Source address (Source address) 1: MAC address 2 [47: 0] for SA field comparison of received frame, 0: MAC Address 2 [47: 0] DA field comparison for received frame
29:24	MBC	RW	0	Mask byte control (Mask byte control) Mask control when comparing each MAC address byte. When set to 1, MAC does not compare the DA/SA field in address 2 Bit 29: Mask ETH_MACA2HR [15: 8] field Bit 28: Mask ETH_MACA2HR [15: 8] field Bit 27: Mask ETH_MACA2LR [31: 24] field … Bit 24: Mask ETH_MACA2LR [7: 0] field
23:16	Reserved	-	-	-
15: 0	MACA2H	RW	FFFF	The upper 16 bits of MAC address 2 (MAC address 2 high [47: 32]) Contains the upper 16 bits of a 6-byte MAC address 2 for MAC address filtering of received frames and inserting MAC addresses when sending flow control (pause) frames

33.7.18 EthernetMACAddress2Low Order Register (ETH_MACA2LR)

Address offset: 0x0054

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MACA2L [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA2L [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	MACA2L	RW	FFFFFF	Lower 32 bits of MAC address2 low [31: 0] Contains the lower 32 bits of a 6-byte MAC address 2 for MAC address filtering of received frames and inserting MAC addresses when transmitting flow control frames

33.7.19 EthernetMACAddress3High Order Register (ETH_MACA3HR)

Address offset: 0x0058

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA.	MBC.						Res.	Res.	Res.	Res.	Res.			Res.
RW	RW	RW	RW	RW	RW	RW	RW	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA3H															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	AE	RW	0	Address enable (Address enable) 1: Address perfect filtering by MAC address 3 0: The address filter does not perform address filtering
30	SA	RW	0	Source address (Source address) 1: MAC address 3 [47: 0] for SA field comparison of received frame, 0: MAC address 3 [47: 0] DA field comparison for received frame
29:24	MBC	RW	0	Mask byte control (Mask byte control) Mask control when comparing each MAC address byte. When set to 1, MAC does not compare the DA/SA field in address 3 Bit 29: Mask ETH_MACA3HR [15: 8] field Bit 28: Mask ETH_MACA3HR [15: 8] field Bit 27: Mask ETH_MACA3LR [31: 24] field … Bit 24: Mask ETH_MACA3LR [7: 0] field
23:16	Reserved	-	-	-
15: 0	MACA3H	RW	FFFF	The upper 16 bits of MAC address 3 (MAC address 3 high [47: 32]) Contains the upper 16 bits of a 6-byte MAC address 3 for MAC address filtering of received frames and inserting MAC addresses when sending flow control (pause) frames

33.7.20 EthernetMACAddress3Low Order Register (ETH_MACA3LR)

Address offset: 0x005C

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MACA3L [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MACA3L [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	MACA3L	RW	FFFFFF	Lower 32 bits of MAC address 3 (MAC address3 low [31: 0] Contains the lower 32 bits of 6 byte MAC address 3 for MAC address filtering of received frames and inserting MAC addresses when transmitting flow control frames

33.7.21 EthernetMMCControl Register (ETH_MMCCR)

Address offset: 0x0100

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res			Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res	Res.	Res.	Res.	UCDBC	Res.	Res.	CNTPRSTL	CNTPRST.	MCF	ROR	CSR.	CR.
-	-	-	-	-	-	-	RW	-	-	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	UCDBC	RW	0	Counting MMC dropped broadcast frames 1: Update counter 0: Not updating counter
7: 6	Reserved	-	-	-
5	CNTPRSTL	RW	0	Full./Half Full Preset 1: When CNTPRST is 1, the frame counter is preset to 0xFFFF_FFF0 0: When CNTPRST is 1, the frame counter is preset to 0x7FFF_FFF0
4	CNTPRST.	RW	0	Counter preset When set to 1, all counters are initialized or preset to a full/half full value, depending on CNTPRSTLVL, and self-clears after one cycle
3	MCF	RW	0	MMC counter freeze 1: Freeze all MMC counters and keep the current value. (When this bit is not cleared, the counter is not updated with received and transmitted frames.) If any MMC counter is read when the ROR bit is 1, the counter is cleared)
2	ROR	RW	0	Read clear (Reset on read) 1: The MMC counter will be cleared after the read operation, and the counter will be cleared when the least significant byte [7: 0] is read
1	CSR	RW	0	Counter stops flipping Counter stop rollover 1: The counter does not flip to 0 after reaching the maximum value
0	CR.	RW	0	Counter reset 1: All counters are reset, and this bit clears itself after one cycle

33.7.22 Ethernet MMC receiveinterrupt register (ETH_MMCRIR)

Address offset: 0x0104

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res	Res	RGUFS	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RC_R	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAES	RFCES	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	RC_R	RC_R	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	RGUFS	RC_R	0	Received good unicast frame status Received Good Unicast Frames Status When the count reaches half of the maximum value when a good unicast frame is received, this bit is pulled up
16: 7	Reserved	-	-	-
6	RFAES	RC_R	0	Received alignment error frame status Received frames alignment error status This bit is pulled up when the count reaches half of the maximum value when an alignment error frame is received
5	RFCES	RC_R	0	Received frames CRC error status This bit is pulled up when the count reaches half of the maximum value when a CRC error frame is received
4: 0	Reserved	-	-	-

33.7.23 EthernetMMCSend Interrupt Register (ETH_MMCTIR)

Address offset: 0x0108

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	TGFS	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	RC_R	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGFMSCS.	TGFSCS.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RC_R	RC_R	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	-
21	TGFS	RC_R	0	Good frame status sent (Transmitted good frames status) This bit is pulled up when the count reaches half of the maximum when a good frame is transmitted
20:16	Reserved	-	-	-
15	TGFMSCS.	RC_R	0	Status of a good frame sent after multiple collisions (Transmitted good frames more single collision status) When a good frame is sent after multiple collisions, the count reaches half of the maximum value, the bit is pulled up
14	TGFSCS	RC_R	0	The state of a good frame sent after a single collision (Transmitted good frames single collision status) This bit is pulled up when a good frame is sent after a single collision and the count reaches half of the maximum value
13: 0	Reserved	-	-	-

33.7.24 Ethernet MMC receiveinterrupt mask register (ETH_MMCRIMR)

Address offset: 0x010C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res	Res	RGUFM	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RFAEM	RFCEM	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	RW	RW	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	-
17	RGUFM	RW	0	Received good unicast frame masking Received Good Unicast Frames mask 1: When the unicast frame is received, no interrupt is generated when the count reaches half of the maximum value
16: 7	Reserved	-	-	-
6	RFAEM	RW	0	Received Alignment Error Frame Mask Received frames alignment error mask 1: No interrupt is generated when the count reaches half of the maximum value when an alignment error frame is received
5	RFCEM	RW	0	Received frames CRC error mask 1: When CRC error frame is received, no interrupt is generated when the count reaches half of the maximum value
4: 0	Reserved	-	-	-

33.7.25 Ethernet MMC SendInterrupt Mask Register (ETH_MMCTIMR)

Address offset: 0x0110

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	TGFM	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	RW	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGFMSCM.	TGFSCM.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	-
21	TGFM	RW	0	Good frame mask sent (Transmitted good frames mask) 1: When a good frame is transmitted, no interrupt is generated when the count reaches half of the maximum value
20:16	Reserved	-	-	-
15	TGFMSCM.	RW	0	Mask of transmitting good frames after multiple collisions (Transmitted good frames more single collision mask) 1: When a good frame is sent after multiple collisions and the count reaches half of the maximum value, no interrupt is generated

14	TGFSCM	RW	0	Masking of a good frame sent after a single collision (Transmitted good frames single collision mask) 1: When a good frame is sent after a single collision and the count reaches half of the maximum value, no interrupt is generated
13: 0	Reserved	-	-	-

33.7.26 Ethernet MMCsends good frame counter after a single collision (ETH_MMCTGFSCCR)

Address offset: 0x014C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TGFSCC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGFSCC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TGFSCC	RW	0	Send good frame count after a single collision (Transmitted good frames single collision counter/Transmitted good frames after a single collision counter)

33.7.27 Ethernet MMCsends good frame count register after multiple collisions (ETH_MMCTGFMSCCR)

Address offset: 0x0150

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TGFMSCC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGFMSCC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TGFMSCC	RW	0	Send good frame count after multiple collisions

33.7.28 Ethernet MMC send goodframe count register (ETH_MMCTGFR)

Address offset: 0x0168

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TGFC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TGFC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TGFC	RW	0	Send good frame count (Transmitted good frames counterr)

33.7.29 EthernetMMC CRCError Receive Frame Count Register (ETH_MMCRFCECR)

Address offset: 0x0168

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFCEC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFCEC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	RFCEC	RW	0	Received CRC Error Frame Count (Received frames CRC error counter Received frames with CRC error counter

33.7.30 EthernetMMCAlignment Error Received Frame Count Register (ETH_MMCRFAECR)

Address offset: 0x0198

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFAEC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFAEC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	RFAEC	RW	0	Received Alignment Error Frame Count Received frames alignment error counter Received frames with alignment error counter

33.7.31 EthernetMMCAlignment Error Received Frame Count Register (ETH_MMCRFAECR)

Address offset: 0x01C4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RGUFC [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGUFC [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	RGUFC	RW	0	Received good unicast frame count (Received good unicast frames counter)

33.7.32 EthernetPTPTimestamp Control Register (ETH_PTPTSCR)

Address offset: 0x0700

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res			Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSARU	TSITE	TSSTU	TSSTI	TSFCU	TSE
-	-	-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	-	-
5	TSARU	RW	0	Timestamp wig register update (Time stamp add register update) 1: The data in the timestamp addend register is updated to the PTP module for fine correction. This bit is cleared when the update is complete. This register must be read to 0 before setting it.
4	TSITE	RW	0	Timestamp Interrupt Trigger Enable (Time stamp interrupt trigger enable) 1: Generate a timestamp interrupt when the system time is larger than the value in the target time register. This bit is cleared when the generation of a timestamp triggers an interrupt
3	TSSTU	RW	0	Timestamp System Time Update Time stamp system time update 1: The system time is updated to the value in the timestamp high update and low update registers. This register must be read to 0 before setting it. This bit is cleared when the hardware update is complete
2	TSSTI	RW	0	Timestamp system time initialization Time stamp system time initialize 1: System time is initialized to the values in the timestamp high update and low update registers. This register must be read to 0 before setting it. When initialization is finished, this bit is cleared.
1	TSFCU	RW	0	Timestamp fine and coarse updates Time stamp fine or coarse update 1: System timestamp updated by fine method 0: System timestamp updated by coarse method

0	TSE	RW	0	Timestamp enable Time stamp enable 1: Enable timestamps in transmit and receive frames 0: The timestamp function is suspended and the timestamp does not accumulate in the transmit and receive frames. Because the held system time has been paused, after setting this bit to 1, you need to initialize the timestamp feature (system time)

33.7.33 EthernetPTPSubsecond Increment Register (ETH_PTPSSIR)

Address offset: 0x0700

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res			Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STSSI							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	STSSI	RW	0	System time sub-second increment (System time subsecond increase) The register is programmed to add a sub-second value to the system time each update cycle. For example: To achieve 20ns accuracy, this value = 20/0. 467 is approximately equal to 43 (0x2A)

33.7.34 EthernetPTPTimestamp High Bit Register (ETH_PTPTSHR)

Address offset: 0x0708

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STS [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	STS	RW	0	System time seconds (System time second) This register value shows the system clock second value currently held by the controller

33.7.35 EthernetPTPTimestamp Low Bit Register (ETH_PTPTSLR)

Address offset: 0x070C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STPNS	STSS [30:16]														
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STSS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	STPNS	RW	0	System positive or negative time indication (System time positive or negative sign) 1: Time is negative 0: Time is positive. Since time should always be positive, this bit is typically 0
30: 0	STSS	RW	0	System time subsecond System time subseconds This field value represents sub-second time with an accuracy of 0.46 ns

33.7.36 EthernetPTPTimestamp Update Register (ETH_PTPTSHUR)

Address offset: 0x0710

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSUS [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSUS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TSUS	RW	0	Timestamp update seconds Time stamp update second The value of this field indicates the number of seconds that the system was initialized or updated

33.7.37 EthernetPTPTimestamp Low Bit Update Register (ETH_PTPTSLUR)

Address offset: 0x0710

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSUPNS	TSUSS [30:16]														
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSUSS [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	TSUPNS	RW	0	Timestamp updates indication of positive and negative values Time stamp update positive or negative sign 1: Time is negative 0: Time is positive When TSSTI is set to 1, this bit should be cleared. If this bit is set to 1 at this time, the system time will be subtracted from the value of the timestamp update register, and vice versa, the system time will be added to the value of the timestamp update register
30: 0	TSUSS	RW	0	Timestamp update sub-second Time stamp update subseconds The value of this field indicates the sub-second time when the system was initialized or updated. The accuracy of this value is 0.46 ns (0x0000_0001 is 0.46 ns)

33.7.38 EthernetPTPTimestamp Addition Register (ETH_PTPTSAR)

Address offset: 0x0718

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSA [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSA [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TSA	RW	0	Timestamp Addition (Time stamp add) This register indicates that a 32-bit value is added to the count register for synchronization of events

33.7.39 EthernetPTP TimestampTarget Time High Bit Register (ETH_PTPTTHR)

Address offset: 0x071C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TTSH [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTSH [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TTSH	RW	0	Target timestamp high (Target time stamp high) This register stores second event values. When the value of the timestamp equals or exceeds the target event value, the MAC generates an interrupt

33.7.40 EthernetPTPTimestamp Target Time Low Bit Register (ETH_PTPTTLR)

Address offset: 0x0720

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TTSL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTSL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TTSL	RW	0	Target timestamp low Target time stamp low This register stores sub-second event values. When the value of the timestamp equals or exceeds the target event value, the MAC generates an interrupt

33.7.41 EthernetDMA Bus ModeRegister (ETH_DMABMR)

Address offset: 0x1000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RIB	Res.	Res.	Res.	TXPR	MB	AAB.	FPM	USP	RDP						FB
RW	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PM		PBL						Res.	DSL					DA	SR
RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	RIB	RW	0	INCRx Burst Reconstruction of 1: When the AHB main interface gets EBT (Retry, Split, Losing bus grant. The AHB master interface can reconstruct the pending beats of the burst transmission initiated by INCRx, the AHB master interface reconstructs the beats using the specified burst combination of INCRx and SINGLE, and by default the AHB master interface reconstructs the pending beats of the EBT with an unspecified (INCR) burst
30:28	Reserved	-	-	-
27	TXPR	RW	0	Transmitter priority 1: When set, this bit indicates that the priority of the transmit DMA is higher than that of the receive DMA during arbitration of the system side bus
26	MB	RW	0	Mixed burst 1: When FB is 0, the AHB main interface starts all bursts with length greater than 16 with INCR (undefined burst). While for burst lengths of 16 and below, it reverts to fixed burst transmission (INCRx and SINGLE)
25	AAB.	RW	0	Address alignment (Address-aligned beats) 1: When FB is equal to 1, the AHB interface will generate all bursts aligned with the start address of the LS bit. If FB is equal to 0, the first burst is una-ligned, and the subsequent bursts are aligned with the address
24	FPM	RW	0	4x PBL mode (4xPBL mode) 1: This bit multiplies the programmed PBL value (bits [22: 17] and bits [13: 8]) by a factor of four, therefore, the data transferred by the DMA is up to 4, 8, 16, 32, 64 and 128 beats depending on the PBL value.
23	USP	RW	0	Use separate PBL value 1: Configure RxDMA to use the value of RDP as PBL, while the configuration value of actual PBL is only used for TxDMA. 0: Value of PBL for RxDMA and TxDMA
22:17	RDP	RW	0	Burst length of RxDMA (RxDMA PBL) These bits represent the maximum number of beats to be transmitted in an RxDMA transaction. This is the maximum value used in a single read/write operation. Each time a burst transmission is initiated on the host bus, RxDMA always tries to burst in the manner specified in the RDP. The RDP can be

				programmed with allowed values 1, 2, 4, 8, 16, and 32, other values are invalid. Note: Only valid if USP is 1
16	FB	RW	0	Fixed burst 1: AHB interface uses fixed burst transmission, including SINGLE, INCR4, INCR8 or INCR16 0: AHB interface uses SINGLE, INCR burst mode
15:14	PM	RW	0	Rx Tx priority ratio 00: 1: 1 01: 2: 1 10: 3: 1 11: 4: 1 Note: Only valid when DA bit is cleared
13: 8	PBL	RW	0	Burst length (Programmable burst length) These bits represent the maximum number of beats to be transmitted in a DMA transaction. This is the maximum value used in a single read/write operation. Each time a burst transmission is initiated on the host bus, RxDMA always tries to burst in the manner specified in the RDP. The RDP can be programmed with allowed values 1, 2, 4, 8, 16, and 32, other values are invalid. Note: Only valid if USP is 1 The initialization of PBL value needs to pay attention to the following points: 1. The maximum value of PBL depends on the size of TxFIFO and RxFIFO 2. FIFO has a limitation, that is, the maximum beat supported is half the FIFO depth 3. If PBL is used to receive and transmit DMA, the minimum FIFO depth needs to be considered 4. Do not program PBL values outside the range, otherwise the system may not function properly
7	Reserved	-	-	-
6: 2	DSL	RW	0	Descriptor skip length (Descriptor skip length) This bit segment defines the number of words skipped between two non-chained descriptors. Address skip starts at the end of the current descriptor to the beginning of the next descriptor. If the value of DSL is equal to 0, in ring mode, the descriptor table is treated as a connection by the DMA
1	DA	RW	0	DMA Arbitration 0: Rx: Tx Priority Loop in bits [15:14] 1: RX has higher priority than Tx
0	SR	RW	0	Soft reset Software reset 1: The DMA controller of the MAC resets the internal registers and logic of all MAC sublayers. When the reset is completed, it automatically clears. The median value in this bit is read to 0 before any register is reprogrammed.

33.7.42 EthernetDMASend Query Request Register (ETH_DMATPDR)

Address offset: 0x1004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TPD [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPD [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	TPD	RW	0	Send a polling request (Transmit poll demand) When any value is written to this register, DMA reads the address pointed to by the current descriptor (ETH_DMACHTDR register). If the descriptor is CPU-occupied, the sender returns a pending state and bit2 of the ETH_DMASR register is set to 1. If the descriptor is not CPU-occupied, the send resumes

33.7.43 EthernetDMAReceive Query Request Register (ETH_DMARPDR)

Address offset: 0x1008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPD [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPD [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	RPD	RW	0	Receive a polling request (Receive poll demand) When any value is written to this register, DMA reads the address pointed to by the current descriptor (ETH_DMACHRDR register). If the descriptor is CPU-occupied, the receiver returns a pending state and bit7 of the ETH_DMASR register is set to 1. If the descriptor is not CPU-occupied, the receive resumes

33.7.44 Ethernet DMA receivedescriptor list address register (ETH_DMARDLAR)

Address offset: 0x100C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	SRL	RW	0	Start of receive list (Start of received list) This register contains the base address of the first descriptor in the list of received descriptors. Its LSB (bit [1/2/3: 0] corresponds to 32/64/128-bit bus width) is ignored internally and considered to be 0 by DMA. Therefore LSB is read-only.

33.7.45 EthernetDMASend Descriptor List Address Register (ETH_DMATDLAR)

Address offset: 0x1010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STL [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STL [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	STL	RW	0	Start of Send List Start of transmit list This register contains the base address of the first descriptor in the transmit descriptor list. Its LSB (bit [1/2/3: 0] corresponds to 32/64/128-bit bus width) is ignored internally and considered to be 0 by DMA. Therefore LSB is read-only.

33.7.46 EthernetDMAStatus Register (ETH_DMASR)

Address offset: 0x1014

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	TSTS	PMTS	MMCS	Res.	EBS			TPS			RPS			NIS
-	-	R	R	R	-	R	R	R	R	R	R	R	R	R	RC_W1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIS.	ERS	FBES	Res.	Res.	ETS	RWTS	RPSS	RBUS	RS	TUS	ROS	TJTS	TBUS	TPSS.	TS.
RC_W1	RC_W1	RC_W1	-	-	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29	TSTS	R	0	Timestamp trigger state Time stamp trigger status This bit indicates that the timestamp generation module of the MAC controller generated an interrupt. The software needs to read the MAC controller status register to clear its source (bit9), and then clear the bit.
28	PMTS	R	0	PMT Interrupt This bit indicates that the PMT module of the MAC controller generated an interrupt. The software must read the corresponding register in the MAC controller to get the exact cause of the interrupt and clear its source to reset the bit to 0.
27	MMCS	R	0	MMC Interrupt This bit indicates that an interrupt has been generated by the MMC module of the MAC controller. The software must read the corresponding register in the MAC controller to get the exact cause of the interrupt and clear its source to reset the bit to 0.
26	Reserved	-	-	-
25:23	EBS	R	0	Error status Error bits status This bit indicates the type of error that causes the bus error. Valid when the fatal bus error bit ETH_DMASR register [13]) is set to 1. This bit does not generate an interrupt. Bit23: 1: TxDMA error sending data 0: Error receiving data at RxDMA Bit24: 1: Error during read operation 0: Error during write operation Bit25: 1: Error accessing descriptor 0: Error accessing data cache area
22:20	TPS	R	0	Send process status (Transmit process state) This bit segment represents the state machine that sends the DMA and does not generate interrupts 000: stopped state, issue reset or stop sending commands 001: ruuning state, get send descriptor 010: running state, waiting for state information 011: running state, read data from memory and queue it for transfer to buffer (Tx FIFO) 100,101: Reserved 110: suspended state, the send descriptor is invalid or the send cache area is underflow 111: running state: transmission descriptor closed
19:17	RPS	R	0	Receive process status (Receive process state) This bit segment represents the state machine receiving the DMA and does not generate an interrupt 000: stopped state, issue reset or stop receiving commands 001: ruuning state, get receive descriptor 010: Reserved 011: running state, waiting to receive data packet 100: suspended state, receive descriptor is invalid 101: running state: transmission descriptor closed 110: Reserved 111: running state, sending data from Rx FIFO to system memory
16	NIS	RC_W1	0	Regular interruptions and (Normal interrupt summary) This bit segment represents the state machine receiving the DMA and does not generate an interrupt This bit is the OR logic of the following interrupts: -ETH_DMASR [0]: Send interrupt -ETH_DMASR [2]: Send cache invalid -ETH_DMASR [6]: Reception interrupt -ETH_DMASR [14]: Early reception interrupt Only unmasked interrupts can affect regular interrupts and This is a sticky bit that must be cleared each time the corresponding bit causing the NIS setting is cleared (by writing a 1 to the bit)
15	AIS.	RC_W1	0	Unconventional interruptions and Abnormal interrupt summary

				This bit segment represents the state machine receiving the DMA and does not generate an interrupt This bit is the OR logic of the following interrupts: -ETH_DMASR [1]: Send process stopped -ETH_DMASR [3]: Send jabber timer overflow -ETH_DMASR [4]: received FIFO overflow -ETH_DMASR [5]: Send underflow -ETH_DMASR [7]: Receive cache invalid -ETH_DMASR [8]: Receive process stops -ETH_DMASR [9]: WDT overflow -ETH_DMASR [10]: Early transmission interrupt -ETH_DMASR [13]: Fatal bus error Only unmasked interrupts can affect regular interrupts and This is a sticky bit that must be cleared each time the corresponding bit causing the NIS setting is cleared (by writing a 1 to the bit)
14	ERS	RC_W1	0	Early receive status Early receive status This bit indicates that the DMA has filled the first data buffer area of the packet. Receive interrupt ETH_DMASR [6] automatically clears this bit)
13	FBES	RC_W1	0	Fatal bus error status (Fatal bus error status) This bit indicates that a fatal bus error has occurred. When set to 1, the corresponding DMA engine will disable all bus access
12:11	Re-served	-	-	-
10	ETS	RC_W1	0	Early send status Early transmit status This bit indicates that the frame to be transmitted has been fully transmitted to the transmit FIFO.
9	RWTS	RC_W1	0	Receive watchdog timer overflow (Receive watchdog timeout status) This bit indicates that the received frame length is greater than 2048 bytes
8	RPSS	RC_W1	0	Receive watchdog timer overflow (Receive watchdog timeout status) This bit indicates that the received frame length is greater than 2048 bytes
7	RBUS	RC_W1	0	Receive buffer area invalid status Receive buffer unavailable status This bit indicates that the next descriptor in the receive list is occupied by CPU and cannot be requested by DMA. The receiving process is suspended. In order to resume processing the receive descriptor, the CPU needs to change the ownership of the descriptor and issue a command to receive the polling request. If no polling command is issued, the receiving process resumes when the next identified incoming frame is received. This position 1 is only if the previous receive descriptor is occupied by the DMA
6	RS	RC_W1	0	Received status (Receive status) This bit indicates that frame reception is complete. Specific frame status information has been published in the descriptor. The receiver is still operational
5	TUS	RC_W1	0	Send underflow status (Transmit underflow status) This bit indicates that a transmit buffer underflow occurred during transmission. The transmitter will enter a suspended state and TDES0 [1] will be set to 1
4	ROS	RC_W1	0	Receive overflow status (Receive overflow status) This bit indicates that a receive buffer overflow occurred during receive. If a partial frame is transferred to the application, the overflow state is set in RDES0 [11].
3	TJTS	RC_W1	0	Transmit jabber timeout status This bit indicates that the send jabber timer overflows, which means that the transmitter is overactive. The sending process is aborted and enters the stopped state. TDES0 [14] Will set 1
2	TBUS	RC_W1	0	Send buffer area invalid status Transmit buffer unavailable status This bit indicates that the next descriptor in the send list is occupied by CPU and cannot be requested by DMA. The receiving process is suspended. In order to resume processing the receive descriptor, the CPU needs to change the ownership of the descriptor and issue a command to send a polling request
1	TPSS.	RC_W1	0	Send process stop status (Transmit process stopped status) This bit indicates that the sending process enters stopped
0	TS	RC_W1	0	Send Status (Transmit status) This bit indicates that the frame transmission is complete and TDES1 [31] in the first descriptor is set to 1

33.7.47 EthernetDMAoperating mode register (ETH_DMAOMR)

Address offset: 0x1018

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	DTCEDF.	RSF	DFRF	Res.	Res.	TSF	FTF	Res.	Res.	Res.	TTC[2]
-	-	-	-	-	RW	RW	RW	-	-	RW	RW	-	-	-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTC [1: 0]		ST	Res.	Res.	Res.	Res.	Res.	FEF	FUGF	Res.	RTC		OSF	SR	Res.
RW	RW	RW	-	-	-	-	-	RW	RW	-	RW	RW	RW	RW	-

Bit	Name	R/W	Reset Value	Function
31: 27	Reserved	-	-	-
26	DTCEDF	RW	0	Dropping of TCP/IP checksum error frames disable 1: The controller does not discard frames where only the receiving checksum load shedding engine detects an error. Such a frame does not have any errors (including FCS errors) in the Ethernet frame received by the MAC, but only in the encapsulated payload. 0: All error frame are discarded if FEF is 0
25	RSF	RW	0	Store and forward mode Receive store and forward 1: When the frame is fully written to the Rx FIFO, start reading the frame, ignoring the RTC bit. 0: The Rx FIFO operates in pass-through mode, subject to a threshold specified by the RTC bit.
24	DFRF	RW	0	Prohibit clearing of received frames (Disable flushing of received frames) 1: Since the receive descriptor/buffer is not available, RxDMA does not clear any frames like it would normally clear this bit
23: 22	Reserved	-	-	-
21	TSF	RW	0	Send-store-forward mode Transmit store and forward 1: When the frame is fully written to the Tx FIFO, start sending the frame, ignoring the TTC bit. 0: The Tx FIFO operates in pass-through mode, subject to a threshold specified by the RTTC bit. Note: This bit can be changed only when transmission is stopped
20	FTF	RW	0	Clear send frame Flush transmit FIFO 1: The transmit FIFO controller logic is reset to the default value and all data in the Tx FIFO is cleared. When the emptying action is completed, this bit is automatically cleared to 0. Cannot be written until the bit has not been cleared.
19: 17	Reserved	-	-	-
16: 14	TTC	RW	0	Transmission threshold control Transmit threshold control This bit segment controls the threshold point at which the FIFO is transmitted. When the size of the buffer frame in the Tx FIFO is greater than the threshold value, the transmission process is started. In addition, if it is a complete frame, transmission will be turned on even if the size is less than the threshold value. This bit segment is valid when the TSF is cleared 000: 64 001: 128 010: 192 011: 256 100: 40 101: 32 101: 24 101: 16
13	ST	RW	0	Start/stop transmission 1: The transmission enters the running state, and the DMA checks the frame to be transmitted in the transmission list at the current position. A descriptor acquisition attempt is made either at the position of the current list (the base address of the transmission list is set by the ETH_DMATDLAR register) or from the position held when the transmission was previously stopped. If no descriptor is owned by the DMA, the transmission goes into a suspended state and the transmission cache invalid position 1 (ETH_DMASR [2]). The start send command is only valid when the send is stopped. If the command

				is issued before the DMA ETH_DMATDLAR register is set, the behavior of DNA is unpredictable. 0: The transmission process enters the stopped state after the transmission of the current frame is completed. The next descriptor position in the transmit list is saved and becomes the current position when transmission is re-started. The stop sending command is valid only when the current frame is finished sending or when sending is in a suspended state
12: 8	Reserved	-	-	-
7	FEF	RW	0	Forward error frame (Forward error frames) 1: All frames except short error frames are forwarded to DMA 0: Rx FIFO discards the frame corresponding to the error state (CRC error, collision error, huge frame, watchdog overflow, cache overflow). However, if the start byte pointer of the frame has been sent to the read controller side (in threshold mode), then the frame is not discarded. When the frame start byte is not sent to the ARI bus, the error frame is discarded
6	FUGF	RW	0	Forward good frames of small size Forward undersized good frames 1: Rx FIFO forwards frames of small size (no errors but less than 64 bytes in length, including padding and crc) 0: Frames less than 64 bytes are discarded unless such frames have already been transmitted due to low reception threshold.
5	Reserved	-	-	-
4: 3	RTC	RW	0	Receive threshold control (Receive threshold control) This bit segment controls the threshold point at which the FIFO is received. When the size of the buffer frame in the Rx FIFO is greater than the threshold, the transmission of the frame to the memory is started. In addition, if it is a complete frame, transmission will be turned on even if the size is less than the threshold value. This bit segment is valid when the RSF is cleared 00: 64 01: 32 10: 96 11: 128
2	OSF	RW	0	Second frame continuous mode (Operate on second frame) 1: DMA transmits second frame before receiving status information of first frame
1	SR	RW	0	Start/stop receive 1: The receiver enters the running state, and the DMA checks the frame to be transmitted in the transmission list at the current position. The descriptor acquisition attempt is made at the position of the current list (the base address ETH_DMATDLAR register setting of the received list) or from the position held when the transmission was previously stopped. If no descriptor is owned by the DMA, the reception goes into a suspended state and the reception buffer invalid bit is set to 1 (ETH_DMASR [7]). The start receive command is only valid when the receive stops. If the command is issued before the DMA ETH_DMATDLAR register is set, the behavior of DNA is unpredictable. 0: The transmission process enters the stopped state after the reception of the current frame is completed. The next descriptor position in the reception list is saved and becomes the current position when reception is restarted. The stop receive command is only valid if the receiving process enters the suspended or running state
0	Reserved	-	-	-

33.7.48 EthernetDMAinterrupt enable register (ETH_DMAIER)

Address offset: 0x101C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NISE
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AISE	ERIE	FBEIE	.		ETIE.	RWTIE.	RPSIE	RBUIE	RIE	TUIE.	ROIE	TJTIE	TBUIE	TPSIE	TIE.
RW	RW	RW	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	NISE	RW	0	General interrupts and enables (Normal interrupt summary enable) 1: Regular interrupt enabled 0: Regular interrupt disabled

				-ETH_DMASR [0]: Send interrupt -ETH_DMASR [2]: Invalid transmit buffer -ETH_DMASR [6]: Reception interrupt -ETH_DMASR [14]: Early reception interrupt
15	AISE	RW	-	Unconventional interruptions and enablations (Abnormal interrupt summary enable) 1: Unconventional interrupt enabled 0: Unconventional interrupt disabled -ETH_DMASR [1]: Send process stopped -ETH_DMASR [3]: Send jabber timer overflow -ETH_DMASR [4]: Receive overflow -ETH_DMASR [5]: Send underflow -ETH_DMASR [7]: Receive buffer area invalid -ETH_DMASR [8]: Receive process stops -ETH_DMASR [9]: Received watchdog overflow -ETH_DMASR [10]: Early transmission interrupt -ETH_DMASR [13]: Fatal bus error
14	ERIE	RW	0	Early receive interrupt enable (Early receive interrupt enable) 1: Early receive interrupt enabled when non-conventional interrupt and are enabled 0: Early reception interrupt disabled
13	FBEIE	RW	0	Fatal Bus Error Interrupt Enable (Fatal bus error interrupt enable) 1: Fatal bus error interrupt is enabled when regular interrupt and are enabled 0: Fatal bus error interrupt disabled
12: 11	Reserved	-	-	-
10	ETIE.	RW	0	Early Send Interrupt Enable Early transmit interrupt enable 1: Early transmit interrupt is enabled when non-conventional interrupt and are enabled 0: Early transmission interrupt is disabled
9	RWTIE	RW	0	Receive Watchdog Overflow Interrupt Interrupt Enable (receive watchdog timeout interrupt enable) 1: Receive watchdog overflow interrupt is enabled when non-conventional interrupt and are enabled 0: Receiving watchdog overflow interrupt disabled
8	RPSIE	RW	0	Receive Process Stop Interrupt Interrupt Enable (Receive process stopped interrupt enable) 1: Receive process stop interrupt enabled when non-conventional interrupt and are enabled 0: Receiving process stop interrupt is disabled
7	RBUIE	RW	0	Receive buffer invalid interrupt interrupt enable (Receive buffer unavailable interrupt enable) 1: Receive buffer invalid interrupt is enabled when unconventional interrupt and are enabled 0: Receive cache invalid interrupt disabled
6	RIE	RW	0	Receive interrupt enable (Receive interrupt enable) 1: Receive interrupt enabled when non-conventional interrupt and are enabled 0: Reception interrupt disabled
5	TUIE	RW	0	Underflow interrupt enable 1: Underflow interrupt is enabled when non-conventional interrupt and are enabled 0: Underflow interrupt disabled
4	ROIE	RW	0	Overflow interrupt enable 1: Overflow interrupt is enabled when non-conventional interrupt and are enabled 0: Overflow interrupt disabled
3	TJTIE	RW	0	Transmit jabber timeout interrupt enable 1: Send jabber overflow interrupt is enabled when unconventional interrupt and is enabled 0: Send jabber overflow interrupt is forbidden
2	TBUIE	RW	0	Send cache invalid interrupt enable (Transmit buffer unavailable interrupt enable) 1: Send buffer invalid interrupt is enabled when regular interrupt and are enabled 0: Send cache invalid interrupt disabled
1	TPSIE	RW	0	Send process stop interrupt enable (Transmit process stopped interrupt enable)

				1: Send process stop interrupt enabled when regular interrupt and are enabled 0: Send process stop interrupt disabled
0	TIE	RW	0	Send interrupt enable (Transmit interrupt enable) 1: Send interrupt enabled when non-conventional interrupt and are enabled 0: Transmission interrupt disabled

33.7.49 EthernetDMALost Frame and Buffer Overflow Count Register (ETH_DMAMFBOCR)

Address offset: 0x1020

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	OFOC.	MFA											OMFC
-	-	-	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MFC															
RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R

Bit	Name	R/W	Reset Value	Function
31: 29	Reserved	-	-	-
28	OFOC	RC_R	0	Overflow bit for FIFO overflow counter
27: 17	MFA	RC_R	0	Counter of frames missed by the application Missed frames by the application
16	OMFC	RC_R	-	Overflow bit of missed frame counter Overflow bit for missed frame counter
15: 0	MFC	RC_R		Controller missed frame counter Missed frames by the controller Counts frames missed by the controller due to unavailability of the CPU receive buffer. The counter is incremented every time the DMA drops an incoming frame.

33.7.50 EthernetDMAcurrent transmit descriptor register (ETH_DMACHTDR)

Address offset: 0x1048

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTDAP [31: 16]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTDAP [15: 0]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	R/W	Reset Value	Function
31: 0	HTDAP	RO	0	Master device sends descriptor address pointer Host transmit descriptor address pointer Clear on reset, this pointer is updated by DMA in operation

33.7.51 EthernetDMACurrent Receive Descriptor Register (ETH_DMACHRDR)

Address offset: 0x104C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HRDAP [31: 16]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRDAP [15: 0]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	R/W	Reset Value	Function
31: 0	HRDAP	RO	0	Master device receives descriptor address pointer (Host received descriptor address pointer) Clear on reset, this pointer is updated by DMA in operation

33.7.52 EthernetDMAcurrent send buffer address register (ETH_DMACHTBAR)

Address offset: 0x1050

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTBAP [31: 16]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTBAP [15: 0]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	R/W	Reset Value	Function
31: 0	HTBAP	RO	0	The master device sends a buffer address pointer Host transmit buffer address pointer Clear on reset, this pointer is updated by DMA in operation

33.7.53 EthernetDMAcurrent receive buffer address register (ETH_DMACHRBAR)

Address offset: 0x1054

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HRBAP [31: 16]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRBAP [15: 0]															
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	R/W	Reset Value	Function
31: 0	HRBAP	RO	0	The master device receives a buffer address pointer (Host received buffer address pointer) Clear on reset, this pointer is updated by DMA in operation

34. CAN bus controller

34.1 Introduction

The controller local area network (CAN) subsystem consists of two CAN modules and a shared message RAM memory. Please refer to section [Memory Mapping] for the base addresses of these three components.

The CAN module complies with ISO 11898-1: 2015 (CAN Protocol Specification Version 2.0 Parts A, B) and CAN FD Protocol Specification Version 1.0 (CAN with Flexible Data-Rate Specification Version 1.0).

FDCAN1 follows the CAN bus CAN2.0 (2.0 A, CAN2.0B) protocol.

FDCAN2 follows the CAN bus CAN2.0 (2.0 A, CAN2.0B) and CAN FD protocols.

The 1.6 KB message RAM memory can implement the functions of filter (Rx Filter), receive FIFO (Rx FIFO), receive buffer (Rx Buffer), send event FIFO (Tx Event FIFO) and send buffer (Tx Buffer). The message RAM is shared between the FDCAN1 and FDCAN2 modules.

34.1.1 Main features

- ISO 11898-1: 2015 standard is applicable
- CAN FD supports sending and receiving up to 64 data bytes
- Support CAN error logging
- Supports AUTOSAR and SAE J1939
- Support receive filtering function
- Two configurable receive FIFOs
- Individual signal indication when high priority messages are received
- Up to 3 dedicated receiving buffers
- Up to 3 dedicated send buffers
- Configurable send FIFO or queue
- Configurable Send Event FIFO
- Host CPU has direct access to message RAM
- The FDCAN1 and FDCAN2 modules share the same message RAM
- Supports programmable loopback test mode
- Supports maskable module interrupts
- Two clock domains: CAN core clock and APB bus interface clock

34.1.2 CAN block diagram

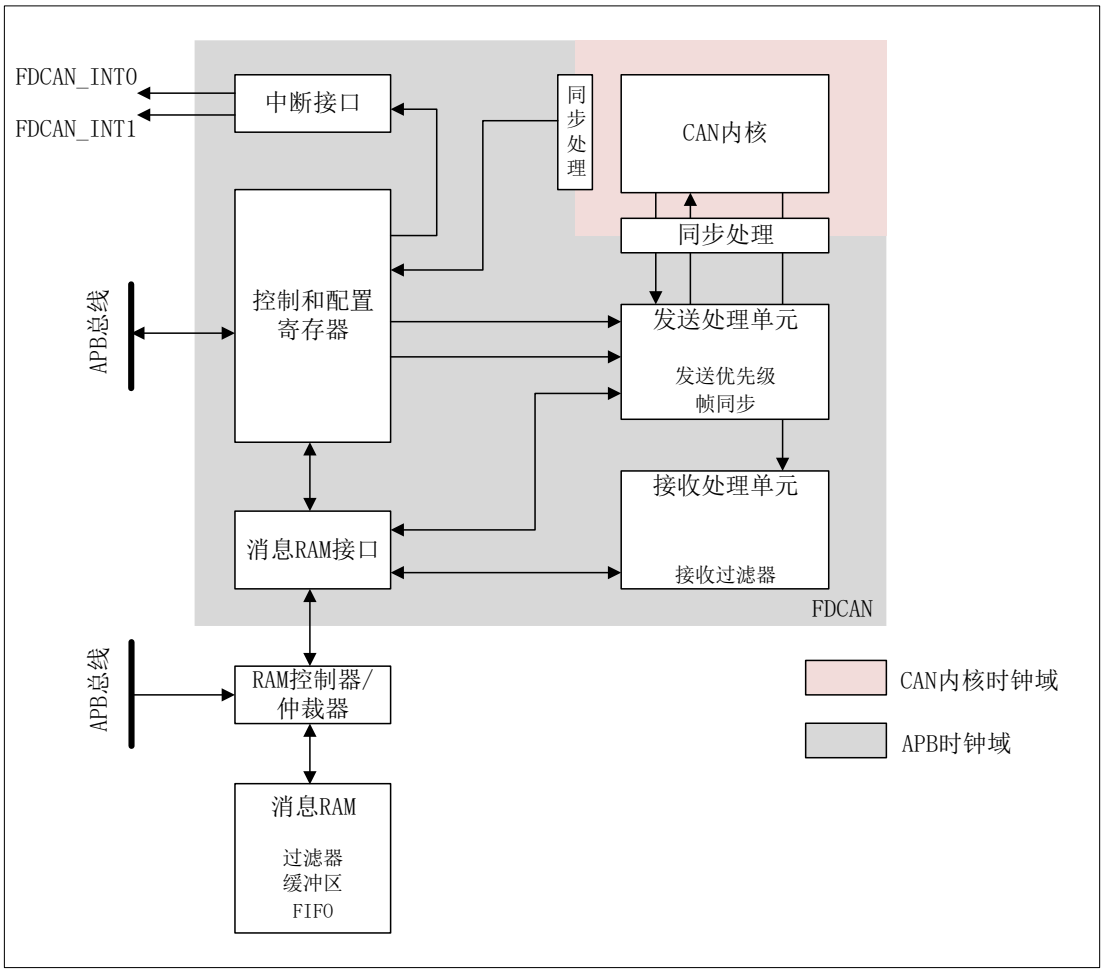


Figure 34-1 Module block diagram

CAN kernel

The CAN core contains a protocol controller, receive and send shift registers. It handles all protocol functions of ISO 11898-1: 2015 and supports 11-bit standard ID and 29-bit extended ID.

synchronization

The synchronization module synchronizes signals from the APB clock domain to the CAN core clock domain and synchronizes signals from the CAN core clock domain to the APB clock domain.

Clocks

The clock of the APB clock domain is the FDCAN control logic clock, and the clock of the CAN kernel clock domain is the FDCAN communication clock, please refer to the section [RCC]. In order to ensure the stability of FDCAN communication, the control logic clock frequency must be greater than or equal to the communication clock frequency.

Transmission processing unit

The processing unit controls the transfer of messages from the external message RAM to the CAN core. Up to 32 send buffers can be configured for sending. The transmit buffer may be used as a dedicated transmit buffer, a transmit FIFO (or queue), or a combination of both. The transmission event FIFO stores the transmission timestamp with the corresponding message ID. Cancellation of sending is also supported.

Receiving processing unit

The processing unit controls the transmission of received messages from the CAN core to the external message RAM. The receive processing unit supports two receive FIFOs (each FIFO is configurable in size) and up to 64 dedicated receive buffers for storing all messages filtered by receive. Unlike the receive FIFO, the dedicated receive buffer is only used to store messages with a specific ID. The reception timestamp is stored with its message. For an 11-bit standard ID, up to 128 filters can be defined; For a 29-bit extended ID, a maximum of 64 filters can be defined.

APB interface

Connect FDCAN to the APB bus.

Message RAM

The interface connects the FDCAN access to the external 2KB message RAM via the RAM controller and arbiter.

Double interrupt line

FDCAN provides two interrupt lines, FDCAN_INT0 and FDCAN_INT1. Program the FDCAN_ILE.EINT0 and FDCAN_ILE.EINT1 bits to enable or disable the interrupt lines, respectively.

34.2 FDCAN functional description

34.2.1 Operating mode

34.2.1.1 Software Initialization

Software initialization begins by setting FDCAN_CCCR.INIT. Software reset, hardware reset, or entering Bus_Off sets FDCAN_CCCR.INIT. When FDCAN_CCCR.INIT is set, both incoming and outgoing messages on the CAN bus stop, and the status of the FDCAN_TX pin becomes recessive (high). The counter for the Error Logic Management Unit (EML) remains the same. Setting FDCAN_CCCR.INIT does not change any configuration registers. Clearing FDCAN_CCCR.INIT completes software initialization, and then the Bitstream Processing Unit (BSP) waits for a sequence of 11 consecutive recessive bits (Bus_Idle) to appear on the bus to synchronize itself with data transmission on the CAN bus before participating in bus activities and starting message transmission.

The FDCAN configuration register can only be accessed when both FDCAN_CCCR.INIT and FDCAN_CCCR.CCE are set. FDCAN configuration registers are shown in the following table.

Table 34-1 FDCAN Configuration Register List

Register	Description
FDCAN_DBTP	FDCAN Data Bit Time and Prescalation Register
FDCAN_TEST	FDCAN test register
FDCAN_RWD	FDCAN RAM Watchdog Register
FDCAN_CCCR	FDCAN CC Control Register
FDCAN_NBTP	FDCAN Nominal Bit Time and Prescalation Register
FDCAN_TSCC	FDCAN timestamp counter configuration register
FDCAN_TOCC	FDCAN timeout counter configuration register
FDCAN_TDCR	FDCAN transmitter delay compensation register
FDCAN_GFC	FDCAN global filter configuration register
FDCAN_SIDFC	FDCAN standard ID filter configuration register
FDCAN_XIDFC	FDCAN Extended ID Filter Configuration Register
FDCAN_XIDAM	FDCAN Extension ID and Mask Register
FDCAN_RXF0C	FDCAN receives FIFO0 configuration register
FDCAN_RXBC	FDCAN receive buffer configuration register
FDCAN_RXF1C	FDCAN receive FIFO1 configuration register
FDCAN_RXESC	FDCAN receive buffer and FIFO element size configuration register
FDCAN_TXBC	FDCAN send buffer configuration register

FDCAN_TXESC	FDCAN send buffer element size configuration register
FDCAN_TXEFC	FDCAN send event FIFO configuration register

FDCAN_CCCR.CCE can only be set or cleared when FDCAN_CCCR.INIT is set, and automatically cleared when FDCAN_CCCR.INIT is cleared.

When FDCAN_CCCR.CCE is set, the following registers are reset:

– FDCAN_HPMS-High priority message status

– FDCAN_RXF0S-Receive FIFO 0 status

– FDCAN_RXF1S-Receive FIFO 1 status

– FDCAN_TXFQS-Send FIFO/queue status

– FDCAN_TXBRP-Send buffer request suspended

– FDCAN_TXBTO-Send buffer Send occurs

– FDCAN_TXBCF-Send buffer cancellation complete

– FDCAN_TXEFS-Send event FIFO status

– FDCAN_TXBAR-Send buffer addition request

– FDCAN_TXBCR-Send buffer cancellation request

When CCCR.CCE is set, the timeout counter value TOCV.TOC is preset to the value configured by TOCC.TOP.

The following registers are writable only if CCCR.CCE = '0'

• TXBAR-Tx Buffer add request

• TXBCR-Tx Buffer cancellation request

The host can set FDCAN_CCCR.TEST and FDCAN_CCCR.MON only if both FDCAN_CCCR.INIT and FDCAN_CCCR.CCE are set, and the host can reset FDCAN_CCCR.TEST and FDCAN_CCCR.MON at any time. FDCAN_CCCR.DAR can be set or cleared only if both FDCAN_CCCR.INIT and FDCAN_CCCR.CCE are set.

34.2.1.2 Normal operations

After FDCAN is initialized and FDCAN_CCCR.INIT is cleared, FDCAN synchronizes itself with the CAN bus and is ready to communicate. When receiving a message, the message received through the receive filter (containing the message ID and DLC) is stored in a dedicated receive buffer, either receive FIF0 or receive FIF1. When sending messages, initialize or update the dedicated send buffer and/or send FIFO (queue) first. Automatic sending after receiving a remote frame is not supported.

34.2.1.3 CAN FD operation

There are two types of CANFD frame transmission, the first is CANFD frame without bit rate switching, and the second is CANFD frame with bit rate switching (the transmission bit rate of the control segment, data segment and CRC segment is higher than that at the beginning and end of the frame).

The r0 bit in the classic CAN (ClassicalCAN) standard frame (after the IDE bit, the reserved bit in the standard frame format) and the r1 bit in the extended frame (after the RTR bit, the first reserved bit in the extended frame format) are decoded as the FDF bit in the CANFD frame format. The FDF bit is implicit (logic 1) to indicate a CANFD frame, and explicit (logic 0) to indicate a classic CAN frame. In the CANFD frame, two bits of res and BRS (BitRateSwitch) after the FDF bit determine whether to switch the bit rate in the CANFD frame, and indicate that the bit rate has been switched by res explicit and BRS implicit. The res bit implicit encoding is reserved for future protocol extensions. If both the

FDF bit and the res bit in the frame received by FDCAN are implicit, a protocol exception event will be issued by setting FDCAN_PSR.PXE. If protocol exception handling is enabled (FDCAN_CCCR.PXHD = 0), FDCAN will change the working state from receive (FDCAN_PSR.ACT = 0b10) to synchronous (FDCAN_PSR.ACT = 0b00) at the next sampling point. If protocol exception handling is disabled (FDCAN_CCCR.PXHD = 1), FDCAN treats the implicit res bit as a format error and will respond with an error frame.

The CAN FD mode is enabled by programming FDCAN_CCCR.FDOE. If FDCAN_CCCR.FDOE = 1, transmission and reception of CAN FD frames are enabled. Classic CAN frames are always available for transmission and reception. Whether to transmit a CAN FD frame or a classic CAN frame CAN be configured by the FDF bit in the corresponding transmit buffer element. If FDCAN_CCCR.FDOE = 0, the received frame is treated as a classic CAN frame, and an error frame is transmitted when the CAN FD frame is received. If the CAN FD mode is disabled, no CAN FD frame is transmitted even if the FDF position 1 of the buffer element is transmitted. You can change FDCAN_CCCR.FDOE and FDCAN_CCCR.BRSE only if FDCAN_CCCR.INIT and FDCAN_CCCR.CCE are both set.

If FDCAN_CCCR.FDOE = 0, the setting of the FDF and BRS bits is ignored, and the frame is transmitted in the classic CAN format.

If FDCAN_CCCR.FDOE = 1andFDCAN_CCCR.BRSE = 0, only the FDF bits of the transmit buffer element are evaluated.

If FDCAN_CCCR.FDOE = 1and FDCAN_CCCR.BRSE = 1, the bit rate switching transmission of the CAN FD frame is enabled. The transmit buffer elements of all FDF and BRS position bits are transmitted in a bit rate switched CAN FD format.

It is recommended to switch modes during CAN operation only if the following conditions are met:

– The failure rate of the CAN FD data phase is significantly higher than that of the CAN FD arbitration phase. In this case, the transmitted CAN FD bit rate switching should be disabled.

– During system startup, all nodes will send classic CAN messages before being verified to be able to communicate in CAN FD format. If verified, all nodes switch to CAN FD operation.

– Wake-up messages in CAN Partial Networking must be sent in classic CAN format.

When EOL programming (End-of-line programming), not all nodes support CANFD, and non-CANFD nodes remain in silent mode until the end of programming. Subsequently, all nodes switch back to classic CAN communication.

In the CANFD format, the encoding (corresponding data byte length) of DLC codes 0 to 8 is the same as that of classic CAN, and the encoding of codes 9 to 15 is different from that of classic CAN, as shown in the following table

Table 34-2 DLC encoding in FDCAN

DLC encoding		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Data Byte count	Classic CAN	0	1	2	3	4	5	6	7	8	8	8	8	8	8	8	8
	CAN FD	0	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64

In a CAN FD frame, if the BRS bit is implicit, the bit duration will be switched within the frame after the BRS (bit rate switching) bit. Before the BRS bit, in the CAN FD arbitration section, the bit duration is executed as defined by the register FDCAN_NBTP (Nominal Bit Timing & Prescaler Register). In the

following CAN FD data segment, the bit duration is executed according to the definition of the register FDCAN_DBTP (Data Bit Timing & Prescaler Register). The bit duration is switched back from the data segment bit duration after the CRC delimiter, or when an error is detected, whichever occurs first.

The maximum configurable bit rate of the CAN FD data segment depends on the CAN communication clock frequency. For example, if the CAN communication clock frequency is 20 MHz and the shortest configurable bit time is 4 time slices (tq), the bit rate of the data phase is 5 Mbps.

In CAN FD data frame formats with and without bit rate switching, the value of bit ESI (Error Status Indicator) is determined by the error state of the transmitting node at the beginning of transmission. If the transmitting node is in an error passive state, the ESI is transmitted as a recessive bit, otherwise it is transmitted as a dominant bit.

34.2.1.4 Transceiver delay compensation

In the data segment transmitted by CAN FD, only one node sends, and all other nodes are receiving nodes. The length of the bus has no effect. When transmitting over pin FDCAN_TX, FDCAN receives the transmitted data from its local CAN transceiver over pin FDCAN_RX. Received data is delayed due to transmitter delay, and if this delay is greater than TSEG1 (the time period before the sampling point), an in-bit error is detected. In order to make the data segment time shorter than the transmitter delay, delay compensation is introduced. Without transmitter delay compensation, the bit rate of the CAN FD frame data segment will be limited by the transmitter delay.

The protocol unit of FDCAN implements a delay compensation mechanism to compensate for the transmitter delay, thereby enabling transmission at a higher bit rate in the CAN FD data segment without being affected by the specific CAN transceiver delay.

In order to check whether the data segment of the transmitting node has bit errors, the delayed transmitted data is compared with the received data at the second sampling point SSP. If a bit error is detected, the transmitting node will react to the bit error at the next regular sampling point. In the quorum segment, delay compensation is always disabled.

Transmitter delay compensation supports configurations where the data bit time is shorter than the transmitter delay, as described in detail in ISO 11898-1: 2015, and is enabled by setting FDCAN_DBTP.TDC.

The received bits are compared with the transmitted bits at the second sampling point SSP. The position of the SSP is defined as the sum of the delay measured from the FDCAN transmit output pin FDCAN_TX through the transceiver to the receive input pin FDCAN_RX and the transmitter delay compensation offset configured by FDCAN_TDCR.TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP in the received bits (e.g. half the bit time of the data segment). The position of the second sampling point is rounded down to the next integer mtq (Minimum time quantum, that is, the minimum time slice, which is one FDCAN communication clock cycle).

FDCAN_PSR.TDCV displays the actual transmitter delay compensation value. When FDCAN_CCCR.INIT is set, FDCAN_PSR.TDCV is cleared; When FDCAN_DBTP.TDC is set, FDCAN_PSR.TDCV is updated each time an FD frame is transmitted.

For transmitter delay compensation implemented in FDCAN, the following boundary conditions must be considered:

– The sum of the measured delay from FDCAN_TX to FDCAN_RX and the configured transmitter delay compensation offset FDCAN_TDCR.TDCO must be less than 6 bits of the data segment.

– The sum of the measured delay from FDCAN_TX to FDCAN_RX and the configured transmitter delay compensation offset FDCAN_TDCR.TDCO must be less than or equal to 127mtq. If this sum is greater than 127 mtq, the transmitter delay compensation uses a maximum value of 127 mtq.

– The data segment ends at the sampling point of the CRC delimiter and will stop checking the received bit at the SSP.

If transmitter delay compensation is enabled by programming FDCAN_DBTP.TDC = 1, the measurement starts at the falling edge of the FDF bit to res bit of each CAN FD frame. When the edge is detected on the FDCAN_RX pin of the transmitting node, the measurement is stopped. The resolution of this measurement is one mtq.

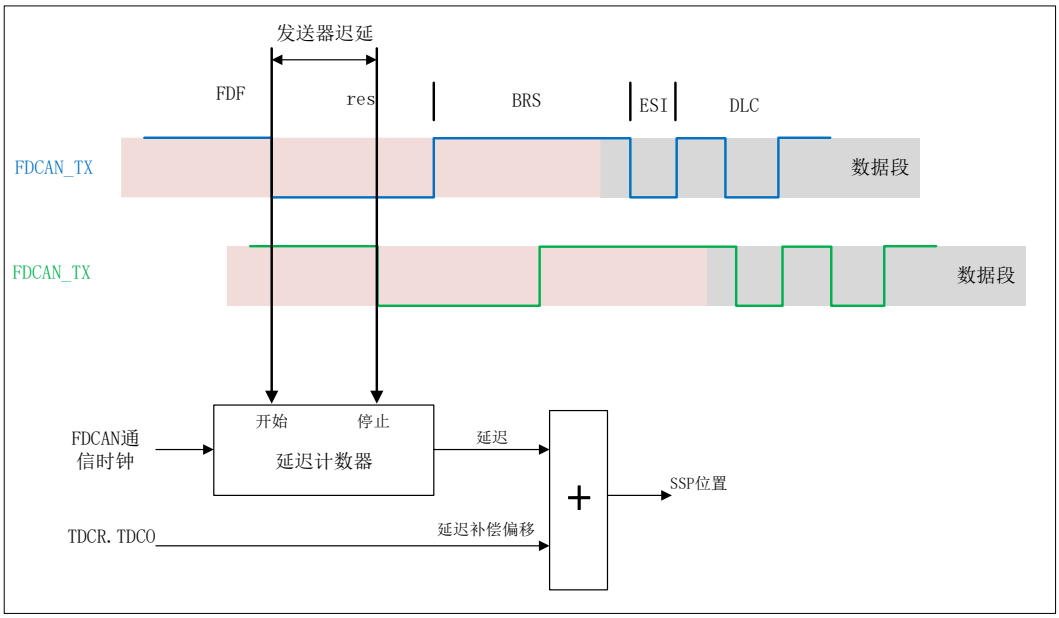


Figure 34-2 Transceiver Delay Measurement

In order to avoid dominant glitches in the received FDF bits, the delay compensation measurement is ended before the falling edge of the received res bit, resulting in an advance of the SSP position, the transmitter delay compensation filtering window may be enabled by programming FDCAN_TDCR.TDCF, thereby defining a minimum value of the SSP position. The dominant edge on the FDCAN_RX pin causes the SSP position to be more advanced, but this will be ignored for transmitter delay measurements. When the SSP position is at least FDCAN_TDCR.TDCF and the FDCAN_RX pin is low, the measurement stops.

34.2.1.5 Limited operating mode

In restricted operating mode, nodes can receive data frames and remote frames, and can acknowledge valid frames, but do not send data frames, remote frames, active error frames, or overload frames. In the event of an error or overload condition, the dominant bit is not sent, but waits for a bus idle condition to occur to resynchronize to CAN communication. The error counters (FDCAN_TDCR.REC, FDCAN_TDCR.TEC) are frozen during the error logging (FDCAN_TDCR.CEL) action. The host can set FDCAN to restricted operating mode by setting FDCAN_CCCR.ASM, which the host can only set

when both FDCAN_CCCR.CCE and FDCAN_CCCR.INIT are set to 1. FDCAN_CCCR.ASM can be cleared by the host at any time.

When the sending processing unit cannot read data from the message RAM in time, it will automatically enter the restricted operation mode. To exit restricted operating mode, the host CPU must clear FDCAN_CCCR.ASM.

The restricted operating mode CAN be used to accommodate applications with different CAN bit rates. In this case, the application tests for a different bitrate and exits the restricted mode of operation after receiving a valid frame.

Note: Restricted operating mode must not be used in conjunction with loopback mode (internal or external).

34.2.1.6 Bus listening mode

Configure FDCAN_CCCR.MON to 1 and set FDCAN to bus listening mode. In bus listening mode (see ISO 11898-1: 2015, 10.14 Bus monitoring), FDCAN is able to receive valid data frames and valid remote frames, but cannot initiate transmission. In this mode, FDCAN only sends recessive bits on the CAN bus, if FDCAN needs to send a recessive bit (ACK bit, overload flag, active error flag), that bit will be internally diverted to send so that FDCAN CAN listen for the recessive bit, but the CAN bus CAN remain recessive. In bus listen mode, the register FDCAN_TXBRP remains reset.

The bus listening mode CAN be used to analyze the traffic on the CAN bus without affecting the bus by sending dominant bits. Figure 34-3 shows the connection of FDCAN_TX and FDCAN_RX signals to FDCAN in bus listening mode.

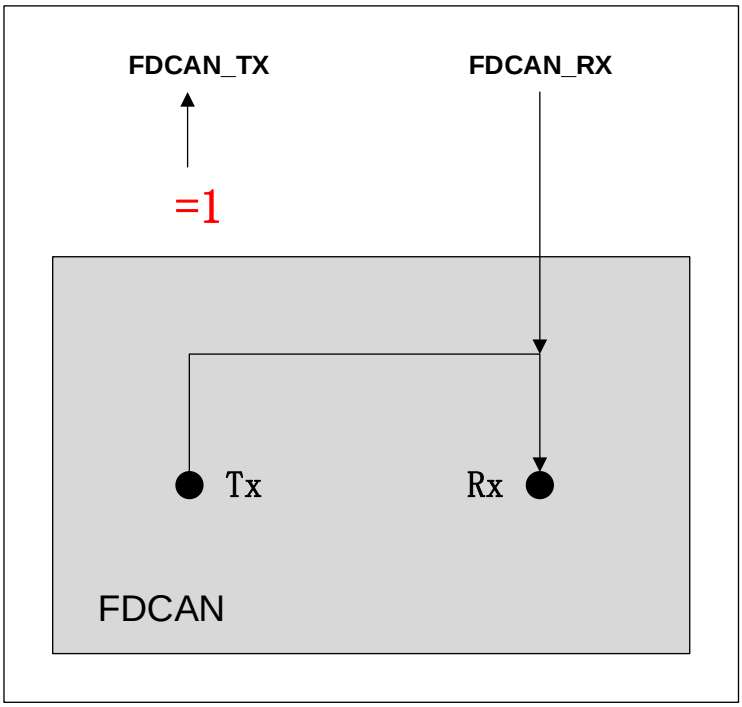


Figure 34-3 Pin control in bus listening mode

34.2.1.7 Disable Automatic Retransmission (DAR) Mode

According to the CAN specification (see ISO 11898-1: 2015, 8.3. 4 Recovery Management), FDCAN CAN automatically retransmit frames that have failed arbitration or have been erroneously interfered during transmission. Automatic retransmission is enabled by default.

In DAR mode, all transmissions are automatically cancelled after they are initiated on the CAN bus. The send buffer send request suspend bit FDCAN_TXBRP.TRPx is reset after a successful send, and is also reset if the send has not started at the time of cancellation, the send has been aborted due to an arbitration failure, or an error occurs during frame sending.

- Successfully sent:
  – The corresponding send buffer sends the occurred bit FDCAN_TXBTO. TOx set
  – The corresponding send buffer cancellation completed bit FDCAN_TXBCF.CFx is not set
- Successfully sent even though cancelled:
  – The corresponding send buffer sends the occurred bit FDCAN_TXBTO. TOx set
  – The corresponding send buffer cancels the completed bit FDCAN_TXBCF.CFx set
- Arbitration failed or frame transmission was interfered:
  – The corresponding send buffer sends the occurred bit FDCAN_TXBTO. TOx is not set
  – The corresponding send buffer cancels the completed bit FDCAN_TXBCF.CFx set

If the frame is transmitted successfully and storage of the transmit event is enabled, the transmit event FIFO element is written with event type ET = 0b10 (transmit will occur even if cancelled).

34.2.1.8 Power Down (Sleep Mode)

The FDCAN can be set to power down mode through the CC control register FDCAN_CCCR.CSR. The FDCAN_CCCR.CSR bit reads to 1 as long as the clock stop request is valid.

When all pending send requests are completed, FDCAN will wait until the bus idle state is detected. After detecting the bus idle state, FDCAN sets FDCAN_CCCR.INIT to 1 to prevent any further CAN transmission. FDCAN then confirms that it is ready to enter the power down state by setting FDCAN_CCCR.CSA to 1. The registers can continue to be accessed until the module clock (control logic clock and communication clock) is turned off, but FDCAN_CCCR.INIT remains at 1.

Notes:

In the case of severe interference of the CAN bus, the idle state may never be reached, so FDCAN does not set FDCAN_CCCR.INIT. This condition can be detected by polling FDCAN_PSR.ACT. If the FDCAN does not enter the idle state, the software CAN write CCCR.INIT = 1, which will immediately stop the CAN communication of the FDCAN regardless of whether sending or receiving is in progress.

To exit power-down mode, the application must turn on the module clock before resetting the CC control register flag FDCAN_CCCR.CSR. FDCAN will confirm that the clock is on by resetting FDCAN_CCCR.CSA. After that, the application CAN restart CAN communication by resetting FDCAN_CCCR.INIT.

34.2.1.9 Test Mode

To enable write access to the FDCAN test register FDCAN_TEST (see section [FDCAN Test Register (FDCAN_TEST)]), FDCAN_CCCR.TEST must be set to 1 so that the test mode and test functionality can be configured.

By programming FDCAN_TEST.TX, the send pin FDCAN_TX can achieve four output functions. In addition to the default function of serial data output, the CAN sampling point signal CAN be driven to monitor the bit duration of FDCAN and CAN output a constant dominant or implicit level. The actual

value of pin FDCAN_RX can be read from FDCAN_TEST.RX. Both functions (FDCAN_TEST.TX and FDCAN_TEST.RX) CAN be used to check the physical layer of the CAN bus.

Due to the synchronization mechanism between the CAN communication clock and the control logic clock, there may be a delay of several control logic clock cycles between configuring FDCAN_TEST.TX and pin FDCAN_TX implementing the specified function. This also applies when the pin FDCAN_RX is read through FDCAN_TEST.RX.

Notes:

The test mode is limited to production testing and self-testing. Software control of the FDCAN_TX pin interferes with all CAN protocol functions. It is not recommended to use test mode for practical applications.

34.2.1.10 External loopback mode

Configure FDCAN_TEST.LBCK to 1 and set FDCAN to external loopback mode. In loopback mode, FDCAN handles messages it sends itself as received messages and stores the messages (if they pass the receive filtering) into the receive buffer or receive FIFO. The following diagramshows the connection of the FDCAN_TX and FDCAN_RX pin signals to FDCAN in external loopback mode.

This mode is used for hardware self-test. In order to be protected from external influences, FDCAN ignores ACK errors in loopback mode (sampling to recessive bits in the ACK field of data frames or remote frames). In this mode, the signal output by FDCAN_TX is internally fed back directly to the receiving processing unit, and the actual value of the FDCAN_RX input pin is ignored. The sent message can be listened to from the pin FDCAN_TX.

34.2.1.11 Internal loopback mode

Configure FDCAN_TEST.LBCK and FDCAN_CCCR.MON to 1 and set FDCAN to internal loopback mode. This mode CAN be used for "Hot Selftest", that is, FDCAN CAN detect without affecting the connected running CAN system. In this mode, pin FDCAN_RX is disconnected from FDCAN and pin FDCAN_TX remains recessive. The following diagramshows the connection of the FDCAN_TX and FDCAN_RX pin signals to FDCAN in internal loopback mode.

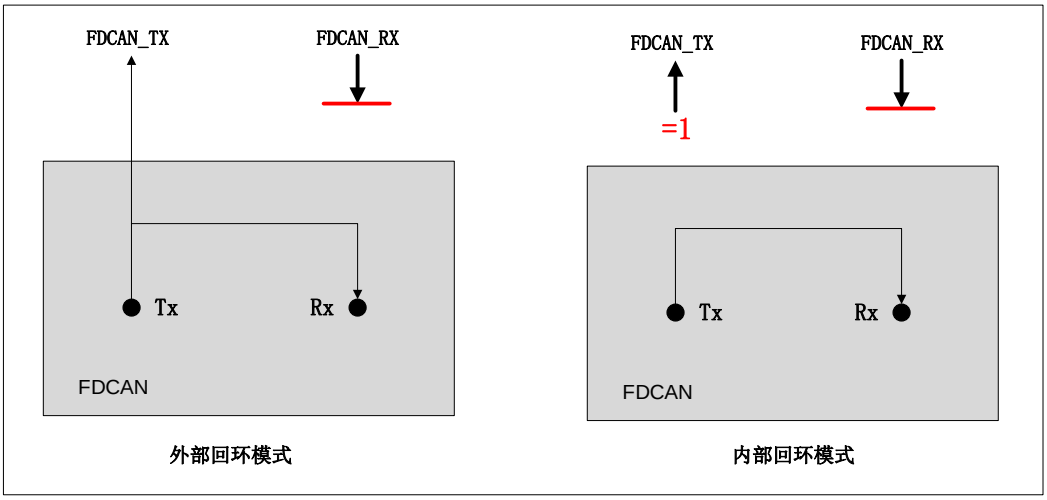


Figure 34-4 Pin Control for Loopback Mode

34.2.2 Timestamp generation

FDCAN provides a 16-bit cycle counter to generate internal timestamps. The counting time unit CAN be configured to be 1 to 16 times the CAN bit time through the prescaler FDCAN_TSCC.TCP. The counter value can be read via FDCAN_TSCV.TCV. A write access to register FDCAN_TSCV resets the counter to 0. When the timestamp counter loops, the interrupt flag FDCAN_IR.TSW is set.

When a SOF (Start Of Frame) is received or transmitted, the count value is captured and stored to either the receive buffer or the timestamp portion of the receive FIFO (RXTS [15: 0]) or transmit event FIFO (TXTS [15: 0]) element.

34.2.3 Timeout counter

FDCAN provides a 16-bit timeout counter to indicate timeout conditions for receive FIF0, receive FIF1, and send event FIFO. It works as a decrement counter and, like the timestamp counter, uses a pre-scaler controlled by FDCAN_TSCC.TCP. The timeout counter is configured via the register FDCAN_TOCC. The actual counter value can be read from FDCAN_TOCV.TOC. The timeout counter starts only if FDCAN_CCCR.INIT = 0. When FDCAN_CCCR.INIT = 1, the timeout counter stops counting, for example, when CAN enters the Bus_Off state.

The operating mode is selected via FDCAN_TOCC.TOS. When operating in continuous mode, the counter starts when FDCAN_CCCR.INIT is reset. Writing to FDCAN_TOCV presets the counter to the value configured by FDCAN_TOCC.TOP and continues to decrement the count.

When the timeout counter is controlled by one of the FIFOs, a null FIFO presets the counter to the value configured by FDCAN_TOCC.TOP. After the first FIFO element is stored, the decrement count begins. Performing a write to FDCAN_TOCV does not work.

When the counter reaches 0, the interrupt flag FDCAN_IR.TOO is set. In continuous mode, the counter restarts immediately after reaching FDCAN_TOCC.TOP.

Notes:

The clock signal of the timeout counter comes from the CAN core sampling point signal. Therefore, the point in time at which the timeout counter is decremented may vary due to the CAN kernel synchronization (resynchronization) mechanism. If the bit rate switching function is used in the CAN FD, the timeout counter uses different clock counts in the arbitration field and the data field.

34.2.4 Receiving processing

The receive processing unit controls the receive filtering, the transmission of received messages to the receive buffer or one of the receive FIFOs (of two in total), and the receive FIFO putting and getting indexes.

34.2.4.1 Receive filtering

FDCAN is able to configure two sets of receive filters, one for the standard ID and the other for the extended ID. These two sets of filters may be assigned to receive buffer, receive FIFO 0, or receive FIFO 1. For receive filtering, each filter list performs a comparison starting from the 0th element until the first matching element. Receive filtering stops at the first matched element. The remaining filter elements are not evaluated for this message.

- The main features are:
  - Each filter element can be configured as

* ID Range Filter (Start and End Range)

* Filters with one or two dedicated IDs

* Classic filter for ID bit masking

– Each filter element can be configured to implement receive filtering or reject filtering

– Each filter element can be enabled/disabled individually

– Compare filter elements sequentially and stop execution after the first matching filter element

▪ Relevant configuration register bits:

– Global filter configuration (FDCAN_GFC)

– Standard ID filter configuration (FDCAN_SIDFC)

– Extended ID filter configuration (FDCAN_XIDFC)

– Extended ID and Operation Mask (FDCAN_XIDAM)

▪ Depending on the configuration of the filter element (SFEC/EFEC), one of the following actions is triggered when a match occurs:

– Store received frames into receive FIFO0 or FIFO1

– Store received frames in receive buffer

– Reject received frames

– Set high priority message interrupt flag FDCAN_IR.HPM to 1

– Set the high-priority message interrupt flag FDCAN_IR.HPM to 1, and store the received frame in the receiving FIF0 or FIF1. After receiving the complete ID, start receiving filtering. After the receive filtering is completed, if a matching receive buffer or receive FIFO is found, the message processing unit will start writing the received message data in 32 bits into the matching receive buffer or receive FIFO. If the CAN protocol controller detects an error condition, such as a CRC error, this message is discarded with the following effects:

Receive buffer

The new data flag that matches the receive buffer will not be set to 1, but the receive buffer (part of it) will be overwritten by the received data. For error types, see FDCAN_PSR.LEC and FDCAN_PSR.DLEC.

Receive FIFO

The put-in index that matches the receive FIFO is not updated, but the relevant receive FIFO element (part) is overwritten by the received data. For error types, see FDCAN_PSR.LEC and FDCAN_PSR.DLEC. If the matching receive FIFO operates in coverage mode, the boundary conditions introduced in the receive FIFO coverage mode must be considered.

Notes:

When a received message is written to one of the two receive FIFOs or written to the receive buffer, the unmatched received ID is stored independently of the filter used. The process of receiving filtering depends largely on the order of the filter elements configured.

34.2.4.2 ID Range Filter

This filter checks all received frames for match with message IDs within the range defined by SF1ID/SF2ID and EF1ID/EF2ID. When range filtering is used with extended frames, there are two possible scenarios:

- EFT = 0b00: The message ID and extension ID of the received frame are AND with the operation mask (FDCAN_XIDAM) before the scope filter is applied
- EFT = 0b11: Extended ID and operation mask (FDCAN_XIDAM) are not used for range filtering

34.2.4.3 Dedicated ID Filter

The filter element can be configured to filter one or two specific message IDs. To filter a specific message ID, the filter elements must be configured as SF1ID = SF2ID and EF1ID = EF2ID.

34.2.4.4 ID bit mask filter

ID bit masking filtering is used to filter groups of message IDs by masking certain bits of received message IDs. When performing ID bit mask filtering, SF1ID/EF1ID is used as a message ID filter and SF2ID/EF2ID is used as a filter ID Mask.

A bit of 0 in the ID mask will mask the corresponding bit of the configured ID filter, i.e., the value of the received message ID at this bit position is not compared with the value of the corresponding bit of the receive filter, only the bit of the received message ID corresponding to the mask bit of 1 is compared. If all mask bits are 1, a match will only occur if the received message ID and message ID filter are exactly the same. If all mask bits are 0, all message IDs match.

34.2.4.5 Standard message ID filtering

Figure 34-5 illustrates the filtering process for standard message IDs (11-bit IDs). The chapter [Standard Message ID Filter Elements] describes the standard message ID filter elements.

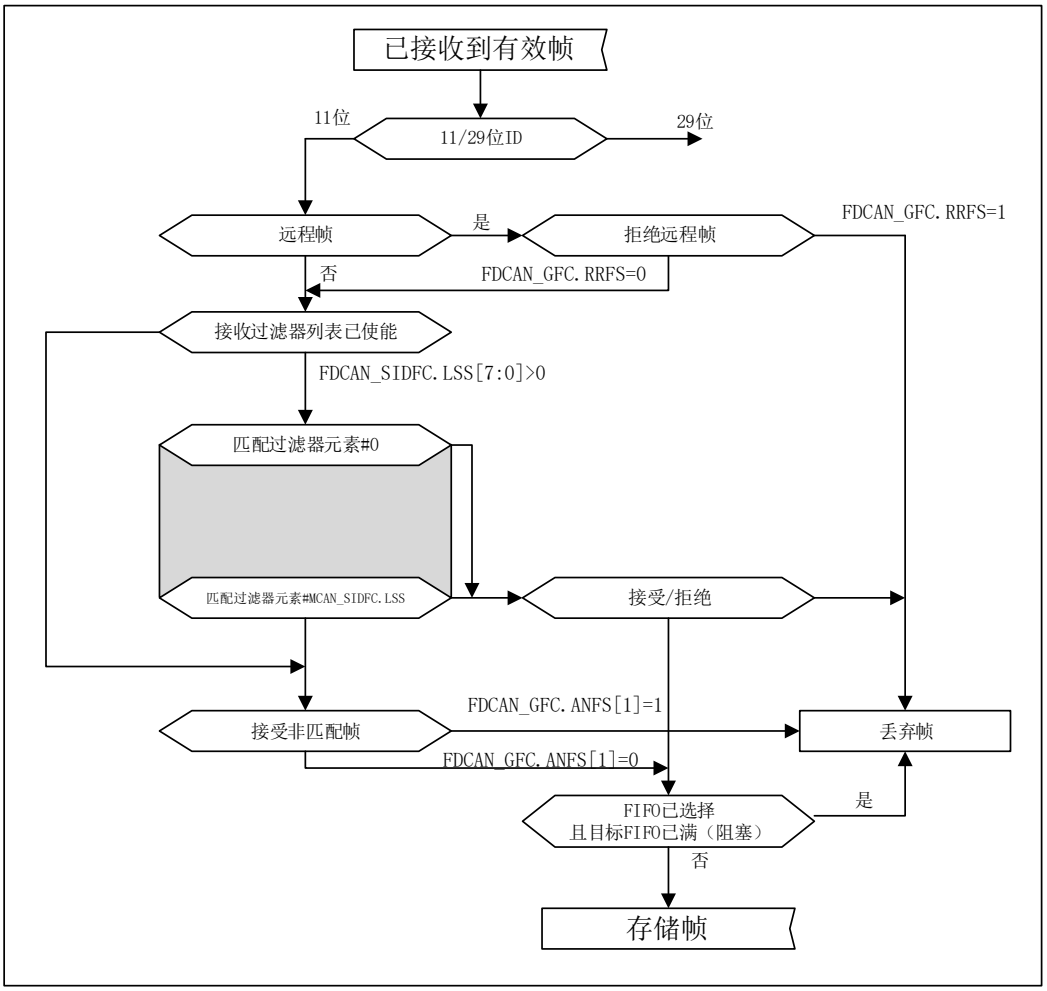


Figure 34-5 Standard message ID filter path

Under the control of the Global Filter Configuration Register (FDCAN_GFC) and the Standard ID Filter Configuration Register (FDCAN_SIDFC), the ID, Remote Transfer Request Bit (RTR), and ID Extension Bit (IDE) of the received message are compared to the configured list of filter elements.

34.2.4.6 Extended message ID filtering

Figure 34-6 illustrates the filtering process for extended message IDs (29-bit IDs). The chapter [Extended Message ID Filter Elements] introduces the extended message ID filter element.

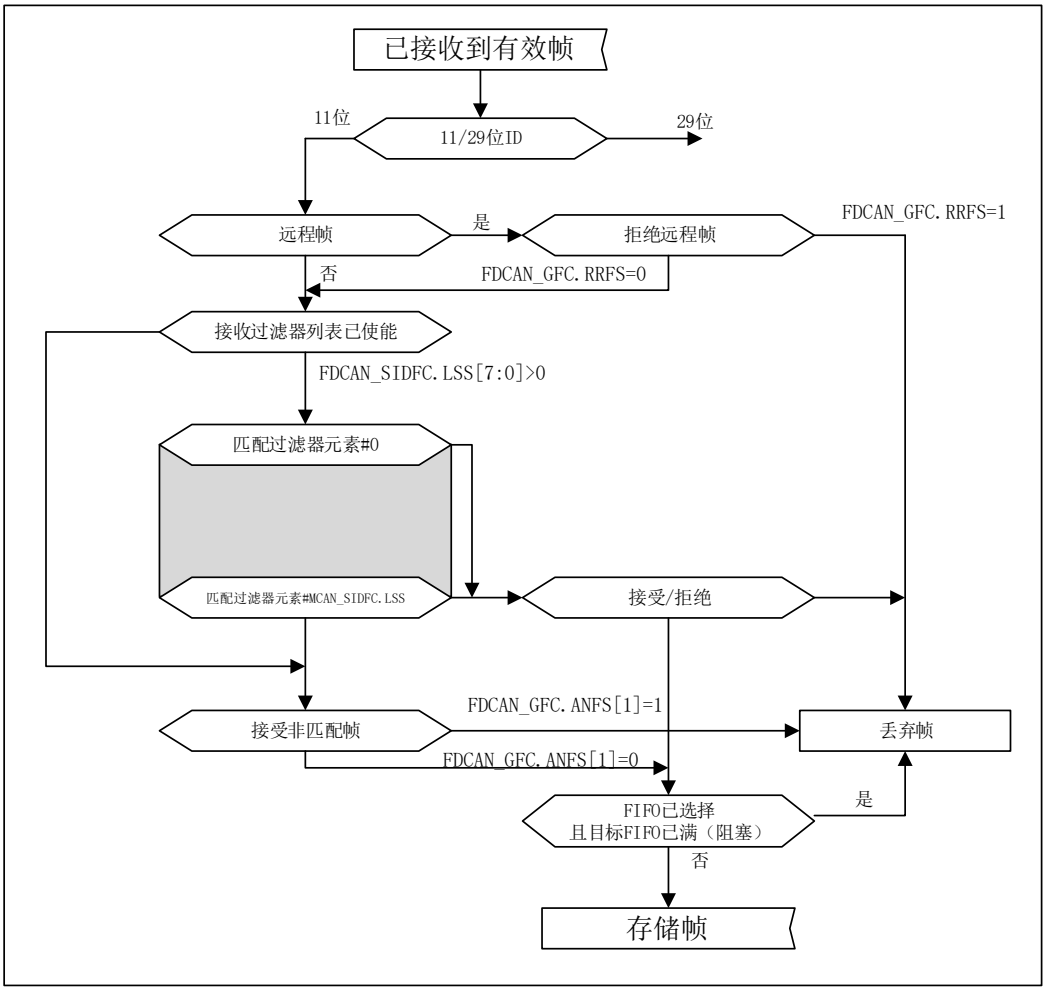


Figure 34-6 Extended Message ID Filter Path

Under the control of the Global Filter Configuration Register (FDCAN_GFC) and the Extended ID Filter Configuration Register (FDCAN_XIDFC), the ID, Remote Transfer Request Bit (RTR), and ID Extension Bit (IDE) of the received message are compared to the configured list of filter elements. The extended ID and mask (FDCAN_XIDAM) ANDs the received ID before the filter list is executed.

34.2.4.7 Receive FIFO

Receive FIFO0 and receive FIFO1 can store up to 64 elements respectively (the actual number of elements is related to the message RAM capacity). The configuration of the two receiving FIFOs is done through the registers FDCAN_RXFOC and FDCAN_RXF1C.

Messages filtered by receive are transmitted to the receive FIFO that matches the filter element configuration. For a description of the filter mechanisms available for receive FIFO0 and receive FIFO1, see the section [Receive Filtering]. The chapter [Receive Buffer and FIFO Elements] introduces the receive buffer and FIFO elements.

To avoid receiving FIFO overflows, a receiving FIFO Watermark may be used. When the received FIFO fill level reaches the received FIFO water mark configured by FDCAN_RXFnC.FnWM, the interrupt flag FDCAN_IR.RFnW is set. When the put index and the fetch index of the receive FIFO meet, the condition that the receive FIFO is full is generated, and the flag bit FDCAN_RXFnS.FnF is set, and the interrupt flag bit FDCAN_IR.RFnF is set.

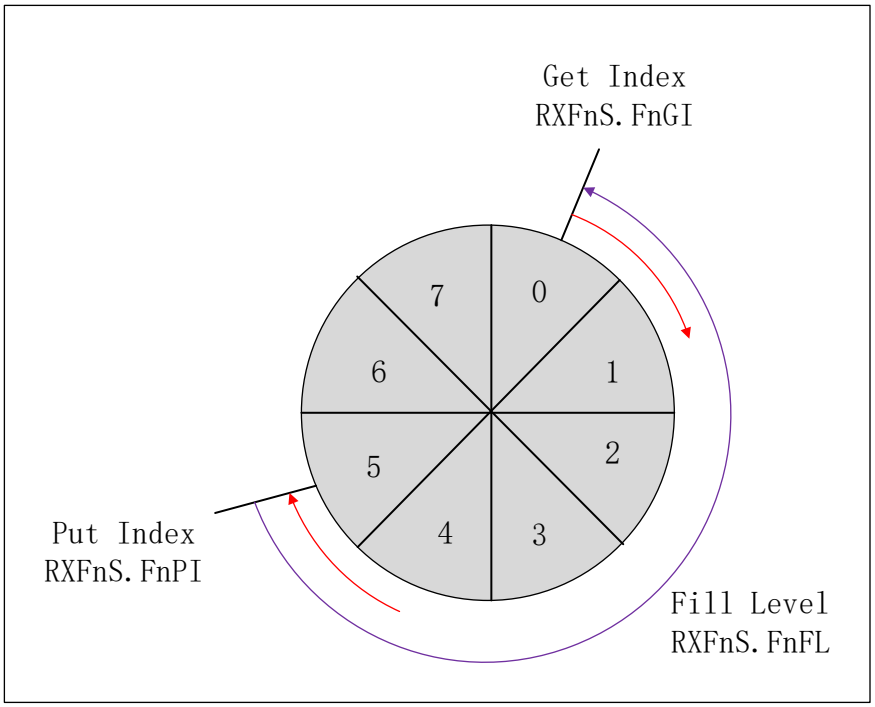


Figure 34-7 Receive FIFO State

When a message is read from a receiving FIFO, the receiving FIFO fetch index FDCAN_RXFnS.FnGI multiplied by the FIFO element size must be added to the start address FDCAN_RXFnC.FnSA.

Table 34-3 receive buffer and FIFO element size

RXESC.RBDS [2: 0] **RXESC.FnDS [2: 0]**	**Data segment (bytes)**	**Element size (word)**
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

34.2.4.8 Receive FIFO blocking mode

The receive FIFO blocking mode is configured by FDCAN_RXFnC.FnOM = 0, which is the default operating mode of the receive FIFO.

When the receive FIFO full condition is met (FDCAN_RXFnS.FnPI = FDCAN_RXFnS.FnGI), the flag bit RXFnS.FnF = 1, and the interrupt flag FDCAN_IR.RFnF = 1. At this point, the FDCAN does not continue to write messages to the corresponding receive FIFO unless the message is read from the corresponding receive FIFO and its fetch index has been incremented.

If a message is received when the corresponding receive FIFO is full, this message is discarded and indicates that the message is lost by FDCAN_RXFnS.RFnL = 1. In addition, the interrupt flag FDCAN_IR.RFnL is also set.

34.2.4.9 Receive FIFO overlay mode

The receive FIFO overlay mode is configured by FDCAN_RXFnC.FnOM = 1.

When the receive FIFO full condition is met (FDCAN_RXFnS.FnPI = FDCAN_RXFnS.FnGI), the flag bit RXFnS.FnF = 1. At this point, the next message received by FIFO will overwrite the earliest received message, and both put in index and get index will be added by 1.

If the receive FIFO is operating in override mode and the receive FIFO is full, the receive FIFO element should be read from at least the fetch index plus 1. This is because when the CPU is reading messages from the message RAM, there happens to be a received message being written to the message RAM, which may cause the data read by the CPU to be inconsistent with the actual, and adding an offset to the fetch index when reading data from the receive FIFO can avoid this problem. The offset depends on how quickly the CPU accesses the receiving FIFO. Figure 34-8 shows that the offset of the fetch index is 2 when reading the receive FIFO. In this case, the two messages stored in elements 1 and 2 are lost.

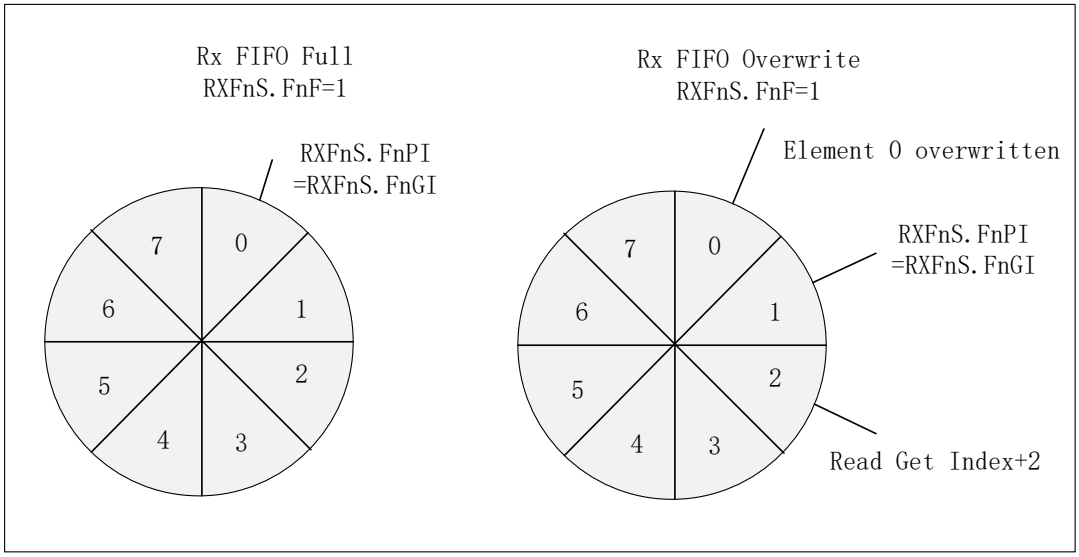


Figure 34-8 Receive FIFO Overflow Processing

After reading data from the receiving FIFO, the number of the last element read must be written to the receiving FIFO acknowledgement index register FDCAN_RXFnA.FnA so that the incremented acquisition index points to the next element. If the drop index is not increased to the receive FIFO element number, the receive FIFO full flag is reset (FDCAN_RXFnS.FnF = 0).

34.2.4.10 Dedicated receive buffer

FDCAN supports up to 64 dedicated receive buffers. The start address of the dedicated receive buffer is configured via RXBC.RBSA.

To use a dedicated receive buffer, the standard message ID filter element must be configured to have an SFEC bit of 0b111 and an SFID2 bit of 0b00 for standard ID messages (see section [Standard message ID filter elements]); For extended ID messages, the extended message ID filter element must be configured to have an EFEC bit of 0b111 and an EFID2 bit of 0b00 (see section [Extended Message ID Filter Elements]). The following table shows an example filter configuration for a dedicated receive buffer.

After received messages are accepted by the filter element, they are stored in the receive buffer in the message RAM referenced by the filter element. The storage format is the same as the format in which the FIFO element is received. In addition, the flag FDCAN_IR.DRX (messages stored in the dedicated receive buffer) in the interrupt register is also set to 1.

Table 34-4 Receive Buffer Filter Configuration Example

Filter element	**SFID1 [10: 0] EFID1 [28: 0]**	**SFID2 [10: 9] EFID2 [10: 9]**	**SFID2 [5: 0] EFID2 [5: 0]**
0	ID Message 1	00	00 0000
1	ID Message 2	00	00 0001
2	ID Message 3	00	00 0010

After the last word of the matching received message is written to the message RAM, the corresponding new data flag in registers FDCAN_NDAT1/FDCAN_NDAT2 is set. As long as the new data flag remains set, the corresponding receive buffer is locked so as not to be updated to a new matching message. The new data flag must be reset by the host writing 1 to the corresponding bit.

When the new data flag of the reception buffer is set, the corresponding message ID filter element will not perform reception filtering, and other filter elements will continue to perform reception filtering. Other message ID filter elements may cause the received message to be stored in another receive buffer or receive FIFO, or rejected, depending on the filter configuration.

34.2.4.11 Receive buffer processing

- Reset interrupt flag FDCAN_IR.DRX
- Read new data register
- Read messages from message RAM
- Reset new data flags for processed messages

34.2.5 Send processing

The transmission processing unit processes the transmission requests of the dedicated transmission buffer, the transmission FIFO and the transmission queue. It controls the transmission of send messages to the CAN kernel, putting in and getting indexes, and sending event FIFO. Up to 32 send buffers CAN be set for message sending, and each send buffer CAN be independently configured with send mode (classic CAN mode or CAN FD mode). See section [Send Buffer Elements] for a description of the send buffer. The following table describes possible configurations for transmitting frames.

Table 34-5 Possible Configuration of Send Frame

FDCAN_CCCR		**Send buffer element**		**Send a frame**
BRSE	**FDOE**	**FDF**	**BRS**	
lose sight of	0	lose sight of	lose sight of	Classic CAN
0	1	0	lose sight of	Classic CAN
0	1	1	lose sight of	FD without switching bitrate
1	1	0	lose sight of	Classic CAN
1	1	1	0	FD without switching bitrate
1	1	1	1	FD of switching bitrate

Notes:

AUTOSAR requires at least three send queue buffers and supports send cancellation.

When the send buffer request pending register FDCAN_TXBRP is updated, the send processing unit initiates a send scan to check for the highest priority pending send request (the send buffer with the

smallest message ID). FDCAN_TXBRP is updated on the condition that a transmission buffer is completed, or the host adds a transmission request by writing FDCAN_TXBAR, or the host cancels a pending transmission by writing FDCAN_TXBCR.

34.2.5.1 Send pause

The transmission suspension function is used in a CAN system in which the CAN message ID is (permanently) specified as a specific value and cannot be easily changed. The CAN arbitration priority of these message IDs may be higher than other defined messages, and their relative arbitration priority should be reversed in a specific application. This may lead to a situation in which one ECU bursts a plurality of CAN messages of higher arbitration priority and another ECU CAN message of lower arbitration priority is delayed.

For example, if the CAN ECU-1 has this feature enabled and its application software requests four messages to be sent, after successful sending of the first message, it will wait for a CAN bit time for both buses to be idle before allowing the next requested message to begin sending. If other ECUs have pending messages, they will start sending them during idle time and do not need to arbitrate with the next message of ECU-1. Upon receiving the message, the ECU-1 may start the next transmission immediately after the received message releases the CAN bus.

This functionality is controlled via FDCAN_CCCR.TXP. If this is bit, after each successful message transmission, FDCAN will pause for two CAN bits before starting the next transmission. In this way, messages can be sent even if the ID priority of other nodes in the network is lower. This feature is disabled by default (FDCAN_CCCR.TXP = 0). This feature makes multiple messages from a single node burst sparse, avoiding ECUs sending lower priority messages from applications mistakenly thinking they are requesting too many transmissions.

34.2.5.2 Dedicated send buffer

The dedicated send buffer can send messages completely under the control of the host CPU. Each dedicated send buffer is configured with a specific message ID. If multiple send buffers are configured with the same message ID, the messages in the send buffer with the smallest number are sent first, and these send buffers should be requested in ascending order, that is, the smallest send buffer number is requested first, or simultaneously via a single write FDCAN_TXBAR.

If there is an update to the message in the buffer, it is sent by setting FDCAN_TXBAR.ARn request. 。

The requested message will be arbitrated internally with the message in the sending FIFO or sending queue according to the message ID, then externally with the message on the CAN bus, and finally sent out according to the arbitration result.

The allocated dedicated send buffer element size in the message RAM is several 32-bit words (4 bytes). Therefore, the start address of the dedicated send buffer in the message RAM is calculated by multiplying the send buffer index (0 ~ 31) by the element size (word), then multiplying it by 4 (converted to bytes), and then adding it to FDCAN_TXBC.TBSA.

Table 34-6 send buffer, FIFO (queue) element size

TXESC.TBDS2 [2: 0]	Data segment (bytes)	Element size (word)
000	8	4
001	12	5
010	16	6
011	20	7

100	24	8
101	32	10
110	48	14
111	64	18

34.2.5.3 Send FIFO

The transmission FIFO mode is achieved by configuring FDCAN_TXBC.TFQM to 0. The message stored in the transmit FIFO is transmitted from the message pointed to by the fetch index FDCAN_TXFQS.TFGI. After each send, the fetch index is cyclically incremented until the send FIFO is empty. Send FIFO sends messages from different send buffers but with the same ID in the order in which the messages were written. FDCAN calculates the difference between the acquired index and the put index to obtain the number of available (idle) transmitted FIFO elements (TXFQS.TFFL).

The new message must be written to be placed in the send FIFO at the beginning of the send buffer pointed to by the index FDCAN_TXFQS.TFQPI. Adding a send request increments the put-in index to the next free send FIFO element. When the drop index meets the fetch index, it indicates that the send FIFO is full (FDCAN_TXFQS.TFQF = 1). In this case, you should not continue to write messages to the sending FIFO until the next message has been sent and the fetch index is incremented.

If a single message is added to the transmission FIFO, the transmission is requested by writing 1 to the FDCAN_TXBAR bit corresponding to the index of the transmission FIFO.

If multiple (assuming n) messages are added to the send FIFO, they are written to n consecutive send buffers starting with the drop-in index. It is then sent via the FDCAN_TXBAR request, putting in the index cyclically incremented by n. The number of send buffers requested to be sent should not exceed the number of free send buffers indicated by the send FIFO idle level (FDCAN_TXFQS.TFFL).

If the send request for the send buffer pointed to by the fetch index has been canceled, the fetch index is incremented to the next send buffer with a send request pending and the send FIFO idle level (FDCAN_TXFQS.TFFL) is recalculated. If you cancel the sending of any other send buffer, the fetch index and FIFO idle level remain unchanged.

The allocated transmit FIFO element size in the message RAM is several 32-bit words (4 bytes). Therefore, the start address of the next available (idle) transmit FIFO is calculated by multiplying the put-in index FDCAN_TXFQS.TFQPI (0-31) by the element size (word), multiplying it by 4, and then adding it to FDCAN_TXBC.TBSA.

34.2.5.4 Send Queue

Send queue mode is achieved by configuring FDCAN_TXBC.TFQM to 1. Messages stored in the send queue are sent first from the message ID with the smallest (highest priority). If multiple send queues are configured with the same message ID, if multiple send queues are configured with the same message ID, the send order cannot be predicted because the send order is related to the number of the send queues in which the message is stored, and the number of these send queues depends on the state currently put into the index.

The new message must be written into the send buffer pointed to by the index FDCAN_TXFQS.TFQPI. The drop-in index always points to the least numbered free buffer in the send queue. If the send queue is full (FDCAN_TXFQS.TFQF = 1), the put-in index is invalid and messages should not continue to be

written to the send queue until at least one requested message has been issued or a pending send request has been canceled.

The application can use the register FDCAN_TXBRP instead of putting the index, and can put the message into any send buffer where there is no pending send request.

The allocated send queue buffer element size in message RAM is several 32-bit words (4 bytes). Therefore, the start address of the next available (free) send queue buffer is calculated by multiplying the put-in index FDCAN_TXFQS.TFQPI (0-31) by the element size (word), multiplying it by 4, and then adding it to FDCAN_TXBC.TBSA.

34.2.5.5 Mixing dedicated send buffer and send FIFO

In this case, the send buffer in the message RAM is divided into a set of dedicated send buffers and a send FIFO. The number of dedicated send buffers is configured via FDCAN_TXBC.NDTB. The number of transmit buffers allocated to the transmit FIFO is configured by FDCAN_TXBC.TFQS. If FDCAN_TXBC.TFQS is configured to 0, only the dedicated send buffer will be used.

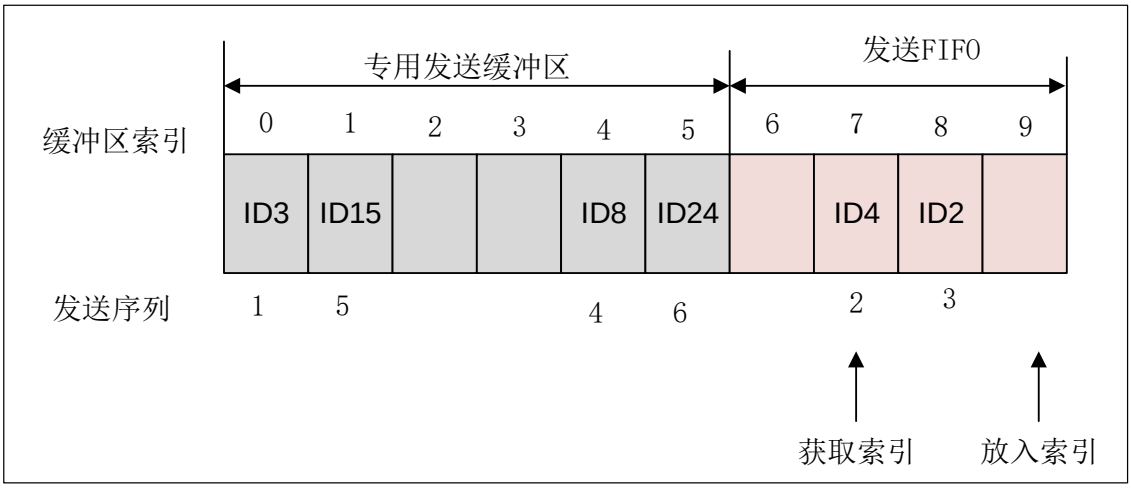


Figure 34-9 Example of mixed configuration of dedicated send buffer and send FIFO

Send priority:

–Scan all dedicated send buffers with activated send requests and the send FIFO buffer with the earliest pending send requests (the buffer pointed to by FDCAN_TXEFS.TFGI)

– The buffer with the smallest message ID has the highest priority, and the message of this buffer will be sent next time

34.2.5.6 Mixing dedicated send buffers and send queues

In this case, the send buffer in the message RAM is divided into a set of dedicated send buffers and a send queue. The number of dedicated send buffers is configured via FDCAN_TXBC.NDTB. The number of send queue buffers is configured via FDCAN_TXBC.TFQS. If FDCAN_TXBC.TFQS is configured to 0, only the dedicated send buffer will be used.

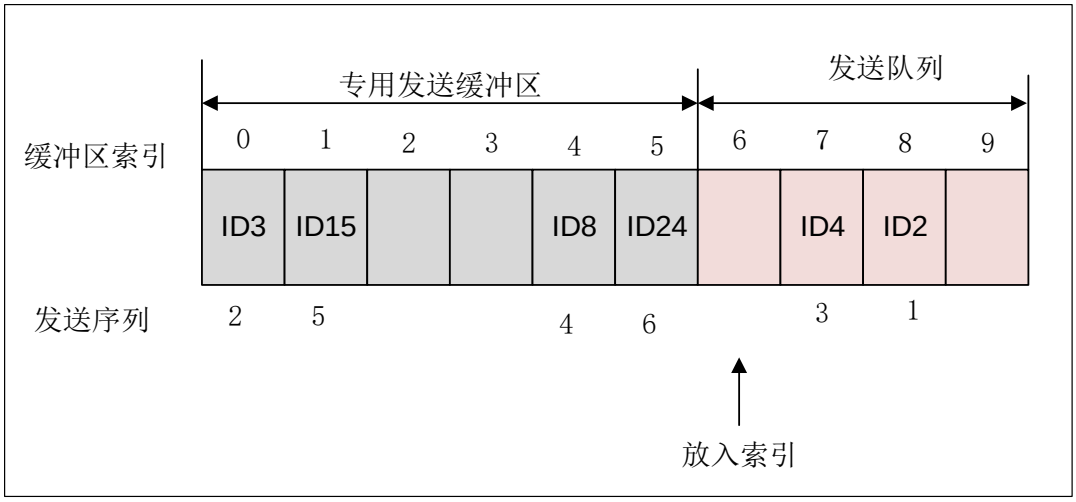


Figure 34-10 Example of mixed configuration of dedicated send buffer and send queue

Send priority:

– Scan all dedicated send buffers and send queue buffers where send requests are activated

– The sending buffer with the smallest message ID has the highest priority, and the message of this buffer will be sent next time

34.2.5.7 Send Cancel

FDCAN supports the send cancel function. This feature is particularly suitable for gateway applications and AUTOSAR-based applications. To cancel a send request for the dedicated send buffer or send queue buffer, the host must write 1 to the corresponding bit in register FDCAN_TXBCR (the bit corresponding to the send buffer index). Send Cancel does not apply to Send FIFO mode.

After successful cancellation, the corresponding bit of register FDCAN_TXBCF is set to 1.

If a request is made to cancel the transmission during the buffer transmission, the corresponding bit of FDCAN_TXBRP will remain set to 1 as long as the transmission is in progress. If the transmission is successful, the corresponding bits of FDCAN_TXBTO and FDCAN_TXBCF are set to 1. If the transmission is unsuccessful, the transmission will not be repeated, only the corresponding position bit of FDCAN_TXBCF will be transferred.

Notes:

If a pending transmission is cancelled immediately before transmission is started, even if another message is pending in that node, the cancellation will be followed by a short window during which no transmission will be started. In this way, another node can send messages that may have a lower priority than the second message in that node.

34.2.5.8 Send event handling

FDCAN supports the handling of send events with send event FIFO. After FDCAN sends a message on the CAN bus, the message ID and timestamp are stored in the Send Event FIFO element. In order to associate the send event with the send event FIFO element, the message tag in the sent send buffer is copied into the send event FIFO element.

The Send Event FIFO can be configured with up to 32 elements, and the chapter [Send Event FIFO Elements] describes the Send Event FIFO elements.

The purpose of the transmit event FIFO is to separate the processing of the transmit status information from the processing of the transmit messages, that is, let the transmit buffer only hold the messages to be sent, and store the transmit status separately in the transmit event FIFO. This has advantages, especially when dealing with dynamically managed send queues, where the send buffer can be used for new messages immediately after successful sending. Before rewriting the transmission buffer, it is not necessary to save the transmission status information of the transmission buffer.

When FDCAN_IR.TEFF indicates that the transmit event FIFO is full, writing of elements to the transmit event FIFO will not continue until at least one element has been read and the transmit event FIFO fetch index has been incremented. If a send event occurs while the send event FIFO is full, this event is discarded and the interrupt flag FDCAN_IR.TEFL is set.

To avoid the Send Event FIFO overflow, you can use the Send Event FIFO watermark. The interrupt flag FDCAN_IR.TEFW is set when the transmit event FIFO fill quantity reaches the transmit event FIFO water mark configured by TXEFC.EFWM.

When reading data from the send event FIFO, you must multiply the send event FIFO fetch index TXEFS.EFGI by 2 (the send event FIFO element size is two words), multiply it by 4, and then add it to the send event FIFO start address TXEFC.EFSA to get the address where the target data is located.

34.2.6 FIFO acknowledgement processing

The acquisition indexes of receive FIF0, receive FIF1 and send event FIFO are controlled by writing the corresponding FIFO acknowledgement index registers, see chapters [FDCAN receive FIFO0 acknowledgement register (FDCAN_RXF0A)], [FDCAN receive FIFO1 acknowledgement register (FDCAN_RXF1A)] and [FDCAN send event FIFO acknowledgement register (FDCAN_TXEFA)]. Writing to the FIFO acknowledgement index sets the FIFO acquisition index to the FIFO acknowledgement index plus 1, which in turn updates the FIFO padding level. There are two use cases:

1. If only one element (the element pointed to by the fetch index) is read from the FIFO, the fetch index value should be written to the FIFO acknowledgement index register.

2. If a sequence of elements has been read from the FIFO, the FIFO acquisition index can be updated only by writing the index value of the last element into the FIFO confirmation index register at the end of the sequence read.

Since the CPU has free access to the message RAM of the FDCAN, special care is needed when reading FIFO elements in any order (regardless of the fetch index), especially when reading high priority messages from one of the two receiving FIFOs. In this case, no write operation should be done to the FIFO acknowledgement index register, otherwise it will cause the fetch index to point to the wrong location, and it will also change the FIFO padding level, and some of the earlier FIFO elements may be lost.

Notes:

The application must ensure that the value written to the FIFO acknowledgement index register is correct, and FDCAN will not check whether the value is correct.

34.2.7 Message RAM

34.2.7.1 Message RAM configuration

The width of the message RAM is 32 bits. The FDCAN module can allocate up to 212 words in the message RAM as shown in the figure below. In practical applications, it is not required that all parts be configured, and there is no limitation on the order between parts.

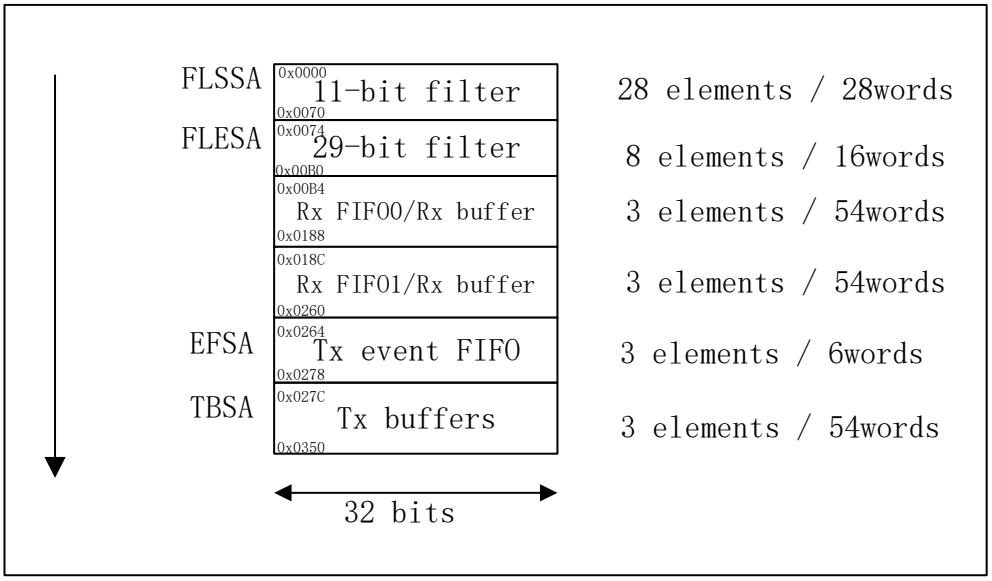


Figure 34-11 Message RAM Configuration

When FDCAN accesses message RAM, it needs to be addressed by 32-bit aligned addresses, not byte aligned addresses. The configurable start address is a 32-bit word address, i.e. only 15 to 2 bits of the address are evaluated, ignoring the two least significant bits.

Notes:

FDCAN does not check whether there are errors in the configuration of message RAM, so you must pay more attention when configuring the start address of different parts and the number of elements in each part to avoid data intrusion or loss.

34.2.7.2 Receive buffer and FIFO element

Up to 2 receive FIFO/receive buffers can be configured in the message RAM. Each receive FIFO/receive buffer can be configured to store up to 3 messages. The structure of the receiving buffer and the receiving FIFO element is shown in the following table. Please see the following table for related instructions. The element size can be configured through the register FDCAN_RXESC, and the data segment of the CAN FD message can be up to 64 bytes.

Table 34-7 receive buffer and FIFO element

Bit	**31 24**			**23 16**				**15 8**	**7 0**
R0	ESI	XTD	RTR	ID [28: 0]					
R1	ANMF	FIDX [6: 0]		Res	FDF	BRS	DLC [3: 0]	RXTS [15: 0]	
R2	DB3 [7: 0]			DB2 [7: 0]				DB1 [7: 0]	DB0 [7: 0]
R3	DB7 [7: 0]			DB6 [7: 0]				DB5 [7: 0]	DB4 [7: 0]
...	...			...				...	...
Rn	DBm [7: 0]			DBm-1 [7: 0]				DBm-2 [7: 0]	DBm-3 [7: 0]

Table 34-8 Receive Buffer and FIFO Element Description

Bitfield	mark	Description
R0[31]	ESI	Error State Indicator 0: The sending node is in an error active state 1: The sending node is in an error passive state
R0[30]	XTD	The Extended Identifier indicates to the host whether the ID of the received frame is a standard ID or an extended ID. 0: 11-bit standard ID1: 29-bit extended ID
R0[29]	RTR	A Remote Transmission Request indicates to the host whether a data frame or a remote frame is received. 0: The received frame is a data frame 1: The received frame is a remote frame Note: There is no remote frame in CAN FD. In the CAN FD frame (FDF = 1), the dominant bit RRS (Remote Request Sub stitution) replaces the RTR bit.
R0 [28: 0]	ID [28: 0]	ID (Identifier) The standard ID or extended ID depends on the XTD bit, and the standard ID is stored in ID [28: 18].
R1[31]	ANMF	Accepted Non-matching Frame(Accepted Non-matching Frame) may enablereception of non-matching frames throughFDCAN_GFC.ANFSandFDCAN_GFC.ANFE. 0 : The received frame matches the filter element pointed to by the filter index FIDX 1: The received frame does not match any of the received filter elements
R1 [30: 24]	FIDX [6: 0]	Filter Index 0 ~ 127: The index of the matching receive filter element (invalid when ANMF = 1) ranges from 0 to FDCAN_SIDFC.LSS] minus1 or FDCAN_XIDFC.LSEminus1.
R1[21]	FDF	FD Format 0: Classic CAN frame format 1: CAN FD frame format (new DLC encoding and CRC)
R1[20]	BRS	Bit Rate Switch 0 : Received frame has no Bit Rate Switch 1: Received frame has Bit Rate Switch
R1 [19:16]	DLC [3: 0]	Data Length Code (Data Length Code) 0 ~ 8: Classic CAN and CAN FD: The received frame contains 0 ~ 8 data bytes 9 ~ 15: CAN: The received frame contains 8 data bytes 9 ~ 15: CAN FD: The received frame contains 12/16/20/24/32/48/64 data bytes
R1 [15: 0]	RXTS [15: 0]	Receive Timestamp (Rx Timestamp) Timestamp counter value captured when SOF (Start Of Frame) is received. The resolution depends on the configuration ofthe timestamp counter prescaler FDCAN_TSCC.TCP.
R2 [31: 24]	DB3 [7: 0]	Data bytes 3 (Data Byte 3)
R2 [23: 16]	DB2 [7: 0]	Data bytes 2 (Data Byte 2)
R2 [15: 8]	DB1 [7: 0]	Data bytes 3 (Data Byte 1)
R2 [7: 0]	DB0 [7: 0]	Data bytes 2 (Data Byte 0)
R3 [31: 24]	DB7 [7: 0]	Data bytes 7 (Data Byte 7)
R3 [23: 16]	DB6 [7: 0]	Data bytes 6 (Data Byte 6)
R3 [15: 8]	DB5 [7: 0]	Data bytes 5 (Data Byte 5)
R3 [7: 0]	DB4 [7: 0]	Data bytes Data Byte 4
...	...	...
Rn [31: 24]	DBm [7: 0]	Data bytes m (Data Byte m)
Rn [23:16]	DBm-1 [7: 0]	Data bytes m-1 (Data Byte m-1)
Rn [15: 8]	DBm-2 [7: 0]	Data bytes m-2 (Data Byte m-2)
Rn [7: 0]	DBm-3 [7: 0]	Data bytes m-3 (Data Byte m-3)

34.2.7.3 Send buffer element

The send buffer can be configured as a dedicated send buffer and a send FIFO (or queue). If the transmit buffer is shared by the dedicated transmit buffer and the transmit FIFO (or queue), the dedicated transmit buffer is at the beginning of the transmit buffer, followed by the transmit FIFO (or queue). The transmit processing unit evaluates the transmit buffer configurations FDCAN_TXBC.TFQS and FDCAN_TXBC.NDTB to distinguish between a dedicated transmit buffer and a transmit FIFO (queue). The structure of the send buffer elements is shown in the following table. Please see the following

table for relevant instructions. The element size can be configured through the register FDCAN_TXESC, and the data segment of the CAN FD message can be up to 64 bytes.

Table 34-9 Send Buffer Element

Bit	**31 24**			**23 16**						**15 8**	**7 0**
T0	ESI	XTD	RTR	ID [28: 0]							
T1	MM [7: 0]			EFC	Res	FDF	BRS	DLC [3: 0]	MM [15: 8]		Res
T2	DB3 [7: 0]			DB2 [7: 0]						DB1 [7: 0]	DB0 [7: 0]
T3	DB7 [7: 0]			DB6 [7: 0]						DB5 [7: 0]	DB4 [7: 0]
...	...			...						...	...
Tn	DBm [7: 0]			DBm-1 [7: 0]						DBm-2 [7: 0]	DBm-3 [7: 0]

Table 34-10 Send Buffer Element Description

Bitfield	**mark**	**Description**
T0[31]	ESI	Error State Indicator 0: ESI bits in CAN FD format only depend on error passive flag 1: ESI bits in CAN FD format are sent as implicit bits Note: The ESI bits of the transmit buffer are OR with the error passive flag to determine the value of the FD frame ESI bits to be sent. According to the requirements of the CAN FD protocol specification, the error active node can choose to send the recessive ESI bit, but the error passive node will always send the recessive ESI bit.
T0[30]	XTD	Extended Identifier 0: 11 bits Standard ID1: 29 bits Extended ID
T0[29]	RTR	Remote Transmission Request 0: Send Data Frame 1: Send Remote Frame Note: WhenRTR = 1, even ifFDCAN_CCCR.FDOE enables transmission in CAN FDformat,FDCANwill send the remote frame according toISO 11898-1: 2015.
T0 [28: 0]	ID [28: 0]	ID (Identifier) Standard ID or extended ID, depending on the XTD bit, the standard ID must be written in ID [28: 18].
T1 [31: 24]	MM [7: 0]	Message tagging (Message Marker) Written by the CPU during send buffer configuration. Copies into the Send Event FIFO element to identify the send message status.
T1[23]	EFC	Event FIFO Control 0: Send event not stored 1: Send event stored
T1[21]	FDF	FD Format 0: Send frame in classic CAN format 1: Send frame in CAN FD Format
T1[20]	BRS	Bit Rate Switch 0: CAN FD frame transmission does not switch bit rate 1: CAN FD frame transmission switch bit rate Note: ESI,FDF, andBRS bits are evaluated only whenCAN FDis enabled (FDCAN_CCCR.FDOE = 1). The BRS bitis evaluated only whenFDCAN_CCCR.BRSE = 1 is additionally set.
T1 [19:16]	DLC [3: 0]	Data Length Code (Data Length Code) 0 ~ 8: Classic CAN and CAN FD: Send frame contains 0 ~ 8 data bytes 9 ~ 15: CAN: Send frame contains 8 data bytes 9 ~ 15: CAN FD: Send frame contains 12/16/20/24/32/48/64 data bytes
T1 [15: 8]	MM [15: 8]	The High byte of Wide Message Marker is written by the CPU during send buffer configuration. Copies into the Send Event FIFO element to identify the send message status. Valid only if FDCAN_CCCR.WMM = 1.
T2 [31: 24]	DB3 [7: 0]	Data bytes 3 (Data Byte 3)
T2 [23:16]	DB2 [7: 0]	Data bytes 2 (Data Byte 2)

T2 [15: 8]	DB1 [7: 0]	Data bytes 3 (Data Byte 1)
T2 [7: 0]	DB0 [7: 0]	Data bytes 2 (Data Byte 0)
T3 [31: 24]	DB7 [7: 0]	Data bytes 7 (Data Byte 7)
T3 [23:16]	DB6 [7: 0]	Data bytes 6 (Data Byte 6)
T3 [15: 8]	DB5 [7: 0]	Data bytes 5 (Data Byte 5)
T3 [7: 0]	DB4 [7: 0]	Data bytes Data Byte 4
...	...	...
Tn [31: 24]	DBm [7: 0]	Data bytes m (Data Byte m)
Tn [23:16]	DBm-1 [7: 0]	Data bytes m-1 (Data Byte m-1)
Tn [15: 8]	DBm-2 [7: 0]	Data bytes m-2 (Data Byte m-2)
Tn [7: 0]	DBm-3 [7: 0]	Data bytes m-3 (Data Byte m-3)

34.2.7.4 Send Event FIFO Element

Each element stores information related to the sent message. By reading the send event FIFO, the host CPU obtains this information in the order in which the messages were sent. Information on the status of the transmit event FIFO can be obtained from the register FDCAN_TXEFS. The structure of the Send Event FIFO element is shown in the following table. Please see the following table for related instructions.

E1A: When FDCAN_CCCR.WMM = 0, E1A.TXTS [15: 0] holds a 16-bit timestamp generated by the internal timestamp logic of FDCAN.

E1B: When the 16-bit wide message flag is enabled (FDCAN_CCCR.WMM = 1), E1B. MM [15: 8] is the upper 8 bits of the wide message flag.

Table 34-11 Send Event FIFO Element

Bit	**31 24**				**23 16**				**15 8**	**7 0**
E0	ESI	XTD	RTR	ID [28: 0]						
E1A	MM [7: 0]				EF [1: 0]	FDF	BRS	DLC [3: 0]	TXTS [15: 0]	
E1B	MM [7: 0]				EF [1: 0]	FDF	BRS	DLC [3: 0]	MM [15: 8]	Res

Table 34-12 Send Event FIFO Element Description

Bitfield	**mark**	**Description**
E0[31]	ESI	Error State Indicator 0: The sending node is in an error active state 1: The sending node is in an error passive state
E0[30]	XTD	Extended Identifier 0: 11 bits Standard ID1: 29 bits Extended ID
E0[29]	RTR	Remote Transmission Request 0: The sending frame is a data frame 1: The sending frame is a remote frame
E0 [28: 0]	ID [28: 0]	Identifier Standard ID or extended ID, depending on the XTD bit. The standard ID is stored in ID [28: 18].
E1A/B [31: 24]	MM [7: 0]	The Message Marker is copied from the send buffer to the send event element and is used to identify the send message status.
E1A/B [23: 22]	EF [1: 0]	Event Type 00: Reserve 01: Send Event 10: Send after cancellation (always this value when sending in DAR mode) 11: Reserve
E1A/B [21]	FDF	FD Format 0: Classic CAN Frame Format 1: CAN FD Frame Format (new DLC encoding and CRC)
E1A/B [20]	BRS	Bit Rate Switch 0: There is no Bit Rate Switch when

		CAN FD frame is sent 1: There is a Bit Rate Switch when CAN FD frame is sent
E1A/B [19:16]	DLC [3: 0]	Data Length Code (Data Length Code) 0 ~ 8: Classic CAN and CAN FD: Send frame contains 0 ~ 8 data bytes 9 ~ 15: Classic CAN: Send frame contains 8 data bytes 9 ~ 15: CAN FD: Send frame contains 12/16/20/24/32/48/64 data bytes
E1A [15: 0]	TXTS [15: 0]	Send Timestamp (Tx Timestamp) Timestamp counter value captured when sending SOF (Start Of Frame). The resolution depends on the configuration ofthe timestamp counter prescaler FDCAN_TSCC.TCP.
E1B [15: 8]	MM [15: 8]	The High byte of Wide Message Marker is written by the CPU during send buffer configuration. Copies into the Send Event FIFO element to identify the send message status.

34.2.7.5 Standard Message ID Filter Element

Up to 128 filter elements (1 word in size each) can be configured for an 11-bit standard ID. Its address is the filter element index (0 ~ 127) multiplied by 4 (converted to bytes), plus the starting address FDCAN_SIDFC.FLSSA. The structure of the standard message ID element is shown in the following table. Please see the following table for related descriptions.

Table 34-13 Standard Message ID Filter Element

Bit	**31 24**		**23 16**	**15 8**	**7 0**
S0	SFT [1: 0]	SFEC [2: 0]	SFID1 [10: 0]	Res	SFID2 [10: 0]

Table 34-14 Standard Message ID Filter Element Description

Bitfield	**mark**	**Description**
S0 [31: 30]	SFT [1: 0]	Standard Filter Type 00: Range filter from SFID1 to SFID2 (SFID2 ≥ SFID1) 01: Dual ID filter defined by SFID1 or SFID2 10: Classic filter: SFID1 = ID, SFID2 = mask 11: Disable this filter element Note: When ***SFT = 0b11***, this filter element is disabled and receive filtering continues (same behavior as when ***SFEC = 0b000***).
SO [29: 27]	SFEC [2: 0]	Standard Filter Element Configuration All enabled filter elements are used for receive filtering of standard frames. When an enabled filter element is matched, or the end of the filter list is reached, receive filtering is stopped. If SFEC = 0b100, 0b101 or 0b110, When a match occurs, the interrupt flag FDCAN_IR.HPM is set and generates an interrupt (if enabled). In this case, the register FDCAN_HPMS is updated to a priority matching state. 000: prohibit this filter element 001: if matched, store into receive FIFO0 010: if matched, store into receive FIFO1 011: if matched, reject ID100: if matched, set priority, not stored 101: if matched, set priority and store into FIFO0 110: if matched, set priority and store into FIFO1 111: store into receive buffer or as debug message, ignore configuration of SFT [1: 0]
SO [26: 16]	SFID [10: 0]	Standard Filter ID1 Standard ID The first ID of the filter element. When filtering for receive buffer or debug messages, this bit field defines the ID of the standard message to store. The received ID must match exactly, without using a masking mechanism.
S0 [10: 0]	SFID2 [10: 0]	Standard Filter ID 2 This bit field has different meanings, depending on the configuration of

		the SFEC:-When SFEC = 0b001 ~ 0b110: second ID of the standard ID filter element -When SFEC = 0b111: filter used to receive buffer or debug messages
	SFID2 [10: 9]	Determines whether the received message is stored in the receive buffer or processed as message A, B, or C in the debug message sequence . 00: Store message in receive buffer 01: Debug message A10: Debug message B11 : Debug message C
	SFID2 [5: 0]	Defines the offset (buffer index) of the receive buffer storing the matching message to the start address RXBC.RBSA.

34.2.7.6 Extended Message ID Filter Element

Up to 64 filter elements (2 words in size each) can be configured for a 29-bit extended ID. When accessing an extended message ID filter element, its address is the filter element index (0-63) multiplied by 2, then multiplied by 4 (converted to bytes), and then the starting address FDCAN_XIDFC.FLESA is added. The structure of the extended message ID element is shown in the following table. Please see the following table for related instructions.

Table 34-15 Extension Message ID Filter Element

Bit	**31 24**		**23 16**	**15 8**	**7 0**
F0	EFEC [2: 0]		EFID1 [28: 0]		
F1	EFT [1: 0]	Res	EFID2 [28: 0]		

Table 34-16 Extension Message ID filter element description

Bitfield	**mark**	**Description**
F0 [31: 29]	EFEC [2: 0]	Extended filter element configuration Extended Filter Element Configuration All enabled filter elements are used for receive filtering of extended frames. When an enabled filter element is matched, or the end of the filter list is reached, receive filtering is stopped. If EFEC = 0b100, 0b101, or 0b110, the on-match interrupt flag FDCAN_IR.HPM is set and an interrupt is generated (if enabled). In this case, the register FDCAN_HPMS is updated to a priority matching state. 000: prohibit this filter element 001: if match, store in receive FIFO0 010: if match, store in receive FIFO1 011: if match, reject ID100: if match, set priority 101: if match, set priority and store in FIFO0 110: if match, set priority and store in FIFO1 111: store in receive buffer or as debug message ignoring configuration of EFT [1: 0]
F0 [28: 0]	EFID [28: 0]	Extended Filter ID1 Extended ID The first ID of the filter element. This bit field defines the ID of the extended message to store when filtering for a receive buffer or debug message. The received ID must match exactly, using only the FDCAN_XIDAM masking mechanism.
F1 [31: 30]	EFT [1: 0]	Extended Filter Type 00: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1) 01: Dual ID filter defined by EFID1 or EFID2 10: Classic filter: EFID1 = ID, EFID2 = mask 11: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1) without FDCAN_XIDAM mask
F1 [28: 0]	EFID2 [28: 0]	Extended Filter ID 2 This bit field has different meanings, depending on the configuration of the EFEC:-When SFEC = 0b001 ~ 0b110: second ID of the extended ID filter element -When SFEC = 0b111: filter used to receive buffer or debug messages
	SFID2 [10: 9]	Determines whether the received message is stored in the receive buffer or processed as message A, B, or C in the debug message sequence. 00: Store message in receive buffer 01: Debug message A10: Debug message B11

		: Debug message C
	SFID2 [5: 0]	Defines the offset (buffer index) of the receive buffer storing the matching message to the start address RXBC.RBSA.

34.3 TIMx registers

FDCAN registers support byte write operations.

Table 34-17 FDCAN Base Address

Pin name	Base address	Description
FDCAN1	0x4000CC00	FDCAN1 base address
FDCAN2	0x4000E000	FDCAN2 base address

34.3.1 FDCAN byte sequence register (FDCAN_ENDN), Address = 0x04

Address offset: 0x004

Reset value: 0x87654321

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	ETV	R	0x87	Byte order test value

34.3.2 FDCAN Data Bit Time and Prescalation Register (FDCAN_DBTP), Address = 0x0C

Address offset: 0x00C

Reset value: 0x00000A33

Write access to this register can only be made if both the FDCAN_CCCR.CCE and FDCAN_CCCR.INIT bits are set.

The programmable range of CAN bit time is from 4 to 49 time slices (Time quanta, tq). The CAN time slice is programmable from 1 to 32 FDCAN communication clock cycles (Minimum time quantum, mtq). tq = (DBRP + 1) mtq.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore, the length of the bit time is either (program value) [DTSEG1 + DTSEG2 + 3] tq or (function value) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

The information processing time (IPT) is zero, which means that the next bit of data is available at the first clock edge after the sampling point.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.								TDC	Res.		DBRP [4: 0]				
-								RW	-		RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			DTSEG1 [4: 0]					DTSEG2 [3: 0]				DSJW [3: 0]			
-			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	0	-
23	TDC	RW	0	Transmitter delay compensation 0: Disable transmitter delay compensation 1: Enable transmitter delay compensation
22: 21	Reserved	-	0	-

20: 16	DBRP [4: 0]	RW	0	Data bitrate prescalation The FDCAN communication clock prescalation value of the bit time slice is generated, the bit time being a multiple of the time slice. The effective value of prescalation is 0 to 31. When TDC = 1, the range is limited to 0 to 1. The actual value used by hardware is Add 1 to the value of the process.
15: 13	Reserved	-	0	-
12: 8	DTSEG1 [4: 0]	RW	0	Data time prior to sampling point The valid values are 0 to 31, and the actual hardware usage value is the programmed value plus 1.
7: 4	DTSEG2 [3: 0]	RW	0	Data time after sampling point The valid values are 1 to 15, and the actual hardware usage value is the programmed value plus 1.
3: 0	DSJW [3: 0]	RW	0	Data (re) synchronization jump width The valid values are 0 to 15, and the actual hardware usage value is the programmed value plus 1.

Note: When the FDCAN communication clock is 8MHz, the reset value 0x0000 0A33 configures the bitrate of the FDCAN data segment to 500Kbps. The CAN FD data segment bit rate configured by DBTP cannot be less than the arbitration segment bit rate configured by NBTP.

34.3.3 FDCAN Test Register (FDCAN_TEST), Address = 0x10

Address offset: 0x010

Reset value: 0x000000X0 (the reset value of bit 7 is determined by the actual level of the FDCAN_RX pin)

Write access to this test register can only be made if the FDCAN_CCCR.TEST bit is set to 1. When the FDCAN_CCCR.TEST bit is cleared, all test register functions are set to their reset value.

Loopback mode and software control FDCAN_TX pin belong to hardware test mode. TX may interfere with message transmission on the CAN bus when configured to not 0b00.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.										SVAL	TXBNS [4: 0]				
-										R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.		PVAL	TXBNP [4: 0]					RX	TX [1: 0]		LBCK	Res.			
-		R	R	R	R	R	R	R	RW		RW	-			

Bit	Name	R/W	Reset Value	Function
31: 22	Reserved	-	0	-
21	SVAL	R	0	Started valid 0: Invalid value of TXBNS 1: The value of TXBNS is valid
20: 16	TXBNS [4: 0]	R	0	Buffer number that started sending The last send buffer number to start sending. When the SVAL bit is set to 1, it is valid, and the valid range is 0 to 31.
15: 14	Reserved	-	0	-
13	PVAL	R	0	Ready to be valid 0: The value of TXBNP is invalid 1: The value of TXBNP is valid
12: 8	TXBNP [4: 0]	R	0	Buffer number ready to send Send buffer number ready to send. When the PVAL bit is set to 1, it is valid, and the valid range is 0 to 31.
7	RX	R	0	Receive pin Actual value of FDCAN_RX pin monitored 0: CAN bus is dominant (FDCAN_RX = 0) 1: CAN bus is implicit (FDCAN_RX = 1)
6: 5	TX [1: 0]	RW	0	Send pin control 00: Reset value, FDCAN_TX pin controlled by CAN core, updated at the end of CAN bit time 01: Sampling point can be monitored at FDCAN_TX pin

				10: FDCAN_TX pin output dominant level (0) 11: FDCAN_TX pin output recessive level (1)
4	LBCK	RW		Loopback mode 0: reset value, loopback mode is prohibited 1: Enable loopback mode (see chapters [External loopback mode] and [Internal loopback mode] for details)
3: 0	Reserved	-	0	-

34.3.4 FDCAN RAM Watchdog Register (FDCAN_RWD), Address = 0x14

Address offset: 0x014

Reset value: 0x0000 0000

The RAM watchdog monitors the READY signal output of the message RAM. Access to message RAM starts a message RAM watchdog counter whose starting value is configured by the FDCAN_RWD.WDC bit.

This counter reloads the value of FDCAN_RWD.WDC when the message RAM indicates that the access has successfully completed by activating its READY signal output. If the message RAM does not respond before the counter is decremented to 0, the counter will stop counting and will interrupt the flag FDCAN_IR.WDI position 1. The RAM watchdog counter is counted by the FDCAN control logic clock.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDV [7: 0]								WDC [7: 0]							
R								R							

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	0	
15: 8	WDV [7: 0]	R	0	Watchdog Count Value Actual value of message RAM watchdog counter
7: 0	WDC [7: 0]	R	0	Watchdog Configuration The starting value of the message RAM watchdog counter. If the initial value of 0x00 is used, the counter is disabled.

34.3.5 FDCAN CC Control Register (FDCAN_CCCR), Address = 0x18

Address offset: 0x018

Reset value: 0x0000 0001

Please refer to the chapter [Software Initialization] for setting conditions of each bit of the register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	WMM	Res.	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	R	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	0	-
15	NISO	RW	0	Non-ISO mode selection 0: Use the CAN FD frame format specified in ISO 11898-1: 2015 1: Use the CAN FD frame format specified in Bosch CAN FD specification V1.0
14	TXP	RW	0	Transmission suspension If this bit is set to 1, FDCAN will pause for two bits after successfully transmitting one frame of data, and then proceed Transmission of the next frame of data. 0: Prohibit transmission suspension 1: Enable transmission suspension
13	EFBI	RW	0	Edge filtering during bus synchronization 0: Edge filtering is prohibited

				1: Two consecutive dominant tqs are needed to detect hard synchronization edges
12	PXHD	RW	0	Protocol exception handling prohibition 0: Enable protocol exception handling 1: Prohibit protocol exception handling Notes: When protocol exception handling is disabled, FDCAN will transmit an error frame when a protocol exception condition is detected.
11	WMM	RW	0	Wide message tagging The 16-bit internal timestamp in the transmit event FIFO is invalid when a 16-bit wide message mark (WMM = 1) is used. 0: Use 8-bit message tag 1: Use a 16-bit message flag to replace the 16-bit timestamp in the transmit event FIFO
10	Reserved	-	0	-
9	BRSE	RW	0	Bitrate switching 0: Bit rate switching when transmission is prohibited 1: Bit rate switching when transmission is enabled Notes: The setting of BRSE does not work when the CAN FD mode is invalid (FDOE = 0).
8	FDOE	RW	0	FD Mode Enable 0: FD mode disabled 1: FD mode enabled
7	TEST	RW	0	Test Mode Enable 0: Normal mode, test register FDCAN_TEST holds reset value 1: Test mode, enabling write access to test register FDCAN_TEST
6	DAR	RW	0	Disable automatic retransmission 0: Enable automatic retransmission of unsuccessfully sent messages 1: Automatic retransmission is prohibited
5	MON	RW	0	Bus listening mode The host can only place the MON position 1 when both the CCE and INIT bits are set to 1. This bit can be reset by the host at any time. 0: Disable bus listening mode 1: Enable bus listening mode
4	CSR	RW	0	Clock Stop Request When the requested clock stops, all pending transmission requests have been completed and the CAN bus has reached an idle state After the state, the INIT bit and CSA bit will be set to 1 in turn. 0: No clock stop requested 1: Clock stop requested
3	CSA	RW	0	Clock Stop Request When the requested clock stops, all pending transmission requests have been completed and the CAN bus has reached an idle state After the state, the INIT bit and CSA bit will be set to 1 in turn. 0: No clock stop requested 1: Clock stop requested
2	ASM	RW	0	Transmission suspension If this bit is set to 1, FDCAN will pause for two bits after successfully transmitting one frame of data, and then proceed Transmission of the next frame of data. 0: Prohibit transmission suspension 1: Enable transmission suspension
1	CCE	RW	0	Configuration Change Enable 0: CPU has no write access to protected configuration registers 1: CPU has write access to protected configuration registers (when INIT = 1)
0	INIT	RW	0	0: Working normally 1: Start initialization Notes: Due to the synchronization mechanism between the two clock domains, the value of INIT may have some delay after writing before it can be read correctly. Therefore, before resetting the INIT value, verify that the previous setting value has indeed been written.

34.3.6 FDCAN Nominal Bit Time and Prescalation Register (FDCAN_NBTP), Address = 0x1C

Address offset: 0x01C

Reset value: 0x0600 0A03

Write access to this register can only be made if both the FDCAN_CCCR.CCE and FDCAN_CCR.INIT bits are set to 1. The programmable range of CAN nominal bit time is 4 to 385 time slices (Time quanta, tq). The CAN time slice is programmable from 1 to 512 FDCAN communication clock cycles (Minimum time quantum, mtq). tq = (NBRP + 1) mtq. NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore, the length of the bit time is either (program value) [NTSEG1 + NTSEG2 + 3] tq or (function value) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

The information processing time (IPT) is zero, which means that the next bit of data is available at the first clock edge after the sampling point.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NSJW [6: 0]							NBRP [8: 0]								
RW							RW								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NTSEG1 [7: 0]								Res.	NTSEG2 [6: 0]						
RW								-	RW						

Bit	Name	R/W	Reset Value	Function
31: 25	NSJW [6: 0]	RW	0x6	The nominal (heavy) synchronous jump width valid values are 0 to 127, and the actual hardware usage value is the programmed value plus 1.
24: 16	NBRP [8: 0]	RW	0	Nominal bitrate prescalation The FDCAN communication clock prescalation value of the bit time slice is generated, the bit time being a multiple of the time slice. Pre-division The frequency effective values are 0 to 511, and the actual hardware usage value is the programmed value plus 1.
15: 8	NTSEG1 [7: 0]	RW	0xA	Nominal time period before sampling point The valid values are 1 to 255, and the actual hardware usage value is the programmed value plus 1.
7	Reserved	-	0	-
6: 0	NTSEG2 [6: 0]	RW	0x3	Nominal time period after sampling point The valid values are 1 to 127, and the actual hardware usage value is the programmed value plus 1.

34.3.7 FDCAN Timestamp Counter Configuration Register (FDCAN_TSCC), Address = 0x20

Address offset: 0x020

Reset value: 0x0000 0000

This register is used to configure an internal 16-bit timestamp counter. For an introduction to internal timestamp processing, please refer to chapter [Timestamp Generation].

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCP [3: 0]			
-												RW			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSS [1: 0]	
-														R	

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	0	
19: 16	TCP [3: 0]	RW	0	Timestamp counter prescalation Configure the timestamp and timeout counter time units to be 1 to 16 times the CAN bit time.

				The effective setting value is 0 ~ 15, and the actual hardware usage value is the programmed value plus 1.
15: 2	Reserved	-	0	
1: 0	TSS [1: 0]	R	0	Timestamp selection 00: The value of the timestamp counter is always 0x0000 01: Timestamp counter value is incremented according to TCP 10 Prohibition setting 11: The value of the timestamp counter is always 0x0000 (equivalent to the set value "00")

34.3.8 FDCAN Timestamp Counter Value Register (FDCAN_TSCV), Address = 0x24

Address offset: 0x024

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TSC [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	0	-
15: 0	TSC [15: 0]	RW	0	Timestamp counter The internal timestamp counter value captured when a SOF (Start Of Frame) is received or sent. When FDCAN_TSCC.TSS = 0b01, the timestamp counter is incremented by 1 to 16 times of the CAN bit time, and the specific multiple is determined by the set value of FDCAN_TSCC.TCP. When counting rollbacks, the interrupt flag FDCAN_IR.TSW is set to 1. A write will clear the counter. Notes: 1. "Roll-back" means that the value of the timestamp counter changes from non-zero to zero, and is not cleared due to the write operation of FDCAN_TSCV. 2. During byte access, writing to any byte of the register will cause the timestamp register to be cleared.

34.3.9 FDCAN timeout counter configuration register (FDCAN_TOCC), Address = 0x28

Address offset: 0x028

Reset value: 0xFFFF 0000

For an introduction to the timeout counter, please refer to the chapter [Timeout Counter] chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOP [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.													TOS [1: 0]		ETOC
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	TOP [15: 0]	RW	0	Timeout period The starting value of the timeout counter (decrement counter), configuring the timeout period.
15: 3	Reserved	-	0	-
2: 1	TOS [1: 0]	RW	0	Timeout selection When operating in continuous mode, write access to FDCAN_TOCV presets the counter to the value configured by FDCAN_TOCC.TOP and continues to decrement down. When the timeout counter is controlled by one of the FIFOs, a null FIFO presets the counter to the value configured by FDCAN_TOCC.TOP. When the first FIFO element is stored, the counter begins to count down. 00: continuous mode 01: Timeout controlled by send event FIFO

				10: Timeout controlled by receive FIFO 0 11: Timeout controlled by receive FIFO1
0	ETOC	RW	0	Timeout Counter Enable 0: Disable timeout counter 1: Enable timeout counter

34.3.10 FDCAN Timeout Counter Value Register (FDCAN_TOCV), Address = 0x2C

Address offset: 0x02C

Reset value: 0x0000 FFFF

For an introduction to the timeout counter, please refer to the chapter [Timeout Counter] chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	0	
15: 0	TOC [15: 0]	RW	0xFFFF	Timeout counter The timeout counter is decremented by 1 to 16 times the CAN bit time, and the specific multiple is determined by the set value of FDCAN_TSCC.TCP. When the counter is decremented to zero, the interrupt flag bit FDCAN_IR.TOO is set to 1 and the counter stops counting. Start and reset (or restart) conditions can be set via FDCAN_TOCC.TOS. Notes: On byte access, when FDCAN_TOCC.TOS = 0b00, writing to any byte of this register presets the timeout counter to the value configured by FDCAN_TOCC.TOP.

34.3.11 FDCAN Error Counter Register (FDCAN_ECR), Address = 0x40

Address offset: 0x040

Reset value: 0x0000 0000

For an introduction to the timeout counter, please refer to the chapter [Timeout Counter] chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.								CEL [7: 0]							
-								R							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC [6: 0]							TEC [7: 0]							
R	R							R							

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	0	
23: 16	CEL [7: 0]	R	0	CAN Error Logging When a CAN protocol error causes the transmit error counter TEC or the receive error counter REC to be incremented, the error logging counter CEL is incremented. When the counter value reaches 0xFF, it stops counting. CEL is not incremented when there are only RP positions but REC does not change. CEL increases after increasing REC and TEC. Read access to the CEL resets the counter. When the counter value reaches 0xFF, counting is stopped, and the interrupt flag FDCAN_IR.ELO is set when the TEC or REC is next incremented. Notes: On byte access, reading byte 2 of the register clears the CEL, reading byte 3, 1, or 0 has no effect.
15	RP	R	0	Receive error passive 0: Receive error counter below error passive level 128 1: Receive error counter has reached error passive level 128
14: 8	REC [6: 0]	R	0	Receive error counter Receive the actual state of the error counter, and the count value range is 0 ~ 127.

7: 0	TEC [6: 0]	R	0	Send error counter Send the actual status of the error counter, and the count value range is 0 ~ 255.

Note: If FDCAN_CCCR.ASM is set to 1, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

34.3.12 FDCAN protocol status register (FDCAN_PSR), Address = 0x44

Address offset: 0x044

Reset value: 0x0000 0707

For an introduction to the timeout counter, please refer to the chapter [Timeout Counter] chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.									TDCV [6: 0]						
-									R						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	PXE	RFDF	RBRS	RESI	DLEC [2: 0]			BO	EW	EP	ACT [1: 0]		LEC [2: 0]		
-	R	R	R	R	R			R	R	R	R		R		

Bit	Name	R/W	Reset Value	Function
31: 23	Reserved	-	0	-
22: 16	TDCV [6: 0]	R	0	Transmitter delay compensation value The position of the Secondary Sample Point (SSP) is determined by the sum of the delay value measured between the FDCAN_TX pin and the FDCAN_RX pin and FDCAN_TDCR.TDCO. The SSP position is in the data segment. TDCV is the number of mtqs between the starting point of the transmitted bit and the second sampling point, and the effective value is 0 to 127 mtqs. On byte access, reading byte 2 of the register clears the CEL, reading byte 3, 1, or 0 has no effect.
15	Reserved	-	0	-
14	PXE	R	0	Protocol exception event 0: No protocol exception event occurred since last read access 1: Protocol exception event has occurred The access type is RX: reset when read access is made. Notes: On byte access, reading byte 0 of the register will clear the PXE, reading byte 3/2/1 has no effect.
13	RFDF	R	0	Received CAN FD message The set of this bit is independent of receive filtering. 0: No message in CAN FD format has been received since last cleared by CPU 1: Received message in CAN FD format with FDF flag of 1 Notes: On byte access, reading byte 0 of the register will clear the RFDF, and reading byte 3/2/1 has no effect.
12	RBRS	R	0	BRS flag of last received CAN FD message This bit is set with the RFDF regardless of receive filtering. 0: BRS flag of last received CAN FD message is not set 1: The BRS flag of the last received CAN FD message is set Notes: On byte access, reading byte 0 of the register will clear the RBRS, and reading byte 3/2/1 has no effect.
11	RESI	R	0	ESI flag of last received CAN FD message This bit is set together with the RFDF regardless of the acceptance filter. 0: ESI flag of last received CAN FD message is not set 1: The ESI flag of the last received CAN FD message is set Notes: On byte access, reading byte 0 of the register will clear the RESI, reading byte 3/2/1 has no effect.
10: 8	DLEC [2: 0]	R	0	Last error code of data segment The type of the last error that occurred in the data segment of the CAN FD frame to which the BRS flag is set. Error Code and LEC

				The same. When there is no error in the transmission (reception or transmission) of the CAN FD frame with the BRS flag set, the DLEC is cleared. Notes: On byte access, reading byte 0 of the register sets DLEC to 0b111, reading byte 3/2/1 has no effect.
7	BO	R	0	Buss_Off state 0: FDCAN is not in Bus_Off state 1: FDCAN is in Bus_Off state
6	EW	R	0	Warning status 0: Both error counters are less than the Error_Warning upper limit 96 1: At least one error counter has reached the Error_Warning limit 96
5	EP	R	0	Error passivity 0: FDCAN is in the Error_Active state. FDCAN normally participates in bus communication and sends an active error flag when an error is detected. 1: FDCAN is in Error_Passive state
4: 3	ACT [1: 0]	R	0	Communication status Monitor the communication status of FDCAN. 00: In synchronization, the node is performing CAN communication synchronization 01: Idle, node is neither receiver nor transmitter 10: Receiver, node operates as receiver 11: Transmitter, node works as transmitter Notes: Protocol exception events will set ACT to 0b00
2: 0	LEC [2: 0]	R	0	Last error code 0: No Error. No errors have occurred since successful reception or transmission to reset the LEC. 1: Stuff Error. In the received message, there are more than 5 identical ones in a certain part Bits, this situation is not allowed. 2: Form Error. There is a format error in the fixed format portion of the received frame. 3: ACK Error. The message sent by FDCAN is not acknowledged by other nodes. 4: Bit1 Error. In the process of sending a message (except for the arbitration segment), the device wants to send a hidden Sex level (logic value 1), but the monitored bus level is dominant (logic value 0). 5: Bit0 Error. During the transmission of the message (including ACK bit, active error flag, overload Flag), the device wants to send an explicit level (logical value of data or ID 0), but the detected bus level is Implicit (logical value 1). During bus Bus_Off recovery, each time an 11-bit recessive bit sequence is detected, it is set Set to this state. This allows the CPU to monitor the progress of the bus recovery sequence (indicating that the bus is not stuck in an explicit state Or persistently disturbed). 6: CRC Error. There is an error in the CRC of the received message. CRC received and data received with The calculated CRC does not match. 7: No Change. Any read access to the protocol status register reinitializes the LEC to 7. When the LEC displays a value of 7, it indicates that the protocol status register has not been detected since the last read access by the CPU CAN bus events. Notes: 1. When the CAN FD frame with the BRS flag set has reached the data segment, the next CAN event (error or valid frame) will be displayed in the DLEC instead of the LEC. An error with a fixed padding bit in the CAN FD CRC sequence will appear as a Form Error instead of a Stuff Error. 2. Bus_Off recovery sequence (refer to ISO11898-1: 2015) cannot be set or cleared FDCAN_CCCR.INIT to shorten. If the device enters the Bus_Off state, it automatically sets FDCAN_CCCR.INIT, stopping all bus activity. Once FDCAN_CCCR.INIT is cleared by the CPU, the device will wait

				for 129 bus idles (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the error management counter is reset. During the waiting period after clearing FDCAN_CCCR.INIT, each time an 11-bit recessive sequence is detected, a Bit0 error code is written to FDCAN_PSR.LEC, enabling the CPU to check whether the CAN bus is dominant or continuously disturbed at any time, and to monitor the Bus_Off recovery sequence. FDCAN_ECR.REC was used to count these sequences. 3. During byte operation, reading byte 0 of the register will set LEC to 0b111, and reading byte 3/2/1 has no effect.

34.3.13 FDCAN transmitter delay compensation register (FDCAN_TDCR), Address = 0x48

Address offset: 0x048

Reset value: 0x0000 0000

For an introduction to the timeout counter, please refer to the chapter [Timeout Counter] chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDCO [6: 0]							Res.	TDCF [6: 0]						
-	RW							-	RW						

Bit	Name	R/W	Reset Value	Function
31: 15	Reserved	-	0	-
14: 8	TDCO [6: 0]	RW	0	Transmitter delay compensation SSP offset The offset value refers to the distance between the delay measured from the FDCAN_TX pin to the FDCAN_RX pin and the second sampling point (SSP). The effective setting value is 0 ~ 127mtq.
7	Reserved	-	0	-
6: 0	TDCF [6: 0]	RW	0	Transmitter delay compensation filter window width Define the minimum value of the SSP position. For measurements of transmitter delay, the dominant edge on the FDCAN_RX pin that causes the SSP position to advance will be ignored. This function is enabled when the TDCF setting is greater than the TDCO. The effective setting value is 0 ~ 127mtq.

34.3.14 FDCAN interrupt register (FDCAN_IR), Address = 0x50

Address offset: 0x050

Reset value: 0x0000 0000

When one of the conditions in the following list is detected, the corresponding flag bit is set. Remains set until the host clears the flag bit to zero. All flag bits are cleared by writing 1, and writing 0 has no effect. A hard reset will clear the registers. Whether an interrupt is generated is controlled by the register FDCAN_IE; Which interrupt line generates an interrupt is controlled by the register FDCAN_ILS.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.		ARA	PED	PEA	WDI	BO	EW	EP	ELO	Res.		DRX	TOO	MRAF	TSW
-		RW								-		RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
RW															

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	0	-
29	ARA	RW	0	Access reserved addresses Access to Reserved Address 0: No access to reserved address 1: The reserved address has been accessed
28	PED	RW	0	Data segment protocol error (using data bit time Protocol Error in Data Phase (Data Bit Time isused) 0: No segment protocol error detected

				1: Data segment protocol error detected (FDCAN_PSR.DLEC ≠ 0/7)
27	PEA	RW	0	Arbitration segment protocol error (using nominal bit time Protocol Error in Arbitration Phase (Nominal Bit Time is used) 0: No arbitration segment protocol error detected 1: Arbitration segment protocol error detected (FDCAN_PSR.LEC ≠ 0/7)
26	WDI	RW	0	Watchdog Interrupt 0: No message RAM watchdog event occurred 1: Message RAM watchdog event occurred due to missing READY signal
25	BO	RW	0	Bus_Off Status 0: Bus_Off state has not changed 1: Bus_Off state has changed
24	EW	RW	0	Warning status (Warning Status) 0: Error_Warning status not changed 1: Error_Warning status has changed
23	EP	RW	0	Error passivity (Error Passive) 0: Error_Passive status has not changed 1: Error_Passive status has changed
22	ELO	RW	0	Error record overflow Error Logging Overflow 0: CAN error record counter not overflowing 1: CAN error record counter has overflowed
21:20	Re-served	-	0	-
19	DRX	RW	0	Messages are stored to a dedicated receive buffer Message stored to Dedicated Rx Buffer This flag is set whenever a received message has been stored in a dedicated receive buffer. 0: Reception buffer area not updated 1: At least one message has been stored in the receive buffer
18	TOO	RW	0	Timeout occurred Timeout Occurred 0: No timeout occurred 1: Timeout has occurred
17	MRAF	RW	0	Message RAM Access Failure Set conditions in receive processing: 1) The reception filtering or storage of the received message has not been completed before the arbitration segment of the subsequent message is received. In this case, the reception filtering or message storage is terminated and the reception processing unit will start processing subsequent messages. 2) The message cannot be written to the message RAM, and the message store will terminate. In either case, the FIFO drop-in index is not updated, the new data flag of the dedicated receive buffer is not set, and the partially stored message is overwritten when the next message is stored at that location. Set condition in send processing: 1) The sending processing unit cannot read the message from the message RAM in time. In this case, the message sending will terminate. FDCAN is switched to a restricted operating mode in the event of a failed access to the transmit processing unit (see section [Restricted operating mode] for details). To exit restricted operating mode, the host CPU must reset FDCAN_CCCR.ASM. 0: No message RAM access failure occurred 1: Message RAM access failure occurred
16	TSW	RW	0	Timestamp Wraparound 0: Timestamp counter not rolled back 1: Timestamp counter has been rolled back
15	TEFL	RW	0	Tx Event FIFO Element Lost 0: Send event FIFO element is not missing 1: The FIFO element of the send event is missing; This bit is also set when writing to a send event FIFO with space size 0
14	TEFF	RW	0	Tx Event FIFO Full 0: Send event FIFO is not full 1: Send event FIFO is full
13	TEFW	RW	0	Tx Event FIFO Watermark Reached 0: The fill level of the send event FIFO is below the water mark 1: The fill level of the send event FIFO reaches the water mark
12	TEFN	RW	0	Tx Event FIFO New Entry 0: No change in transmit event FIFO 1: The transmission processing unit has written the transmission event FIFO element

11	TFE	RW	0	Send FIFO Empty (Tx FIFO Empty) 0: The transmission FIFO is not empty 1: Send FIFO is empty
10	TCF	RW	0	Send Cancel Completed (Transmission Cancellation Finished) 0: Transmission cancellation not completed 1: Send cancellation completed
9	TC	RW	0	Send Complete Transmission Completed 0: Transmission not complete 1: Send completed
8	HPM	RW	0	High priority messages High Priority Message 0: No high priority message received 1: High priority message received
7	RF1L	RW	0	Rx FIFO1 Message Lost 0: Received FIFO1 message not lost 1: There is a loss in receiving FIFO1 message; This bit is also set when writing to receive FIFO1 with space size 0
6	RF1F	RW	0	Rx FIFO 1 Full (Rx FIFO 1 Full) 0: Received FIFO1 is not full 1: Received FIFO1 is full
5	RF1W	RW	0	Rx FIFO 1 Watermark Reached 0: The fill level of the received FIFO1 is below the water mark 1: The filling level of the receiving FIFO1 reaches the water mark
4	RF1N	RW	0	Receive a FIFO1 New Message 0: No new message is written to receive FIFO1 1: There is a new message written to receive FIFO1
3	RF0L	RW	0	Rx FIFO 0 Message Lost 0: Received FIFO0 message not lost 1: There is a loss in receiving FIFO0 message; This bit is also set when writing to receive FIFO0 with space size 0
2	RF0F	RW	0	Receive FIFO 0 Full (Rx FIFO 0 Full) 0: Received FIFO 0 is not full 1: Received FIFO 0 is full
1	RF0W	RW	0	Rx FIFO 0 Watermark Reached 0: The fill level of the received FIFO 0 is below the water mark 1: The filling level of the receiving FIFO0 reaches the water mark
0	RF0N	RW	0	Receive Rx FIFO 0 New Message 0: No new message is written to receive FIFO0 1: There is a new message written to receive FIFO0

34.3.15 FDCAN interrupt enable register (FDCAN_IE), Address = 0x54

Address offset: 0x054

Reset value: 0x0000 0000

The setting in the interrupt enable register determines which state changes in the interrupt register (FDCAN_IR) are indicated on the interrupt line.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	Res		DRXE	TOOE	MRAFE	TSWE
-		RW								-		RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
RW															

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	0	-
29	ARAE	RW	0	Access reserved address enable (Access to Reserved Address Enable) 0: Disable interrupt 1: Enable interrupt
28	PEDE	RW	0	Protocol Error Enable for Data Phase Protocol Error in Data Phase Enable 0: Disable interrupt 1: Enable interrupt
27	PEAE	RW	0	Agreement Error Enablement in Arbitration Phase Protocol Error in Arbitration Phase Enable 0: Disable interrupt 1: Enable interrupt

26	WDIE	RW	0	Watchdog interrupt enable (Watchdog Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
25	BOE	RW	0	Bus_Off State Interrupt Enable 0: Disable interrupt 1: Enable interrupt
24	EWE	RW	0	Warning state interrupt enabled (Warning Status Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
23	EPE	RW	0	Error passive interrupt enable (Error Passive Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
22	ELOE	RW	0	Error Log Overflow Interrupt Enable Error Logging Overflow Interrupt Enable 0: Disable interrupt 1: Enable interrupt
21:20	Reserved	-	0	-
19	DRXE	RW	0	Message store to dedicated receive buffer interrupt enable Message stored to Dedicated Rx Buffer Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
18	TOOE	RW	0	Timeout interrupt enable occurred Timeout Occurred Interrupt Enable 0: Disable interrupt 1: Enable interrupt
17	MRAFE	RW	0	Message RAM Access Failure Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
16	TSWE	RW	0	Timestamp rollback interrupt enable (Timestamp Wraparound Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
15	TEFLE	RW	0	Tx Event FIFO Element Lost Interrupt Enable 0: Disable interrupt 1: Enable interrupt
14	TEFFE	RW	0	Tx Event FIFO Full Interrupt Enable 0: Disable interrupt 1: Enable interrupt
13	TEFWE	RW	0	Tx Event FIFO Watermark Reached Interrupt Enable 0: Disable interrupt 1: Enable interrupt
12	TEFNE	RW	0	Tx Event FIFO New Entry Interrupt Enable 0: Disable interrupt 1: Enable interrupt
11	TFEE	RW	0	Tx FIFO Empty Interrupt Enable 0: Disable interrupt 1: Enable interrupt
10	TCFE	RW	0	Send Cancel Completed Interrupt Enable (Transmission Cancellation Finished Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
9	TCE	RW	0	Send Complete Interrupt Enable (Transmission Completed Interrupt Enable) 0: Disable interrupt 1: Enable interrupt
8	HPME	RW	0	High Priority Message Interrupt Enable High Priority Message Interrupt Enable 0: Disable interrupt 1: Enable interrupt
7	RF1LE	RW	0	Rx FIFO1 Message Lost Interrupt Enable 0: Disable interrupt 1: Enable interrupt
6	RF1FE	RW	0	Rx FIFO 1 Full Interrupt Enable 0: Disable interrupt 1: Enable interrupt

5	RF1WE	RW	0	Rx FIFO 1 Watermark Reached Interrupt Enable 0: Disable interrupt 1: Enable interrupt
4	RF1NE	RW	0	Rx FIFO1 New Message Interrupt Enable 0: Disable interrupt 1: Enable interrupt
3	RF0LE	RW	0	Rx FIFO 0 Message Lost Interrupt Enable 0: Disable interrupt 1: Enable interrupt
2	RF0FE	RW	0	Rx FIFO 0 Full Interrupt Enable 0: Disable interrupt 1: Enable interrupt
1	RF0WE	RW	0	Rx FIFO 0 Watermark Reached Interrupt Enable 0: Disable interrupt 1: Enable interrupt
0	RF0NE	RW	0	Rx FIFO 0 New Message Interrupt Enable 0: Disable interrupt 1: Enable interrupt

34.3.16 FDCAN interrupt line select register (FDCAN_ILS), Address = 0x58

Address offset: 0x058

Reset value: 0x0000 0000

An interrupt line select register that assigns an interrupt generated by a specific interrupt flag in the interrupt register (FDCAN_IR) to one of the two interrupt lines. To generate an interrupt, the corresponding interrupt lines must be enabled via FDCAN_ILE.EINT0 and FDCAN_ILE.EINT1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.		ARAL	PEDL	PEAL	WDIL	BOL	EWL	EPL	ELOL	Res.		DRXL	TOOL	MRAFL	TSWL
-		RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	0	-
29	ARAL	RW	0	Access reserved address interrupt line (Access to Reserved Address Line) 0: Select interrupt line 0 1: Select interrupt line 1
28	PEDL	RW	0	Data segment protocol error interrupt line (Protocol Error in Data Phase Line) 0: Select interrupt line 0 1: Select interrupt line 1
27	PEAL	RW	0	Arbitration segment agreement error interrupt line Protocol Error in Arbitration Phase Line 0: Select interrupt line 0 1: Select interrupt line 1
26	WDIL	RW	0	Watchdog interrupt line Watchdog Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
25	BOL	RW	0	Bus_Off State Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
24	EWL	RW	0	Warning status interrupt line Warning Status Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
23	EPL	RW	0	Error passive interrupt line Error Passive Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
22	ELOL	RW	0	Error record overflow interrupt line Error Logging Overflow Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1

21:20	Re-served	-	0	-
19	DRXL	RW	0	Message store to a dedicated receive buffer interrupt line Message stored to Dedicated Rx Buffer Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
18	TOOL	RW	0	Timeout interrupt line occurred Timeout Occurred Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
17	MRAFL	RW	0	Message RAM Access Failure Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
16	TSWL	RW	0	Timestamp rollback interrupt line Timestamp Wraparound Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
15	TEFLL	RW	0	Tx Event FIFO Element Lost Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
14	TEFFL	RW	0	Tx Event FIFO Full Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
13	TEFWL	RW	0	Tx Event FIFO Watermark Reached Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
12	TEFNL	RW	0	Tx Event FIFO New Entry Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
11	TFEL	RW	0	Tx FIFO Empty Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
10	TCFL	RW	0	Send Cancel Completed Break Line (Transmission Cancellation Finished Interrupt Line) 0: Select interrupt line 0 1: Select interrupt line 1
9	TCL	RW	0	Send completion interrupt line (Transmission Completed Interrupt Line) 0: Select interrupt line 0 1: Select interrupt line 1
8	HPML	RW	0	High priority message interrupt line High Priority Message Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
7	RF1LL	RW	0	Rx FIFO1 Message Lost Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
6	RF1FL	RW	0	Rx FIFO 1 Full Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
5	RF1WL	RW	0	Rx FIFO 1 Watermark Reached Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
4	RF1NL	RW	0	Rx FIFO 1 New Message Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
3	RF0LL	RW	0	Rx FIFO 0 Message Lost Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1
2	RF0FL	RW	0	Receive FIFO 0 Full Interrupt Line (Rx FIFO 0 Full Interrupt Line) 0: Select interrupt line 0 1: Select interrupt line 1
1	RF0WL	RW	0	Rx FIFO 0 Watermark Reached Interrupt (Rx FIFO 0 Watermark Reached Interrupt Line) 0: Select interrupt line 0 1: Select interrupt line 1
0	RF0NL	RW	0	Rx FIFO 0 New Message Interrupt Line 0: Select interrupt line 0 1: Select interrupt line 1

34.3.17 FDCAN interrupt line enable register (FDCAN_ILE), Address = 0x5C

Address offset: 0x05C

Reset value: 0x0000 0000

Each of the two interrupt lines connected to the CPU can be enabled or disabled by setting EINT0 and EINT1 respectively.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EINT1	EINT0
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	0	-
1	EINT1	RW	0	Interrupt Line 1 Enable 0: Interrupt line 1 disabled 1: Interrupt line 1 enabled
0	EINT0	RW	0	Interrupt line 0 enabled 0: Interrupt line 0 disabled 1: Interrupt line 0 enabled

34.3.18 FDCAN Global Filter Configuration Register (FDCAN_GFC), Address = 0x80

Address offset: 0x080

Reset value: 0x0000 0000

Global settings for message ID filtering. Global filter configuration, which controls the filtering path of standard messages and extended messages, is shown in Figure 34-5 and Figure 34-6.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ANFS [1: 0]		ANFE [1: 0]		RRFS	RRFE
-										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 6	Reserved	-	0	-
5: 4	ANFS [1: 0]	RW	0	Accept mismatched standard frames Defines how to handle received 11-bit standard ID messages that do not match any element of the filter list. 00: Unmatched standard frame accepted in receive FIFO 0 01: Unmatched standard frame accepted in reception FIFO1 10: Reject mismatched standard frames 11: Reject mismatched standard frames
3: 2	ANFE [1: 0]	RW	0	Accept mismatched extended frames Defines how to handle received 29-bit extended ID messages that do not match any element of the filter list. 00: Accept mismatched extended frame in receive FIFO 0 01: Accepting mismatched extended frame in reception FIFO1 10: Reject mismatched extended frames 11: Reject mismatched extended frame
1	RRFS	RW	0	Deny remote standard frames 0: Remote frames filtered with 11-bit standard ID 1: Reject all remote frames with 11-bit standard ID
0	RRFE	RW	0	Deny remote extension frame 0: Filter remote frames with 29-bit extended ID 1: Deny all remote frames with 29-bit extended ID

34.3.19 FDCAN standard ID filter configuration register (FDCAN_SIDFC), Address = 0x84

Address offset: 0x084

Reset value: 0x0000 0000

Settings for the 11-bit standard ID filter. Standard ID filter configuration, which controls the filter path of standard messages, is shown in Figure 34-5.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSS [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLSSA [15: 2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	0	-
23: 16	LSS [7: 0]	RW	0	Standard ID Filter List Size 0: No standard ID filter 1 ~ 128: Number of standard ID filter elements > 128: A set value greater than 128 is resolved to 128
15: 2	FLSSA [15: 2]	RW	0	Standard ID Filter List Start Address Set the starting address of the standard ID filter list (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	-

34.3.20 FDCAN Extended ID Filter Configuration Register (FDCAN_XIDFC), Address = 0x88

Address offset: 0x088

Reset value: 0x0000 0000

Settings for the 29-bit extended ID filter. Extended ID filter configuration, which controls the filtering path of extended messages, see Figure 34-6.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSE [6: 0]						
-									Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLESA [15: 2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	0	-
22:16	LSE [7: 0]	RW	0	Extended ID Filter List Size 0: No extended ID filter 1-64: Number of extended ID filter elements > 64: A set value greater than 64 is resolved to 64
15: 2	FLESA [15: 2]	RW	0	Extended ID Filter List Start Address The starting address of the extended ID filter list (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	-

34.3.21 FDCAN Extension ID and Mask Register (FDCAN_XIDAM), Address = 0x90

Address offset: 0x090

Reset value: 0x1FFF FFFF

Settings for the 29-bit extended ID filter. Extended ID filter configuration, which controls the filtering path of extended messages, see Figure 34-6.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	EIDM [28:16]												
			RW												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIDM [15: 0]															

RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	0	
28: 0	EIDM [28: 0]	RW	0	Extended ID mask For the reception filtering of the extended frame, the EIDM is AND with the ID of the received frame. Used to mask 29-bit IDs in SAE J1939. The reset value of all bits is set to 1, and the mask is invalid.

34.3.22 FDCAN High Priority Message Status Register (FDCAN_HPMS), Address = 0x94

Address offset: 0x094

Reset value: 0x0000 0000

A message ID filter element configured to generate priority events that updates this register every time it matches.

This can be used to monitor the status of received high-priority messages and provide quick access to these messages.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLST	FIDX [6: 0]							MSI [1: 0]		BIDX [5: 0]					
R	R							R		R					

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	-
15	FLST	R	0	Filter List Indicates a list of matching filters. 0: List of standard filters 1: Extended filter list
14: 8	FIDX [6: 0]	R	0	Filter Index The index of the matched filter element. The range is 0 to FDCAN_SIDFC.LSS minus 1 or FDCAN_XIDFC.LSE minus 1.
7: 6	MSI [1: 0]	R	0	Message memory indicator 00: FIFO not selected 01: FIFO message lost 10: Message stored in FIFO0 11: Message stored in FIFO1
5: 0	BIDX [5: 0]	R	0	Cache area index Stores the index of the received FIFO element of the message. Only valid if MSI [1] = 1.

34.3.23 FDCAN New Data 1 Register (FDCAN_NDAT1), Address = 0x98

Address offset: 0x098

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ND [31: 0]	RW	0	New data This register holds new data flags for receive buffers 0 through 31. When the received frame updates the received buffer, the corresponding flag bit is set to 1. The flag bit remains set to 1 until the host clears the flag bit. The flag bit is cleared by writing a 1 to the corresponding bit, and writing a 0 has no effect. A hard reset will clear the registers. 0: Receive buffer not updated 1: Receive buffer updated with new message

34.3.24 FDCAN New Data 2 Register (FDCAN_NDAT2), Address = 0x9C

Address offset: 0x09C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND [63: 48]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ND [47: 32]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ND [63: 32]	RW	0	New data This register holds new data flags for receive buffers 32 through 63. When the received frame updates the received buffer, the corresponding flag bit is set to 1. The flag bit remains set to 1 until the host clears the flag bit. The flag bit is cleared by writing a 1 to the corresponding bit, and writing a 0 has no effect. A hard reset will clear the registers. 0: Receive buffer not updated 1: Receive buffer updated with new message

34.3.25 FDCAN receives FIFO0 configuration register (FDCAN_RXF0C), Address = 0xA0

Address offset: 0x0A0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F0OM	FOWM [6: 0]							Res.	FOS [6: 0]						
RW	-							-	-						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FOSA [15: 2]														Res.	
-														-	

Bit	Name	R/W	Reset Value	Function
31	F0OM	RW	0	Receive FIFO0 operating mode FIFO0 can work in Blocking mode or Overwrite mode 0: Received FIFO0 is in blocking mode 1: Received FIFO0 is in overlay mode
30:24	FOWM [6: 0]	RW	0	Receive FIFO0 water mark 0: Prohibit reception of FIFO0 water mark interrupt 1 to 64: Received level of FIFO0 water mark interrupt (FDCAN_IR.RF0W) > 64: Prohibit reception of FIFO 0 water mark interrupt
23	Reserved	-	0	
22:16	FOS [6: 0]	RW	0	Receive FIFO0 size 0: No received FIFO 0 1 to 64: Number of received FIFO0 elements > 64: A set value greater than 64 is parsed to 64. The index of the received FIFO0 element is 0 to FOS minus 1
15: 2	FOSA [15: 2]	RW	0	Receive FIFO0 Start Address message The start address of receive FIFO0 in RAM (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	

34.3.26 FDCAN Receive FIFO0 Status Register (FDCAN_RXFOS), Address = 0xA4

Address offset: 0x0A4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	RF0L	F0F	Res.	Res.	FOPI [5: 0]					
-						R	. R			R					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	FOGI [5: 0]						Res.	FOFL [6: 0]						
-		R						-	R						

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	0	-
25	RF0L	R	0	Received FIFO0 message lost This bit is a copy of the interrupt flag FDCAN_IR.RFOL. When FDCAN_IR.RFOL is reset, this bit is also reset 0: Received FIFO0 no message loss 1: There is a message loss when receiving FIFO0; This bit is also set when writing to receive FIFO0 with space size 0 Notes: When FDCAN_RXFOC.FOOM = 1, the oldest message is overwritten and the flag bit is not set.
24	F0F	R	0	Receive FIFO0 full 0: Received FIFO 0 is not full 1: Received FIFO 0 is full
23:22	Reserved	-	0	-
21:16	FOPI [5: 0]	R	0	Receive FIFO0 put in index Receive FIFO0 write index pointer, range 0 ~ 63
15:14	Reserved	-	0	-
13: 8	FOGI [5: 0]	R	0	Receive FIFO0 Get Index Receive FIFO0 read index pointer, range 0 ~ 63
7	Reserved	-	0	-
6: 0	FOFL [6: 0]	R	0	Receive FIFO0 fill level Receive the number of elements stored in FIFO0, ranging from 0 to 64

34.3.27 FDCAN receive FIFO0 acknowledgement register (FDCAN_RXF0A), Address = 0xA8

Address offset: 0x0A8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FOAI [5: 0]					
-										RW					

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	0	-
5: 0	FOAI [5: 0]	RW	0	Receive FIFO0 acknowledgement index After the host reads a message or sequence of messages from the receiving FIFO0, it must write the buffer index of the last element read from the receiving FIFO0 to F0AI. This action sets the receive FIFO0 fetch index FDCAN_RXF0S.F0GI to F0AI plus 1 and updates the FIFO0 fill level FDCAN_RXF0S.F0FL

34.3.28 FDCAN receive buffer configuration register (FDCAN_RXBC), Address = 0xAC

Address offset: 0x0AC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBSA [15: 2]														Res.	Res.
RW														-	

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	-
15: 2	RBSA [15: 2]	RW	0	Start address of receive buffer Configure the start address (32-bit address) of the receive buffer in the message RAM, also used to reference debug messages A, B, C
1: 0	Reserved	-	0	-

34.3.29 FDCAN receives FIFO1 configuration register (FDCAN_RXF1C), Address = 0xB0

Address offset: 0x0B0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F1OM	F1WM [6: 0]							Res.	F1S [6: 0]						
RW	RW							-	RW						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1SA [15: 2]														Res.	
RW														-	

Bit	Name	R/W	Reset Value	Function
31	F1OM	RW	0	Receive FIFO1 Operating Mode FIFO1 can work in Blocking mode or Overwrite mode 0: Received FIFO1 in blocking mode 1: Received FIFO1 is in overlay mode
30:24	F1WM [6: 0]	RW	0	Receive FIFO1 watermark 0: Prohibit reception of FIFO1 water mark interrupt 1 to 64: Received level of FIFO1 water mark interrupt (FDCAN_IR.RF1W) > 64: Prohibit reception of FIFO1 water mark interrupt
23	Reserved	-	0	-
22:16	F1S [6: 0]	RW	0	Receive FIFO1 size 0: No receive FIFO1 1 to 64: Number of received FIFO1 elements > 64: A set value greater than 64 is parsed to 64. The index of the received FIFO1 element is 0 to F1S minus 1
15: 2	F1SA [15: 2]	RW	0	Receive FIFO1 start address The start address of receiving FIFO1 in the message RAM (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	-

34.3.30 FDCAN Receive FIFO1 Status Register (FDCAN_RXF1S), Address = 0xB4

Address offset: 0x0B4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMS [1: 0]		Res.	Res.	Res.	Res.	RF1L	F1F	Res.	Res.	F1PI [5: 0]					
R		-				R	R	-		RW					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	F1GI [5: 0]						Res.	F1FL [6: 0]						
-		R						-	R						

Bit	Name	R/W	Reset Value	Function
31:30	DMS [1: 0]	R	0	Debug message status 00: Idle state, waiting to receive debug messages 01: Received debug message A 10: Received debug message A, B 11: Debug messages A, B, C received
29:26	Reserved	-	0	-
25	RF1L	R	0	Received FIFO1 message lost 0: Received FIFO1 no element missing 1: There is an element missing in the receiving FIFO1; This bit is also set when writing to receive FIFO1 with space size 0 Notes: When FDCAN_RXF1C.F1OM = 1, the oldest message is overwritten and the flag bit is not set.
24	F1F	R	0	Receive FIFO1 full 0: Received FIFO1 is not full 1: Received FIFO1 is full
23:22	Reserved	-	0	-
21:16	F1PI [5: 0]	RW	0	Receive FIFO1 put in index Receive FIFO1 write index pointer, range 0 ~ 63
15:14	Reserved	-	0	-

13: 8	F1GI [5: 0]	R	0	Receive FIFO1 Get Index Receive FIFO1 read index pointer, range 0 ~ 63
7	Reserved	-	0	-
6: 0	F1FL [6: 0]	R	0	Receive FIFO1 fill level Receive the number of elements stored in FIFO1, ranging from 0 to 64

34.3.31 FDCAN receives FIFO1 acknowledgement register (FDCAN_RXF1A), Address = 0xB8

Address offset: 0x0B8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F1AI [5: 0]					
-										RW					

Bit	Name	R/W	Reset Value	Function
31:6	Re-served	-	0	-
5: 0	F1AI	RW	0	Receive FIFO1 acknowledgement index After the host reads a message or sequence of messages from the receiving FIFO1, it must write the buffer index of the last element read from the receiving FIFO1 to F1AI. This action sets the receive FIFO1 fetch index FDCAN_RXF1S.F1GI to F1AI plus 1 and updates the FIFO1 fill level FDCAN_RXF1S.F1FL

34.3.32 FDCAN receive buffer and FIFO element size configuration register (FDCAN_RXESC), Address = 0xBC

Address offset: 0x0BC

Reset value: 0x0000 0000

Configure the number of bytes of data for the receive buffer and the receive FIFO element. Data segments larger than 8 bytes are only available in CAN FD mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	RBDS [2: 0]			Res.	F1DS [2: 0]			Res.	FODS [2: 0]		
-					RW			-	RW			-	RW		

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	0	-
10: 8	RBDS [2: 0]	RW	0	Receive buffer data segment size 000: 8 bytes 001: 12 bytes 010: 16 bytes 011: 20 bytes 100: 24 bytes 101: 32 bytes 110: 48 bytes 111: 64 bytes
7	Reserved	-	0	-
6: 4	F1DS [2: 0]	RW	0	Receive FIFO1 data segment size 000: 8 bytes 001: 12 bytes 010: 16 bytes 011: 20 bytes 100: 24 bytes 101: 32 bytes 110: 48 bytes 111: 64 bytes

3	Reserved	-	0	-
2: 0	FODS [2: 0]	RW	0	Receive FIFO0 data segment size 000: 8 bytes 001: 12 bytes 010: 16 bytes 011: 20 bytes 100: 24 bytes 101: 32 bytes 110: 48 bytes 111: 64 bytes

34.3.33 FDCAN Send Buffer Configuration Register (FDCAN_TXBC), Address = 0xC0

Address offset: 0x0C0

Reset value: 0x0000 0000

Write access to the FDCAN_CCCR register can only be made when both the CCE bit and the INIT bit of the register are set to 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	TFQM	TFQS [5: 0]						Res.	Res.	NDTB [5: 0]					
-	RW	RW						-		RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TBSA [13: 0]														Res.	Res.
RW														-	

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	0	-
30	TFQM	RW	0	Send FIFO/Queue Mode 0: Transmit FIFO mode 1: Send queue mode
29:24	TFQS [5: 0]	RW	0	Send FIFO/Queue Size 0: No transmission FIFO or queue 1-32: Buffer size of sending FIFO queue > 32: A set value greater than 32 is resolved to 32
23:22	Reserved	-	0	-
21:16	NDTB [5: 0]	RW	0	Dedicated send buffer size 0: No dedicated transmit buffer 1-32: Dedicated send buffer size > 32: A set value greater than 32 is resolved to 32
15: 2	TBSA [15: 2]	RW	0	Send buffer start address The start address of the send buffer portion of the message RAM (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	-

Note:

The sum of TFQS and NDTB shall not be greater than 32. FDCAN does not monitor this setting error. The send buffer portion of the message RAM starts with a dedicated send buffer.

34.3.34 FDCAN send FIFO/queue status register (FDCAN_TXFQS), Address = 0xC4

Address offset: 0x0C4

Reset value: 0x0000 0000

The status of the transmit FIFO or queue is related to the pending transmit request in the register FDCAN_TXBRP. As a result, the effect of adding or canceling requests may be delayed due to a running send scan (TXBRP has not been updated yet).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQF	TFQPI [4: 0]				
-										R	R				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	TFGI [4: 0]					Res.	Res.	TFFL [5: 0]					
-			R					-		R					

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	0	-
21	TFQF	R	0	Send FIFO/Queue full 0: Send FIFO or queue is not full 1: Send FIFO or queue is full
20:16	TFQPI [4: 0]	R	0	Send FIFO/Queue Into Index Send a FIFO or queue drop index pointer, range 0-31
15:13	Reserved	-	0	-
12: 8	TFGI [4: 0]	R	0	Send FIFO to get index Send the read index pointer of the FIFO, range 0-31. If configured to send queue mode (FDCAN_TXBC.TFQM = 1), this bit reads 0
7: 6	Reserved	-	0	-
5: 0	TFFL [5: 0]	R	0	Send FIFO Idle Level Number of consecutively idle transmit FIFO elements from TFGI, range 0-32. If configured to send queue mode (FDCAN_TXBC.TFQM = 1), this bit reads 0

Note: When the dedicated transmit buffer is configured in combination with the transmit FIFO or transmit queue, the put index and get index refer to the transmit buffer number from the first dedicated transmit buffer.

For example, for a combined configuration with 12 dedicated send buffers and 20 send FIFO buffers, put index 15 to point to the 4th buffer of the send FIFO.

34.3.35 FDCAN Send Buffer Element Size Configuration Register (FDCAN_TXESC), Address = 0xC8

Address offset: 0x0C8

Reset value: 0x0000 0000

Configures the number of bytes of data to send buffer elements. Data segments larger than 8 bytes are only available in CAN FD mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.													TBDS [2: 0]		
-													RW		

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	0	-
2: 0	TBDS [2: 0]	RW	0	Send buffer data segment size 000: 8 bytes 001: 12 bytes 010: 16 bytes 011: 20 bytes 100: 24 bytes 101: 32 bytes 110: 48 bytes 111: 64 bytes

Note: If the data length of the send buffer element (configured by the DLC) is greater than the data length configured by the TBDS, undefined data bytes in the send buffer will be sent at 0xCC (padding byte).

34.3.36 FDCAN Send Buffer Request Suspend Register (FDCAN_TXBRP), Address = 0xCC

Address offset: 0x0CC

Reset value: 0x0000 0000

Configures the number of bytes of data to send buffer elements. Data segments larger than 8 bytes are only available in CAN FD mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRP [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRP [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	TRP [31: 0]	R	0	Send request pending flag Each send buffer has its own send request suspend bit. These bits are set through the register FDCAN_TXBAR; Resets after the sending of the request has been completed, or the sending of the request has been canceled through the register FDCAN_TXBCR. Only the TXBRP bit corresponding to the transmit buffer configured by FDCAN_TXBC will be set. A send scan is initiated after the TXBRP position bit (see section [Send Processing]) to check for the highest priority pending send requests (i.e., the send buffer with the smallest message ID). Cancelling the send request resets the corresponding TXBRP bit. If the transmission has already started when the request is cancelled, the TXBRP bit is reset at the end of the transmission regardless of whether the transmission is successful or not. After the TXBRP bit is reset, the corresponding cancel request bit will also be reset immediately. After the cancellation is requested, the following conditions indicate through the register FDCAN_TXBCF that the cancellation is complete: • After successful transmission and the corresponding FDCAN_TXBTO position bit • Send has not started when cancelled • Send aborted due to arbitration failure • Error occurred during frame transmission In DAR mode, all sends are automatically cancelled after failure. For all failed transmissions, the corresponding FDCAN_TXBCF bit will be set. 0: No send request pending 1: Send request pending Note: The TXBRP bits set during the transmission scan are ignored during this transmission scan. During this period, if these transmissions are requested to be cancelled, the transmissions are immediately cancelled and the corresponding TXBRP bits are reset.

34.3.37 FDCAN send buffer add request register (FDCAN_TXBAR), Address = 0xD0

Address offset: 0x0D0

Reset value: 0x0000 0000

Configures the number of bytes of data to send buffer elements. Data segments larger than 8 bytes are only available in CAN FD mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	AR [31: 0]	RW	0	Add Request Each send buffer has its own add request bit, write 1 sets the corresponding add request bit, and write 0 has no effect. The host may set send requests for multiple send buffers at a time. Only the TXBAR bit corresponding to the transmit buffer configured by FDCAN_TXBC will be set. These bits are immediately reset when a send scan is not running, otherwise they remain set until the send scan is complete. 0: No send request added 1: Send request added Notes: If the send buffer of the add request already has a send request pending (the corresponding bit of the FDCAN_TXBRP register is set), the add request is ignored.

34.3.38 FDCAN Send Buffer Cancel Request Register (FDCAN_TXBCR), Address = 0xD4

Address offset: 0x0D4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CR [31: 0]	RW	0	Cancel request Each send buffer has its own cancel request bit, write 1 sets the corresponding cancel request bit, and write 0 has no effect. The host may set cancellation requests for multiple send buffers at a time. Only the TXBCR bit corresponding to the transmit buffer configured by FDCAN_TXBC will be set. These bits remain set until the corresponding FDCAN_TXBRP bit is reset. 0: No cancellation request pending 1: There is a cancel request pending

34.3.39 FDCAN Send Buffer Send Occurred Register (FDCAN_TXBTO), Address = 0xD8

Address offset: 0x0D8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TO [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TO [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	TO [31: 0]	R	0	Sending has occurred Each transmit buffer has its own transmit occurred flag bit. When the transmission buffer is completed and the corresponding bit of FDCAN_TXBRP is cleared, the corresponding transmission has occurred flag position bit. When a new transmission is requested to the corresponding bit write 1 of FDCAN_TXBAR, the corresponding transmission has occurred flag bit reset. 0: Transmission not performed 1: Transmission performed

34.3.40 FDCAN Send Buffer Cancel Completed Register (FDCAN_TXBCF), Address = 0xDC

Address offset: 0x0DC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CF [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	CF [31: 0]	R	0	Cancellation Completed Each send buffer has its own Cancel Completed flag bit. When the cancellation requested by FDCAN_TXBCR is completed and the corresponding bit of FDCAN_TXBRP is cleared, the corresponding cancellation completed flag position bit is set. If the corresponding bit of FDCAN_TXBRP is not set at the time of cancellation, the cancellation complete flag bit is set immediately. When a new transmission is requested by writing 1 to the corresponding bit of FDCAN_TXBAR, the corresponding cancellation completed flag bit is reset. 0: No transmit buffer cancelled

				1: Send buffer cancelled

34.3.41 FDCAN Send Buffer Send Interrupt Enable Register (FDCAN_TXBTIE), Address = 0xE0

Address offset: 0x0E0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIE [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIE [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	TIE [31: 0]	RW	0	Send interrupt enable Each send buffer has its own send interrupt enable bit. 0: Disable transmission interrupt 1: Enable transmission interrupt

34.3.42 FDCAN Send Buffer Cancel Completed Interrupt Enable Register (FDCAN_TXBCIE), Address = 0xE4

Address offset: 0x0E4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CFIE [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFIE [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	CFIE [31: 0]	RW	0	Unenable completed interrupts Each send buffer has its own cancel completed interrupt enable bit. 0: Disable cancellation of completed interrupt 1: Enable cancel completed interrupt

34.3.43 FDCAN send event FIFO configuration register (FDCAN_TXEFC), Address = 0xF0

Address offset: 0x0F0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.		EFWM [5: 0]						Res.		EFS [5: 0]					
-		RW						-		RW.					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EFSA [15: 2]														Res	
RW														-	

Bit	Name	R/W	Reset Value	Function
31: 30	Reserved	-	0	-
29: 24	EFWM [5: 0]	RW	0	Event FIFO water mark 0: Prohibit water mark interruption 1 ~ 32: Level of transmission event FIFO water mark interrupt (FDCAN_IR.TEFW) > 32: Water line interruption is prohibited
23: 22	Reserved	-	0	-
21: 16	EFS [5: 0]	RW	0	Event FIFO Size 0: Prohibit transmission of event FIFO 1-32: Number of elements to send event FIFO

				> 32: A set value greater than 32 is resolved to 32 The index of the Send Event FIFO element is 0 to EFS minus 1
15: 2	EFSA [15: 2]	RW	0	Event FIFO start address Start address of the event FIFO in the message RAM (32-bit word address, see Figure 34-11)
1: 0	Reserved	-	0	-

34.3.44 FDCAN send event FIFO status register (FDCAN_TXEFS), Address = 0xF4

Address offset: 0x0F4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						TEFL	EFF	Res			EFPI [4: 0]				
-						R	R	-			R				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	EFGI [4: 0]					Res			EFFL [5: 0]				
-			R					-			-				

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	0	-
25	TEFL	R	0	Send Event FIFO Element Missing This bit is the interrupt flag FDCAN_IR. Copy of TEFL. When FDCAN_IR. When the TEFL is reset, this bit is also reset 0: Send event FIFO no element missing 1: There is an element missing in the sending event FIFO; This bit is also set when writing to a send event FIFO with space size 0
24	EFF	R	0	Event FIFO is full 0: Send event FIFO is not full 1: Send event FIFO is full
23:21	Reserved	-	0	-
20:16	EFPI [4: 0]	R	0	Event FIFO put in index Send event FIFO writes index pointer, range 0 ~ 31
15:13	Reserved	-	0	-
12: 8	EFGI [4: 0]	R	0	Event FIFO Get Index Send event FIFO reads index pointer, range 0 ~ 31
7: 6	Reserved	-	0	-
5: 0	EFFL [5: 0]	R	0	Event FIFO Fill Level Receive the number of elements stored in FIFO0, ranging from 0 to 32

34.3.45 FDCAN Send Event FIFO Acknowledgement Register (FDCAN_TXEFA), Address = 0xF8

Address offset: 0x0F8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EFAI [4: 0]				
-											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 5	Reserved	-	0	-
4: 0	EFAI [4: 0]	RW	0	Event FIFO Acknowledgement Index After the host reads an element or sequence of elements from the sending event FIFO, it must write the index of the last element read from the event FIFO to the EFAI. This action sets the send event FIFO fetch index FDCAN_TXEFS.EFGI to EFAI plus 1 and updates the event FIFO fill level FDCAN_TXEFS.EFFL.

35. Serial Peripheral Interface (SPI/I2S)

35.1 Overview

The SPI interface may be configured to support the SPI protocol or to support the I2S audio protocol. The SPI interface works in SPI mode by default, and functions can be switched from SPI mode to I2S mode through software.

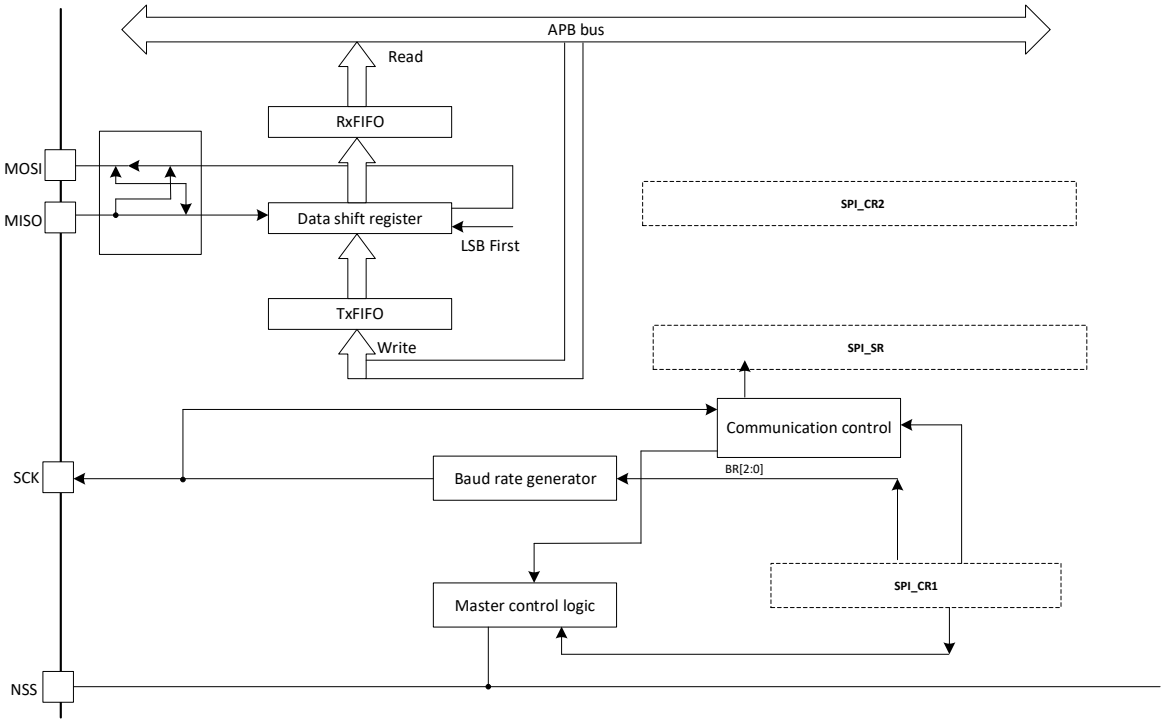


Figure 35-1 SPI block diagram

The Serial Peripheral Interface (SPI) allows the chip to communicate with external devices in half/full duplex, synchronous, serial mode. The interface can be configured as master and in this case it provides the serial clock (SCK) to the external slave device. The interface is also capable of operating in multimaster configuration. It can be used for a variety of purposes, including two-wire simplex synchronous transmission using one bi-directional data line, and reliable communication with CRC verification. I2S is also a 3-pin synchronous serial interface communication protocol. It supports four audio standards, including the Philips I2S standard, the MSB and LSB alignment standards, and the PCM standard. It is in half-duplex communication and can work in master and slave 2 modes. When it acts as a master device, it provides a clock signal to an external slave device through an interface.

35.1.1 SPI features

- 3-wire full-duplex simultaneous transmission
- 2-wire half-duplex synchronous transmission (with bidirectional data line)
- 2-wire simplex synchronous transmission (no bidirectional data line)
- 8 or 16-bit transport frame format selection
- Master or Slave Operation
- Support multi-master mode

- 8 main mode baud rate prescaling coefficients (up to fPCLK/2)
- Slave mode frequency (up to fPCLK/2)
- Both Master and Slave modes can be managed by software or hardware NSS: dynamic change of Master/Slave operating mode
- Programmable clock polarity and phase
- Programmable data order, MSB first or LSB first
- Dedicated transmit and receive flags that can trigger interrupts
- SPI bus busy status flag
- Hardware CRC feature for reliable communication
  - In transmit mode, the CRC value can be transmitted as last byte
  - In full-duplex mode, the last received byte is automatically checked for CRC.
- Primary mode failure, overload, and CRC error flags that can trigger interrupts
- Two embedded Rx and Tx FIFOs with a depth of 4 and a width of 16 bits (8 bits when the data frame is set to 8 bits) with DMA functionality

35.1.2 I2S features

- Half-duplex communication (only transmitter or receiver)
- Master or slave operations
- 8-bit linear programmable prescaler for accurate audio sampling frequency (8-192 kHz)
- Data format may be 16-bit, 24-bit or 32-bit
- The audio channel fixed packet frame is 16-bit (16-bit data frame) or 32-bit (16, 24 or 32-bit data frame)
- Programmable clock polarity (steady state)
- Underflow flag in slave transmit mode and Overflow flag in master/slave receive mode
- 16-bit register for transmission and reception with one data register for both channel sides
- Support I2S protocols:

  -I2S Philips Standard

  -MSB Alignment Standard (Left

  Alignment)-LSB Alignment Standard (Right Alignment)-PCM

  Standard (16-bit channel frame with long or short frame synchronization or 16-bit data frame expansion to 32-bit channel frame)
- Data direction is always MSB first
- DMA capability for transmission and reception
- Master clock may be output to drive an external audio component. Ratio is fixed at 256 × FS (where FS is the audio sampling frequency)

35.2 SPI functional description

35.2.1 Communications between one master and one slave

The SPI allows the MCU to communicate using different configurations, depending on the device targeted and the application requirements. These configurations use 2 or 3 wires (with software NSS management) or 4 wires (with hardware NSS management).

35.2.1.1 Full-duplex communication

By default, the SPI is configured for full-duplex communication. In this configuration, the shift registers of the master and slave are linked using two unidirectional lines between the MOSI and the MISO pins. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master transmits the data to be sent to the slave via the MOSI line and receives data from the slave via the MISO line. When the transmission of the data frame is completed, the information exchange between the master and the slave is terminated.

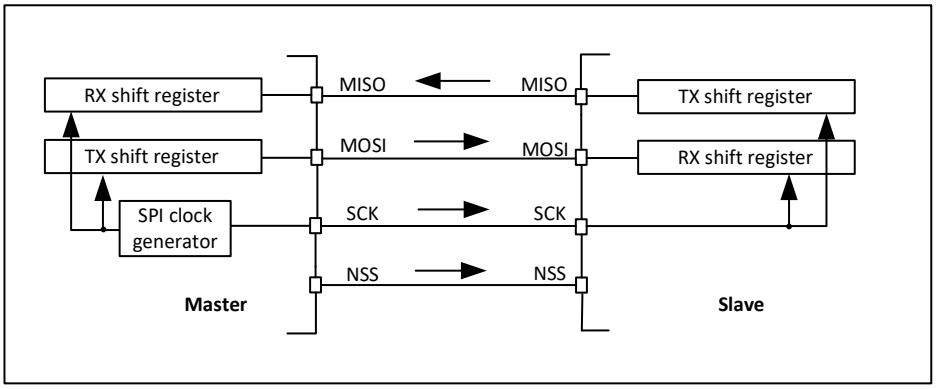


Figure 35-2 Full-duplex single master/ single slave application

35.2.1.2 Half-duplex communication

The SPI can communicate in half-duplex mode by setting the BIDIMODE bit in the SPI_CR1 register. In this configuration, one single cross connection line is used to link the shift registers of the master and slave together. During this communication, the data is synchronously shifted between the shift registers on the SCK clock edge in the transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPI_CR1 registers. In this configuration, the MISO of the master and the MOSI of the slave may be released as general-purpose ports for use by other applications.

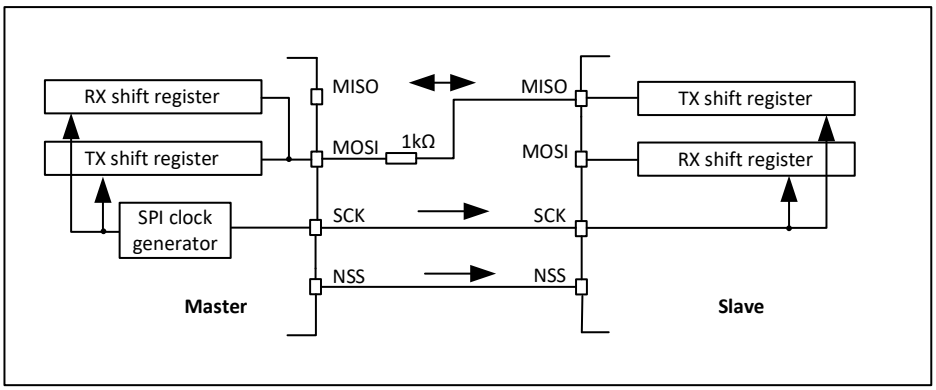


Figure 35-3 Half-duplex single master/ single slave application

The NSS may be used for hardware control between the master and the slave. NSS simplex communication can also be controlled using software.

The SPI can communicate in simplex mode by setting the SPI in transmit-only or in receive-only using the RXONLY bit in the SPI_CR1 register. In this configuration, only one line is used for the transfer between the shift registers of the master and slave. In this configuration, the MISO of the master and the MOSI of the slave may be released as general-purpose ports for use by other applications.

- Receive-only mode (RXONLY=1): The application can disable the SPI output functionby setting the RXONLY bit. In slave configuration, the MISO output is disabled and the pin can be used as a GPIO. The slave continues to receive data from the MOSI pin while its slave select signal is active. In the master configuration, the MOSI output is disabled and the pin can be used as a GPIO. The clock signal is generated continuously as long as the SPI is enabled. The only way to stop the clock is to clear the RXONLY bit or the SPE bit and wait until the incoming pattern from the MISO pin is finished.
- Transmit-only mode (RXONLY=0): The configuration settings are the same as for full- duplex. An unused port may be used as a standard GPIO.

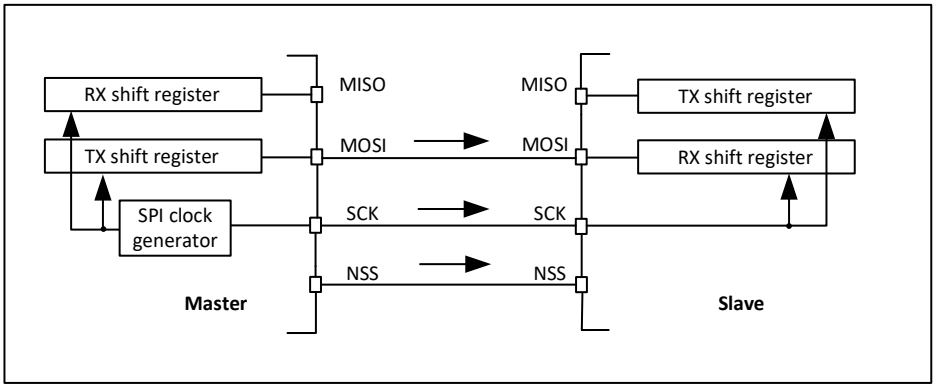


Figure 35-4 Single master/ single slave application

(master in transmit-only/slave in receive-only mode)

(1) NSS can be used for hardware control between the master and the slave. NSS optional software control.

(2) In the standard transmit-only mode, all events related to transmission and reception must be ignored.

(3) In this configuration, both MISO pins can be used as GPIOs.

Note: Set BIDIMODE bidirectional mode, any simplex communication can be changed to half-duplex communication.

35.2.2 Multi-slave communication

In an application scenario with two or more independent slaves, the host uses GPIO to control NSS to manage the slaves. The master must select a slave by pulling down the connected slave NSS. When this is done, communication between the master and the selected slave is established.

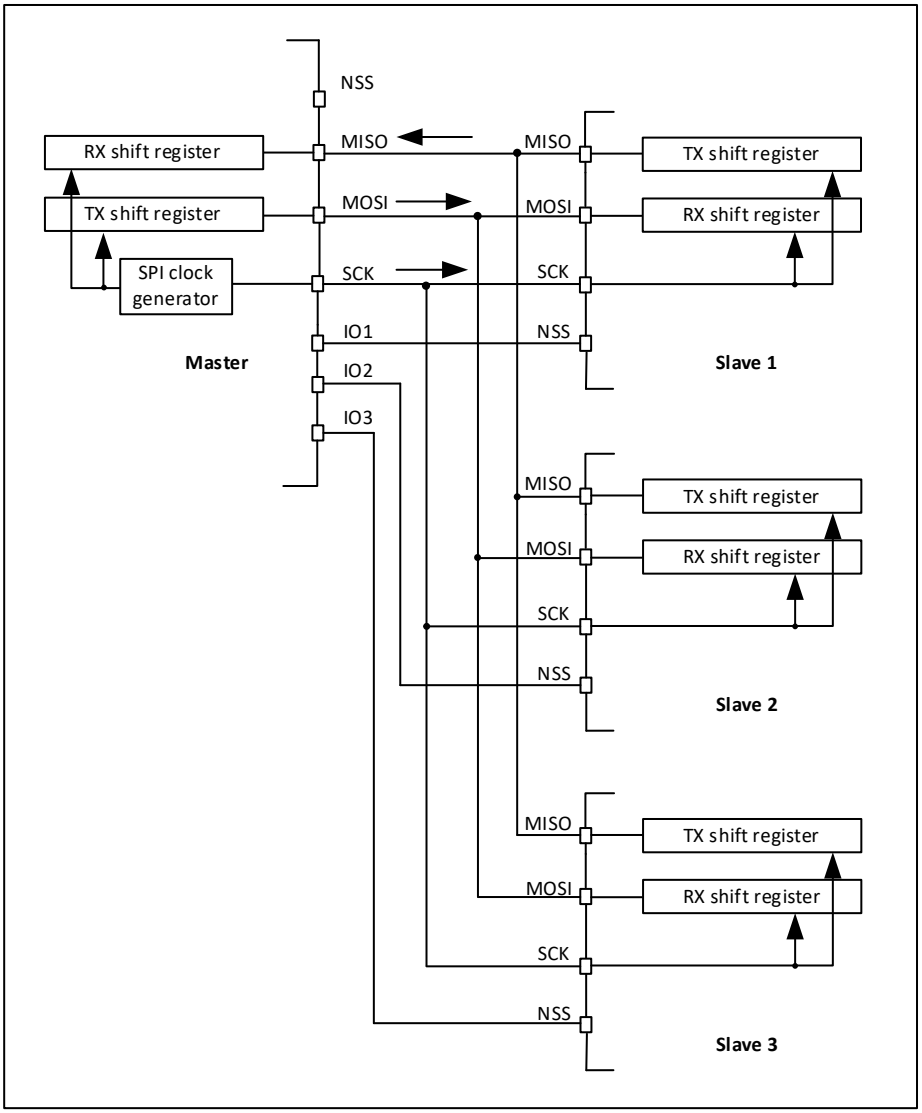


Figure 35-5 Master and three independent slaves

NSS pin is not used on master side at this configuration. It has to be managed internally (SSM=1, SSI=1) to prevent any MODF error.

As MISO pins of the slaves are connected together, all slaves must have the GPIO configuration of their MISO pin set as alternate function open-drain.

35.2.3 Multi-master communication

The SPI bus is designed to have multi-host functionality, and users can use its embedded feature, which can find potential conflicts between two nodes trying to control the bus simultaneously. For this detection, NSS pin is used configured at hardware input mode.

The connection of more than two SPI nodes working at this mode is impossible as only one node can apply its output on a common data line at time.

When nodes are non active, both stay at slave mode by default. Once the node wants to control the bus, it switches to host mode and gives the valid level to the NSS inputs of the remaining nodes via GPIO. After the process is completed, the valid slave select signal is released, and the node controlling the bus temporarily returns to standby and waits for a new process to start.

If both nodes give a control request at the same time, a bus conflict will generate a MODF event.

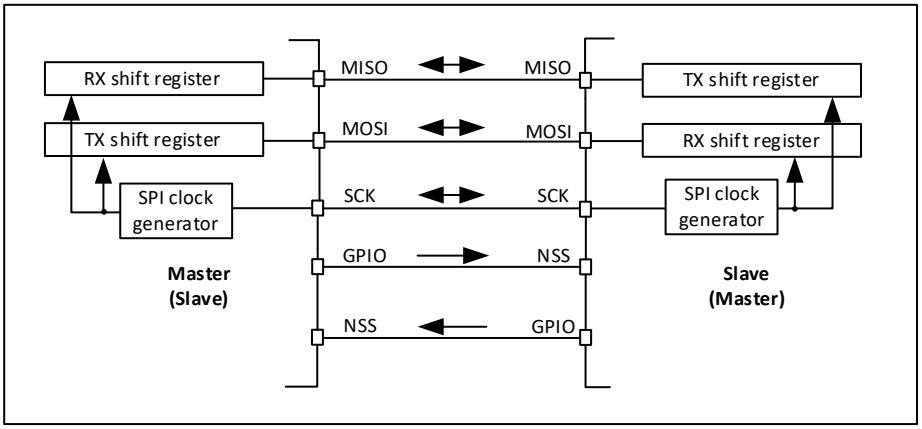


Figure 35-6 Multi-master application

35.2.4 Slave Select (NSS) Pin Management

In slave mode, the NSS works as a standard “chip select” input and lets the slave communicate with the master. In master mode, NSS can be used either as output or input. When used as input, it prevents bus collisions for multiple masters, and when used as output, it can drive the slave select pin of a single slave.

Throughthe SSM bitof theSPI_CR1register, you can choose hardware or software for slave management:

- Software NSS management (SSM = 1): in this configuration, slave select information is driven internally by the SSI bit value in register SPI_CR1. The external NSS pin is free for other application uses.
- Hardware NSS management (SSM = 0): in this case, there are two possible configurations.
  1) NSS output enabled (SSM = 0, SSOE = 1): This configuration is used only when the device operates in master mode. The NSS pin is managed by the hardware. The NSS signal is driven low as soon as the SPI is enabled in master mode (SPE=1), and is kept low until the SPI is disabled (SPE =0). The SPI cannot work in multimaster configuration with this NSS setting.
  2) NSS Output Disabled (SSM = 0, SSOE = 0): This configuration allows multi-host communication if the MCU is the host on the bus. If the NSS pin is pulled low at this time, the SPI enters the master mode default state and is automatically configured to slave mode. In slave mode, the NSS pin works as a standard “chip select” input and the slave is selected while NSS line is at low level.

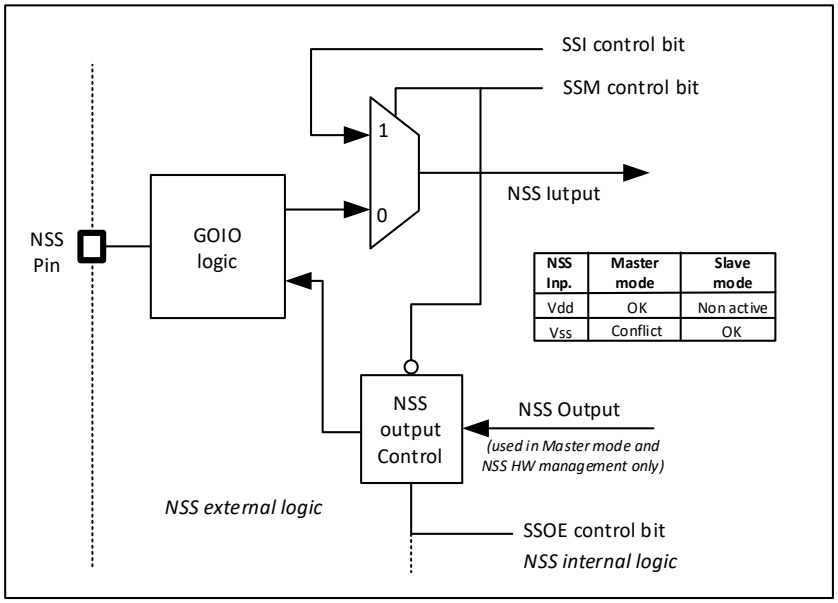


Figure 35-7 Hardware/software slave select management

35.2.5 Communication formats

During SPI communication, receive and transmit operations are performed simultaneously The serial clock (SCK) synchronizes the shifting and sampling of the information on the data lines. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slaves devices must follow the same communication format.

35.2.5.1 Clock phase and polarity controls

Four possible timing relationships can be selected by software through the CPOL and CPHA bits in the SPI_CR1 register.

The CPOL (clock polarity) bit controls the state of the clock level when no data is transmitted. This bit affects both master and slave modes. If CPOL is reset, the SCK pin is low in the idle state. If CPOL is set to 1, the SCK pin is high in the idle state.

If CPHA (clock phase) position 1, the second edge on the SCK pin (falling edge if CPOL bit is reset, rising edge if CPOL position 1) samples the MSBit. That is, the data is latched at the second clock edge.

If the CPHA bit is reset, the first edge on the SCK pin (falling edge if CPOL position 1, rising edge if CPOL bit is reset) samples the MSBit. That is, the data is latched at the first clock edge.

The combination of CPOL and CPHA selects the data sampling clock edge.

Prior to changing the CPOL/CPHA bits the SPI must be disabled by resetting the SPE bit.

The IDLE state ofthe SCKmust correspond tothe polarity selectedby the SPI_CR1register.

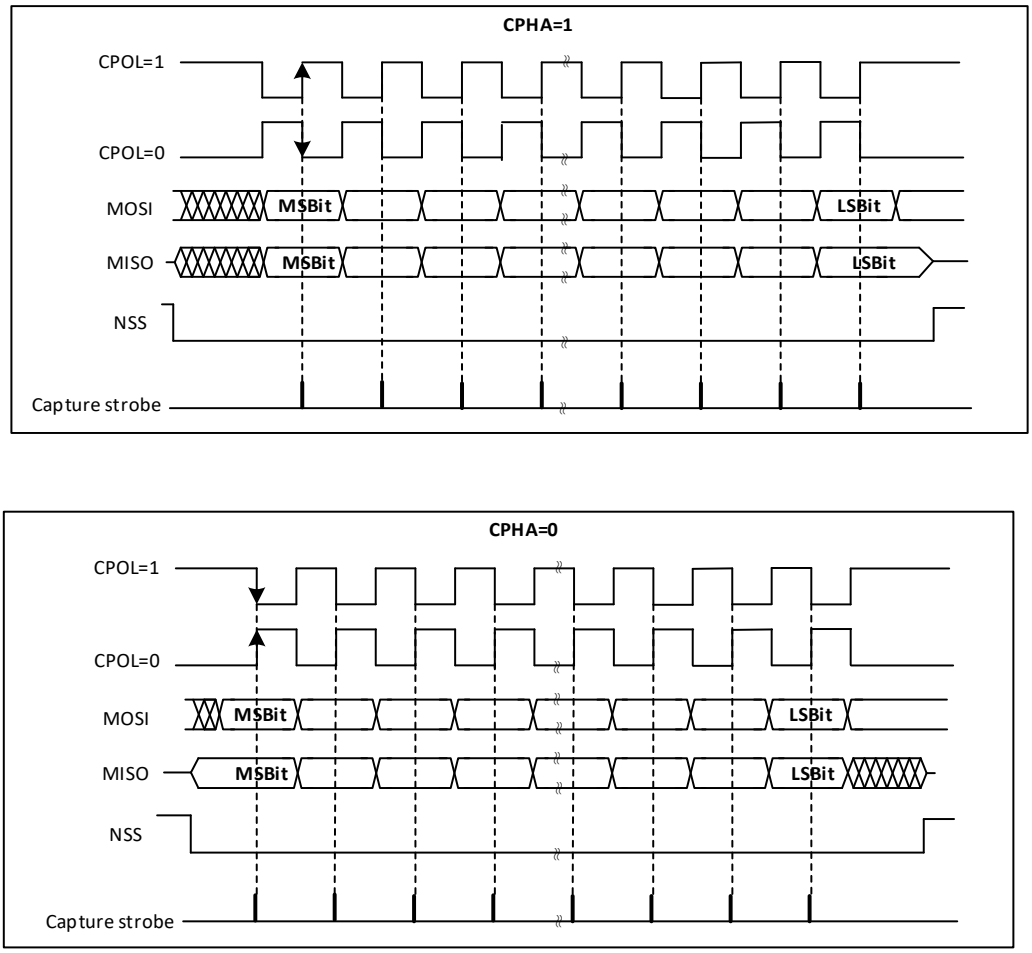


Figure 35-8 Data clock timing diagram

The order of the data bits depends onthe settingof LSBFIRST (SPI _CR1 register).

35.2.5.2 Data frame format

The SPI shift registermay be set to MSB-FIRST orLSB-FIRST by LSBFIRST(SPI _CR1 register). By using the DFF (SPI _CR1 register), the number of bits of the data frame is selected, which can be selected to be 8 bits or 16 bits in length. This setting is applicable to both transmission and reception.

35.2.6 Configuration of SPI

The configuration procedure is almost the same for master and slave. When a standard communication is to be initialized, perform these steps:

1. Write GPIO registers: Configure MOSI, MISO, and SCK pins
2. Write to the SPI_CR1 register:
    1) Configuring clock baud rate via BR [2: 0]
    2) Configure the CPOL and CPHA bits
    3) Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE
    4) Configure the LSBFIRST bit
    5) Configure SSM and SSI
    6) Configure MSTR (in multi-host NSS configuration, to avoid NSS conflicts, the host configures MSTR to prevent MODF errors)
3. Write to SPI_CR2 register:

1) Configure the DS bit to select the data length for the transfer
2) Configure SSOE (not required for slave mode)

35.2.7 Procedure for enabling SPI

The SPI slave is enabled before the master sends the clock. If not, undesired data transmission might occur. Before starting communication with the master, the data register of the slave must write the data to be sent. The SCK signal must be settled at an idle state level corresponding to the selected polarity before the SPI slave is enabled.

In full duplex mode (or send only), when SPI is enabled and TXFIFO is not empty, or the next write is made to TXFIFO, the host starts communication.

In any host receive-only mode (RXONLY = 1, or BIDIMODE = 1 and BIDIOE = 0), after SPI is enabled, the host starts communicating and immediately provides the clock.

35.2.8 Data transmission and reception procedures

35.2.8.1 RXFIFO and TXFIFO

SPI All data communication is through FIFO, with a depth of 4 and a width of 16 bits (when the data frame is set to 8 bits, the width is 8 bits). This enables the SPI to work in a continuous flow, and prevents overruns when the data frame size is short. Each direction has its own FIFO called TXFIFO and RXFIFO. These FIFOs are used in all SPI modes.

The handling of FIFOs depends on the data exchange mode (duplex, simplex) and data frame format.

A read access to the SPI_SR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPI_DR stores the written data in the TXFIFO at the end of a send queue. The read access must be always aligned with the RXFIFO threshold. The FTLVL and FRLVL bits show the current occupancy of the two FIFOs.

A read access to the SPI_DR register must be managed by the RXNE event. This event is triggered when data is stored in RXFIFO and the threshold is reached. When RXNE is cleared, RXFIFO is considered to be empty.

Write the data frame to be sent, managed by TXE events. This event is triggered when the TXFIFO level is less than or equal to half of its capacity. Otherwise, TXE is cleared and the TXFIFO is considered as full.

In this way, RXFIFO can store up to two data frames.

Both TXE and RXNE events can be polled or handled by interrupts.

If the next data is received when the RXFIFO is full, an overrun event occurs. An overrun event can be polled or handled by an interrupt.

The set BSY bit indicates that communication is in progress for the current data frame. When the clock signal runs continuously, the BSY flag stays set between data frames at master. However, BSY will keep a minimum of 1 SPI clock low level between each data frame transmission on the slave side.

35.2.8.2 Sequence handling

A few data frames can be passed at single sequence to complete a message. When transmission is enabled, a sequence begins and continues while any data is present in the TXFIFO of the master. The clock signal is continuously supplied by the host until TXFIFO becomes empty, and then stops waiting for additional data.

In receive-only modes, half-duplex (BIDIMODE=1, BIDIOE=0) or simplex (BIDIMODE=0,RXONLY=1) the master starts the sequence immediately when both SPI is enabled and receive-only mode is activated. The clock signal is provided by the master, and it does not stop until either SPI or receive-only mode is disabled by the master. The master receives data frames continuously up to this moment When the master communicates in continuous mode (the SCK signal is continuous), the master must consider the ability of the slave to process the data stream. When necessary, the host must slow down the communication speed and provide a slower clock, or send separate frames. It should be noted that there is no underflow error signal for the master or slave, and the data from the slave is processed by the master (even if the slave cannot send the correct data in time).

Each sequence must be managed by the NSS pin, and at the same time, one of the slaves to communicate in the multi-slave system is selected. In a single slave system, it is not necessary to use NSS to control the slave, but it is usually preferable to provide NSS to synchronize the beginning of each data sequence of communication. NSS can be managed by both software and hardware.

When the BSY bit is set it signifies an ongoing data frame transaction. When the dedicated frame transaction is finished, the RXNE flag is raised. The last bit is sampled and the entire data frame is stored in the RXFIFO.

35.2.8.3 Procedure for disabling the SPI

In full duplex or single transmission mode, the host can complete the interaction after providing the data to be sent. In this case, after the last data interaction, the clock is stopped. In these modes, the user must use the standard shutdown process before the SPI can shut down. SPI is prohibited when the host sends. If one frame interaction is in progress at this time, or the next data frame exists in TXFIFO, the function of SPI cannot be guaranteed.

When the host is in any receive-only mode, the only way to stop the continuous clock is to turn off SPI (SPE = 0).

Data received but not read remains stored in RXFIFO when the SPI is disabled, and must be processed the next time the SPI is enabled, before starting a new sequence. To prevent having unread data, ensure that RXFIFO is empty when disabling the SPI.

Standard disable procedure is based on pulling BSY status together with FTLVL to check if a transmission session is fully completed. The communication being interacted can also be authenticated by specific checks, such as:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave.
- When transactions' streams from FIFO are completed while the last data frame is still ongoing in the peripheral bus.

The correct disable procedure is (except when receive only mode is used):

1. Wait for FTLVL = 0 (no data to send)
2. Wait until BSY=0 (the last data frame is processed)
3. Disable the SPI (SPE=0).
4. Read data until FRLVL = 0 (read all the received data)

The correct disable procedure for certain receive only modes is:

1. Interrupt the receive flow by disabling SPI (SPE=0) in the specific time window while the last data frame is ongoing.
2. Wait until BSY=0 (the last data frame is processed).
3. Read data until FRLVL = 0 (read all the received data)

35.2.8.4 Data packing

When the data frame size fits into 8 bits, data packing is used automatically when any read or write 16-bit access is performed on the SPI_DR register. The double data frame pattern is handled in parallel in this case.

The figure below provides an example of data packing mode sequence handling. After a single 16-bit access by the sender, two data frames are sent. On the receiver side, if the RXFIFO width is 16 bits, an RXNE event is immediately generated. The receiver then has to access both data frames by a single 16-bit read of SPI_DR as a response to this single RXNE event. The RxFIFO threshold setting and the following read access must be always kept aligned at the receiver side, as data can be lost if it is not in line.

At the sending end, an odd sequence can be written using 8-bit access mode to complete communication; At the receiving end,in order to generate an RXNE event, the receiver must changethe Rx_FIFO thresholdwhen the last data frame is received.

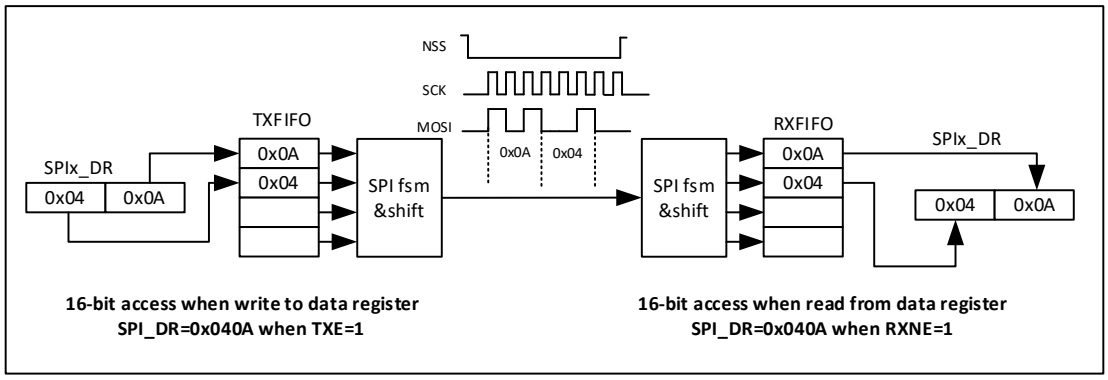


Figure 35-9 Packing data in FIFO for transmission and reception

35.2.8.5 Communication timing

Some typical timing schemes are explained in this section. These schemes are valid no matter if the SPI events are handled by polling, interrupts. For simplicity, the LSBFIRST=0, CPOL=0 and CPHA=1 setting is used as a common assumption here.

1. When NSS is valid, SPI is enabled, and the slave starts controlling MISO. When NSS is released or SPI is turned off; The slave loses control of the MISO. Sufficient time must be provided for the slave to prepare data dedicated to the master in advance before its transaction starts. At the master, the SPI peripheral takes control at MOSI and SCK signals (occasionally at NSS signal as well) only if SPI is enabled. If the SPI is turned off, the SPI peripherals are disconnected from the GPIO, and the level values on these lines depend on the GPIO setting.
2. At the master, BSY stays active between frames if the communication is continuous. At the slave, BSY signal always goes down for at least one clock cycle between data frames.
3. The TXE signal is cleared only if TXFIFO is full.

4. The TXE interrupt is generated just after the TXEIE is set. When the TXE signal is active, start writing data to the TXFIFO until the TXFIFO becomes full.

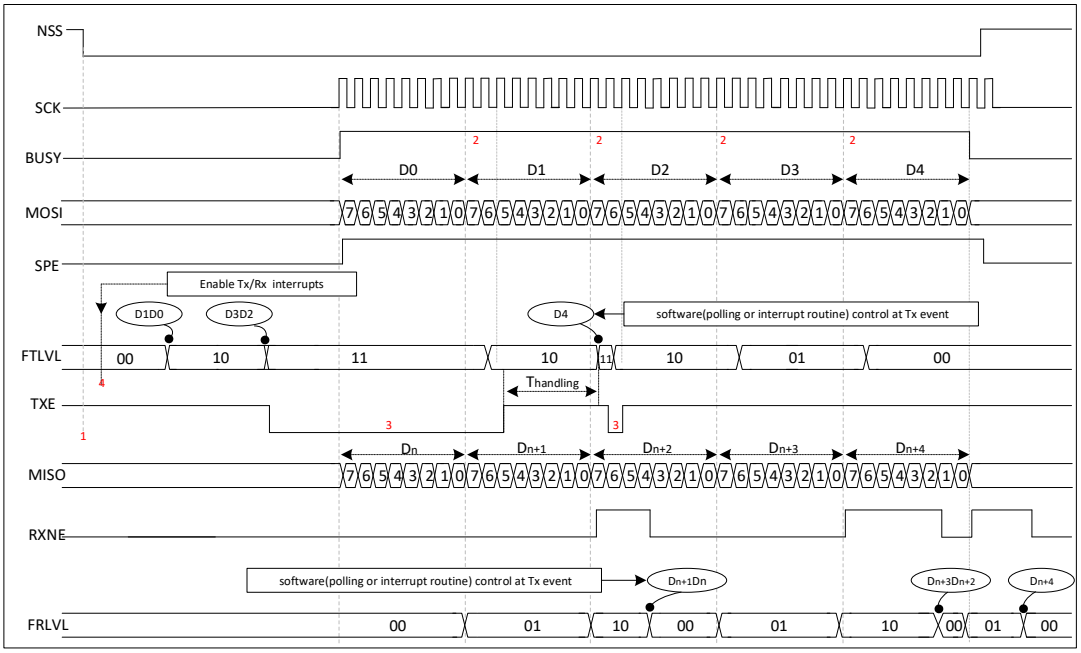


Figure 35-10 hostfull-duplex communication diagram (bit frame = 8)

35.2.9 SPI status flags

Three status flags are provided for the application to completely monitor the state of the SPIbus.

35.2.9.1 Tx FIFO empty flag (TXE)

The TXE flag is set when transmission TXFIFO has enough space to store data to send. TXE flag is linked to the TXFIFO level. The flag goes high and stays high until the TXFIFO level is lower or equal to 1/2 of the FIFO depth. An interrupt can be generated if the TXEIE bit in the SPI_CR2 register is set.

35.2.9.2 Rx buffer not empty (RXNE)

The RXNE flag is set depending on the FRXTH bit value in the SPI_CR2 register.

35.2.9.3 Busy flag (BSY)

The BSY flag is set and cleared by the hardware (writing this bit has no effect). This flag indicates the SPI communication status.

When it is set to '1' by hardware, it indicates that the SPI is in communication.

Before the software wants to shut down the SPI module, the BSY flag can be used to detect whether the transmission is finished, which can avoid corrupting the last transmission.

The BSY flag is cleared under any one of the following conditions:

- When the SPI is correctly disabled
- Host mode, when the MODF flag is generated
- Host mode, when the transmission is complete, there is no more data to send
- In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: Do not use the BSY flag for each data transmission and reception. It is better to use theTXE and RXNE flags instead.

35.2.10 Error flags

35.2.10.1 Mode fault (MODF)

Master mode failure (MODF) only occurs when: when NSS is used as the input signal (SSOE = 0), the NSS pin of the master device is pulled low under hardware mode management; Or when the SSI bit is set to '0' under software mode management of the NSS pin. This automatically sets the MODF bit. Master mode fault affects the SPI interface in the following ways:

- The MODF bit is set and an SPI interrupt is generated if the ERRIE bit is set.
- The SPE bit is cleared. This blocks all output from the device and disables the SPI interface.
- The MSTR bit is cleared, thus forcing the device into slave mode.

Use the following software sequence to clear the MODF bit:

1. Make a read or write access to the SPI_SR register while the MODF bit is set.
2. Then write to the SPI_CR1 register.

To avoid any multiple slave conflicts in a system comprising several MCUs, the NSS pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits can be restored to their original state after this clearing sequence.

As a security, hardware does not allow the SPE and MSTR bits to be set while the MODF bit is set. In the slave device, the MODF position 1 cannot be placed. However, in a multi-master mode configuration, the device may be in a slave mode when at MODF position 1. In this case, the MODF bit indicates that the system control may have a multi-master mode conflict. You can use the interrupt procedure to fully recover from this state by performing a reset or returning to the default state.

35.2.10.2 Overflow Error (OVR)

An overflow error occurs when data is received by a master or a slave, and the RXFIFO does not have enough space to store the received data. This error occurs if the software does not have enough time to read away the previously received data (stored in the RXFIFO).

When an overflow error occurs, the newly received data will not overwrite the data previously stored in RXFIFO. The newly received value is discarded, and all data transmitted subsequently is lost.

Clearing the OVR bit is done by a read access to the SPI_DR register followed by a read access to the SPI_SR register.

35.2.11 SPI interrupts

Table 35-1 SPI interrupt requests

Interrupt event	Event flag	Enable control bit
TXFIFO empty	TXE	TXEIE
RXFIFO non-empty	RXNE	RXNEIE
Master Mode fault event	MODF	ERRIE
Overrun error	OVR	ERRIE

35.2.12 CRC Calculation

Two separate CRC calculators are implemented in order to check the reliability of transmitted and received data. SPI provides CRC8 or CRC16 calculations independent of frame data length and can be fixed to 8 or 16 bits. For all other data frame lengths, no CRC is available.

35.2.12.1 CRC standard

Before SPI is enabled (SPE = 1), CRC calculation is enabled by setting the CRCEN bit in the SPI_CR1 register. The calculation is processed at the sample clock edge defined by the CPHA and CPOL bits in the SPI_CR1 register. The calculated CRC value is automatically checked at the end of the data block, and the transmission is managed by the CPU or DMA. When the CRC calculated internally of the received data does not match the received CRC, the CRCERR flag is set to indicate a data error. The correct procedure to handle the CRC calculation depends on the SPI configuration and the transfer management mode selected.

35.2.12.2 CRC transfer managed by CPU

The CRCNEXT bit must be set before the end of transmission of the last data frame is received, and is used to indicate that the transmission of the CRC frame begins after the currently processed data frame. CRC calculation is frozen during CRC transaction.

The received CRC is stored in the RXFIFO like a data byte or word. In CRC mode, the receive buffer must be treated as a single 16-bit buffer for receiving only one data frame at a time. The CRC format transmission typically requires an extra data frame at the end of the data transmission. When setting up an 8-bit data frame that passes a 16-bit CRC check, two more frames are needed to send the full CRC value.

When the last CRC data is received, an automatic check is performed, comparing the received value with the SPI_RXCRC register. The software must check the CRCERR flag in the SPI_SR register to determine the data transfer status. The software clears the CRCERR flag by writing a "0" to it. After CRC is received, the CRC value is stored in the RXFIFO and the SPI_DR register must be read to clear the RXNE flag

35.2.12.3 CRC transmission under DMA

When SPI communication is enabled using DMA mode with CRC, the transmission and reception of CRC at the end of communication is done automatically (except for reading CRC data in receive-only mode) and the CRCNEXT bit does not have to be processed by software. The counter of the SPI transmission DMA channel must be set to the number of data frames to be transmitted, which does not include CRC frames. At the receiving end, the received CRC value is automatically processed by the DMA at the end of the transmission, but the user must take care to clear the received CRC information from the RXFIFO, as it is always deposited in the RXFIFO. In the full-duplex mode, the counter of the received DMA channel may be set to the number of data frames including CRC to be received.

In receive only mode, the DMA receive channel counter should contain only the amount of data transmitted, not the CRC frame. At the end of the data and CRC transfer, if an error occurs during the transfer, the CRCERR flag in the SPI_SR register is set.

35.2.13 TI mode

The SPI interface is compatible with TI protocol. The SPI can be configured using the FRF bit of the SPI_CR2 register to be compatible with this protocol.

Both the clock polarity and phase are forced to follow the TI protocol regardless of the settings in SPI_CR1. NSS management is also specific to the TI protocol, in which case NSS management cannot be configured through the SPI _CR1 and SPI_CR2 registers (SSM, SSI, and SSOE).

In slave mode, the SPI baud rate prescaler is used to control the moment when the MISO pin is switched to the high impedance state when the current transmission is complete. Any baud rate can be used, making it possible to determine this moment with optimal flexibility. The baud rate is usually set to the external master clock baud rate. The release for the MISO signal to become a high impedance state depends on the internal resynchronization and the baud rate value set by the BR [2: 0] bit of the SPI_CR1 register. It is given by the formula:

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

If the slave device detects a misaligned NSS pulse during data frame transmission, the TIFRE flag will be set to 1.

35.3 I2S functional description

35.3.1 Introduction

The I2S interface uses roughly the same pins, flags, and interrupts as the SPI interface.

The I2S shares three common pins with the SPI:

SD: serial data (mapped to MOSI pin), used to send and receive data of two time division multiplexed channels;

WS: word selection (mapped to NSS pin), output as data control signal in master mode and input in slave mode;

CK: Serial clock (mapped to SCK pin), output as clock signal in master mode and input in slave mode.

An additional pin can be used when a master clock output is needed for some external audio devices:

MCK: Master clock (independent mapping), used as an extra clock signal pin for outputting when I2Sis configured in master mode and the MCKOE bit of register SPI_I2SPR is' 1 '. The frequency of the output clock signal is preset to 256 × Fs, where Fs is the sampling frequency of the audio signal.

The I2S uses its own clock generator to produce the communication clock when it is set in master mode. This clock generator is also the source of the master clock output. There are two additional registers in I2S mode, one is the register SPI_I2SPR related to the clock generator configuration, and the other is the I2S general configuration register SPI_I2SCFGR (which can set parameters such as audio standard, slave/master mode, data format, packet frame, clock polarity, etc.).

The I2S function can be enabled by setting the I2SMOD bit of the register SPI_I2SCFGR to '1'. At this time, the SPI module may be used as an I2S audio interface.

The SPI_CR1 register and all CRC registers are not used in the I2S mode. Likewise, the SSOE bit in the SPI_CR2 register and the MODF and CRCERR bits in the SPI_SR are not used.

I2S uses the same register SPI_DR as SPI for 16-bit mode data transfer.

35.3.2 Supported audio protocols

The three-wire bus supports time division multiplexing of audio data on two channels: the left channel and the right channel, but only one 16-bit register is used for transmission or reception. Therefore, when writing data to the data register, the software must write the corresponding data according to the channel currently in transmission; Similarly, when the register data is read, it is determined to which channel the

received data belongs by checking the CHSIDE bit of the register SPI_SR. The left channel always sends data before the right channel (the CHSIDE bit is meaningless under the PCM protocol).

There are four combinations of data and frame formats, and data can be sent in the following formats:

- 16-bit data packed in a 16-bit frame
- 16-bit data packed in a 32-bit frame
- 24-bit data packed in a 32-bit frame
- 32-bit data packed in a 32-bit frame

When extending to a 32-bit frame using 16-bit data, the first 16 bits (MSB) are meaningful data and the last 16 bits (LSB) are forced to 0, which requires no software intervention and no DMA request (only one read/write operation is required). The 24-bit and 32-bit data frames need two CPU read or write operations to/from the SPI_DR register or two DMA operations if the DMA is preferred for the application. For 24-bit data frame specifically, the 8 non-significant bits are extended to 32 bits with 0-bits (by hardware). For all data formats and communication standards, the highest bit (MSB) is always sent first.

The I2S interface supports four audio standards, configurable using the I2SSTD [1:0] and PCMSYNC bits in the SPI_I2SCFGR register.

35.3.2.1 I2S Phillps standard

For this standard, the WS signal is used to indicate which channel is being transmitted. This pin is active 1 clock cycle before the first bit of data (MSB) is sent.

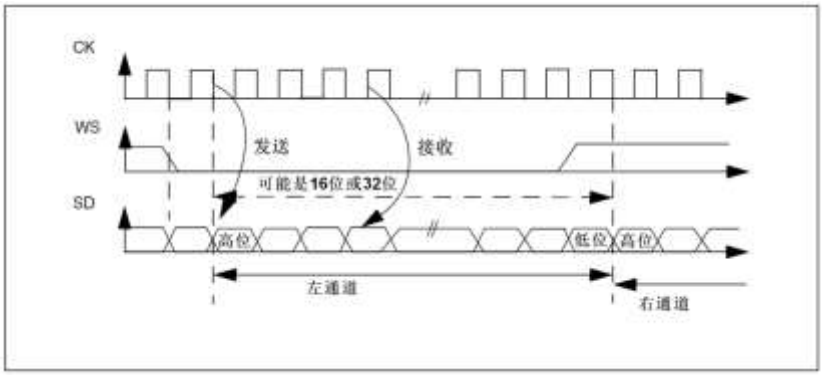


Figure 35-11 I2S Philips protocol waveform (16/32 bit full accuracy, CKPOL = 0)

When CKPOL = 0, the transmitter changes data on the falling edge of the clock signal (CK) and the receiver reads data on the rising edge. The WS signal is also latched on the falling edge of clock signal. When CKPOL = 1, the transmitter also changes data at the falling edge of the clock signal (CK).

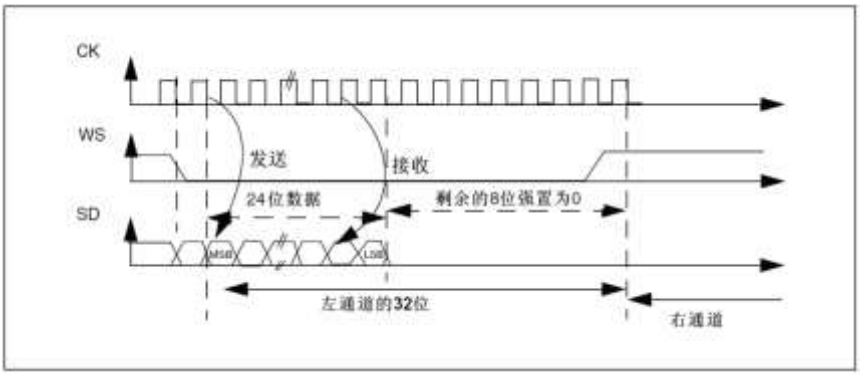


Figure 35-12 I2S Philips protocol standard waveform (24-bit frame, CKPOL = 0)

This mode needs two write or read operations to/from the SPI_DR register.

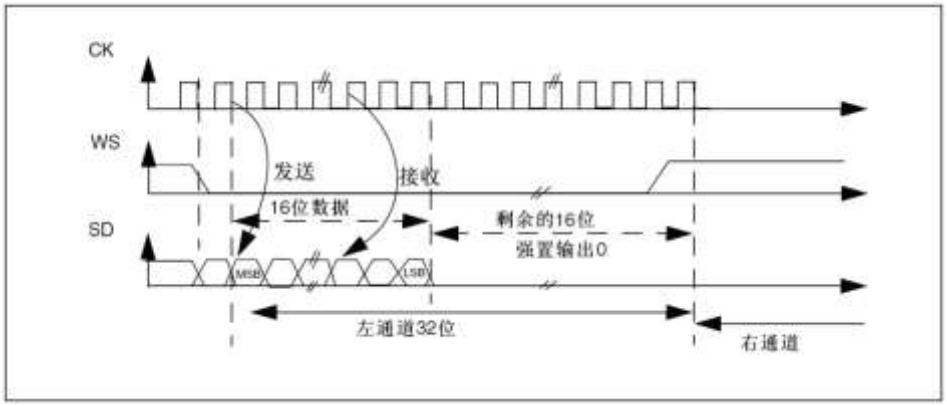


Figure 35-13 I2S Philips protocol standard waveform (16-bit extension to 32-bit packet frame, CKPOL = 0)

When CKPOL = 1, the transmitter also changes data at the falling edge of the clock signal (CK)

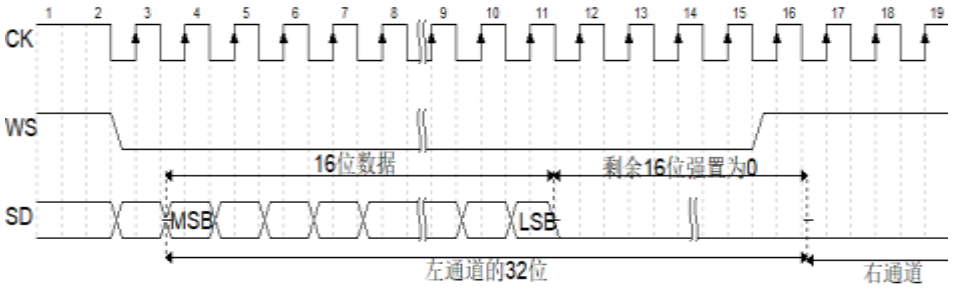


Figure 35-14 I2S Philips protocol standard waveform (16-bit extension to 32-bit packet frame, CKPOL = 1)

When 16-bit data frame extended to 32-bit channel frame is selected during the I2S configuration phase, only one access to the SPI_DR register is required. The 16 remaining bits are forced by hardware to 0x0000 to extend the data to 32-bit format.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x76A30000), the required operation is shown in the figure below.

Figure 35-15 I2S Data Reception

The MSB needs to be written to the register SPI_DR when transmitting; A flag bit TXE of '1' indicates that new data can be written, and an interrupt can be generated if a corresponding interrupt is allowed. This takes place even if 0x0000 have not yet been sent because it is done by hardware. On receipt, the flag bit RXNE is set to '1' after receiving a 16-bit half-word higher (MSB), and if the corresponding interrupt is allowed, an interrupt can be generated.

In this way, more time is provided between two write or read operations to prevent underrun or overrun conditions.

35.3.2.2 MSB justified standard

Under this standard, the WS signal and the first data bit, the highest bit (MSB), are generated simultaneously.

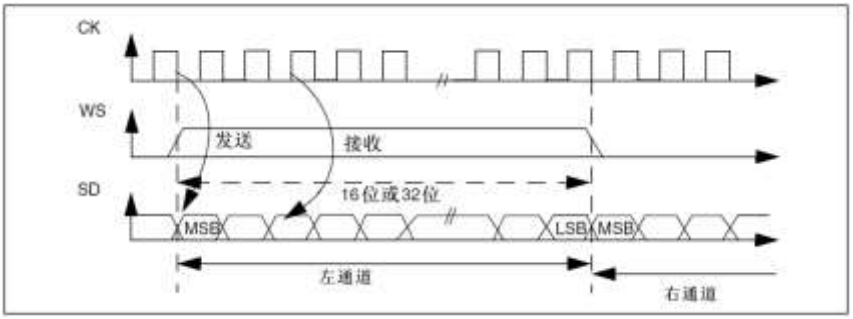


Figure 35-16 MSB Alignment 16-bit or 32-bit Full Precision, CPKOL = 0

When CKPOL = 1, the transmitter also changes data at the falling edge of the clock signal (CK)

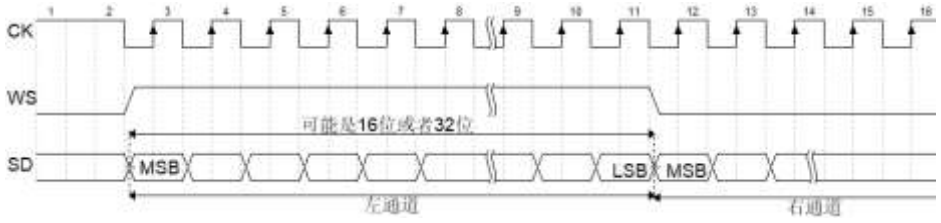


Figure 35-17 MSB Alignment 16-bit or 32-bit Full Precision, CPKOL = 1

Data are latched on the falling edge of CK (for transmitter) and are read on the rising edge (for the receiver).

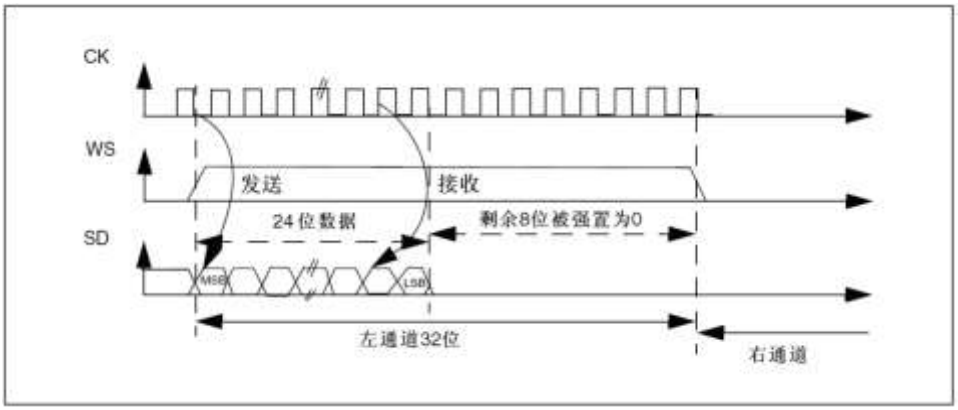


Figure 35-18 MSB aligned 24-bit data, CKPOL = 0

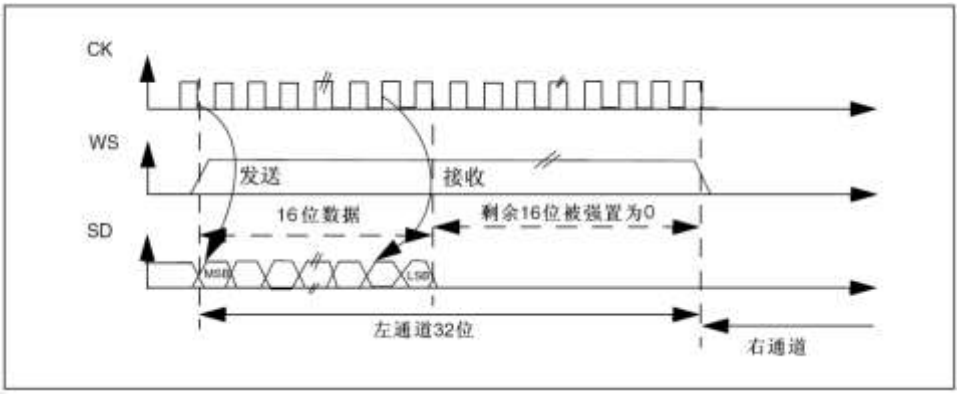


Figure 35-19 MSB aligned 16-bit data extension to 32-bit packet frame, CPKOL = 0

When CKPOL = 1, the transmitter also changes data at the falling edge of the clock signal (CK). The next TXE event occurs as soon as valid data begins to be sent from the SD pin. Upon reception, an RXNE event occurs once valid data is received (instead of the 0x0000 portion).

35.3.2.3 LSB justified standard

This standard is similar to the MSB alignment standard (no difference in 16-bit or 32-bit full-precision frame formats). When CKPOL = 1, the transmitter also changes data at the falling edge of the clock signal (CK)

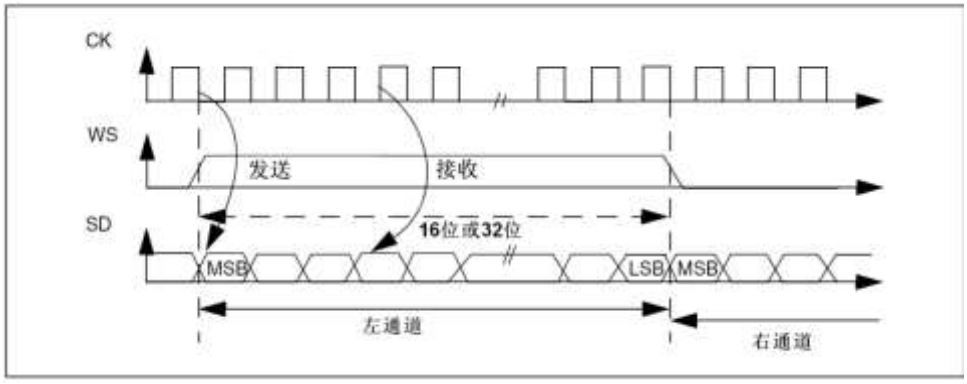


Figure 35-20 LSB alignment 16-bit or 32-bit full accuracy, CKPOL = 0

● In send mode

If data 0x3478AE have to be transmitted, two write operations to the SPI_DR register are required by software or by DMA. The operation flow is shown below.

Figure 35-21 Requires Operation to Read 0x3478AE

● In receive mode

If you want to receive data 0x3478AE, you need to read the register SPI_DR once when two consecutive RXNE events occur.

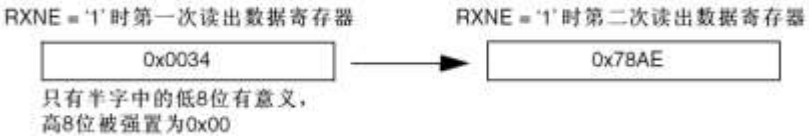


Figure 35-22 requires operation to receive 0x3478AE

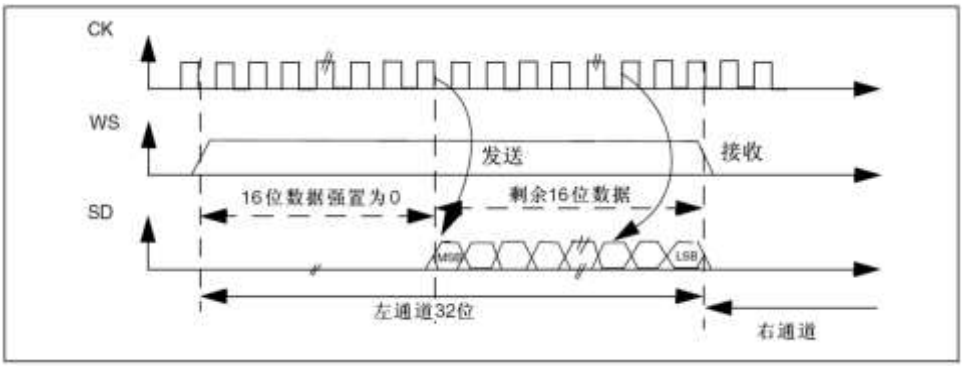


Figure 35-23 LSB aligned 16-bit data extension to 32-bit packet frame, CPOL = 0

When CKPOL = 1, the sender also changes data at the falling edge of the clock signal (CK). When 16-bit data frame extended to 32-bit channel frame is selected during the I2S configuration phase, only one access to the SPI_DR register is required. At this time, the upper half word (16-bit MSB) extended to 32 bits is set to 0x0000 by hardware.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x0000 76A3), the required operation: SPI_DR only needs to be operated once.

At the time of transmission, if the TXE is' 1 ', the user needs to write the data to be transmitted (i.e., 0x76A3). The 0x0000 used to expand to 32 bits is partially sent out by the hardware first. Once valid data starts to be sent out from the SD pin, the next TXE event occurs. Upon reception, an RXNE event occurs once valid data is received (instead of the 0x0000 portion). In this way, more time is provided between two write or read operations to prevent underrun or overrun conditions.

35.3.2.4 PCM standard

For the PCM standard, there is no need to use channel-side information. The two PCM modes (short and long frame) are available and configurable using the PCMSYNC bit in SPI_I2SCFGR register.

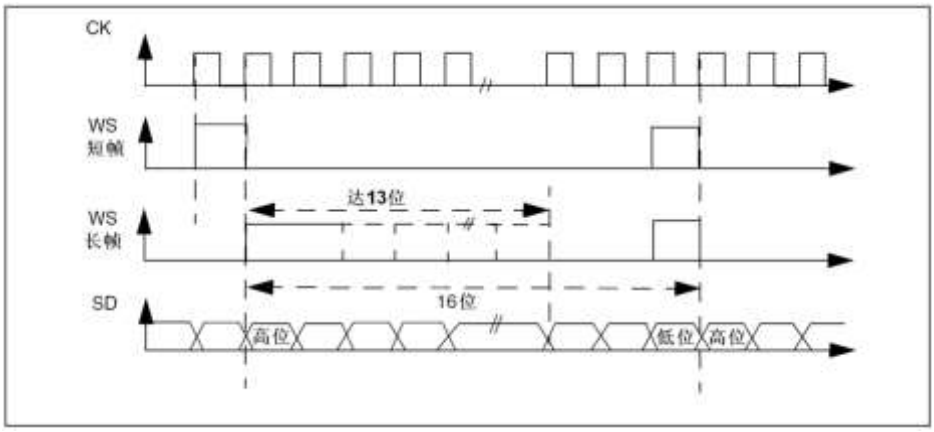


Figure 35-24 PCM standard waveform (16 bits)

When CKPOL = 1, the transmitter also changes data at the rising edge of the clock signal (CK). For long frames, in main mode, the valid time of the WS signal used for synchronization is fixed to 13 bits. For short frames, the length of the WS signal used for synchronization is only 1 bit.

When CKPOL = 1, the transmitter also changes data at the rising edge of the clock signal (CK)

Regardless of which mode (master or slave), which synchronization mode (short frame or long frame), the time difference between two consecutive frames of data and between two synchronization signals

(even in the slave mode) needs to be determined by setting the DATLEN bit and the CHLEN bit of the SPI_I 2 SCFGR register.

35.3.3 Clocks

The I2S bit rate determines the data flow on the I2S data line and the I2S clock signal frequency. I.e

I2S bit rate = number of bits per channel × number of channels × audio sampling frequency

For a 16-bit audio, left and right channel, the I2S bit rate is calculated as follows:

I2S bitrate = 16 × 2 × fs;

If the packet length is 32 bits, there is:

I2S bitrate = 32 × 2 × fs

Figure 35-25 Audio Sampling Definition

When the master mode is configured, a specific action needs to be taken to properly program the linear divider in order to communicate with the desired audio frequency. The sampling frequency of the audio may be 96 kHz, 48 kHz, 44.1 kHz, 32 kHz, 22.05 kHz, 16 kHz, 11.025 kHz, or 8 kHz (or any value within this range). In order to obtain the required frequency, it is obtained by setting the linear frequency divider according to the following formula, when the master clock needs to be generated (the MCKOE bit of the register SPI_I2SPR is' 1 '):

When the frame length of the channel is 16 bits, Fs = I2SxCLK/[(16 * 2) * ((2 * I2SDIV) + ODD) * 8]

When the frame length of the channel is 32 bits, Fs = I2SxCLK/[(32 * 2) * ((2 * I2SDIV) + ODD) * 4]

When the master clock is turned off (MCKOE bit is' 0 '):

When the frame length of the channel is 16 bits, Fs = I2SxCLK/[(16 * 2) * ((2 * I2SDIV) + ODD)]

When the frame length of the channel is 32 bits, Fs = I2SxCLK/[(32 * 2) * ((2 * I2SDIV) + ODD)]

Use the standard 8MHz HSE clock to get the accurate audio frequency, see EXCEL table.

35.3.4 Sending and receiving

35.3.4.1 Master mode

The I2S can be configured in master mode. This means that the serial clock is generated on the CK pin as well as the Word Select signal WS. Outputting or not outputting the master clock (MCK) can be selected by setting the MCKOE bit of the register SPI_I2SPR.

1) Sending process

When one and a half word (16 bits) of data is written to the transmission buffer, the transmission flow begins.

Lets assume the first data written into the Tx buffer corresponds to the left channel data. When data are transferred from the Tx buffer to the shift register, TXE is set and datacorresponding to the right channel have to be written into the Tx buffer. The CHSIDE flagindicates which channel is to be transmitted. It

has a meaning when the TXE flag is set because the CHSIDE flag is updated when TXE goes high. A full frame has to be considered as a left channel data transmission followed by a right channel data transmission. It is not possible to have a partial frame where only the left channel is sent.

The data half-word is parallel loaded into the 16-bit shift register during the first bit transmission, and then shifted out, serially, to the MOSI/SD pin, MSB first. The TXE flag is set after each transfer from the Tx buffer to the shift register and an interrupt is generated if the TXEIE bit in the SPI_CR2 register is set.

The operation of writing data depends on the chosen I2S standard, as detailed in Section 5.2. To ensure a continuous audio data transmission, it is mandatory to write the SPI_DR register with the next data to transmit before the end of the current transmission. To switch off the I2S, by clearing I2SE, it is mandatory to wait for TXE = 1 and BSY = 0.

2) Receiving Process

Whatever the data or channel length, the audio data are received by 16-bit packets. This means that each time the Rx buffer is full, the RXNE flag is set and an interrupt is generated if the RXNEIE bit is set in SPI_CR2 register. Depending on the data and channel length configuration, the audio value received for a right or left channel may result from one or two receptions into the Rx buffer. Clearing the RXNE bit is performed by reading the SPIx_DR register. CHSIDE is updated after each reception. It is sensitive to the WS signal generated by the I2S cell. The operation of reading the data depends on the selected I2S standard.

If data are received while the previously received data have not been read yet, an overrun is generated and the OVR flag is set. If the ERRIE bit is set in the SPI_CR2 register, an interrupt is generated to indicate the error.

To switch off the I2S, specific actions are required to ensure that the I2S completes the transfer cycle properly without initiating a new data transfer.

Note: The BSY flag is kept low during transfers.

35.3.4.2 Slave mode

For the slave configuration, the I2S can be configured in transmission or reception mode. The operating mode is following mainly the same rules as described for the I2S master configuration. In slave mode, there is no clock to be generated by the I2S interface. The clock and WS signals are input from the external master connected to the I2S interface. There is then no need, for the user, to configure the clock.

1) Sending process

The transmission sequence begins when the external master device sends the clock and when the NSS_WS signal requests the transfer of data The slave has to be enabled before the external master starts the communication. The I2S data register has to be loaded before the master initiates the communication.

For the I2S, MSB justified and LSB justified modes, the first data item to be written into the data register corresponds to the data for the left channel. When the communication starts, the data are transferred from the Tx buffer to the shift register. The TXE flag is then set in order to request the right channel data to be written into the I2S data register.

The CHSIDE flagindicates which channel is to be transmitted. Compared to the master transmission mode, in slave mode, CHSIDE is sensitive to the WS signal coming from the external master. This means that the slave needs to be ready to transmit the first data before the clock is generated by the master. WS assertion corresponds to left channel transmitted first.

Note: The I2SE has to be written at least two PCLK cycles before the first clock of the master comes on the CK line.

The data half-word is parallel-loaded into the 16-bit shift register (from the internal bus) during the first bit transmission, and then shifted out serially to the MOSI/SD pin MSB first. The TXE flag is set after each transfer from the Tx buffer to the shift register and an interruptis generated if the TXEIE bit in the SPI_CR2 register is set. Note that the TXE flag should be checked to be at 1 before attempting to write the Tx buffer.

To ensure a continuous audio data transmission, it is mandatory to write the SPI_DR register with the next data to transmit before the end of the current transmission. An underrun flag is set and an interrupt may be generated if the data are not written into the SPI_DR register before the first clock edge of the next data communication. This indicates to the software that the transferred data are wrong. If the ERRIE bit is set into the SPI_CR2 register, an interrupt is generated when the UDR flag in the SPI_SR register goes high. In this case, it is mandatory to switch off the I2S and to restart a data transfer starting from the left channel.

To switch off the I2S, by clearing the I2SE bit, it is mandatory to wait for TXE = 1 and BSY = 0.

2) Receiving Process

Whatever the data length or the channel length, the audio data are received by 16-bit packets. This means that each time the RX buffer is full, the RXNE flag in the SPI_SR register is set and an interrupt is generated if the RXNEIE bit is set in the SPI_CR2 register. Depending on the data length and channel length configuration, the audio value received for a right or left channel may result from one or two receptions into the RX buffer. The CHSIDE is updated each time data is received (to be read out from the SPI_DR), which corresponds to the WS signal generated by the I2Sunit. Clearing the RXNE bit is performed by reading the SPI_DR register.

When the previously received data is not read out and new data is received, overflow is generated, and the flag bit OVR is set to'1'; If the ERRIE bit of the register SPI_CR2 is' 1 ', an interrupt is generated indicating that an error has occurred.

To turn off the I2S function, the I2SE bit needs to be cleared '0' when the last RXNE = 1 is received.

Note: The external master I2S device needs to have the ability to send/receive 16-bit or 32-bit packets over the audio channel.

35.3.5 Flag bit

35.3.5.1 Status flag

Three status flags are provided for the application to fully monitor the state of the I2S bus.

1) Busy flag bit (BSY)

The BSY flag is set and cleared by the hardware (writing this bit has no effect), and this flag bit indicates the status of the I2S communication layer. When this bit is' 1 ', it indicates that I2S communication is in progress, with one exception: in primary reception mode (I2SCFG = 11), the BSY flag is always low

during reception. The BSY flag can be used in certain modes to detect the end of a transfer so that the software can disable the SPI which does not provide a clock for the peripheral. This avoids corrupting the last transfer. The BSY flag is set when a transfer starts, except when the I2S is in master receiver mode.

The BSY flag is cleared:

● When the transmission ends (except for the main transmit mode, where communication is continuous);

● When the I2S module is turned off.

When communication is continuous:

● In the main transmit mode, the BSY flag is always high throughout the transmission;

● In slave mode, the BSY flag goes low for 1 I2S clock cycle between each data item transmission.

2) Send buffer null flag bit (TXE)

When set, this flag indicates that the Tx buffer is empty and the next data to be transmitted can then be loaded into it. The TXE flag is reset when the Tx buffer already contains data to be transmitted. When I2S is turned off (I2SE bit is' 0 '), the flag bit is also' 0 '.

3) Receive buffer non-empty flag bit (RXNE)

When set, this flag indicates that there are valid received data in the RX Buffer. It is reset when SPI_DR register is read.

4) Channel Flag (CHSIDE)

In transmission mode, this flag is refreshed when TXE goes high. It indicates the channel side to which the data to transfer on SD has to belong. In case of an underrun error event in slave transmission mode, this flag is not reliable and I2S needs to be switched off and switched on before resuming the communication. In reception mode, this flag is refreshed when data are received into SPI_DR. It indicates from which channel side data have been received. If an error occurs (e.g., overflow OVR), this flag bit is meaningless and I2S needs to be turned off and turned on again (and the configuration of I2S should be modified if necessary).

This flag has no meaning in the PCM standard (for both Short and Long frame modes).

When the OVR or UDR flag in the SPI_SR is set and the ERRIE bit in SPI_CR2 is also set, an interrupt is generated. This interrupt can be cleared by reading the SPI_SR status register.

35.3.5.2 Error flags

There are two error flags for the I2S cell.

1) Underflow flag bit (UDR)

In slave transmission mode, this flag is set when the first clock for data transmission appears while the software has not yet loaded any value into SPI_DR. It is available when the I2SMOD bit in the SPI_I2SCFGR register is set. An interrupt may be generated if the ERRIE bit in the SPI_CR2 register is set. The UDR bit is cleared by a read operation on the SPI_SR register.

2) Overflow flag bit (OVR)

This flag is set when data are received and the previous data have not yet been read from the SPI_DR register. As a result, the incoming data are lost. An interrupt may be generated if the ERRIE bit is set in the SPI_CR2 register. At this time, the contents of the received cache are not refreshed to new data sent from the transmitting device. A read operation to register SPI_DR returns the last correctly received

data. All other 16-bit data sent by the sending device after the overflow occurs will be lost. The flag bit is cleared by first reading register SPI_SR and then reading register SPI_DR.

35.3.6 DMA functionality

Except for CRC function, same as SPI. Because there is no data transfer protection function in I2S mode.

35.3.7 Interrupts and events

Table 35-2 I2S interrupt requests

Interrupt event	Event flag	Enable control bit
Transmit buffer empty flag	TXE	TXEIE
Receive buffer not empty flag	RXNE	RXNEIE
Underrun error	OVR	ERRIE
Overrun error	UDR	

35.4 TIMx registers

SPI1 register base address: 0x4001 3000

SPI2 register base address: 0x4000 3800

SPI3 register base address: 0x4000 3C00

35.4.1 SPI_ CR1Register

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR [2: 0]			MSTR	CPOL	CPHA
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: Select "two- wire bidirectional" mode; 1: Select "single-wire bidirectional" mode. Note: It is not used in I2S mode.
14	BIDIOE	RW	0	Together with the BIDIMODE bit, the Output enable in bidirectional mode determines the output direction of the data in "single wire bidirectional" mode 0: Output disabled (receive only mode); 1: Output enable (send only mode). In master mode, the MOSI pin is used while the MISO pin is used in slave mode. Note: It is not used in I2S mode.
13	CRCEN	RW	0	Hardware CRC calculation enable 0: Disable CRC calculation; 1: Start CRC calculation. Note: This bit can only be written when SPI is disabled (SPE = 0), otherwise an error will occur. This bit can only be used in full duplex mode. Note: It is not used in I2S mode.
12	CRCNEXT	RW	0	Next transmit CRC (Transmit CRC next) 0: The value to be transmitted next comes from the transmit buffer. 1: Next transmit value is from Tx CRC register. Note: This bit has to be written as soon as the last data is written in the SPI_DR register. Note: It is not used in I2S mode.
11	DFF	RW	0	Data frame format 0: 8-bit data frame format is selected for transmission/reception 1: 16-bit data frame format is selected for transmission/reception Note: This bit should be written only when SPI is disabled (SPE = '0') for correct operation. Note: It is not used in I2S mode.
10	RXONLY	RW	0	The Receive only bit, together with the BIDIMODE bit, determines the direction of transmission in the "two-wire bidirectional" mode. In the configuration of multiple slave

				devices, this bit is set to 1 on the unaccessed slave device, so that only the accessed slave device has an output, thus not causing data collision on the data line. 0: full duplex (transmission and reception); 1: Output is disabled (reception only mode). Note: It is not used in I2S mode.
9	SSM	RW	0	Software slave management When SSM is set, the level on the NSS pin is determined by the value of the SSI bit. 0: Prohibit software from device management; 1: Enable software slave device management. Note: It is not used in I2S mode.
8	SSI	RW	0	Internal slave select This bit is only meaningful if the SSM bit is' 1 '. It determines the level on the NSS, and I/O operation on the NSS pin is invalid. Note: It is not used in I2S mode.
7	LSBFIRST	RW	0	Frame format 0: MSB is sent first; 1: LSB is sent first. Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur. Note: It is not used in I2S mode.
6	SPE	RW	0	SPI enable 0: Disable the SPI device; 1: Turn on the SPI device. Note: It is not used in I2S mode.
5: 3	BR [2: 0]	RW	0	Baud rate control (Baud rate control) 000: fPCLK/2 001: fPCLK/4 010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur. Note: Not used in I2S mode.
2	MSTR	RW	0	Master selection 0: Configure as a slave; 1: Configure as a master. Note: This bit cannot be modified while communication is in progress. Note: It is not used in I2S mode.
1	CPOL	RW	0	Clock polarity 0: In idle state, the SCK remains low; 1: in idle state, the SCK remains high. Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur. Note: It is not used in I2S mode.
0	CPHA	RW	0	Clock phase 0: data sampling starts from the first clock edge; 1: data sampling starts from the second clock edge. Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur. Note: It is not used in I2S mode.

35.4.2 SPI_CR2 Register

Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRF	Res		FRXTH	Res				TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	TXDMAEN	RXDMAEN
RW	-		RW	RW	RW	-		RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	FRF	RW	0	Frame format: 0: SPI Motorola mode 1: SPI Ti mode You can only operate on this bit after SPI is disabled (SPE = 0)
14: 13	Reserved	-	-	-
12	FRXTH	RW	0	FIFO reception threshold This bit is used to set the RXFIFO threshold for triggering the RXNE event. 1: If FIFO level is greater than or equal to 1/2, RXNE is generated

				0: There is data in FIFO
11: 8	Reserved	-	-	-
7	TXEIE	RW	0	Send buffer null interrupt enable (Tx buffer empty interrupt enable) 0: Disable TXE interrupt; 1: TXE interrupt is allowed, and an interrupt request is generated when the TXE flag is set to '1'.
6	RXNEIE	RW	0	Receive buffer non-empty interrupt enable (RX buffer not empty interrupt enable) 0: Disable RXNE interrupt; 1: RXNE interrupt is allowed, and an interrupt request is generated when the RXNE flag is set.
5	ERRIE	RW	0	Error interrupt enable (Error interrupt enable) When an error (CRCERR, OVR, MODF) is generated, this bit controls whether an interrupt is generated 0: Error interrupt is disabled; 1: Error interrupt is allowed.
4	CLRTXFIFO	RW	0	Clear TXFIFO Set by software and reset by hardware 0: No action 1: Clear TXFIFO Note: This bit should be written only when SPI is disabled (SPE = '0') for valid operation.
3	Reserved	-	-	-
2	SSOE	RW	0	SS output enable (SS output enable) 0: Prohibit SS output in main mode, and the device can work in multi-master mode; 1: When the device is turned on, turn on SS output in main mode, and the device cannot work in multi-master mode. Note: It is not used in I2S mode.
1	TXDMAEN	RW	0	Tx buffer DMA enable When this bit is set, a DMA request is issued once the TXE flag is set 0: Disable send buffer DMA; 1: Enable send buffer DMA.
0	RXDMAEN	RW	0	Rx buffer DMA enable When this bit is set, the RXNE flag issues a DMA request once it is set 0: Disable receive buffer DMA; 1: Enable receive buffer DMA.

35.4.3 SPI_SR Register

Address offset: 0x08

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			FTLVL		FRLVL		FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
-			R		R		R	R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
15: 13	Reserved	-	-	-
12: 11	FTLVL	R	0	FIFO transmission level These bits are set and cleared by hardware. 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full (considered as FULL when the FIFO threshold is greater than 1/2) Note: This bit is not used in I2S mode.
10: 9	FRLVL	R	0	FIFO reception level These bits are set and cleared by hardware. 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO is full Note: These bits are not used in I²S mode and in SPI receive-only mode while CRC calculation is enabled.

8	FRE	R	0	FRE: Frame format error This flag is used for SPI in TI slave mode This flag is set to 1 by the hardware and reset by the software when SPIx_SR is read. 0: No frame format error occurred 1: Frame format error occurred
7	BSY	R	0	Busy flag 0: the SPI is not busy; 1: the SPI is busy communicating, or the transmission buffer is not empty. This bit is set or reset by hardware.
6	OVR	R	0	Overrun flag 0: No overflow error occurred; 1: An overflow error occurred. TThis flag is set by hardware and reset by a software sequence.
5	MODF	RW	0	Mode fault 0: No mode error occurred ; 1: A mode error occurred. TThis flag is set by hardware and reset by a software sequence. Note: It is not used in I2S mode.
4	CRCERR	RCW0	0	CRC error flag 0: The received CRC value matches the value in the SPI_RXCRCR register; 1: The received CRC value does not match the value in the SPI_RXCRCR register. This flag is set by hardware and cleared by software writing 0. Note: It is not used in I2S mode.
3	UDR	R	0	Underrun flag 0: No underflow occurred; 1: Underflow occurred. This flag is set by hardware and reset by a software sequence. Note: This bit is not used in SPI mode.
2	CHSIDE	R	0	Channel side 0: the left channel needs to be transmitted or received; 1: the right channel needs to be transmitted or received. Note: This bit is not used in SPI mode. It has no significance in PCM mode.
1	TXE	R	1	Transmit buffer empty 0:Transmit buffer is not empty; 1: Transmit buffer empty.
0	RXNE	R	0	Receive buffer not empty 0:Receive buffer is empty 1: Receive buffer not empty.

35.4.4 SPI_DR Register

Address offset: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 0	DR [15: 0]	RW	0	Data register Used as an interface between the Rx and Tx FIFO. When reading the data register, RxFIFO is accessed, while writing the data register accesses TxFIFO Note: Data is always right-aligned. Unused bits are ignored when writing to the register, and read as zero when the register is read. The Rx threshold setting must always correspond to the currently used read access.

35.4.5 SPI_CRCPR Register

Address offset: 0x10

Reset value: 0x0007

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCPOLY [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15: 0	CRCPOLY [15: 0]	RW	0	CRC polynomial register (CRC polynomial register) This register contains polynomials used in CRC calculations. The CRC polynomial (0x0007) is the reset value of this register. Another polynomial can beconfigured as required. Note: These bits are not used in I2S mode. Note: The polynomial value should only be odd. No even values are supported.

35.4.6 SPI_RXCRCR Register

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxCRC [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
15: 0	RxCRC [15: 0]	R	0	When CRC calculation is enabled in the Receive CRC register, RXCRC [15: 0] contains the CRC value calculated from the received bytes. This register is reset when the CRCEN bit in SPI_CR1register is written to 1. The CRC is calculated serially using the polynomial programmed inthe SPI_CRCPR register. Only the 8 LSB bits are considered when the CRC frame format is set to be 8-bit length. CRC calculation is done based on any CRC8 standard. The entire 16-bits of this register are considered when a 16-bit CRC frame format is selected. CRC calculation is done based on any CRC16 standard. Note: A read to this register when the BSY Flag is set could return an incorrect value. Note: These bits are not used in I2S mode.

35.4.7 SPI_TXCRCR Register

Address offset: 0x18

Reset value: 0x0000

5	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TxCRC [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
15: 0	TxCRC [15: 0]	R	0	Tx CRC register When CRC calculation is enabled, the TXCRC[15:0] bits contain the computed CRC value of the subsequently transmitted bytes. This register is reset when the CRCEN bit in SPI_CR1 register is written to 1. The CRC is calculated serially using the polynomial programmed inthe SPI_CRCPR register. Only the 8 LSB bits are considered when the CRC frame format is set to be 8-bit length. CRC calculation is done based on any CRC8 standard. The entire 16-bits of this register are considered when a 16-bit CRC frame format is selected. CRC calculation is done based on any CRC16 standard. Note: A read to this register when the BSY Flag is set could return an incorrect value. Note: These bits are not used in I2S mode.

35.4.8 SPI_I2S_CFGR Register

Address offset: 0x1c

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				I2SMOD	I2SE	I2SCFG		PCMSYNC	Res.	I2SSTD		CKPOL	DATLEN		CHLEN
-				RW	RW	RW		RW	-	RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
15: 12	Reserved	-	-	-
11	I2SMOD	RW	0	I2S mode selection 0: Select SPI mode; 1: Select I2S mode. Note: This bit can only be configured when SPI or I2S is disabled.
10	I2SE	RW	0	I2S enable 0: Turn off I2S; 1: I2S enable. *Note: This bit is not used in SPI mode.*
9: 8	I2SCFG	RW	0	I2S configuration mode 00: transmitted from device 01: received from device 10: Master device transmission 11: Master device receives. Note: This bit can only be configurated when I2S is disabled. This bit is not used in SPI mode.
7	PCMSYNC	RW	0	PCM frame synchronization 0: Short frame synchronization 1: Long frame synchronization. Note: This bit is only meaningful if I2SSTD = 11 (using the PCM standard). This bit is not used in SPI mode.
6	Reserved	-	-	-
5: 4	I2SSTD	RW	0	I2S standard selection 00: I2S Philips standard; 01: High byte alignment standard (left alignment); 10: Low byte alignment standard (right alignment); 11: PCM Criteria. Note: For correct operation, these bits should be configured when the I2S is disabled. This bit is not used in SPI mode.
3	CKPOL	RW	0	Stationary clock polarity Steady state clock polarity 0: I2S clock quiescent state is low level; 1: The I2S clock stationary state is high. Note: For correct operation, these bits should be configured when the I2S is disabled. This bit is not used in SPI mode.
2: 1	DATLEN	RW	0	Data length to be transmitted (Data length to be transferred) 00: 16-bit data length; 01: 24-bit data length; 10: 32-bit data length; 11: Not allowed. Note: For correct operation, these bits should be configured when the I2S is disabled. This bit is not used in SPI mode.
0	CHLEN	RW	0	Channel length (number of data bits per audio channel Channel length (number of bits per audio channel) 0: 16 bits wide; 1: 32 bits wide. The write operation of this bit only makes sense if DATLEN = 00, otherwise the channel length is fixed to 32 bits by hardware. Note: For correct operation, these bits should be configured when the I2S is disabled. This bit is not used in SPI mode.

35.4.9 SPI_I2SPR Register

Address offset: 0x20

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.						MCKOE	ODD	I2SDIV							
-						RW	RW	RW							

Bit	Name	R/W	Reset Value	Function
15: 10	Reserved	-	0	-
9	MCKOE	RW	0	Master device clock output enabled (Master clock output enable) 0: Turn off the master device clock output; 1: Master device clock output enabled. Note: For correct operation, these bits should be configured when the I2S is disabled. It is used only when the I2S is in master mode. This bit is not used in SPI mode.
8	ODD	RW	0	Odd coefficient pre-division Odd factor for the prescaler 0: actual frequency division coefficient = I2SDIV * 2; 1: actual frequency division coefficient = (I2SDIV * 2) +1. Note: For correct operation, these bits should be configured when the I2S is disabled. It is used only when the I2S is in master mode. This bit is not used in SPI mode.
7: 0	I2SDIV	RW	0	I2S linear prescaler Disable setting I2SDIV [7: 0] = 0 or I2SDIV [7: 0] = 1 Note: For correct operation, these bits should be configured when the I2S is disabled. It is used only when the I2S is in master mode. This bit is not used in SPI mode.

36. Inter-integrated circuit (I2C) interface

36.1 Introduction

The I2C (Inter-integrated circuit) bus interface handles communications between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports Standard-mode (Sm), Fast-mode (Fm) and Fast-mode plus (Fm+). Depending on the needs of a specific device, DMA can be used to reduce the burden on the CPU.

36.2 Main Features

- Slave and master modes
- Multi-master function: can be used as a master or a slave
- As Master
  - − Generate clock
  - − Generate start and stop signals
- As Slave
  - − Programmable I2C address detection
  - − Dual address capability that can respond to 2 slave addresses (with 1 configurable mask)
  - − Detect stop bit
- Supports 7/10-bit addressing mode
- Support broadcast call function
- Supports different communication speeds
  - − Standard (up to 100 KHz)
  - − Fast (up to 400 KHz)
  - − Fast enhancement (up to 1 MHz)
- Status flag bit:
  - − Send/receive mode flag
  - − Byte transfer completion flag
  - − I 2 C Bus busy flag
- I2S error flag
  - − Host Mode Arbitration Lost
  - − Answer (ACK) after address/data transfer failed
  - − Error start/stop condition
  - − Disable overflow or underflow when stretching the clock function
- 2 interrupt vectors
  - − 1 event interrupt for address/data communication success
  - − 1 error interrupt
- Optional length clock function
- Support DMA function
- Generation or verification of configurable PEC (Packet Error Detection):
  - − The PEC value can be transmitted as the last byte in the transmit mode

- PEC error check for last received byte

- SMBus 2.0 support
  - 25 ms Clock low timeout delay
  - 10 ms master accumulated clock low extension time
  - 25 ms slave accumulated clock low extension time
  - Hardware PEC generation/verification with ACK control
  - Supports Address Resolution Protocol (ARP)
  - Timeout and idle condition detection
- Support SMBus
- Supports wake-up from Stop mode when addresses match
  - Supports configurable timeout time during wake-up

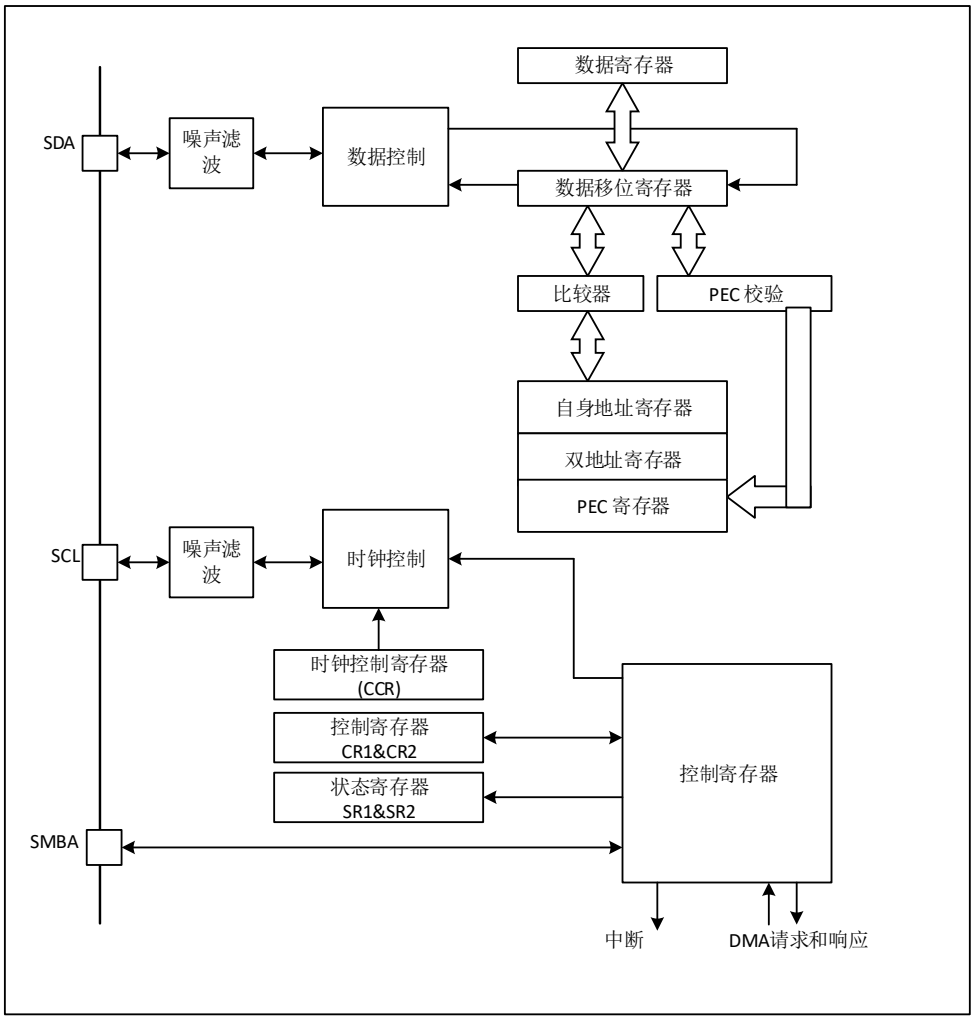


Figure 36-1 I2C Module

36.3 I2C functional description

36.3.1 Introduction

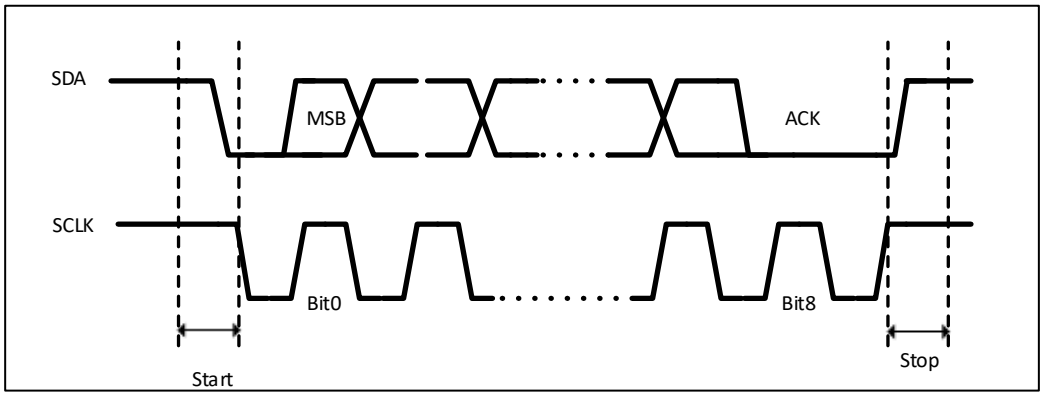


Figure 36-2 I2C bus protocol

The interface can operate in one of the four following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

By default, the I2C interface operates in Slave mode. The interface automatically switches from the slave mode to the master mode after generating the start condition; When the arbitration is lost or a stop signal is generated, the master mode is switched to the slave mode in the case where the multi-master function is allowed

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

As a slave, the I2C interface can recognize its own address (7-bit or 10-bit) and the broadcast call address. The General Call address detection may be enabled or disabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The 1 or 2 bytes following the start condition are addresses (1 byte for 7-bit mode and 2 bytes for 10-bit mode). Addresses are only sent in master mode.

Within the first always clock after a byte is transmitted, the receiver must send back an acknowledgement bit (ACK) to the sender.

36.3.2 Clocks

The input clock of the I2C module is the APB clock. The module generates an SCL clock internally and then outputs it to the bus to the slave.

The SCL generation mechanism is as follows: There is a counter that counts the high and low levels according to the value of the CCR register (the CCR register determines the length of the high and low levels). In addition, the CCR register also controls the selection of standard mode, fast mode and fast boost mode, as well as the duty cycle of SCL high and low levels.

36.3.3 Packet Error Check (PEC)

The packet error check (PEC) calculator is used to improve the reliability of communication. This calculator uses a programmable polynomial to calculate each bit of data.

● The PEC calculation is enabled bythe ENPEC bit of the I2C_CR1 register. The PEC uses the CRC-8 algorithm to calculate all information bytes, including addresses and read/write bits.

− On transmission: The PEC bit in the I2C_CR1register is set at the last TxE event, and the PEC willbe transmitted afterthis byte.

− On receive: Set the PEC bit in the I2C_CR1 register at the last RxNE event, and if the next received byteisnot equal to the internally calculated PEC,the receiverwillsend a NACK. If it is the main receiver, regardless of the result of proofreading, NACK will be sent after PEC. The PEC bit must be set before receiving an ACK pulse for the current byte.

● The PECERR error flag/interrupt is available in the I2C_SR1 register.

● If both DMA and PEC calculators are activated:

− At the time of transmission: When the I2C interface receives an EOT (corresponding to the number of bytes transmitted by DMA) signal from the DMA controller, it automatically sends a PEC after the last byte.

− On reception: When the I2C interface receives an EOT_1signal from the DMA, it will automatically take the next byte as the PEC and will check it. A DMA request is generated after the PEC is received.

● To allow intermediate PEC transmissions, there is a control bit (LAST bit) in the I2C_CR2 register that determines whether it is really the LAST DMA transmission. If it is indeed the DMA request of the last master receiver, the NACK is automatically sent after the last byte is received.

● PEC calculation is invalid when arbitration is lost.

36.3.4 Slave mode of I2C

By default, the I2C interface operates in Slave mode. To switch from slave mode to master mode, you need to generate a start condition. The peripheral input clock must be programmed in the I2C_CR2 register in order to generate correct timings. The peripheral input clock frequency must be at least:

2 MHz in Sm mode

4 MHz in Fm mode

In Fast mode plus: 16MHz

Set slave address and address masking:

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register. Then it is compared with the address of the interface (OAR1) or the General Call address (if ENGC = 1). If extra addresses are enabled (ENDUAL = 1), the address is compared to the chip's OAR2 address, and the address in the OAR2 register can be masked by OA2MSK [2: 0], starting at the 7th bit upper of OA2, only OA2 [7: 2], OA2 [7: 3], OA2 [7: 4], OA2 [7: 5], OA2 [7: 6] or OA2 [7] are compared to the received address, respectively, when OA2MSK is configured to 1 through 6. If OA2MSK = 7, all 7-bit addresses received (except reserved addresses) are answered.

Address mismatch (header segment or address mismatch): The I2C interface ignores it and waits for another start condition.

Address matching: The I2C interface generates the following timing:

● An acknowledged pulse if the ACK bit is set

● The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set

● If ENDUAL = 1, the software must read the DUALF bit to confirm which slave address was responded to

In slave mode that TRA bit indicate whether it is currently in receiver mode or transmitter mode

36.3.4.1 Slave send mode

After receiving the address and clearing the ADDR bit, (if the lowest bit of the address byte is 1) Slave sends data (bytes) from the DR register to the SDA via the internal shift register.

Slave pulls the SCL low until the ADDR bit is cleared and the data to be sent has been written to the DR register (refer to EV1, EV3).

When the acknowledge pulse is received: The TxE bit is set by hardware with an interrupt if the ITEVFEN and the ITBUFEN bits are set.

If TxE is set and some data were not written in the I2C_DR register before the end of the next data transmission, the BTF bit is set. Slave pulls the SCL down until the BTF bit is cleared by the software (after reading I2C_ SR1, writing to the I2C_DR register).

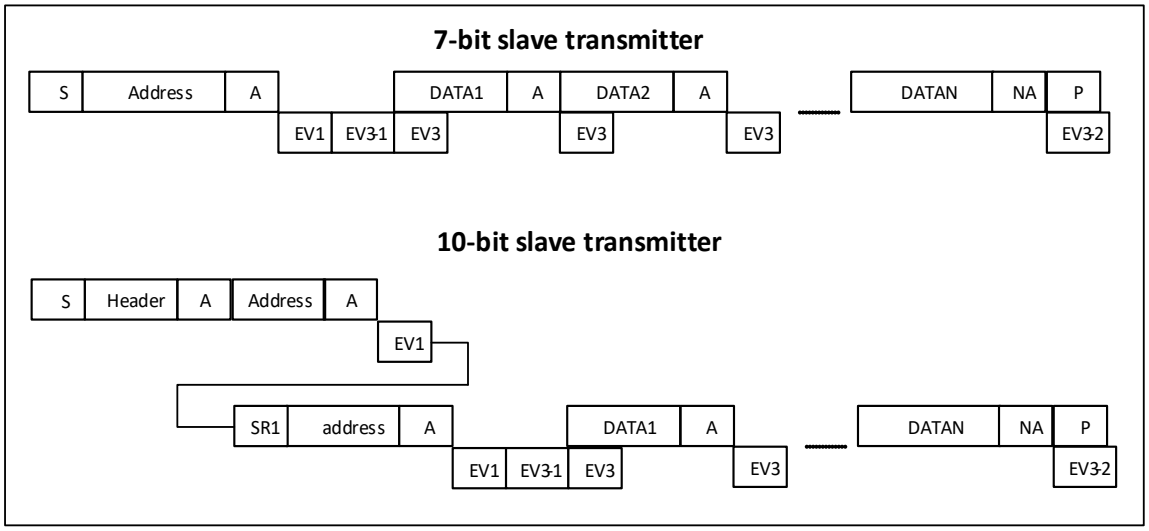


Figure 36-3 Transfer sequence diagram for slave transmitter

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV1: ADDR = 1, the ADDR bit is cleared by reading the SR1 register first and then the SR2 register

EV3-1: TxE=1, shift register empty, data register empty, write Data1 in DR.

EV3: TxE=1, shift register not empty, data register empty, cleared by writing DR

EV3-2: AF = 1; The software writes 0 to the AF bit to clear the bit

Notes:

1) EV1 and EV3-1 events pull down the SCL until the end of the corresponding software sequence.

2) The software sequence of EV3 must be completed before the end of the current byte transfer

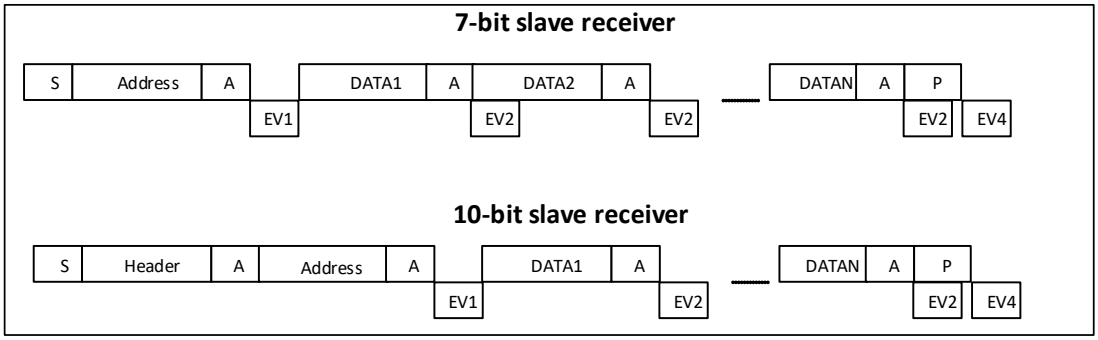
36.3.4.2 Slave receive mode

After receiving the address and clearing the ADDR, (if the lowest bit of the address byte is 0) slave will store the byte received from the SDA line into the DR register through the internal shift register. After each byte the interface generates in sequence:

● If the ACK bit is set, an answer pulse is generated

● Hardware settings RxNE = 1. An interrupt is generated if the ITEVTEN and ITBUFEN bit is set.

If RxNE is set and the DR register is not read until the end of the reception of new data, the BTF bit is set and slave keeps pulling down the SCL until the BTF is cleared (reading the I2C_DR register after reading the I2C_SR1). (See figure below).



Transmission sequence diagram 36-4from receiver

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV1: ADDR = 1, the ADDR is cleared by reading SR1 first and then reading SR2

EV2: RxNE=1 cleared by reading DR register

EV4: STOPF=1, cleared by reading SR1 register followed by writing to the CR1 register.

Notes:

1) The EV1 event pulls down the SCL until the end of the corresponding software sequence.

2) The EV2 software sequence must be completed before the current byte transfer is complete.

3) When the user checks the contents of the SR1 register, a complete clear sequence should be performed for each set flag bit. Thus, for ADDR and STOPF flags, the following sequence is required inside the I2C interrupt routine:

If ADDR = 1, reading SR1 followed by reading SR2; if STOPF =1, reading SR1 followed by reading CR1.

The purpose is to make sure that both ADDR and STOPF flags are cleared if both are found set.

36.3.4.3 Closing slave communication

After the last data byte is transferred, a Stop Condition is generated by the master. The interface detects this condition and sets:

● The hardware sets STOPF and generates an interrupt if the ITEVTEN bit is set.

The STOPF bit is cleared by a read of the SR1 register followed by a write to the CR1 register.

36.3.5 Master mode of I2C

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition.

Master mode is selected as soon as the Start condition is generated on the bus with a START bit.

The following is the required sequence in master mode:

- Program the peripheral input clock in I2C_CR2 register in order to generate correct timings
- Configure the clock control registers
- Configure the rise time register
- Program the PE bit in I2C_CR1 register to enable the peripheral
- Set the START bit in the I2C_CR1 register to generate a Start condition

The peripheral input clock frequency must be at least:

- 2 MHz in Sm mode
- 4 MHz in Fm mode
- In Fast mode plus: 16MHz

36.3.5.1 SCL master clock generation

The CCR register counts with the rising edge of the input clock, generating the high and low levels of the SCL. As a slave may stretch the SCL line, the peripheral checks the SCL input from the bus at the end of the time programmed in I2C_TRISE register after rising edge generation.

— If the SCL line is low, it means that a slave is stretching the bus, and the high level counter stops until the SCL line is detected high. This allows to guarantee the minimum HIGH period of the SCL clock parameter.

— If the SCL line is high, the high level counter keeps on counting.

Indeed, the feedback loop from the SCL rising edge generation by the peripheral to the SCL rising edge detection by the peripheral takes time even if no slave stretches the clock. This loopback duration is linked to the SCL rising time, plus delay due to the noise filter present on the SCL input path, plus delay due to internal SCL input synchronization with APB clock. The maximum time of the feedback loop is set in the TRISE register, so the frequency of the SCL remains stable regardless of the SCL rise time.

36.3.5.2 Start condition

Setting the START bit causes the interface to generate a Start condition and to switch to Master mode (MSL bit set) when the BUSY bit is cleared.

Note: Setting the START bit in host mode will generate a restart condition by the hardware after the current byte transfer is completed.

Once the Start condition is sent:

● The SB bit is set by hardware, and if the ITEVTEN bit is set, an interrupt is generated.

●The masterreadsthe I2C_ SR1 register andwrites the slave addresstothe I2C_ DR register.

36.3.5.3 Slave address transmission

Then the slave address is sent to the SDA line via the internal shift register.

● In 10-bit address mode, sending a header sequence produces the following events:

The − ADD10 bit is set by hardware, and if the ITEVFEN bit is set, an interrupt is generated.

The host then reads the SR1 register and writes the second address byte to the DR register.

− The ADDR bit is set by hardware, and if the ITEVFEN bit is set, an interrupt is generated.

The host then reads the SR1 register and then the SR2 register.

● In 7-bit address mode, an address byte will be sent out.

Once the address byte is sent out,

− The ADDR bit is set by hardware, and if the ITEVFEN bit is set, an interrupt is generated.

The master then waits for a read of the SR1 register, followed by a read of the SR2 register.

According to the lowest bit of the outgoing slave address, the master device decides whether to enter the transmitter mode or the receiver mode.

● In 7-bit address mode,

− To enter transmitter mode, the master device sends the slave address so that the lowest bit is equal to 0.

− To enter receiver mode, the master device sends the slave address so that the lowest bit is equal to 1.

● In 10-bit address mode

− To enter transmitter mode, the master device first sends the header byte (11110xx0), and then sends the slave address with the lowest bit equal to 0. (The xx in the header byte is the highest 2 bits of the 10-bit address.)

− To enter receiver mode, the master device first sends the header byte (11110xx0), and then sends the slave address with the lowest bit equal to 1. A start condition is then retransmitted, followed by the header byte (11110xx1). (xx in the header byte is the highest 2 bits of the 10-bit address).

The TRA bit indicate whether the master device is in receiver mode or transmitter mode

36.3.5.4 Host send mode

After sending the address and clearing the ADDR bit, the host sends the data bytes from the DR register to the SDA line through the internal shift register.

The host waitsuntilthe first data byte is written tothe DR register(seeEV8_1).

When the acknowledge pulse is received, the TxE bit is set by hardware and an interrupt isgenerated if the ITEVFEN and ITBUFEN bits are set.

If TxE is set and some data were not written in the DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read from I2C_SR1 followed by a write to I2C_DR, stretching SCL low.

Closing slave communication

After the last byte is written in the DR register, a STOP condition is generated by setting the STOP bit (see EV8_2 in Figure), and then the I2C interface will automatically return to slave mode (MSL bit clear).

Note: Stop condition should be programmed during EV8_2 event, when either TxE or BTF is set.

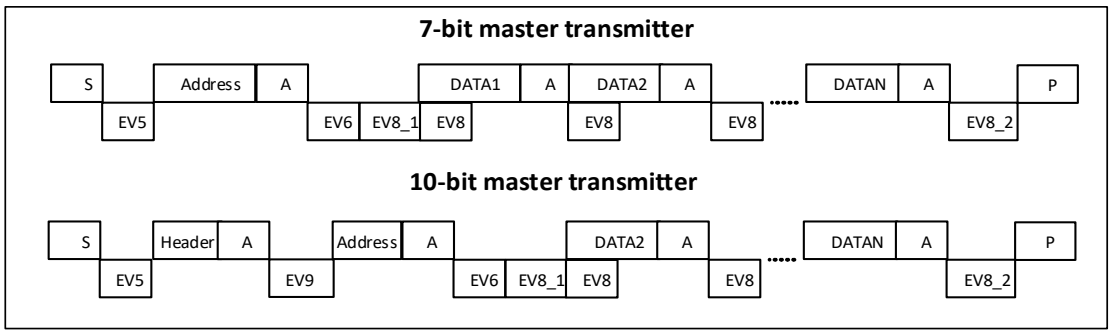


Figure 36-5 Transfer sequence diagram for master transmitter

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV5: SB = 1, cleared by reading SR1 register followed by writing DR register with Address.

EV6: ADDR = 1, read SR1, read SR2 again, clear the event

EV8_1: TXE = 1, shift register empty

EV8: TXE = 1, cleared by writing DR register

EV8_2: TXE = 1, BTF = 1, cleared by hardware by the Stop condition

EV9: ADDR10 = 1, read SR1 and write DR register to clear the event

Notes:

The 1. EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

The 2. EV8 software sequence must complete before the end of the current byte transfer. If the software sequence of EV8 cannot be completed before the end of the currently transmitted byte, BTF is recommended instead of TxE

36.3.5.5 Host Receive Mode

Following the address transmission and after clearing ADDR, the I2C interface enters Master Receiver mode. In this mode the interface receives bytes from the SDA line into the DR register via the internal shift register. After receiving each byte, the I2C interface performs the following operations in turn:

● If the ACK bit is set, an answer pulse is emitted.

● The RxNE bit is set and an interrupt is generated if the INEVFEN and ITBUFEN bits are set.

If the RxNE bit is set and the data in the DR register is not read before the end of the last data reception, the BTF bit is set by hardware and the interface waits until BTF is cleared by a read in the I2C_SR1 register followed by a read in the I2C_DR register, stretching SCL low.

Closing slave communication

Method 1: The application scenario of this method is that when the I2C interrupt is set to the highest priority in the application program

After receiving the last byte from the slave, the master sends a NACK. After receiving this NACK, the slave releases the control of the SCL and SDA lines. Then the master can send a Stop/Restart condition.

1) To generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just after reading the second last data byte (after second last RxNE event).

2) To generate the Stop/Restart condition, software must set the STOP/START bit just after reading the second last data byte (after the second last RxNE event).
3) Whena single byteis received,the generation bit forclearing ACKand stop conditions shouldbe just after EV6 (after clearing ADDR at EV6_1). After the Stop condition generation, the interface goes automatically back to slave mode (MSL bit cleared).

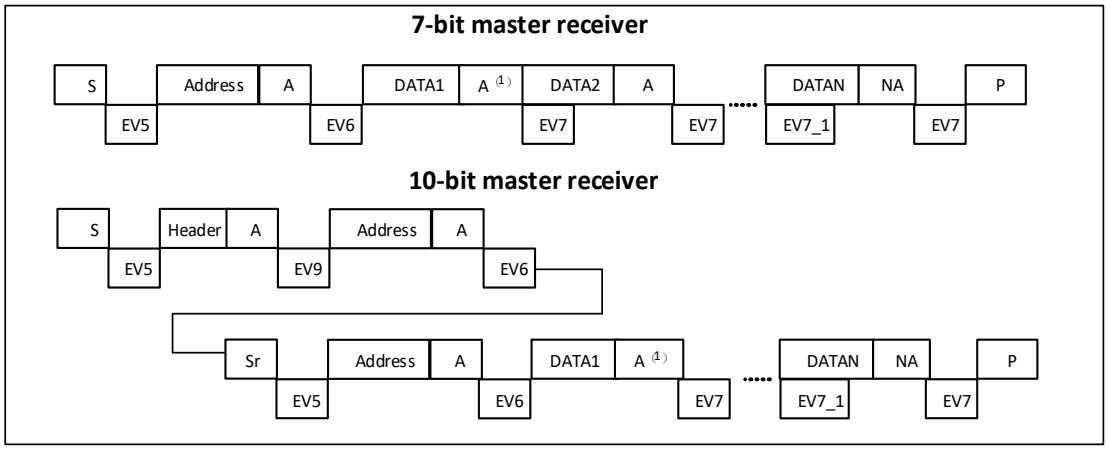


Figure 36-6 Method 1: transfer sequence diagram for master receiver

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV6_1: no associated flag event, used for 1 byte reception only.

EV7: RxNE=1 cleared by reading DR register.

EV7_1: RxNE=1 cleared by reading DR register, program ACK=0 and STOP request

EV9: ADD10 = 1, read SR1 and write DR register to clear the event

Notes:

1) If a single byte is received, it is NA.
2) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
3) The EV7 software sequence must complete before the end of the current byte transfer. In EV7, if the software sequence cannot be executed before the currently transmitted byte transfer is completed, it is recommended to use BTF instead of RXNE.
4) The software sequence ofEV6_1 orEV7_ 1 must becompleted beforethe ACK bitof the current byte transmission is transmitted.

Method 2: The application scenario of this method is that the interrupt of I2C is not the highest priority in the application, or the query method is used

With this method, if DataN-2 is not read, after DataN-1, the communication will be stretched (both RxNE and BTF are set). Then, clear the ACK bit before reading DataN-2 in DR register to ensure it is cleared before the DataN Acknowledge pulse. After reading DataN-2, the STOP/START bits are set and DataN-1 is read. After RxNE is set, read DataN.

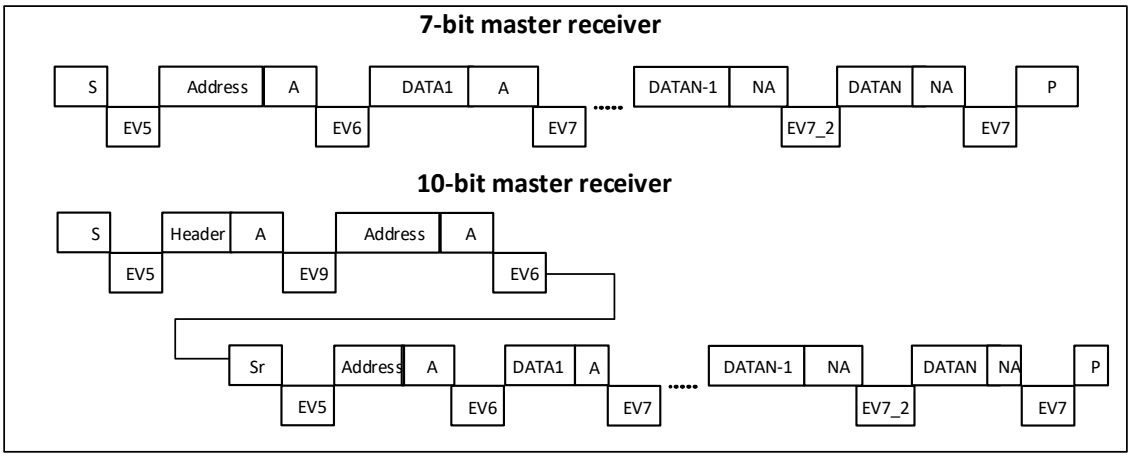


Figure 36-7 Method 2: transfer sequence diagram for master receiver when N>2

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV5: SB = 1, read the SR1 register first, read the SR2 register first, then write the DR register, and clear this bit

EV6: ADDR, read SR1 first, then read SR2, clear this bit

EV7: RxNE=1 cleared by reading DR register

EV7_2: BTF = 1, DataN-2 in DR register and DataN-1 in shift register, program ACK = 0, Read DataN-2 in DR. Program STOP = 1, read DataN-1.

EV9: ADD10 = 1, read SR1 and write DR register to clear the event

Notes:

1) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
2) The EV7 software sequence must complete before the end of the current byte transfer. In EV7, if the software sequence cannot be executed before the currently transmitted byte transfer is completed, it is recommended to use BTF instead of RXNE.

● When 3 bytes are to be read away:

— RXNE = 1, DataN-2 Not reading.
— DataN-1 received
— BTF = 1, Shift and DR registers full. The DR register stores DataN-2 and the shift register stores DataN-1. At this time, the SCL is pulled low and there is no other data to be received on the bus
— Clear ACK bit
— Read DataN-2 in the DR register, which will start the reception of DataN by the shift register
— DataN received (with a NACK)
— Program START/STOP bit
— Read DataN-1
— RxNE=1
— Read DataN

The procedure described above is valid for N>2. The cases where a single byte or two bytes are to be received should be handled differently, as described below:

● Case of 2 bytes received

— Set POS and ACK bit

— Wait for the ADDR flag to be set

— Clear ADDR bit

— Clear ACK bit

— Wait for BTF to be set

— Program STOP bit

— Read DR twice

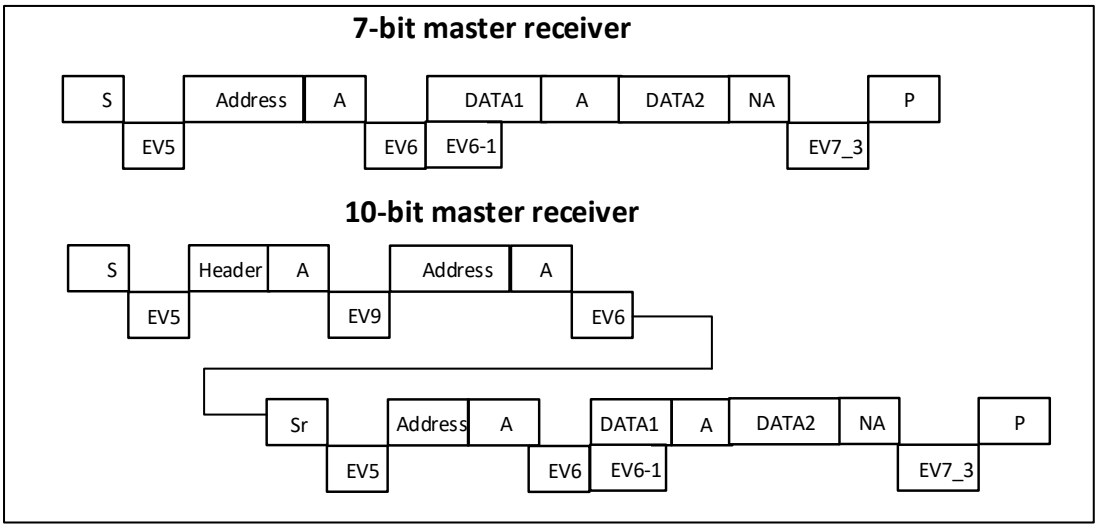


Figure 36-8 Method 2: transfer sequence diagram for master receiver when N=2

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV5: SB=1, cleared by reading SR1 register followed by writing the DR register.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV6_1: No associated flag event. The acknowledge disable should be done just after EV6, that is after ADDR is cleared.

EV7_3: BTF = 1, program STOP = 1, read DR twice (Read Data1 and Data2) just after programming the STOP.

EV9: ADD10 = 1, read SR1 and write DR register to clear the event

Notes:

1) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
2) The software sequence of EV6_1must becompletedbeforethe ACK bitof the current byte transmission is sent.

● Case of single byte reception

— In the ADDR event, clear the ACK bit

— Clear ADDR

— Program STOP/START bit

— Read the data after the RxNE flag is set.

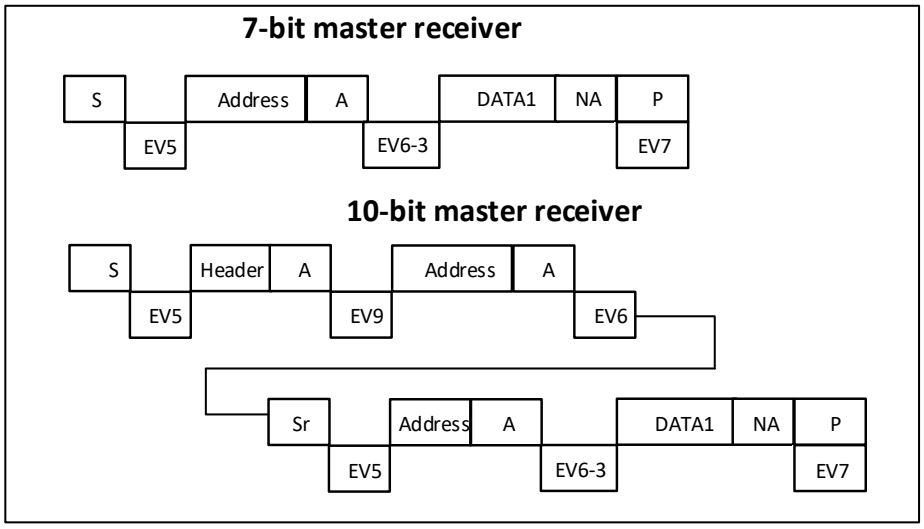


Figure 36-9 Method 2: transfer sequence diagram for master receiver when N=1

Description: S = Start, SR = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV5: SB=1, cleared by reading SR1 register followed by writing the DR register.

EV6_3: ADDR = 1, program ACK = 0. Clear ADDR by reading SR1 register followed by reading SR2 register. Program STOP =1 just after ADDR is cleared.

EV7: RxNE=1 cleared by reading DR register

EV9: ADD10 = 1, read SR1 and write DR register to clear the event

Notes:

EV5, EV6, EV8_1, and EV8_2 events stretch the low level of the SCL untilthe end of execution ofthe corresponding software sequence.

36.3.6 Error flags

Bus error (**BERR)**

This error occurs when the I2C interface detects an external Stop or Start condition during an address or a data transfer. In this case:

- BERR Bit is set to '1'; If the ITERREN bit is set, an interrupt is generated;
- In slave mode: the data is discarded and the hardware releases the bus.
  - In case of a misplaced Start, the slave considers it is a restart and waits for an address, or a Stop condition
  - In case of a misplaced Stop, the slave behaves like for a Stop condition and the lines are released by hardware
- In host mode: The hardware does not release the bus and does not affect the current transmission state. It is up to the software to abort or not the current transmission.

Answer Failure (AF)

This error occurs when the interface detects a non-acknowledge bit. In this case:

- The AF bit is set and if the ITERREN bit is set, an interrupt is generated
- When the transmitter receives a NACK, the communication must be reset.

— If in slave mode, hardware releases the bus

— If in host mode, the software must generate a stop condition or restart

Arbitration Loss (ARLO)

This error occurs when the I2C interface detects an arbitration lost condition. In this case:

● The ARLO bit is set by hardware, and if the ITERREN bit is set, an interrupt is generated

● The I2C interface automatically returns to slave mode (the MSL bit is cleared). When the I2C loses the arbitration, it is not able to acknowledge its slave address in the same transfer, but it can acknowledge it after a repeated Start from the winning master.

● Hardware release bus

Overload/Underload Error (OVR)

An overrun error can occur in slave mode when clock stretching is disabled and the I2C interface is receiving data. The interface has received a byte (RxNE=1) and the data in DR register has not been read, before the next byte is received by the interface.

In this case:

● The last received data is discarded

● In the event of an overload error, the software shall clear the RxNE bit and the transmitter shall resend the last transmitted byte

Underrun error can occur in slave mode when clock stretching is disabled and the I2C interface is transmitting data. The interface has not updated the DR with the next byte (TxE=1), before the clock comes for the next byte. In this case:

● The previous byte in the DR register will be issued repeatedly

● The user should determine that in the event of an underload run error, the receiving end should discard repeatedly received data. The next bytes are written within the clock low time specified in the I2C bus standard.

For the first byte to be transmitted, the DR must be written after ADDR is cleared and before the first SCL rising edge. If not possible, the receiver must discard the first data.

36.3.7 DMA functionality

The DMA request (when enabled) is used only for data transmission. DMA requests are generated by data register becoming empty in transmission and data register becoming full in reception. The DMA must be initialized and enabled before the I2C data transfer. The DMAEN bit must be set in the I2C_CR2 register before the ADDR event.

In master or slave mode, when the clock extension function is enabled, DMAEN can be set during the ADDR event before the ADDR is cleared. The DMA request must be served before the end of the current byte transfer. When the DMA transmission data length reaches the value set by DMA, the DMA controller sends an EOT (End of transfer) to the I2C and generates a transmission completion interrupt (if the interrupt enable bit is valid):

● Host send mode: In the EOT interrupt service routine, DMA requests need to be prohibited, and then the STOP condition needs to be set after waiting for the BTF event.

●Host acceptance mode: When the number of data to be received is greater than or equal to 2,the DMA controllersends a hardware signal EOT_1 (corresponding to the number of DMA transmission bytes-1). If, in the I2C_CR2 register, the LAST bit is set, I2C automatically sends a NACK after the next byte following EOT_1. The user can generate a Stop condition in the DMA transfer complete interrupt routine if enabled.

36.3.8 Support wakeup from STOP mode

When the MCU is in STOP mode, the address can be received through I2C. If the address matches, the MCU can be awakened, otherwise, the STOP state can be maintained.

1) To implement the above functions, it is necessary toenable the PE and WUPEN of I2C_CR1 before entering the STOP mode, and to enable the clock of I2C in the RCC.
2) The STOP modewake-up processalso supports a configurable timeout time, which can be configured by setting WKUP_CNT and WKUP_DIV of I2C_CR1, wakes up the MCU after generating timeout if ITERREN is enabled, andmaintains the STOP stateafter generating timeout if ITERREN is not enabled.
3) Before enteringSTOP mode, I2C_CR1needs to be reasonably configured according to the current communication speed. WKUP_CNTandI2C_ CR1.WKUP_DIV, otherwise it will result in timeout and the address cannot be matched.

36.3.9 Interrupts and events

Table 36-1 I²C interrupt requests

Interrupt event	Event flag	Enable control bit
Start bit sent (Master)	SB	ITEVTEN
Address sent (Master) or address match (Slave)	ADDR	
ADD10	10-bit header sent (primary)	
Slave Received	STOPF	
Data byte transfer finished	BTF	
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error (BERR)	BERR	ITERREN
Arbitration Lost (Master)	ARLO	
Acknowledge failure	AF	
Overrun/Underrun	OVR	
PEC error	PECERR	
Timeout/Tlow error	TIMEOUT	
SMBus Alert	SMBALERT	

36.3.10 System management bus SMbus

The System Management Bus (SMBus) is a two-wire interface. Through it, various devices can communicate with each other and with the rest of the system. It is based on I2C operating principles. SMBus provides a control bus for system and power management-related tasks. A system using SMBus can communicate information with multiple devices without the need to use separate control lines.

The System Management Bus (SMBus) standard relates to three categories of devices. Slave device: The device that receives or responds to the command. Master: The device used to issue orders, generate clocks, and terminate transmissions. Host: is a dedicated master device that provides the main interface with the system CPU. The master must have master-slave functionality and must support the SMBus reminder protocol. Only one host is allowed in a system.

Using the system management bus, the device can provide manufacturer information, tell the system its model/part number, save the status of suspended events, report different types of errors, receive control parameters, and return its status. SMBus provides a control bus for system and power management-related tasks.

The I2C module of this product supports SMbus/PMbus functionality.

Table 36-2 Differences between SMbus and I2C Comparison:

SMBus	**I2C**
Maximum speed 100 kHz	Maximum speed 400 kHz
Minimum clock speed 10 kHz	No minimum clock speed
35 ms clock low timeout	No timeout
Logic levels are fixed	Logic levels are VCC dependent
Different address types (reserved, dynamic, etc.)	7-bit, 10-bit and general call slave address types
Different bus protocols (quick command, process call etc.)	No bus protocols

Similarities between SMBus and I2C:

● Two-line bus protocol (1 clock, 1 data) + optional SMBus reminder line

● Master-slave communication, master device provides clock

● Multi-host function

● The SMBus data format is similar to I2C's 7-bit address format

36.3.10.1 Address resolution protocol (ARP)

SMBus slave address conflicts can be resolved by dynamically assigning a new unique address to each slave device.

ARP has the following properties:

● Address allocation uses standard SMBus physical layer arbitration mechanism

● While the device maintains power, the assigned address remains the same, allowing the device to retain its address when the power is lost.

● After address allocation, there is no additional SMBus packaging overhead (that is, the device accessing the allocated address takes the same time as the device accessing the fixed address).

● Any SMBus master device can traverse the bus.

36.3.10.2 SMBus alert mode

The SMBus reminder is an optional signal with an interrupt line for devices that want to expand their control capabilities at the expense of one pin. SMBALERT, like SCL and SDA signals, is a line and signal. SMBALERT is commonly used with SMBus broadcast call address. Messages related to SMBus are 2 bytes.

A devicewith onlyslavefunctionalitycansignalto the master that it wishes to communicate viaSMBALERTby setting the ALERT bit on the I2C_CR1 register. The host processes the interrupt and accesses all SMBALERT devices via an Alert Response Address (ARA) (address value 0001100x). Only the device(s) which pulled SMBALERT low will acknowledge the alert response address. This status is identified using SMBALERT Status flag in I2C_SR1 register The host performs a modified receive byte operation. The 7 bit device address provided by the slave transmit device is placed in the 7 most significant bits of the byte. The eighth bit canbe a zero or one.

If multiple devices pull SMBALERT low, the highest priority device (the smallest address) will win communication rights through standard arbitration during address transmission. After confirming the slave

address, the device does not need to pull down its SMBALERT again. If the host still sees that SMBALERT is low after the message transmission is completed, it knows that it needs to read the ARA again.

Hosts that do not implement the SMBALERT signal can access ARA periodically.

36.3.10.3 Timeout error

There are many differences between I2C and SMBus in timing specifications.

SMBus defines a clock low timeout, TIMEOUT of 35 ms. SMBus specifies that TLOW: SEXT is the cumulative clock low expansion time of the slave device. SMBus specifies that TLOW: MEXT is the accumulated clock low expansion time of the master device. For more timeout details, please refer to version 2.0 of the SMBus specification (http://smbus.org/specs/). The status flag TIMEOUT error in I2C_SR1 indicates the status ofthis feature.

36.3.10.4 Interfaces using SMBus mode

TIMOUTEN and TEXTEN positions 1 in the I2C_TIMEOUTR register are used to enable timeout detection. The timer must be programmed in such a way that a timeout condition is detected before the maximum time specified by the SMBus specification.

tTIMEOUT inspect

To enable the tTIMEOUT check, the 12-bit TIMEOUTA [11: 0] bit must be programmed as a timer overload value to check the tTIMEOUT parameter. The TIDLE bit must be configured to "0" to detect an SCL low timeout.

The timer is then enabled by placing the TIMOUTEN position 1 in the I2C_TIMEOUTR register.

If the low duration of the SCL exceeds (TIMEOUTA+1) x 2048 x tPCLK, the TIMEOUT flag of the I2C_SR1 register will be set to 1.

(max tTIMEOUT = 25 ms).

Note: At TIMEOUTEN position 1, it is not allowed to change the configuration of the TIMEOUTA [11: 0] bit and the TIDLE bit.

• tLOW: SEXT and tLOW: MEXT check

The TIMEOUTB timer must be configured to verify tLOW: SEXT for the slave and tLOW: MEXT for the master depending on whether the peripheral is configured as the master or the slave. As the standard specifies only a maximum, the user can choose the same value for the both.

The timer is then enabled by putting the TEXTEN position 1 in the I2C_TIMEOUTR register.

If the accumulation time of the SMBus peripheral extension SCL exceeds (TIMEOUTB+1) x 2048 x tPCLK, the TIMEOUT flag in the I2C_SR1 register will be set to 1.

See below: Example TIMEOUTB settings at different PCLK frequencies.

NOTE: Changes to the TIMEOUTB configuration are not allowed at TEXTEN position 1.

36.3.10.5 Bus idle detection

To enable tIDLE checking, the 12-bit TIMEOUTA [11: 0] field must be programmed as a timer overload value to get the tIDLE parameter. The TIDLE bit must be configured to "1" to detect SCL and SDA high timeouts.

The timer is then enabled by placing the TIMOUTEN position 1 in the I2C_TIMEOUTR register.

If the high level duration of the SCL and SDA lines exceeds (TIMEOUTA+1) x 4 x tPCLK, the TIMEOUT flag in the I2C_SR1 register will be set to 1.

See Table: Examples of TIMEOUTA settings at different PCLK frequencies (maximum tIDLE = 50 µ s).

Note: When TIMEOUTEN is set to 1, it is not allowed to change the TIMEOUTA and TIDLE configuration.

36.3.10.6 PMbus

PMbus is based on SMbus, and the transmission logic is exactly the same as SMbus. The difference is that PMbus defines some functions related to power management (done by software).

36.3.10.7 SMBus: I2C_TIMEOUTR Register Configuration Example

This section is relevant only when SMBus feature is supported.

- Configure the maximum duration of tTIMEOUT to 25 ms:

Table 36-3 Example TIMEOUTA settings at different PCLK frequencies (maximum tTIMEOUT = 25 ms)

fPCLK	TIMEOUTA [11: 0]	TIDLE	TIMEOUTEN	tTIMEOUT
8 MHz	0x61	0	1	98 x 2048 x 125 ns = 25 ms
16 MHz	0xC3	0	1	196 x 2048 x 62.5 ns = 25 ms
32 MHz	0x186	0	1	391 x 2048 x 31.25 ns = 25 ms
48 MHz	0x249	0	1	586 x 2048 x 20.08 ns = 25 ms

- Configure the maximum duration of tLOW: SEXT and tLOW: MEXT to 8 ms:

Table 36-4 Example TIMEOUTB settings at different PCLK frequencies

fPCLK	TIMEOUTB [11: 0]	TEXTEN	tLOW: EXT
8 MHz	0x1F	1	32 x 2048 x 125 ns = 8 ms
16 MHz	0x3F	1	64 x 2048 x 62.5 ns = 8 ms
48 MHz	0xBB	1	188 x 2048 x 20.08 ns = 8 ms

- **Configure the maximum duration of tIDLE to 50 µ s:**

Table 36-5 Example TIMEOUTA settings at different PCLK frequencies (maximum tIDLE = 50 µ s)

fPCLK	TIMEOUTB [11: 0]	TIDLE	TIMEOUTEN	tTIDLE
8 MHz	0x63	1	1	100 x 4 x 125 ns = 50 µs
16 MHz	0xC7	1	1	64 x 4 x 62.5 ns = 50 µs
48 MHz	0x257	1	1	600 x 4 x 20.08 ns = 50 µs

Table 36-6 SMBus Timeout Specification

Symbol	Parameter	Limits		Unit
		Min	Max	
tTIMEOUT	Detect clock low timeout	25	35	ms
tLOW: SEXT (1)	Cumulative Clock Low Extension Time (Slave)	-	25	ms
tLOW: MEXT (2)	Cumulative Clock Low Extension Time (Master)	-	10	ms
tIDLE	SCL and SDA high timeout	-	50	µs

1. tLOW: SEXT is a cumulative period of time, i.e., a given period of time that the clock signal can extend from the initial start of a message to the stop of the device. Other slave devices or master devices may also extend the clock, which in turn causes the total extension of the clock low level to exceed tLOW: SEXT. Therefore, this device should be the only device addressed by the full-speed master device when measuring this parameter.
2. tLOW: MEXT is an accumulated period of time, that is, the time that the clock signal of the master device can extend within each byte of the message (defined as START to ACK, ACK to ACK, or ACK to STOP). The slave device or other master device may also extend the clock, which in turn

causes the total time of the clock low to exceed tLOW: MEXT (for a given byte). Therefore, the full-speed master only addresses one slave when measuring this parameter.

36.4 TIMx registers

The peripheral registers have to be accessed by half-words (16 bits) or words (32 bits).

36.4.1 I2C Control Register 1(I2C_CR1)

Address offset: 0x00

Reset value: 0x0008 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												WKUP_CNT		WKUP_DIV	
-												RW	RW	RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	WUPEN	SMBUS	PE
RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 18	WKUP_CNT	RW	2'h2	Count number of low-power wake-up timeouts 2 'b00: 2 2 'b01: 8 2 'b10: 32 2 'b11: 128 Note: timout time is calculated as (WKUP_DIV/FREQ) * WKUP_CNT us Example:WKUP_ DIV selects 11 as 1024, FREQ as 8M, CNT selects 00 as 2, then the timeout timeis(1024/8) * 2 = 256 μs Note: When PCLK is 72M, the timeout time is 3.5 μs-1817.6 μs At PCLK of 4M, the timeout time is 64 μs-32768 μs
17: 16	WKUP_DIV	RW	2'h0	Frequency division coefficient of PCLK, the divided clock is used to count the timeout time when STOP mode wakes up 2 'b00: 128 2 'b01: 256 2 'b10: 512 2 'b11: 1024
15	SWRST	RW	0	Software reset When set, the I2C is under reset state. Before resetting this bit, make sure the I2C lines are released and the bus is free. 0: I2C Peripheral not under reset 1: I2C Peripheral under reset state Note: This bit can be used to reinitialize the peripheral after an error or a locked state. As an example, if the BUSY bit is set and remains locked due to a glitch on the bus, the SWRST bit can be used to exit from this state.
14	Reserved	-	-	-
13	ALERT	RW	0	SMBus alert 0: Releases SMBALERT pin high. The reminder response address header follows the NACK signal; 1: Drives SMBALERT pin low. The reminder response address header immediately follows the ACK signal; When PE = 0, it is cleared by the hardware.
12	PEC	RW	0	Packet error checking This bit is set and cleared by software, and cleared by hardware when PEC is transferred or by a START or Stop condition or when PE=0. 0: No PEC transfer 1: PEC transfer (in Tx or Rx mode)

				Note: PEC calculation is corrupted by an arbitration loss.
11	POS	RW	0	ACK/PEC (for data reception). This bit is set and cleared by software and cleared by hardware when PE=0. 0: ACK bit controls the (N)ACK of the current byte being received in the shift register. The PEC bit indicates that current byte in shift register is a PEC. 1: ACK bit controls the (N)ACK of the next byte which will be received in the shift register. The PEC bit indicates that the next byte in the shift register is a PEC. Note: The POS bit is used when the procedure for reception of 2 bytes is followed. It must be configured before data reception starts. To NACK the 2nd byte, the ACK bit must be cleared just after ADDR is cleared. In order to detect the PEC of the 2nd byte, the PEC bit must be set at the ADDR extension event after the POS bit is configured.
10	ACK	RW	0	Acknowledge enable This register can be set/cleared by software, or cleared by hardware when PE = 0. 0: No acknowledge returned 1: Acknowledge returned after a byte is received. (when matching addresses or data)
9	STOP	RW	0	Stop generation. The bit is set and cleared by software, cleared by hardware when a Stop condition is detected, set by hardware when a timeout error is detected. In Master Mode: 0: No Stop generation. 1: Stop generation after the current byte transfer or after the current Start condition is sent. In slave mode: 0: No Stop generation. 1: Release the SCL and SDA lines after the current byte transfer.
8	START	RW	0	Start generation This bit is set and cleared by software and cleared by hardware when start is sent or PE=0. In Master mode: 0: No Start generation 1: Repeated start generation In Slave mode: 0: No Start generation 1: Start generation when the bus is free (automatically switching to Master mode by hardware)
7	NOSTRETCH	RW	0	Clock stretching disable (Slave mode) This bit is used to disable clock stretching in slave mode when ADDR or BTF flag is set, until it is reset by software. 0: Clock stretching enabled 1: Clock stretching disabled
6	ENGC	RW	0	General call enable 0: General call disabled. . Address 00h is NACKed. 1: General call enabled. Address 00h is ACKed.
5	ENPEC	RW	0	PEC enable 0: PEC calculation disabled 1: PEC calculation enabled
4	ENARP	RW	0	ARP enable 0: ARP disabled; 1: ARP enabled; SMBus device default address recognized if SMBTYPE=0; SMBus host address recognized if SMBTYPE=1.

3	SMBTYPE	RW	0	SMBus type. 0: SMBus device 1: SMBus host;
2	WUPEN	RW	0	Wake up from mode STOP mode enabled 0: Prohibit wake-up from mode STOP mode 1: Enable wake-up STOP mode from mode
1	SMBUS	RW	0	SMBus Mode Enable 0: I2C mode; 1: SMBus mode
0	PE	RW	0	I2C enable 0: Disabled 1: Enabled Note: If this bit is reset while a communication is on going, the peripheral is disabled at the end of the current communication, when back to IDLE state. All bit resets due to PE=0 occur at the end of the communication. In master mode, this bit must not be reset before the end of the communication.

36.4.2 I2C Control Register 2(I2C_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res -															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Res.	FREQ [6: 0]						
-			RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	LAST	RW	0	DMA last transfer 0: Next DMA EOT is not the last transfer 1: Next DMA EOT is the last transfer Note: This bit is used in master receiver mode to permit the generation of a NACK on the last received data.
11	DMAEN	RW	0	DMA Request Enable 0: Prohibit DMA request 1: Enable DMA request
10	ITBUFEN	RW	0	Buffer interrupt enable 0: TxE = 1 or RxNE = 1 does not generate any interrupt. 1: TxE = 1 or RxNE = 1 generates event interrupt (whatever the state of DMAEN)
9	ITEVTEN	RW	0	Event interrupt enable 0: Disabled 1: Event interrupt enabled This interrupt is generated when: ● SB = 1 (master mode); ● ADDR = 1 (Master/Slave) ● ADD10 = 1 (Master); ● STOPF = 1 (Slave) ● BTF = 1 with no TxE or RxNE event ● TxE event to 1 if ITBUFEN = 1 ● RxNE event to 1if ITBUFEN = 1
8	ITERREN	RW	0	Error interrupt enable 0: Error interrupt disabled; 1: Error interrupt enabled; This interrupt is generated when: ● BERR = 1 ● ARLO = 1 ● AF = 1 ● OVR = 1 ● PECERR = 1 ● TIMEOUT = 1;

				● SMBAlert = 1.
7	Reserved	-	-	-
6: 0	FREQ	RW	0	I2C clock frequency The FREQ bits must be configured with the APB clock frequency value (I2C peripheral connected to APB). The FREQ field is used by the peripheral to generate data setup and hold times compliant with the I2C specifications. The running clock is an even clock. The minimum allowable settable frequencies are 4 MHz and above in 100k mode, greater than 8 MHz in 400k mode, and greater than 16 MHz in 1MHz mode, respectively. 0000000: Disabled 0000001: Disabled ..... 0000011: Disabled 0000100: 4 MHz ….. ….. (See the maximum clock frequency of APB for the maximum frequency that can be configured)

36.4.3 I2C own address register 1(I2C_OAR1)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDMODE	Res.	Res.	Res.	Res.	Res.	ADD [9: 8]		ADD [7: 1]							ADD0
RW	-					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	ADDMODE	RW	0	Addressing mode (slave mode). 0: 7 bit slave address (does not respond to 10 bit addresses); 1: 10 bit slave address (does not respond to 7 bit addresses);
14	Reserved	-	0	-
13: 10	Reserved	-	0	-
9: 8	ADD [9: 8]	RW	2'h0	Interface address. 7-bit addressing mode: don't care 10-bit addressing mode: bits 9:8 of address
7: 1	ADD [7: 1]	RW	7'h0	Bits 7:1 of interface address
0	ADD0	RW	0	Interface address. The register is invalid in 7-bit address mode. Bit 0 of the address in 10-bit address mode.

36.4.4 I2C own address register 2 (I2C_OAR2)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	OA2MSK [2: 0]			ADD2 [7: 1]							ENDUAL
-					RW			RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 11	Reserved	-	-	-

10: 8	OA2MSK [2: 0]	RW	3'h0	Device own address 2 mask bits 000: Unshielded 001: OA2 [1] is masked as an irrelevant bit. Compare only OA2 [7: 2]. 010: OA2 [2: 1] is masked as an irrelevant bit. Compare only OA2 [7: 3]. 011: OA2 [3: 1] is masked as an irrelevant bit. Compare only OA2 [7: 4]. 100: OA2 [4: 1] is masked as an irrelevant bit. Compare only OA2 [7: 5]. 101: OA2 [5: 1] is masked as an irrelevant bit. Compare only OA2 [7: 6]. 110: OA2 [6: 1] is masked as an irrelevant bit. Only OA2 is compared [7]. 111: OA2 [7: 1] is masked as an irrelevant bit. No comparison is done, and all (except reserved) 7-bit received addresses are acknowledged.
7: 1	ADD2 [7: 1]	RW	7'h0	Bits 7:1 of interface address bits 7:1 of address in dual addressing mode
0	ENDUAL	RW	0	Dual addressing mode enable 0: In 7-bit address mode, only OAR1 is recognized; 1: In the 7-bit address mode, both OAR1 and OAR2 are recognized.

36.4.5 I2C Data Register(I2C_DR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR [7: 0]							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	DR [7: 0]	RW	8'h0	The 8-bit data register, actually two independent caches inside the chip share an address, which are used to store the received data (RX_DR) and place the data to be sent to the bus (TX_DR) respectively. **Transmitter mode:** Byte transmission starts automatically when a byte is written in the DR (TX_DR actually) register. A continuous transmit stream can be maintained if the next data to be transmitted is put in DR once the transmission is started (TxE=1). **Receiver mode:** Received byte is copied into DR (RX_DR actually) register (RxNE=1). A continuous transmit stream can be maintained if DR is read before the next data byte is received (RxNE=1). Notes: 1) In slave mode, the address is not copied into the data register 2) Write collision is not managed (DR can be written if TxE=0). 3) If an ARLO event occurs while processing an ACK pulse, the received byte will not be copied into the data register and therefore cannot be read

36.4.6 I2C status register 1(I2C_SR1)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMB	TIMEOUT	Res.	PEC	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB

ALERT			ERR												
RC_W0		-	RC_W0					R	R	-	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	SMBALERT	RC_W0	0	SMBus alert Host Mode: 0: No SMBus reminder; 1: Generate an SMBAlert reminder event on the pin; Slave mode: 0: No SMBAlert response address header sequence; 1: The SMBAlert response address header sequence is received until SMBAlert goes low. Cleared by software writing 0, or by hardware when PE=0.
14	TIMEOUT	RC_W0	0	Timeout or Tlow error. 0: No timeout error; 1: The duration of low SCL has reached 25ms (timeout); Or the host's low-level cumulative clock expansion time exceeds 10ms (Tlow: next); Or the low-level cumulative clock extension time of the slave device exceeds 25ms (Tlow: sext); Or Tidle's time timeout; Alternatively, when the I2Cmodewakes up fromthe STOP mode, the timeout time can be set by setting I2C_CR1. When set in slave mode: slave resets the communication and lines are released by hardware; When set in master mode: Stop condition sent by hardware; Cleared by software writing 0, or by hardware when PE=0. Note: This feature is only useful during the wake-up process from STOP mode in SMBUS mode and I2C mode.
13	Reserved	-	-	-
12	PECERR	RC_W0	0	PEC Error in reception 0: no PEC error: receiver returns ACK after PEC reception (if ACK=1) 1: PEC error: receiver returns NACK after PEC reception (whatever ACK) This bit is cleared by writing software to 0, or by hardware when PE = 0.
11	OVR	RC_W0	0	Overrun/Underrun 0: No overrun/underrun 1: Overrun or underrun Set by hardware in slave mode when NOSTRETCH=1 and: Cleared by software writing 0, or by hardware when PE=0. Note: If the DR write occurs very close to SCL rising edge, the sent data is unspecified and a hold timing error occurs.
10	AF	RC_W0	0	Acknowledge failure 0: No acknowledge failure 1: Acknowledge failure Set by hardware when no acknowledge is returned. Cleared by software writing 0, or by hardware when PE=0.
9	ARLO	RC_W0	0	Arbitration lost (master mode) 0: No Arbitration Lost detected 1: Arbitration Lost detected Set by hardware when the interface loses the arbitration of the bus to another master. Cleared by software writing 0, or by hardware when PE=0. After an ARLO event the interface switches back automatically to Slave mode (MSL=0). Note: In SMBUS mode, arbitration in slave mode only occurs in data interpretation, or response transmission interval (excluding address response).
8	BERR	RC_W0	0	Bus error 0: No misplaced Start or Stop condition 1: Misplaced Start or Stop condition Set by hardware when the interface detects wrong start or end conditions during a byte transfer. Cleared by software writing 0, or by hardware when PE=0.
7	TxE	R	0	Data register empty (transmitters) 0: Data register not empty

				1: Data register empty Set when DR is empty in transmission. TxE is not set during address phase. Cleared by software writing to the DR register or by hardware after a start or a stop condition or when PE=0. If a NACK is received, or the next byte to be sent is PEC (PEC = 1), this bit is not set. Note: TxE is not cleared by writing the first data being transmitted, or by writing data when BTF is set, as in both cases the data register is still empty.
6	RxNE	R	0	Data register not empty (receivers) 0: Data register empty 1: Data register not empty Set when data register is not empty in receiver mode. RxNE is not set during address phase. Cleared by software reading or writing the DR register or by hardware when PE=0. Note: RxNE is not cleared by reading data when BTF is set, as the data register is still full.
5	Reserved	-	-	-
4	STOPF	R	0	Stop detection (slave mode) 0: No Stop condition detected 1: Stop condition detected Set by hardware when a Stop condition is detected on the bus by the slave after an acknowledge (if ACK=1). When software reads the I2C_SR1 register, a write operation to the I2C_CR1 register will clear this bit, or hardware will clear this bit when PE = 0. Note: The STOPF bit is not set after a NACK reception.
3	ADD10	R	0	10-bit header sent (Master mode). 0: No ADD10 event occurred. 1: Master has sent first address byte. In 10-bit address mode, when the master device sends the first byte out, the hardware puts the position 1. Cleared by software reading the I2C_SR1 register followed by a write in the I2C_DR register, or by hardware when PE=0. Main: After receiving NACK, this register is not set.
2	BTF	R	0	The byte transfer sends end flag bit. 0: Data byte transfer not done 1: Byte transfer transmission ended successfully When NOSTRETCH = 0, the hardware sets this register in the following situations (slave mode, when NOSTRETCH = 0; master mode, independent of NOSTRETCH): — In reception when a new byte is received (including ACK pulse) and DR has not been read yet (RxNE=1). — In transmission when a new byte should be sent and DR has not been written yet (TxE=1). Cleared by software reading I2C_SR1 followed by either a read or write in the DR register or by hardware after a start or a stop condition in transmission or when PE=0. Notes: The BTF bit is not set after a NACK reception. If the next byte to be transferred is PEC (I2C_SR1.TRA = 1, I2C_CR1.PEC = 1), the BTF bit is not set.
1	ADDR	R	0	Address sent (master mode)/matched (slave mode) After readingthe I2C_SR2 register, reading the I2C_SR1 register will clear this bit; Either after sending a start or stop condition in transmission, or when PE = 0, it is cleared by hardware. Address matched (slave): 0: Address mismatched or not received. 1: Received address matched. The hardware sets this bit when the received slave address matches the OAR register or broadcast call address, or the SMBus device default address, or the SMBus host recognizes the SMBus reminder address. Note: In slave mode, it is recommended to perform a complete zeroing sequence, that is, after ADDR is set, read the SR1 register first, and then read the SR2 register. Address sent (Master): 0: No end of address transmission

				1: End of address transmission — 10-bit address, when the ACK post-bit of the second byte of the address is received; — For 7-bit addressing, the bit is set after the ACK of the byte. Note: ADDR is not set after a NACK reception.
0	SB	R	0	Start bit (Master mode) 0: No Start condition 1: Start condition generated. —Set when a Start condition generated. —Cleared by software by reading the SR1 register followed by writing the DR register, or by hardware when PE=0

36.4.7 I2C status register2 (I2C_SR2)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res. -															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC [7: 0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Res.	TRA	BUSY	MSL
R	R	R	R	R	R	R	R	R	R	R	R	-	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 8	PEC	R	8'h0	Packet error checking register. This register contains the internal PEC when ENPEC=1.
7	DUALF	R	0	Dual address flag (slave mode). 0: Received address matched with OAR1; 1: Received address matched with OAR2. Cleared by hardware after a Stop condition or repeated Start condition, or when PE=0.
6	SMBHOST	R	0	SMBus master header sequence (slave mode) flag received 0: No SMBus Host address; 1: SMBus host address received when SMBTYPE=1 and ENARP=1. Cleared by hardware after a Stop condition or repeated Start condition, or when PE=0.
5	SMBDEFAULT	R	0	SMBus slave device default address (slave mode) 0: No SMBus device default address; 1: SMBus device default address received when ENARP=1. Cleared by hardware after a Stop condition or repeated Start condition, or when PE=0.
4	GENCALL	R	0	Broadcast call address (slave mode) 0: No General Call; 1: General Call Address received when ENGC=1. Cleared by hardware after a Stop condition or repeated Start condition, or when PE=0.
3	Reserved	-	-	-
2	TRA	R	0	Send/receive flags 0: Data bytes received 1: Data bytes transmitted This bit is set depending on the R/W bit of the address byte, at the end of total address phase. It is also cleared by hardware after detection of Stop condition (STOPF=1), repeated Start condition, loss of bus arbitration (ARLO=1), or when PE=0.
1	BUSY	R	0	Bus busy flag 0: No communication on the bus 1: Communication ongoing on the bus Set by hardware on detection of SDA or SCL low Cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).
0	MSL	R	0	Master-slave mode flag 0: Slave mode 1: Host mode

				- Set by hardware as soon as the interface is in Master mode (SB=1). - Cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1), or by hardware when PE=0.

36.4.8 I2C Clock control register (I2C_CCR)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	F +	Res.	CCR [11: 0]											
RW	RW	RW	-	RW											

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	F/S	RW	0	I2C master mode selection 0: Sm mode I2C 1: Fm mode I2C
14	DUTY	RW	0	Fm mode duty cycle 0: Fm mode tlow/thigh = 2 1: Fm mode tlow/thigh = 16/9
13	F +	RW	0	I2C Rapid Enhancement Mode Selection. 0: Standard mode or fast mode, which one to choose is determined by bit15; 1: Fast mode plus;
12	Reserved	-	-	-
11/0	CCR [11: 0]	RW	12'h0	Clock control register in Fm/Sm mode (Master mode) Controls the SCL clock in master mode. ● Standard mode or SMBus mode: Thigh=CCR x Tpclk Tlow =CCR x Tpclk ● Fast mode: ✓ DUTY=0: Thigh=CCR x Tpclk Tlow = 2 x CCR x Tpclk ✓ DUTY = 1 (up to 400 KHz): Thigh = 9 x CCR x Tpclk Tlow = 16 x CCR x Tpclk ● Quick Plus Mode: ✓ DUTY=0: Thigh = 3 x CCR x Tpclk Tlow = 5 x CCR x Tpclk ✓ DUTY=1: Thigh = 2 x CCR x Tpclk Tlow = 3x CCR x Tpclk Notes: 1. The minimum allowed setting is 0x04, and the minimum allowed in fast mode is 0x01 2. Thigh=tr(SCL)+tw(SCLH) 3. Tlow=tr(SCL)+tw(SCLL) 4. These delays have no filters 5. This register can only be configured when PE = 0;

36.4.9 I2Crise time register(I2C_TRISE)

Address offset: 0x20

Reset value: 0x0000 0082

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	THOLDDATA_SEL	THOLDDATA					TRISE						
-			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	-
12	THOLDDATA_SEL	RW	0	Data retention time selection 0: Default hardware calculation data retention time 1: Configure data retention time through THLDDATA
11: 7	THOLDDATA	RW	5'h1	Maximum data holding time in fast/standard/fast enhance-ment mode (transmission mode) These bits are used for the minimum time to guarantee the data holding time in the data transmission mode. For example: The maximum SDA drop time allowed for standard mode is 300ns. If the value of I2C_CR2.FREQ [6: 0] is equal to 0x08 and Tpclk = 125ns, 0x03 is configured in TRISE (300ns/125ns = 2.4 +1 = 3.4) If the result is not an integer, the integer part is written to THOLDDATA to ensure configuration.
6: 0	TRISE	RW	7'h2	Maximum rise time in Fm/Sm mode (Master mode) These bits should provide the maximum duration of the SCL feedback loop in master mode. The purpose is to keep a sta-ble SCL frequency whatever the SCL rising edge duration. These bits must be programmed with the maximum SCL rise time given in the I2C bus specification, incremented by 1. For instance: in Sm mode, the maximum allowed SCL rise time is 1000 ns. If the value of I2C_CR2.FREQ [5: 0] is equal to 0x08 and Tpclk = 125ns, 0x09 is configured in TRISE (1000ns/125ns = 8 + 1 = 9). If the result is not an integer, TRISE[5:0] must be pro-grammed with the integer part, in order to respect the tHIGH parameter. Note: TRISE[5:0] must be configured only when the I2C is disabled (PE = 0).

36.4.10 I2CTimeout Register(I2C_TIMEOUTR)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEXTEN	Res.	Res.	Res.	TIMEOUTB [11: 0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIMOUTEN	Res.	Res.	TIDLE	TIMEOUTA [11: 0]											
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	TEXTEN	RW	0	Clock signal extension timeout enable 0: Disable clock signal extension timeout detection. 1: Enable clock signal extension timeout detection A TIMEOUT error (TIMEOUT = 1) will be detected when the cumulative time for the I2C interface to perform the SCL extension exceeds tLOW: EXT.
30: 28	Reserved	-	-	-
27: 16	TIMEOUTB [11: 0]	RW	12'h0	Bus timeout B This field is used to configure the cumulative clock exten-sion timeout: In the master mode, the cumulative clock low level exten-sion time (tLOW: MEXT) of the master device will be de-tected In the slave mode, the cumulative clock low level extension time (tLOW: SEXT) of the slave device will be detected. tLOW: EXT = (TIMEOUTB+1) x 2048 x tPCLK Note: These bits can only be written when TEXTEN = 0
15	TIMEOUTEN	RW	0	Clock timeout enable 0: Disable SCL timeout detection. 1: Enable SCL timeout detection: when the low level time of SCL exceeds tTIMEOUT (TIDLE = 0), or the high level of SCL

				When the time exceeds TIDLE (TIDLE = 1), a TIMEOUT error (TIMEOUT = 1) is detected.
14: 13	Reserved	RES	-	-
12	TIDLE	RW	0	Idle clock timeout detection 0: TIMEOUTA is used to detect SCL low timeout 1: TIMEOUTA is used to detect SCL and SDA high timeouts (bus idle conditions) Note: This bit can only be written when TIMOUTEN = 0.
11/0	TIMEOUTA [11: 0]	RW	12'h0	Bus timeout A This field is used to configure: -SCL low timeout condition tTIMEOUT (when TIDLE = 0) tTIMEOUT = (TIMEOUTA+1) x 2048 x tPCLK -Bus idle condition, i.e. SCL and SDA high (when TIDLE = 1) tIDLE = (TIMEOUTA+1) x 4 x tPCLK Note: These bits can only be written when TIMOUTEN = 0.

37. Universal Synchronous Transceiver (USART)

37.1 Introduction

Universal Synchronous Asynchronous Transceiver (USART) can flexibly exchange full-duplex data with external devices, meeting the requirements of external devices for industrial standard NRZ asynchronous serial data format. The USART utilizes a fractional baud rate generator to provide a wide range of baud rate options.

Supports synchronous unidirectional communication and half-duplex single-wire communication, and also supports LIN (local Internet), smart card protocol and IrDA (infrared data organization) specifications, as well as modem (CTS/RTS) operation. It also supports multiprocessor communications.

High-speed data communication can be achieved by using the DMA method of the multi-buffer configuration.

37.2 Main Features

- Full-duplex asynchronous communication
- Configurable 16 or 8 oversampling methods to achieve the best choice between speed and clock tolerance
- Two built-in fifos for sending and receiving
- Both FIFO can be enable and provide status bits by software
- Dual clock domain: A USART_CLK clock that exists in addition to PCLK
- Automatic baud rate detection function
- Configurable data word length (7-bit, 8-bit or 9-bit)
- Configurable data transmission direction: MSB, LSB
- Configurable stop bits-0.5, 1, 1.5 or 2 stop bits are supported
- Provides input and output function in synchronous mode
- Configurable multi-buffer communication using DMA (Direct Memory Access)
- Single-wire Half-duplex communications
- Separate enable bits for transmitter and receiver
- Configurable TX/RX pin signal
- Supported hardware flow control: modem and RS485
- Communication control and error detection signal
- Parity control

  - Send check digit
  - Check the received data
- Multiprocessor communication. enter into mute mode if address match does not occur.

  -Wake up in silent mode (by idle bus detection or address flag detection)

  -Wake-up mode of receiver: address bit (MSB, bit 9), bus idle

37.2.1 USART Extended Functions

- Ability of LIN master to send synchronous disconnected frames and ability of LIN slave to detect disconnected frames

  -Ability to generate 13-bit broken frames and detect broken frames when USART hardware is configured as LIN
- IRDA SIR encoder/decoder
- Smart card emulation function

  -Smart card interface supports asynchronous smart card protocol defined in ISO7816-3 standard

  -Smart card uses 0.5 and 1.5 stop bits
- MODBUS Communications

  -TIMOUT function

  -CR/LF character recognition function

37.2.2 USART Functional Block Diagram

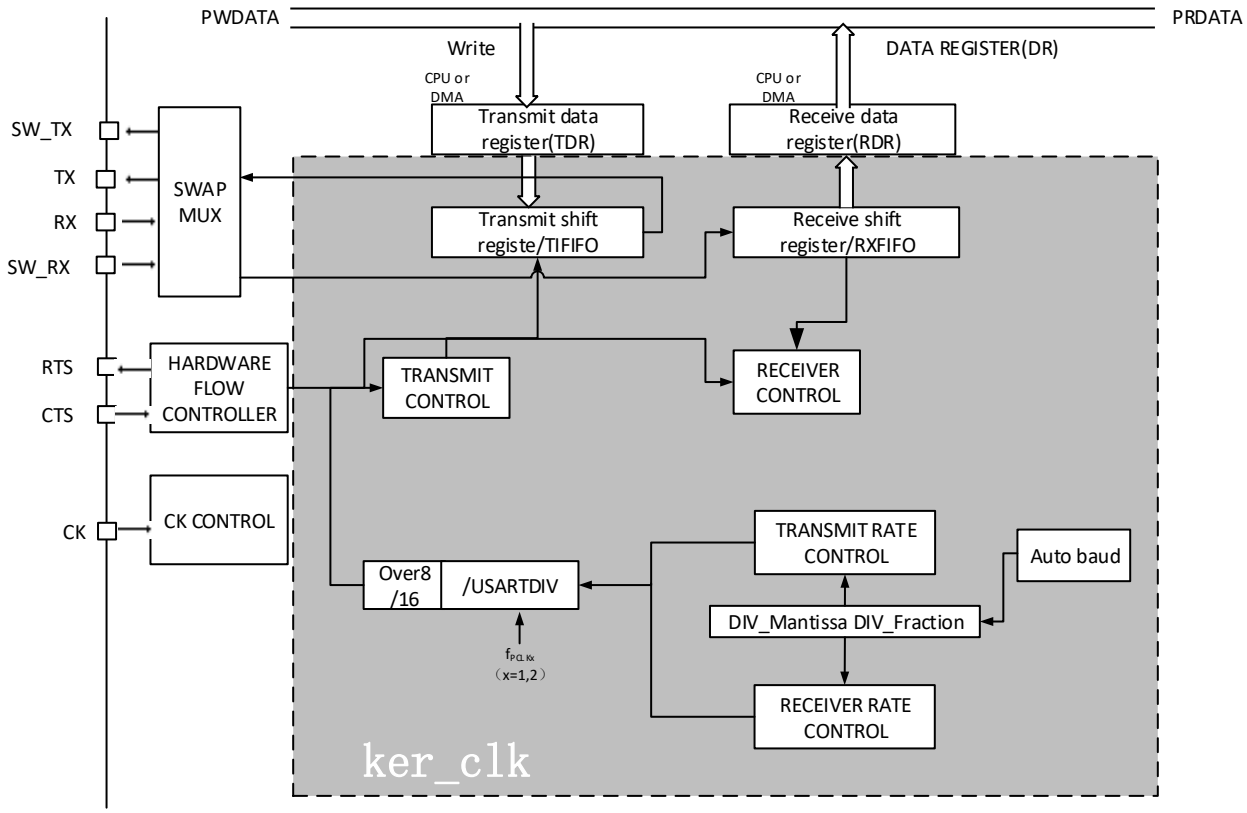


Figure 37-1 USART block diagram

USART has two clock domains, one is PCLK and the other is USART_CLK: There is no requirement for the speed of USART_CLK and PCLK. The only requirement is that the software manages communication as fast as possible.

An idle symbol is regarded as a complete data frame consisting entirely of '1', followed by the start bit of the next frame containing the data (the number of bits of '1' also includes the number of bits of the stop bit).

A disconnected frame is considered to have received '0' throughout one frame period (including during the stop bit, also '0'). At the end of the disconnect frame, the transmitter inserts 2 more stop bits ('1') in response to the start bit.

Transmission and reception are driven by a common baud rate generator, which generates clocks for the transmitter and receiver when their enable bits are set respectively.

37.3 USART functional description

37.3.1 Transfer pin

37.3.1.1 USART Features

Any USART bi-directional communication requires at least two pins: Receive Data Input (RX) and Send Data Output (TX).

RX: Receive data serial input. Data is recovered by oversampling techniques to distinguish data from noise.

TX: Transmit data output. When the transmitter is disabled, the output pin returns to its IO port configuration. When the transmitter is enabled but not sending data, the TX pin is high. In single-wire mode and smart card mode, this I/O port is used for both data transmission and reception.

37.3.1.2 USART Hardware Flow Control

RS232 hardware flow control

CTS: Transmission starts at low level, and transmission ends to high level.

RTS: Low indicates that USART is ready to receive data.

RS485 hardware flow control

DE: This signal is used to control the transmission of an external receiver

37.3.1.3 Sync mode and smart card mode

In synchronous master mode and smart card mode, the following pins are required:

CK: Transmitter clock output, in synchronous master mode, this pin outputs the transmitter data clock for synchronous transmission corresponding to SPI master mode (no clock pulses on the start and stop bits, and a software option to send clock pulses on the last data bit). In synchronous mode, data can be received synchronously on the RX pin. This mechanism can be used to control peripherals with shift registers (e.g., LCD drivers). The clock phase and polarity are software programmable.

37.3.2 USART Feature Description

The transmitter sends 7-bit, 8-bit, or 9-bit data depending on the status of the M-bit. When the transmit enable bit (TE) is set, the data in the transmit shift register is output on the TX pin and the corresponding clock pulse is output on the CK pin.

Notes:

1. In 7-bit data mode, smart card mode, LIN host mode is not supported.
2. The TE bit should not be reset during transmission of data. Resetting the TE bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.
3. An idle frame will be sent after the TE bit is enabled.
4. Every character is preceded by a start bit, which is a logic level low for one bit period. The character is terminated by a configurable number of stop bits.

5. The following stop bits are supported by USART: 0.5, 1, 1.5 and 2 stop bits.

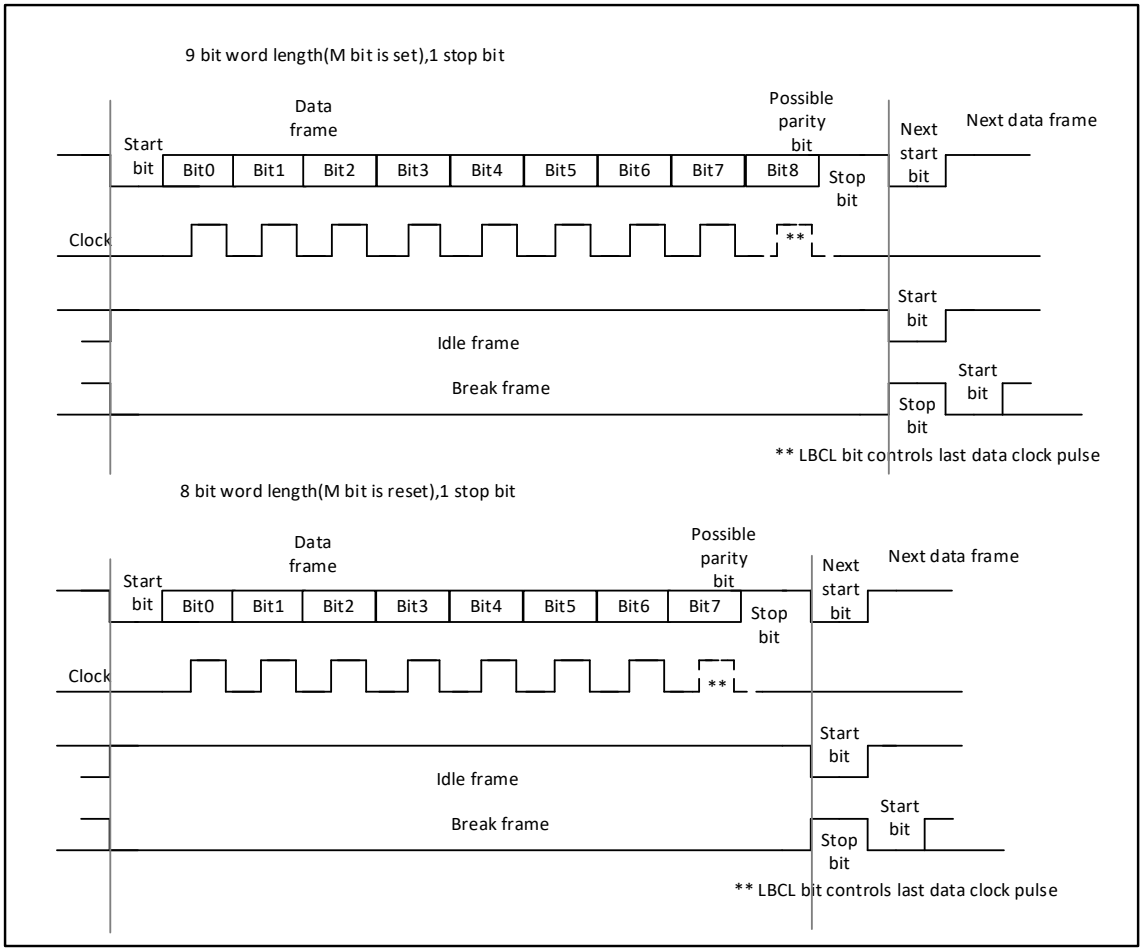


Figure 37-2 word length programming

37.3.3 USART FIFO And threshold

USART can work in FIFO mode. USART has a transmit FIFO (TXFIFO) and a receive FIFO (RXFIFO). Enable FIFO mode by setting USART_CR1. FIFOEN. This mode only supports UART, Sync Mode (Send) and Smart Card Mode TX.

Since the maximum data word length is 9 bits, the width of the TXFIFO is 9 bits. However, the default width of RXFIFO is 12 bits. This is because the receiver not only stores data in the FIFO, but also stores error flags (parity error, noise error, and frame error flags) associated with each character.

The received data is stored in the RXFIFO together with the corresponding flags. However, when reading the DR, only data is read. The status flag is available in the USART_SR register. You can configure the TXFIFO and RXFIFO levels that trigger TX and RX interrupts. These thresholds are programmed by the RXFTCFG and TXFTCFG bits in USART_CR1.

In this case, when the amount of data received by the RXFIFO reaches the threshold programmed in the RXFTCFG bit field, the RXFT flag is set in the USART_CR3 register, generating the corresponding interrupt (if enabled).

When the number of empty positions in the TXFIFO reaches the threshold programmed in the TXFTCFG bit field, the TXFT flag is set in the USART_CR3 register, generating the corresponding interrupt (if enabled).

Enable FIFO mode When data is received, the RXFNE bit is used to indicate that the RXFIFO is not empty. Read USART_DR Returns the oldest data entered in the RXFIFO. When data is received, it is stored in the RXFIFO along with the corresponding error bit.

• In multiprocessor communication mode: When FIFO mode is disabled, the RXNE flag is set after each byte is received. When DMA reads the receive data register, it is cleared. When FIFO mode is enabled, the RXFNE flag is set when RXFIFO is not empty. After each DMA request, 1 data is received from the RXFIFO. A DMA request is triggered when the RXFIFO is not empty (there is data to read from the RXFIFO).

In single buffer mode:

When FIFO mode is disabled, clearing the RXNE flag is done by performing a software read from the USART_DR register. The RXNE flag may also be cleared by setting the RXFRQ position at "1" at USART_CR1. The RXNE flag must be cleared before the end of receiving the next character to avoid overflow errors.

When FIFO mode is enabled, RXFNE is set when RXFIFO is not empty. After each read operation from USART_DR, a piece of data is retrieved from RXFIFO. When RXFIFO is empty, the RXFNE flag is cleared. When the RXFIFO is full, the first data in the RXFIFO must be read before the end of receiving the next character to avoid overflow errors. If the RXFNEIE bit is set, the RXFNE flag generates an interrupt.

37.3.4 Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the transmit enable bit (TE) is set, the data in the transmit shift register is output on the TX pin and the corresponding clock pulse is output on the CK pin.

37.3.4.1 Character transmission

During a USART transmission, data shifts out the least significant bit first on the TX pin. In this mode, the USART_DR register consists of a buffer (TDR) between the internal bus and the transmit shift register.

Every character is preceded by a start bit, which is a logic level low for one bit period. The character is terminated by a configurable number of stop bits. The following stop bits are supported by USART: 1 and 2 stop bits.

Notes:

The TE bit should not be reset during transmission of data. Resetting the TE bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.

An idle frame will be sent after the TE bit is enabled.

37.3.4.2 Configure stop bits

The number of stop bits to be transmitted with every character can be programmed in Control register 2, bits 13, 12.

1) 1 stop bit: The default value for the number of stop bits.
2) 2 stop bits: Can be used in regular USART mode, single wire mode, as well as modem mode.

An idle frame transmission will include the stop bits.

The break frame is a 10-bit low followed by a stop bit (when m = 0); Or an 11-bit low followed by a stop bit (when m = 1). It is not possible to transmit longer broken frames (longer than 10 or 11 bits in length).

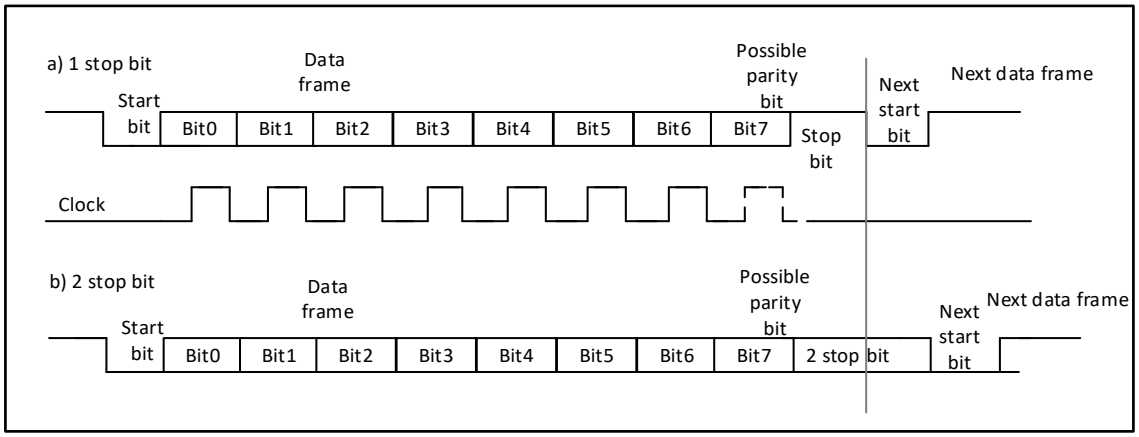


Figure 37-3 Configurable Stop Bits

Procedure:

1) Enable the USART by writing the UE bit in USART_CR1 register to 1.

2) Program the M bit in USART_CR1 to define the word length.

3)Programthe number of bits of the stop bitin USART_CR2.

4)If multi-buffer communication is used, configurethe DMA enable bit(DMAT)inUSART_CR3. Configure the DMA register as explained in multibuffer communication.

5) Select the desired baud rate using the USART_BRR register.

6) Set the TE bit in USART_CR1 to send an idle frame as first transmission.

7)Write the data to be sent intothe USART_DR register(this action clearsthe TXE bit). Repeat this for each data to be transmitted in case of single buffer.

8) After writing the last data into the USART_DR register, wait until TC=1. This indicates that the transmission of the last frame is complete. This is required for instance when the USART is disabled or enters the Halt mode to avoid corrupting the last transmission.

37.3.4.3 Single-byte communication

The TXE bit is always cleared by a write to the data register. The TXE bit is set by hardware and it indicates:

- The data has been moved from TDR to the shift register and the data transmission has started.
- The TDR register is empty.
- The next data can be written in the USART_DR register without overwriting the previous data.

This flag generates an interrupt if the TXEIE bit is set.

When a transmission is taking place, a write instruction to the USART_DR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the USART_DR register places the data directly in the shift register, the data transmission starts, and the TXE bit is immediately set.

If a frame is transmitted (after the stop bit) and the TXE bit is set, the TC bit goes high. An interrupt is generated if the TCIE bit is set in the USART_CR1 register.

Afterthe last data word has been written inthe USART_DR register, you must wait forTC = 1before turning off the USART moduleor setting the microcontroller into low power mode(see the figure below for details).

The TC bit is cleared by the following software sequence:

1. A read from the USART_SR register;
2. A write to the USART_DR register.

Note: The TC bit is cleared by writing 0 to it. This clearing sequence is recommended only for Multi-buffer communication.

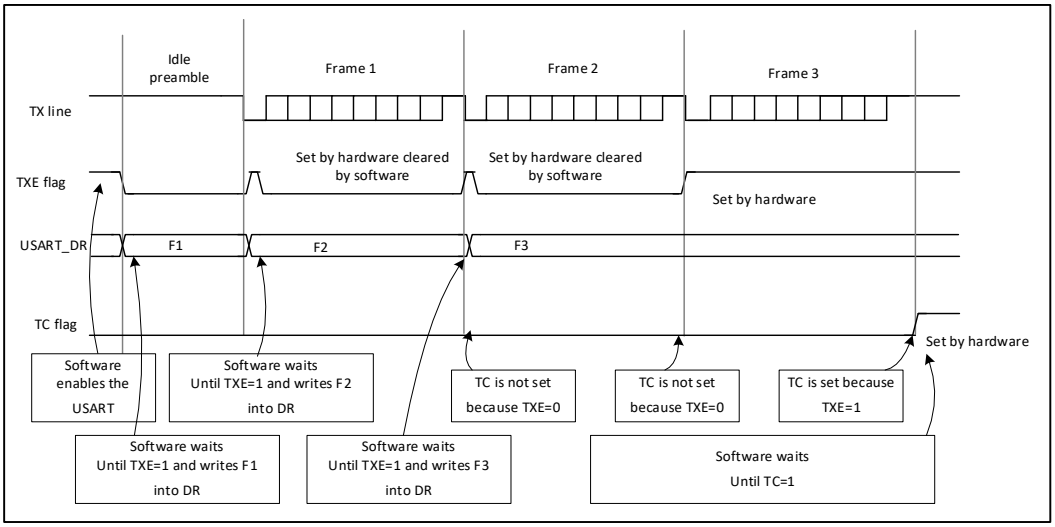


Figure 37-4 TC/TXE behavior when transmitting

37.3.4.4 Break characters

Setting the SBK bit transmits a break character. The break frame length depends on the Mbit. If the SBK bit is set to ‘1’ a break character is sent on the TX line after completing the current character transmission. The SBK is reset by hardware upon completion of the disconnect character transmission (at the disconnect of the stop bit of the symbol). The USART inserts a logic 1 bit at the end of the last break frame to guarantee the recognition of the start bit of the next frame.

Note: If the software resets the SBK bit before the commencement of break transmission, the break character will not be transmitted. For two consecutive breaks, the SBK bit should beset after the stop bit of the previous break.

37.3.4.5 Idle characters

Setting the TE bit drives the USART to send an idle frame before the first data frame.

37.3.5 Receiver

The USART can receive data words of either 8 or 9 bits depending on the M bit in the USART_CR1 register.

37.3.5.1 Start bit detection

In the USART, the start bit is detected when a specific sequence of samples is recognized. The sequence is 1 1 1 0 X 0 X 0 X 0 0 0

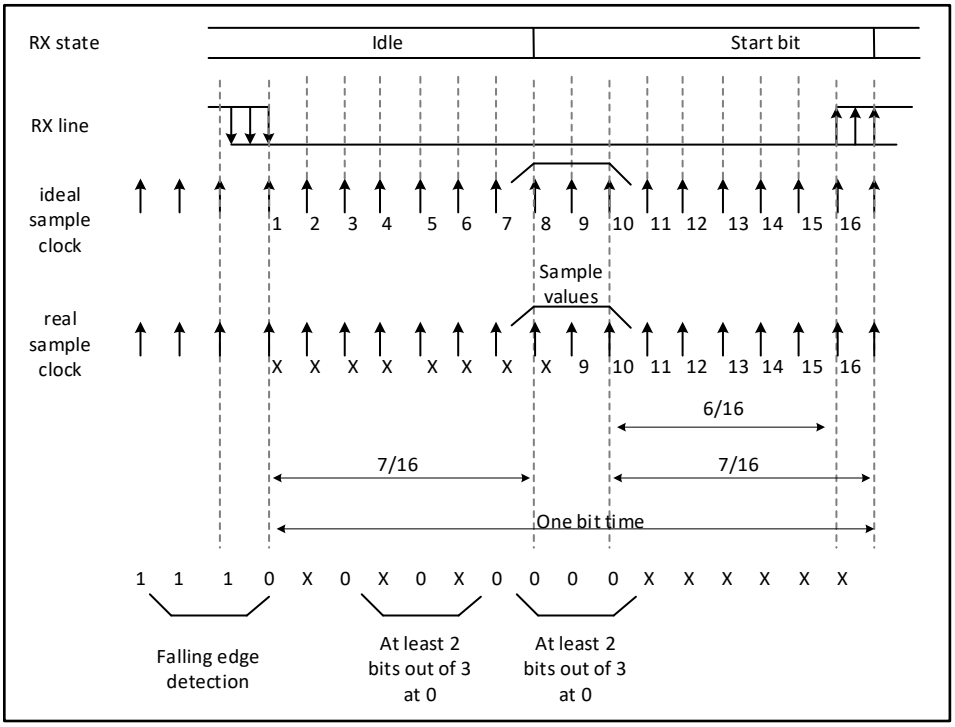


Figure 37-5 Start bit detection

If the sequence is incomplete, the receiving end will exit the start bit detection and return to the idle state (without setting the flag bit) waiting for the falling edge. The start bit is confirmed (RXNE flag set, interrupt generated if RXNEIE=1) if the 3 sampled bits are at 0 (first sampling on the 3rd, 5th and 7th bits finds the 3 bits at 0 and second sampling on the 8th, 9th and 10th bits also finds the 3 bits at 0).

The start bit is validated (RXNE flag set, interrupt generated if RXNEIE=1) but the NE noise flag is set if, for both samplings, at least 2 out of the 3 sampled bits are at 0 (sampling on the 3rd, 5th and 7th bits and sampling on the 8th, 9th and 10th bits). If this condition cannot be met, the start bit detection process is aborted, and the receiver will return to the idle state (no flag bit is set).

If, for one of the samplings (sampling on the 3rd, 5th and 7th bits or sampling on the 8th, 9thand 10th bits), 2 out of the 3 bits are found at 0, the start bit is validated but the NE noise flag bit is set.

37.3.5.2 Character reception

During a USART reception, data shifts in least significant bit first through the RX pin. In this mode, the USART_DR register consists of a buffer (RDR) between the internal bus and the received shift register.

Procedure:

1. The UEof the USART_CR1register isset to1to activateUSART.
2. Programmingthe M-bitdefinitionword length of USART_CR1
3. Write the numberof stop bits inUSART_CR2
4. If multi-buffer communication is required, selectthe DMA enable bit(DMAR)inUSART_CR3. Configure the DMA register as explained in multibuffer communication.
5. The desired baud rateis selected using the baud rate registerUSART_BRR.
6. Setsthe RE bitofUSART_CR1. This enables the receiver which begins searching for a start bit.

When a character is received:

- The RXNE bit is set. It indicates that the content of the shift register is transferred to the RDR. In other words, data has been received and can be read (as well as its associated error flags).
- An interrupt is generated if the RXNEIE bit is set.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.
- In multibuffer, RXNE is set after every byte received and is cleared by the DMA read tothe Data register.
- In single buffer mode, clearing the RXNE bit is performed by a software read to the USART_DR register. The RXNE flag can also be cleared by writing a zero to it. The RXNE bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Note: The RE bit should not be reset while receiving data. If the RE bit is disabled during reception, the reception of the current byte will be aborted.

37.3.5.3 Break characters

When a break character is received, the USART handles it as a framing error

37.3.5.4 Idle characters

When an idle frame is detected, there is the same procedure as a data received character plus an interrupt if the IDLEIE bit is set.

37.3.5.5 Overrun error

An overrun error occurs when a character is received when RXNE has not been reset. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after every byte received. An overrun error occurs if RXNE flag is set when the next data is received or the previous DMA request has not been serviced.

When an overrun error occurs:

- The ORE bit is set.
- The RDR content will not be lost. The previous data is available when a read to USART_DR is performed.
- The shift register will be overwritten. After that point, any data received during overrun is lost.
- An interrupt is generated if either the RXNEIE bit is set or both the EIE and DMAR bits are set.
- The ORE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

Note: The ORE bit, when set, indicates that at least 1 data has been lost. There are two possibilities:

- If RXNE=1, then the last valid data is stored in the receive register RDR and can be read.
- If RXNE=0, then it means that the last valid data has already been read and thus there is nothing to be read in the RDR. This may occur when new (i.e. missing) data is received while the last valid data is read in the RDR. This may also occurwhen new data is received during a read sequence (betweena USART_SRregister read access anda USART_DRread access).

37.3.5.6 Noise error

Over-sampling techniques are used (except in synchronous mode) for data recovery by discriminating between valid incoming data and noise.

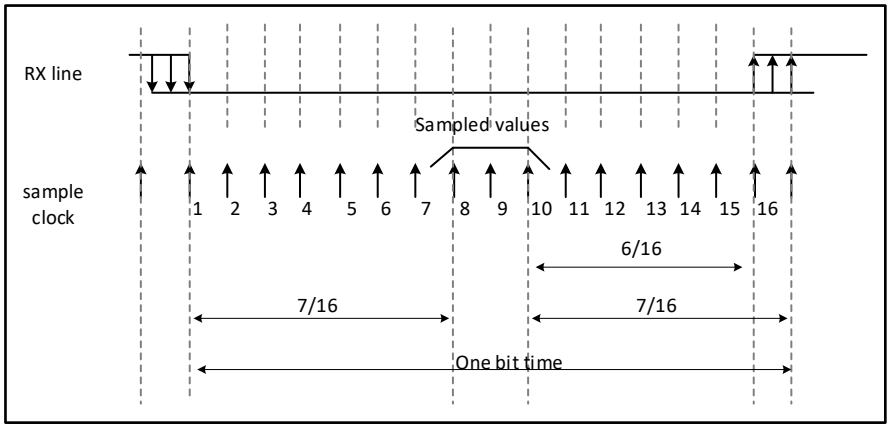


Figure 37-6 Data sampling for noise detection

Table 37-1 Noise detection from sampled dat

Sampled value	NE status	Received bit value	Data validity
000	0	0	Valid
001	1	0	Not Valid
010	1	0	Not Valid
011	1	1	Not Valid
100	1	0	Not Valid
101	1	1	Not Valid
110	1	1	Not Valid
111	0	1	Valid

When noise is detected in a frame:

- The NE is set at the rising edge of the RXNE bit.
- The invalid data is transferred from the Shift register to the USART_DR register.
- No interrupt is generated in case of single byte communication. However this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt will be issued if the EIE bit is set in the USART_CR3 register.
- The NE bit is reset by a USART_SR register read operation followed by a USART_DR register read operation.

37.3.5.7 Framing error

A framing error is detected when:

- The stop bit is not recognized on reception at the expected time, following either a desynchronization or excessive noise.
- When the framing error is detected:
- The FE bit is set by hardware
- The invalid data is transferred from the Shift register to the USART_DR register.
- No interrupt is generated in case of single byte communication. However, this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt is issued if the EIE bit is set in the USART_CR3 register.
- The FE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

37.3.5.8 Configure stop bits on receive

Configurable stop bits during reception The number of received stop bits may be configured by the control bits of the control register 2, and may be 1 or 2 in the normal mode.

- 1 stop bit: sampling for 1 stop bit is done on the 8th, 9th and 10th samples.
- 2 stop bits: sampling for 2 stop bits is done on the 8th, 9th and 10th samples of the first stop bit. The framing error flag is set if a framing error is detected during the first stop bit. The second stop bit is not checked for framing error. The RXNE flag is set at the end of the first stop bit.

Special frames and error handling:

Broken character: When a broken frame is received, USART treats it as an error frame.

Idle character: When an idle frame is detected, it is processed as a normal frame, and if IDLEIE = 1, an interrupt is generated.

Select the appropriate clock domain to:

The selection of clock source is controlled by RCC. Before enabling USART, the clock source must be selected.

The selection of clock source only takes into account two factors:

1. Do you need usart to be used in low power mode
2. Communication speed

You can configure the usart_clk clock source in the RCC to achieve dual clocking. Otherwise, the usart_clk clock is the same as the usart_pclk clock.

Some USART_clk sources enable USART to receive data while the MCU is in low power mode. Depending on the received data and the selected wake-up mode, the USART wakes up the MCU when needed to transmit the received data by performing a software read on the USART_DR register or by DMA.

For other clock sources, the system must be operational to enable USART communication. The receiver implements different user-configurable oversampling techniques (except for synchronous mode) for data recovery by distinguishing incoming data from noise. This allows for a balance between maximum communication speed and noise/clock error immunity.

Noise error:

Data recovery is performed using oversampling techniques (except in synchronous mode) to distinguish valid input data from noise.

37.3.6 Fractional baud rate generation

The baud rates of the receiver and transmitter should be set to the same values in the integer and decimal registers of USARTDIV.

1: 16-bit oversampling mode

Tx/Rx baud rate = $\frac{fck}{(16*USARTDIV)}$

Here the fCK is the clock given to the peripheral.

Note: The baud counters are updated with the new value of the Baud registers after a write to USART_BRR. Hence, the Baud rate register value should not be changed during communication.

Table 37-2 Error calculation when setting baud rate

Baud rate		fck = 8MHZ			fck = 36MHZ			fck = 100MHZ		
No.	**kbps**	**Actual**	**Value programmed in the baud rate register**	**Error %**	**Actual**	**Value programmed in the baud rate register**	**Error%**	**Actual**	**Value programmed in the baud rate register**	**Error %**
1	2.4	2.400	208.375	0.02 %	2.400	937.5	0.00 %	2.39998	2604.1875	0.00 %
2	9.6	9.604	52.0625	0.04 %	9.600	234.375	0.00 %	9.59417	651.4375	0.06 %
3	19.2	19.198	26.04375	0.01 %	19.200	117.1875	0.00 %	19.2012	325.5000	0.01 %
4	57.6	57.554	8.6875	0.08 %	57.600	39.0625	0.00 %	57.6037	108.5000	0.01 %
5	115.2	115.942	4.3125	0.64 %	115.385	19.5	0.16 %	115.207	54.2500	0.01 %
6	230.4	228.571	2.1875	0.80 %	230.769	9.75	0.16 %	230.415	27.1250	0.01 %
7	460.8	470.588	1.0625	2.08 %	461.538	4.875	0.16 %	460.829	13.5625	0.01 %
8	921.6		Impossible		923.077	2.4375	0.16 %	925.926	6.7500	0.47 %
9	2250		Impossible		2250.000	1	0.00 %	2272.73	2.7500	1.00 %
10	4500		Impossible	Impossible	Impossible	Impossible	Impossible	4545.45	1.3750	1.00 %

Notes:

1. The lower the clock frequency of the CPU, the lower the error at a particular baud rate. The upper limit of the achievable baud rate can be fixed with this data.

2: 8-bit oversampling mode

Tx/Rx baud rate = $\frac{fck}{(8*USARTDIV)}$

3. Do not use a frequency division factor less than 1

37.3.7 USART Receiver Tolerance Clock Changes

The USART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the USART receiver. The causes which contribute to the total deviation are:

● DTRA: Variation due to transmitter error (including variation of transmitter-side oscillator)

● DQUANT: Error caused by receiver-side baud rate rounding

● DREC: Receiver Side Oscillator Changes

● DTCL: Variation due to transmission line (usually due to inconsistency between transceiver conversion timing from low to high and conversion timing from high to low).

Need to meet: DTRA + DQUANT + DREC + DTCL < USART receiver tolerance

Table 37-3 USART receiver tolerance when DIV_Fraction is 0

M bit	over8 = 0				over8 = 1			
	NF is an error		NF doesn't care		NF is an error		NF doesn't care	
00	3.83%	-2.71%	3.79%	-5.90%	1.16%	-4.50%	1.16%	-5.93%
01	3.63%	-4.57%	4.17%	-5.06%	2.35%	-3.73%	3.18%	-4.99%
10	3.91%	-2.86%	4.17%	-6.30%	2.78%	-5.56%	1.39%	-6.25%

Table 37-4 USART receiver tolerance when DIV_Fraction is not 0

M bit	over8 = 0				over8 = 1			
	NF is an error		NF doesn't care		NF is an error		NF doesn't care	
00	3.07%	-2.08%	3.78%	-5.62%	0.89%	-4.05%	1.34%	-5.73%
01	3.71%	-4.14%	3.76%	-4.63%	1.96%	-3.50%	3.14%	-4.58%
10	3.56%	-2.14%	3.83%	-5.88%	2.08%	-4.17%	0.69%	-4.69%

37.3.8 USART Auto baud rate detection

- **Auto baud rate detection**

USART can automatically detect the baud rate based on the received characters and configure the baud rate register USART_BRR.

Automatic baud rate detection is suitable for the following situations:

The system communication speed is unknown in advance;

The system uses a less accurate clock source, a mechanism that allows the correct baud rate to be obtained without measuring the clock deviation;

The clock source frequency must be compatible with the desired communication speed:

The baud rate for 1.16 bit oversampling must be between fCK/65535 and fCK/16.

The baud rate for 2.8-bit oversampling must be between fCK/65535 and fCK/8.

Configure the automatic baud rate detection mode via USART_CR2. ABRMOD. There are 2 modes based on different character patterns. In these automatic baud rate modes, the baud rate is measured several times during synchronous data reception, and the result of each measurement is compared with the previous one.

- **Automatic baud rate detection mode**

Mode0: This mode is for cases starting with 1. In this case, the start bit is measured to 1 (falling edge to rising edge).

Mode1: This pattern is for any character beginning with a 10xx bit pattern. In this case, USART measures the duration of the start bit and the first data bit. Measurements are taken falling edge to falling edge, ensuring better accuracy in the case of slow signal tilt.

Parallel data mode: detection is performed for each intermediate transition of the RX line. If the conversion on the RX is not sufficiently synchronized with the receiver (the receiver calculates the baud rate based on 0 bits), an error will occur.

Note: In automatic baud rate detection mode, if the baud rate clock is close to one frequency division of the fck clock, it will lead to a large error.

37.3.9 Multiprocessor communication

It is possible to perform USART multiprocessor communications (with several USARTs connected in a network). For example, a USART device may be the master, and its TX output is connected to the RX input of other USART slave devices; The respective TX output logic of the USART slave devices is coupled together and connected to the RX input of the master device.

In multiprocessor configurations, it is often desirable that only the intended message recipient actively receives the full message contents, thus reducing redundant USART service overhead for all non-addressed receivers. The unaddressed device is configured in silent mode.

The silence function allows non-addressable devices to be placed in silent mode. To use the silent mode feature, the RWU bit must be set in the USART_CR1 register.

When the Mute mode is enabled:

● Any receive status bits will not be set.

● All reception interrupts are disabled.

● The RWU bit in the USART_CR1 register is set to 1. RWU can be controlled automatically by hardware or by software under certain conditions.

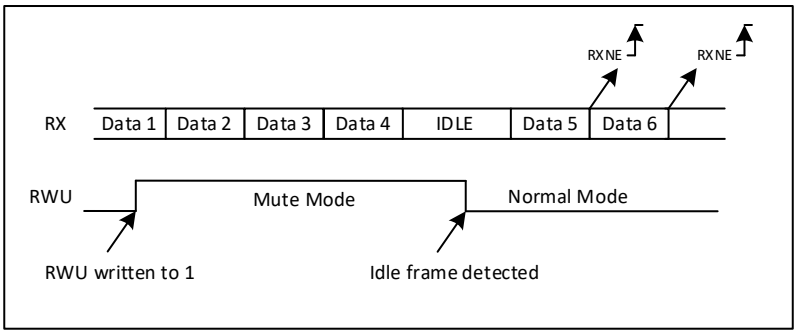
Depending on the status of the WAKE bit in the USART_CR1 register, USART can enter or exit the silent mode in two ways.

● If the WAKE bit is set to 0: Idle bus detection is performed.

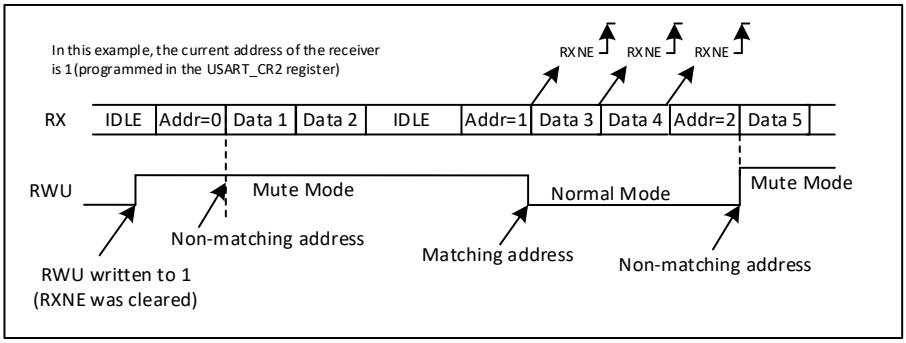
● If the WAKE bit is set to 1: Address tag detection is performed.

37.3.9.1 Idle bus detection (WAKE=0)

The USART enters mute mode when the RWU bit is written to 1. It wakes up when an Idle frame is detected. RWU can also be written to 0 by software. An example of mute mode behavior using idle line detection is given in the figure below.



Silent mode using idle bus detection



37.3.9.2 address mark detection (WAKE = 1) (4-or 7-bit address detection)

In this mode, bytes are recognized as addresses if their MSB is a ‘1’ else they are considered as data. In one address byte, the address of the target receiver is placed in a 4-bit or 7-bit LSB. This address is compared by the receiver to its own address, which is programmed in the ADD of the USART_CR2 register. The selection of 7-bit or 4-bit address detection is done using ADDM 7-bit.

The USART enters mute mode when an address character is received which does not match its programmed address. In this case, the RWU bit is set by hardware. The RXNE flag is not set for this address byte and no interrupt nor DMA request is issued as the USART would have entered mute mode. It exits from mute mode when an address character is received which matches the programmed address. Then the RWU bit is cleared, and subsequent bytes are received normally. The RXNE bit is set for the address character since the RWU bit has been cleared. The RWU bit can be written to as 0 or 1 when the receiver buffer contains no data (RXNE=0 in the USART_SR register). Otherwise, the write attempt is ignored. An example of mute mode behavior using idle line detection is given in the figure below.

Note: In 7-bit and 9-bit data modes, address detection is performed on 6-bit and 8-bit addresses (ADD [5: 0] and ADD [7: 0]), respectively. This address byte does not set the RXNE flag, and no interrupt or DMA request is issued when USART enters silent mode. When FIFO management is enabled, the software should ensure that there is at least one empty spot in the RXFIFO before entering silent mode. When RWU is written to 1, USART also goes into silent mode. In this case, the RWU bit is automatically set as well. USART exits silent mode when an address character matching the programmed address is received. Clear the RWU bit and receive subsequent bytes normally. When the RWU bit has been cleared, the address character can detect the RXNE/RXFNE bit.

37.3.10 USART Modbus Correspondence

The USART offers basic support for the implementation of Modbus/RTU and Modbus/ASCII protocols. Modbus/RTU is a half-duplex block transfer protocol. The control parts of the protocol (address identification, block integrity control, and command interpretation) must be implemented in software. USART provides basic support: end block detection, with no software overhead or other resources.

Modbus/RTU

In this mode, the end of a block is recognized by "silent" (idle line) for more than 2 characters. This function is realized through a programmable timeout function. The timeout function and interrupt must be activated by the RTOEN bit in the USART_CR2 register and the RTOIE in the USART_CR1 register. Timeout values corresponding to 2 character times (e.g. 22 x bit times) must be programmed in the RTO register. When the receive line is idle during this period of time, after the last stop bit is received, an interrupt is generated to inform the software that the current block reception is complete.

Modbus/ASCII

In this mode, the end of the block is identified by a specific (CR/LF) character sequence. USART uses the character matching function to manage this mechanism.

Exit the silent mode when the corresponding character is received by writing the corresponding character in the ADD [7: 0] field.

37.3.11 Parity control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the USART_CR1 register.

Even check: The check bit causes 7 or 8 LSB data in a frame and the number of ones in the check bit to be even. For example: data = 00110101, there are 4 ones, if even check is selected (PS = 0 in USART_CR1), the check bit will be 0.

Odd check: This check bit makes 7 or 8 LSB data in a frame and the number of ones in the check bit odd.

For example: data = 00110101, there are 4 ones, if odd check is selected (PS = 1 in USART_CR1), the check bit will be 1.

Transmission mode: If the PCE bit of USART_CR1 is set, the MSB bit of the data written into the data register is replaced by the check bit and sent out (if even check even ones, if odd check odd ones are selected. If the parity fails, the PE flag in the USART_SR register is set to 1, and an interrupt is generated if the PEIE of the USART_CR1 register is set in advance.

37.3.12 LIN Mode

Configure USART_CR2. LINEN = 1 to select LIN mode. In LIN mode, the following bits must be kept cleared:

1. CLKEN of the USART_CR2 register;
2. STOP/SCEN/HDSEL/IREN of the USART_CR3 register.

37.3.12.1 Transmission

Compared with general USART transmission, LIN transmission has the following differences:

M = 0, the data length is 8 bits;

USART_CR2. LINEN = 1 needs to be configured. After setting SBK, 13 bits of 0 will be transmitted as a disconnected frame.

37.3.12.2 Reception

When the LIN mode is enabled, the open frame detection circuit is activated. The detection is totally independent from the normal USART receiver. The disconnected frame can be detected as soon as it appears, whether when the bus is idle or during the transmission of a data frame, the data frame has not been completed, and the transmission of the disconnected frame is inserted.

When the receiver is activated (RE = 1 for USART_CR1), the circuitdetectsa start signal on RX. The method of detecting the start bit is the same as the detection of broken frames or data. When the start bit is detected, the circuit samples each subsequent bit at the 8th, 9th and 10th oversampling clock points of each bit. If 10 (when LBDL of USART_CR2 = 0) or 11 (when LBDL of USART_CR2 = 1) consecutive bits are 0, followed by a delimiter, the LBD flag of USART_SR is set. If the LBDIE bit=1, an interrupt is generated. Before confirming that the frame is broken, check the delimiter because it means that the RX line has returned to high level.

If a ‘1’ is sampled before the 10 or 11 have occurred, the break detection circuit cancels the current detection and searches for a start bit again. If the LIN mode is disabled, the receiver continues to operate as normal USART, and there is no need to consider detecting a broken frame.

If LIN mode is not activated (LINEN = 0), the receiver still operates normally in USART mode and does not perform disconnection detection.

If LIN mode is activated (LINEN = 1), the receiver stops as soon as a frame error occurs (i.e., the stop bit detects' 0 ', which occurs in a broken frame) until the broken frame detection circuit receives either a 1 (this occurs when the broken frame is not completely issued), or a delimiter (this occurs when a complete broken frame has been detected).

本情况：断开帧长度不够：需要舍弃该帧，不设置LBD

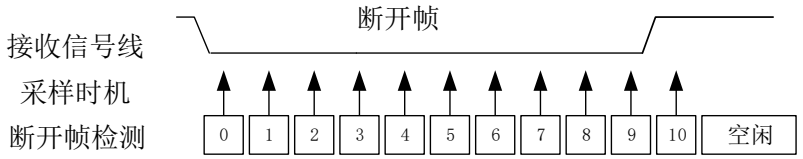
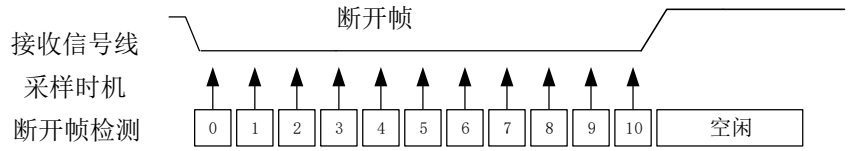


Figure 37-7 Case of insufficient disconnected frame length in LIN mode

本情况：断开帧长度刚好，设置LBD



本情况：断开帧长度很长，设置LBD

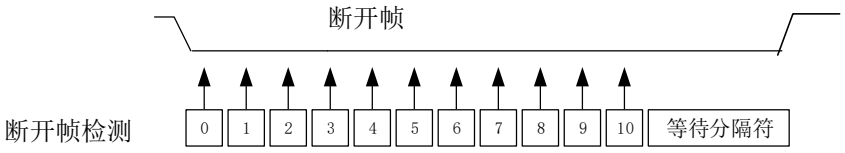
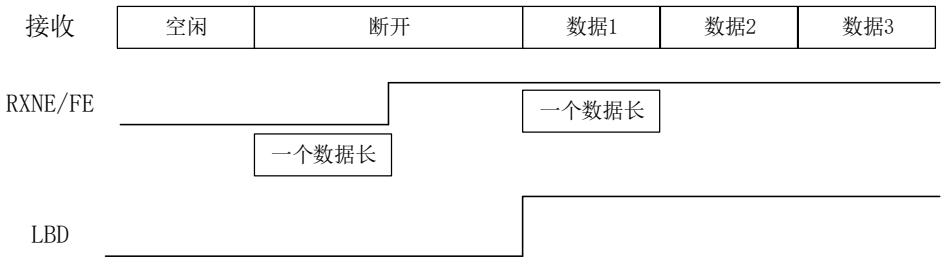


Figure 37-8 Case where the disconnected frame length in LIN mode is sufficient

断开发生在空闲后



断开发生在正在接收数据时

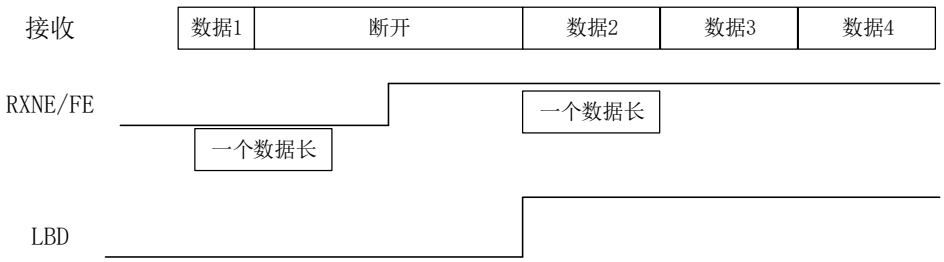


Figure 37-9 Disconnection detection vs. frame error detection in LIN mode

37.3.13 Synchronous Mode

Select the synchronization mode by writing the CLKEN bit on the USART_CR2 register

37.3.13.1 Master mode

In synchronous mode, the following bits must be kept cleared:

LINEN bit in the USART_CR2 register

● SCEN, HDSEL and IREN bits in the USART_CR3 register

The USART can be used to control bidirectional synchronous serial communications in master mode. The CK pin is the output of the USART transmitter clock. No clock pulses are sent to the CK pin during start bit and stop bit. Depending on the state of the LBCL bit in the USART_CR2 register, clock pulses are, or are not, generated during the last valid data bit. The CPOL bit in the USART_CR2 register isused to select the clock polarity, and the CPHA bit in the USART_CR2 register is used to select the phase of the external clock.

The external CK clock is not activated during bus idle periods, until actual data arrives and when a disconnect frame is transmitted. In synchronous master mode, the USART transmitter operates exactly like in asynchronous mode. However, since CK is synchronized with TX (according to CPOL and CPHA), the data on TX is synchronous.

In synchronous master mode, the USART receiver operates in a different way compared to asynchronous mode. If RE is set to 1, the data are sampled on CK (rising or falling edge, depending on CPOL and CPHA), without any oversampling. A given setup and a hold time must be respected (which depends on the baud rate: 1/16 bit time).

Notes:

1. The CK foot and the TX foot work together. Thus, the clock is provided only if the transmitter is enabled (TE = 1) and data are being transmitted (USART_TDR data register written). This means that it is not possible to receive synchronous data without transmitting data.

2. LBCL, CPOL and CPHA should be configured when both the transmitter and receiver are disabled; When the transmitter or receiver is enabled, these bits cannot be changed

3. It is recommended to set TE and RE in the same instruction to reduce the setup time and hold time of the receiver.

4. USART only supports the main mode: it cannot receive or send data with the input clock from other devices (CK is always the output).

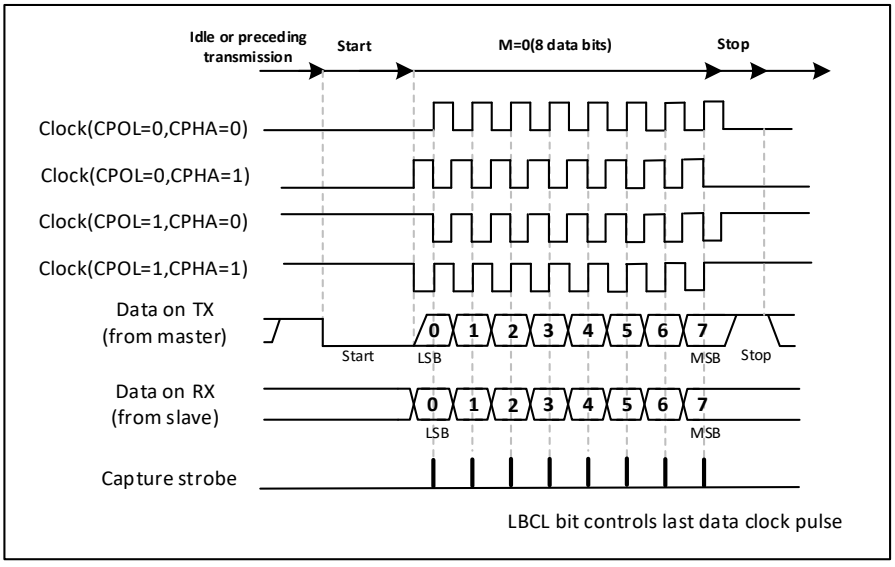


Figure 37-10 USART Data Clock Timing Example (M = 0)

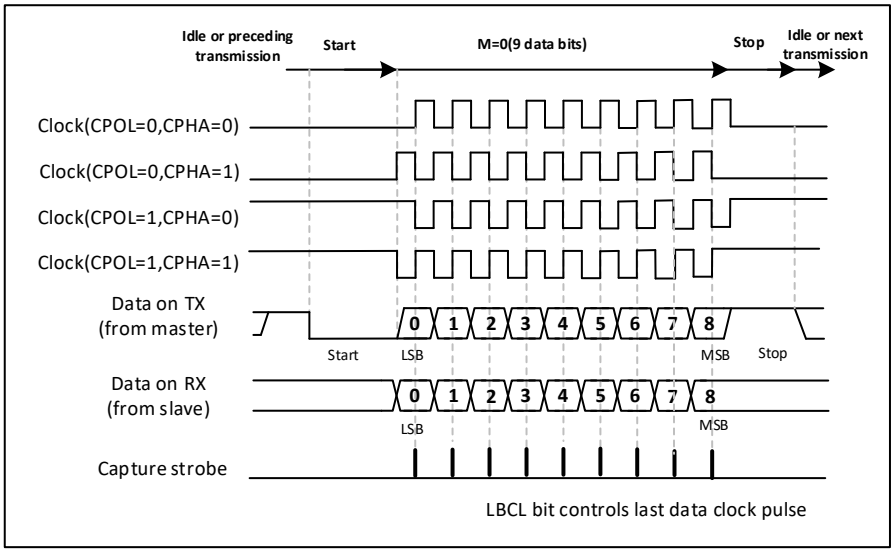


Figure 37-11 RX Data Sample/Hold Time (M = 1)

37.3.14 Single-wire Half-duplex communications

The single-wire half-duplex mode is selected by setting the HDSEL bit in the USART_CR3 register. In this mode, the following bits must be kept cleared:

● LINEN and CLKEN bits of the USART_CR2 register

● SCEN and IREN bits of the USART_CR3 register

The USART can be configured to follow a Single-wire Half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are connected internally. Half-duplex and full-duplex communication are selected using the HDSEL bit in USART_CR3.

When HDSEL is "1":

● RX No longer in use

● When there is no data transmission, TX is always released. Therefore, it behaves as a standard I/O port when idle or received. This means that the I/O must be configured as an open drain when it is not driven by USART.

Apart from this, the communication protocol is similar to normal USART mode. Conflicts on the line are managed by software (e.g. through the use of a central arbiter). In particular, the transmission is never blocked by hardware. The transmission is never blocked by hardware and continues as soon as data are written in the data register while the TE bit is set.

37.3.15 USART receive timeout

The receiver timeout feature is enabled by setting the RTOEN bit in the USART_CR2 control register.

The timeout is set using the RTO bit field in the USART_RTOR register.

When the set timeout time is exceeded, the USART_SR register is setto 1. If the RTOIE of CR1 is set, the corresponding interrupt will be generated.

37.3.16 Smartcard

Smartcard mode is selected by setting the SCEN bit in the USART_CR3 register. In Smartcard mode, the following bits must be kept cleared:

● LINEN bit of USART_CR2 register

● HDSEL and IREN bits of the USART_CR3 register

Moreover, the CLKEN bit may be set in order to provide a clock to the smartcard.

The Smartcard interface is designed to support asynchronous protocol Smartcards as defined in the ISO 7816-3 standard. T = 0 (character mode) and T = 1 (block mode) are both supported.

37.3.16.1 T = 0 (character mode)

The USART should be configured as:

● 8-bit data bit plus check bit: At this time, M = 1 and PCE = 1 in the USART_CR1 register

● 1.5 STOP bits when sending and receiving: that is, STOP = 11 of the USART_CR2 register

Note: It is also possible to choose 0.5 stop bit for receiving but it is recommended to use 1.5 stop bits for both transmitting and receiving to avoid switching between the two configurations.

In T = 0 (character) mode, a parity error is indicated at the end of each character during the guard period. Figure below shows examples of what can be seen on the data line with and without parity error.

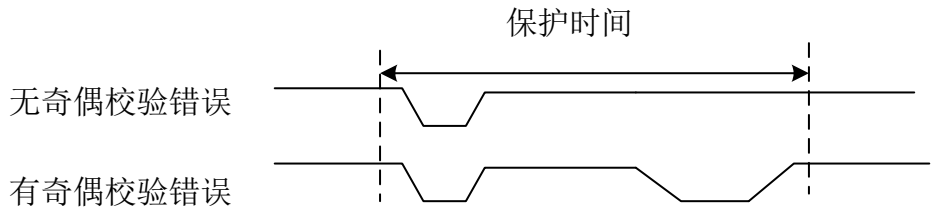


Figure 37-12 ISO7816-3 asynchronous protocol

When connected to a smartcard, the USART TX output drives a bidirectional line that is also driven by the smartcard. In order to do this, SW_RX must be connected to the same I/O port as TX. The output enable bit TX_EN of the transmitter is set during the transmission of the start bitand the data byte, and is released during the transmission of the stop bit (weak pull-up), so the receiver can pull the data line low in the event that a check error is found. If TX_EN is not used, TX is pulled high during the stop bit: in this case, the receiver can drive this line as long as TX is configured as an open drain.

Smartcard is a single wire half duplex communication protocol.

● Data is sent out from the send shift register and is delayed by a minimum of 1/2 baud clock. In normal operation a full transmit shift register will start shifting on the next baud clock edge. In Smartcard mode this transmission is further delayed by a guaranteed 1/2 baud clock.

● If a parity error is detected during reception of a data frame set to 0.5 or 1.5 stop bits, the transmit line is pulled down one clock cycle after the reception of the frame is completed (i.e., at the end of the stop bits). This is to indicate to the Smartcard that the data transmitted to USART has not been correctly received. This NACK signal (pulling down the transmit line by one clock cycle) will produce a frame error at the transmit end (the transmit end is configured with 1.5 stop bits). The application may handle the retransmission data according to the protocol. A parity error is ‘NACK’ed by the receiver if the NACK control bit is set, otherwise a NACK is not transmitted. The TXE bit (TXFNF bit when FIFO mode is enabled) can be set using the TXFRQ bit in the register. If the received character is wrong, the RXNE (RXFNE when FIFO mode is enabled)/receive DMA request is not activated.

● Automatic retry in smart card transmission: A delay of 2.5 clock cycles is inserted between the USART detection NACK and the start bit of the repeated character. The TC bit is set immediately

after the last repeated character is received (unprotected time). If the software wants to repeat it again, it must ensure the minimum 2 clock cycles required by the standard.

● In transmission, USART inserts a guard time (as per the guard time register) between two consecutive characters. Since the guard time is measured after the stop bit of the previous character, the GT [7: 0] register must be programmed to the required CGT (character guard time, defined by the 7816-3 specification) minus 12 (the duration of one character).

● The setting of the TC flag can be delayed by programming the guard time register. In normal operation, TC is asserted when the transmit shift register is empty and no further transmit requests are outstanding. In smart card mode, an empty transmit shift register will trigger the guard time counter to start counting up, and TC is forced down during this time. When the guard time counter reaches the value in the guard time register, TC is set high.

● The revocation of the TC flag is not affected by the smart card mode.

● If the transmitter detects a frame error (receiving a NACK signal from the receiver), the receiving function module of the transmitter does not detect the NACK as the start bit. According to the ISO protocol, the duration of the received NACK may be 1 or 2 clock cycles.

● On the receiver side, if a check error is detected and a NACK is transmitted, the receiver does not detect the NACK as the start bit.

Note:

1. The disconnect frame does not make sense in smart card mode. A 0x00 data with a frame error will be treated as data rather than a broken frame.

2. When switching TE bits back and forth, no IDLE frame is transmitted. The IDLE frame is not defined by the ISO protocol.

Figure below details how the NACK signal is sampled by the USART. In this example, the USART transmits a data and is configured with 1.5 stop bits. The receiver part of the USART is enabled in order to check the integrity of the data and the NACK signal.

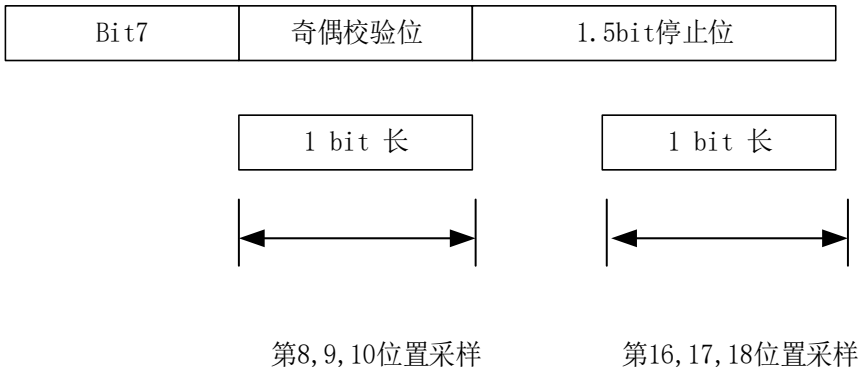


Figure 37-13 1.5 stop bit detection of parity error

The USART can provide a clock to the Smartcard through the CK output. In Smartcard mode, CK is not associated to the communication but is simply derived from the internal peripheral input clock through a 5-bit prescaler. The division ratio is configured in the prescaler register USART_GTPR. The CK frequency can be programmed from fCK/2 to fCK/62, where fCK is the peripheral input clock.

37.3.16.2 T = 1 (block mode)

In block mode, the software must program the RTOR register to a BWT (Block Wait Time)-11 value when making a read request to the smart card. If no reply is received from the card before the expiration of this period, a timeout interrupt is generated. An RXNE/RXFNE interrupt is signaled if the first character is received before the expiration of the period

NOTE: The RXNE/RXFNE interrupt must be enabled even when reading from the smart card in block mode using USART in DMA mode. At the same time, DMA must only be enabled after the first byte is received.

In a smart card protocol definition, the BWT/CWT value should be defined from the beginning of the last character (start bit). Given the length of the last character itself, the RTO register must be programmed as BWT-11 or CWT-11, respectively.

The block length counter is used to count all characters received by USART. When USART sends, this counter is reset. The length of the block is communicated by the smart card in the third byte (preamble byte) of the block. This value must be programmed into the BLEN field of the USART_RTOR register. When using DMA mode, this register field must be programmed to a minimum value before the block starts. According to this value, an interrupt is generated after the 4th character received. The software must read the LEN field (third byte) and its value must be read from the receive buffer.

In interrupt-driven receive mode, the length of the block can be checked by software or by programming the BLEN value. However, the maximum value of BLEN (0xFF) can be programmed before the start of the block. The true value is programmed after receiving the third character.

If the block uses the LRC longitudinal redundancy check (coda byte), then BLEN = LEN. If the block uses the CRC mechanism (2 bytes), then BLEN = LEN+1 must be programmed. The total block length (including preamble, epilogue, and information bytes) is equal to BLEN+4. The end of the block is signaled to the software via an EOBF flag and an interrupt (when the EOBIE bit is set).

37.3.17 IrDA SIR ENDEC Functional module

IrDA mode is selected by setting the IREN bit in the USART_CR3 register. In IrDA mode, the following bits must be kept cleared:

- LINEN, STOP and CLKEN bits of the USART_CR2 register
- The SCEN and HDSEL bits of the USART_CR3 register.

37.3.17.1 IrDA normal mode

The IrDA SIR physical layer specifies the use of an inverted return-to-zero modulation scheme (RZI), which uses an infrared light pulse to represent a logic '0' (see figure below). The SIR transmit encoder modulates the NRZ (non-return-to-zero) bitstream output from the USART. The output pulse stream is transmitted to an external output driver and infrared LED. USART is SIR ENDEC only supports up to 115.2 Kbps speeds. In normal mode the transmitted pulse width is specified as 3/16 of a bit period.

The SIR receive decoder demodulates the return-to-zero bit stream from the infrared detector and outputs the received NRZ serial bit stream to USART. In the idle state, the decoder input is typically high (flag state). The transmit encoder output has the opposite polarity to the decoder input. A start bit is detected when the decoder input is low.

● IrDA is a half-duplex communication protocol. If the transmitter is busy (i.e. USART is sending data to the IrDA encoder), any data on the IrDA receive line will be ignored by the IrDA decoder. If the receiver is busy (i.e. the USART is receiving decoded data from the IrDA decoder), the data on the TX from the USART to the IrDA will not be encoded by the IrDA. While receiving data, transmission should be avoided as the data to be transmitted could be corrupted.

● The SIR transmit logic transmits' 0 'as high and' 1 'as low. The width of the pulse is specified as 3/16th of the selected bit period in normal mode.

● The SIR receive logic interprets the high state as' 1 'and the low pulse as' 0 '.

● The output of the transmit encoder has opposite polarity to the input of the decoder. The SIR output is in low state when idle.

● The SIR decoder converts the IrDA-compliant received signal into a bitstream for the USART.

● The IrDA specification requires pulses to be wider than 1.41 us. The acceptable pulse width is programmable. The pulse detection logic on the receiver side filters out pulses having a width of less than 2 PSC cycles (PSC is a prescaled value programmed in the IrDA low power baud rate register USART_GTPR). Pulses of width less than 1 PSC period are always rejected, but those of width greater than one and less than two periods may be accepted or rejected, those greater than 2 periods will be accepted as a pulse. The IrDA encoder/decoder doesn’t work when PSC=0.

● The receiver may communicate with a low power transmitter.

● In IrDA mode, the STOP bit on the USART_CR2 register must be configured as a STOP bit.

37.3.17.2 IrDA low-power mode

Transmitter:

In low-power mode the pulse width is not maintained at 3/16 of the bit period. Instead, the width of the pulse is 3 times the low-power baud rate which can be a minimum of 1.42 MHz. Usually this value is 1.8432 MHz (1.42 MHz < PSC < 2.12 MHz). A low- power mode programmable divisor divides the system clock to achieve this value.

Receiver:

Receiving in low-power mode is similar to receiving in normal mode. In order to filter out spike interference pulses, USART should filter out pulses with a width of less than 1 PSC. Only a low level signal of an IrDA low power baud rate clock (PSC in USART_GTPR) with a duration greater than 2 cycles is accepted as a valid signal.

Notes:

1. Pulses having a width of less than 2 and greater than 1 PSC cycle may or may not be filtered out.

2. The setup time of the receiver should be managed by the software. The IrDA physical layer specification specifies a minimum delay of 10 ms between transmission and reception (IrDA is a half-duplex protocol).

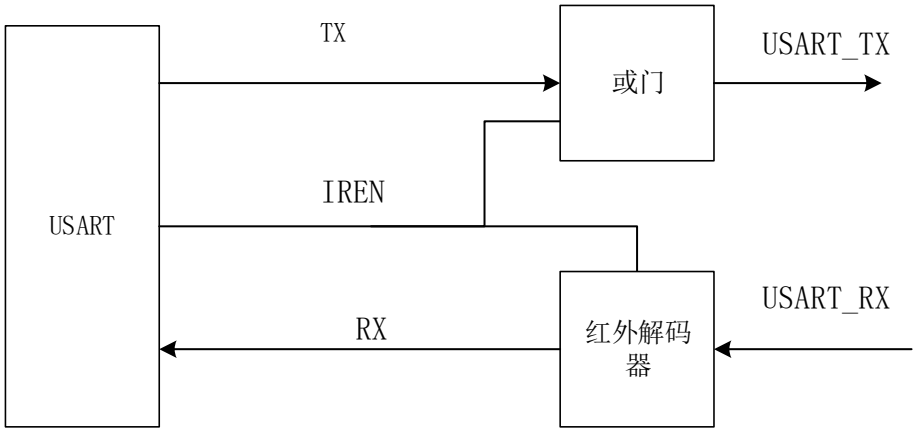


Figure 37-14 IrDA SIR ENDEC block diagram

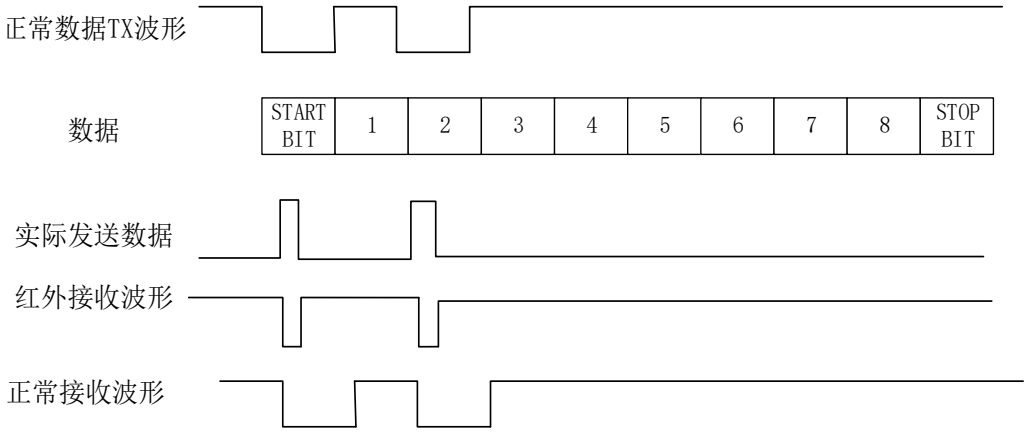


Figure 37-15 IrDA Data Modulation (3/16)-Normal Mode

37.3.18 Continuous Communication Using DMA

The USART is capable of performing continuous communications using the DMA. DMA requests for the RX buffer and the TX buffer are generated separately.

Note: In the USART2_SR register, the TXE/RXNE flag can be cleared to achieve continuous communication.

Send with DMA:

DMA mode can be enabled for transmission by setting DMAT bit in the USART_CR3 register. When the TXE bit is set to '1', the DMA transfers data from the designated SRAM area to the USART_DR register.

1. Write the USART_DR register address in the DMA control register to configure it as the destination of the transfer. Data is also moved from memory to this address after each TXE (if FIFO mode TXFNF is enabled)
2. Write the memory address in the DMA control register to configure it as the transfer source. After each TXE (or TXFNF, if FIFO mode is enabled) event, data is loaded from this memory area into the USART_DR register
3. Configure the total number of bytes to be transferred to the DMA control register.
4. Configure the channel priority in the DMA register
5. Depending on the requirements of the application, the configuration generates a DMA interrupt when the transfer is half or all complete.
6. Activate the channel on the DMA register.

When the number of data transfers programmed in the DMA Controller is reached, the DMA controller generates an interrupt on the DMA channel interrupt vector. Detecting the TC flag of the USART_SR register can confirm whether the USART communication has ended, which can avoid corruption of the last transmitted data before turning off USART or entering the shutdown mode; The software needs to wait for TXE = 1 first and then for TC = 1.

Receiving with DMA:

Reception using DMA can be activated by setting the DMAR bit of the USART_CR3 register. Each time one byte is received, the DMA controller transfers data from the USART_DR register to the designated SRAM area (refer to DMA related description). The steps to assign a DMA channel for USART reception are as follows (x represents the channel number):

1. Configure the USART_DR register address as the source address of the transfer through the DMA control register. The data will be moved from this address to the memory after each RXNE event.
2. Configure the memory address as the destination address of the transfer through the DMA control register. After each RXNE event, data is transferred from USART_DR to this memory.
3. Configure the total number of bytes to be transferred in the DMA control register.
4. Configure the channel priority on the DMA register.
5. According to the requirements of the application, configure whether the DMA interrupt is generated when the transmission is half or all completed.
6. Activate this channel on the DMA control register.

When the number of data transfers programmed in the DMA Controller is reached, the DMA controller generates an interrupt on the DMA channel interrupt vector.

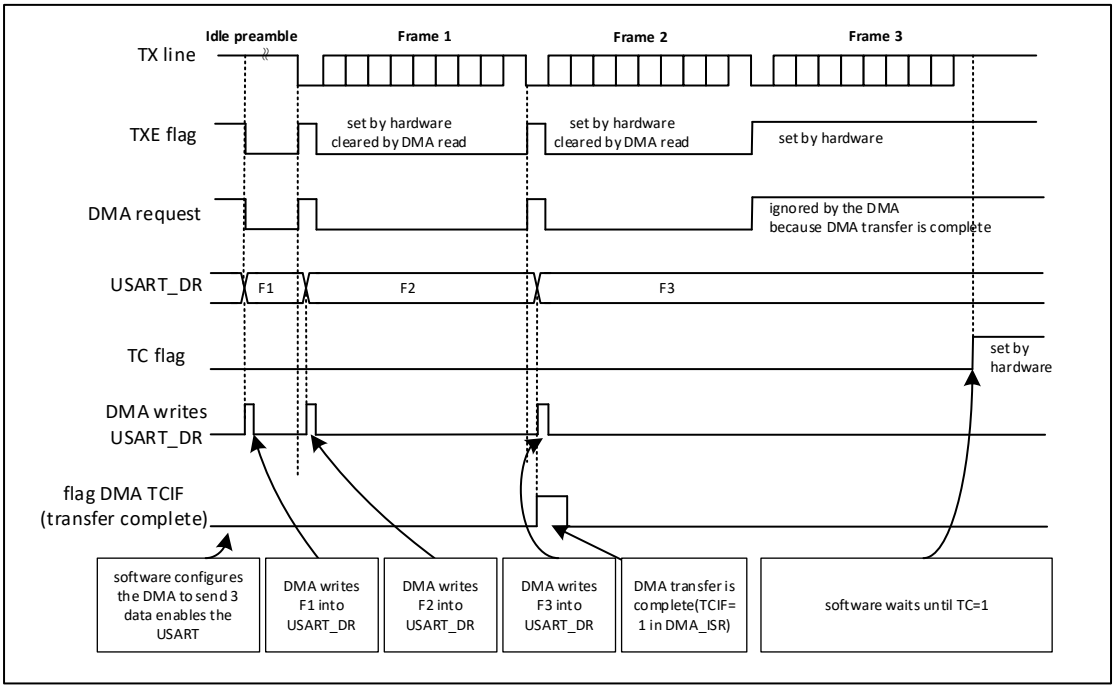


Figure 37-16 transmitted using DMA

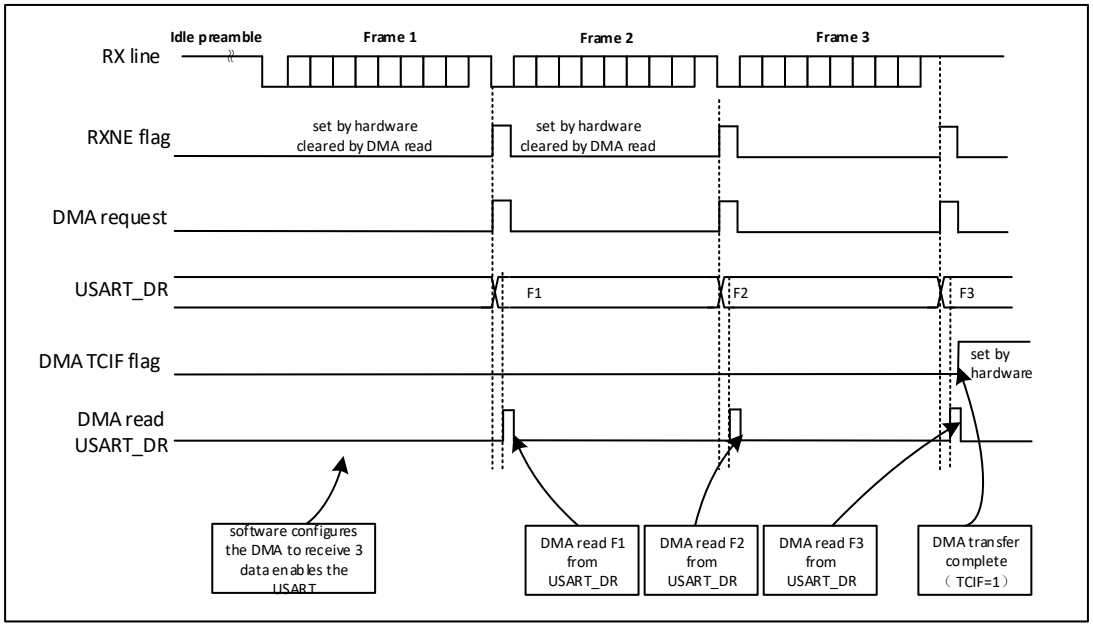


Figure 37-17 Using DMA Receiving

Error flags and interrupt generation in multi-buffer communication

In the case of multi-buffer communication, if any error occurs during the communication, an error flag will be set after the current byte is transmitted.

An interrupt will be generated if the interrupt enable flag is set. For framing error, overrun error and noise flag which are asserted with RXNE incase of single byte reception, there will be separate error flag interrupt enable bit, which if set will issue an interrupt after the current byte with either of these errors.

37.3.19 Hardware flow control

37.3.19.1 RS232 hardware flow control

It is possible to control the serial data flow between 2 devices by using the nCTS input and the nRTS output. The figure below shows how to connect two devices in this mode.

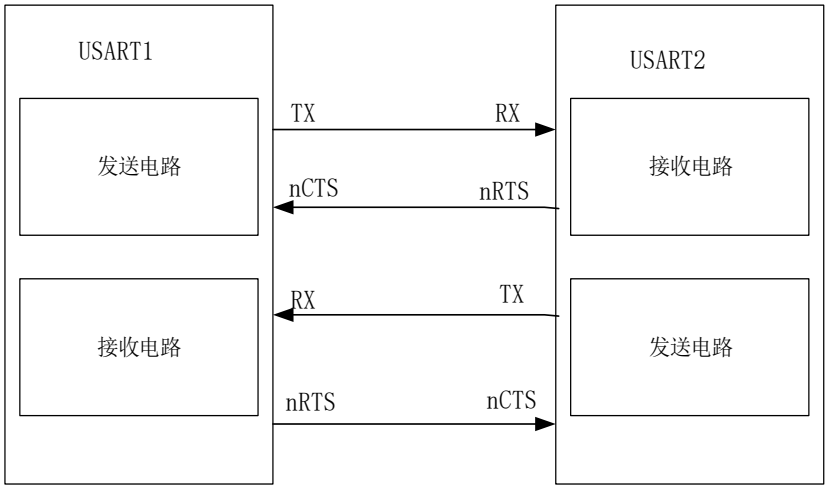


Figure 37-18 Hardware flow control between two USARTs

By setting RTSE and CTSE in USART_CR3, RTS and CTS flow control can be enabled independently, respectively.

RTS Flow Control:

If the RTS flow control is enabled (RTSE=1), then nRTS is asserted (tied low) as long as the USART receiver is ready to receive a new data. When the receive register is full, nRTS is deasserted, indicating that the transmission is expected to stop at the end of the current frame. The following diagram is an example of a communication with RTS flow control enabled.

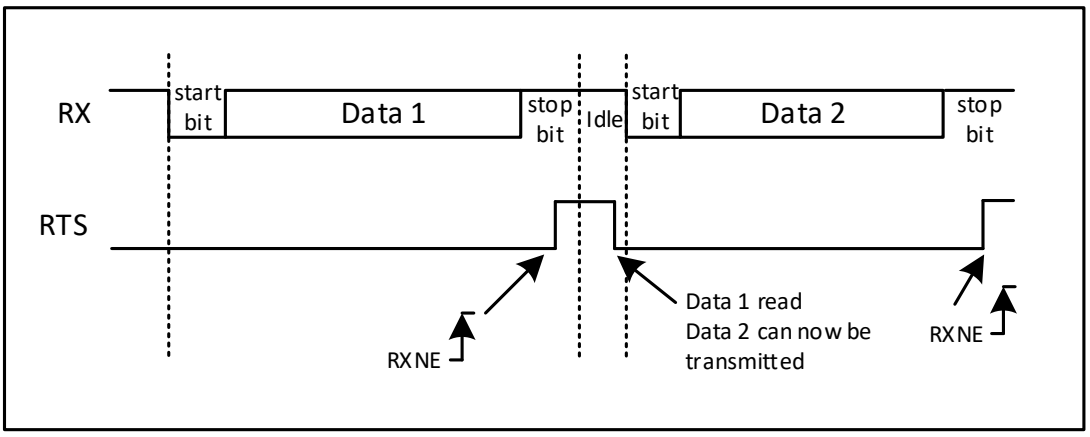


Figure 37-19 RTS flow control

CTS Flow Control:

If the CTS flow control is enabled (CTSE=1), then the transmitter checks the nCTS input before transmitting the next frame. If nCTS is valid (pulled low), the next data is transmitted (assuming that data is ready for transmission, i.e. TXE = 0), otherwise the next frame of data is not transmitted. When CTS is deasserted during a transmission, the current transmission is completed before the transmitter stops.

When CTSE = 1, the CTSIF status bit is automatically set by hardware as soon as the nCTS input toggles. It indicates when the receiver becomes ready or not ready for communication. An interrupt is generated if the CTSIE bit in the USART_CR3 register is set. The following diagram is an example of enabling CTS flow control communication.

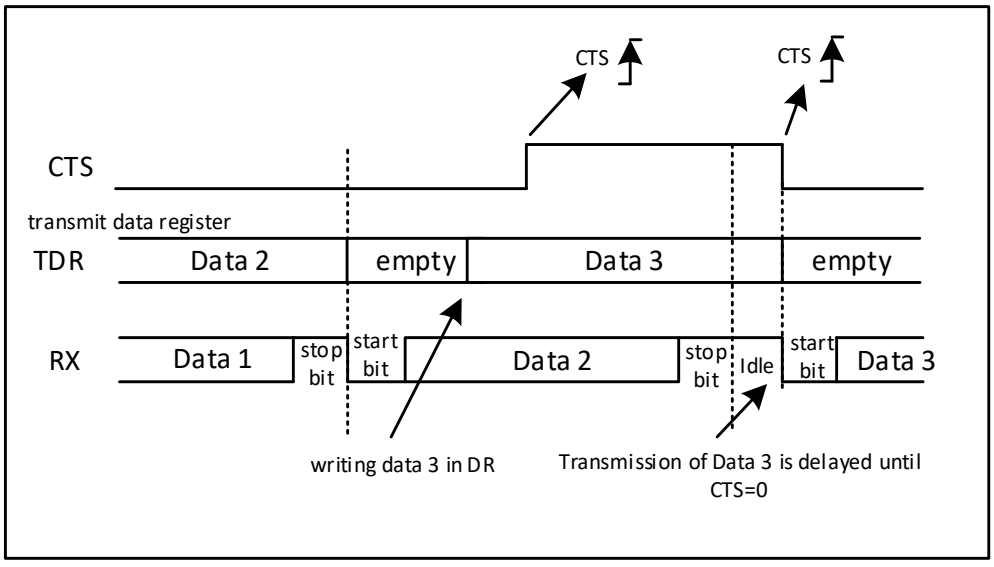


Figure 37-20 CTS flow control

37.3.19.2 RS485 hardware flow control

Enabled by setting the bit DEM in the USART_CR3 control register, the user is able to activate external transceiver controlthrough the DE (drive enable)signal. The assertion time is the time when the DE signal is activated to the start of the start bit. It is programmed using the DEAT [4: 0] bit field in the USART_CR1 control register. The deactivation time is the time from the end of the last stop bit in the transmission data to the deactivation of the DE signal. It is programmed using the DEDT [4: 0] bit field in the USART_CR1 control register. The polarity of the DE signal can be configured using the DEP bit in the USART_CR3 control register.

In USART, DEAT and DEDT are expressed in sampling time units (1/8 or 1/16 bit time, depending on the oversampling rate).

37.3.20 Interrupt requests

Table 37-5 USART interrupt requests

Interrupt vector	Flag bit	Enable Control bit
usart	txe	txeie
	cts	ctsie
	tc	tcie
	rxne	rxneie
	idle	idleie
	pe	peie
	lbd	lbdie
	ne/ore/fe	eie
	txft	txftie
	rxft	rxftie
	rxff	rxffie
	txfe	txfeie
	rtof	rtofie
	eobf	eobfie

37.4 TIMx registers

37.4.1 Status Register (SR)

Address offset: 0x00

Reset value: 0x00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												RTOF	EOBF	Res	TXFE
-												RC_W0	RC__W0	-	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFF	RXFT	TXFT	ABRRQ	ABRE	ABRF	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
R	R	R	W	R	R	RC_W0	RC__W0	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
19	RTOF	RC_W0	0	After the timeout value programmed in the RTOR register has been exceeded, if there is no communication, this bit is set to 1 by the hardware. If RTOIE = 1 in the USART_CR2 register, an interrupt is generated.

				In smart card mode, this timeout corresponds to a CWT or BWT time. 0: Value not reached 1: Timeout value has been reached, no data has been received Note: If the time value programmed in the RTOR register separates 2 characters, RTOF is not set to 1. If this time is greater than this value + 2 sample times (2/16 or 2/8, depending on the oversampling method), the RTOF flag is set to 1. The counter counts even if RE = 0, but RTOF is only set to 1 when RE = 1. If the timeout has expired when RE is set to 1, then RTOF will be set to 1. If USART does not support the receiver timeout function, this bit is reserved and maintains the reset value.
18	EOBF	RC_W0	0	Block End Flag After receiving the complete block, this bit is set to 1 by the hardware (e.g. T = 1 smart card mode). The detection is performed when the number of bytes received (from the beginning of the block, including the start field) is equal to or greater than BLEN + 4. 0: Block end not reached 1: End of block reached (number of characters) Note: If Smartcard mode is not supported, this bit is reserved and kept at reset value.
17	Reserved	-	-	-
16	TXFE	R	0	When TXFIFO is null, this bit is set to 1 by the hardware. When the TXFIFO contains at least one data, the flag is cleared. The TXFE flag may also be set to 1 by writing 1 to the TXFRQ. 0: TXFIFO Non-empty. 1: TXFIFO Is empty.
15	RXFF	R	0	When the amount of data received corresponds to the RXFIFO size + 1 (RXFIFO is full + 1 data in the USART_RDR register), this bit is set to 1 by the hardware.
14	RXFT	R	0	RXFT: RXFIFO threshold flag When the threshold programmed in RXFTCFG is reached, this bit is set to 1 by the hardware. This means that there is (RXFTCFG-1) data in the receive FIFO and one data in the USART_RDR register. If USART_CR3 RXFTIE bit = 1 (bit 27) in the register, then an interrupt is generated. 0: The received FIFO does not reach the programmed threshold. 1: The receiving FIFO has reached the programmed threshold.
13	TXFT	R	0	TXFIFO Threshold Flag When the TXFIFO reaches the threshold programmed in the TXFTCFG (i.e., the TXFIFO contains TXFTCFG positions), this bit is set to 1 by the hardware. 0: TXFIFO has not reached the programmed threshold. 1: TXFIFO has reached the programmed threshold.
12	ABRRQ	W	0	Auto baud rate request Writing 1 to this bit resets the ABRF flag and requests an automatic baud rate measurement on the next received data frame.
11	ABRE	R	0	Auto baud rate error

				This bit is set by hardware if the baud rate measurement failed (baud rate out of range or character comparison failed). It is cleared by software, by writing 1 to the ABRRQ bit.
10	ABRF	R	0	Auto baud rate detection flag. When the automatic baud rate is set (RXNE = 1 is set at the same time, and an interrupt is generated when the interrupt is enabled), or when the automatic baud rate detection operation goes wrong (ABRE = 1, RXNE = 1, FE = 1), this bit is set to 1 by the hardware. The software clears the bit by writing 1 to the ABRRQ bit.
9	CTS	RC_W0	0	CTS: CTS flag If the CTSE bit is set, it is set high by the hardware when the nCTS input changes state. It is cleared by the software. If the CTSIE in USART_CR3 is' 1 ', an interrupt is generated. 0: No change on the nCTS status line; 1: Changes occurred on the nCTS status line.
8	LBD	RC_W0	0	LBD: LIN Disconnect Detection Flag When a LIN broken frame is detected, the bit is set '1' by hardware and cleared '0' by software (writing 0 to the bit). If LBDIE = 1 in USART_CR3, an interrupt is generated. 0: No LIN break frame detected; 1: LIN disconnect frame detected. Note: An interrupt is generated when LBD=1 if LBDIE=1
7	TXE	R	1	TXE: Send data register empty (FIFO disabled) TXE: TXFIFO not full (FIFO enabled) When the data in the DR register is transferred to the shift register by hardware, this bit is set by hardware. If TXEIE in the USART_CR1 register is 1, an interrupt is generated. Read SRfirstand thenwrite to USART_DR, this bitwillbecleared. 0: The data has not been transferred to the shift register; 1: Data has been transferred to the shift register.
6	TC	RC_W0	1	Send completion flag When a frame containing data is transmitted, and TXE = 1, the position is set to 1 by hardware. If TXEIE in the USART_CR1 register is 1, an interrupt is generated. A write operation to USART_DR clears this bit. 0: The transmission has not been completed; 1: Send complete.
5	RXNE	RC_W0	0	Read data register non-empty flag When data in the RDR shift register is transferred to the USART_DR register, this bit is set by hardware. If RXNEIE in the USART_CR1 register is 1, an interrupt is generated. A read operation to USART_DR may clear this bit. RXNE bits can also be cleared by writing 0, which is only recommended in multi-cache communications. 0: Data not received either 1: Received data, can be read out
4	IDLE	R	0	Bus idle flag detected When the bus is detected to be idle, this bit is set by hardware. If IDLEIE is 1 in

				USART_CR1, an interrupt is generated. Because the software sequence clears this bit (read USART_SR first; then read USART_DR). 0: No idle bus detected; 1: Idle line is detected Note: The IDLE bit will not be set high again until the RXNE bit is set (i.e., an IDLE bus is detected again)
3	ORE	R	0	Overload error flag When RXNE = 1, the data currently received in the shift register needs to be transferred to the DR register, and the hardware transfers this position bit. An interrupt is generated if RXNEIE in USART_CR1. It is cleared by the software sequence (read USART_SR first, then USART_DR). 0: No overload error; 1: Overload error detected. Note: When this bit is set, the value in the DR register is not lost, but the data in the shift register is overwritten. If the EIE bit is set, the ORE flag setting will generate an interrupt in multi-buffer communication mode.
2	NE	R	0	NE: Noise Error Flag When noise is detected in the received frame, the position bit is adjusted by hardware. It is cleared by the software sequence (read USART_SR first, then USART_DR). 0: No noise detected; 1: Noise detected. Note: This bit does not generate an interrupt because it appears with RXNE, and the hardware will generate an interrupt when the RXNE flag is set. In multi-buffer communication mode, if the EIE bit is set, an interrupt is generated when the NE flag is set
1	FE	R	0	FE: Frame Error Flag When a synchronization misalignment, excessive noise or a broken frame is detected, this bit is set by hardware. It is cleared by the software sequence (read USART_SR first, then USART_DR). 0: No frame error detected; 1: Frame error detected or frame broken. Note: This bit does not generate an interrupt because it appears with RXNE, and the hardware will generate an interrupt when the RXNE flag is set. If the word currently being transferred causes both frame error and overrun error, it will be transferred, and only the ORE bit will be set. In multi-buffer communication mode, if the EIE bit is set, an interrupt is generated when the FE flag is set.
0	PE	R	0	PE: Verification error flag In receive mode, if a parity error occurs, the hardware bits the location. It is cleared by the software sequence (reading USART_SR and USART_DR sequentially). Before clearing the PE bit, the software must wait for the RXNE flag bit to be referred to 1, and generate an interrupt if the PEIE in USART_CR1 is 1. 0: No parity error; 1: Parity error.

37.4.2 Send Data Register (DR)

Address offset: 0x04

Reset value: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	DR [8: 0]								
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	DR [8: 0]	RW	0	Contains the data characters to be transferred. When parity is enabled (the PCE bit is set to 1 in the USART_CR1 register), the value written in the MSB (bit 7 or bit 8 depending on the data length) has no effect because it is replaced by parity

37.4.3 Baud rate register (BRR)

Address offset: 0x08

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV_Mantissa [11: 0]												DIV_Faction [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 4	DIV_Mantissa [11: 0]	RW	12'h0	12-bit integer These 12 bits are used to define the integer of the USART divisor (USARTDIV)
3: 0	DIV_Fraction [3: 0]	RW	4'h0	4bit decimal These 4 bits are used to define the decimal of the USART divisor (USARTDIV).

37.4.4 USART control register 1 (CR1)

Address offset: 0x0c

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIFOEN	TXFRQ	RXFRQ	Res	DATAINV	TX_INV	RX_INV	RXFTCFG [2: 0]			TXFTCFG [2: 0]			SWAP	ADDM7	Res
RW	W	W	-	RW	RW	RW	RW	RW	RW	RW			RW	RW	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB	M1	UE	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
RW															

Bit	Name	R/W	Reset Value	Function
31	FIFOEN	RW	0	fifo mode enabled: 0: FIFO mode off 1: FIFO mode turned on FIFO mode can be used for standard UART communication, SPI master mode and smart card mode. It cannot be enabled in IrDA and LIN modes. This bit can only be configured when USART is not enabled (UE = 0)
30	TXFRQ	W	0	TXFRQ: Send data refresh request

				When FIFO mode is disabled, writing "1" to this bit sets the TXE flag to 1. This may discard the data. This bit can only be used in smart card mode when no data is sent due to an error (NACK) and the FE flag in the USART_SR register is valid. If USART does not support smart card mode, this bit is reserved and must maintain the reset value. When FIFO is enabled, TXFRQ position 1 to clear the entire FIFO. This sets the TXFE flag. Clear sending FIFO is supported in both UART mode and smart card mode.
29	RXFRQ	W	0	Receive data refresh request Write 1 to clear the entire received FIFO. This enable does not read data but will discard all FIFO received data to avoid overload conditions
28	Reserved	-	-	-
27	DATAINV	RW	0	Binary data flip 0: 1 = H 0 = L 1: 1 = L 0 = H The parity bit is also reverse
26	TX_INV	RW	0	TX Pin Active Level Reversal This bit is set and cleared by the software. 0: The TX pin signal operates using standard logic level (VCC = 1/idle, Gnd = 0/flag) 1: TX pin signal value inverted (VCC = 0/flag, Gnd = 1/idle). This makes it possible to use an external inverter on the TX line. This bit can only be configured when USART is not enabled (UE = 0)
25	RX_INV	RW	0	RXINV: RX Pin Active Level Reverse This bit is set and cleared by software. 0: RX pin signal operates using standard logic level (VCC = 1/idle, Gnd = 0/flag) 1: RX pin signal value inverted (VCC = 0/flag, Gnd = 1/idle). This makes it possible to use an external inverter on the RX line. This bit can only be configured when USART is not enabled (UE = 0).
22: 24	RXFTCFG [2: 0]	RW	3'b0	000: 1/8 of the received FIFO arrival depth 001: 1/4 of FIFO arrival depth received 010: Receive 1/2 of FIFO arrival depth 011: 3/4 of FIFO arrival depth received 100: 7/8 of receive FIFO arrival depth 101: Received FIFO full remaining combination: reserved
19: 21	TXFTCFG [2: 0]	RW	3'b0	XFTCFG [2: 0]: TXFIFO threshold configuration 000: TXFIFO to 1/8 of its depth 001: TXFIFO reaches 1/4 its depth 010: TXFIFO reaches 1/2 of its depth 011: TXFIFO reaches 3/4 of its depth 100: TXFIFO to 7/8 of its depth 101: TXFIFO becomes empty
18	SWAP	RW	1'b0	0: Use the TX/RX pins defined in the standard pins 1: Swap TX and RX pin functions. This makes it possible to work with a cross-connection with another UART. This bit can only be configured when USART is not enabled (UE = 0)
17	ADDM7	RW	1'b0	7-bit address detection/4-bit address detection This bit is used to select a 4-bit address detection or a 7-bit address detection. 0: 4-bit address detection 1: 7-bit address detection (in 8-bit data mode) This bit can only be configured when USART is not enabled (UE = 0)
16	~~Reserved~~	~~-~~	~~-~~	~~-~~
15	MSB	RW	0	0: When transmitting/receiving data, data bit 0 is after the start bit. 1: Data is first transmitted/received through the MSB (bit 7/8), after the start bit. This bit can only be configured when USART is not enabled (UE = 0)

14	M1	RW	0	This and the 12th bit M0 compose M to determine the data size, M: 00: 8 bits 01: 9-bit 10: 7-bit This bit can only be configured when USART is not enabled (UE = 0) In 7-bit data length mode, smart card mode and LIN main mode are not supported.
13	UE	RW	0	UE: USART enabled When this bit is cleared, the divider and output of USART stop working after the current byte transmission is completed, thereby reducing power consumption. This bit is set and cleared by software. 0: USART divider and output disabled; 1: USART module enabled.
12	M0	RW	0	M0: word length This bit defines the length of the data word, which is set and cleared by software. This bit can only be configured when USART is not enabled (UE = 0)
11	WAKE	RW	0	Wake: Ways to Awaken This bit determines the method of waking up USART, which is set and cleared by software. 0: Wake up by idle bus; 1: Wake up by address mark. This bit can only be configured when USART is not enabled (UE = 0)
10	PCE	RW	0	PCE: Verification Control Enabled This bit is used to select whether to perform hardware verification control (for sending, it is the generation of check bits; for receiving, it is the detection of check bits). When this bit is enabled, a check bit is inserted into the highest bit of the transmitted data (if M = 1, the highest bit is the 9th bit; if M = 0, the highest bit is the 8th bit); The received data is checked for its check bits. The software sets "1" or clears "0" for it. Once this bit is set, the check control takes effect until the current byte transfer is complete. 0: prohibit verification control 1: Enable verification control. This bit can only be configured when USART is not enabled (UE = 0)
9	PS	RW	0	PS: Verify selection When the check control is enabled, this bit is used to select whether to use even or odd checks. The software sets "1" or clears "0" on it. The selection takes effect after the current byte transfer is complete. 0: even check; 1: Odd check. This bit can only be configured when USART is not enabled (UE = 0)
8	PEIE	RW	0	PEIE: PE interrupt enable 0: Prohibit generation of interrupt 1: PE interrupt enabled
7	TXEIE	RW	0	TXE: TXE interrupt enable 0: Interrupt is disabled 1: TXE interrupt enabled
6	TCIE	RW	0	TCIE: Send Complete Interrupt Enable 0: Generation of interrupt is prohibited. 1: When ORE or RXNE in USART_SR is 1, a USART interrupt is generated.
5	RXNEIE	RW	0	RXNEIE: Receive buffer non-empty interrupt enabled 0: Prohibit generation of interrupt 1: When ORE or RXNE in USART_SR is "1", a USART interrupt is generated.
4	IDLEIE	RW	0	IDLE: IDLE interrupt enable 0: Prohibit generation of interrupt; 1: When IDLE in USART_SR is "1", USART interrupt is generated

3	TE	RW	0	TE: transmit enabled This bit enables the transmitter. 0: transmission prohibited 1: Enable transmission. During data transmission, except in smart card mode, if there is a 0 pulse on the TE bit (that is, set to '0' and then set to '1'), a "preamble" (idle frame) will be sent after the transmission of the current data word is completed. When the TE is set, there is a delay of one bit before the actual transmission begins.
2	RE	RW	0	RE: Receive enabled 0: Reception is prohibited; 1: Enable receive and start searching for the start bit of the RX pin.
1	RWU	RW	0	RWU: Receive Wake-up This bit is used to determine whether to put USART in silent mode. This bit is set or cleared by the software. When the wake-up sequence arrives, the hardware will also clear it. 0: The receiver is in normal operation mode; 1: The receiver is in silent mode. Before putting the USART in silent mode (setting the RWU bit), the USART has received a data byte. Otherwise, in silent mode, it cannot be awakened by idle bus detection. When configured for address flag detection WAKE-up (WAKE bit = 1), the RWU bit cannot be modified by software while the RXNE bit is set
0	SBK	RW	0	SBK: transmit disconnected frame Software sets this register to send a disconnect frame. After the stop bit of the disconnected frame is transmitted, the hardware clears this register. 0: No disconnected frame is transmitted; 1: Transmit a disconnected frame.

37.4.5 USART control register 2 (CR2)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	RTOEN	DEP	DEM	ADD [7: 1]						
-						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD[0]	LINEN	STOP [1: 0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	Res				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-				

Bit	Name	R/W	Reset Value	Function
31: 26	Reserved	-	-	-
25	RTOEN	RW	1’b0	Receiver timeout enable This bit is set to 1 and cleared by the software. 0: Disable receiver timeout function. 1: Enable the receiver timeout function. When this function is enabled, if the RX line is idle for the duration programmed in the RTOR (Receiver Timeout Register (no receive), the RTOF flag in the USART_SR register is set to 1.
24	DEP	RW	1’b0	Drive enable polarity selection 0: DE signal is high 1: DE signal is low This bit can only be configured when USART is not enabled (UE = 0).
23	DEM	RW	1’b0	Drive enable mode: 0: DE forbidden 1: DE enabled

				This bit can only be configured when USART is not enabled (UE = 0)
22: 15	ADD [7: 0]	RW	8'h0	USART address Address The address at the time of flag detection. In 4-bit address detection, only ADD [3: 0] is used This bit can only be configured when USART is not enabled (UE = 0)
14	LINEN	RW	0	LIN mode enable. 0: LIN mode off; 1: LIN mode off; In LIN mode, by enabling the SBK bit, a LIN synchronous break frame (13 bits) is transmitted, and a break frame is detected. This bit can only be configured when USART is not enabled (UE = 0).
13: 12	STOP [1: 0]	RW	2'b0	Stop bits 00: 1 stop bit; 01: 0.5 stop bits; 10: 2 stop bits; 11: 1.5 stop bits; This bit can only be configured when USART is not enabled (UE = 0)
11	CLKEN	RW	0	Clock enable, this bit is used to enable the CK pin. 0: CK pin disabled 1: CK pin enable; This bit can only be configured when USART is not enabled (UE = 0)
10	CPOL	RW	0	Clock polarity 0: Steady low value on CK pin when the bus is idle; 1: Steady high value on CK pin when the bus is idle; This bit can only be configured when USART is not enabled (UE = 0)
9	CPHA	RW	0	Clock phase 0: data capture at first edge of clock; 1: Data capture at the second edge of the clock. This bit can only be configured when USART is not enabled (UE = 0)
8	LBCL	RW	0	Last bit clock pulse In synchronous mode, this bit is used to control whether the clock pulse corresponding to the last sent byte (MSB) is output on the CK pin. 0: The clock pulse of the last bit of data is not output at CK pin; 1: The clock pulse of the last bit of data is output at CK pin. Note: The last data bit is the eighth or ninth transmitted bit (determined by the M bit in USART_CR1) This bit can only be configured when USART is not enabled (UE = 0)
7	Reserved	-	-	-
6	LBDIE	RW	0	LIN disconnect frame interrupt enabled. 0: Interrupt is disabled 1: generation;
5	LBDL	RW	0	LIN disconnect frame detection length. 0: Detect 10-bit broken frame; 1: Detect 11-bit broken frame;
4: 0	Reserved	-	-	-

37.4.6 USART control register 3 (CR3)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	EOBIE	RTOIE	DEAT [4: 0]					DEDT [4: 0]					FXTFIE	RXFTIE	TXFEIE
-	RW														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFFI	ABRM	Res	ABRE	OVE	CTSI	CTS	RTS	DMA	DMA	SCE	NAC	HDS	IRLP	IREN	EIE

E	OD		N	R8	E	E	E	T	R	N	K	EL			
RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30	EOBIE	RW	0	EOB interrupt enable 0: Interrupt is disabled 1: interrupt enable;
29	RTOIE	RW	0	RTO interrupt enable 0: Interrupt is disabled 1: interrupt enable;
28: 24	DEAT [4: 0]	RW	5'h0	The time from the start bit to the DE pull up signal. Expressed in sample time (1/8 bit or 1/16 bit time, depending on the oversampling configuration) This bit can only be configured when USART is not enabled (UE = 0)
23: 19	DEDT [4: 0]	RW	5'h0	Time of the stop bit of the last data to DE. Expressed in sample time (1/8 bit or 1/16 bit time, depending on the oversampling configuration) This bit can only be configured when USART is not enabled (UE = 0)
18	TXFTIE	RW	1'b0	TXFIFO Threshold Interrupt Enable 0: Disable interrupt 1: Enable interrupt
17	RXFTIE	RW	1'b0	RXFTIE: RXFIFO threshold interrupt enable 0: Disable interrupt 1: Enable interrupt
16	TXFEIE	RW	1'b0	TXFEIE: TXFIFO null interrupt enable 0: Disable interrupt 1: Enable interrupt
15	RXFFIE	RW	1'b0	RXFFIE: RXFIFO full interrupt enable 0: Disable interrupt 1: Enable interrupt
14	ABRMOD	RW	0	Auto baud rate mode 0: Measure baud rate from start bit; 1: falling edge to falling edge measurement;
13	Reserved	-	-	-
12	ABREN	RW	0	Auto baud rate enable 0: Interrupt is disabled 1: Automatic baud rate enable; This bit can only be configured when USART is not enabled (UE = 0)
11	OVER8	RW	0	Oversampling mode 0: 16-bit oversampling; 1: 8-bit oversampling; This bit can only be configured when USART is not enabled (UE = 0)
10	CTSIE	RW	0	CTS interrupt enable 0: Disable interrupt 1: Enable interrupt
9	CTSE	RW	0	CTS enable 0: CTS hardware flow control disabled 1: CTS mode enabled Data is only transferred when the CTS input is 0. If a data is written into the data register while CTS is deasserted, the transmission is postponed until CTS is asserted. This bit can only be configured when USART is not enabled (UE = 0)
8	RTSE	RW	0	RTS enable 0: RTS hardware flow control disabled 1: RTS output enabled, The next data will only be requested if the receive buffer is not full. The transmission of data is expected to cease

				after the current character has been transmitted. If data can be received, set RTS to valid (0). This bit can only be configured when USART is not enabled (UE = 0)
7	DMAT	RW	0	DMA enables send. 0: DMA transmission is prohibited 1: enable transmission DMA;
6	DMAR	RW	0	DMA enables reception. 0: inhibit reception of DMA; 1: Enable receive DMA;
5	SCEN	RW	0	Smartcard mode enable. 0: Interrupt is disabled 1: Enabled This bit can only be configured when USART is not enabled (UE = 0)
4	NACK	RW	0	Smartcard NACK enable 0: NACK transmission in case of parity error is disabled; 1: NACK transmission during parity error is enabled;
3	HDSEL	RW	0	Half-duplex selection 0: Half duplex mode is not selected 1: half-duplex mode This bit can only be configured when USART is not enabled (UE = 0)
2	IRLP	RW	0	IrDA low power 0: Normal mode; 1: Low power mode; This bit can only be configured when USART is not enabled (UE = 0)
1	IREN	RW	0	IrDA mode enable This bit is set and cleared by software. 0: IrDA disabled; 1: IrDA enabled; This bit can only be configured when USART is not enabled (UE = 0)
0	EIE	RW	0	Error interrupt enable 0: Interrupt is disabled 1: frame error (FE), overrun error (ORE), noise (NE) interrupt enable;

37.4.7 Guard Time and Prescalation (GTPR)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GT [7: 0]								PSC [7: 0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 8	GT [7: 0]	RW	8'h0	Guard time value Guard time in Botter clock. This is used in Smartcard mode. Note: This bit can only be configured when USART is not enabled (UE = 0)
7: 0	PSC [7: 0]	RW	8'h0	Prescaler value In infrared (IrDA) low power mode: PSC [7: 0] = infrared low power baud rate. Used for programming the prescaler for dividing the system clock to achieve the low-power frequency: The source clock is divided by the value in the register (8 bits valid) 00000000 Res

				00000001: divides the source clock by 1; 00000010: divides source clock by 2; …… 1. In infrared (IrDA) normal mode: PSC can only be set to 00000001. 2. In smartcard mode: PSC [4: 0]: prescaler value Used for programming the prescaler for dividing the system clock to provide the smartcard clock. The value given in the register (the lower 5 bits are valid) is multiplied by 2 as the frequency division factor for the source clock. 00000: Res 00001: divides the source clock by 2; 00010: divides the source clock by 4; 00011: divides the source clock by 6; …… Note: Bit [7: 5] does not make sense in smart card mode. Note: This bit can only be configured when USART is not enabled (UE = 0)

37.4.8 Receive Timeout Register (RTOR)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLEN [7: 0]								RTO [23:16]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTO [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 24	BLEN [7: 0]	RW	8'h0	Block length: This is the block length received by T = 1 in smart card mode
23: 0	RTO [23: 0]	RW	24'h0	RX receive timeout In the standard mode, no start bit is detected within a set time after the last reached character is received, and the RTOF flag position starts. In smart card mode, this value is used to perform CWT and BWT. In this case, the CWT and BWT are calculated from the last received character start bit

RTOR can be written dynamically. If the new value is less than or equal to the counter, the RTOF flag is set.

38. Universal Asynchronous Receivers/Transmitter (UART)

38.1 Introduction

UART is a programmable universal asynchronous transceiver.

38.2 Main Features

- AMBA APB interface
- Support 5/6/7/8/9-bit serial data
- Supports 1/2 stop bits (for 5-bit data: 1/1. 5 stop bits)
- Support sending address/data
- Support fixed parity check
- Supports disconnected frames
- Detect start bit errors
- Support programmable fractional baud rates
- Support SWAP function
- Support MSB FIRST endianness switching
- Support DMA transmission
- Support 4-bit fractional baud

38.3 UART functional description

38.3.1 UART (RS232) serial protocol

Because the serial communication between the UART and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the start and end. These bits can be used to synchronize the two devices. This serial data structure consisting of start bits and stop bits is called a character, as shown in the following figure.

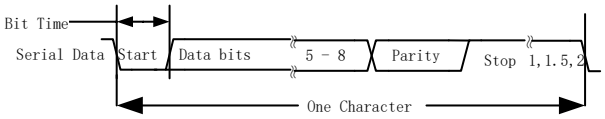


Figure 38-1 serial data format

An additional parity bit can be added to the serial character. This bit occurs after the last data bit in the character structure and before the stop bit in order to provide the UART with the ability to perform error checking on the received data.

The UART Line Control Register ("CR1" section in the Register Description) is used to control the serial character characteristics. The individual bits of the data word are transmitted after the start bit, starting with the least significant bit (LSB). They are followed by an optional parity bit followed by a stop bit, which can be 1, 1.5, or 2.

Note: The stop bit duration for UART implementations may be longer for the following reasons:

- Idle time inserted between some configured characters

All bits in the transmission are transmitted for exactly the same duration; The exception to this is the half stop bit when 1.5 stop bits are used. This duration is referred to as a bit period or bit time; One bit of time is equal to sixteen baud clocks.

To ensure line stability, once the start bit is detected, the receiver samples the serial input data at approximately the midpoint of the bit time.

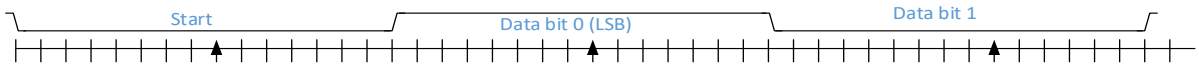


Figure 38-2 receiver serial data sampling points

As part of the 16550 standard, an optional baud clock reference output signal provides timing information for receiving devices that require it. The baud rate of the UART is controlled by a serial clock sclk or pclk in a single-clock implementation and a frequency division latch register (BRR).

The following figure showsthe timing diagram ofthe baudout_noutput for different divisor values.

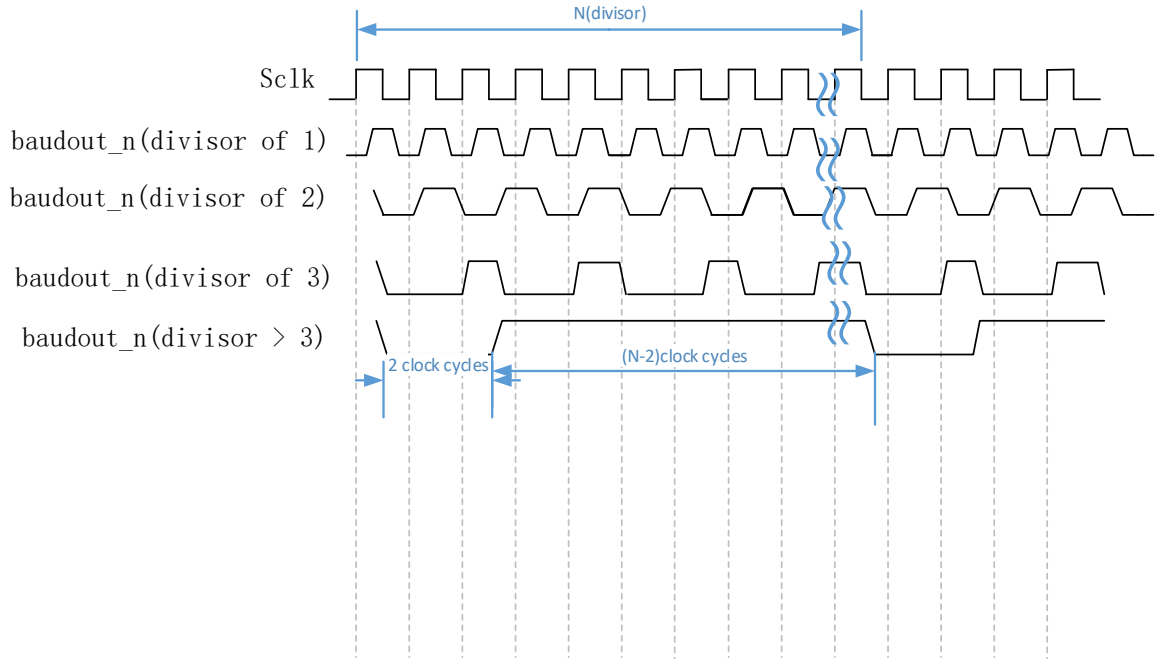


Figure 38-3 timing diagram

38.3.2 9-bit data transfer

The UART may be configured to have 9-bit data transmission in both transmit and receive modes. The 9th bit in the character occurs after the 8th bit of the character and before the parity bit. The figure below shows the serial transfer of characters, where D8 denotes the 9th bit, and also shows the general serial transfer in 9-bit mode. (This is the normal little-endian mode. If it is the big-endian mode, the order of 9 data bits will be switched)

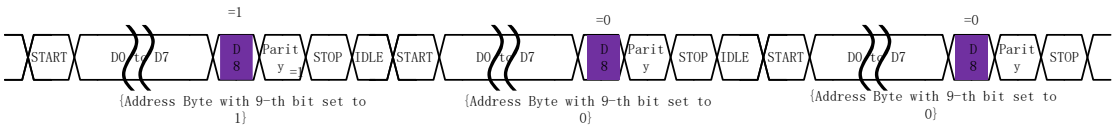


Figure 38-4 9 Bit data transmission

By enabling 9-bit data transfer mode, UART can be used in multipoint systems where one master device is connected to multiple slave devices in the system. The master communicates with one of the

slaves. When the master device wants to transfer a block of data to a slave device, it first sends an address byte to identify the target slave device.

The distinction between address/data bytes is done based on the 9th bit in the input character. If the 9th bit is set to 0, the character represents a data byte. If the 9th bit is set to 1, the character represents the address byte. All slave systems compare the address bytes to their own addresses, and only the target slave system (where the addresses match) is able to receive data from the master system. The master starts sending data bytes to the target slave. The unaddressed slave system ignores incoming data until a new address byte is received.

Note an address followed by 2 data bytes. The address byte is output with the 9th bit (D8) set to 1, while the data byte is emitted with the 9th bit (D8) set to 0. The parity bit is an optional field.

The configuration of the UART for 9-bit data transmission does the following:

- The CR3 [0] bit is used to enable or disable 9-bit data transfer.
- In the case of reception, the CR1 [1] bit is used to select between hardware and software based address matching.
- The CR3 [2] bit is used to enable the transmit address in the case of transmission.
- The CR3 [3] bit is used to select between hardware and software based address transfer.
- The TAR and RAR registers are used to send the address and match the received address, respectively.
- The TDR and RDR registers are 9-bit registers for data transfer in 9-bit mode.
- The SR [8] bit is used to indicate an address reception interrupt.

Send Mode

UART supports two types of transmission modes:

- Transmission Mode 0 (when (CR3 [3]) is set to 0)
- Transmission Mode 1 (when (CR3 [3]) is set to 1)

Transmission Mode 0

In transfer mode 0, addresses are programmed in the transfer address register (TAR) and data is written to the transfer hold register (TDR). The 9th bit of the TDR register is not applicable in this mode. The figure below showsaddress and data transmissionbased on SEND_ADDR(CR3 [2]),TDRNULL conditions.

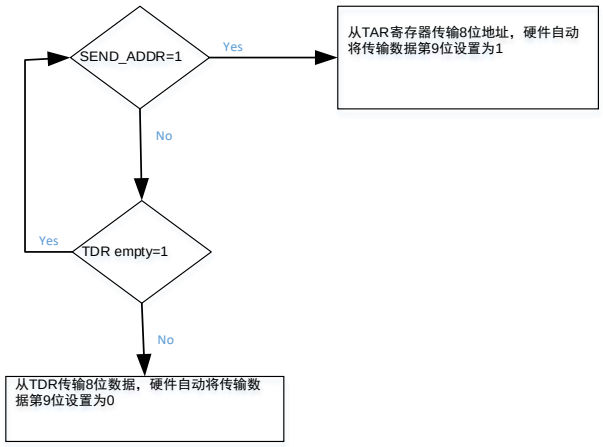


Figure 38-5 Automatic address transfer flow chart

The address of the destination slave to which data is to be transferred is programmed in the TAR register. The SEND_ADDR(CR3 [2]) bitmust be enabled to transfer the destination slave address in the TAR registeronthe serial UART line, where the9thdata bit is set to1, indicating that the address is being sent to the slave. UARTclearsthe SEND_ADDR bitafter the address character begins to be transmitted onthe UART line.

The data needed to be transferred to the target slave is programmed through a transfer hold register (TDR). Data is transmitted on the UART line, and the 9th data bit is set to 0, indicating that data is being sent to the slave.

Transmission Mode 1

In Transfer Mode 1, the TDR register is 9 bits wide and both addresses and data are programmed through the TDR register. UART does not distinguish between address and data. The SEND_ADDR(CR3 [2]) bit and the Transfer Address Register (TAR) are not applicable for this mode. Depending on whether address/data must be sent, the software must write the 9th bit with 1/0.

Table 38-1 Transfer Relationship Table

M_E	TX_MODE	SEND_ADDR	Usage scenarios
0	X	X	Send 8-bit TDR data
1	0	0	Send "0 +8-bit TDR data"
1	0	1	Send "1 +8-bit TAR address"
1	1	X	Send "9-bit TDR data"

Note: This table describes several sending scenarios and corresponding configurations that can be used as a sender.

Receiving Mode

UART supports two reception modes:

■Hardware address match receive mode (whenADDR_Match(CR3 [1]) is set to1)

■Software Address Match Receive Mode (whenADDR_Match(CR3 [1]) is set to0)

Hardware address matching receiving mode

In hardware address matching receive mode, if the 9th bit of the receive character is set to 1, the UART matches the receive character to the address programmed in the receive address register (RAR):

If the received address successfully matches the programmed address in the RAR register, the data byte is then received.

If the address match fails, the UART controller discards the data characters until a matching address is received.

The following figure shows the data byte receiving flow chart based on the address matching function.

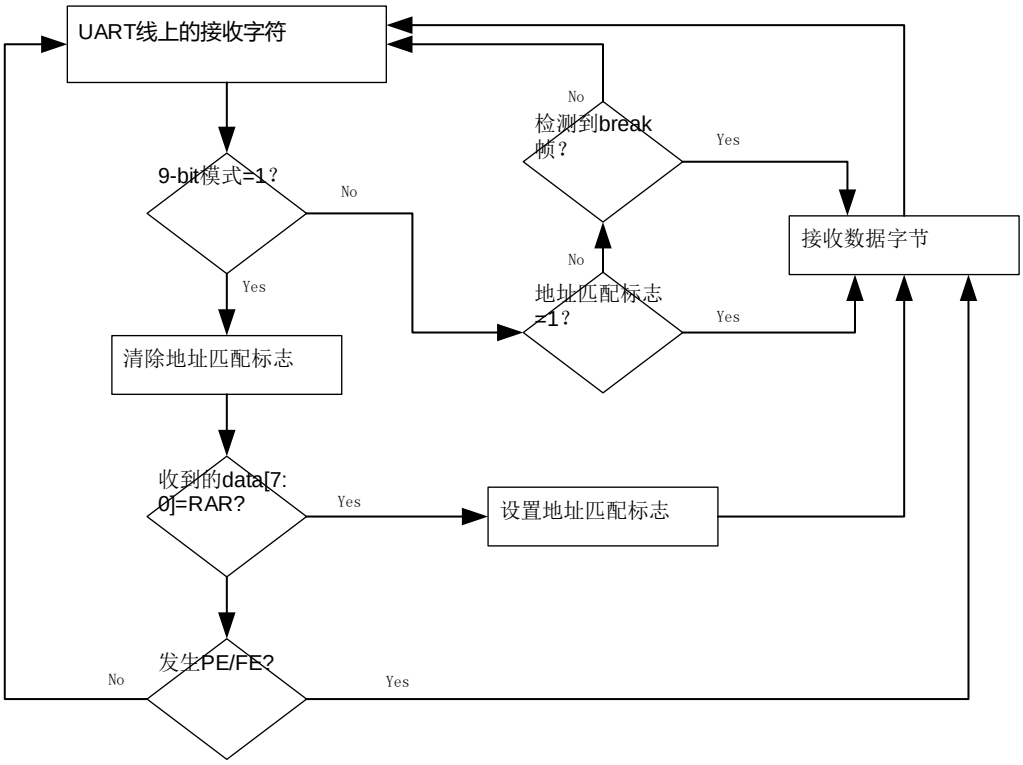


Figure38-6 Data byte receiving flow chart of address matching function

The UART receives characters regardless of whether the 9th bit of data is set to 1. If the 9th bit of the received character is set to 1, it clears the internal address match flag and then compares the received 8-bit character information with the address programmed in the RAR register.

If the received address charactersuccessfully matches the address programmed inthe RAR register, the address match flag is set to 1and the received characteris pushed to the RDR register, and the ADDR_RCVD bitinthe SR registeris set to indicate that the address has been received, andthe subsequent data byte (the 9thbit of the received character is set to0) is pushed tothe RDR.

If the address fails to matchthe RAR registerand a frame error is found in the parity or received address character, the received address string is still pushed tothe RDR register, andthe ADDR_RCVD and PE/FE error bits are set to1.

If any interrupt character is received, UART treats it as a special character and pushes it to the RDR register regardless of the address match flag.

Software address matching receiving mode

In this mode of operation, the UART does not perform address matching to the received address character of the RAR register (the 9th bit of data is set to 1). The UART always receives 9 bits of data, advancing the RDR register. Whenever an address byte is received and indicated by the ADDR_RCVD bitin the line status register, the address must be compared by the user himself.

38.3.3 Baud rate

The baud rate of the UART is controlled by pclk and the divided latch register (BRR) and the fractional baud rate register BRRF.

The baud rate is determined by:

- Serial clock operating frequency (pclk)

- Baud rate generator divisor (consists of BRR register)
- Acceptable baud rate error

The equation for calculating the baud rate is as follows:

Baud rate = serial clock operating frequency/(16 * DIVISOR)--(1)

Where, DIVISOR-the number (hex) used to program the BRR.

Serial Clock Frequency-The frequency at the UART pclk pin.

According to equation (1), DIVISOR can be calculated as: DIVISOR = serial clock frequency/(16 * baud rate)

(The fractional part calculated by the equation is placed in BRRF)

Also from Equation (1), it can also be shown that serial clock frequency = baud rate * 16 * divisor

The error between the baud rate and the selected baud rate calibration is as follows:

Percentage error = (| baud rate-selected baud rate |)/baud rate * 100%

Table 38-2 Error calculation when setting baud rate

Baud rate		**fpclk = 4 MHZ**			**fpclk = 24 MHZ**			**fpclk = 48 MHZ**		
No.	**kbps**	**Actual**	**Value programmed in the baud rate register**	**Error%**	**Actual**	**Value programmed in the baud rate register**	**Error%**	**Actual**	**Value programmed in the baud rate register**	**Error%**
1	2.4	2.404	104	0.16 %	2.4	625	0.00 %	2.4	1250	0.00 %
2	9.6	9.615	26	0.16 %	9.615	156	0.16 %	9.615	312	0.16 %
3	19.2	19.231	13	0.16 %	19.231	78	0.16 %	19.231	156	0.16 %
4	57.6	62.5	4	8.51 %	57.692	26	0.16 %	57.692	52	0.16 %
5	115.2	125	2	8.51 %	115.385	13	0.16 %	115.385	26	0.16 %
6	230.4	250	1	8.51 %	250	6	8.51 %	230.769	13	0.16 %
7	460.8	Impossible	Impossible	Impossible	500	3	8.51 %	500	6	8.51 %
8	921.6	Impossible	Impossible	Impossible	1500	1	62.76 %	1000	3	8.51 %
9	2250	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	3000	1	33.33 %
10	4500	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible	Impossible

Notes:

1. The lower the clock frequency of the CPU, the lower the error at a particular baud rate. The upper limit of the achievable baud rate can be fixed with this data.

2. When configuring the clock, because the baud rate clock is divided on the system clock, there will be an error between the obtained clock frequency and the actual clock frequency. The configured clock error needs to be within 2% to work properly.

38.3.4 The UART receiver tolerates changes in the clock

The UART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the UART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

Need to meet: DTRA + DQUANT + DREC + DTCL < tolerance of UART receiver

Table 38-3 UART receiver tolerance when DIV_Fraction is 0

M bit	NF is an error		NF doesn't care	
5	5.66%	-5.71%	6.67%	-6.74%
6	4.92%	-4.96%	5.66%	5.71%
7	4.35%	-4.38%	4.92%	4.96%
8	3.59%	-5.85%	3.90%	-5.91%

Table 38-4 UART receiver tolerance when DIV_Fraction is not 0

M bit	NF is an error		NF doesn't care	
5	4.72%	-4.76%	5.56%	-5.62%
6	4.92%	-4.13%	4.72%	-4.76%
7	4.35%	-3.65%	4.10%	-4.13%
8	4.53%	-5.2%	4.58%	-5.31%

38.3.5 Continuous Communication Using DMA

The UART may utilize DMA continuous communication. DMA requests for the Rx buffer and the Tx buffer are generated separately.

Transmission using DMA

Transmission with DMAis enabledby configuringUART_CR3. DMAT. When the send data register is empty, the DMA transfers datafrom the designatedSRAM areatothe UART_DR register.

The configuration steps of DMA transmission are as follows:

- Configure the UART_ TDRregister address asthe destination address for the DMA transfer. After each send data register empty event, data will be transferred to this address;
- Configure a memory (e.g. SRAM) address as the source address for DMA transfers. Loading read data from this memory into the UART _TDR registerafter each transmit data register empty event;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

Reception using DMA

Utilizing DMA receptionis enabledby configuring UART_CR3.DMAR. For each byte received, the DMA transfers datafromthe UART_DR registerto the designatedSRAM area.

The configuration steps of DMA reception are as follows:

- Configure the UART_ RDRregister address asthe source address for the DMA transfer. After each received data register non-empty event, data is transferred to memory by reading datafrom this UART_RDR register address;
- Configure the SRAM memory address as the destination address for the DMA transfer. After each reception of a data register non-empty event, data will be transferred from the UART_RDR registerto this address;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

38.3.6 Interrupts and events

Assertion of UART Interrupt Output Signal (intr)-An interrupt occurs as soon as one of several preferential interrupt types is enabled and activated.

When an interrupt occurs, the host may access the IIDR register to determine the interrupt type.

The following interrupt types can be enabled via the CR2 register:

- Receiver error
- Receiver data available
- Send hold register empty (in programmable TDR interrupt mode)
- Busy detection indication

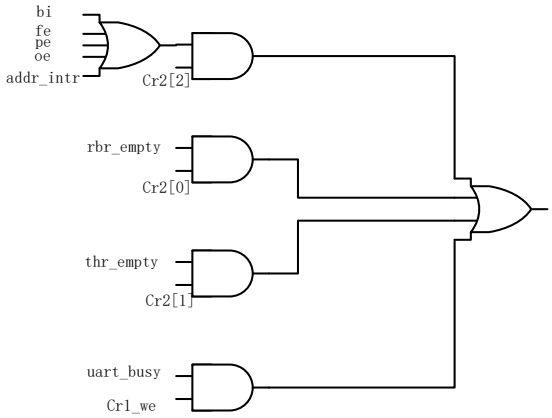


Figure 38-7 USART interrupt mapping diagram

These interrupt types are detailed in the table below.

Table 38-5 interrupt control function

Interrupt ID				Interrupt Set and Reset Functions			
Bit 3	**Bit 2**	**Bit 1**	**Bit 0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0	0	0	1	–	None	None	–
0	1	1	0	Highest	Receiver line status	Overflow/Parity/Frame Error, Interrupt or Address Receive Interrupt	Write 1 to clear the corresponding bit
0	1	0	0	Second	Received data available	Receiver data available	Read receiver buffer register
0	0	1	0	Third	Transmit holding register empty	Send hold register empty	Write TDR
0	1	1	1	Fifth	Busy detect indication	When UART is BUSY (BUSY [0] is set to 1), master tries to write to the CR1 register.	Write 1 to clear the corresponding bit

38.4 TIMx registers

The peripheral registers have to be accessed by half-words (16 bits) or words (32 bits).

Terms: set = set to 1; Clear = Clear is 0;

38.4.1 UART DR, Address = 0X00 (DR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	DR								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 9	Re-served	-	-	-
8: 0	DR	RW	9'b0	Data value The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).

38.4.2 UART BRR, Address = 0X04 (BRR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRR															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 0	BRR	RW	16'b0	This register constitutes a 16-bit divisor, which contains the baud rate divisor of UART. The output baud rate is equal to the serial clock (pclk) frequency divided by sixteen times the baud rate divisor, as follows: baud rate = (serial clock frequency)/(16 * divisor). Note that when the Divisor Latch Register (BRR) is set to zero, the Baud clock is disabled and no serial communication occurs. Furthermore, once the BRR is set, you should wait for at least 8 clock cycles to pass before sending or receiving data.

38.4.3 UART SR, Address = 0X10 (SR)

Address offset: 0x10

Reset value: 0x00000060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	BUSY_ERR	BUSY	ADDR_RCVD	Res	TXE	TDRE	BRI	FE	PE	ORE	RXNE
-					RC_W1	R	RC_W1	-	R	R	RC_W1				R

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
10	BUSY_ERR	RC_W1	1'b0	The misoperation flag bit when busy is detected. Error Flag: Attempt to write to CR1 register when UART is busy. 0: No busy misoperation error. 1: When UART is busy, try to write to the CR1 register. Write 1 to clear this bit
9	BUSY	R	1'b0	UART busy flag bit. This indicates that a serial transfer is in progress, and when cleared it indicates that the UART is idle or inactive. This bit will be set to 1 (busy) in either of the following cases: -Transfer ongoing on serial interface -Receiving on serial interface -Transmission of data present in TDR with non-zero baud divisor (BRR not equal to 0) -Receive data presence in RDR Note: The UART busy bit may be cleared even if a new character may have been sent from another device. That is, if the UART

				has no data in the TDR and RDR, and there is no ongoing transmission, and the start bit of the new character has just arrived at the UART. This is because a valid start is not seen until the middle of the bit period, and this duration depends on the programmed baud divisor. 0: UART is idle or inactive 1: UART busy
8	ADDR_RCVD	RC_W1	1'b0	Address receive. If the 9-bit data mode is enabled, this bit is used to indicate that the 9th bit of received data is set to 1. This bit can also be used to indicate whether the input character is an address or data. 0: Incoming character is data 1: The incoming character is the address
7	Reserved	-	-	-
6	TXE	R	1'b1	Send is empty. This bit is set whenever no data is currently being sent and the TDR is empty. 0: Send is not empty 1: Send is empty
5	TDRE	R	1'b1	Send keeps register empty. This bit indicates that the TDR is null. This bit is set whenever data is transferred from the TDR to the transmitter shift register and no new data is written to the TDR. This also causes a TDR interrupt to occur if it is enabled. 0: TDR non-empty 1: TDR is empty
4	BRI	RC_W1	1'b0	The break interrupt bit. This is used to indicate that an interrupt sequence is detected on the serial input data. If in UART mode, it is set whenever the serial input sin remains in a logic "0" state for longer than the sum of the start time + data bits + parity bits + stop bits. 0: No break sequence detected 1: break sequence detected Write 1 to clear this bit
3	FE	RC_W1	1'b0	Frame error bit. This is used to indicate that a frame error has occurred in the receiver. A frame error occurs when the receiver does not detect a valid STOP bit in the received data. It should be noted that if a break occurs, the frame error bit will also be set, as shown by the break interrupt bit. This happens because the break character implicitly generates a frame error by holding the sin input to a logical 0 longer than the duration of the character. 0: No frame error 1: Frame error Write 1 to clear this bit
2	PE	RC_W1	1'b0	Parity error bit. If a parity enable bit is set, this is used to indicate the occurrence of a parity error in the receiver. It should be noted that if a break occurs, in which case PE is also set if parity generation and detection is enabled and parity is set to odd. 0: No parity error 1: Parity error Write 1 to clear this bit
1	ORE	RC_W1	1'b0	Overflow error bit. This is used to indicate the occurrence of an overflow error. This happens if a new data character is received before the previous data is read. The ORE bit is set when a new character reaches the receiver before the previous character is read from the RDR. When this happens, the data in the RDR is overwritten. 0: No overflow error 1: Overflow error Write 1 to clear this bit
0	RXNE	R	1'b0	Data ready bit. This is used to indicate that the receiver includes at least one character in the RDR. This bit is cleared when the RDR is read. 0: Data not ready 1: Data ready

38.4.4 UART CR1, Address = 0X14 (CR1)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	MSBFIRST	SWAP	Res	SBK	SP	PS	PCE	STOP	M	
-						RW	RW	-	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	-
9	MSBFIRST	RW	1'b0	MSB Priority. 0: After the start bit, the 0th bit data is transmitted and received; 1: After the start bit, send and receive the 5/6/7/8/9th bits of data; This bit cannot be modified during data transfer.
8	SWAP	RW	1'b0	Interchangeable Tx/Rx pins 0: TX/RX pins are not interchangeable; 1: TX/RX pin interchange; When used as cross-wired to connect to other UARTs.
7	Reserved	-	-	-
6	SBK	RW	1'b0	The break control bit. This is used to send the break to the receiving device. If set to 1, the serial output will be forced into an interval (logic 0) state. The sout line is forced low until the SBK bit is cleared. 0: Release serial output for data transfer 1: Serial output is forced into interval state
5	SP	RW	1'b0	Fixed parity. Writable only if UART is not busy. This bit is used to force a fixed parity value. When PCE, PS, and SP are set to 1, a parity bit is sent and checked for logical 0. If PCE and Stick Parity are set to 1, and PS is a logical 0, a parity bit is sent and checked for logical 1. If this bit is set to 0, then fixed parity is disabled. 0: Fixed parity disabled 1: Fixed parity enabled
4	PS	RW	1'b0	Even parity selection. Writable only if UART is not busy. When parity is enabled, this is used to select between even and odd parity. If set to 1, an even number of logical "1s" is transmitted or checked. If set to zero, an odd number of logical "ones" is transmitted or checked. 0: odd parity 1: Even parity
3	PCE	RW	1'b0	Parity enabled. Writable only if UART is not busy. This bit is used to enable and disable parity generation and detection in transmitted and received serial characters, respectively. 0: Disable parity 1: Enable parity
2	STOP	RW	1'b0	Number of Stop bits. Writable only if UART is not busy. This is used to select the number of stop bits for each character that the peripheral device will send and receive. If set to zero, 1 stop bit is transferred in the serial data. If set to 1 and the data bit is set to 5, 1.5 stop bits are sent. Otherwise, 2 stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver will only check the first Stop bit. Note: The STOP bit duration of UART implementations may be longer due to the idle time inserted between characters and the baud clock divisor value in the transmission direction for some configurations; 0: 1 stop bit 1: 1.5/2 stop bits
1: 0	M	RW	2'b0	Data length selection.

				Writable only if UART is not busy. When M_E in CR3 is set to 0, this register is used to select the number of data bits for each character that the peripheral device will send and receive. 00: 5 data bits per character 01: 6 data bits per character 10: 7 data bits per character 11: 8 data bits per character

38.4.5 UART CR2, Address = 0X18 (CR2)

Address offset: 0x18

Reset value: 0x00000008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BUSYERRIE	LSIE	TDREIE	RXNEIE
-												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3	BUSYERRIE	RW	1’b1	Enable the BUSYERR state interrupt. This is used to enable/disable BUSYERR state interrupt generation. When UART is busy, try to write to the CR1 register. Lowest priority = 4. 0: Disable BUSYERR status interrupt 1: Enable BUSYERR Status Interrupt
2	LSIE	RW	1'b0	Enable receiver line status interrupt. This is used to enable/disable the generation of receiver line state interrupts. This is the highest priority interrupt. The interrupt flag is a combination of PE, FE, OE, BI, ADDR_RCV interrupt flags. 0: Disable receiver line state interrupt 1: Enable Receiver Line State Interrupt
1	TDREIE	RW	1'b0	Enable Transfer Keep Register Null interrupt. This is used to enable/disable the generation of a transfer hold register null interrupt. This is the third highest priority interruption. 0: Disable transmission null interrupt 1: Enable transmission null interrupt
0	RXNEIE	RW	1'b0	Enable the Receive Data Availability interrupt. This is used to enable/disable the received data available interrupt. These are the second highest priority interruptions. 0: Disable receive data interrupt 1: Enable receive data interrupt

38.4.6 UART CR3, Address = 0X1C (CR3)

Address offset: 0x1c

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	DMA_SOFT_ACK	DMAT	DMAR	TX_MODE	SEND_ADDR	ADDR_MATCH	M_E
-									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 7	Reserved	-	-	-
6	DMA_SOFT_ACK	RW	1’b0	The software clears the dma send request and the dma receive request. 0: does not affect

				1: Clear dma send request and dma receive request
5	DMAT	RW	1'b0	DMA is enabled when transmitting. 0: Interrupt is disabled 1: DMA mode is enabled for transmission
4	DMAR	RW	1'b0	DMA enable receiver 0: Interrupt is disabled 1: DMA mode is enabled for reception
3	TX_MODE	RW	1'b0	Transfer mode control bit. This bit is used to control the type of transmission mode during the 9-bit data transmission. 1: In this mode of operation, the transfer hold register is 9 bits wide. The user needs to ensure that the address/data of the TDR register is written correctly. Address: 9th digit set to 1 Data: 9th digit set to 0 Note: The Transfer Address Register (TAR) is not suitable for this mode of operation. 0: In this mode of operation, the width of the transfer hold register is 8 bits. The user needs to program the address into the transfer address register and the data into the TDR register. The SEND_ADDR bit serves as a control knob indicating when the UART transmits an address.
2	SEND_ADDR	RW	1'b0	Send the address control bit. This bit serves as a control knob for the user to determine when to send an address in transmission mode. Note: 1. After the address character is issued, this bit is automatically cleared by the hardware. Users should not program this bit to 0. 2. This field is only applicable if the M_E bit is set to 1 and TX_MODE is set to 0. 0: 9-bit character content from TDR register 1: 9-bit character content: The 9th bit is set to 1 and the remaining 8 bits will match what is being programmed in the "Transfer Address Register".
1	ADDR_MATCH	RW	1'b0	Address matching pattern. This bit is used to enable address matching during reception. In address match mode, UART will wait for an incoming character with the 9th bit set to 1. And further check whether the address matches the address programmed in the "Receive Address Match Register". If there is a match, the subsequent characters are considered valid data and UART starts receiving data. In normal mode, the UART will start receiving data and 9-bit characters will form. The user is responsible for reading data and distinguishing between b/n addresses and data. Note: This field applies only if M_E is set to 1. Value: 0: normal mode 1: Address matching mode
0	M_E	RW	1'b0	Enable 9bit This bit is used to enable the 9-bit data used to send and receive transmissions

38.4.7 UART RAR, Address = 0X20 (RAR)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	RAR							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	RAR	RW	8'b0	This is the address match register in receive mode. If the 9th bit is set to 1 in the input character, the remaining 8 bits

				will be checked against the register value. If the match is successful, the subsequent characters with bit 9 set to 0 are treated as data bytes until the next address byte is received. Note:-This register applies only if the ADDR_MATCH and M_E bits are set to 1. The RAR can be programmed at any point in time. However, the user must not change this register value while any reception is in progress.

38.4.8 UART TAR, Address = 0X24 (TAR)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	TAR							
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	TAR	RW	8'b0	This is the address match register in transfer mode. If the M_E bit is enabled and the "SEND_ADDR" bit is set to 1, then UART will send a 9-bit character with the 9th bit set to 1, and the remaining 8-bit address will be sent from this register. Note: This register is only used to send addresses. Normal data should be sent through the programmed TDR register. The hardware automatically clears the "SEND_ADDR" bit after starting sending addresses on the UART serial channel

38.4.9 UART BRRF, Address = 0X28 (BRRF)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BRRF			
-												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	BRRF	RW	4'b0	Baud rate fractional part. The decimal value is determined by (BRRF)/(2 ^ 4).

39. Low-power universal asynchronous receivers/transmitters (LPUART)

The LPUART is an UART which allows bidirectional UART communications with a limited power consumption. Only a 32.768 kHz LSE clock allows for UART communication up to 9600 baud/s. Higher baud rates can be reached when the LPUART is clocked by clock sources different from the LSE clock.

Even when the microcontroller is in low-power mode, the LPUART can wait for an incoming UART frame while having an extremely low energy consumption. The LPUART includes all necessary hardware support to make asynchronous serial communications possible with minimum power consumption.

It supports half-duplex single-wire communication and modem operation (CTS/RTS), and also supports multi-processor communication.

DMA (direct memory access) can be used for data transmission/reception.

39.1 LPUART main features

- AMBA APB interface
- Full-duplex asynchronous communication
- NRZ Standard Mode (Mark/Space)
- Programmable baud rate
- 32.768 kHz clock, baud rate range 300 ~ 9600 baud/s. Higher baud rates require higher clock frequency support
- Dual clock domain: PCLK and dedicated core clock
- Data word length configurable (7 bit/8 bit/9 bit)
- Programmable data sequence, shifting MSB or LSB first
- Number of stop bits is configurable (1/2 bit)
- Single-wire Half-duplex communications
- Support continuous DMA transfer
- Independent enable for transmission and reception
- Independent polarity control for Tx/Rx signals
- Tx/Rx pins are interchangeable
- Support hardware RS - 485/modem flow control
- Transfer detection flag
  - Busy flag
  - End of transmission flags
- Parity control
  - Generate parity bits when transmitted
  - On receive parity
- 4 error flags
  - Overrun error

- Noise detection
- Frame error
- Parity error

- Interrupt sourcrs with flags:
  - CTS change
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Bus idle detected
  - Overrun error
  - Framing error
  - Noise operation
  - Error detection
  - Match address byte
- Multiprocessor communication: LPUART enters silent mode when addresses do not match
- Supports silent mode wake-up (idle frame/address tag detection)

39.2 LPUART functional description

39.2.1 LPUART block diagram

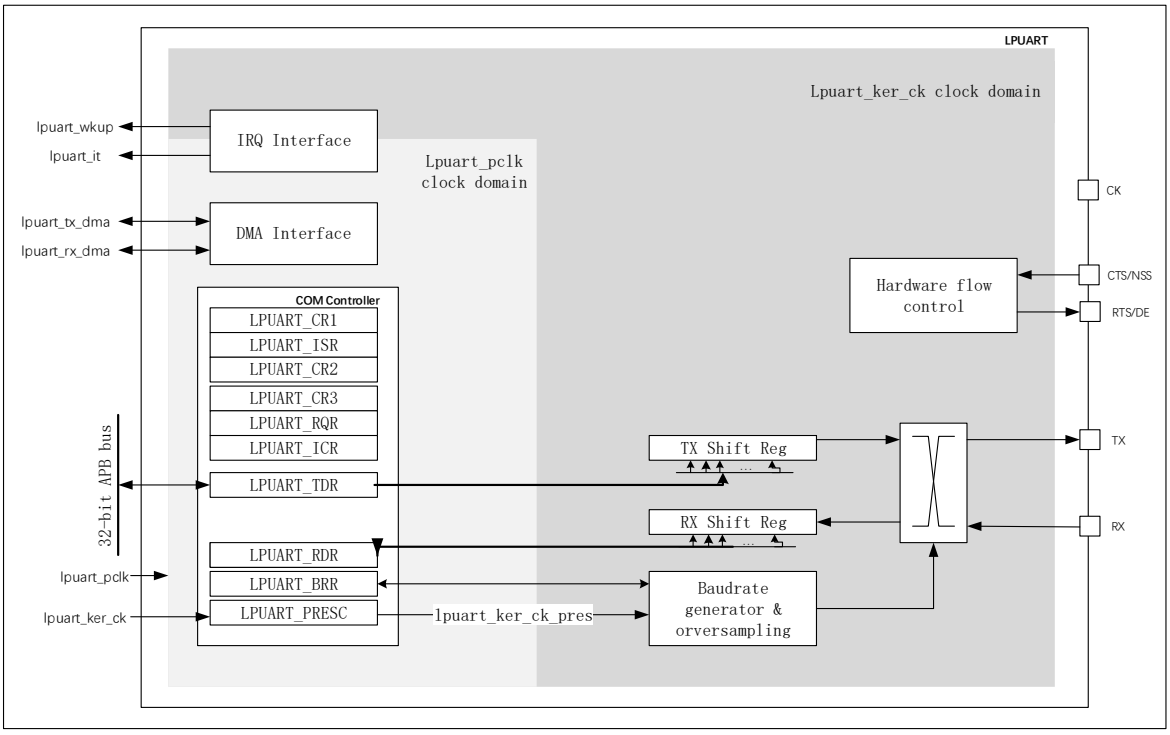


Figure 39-1 LPUART block diagram

39.2.2 LPUART signals

Any LPUART bidirectional communications require a minimum of two pins: Receive Data In (RX) and Transmit Data Out (TX).

- RX: Receive data serial input.
- TX: Transmit data output. When the transmitter is disabled, the output pin returns to its IO port

configuration. When the transmitter is enabled but not sending data, the TX pin is high. In single-wire mode, this IO is used to transmit and receive the data.

Hardware flow control mode

- CTS: transmission starts at low level, and transmission ends to high level
- RTS: Low level indicates that LPUART is ready to receive data

RS485 hardware flow control mode

- DE: Driver enable, activates transmission from external transceiver

Note: DE and RTS multiplex the same pin

39.2.3 LPUART Character Description

According to the {M1, M0} configuration, there are three data lengths of 7 bit/8 bit/9 bits.

M [1: 0] = 2 'b10, 7-position;

M [1: 0] = 2 'b00: 8 bits;

M [1: 0] = 2 'b01: 9 position;

By default, the signal (TX or RX) is in a low state during start bit operation. It is in high state during the stop bit.

These values can be inverted, separately for each signal, through polarity configuration control.

Idle character definition: All 1s within 7-bit/8-bit/9-bit data length + stop bit time;

Break character definition: all zeros for 7 bit/8 bit/9 bit data length time; Disconnect the end of the frame and the transmitter inserts 2 stop bits. That is, 8-bit mode, the length of the disconnected frame is 10 bits 0; 9-bit mode, the disconnected frame length is 11 bits 0.

Transmission and reception are driven by a common baud rate generator. The transmission and reception clocks are generated when the enable bit is set for the transmitter and receiver, respectively.

Below are the waveforms for each frame of different lengths.

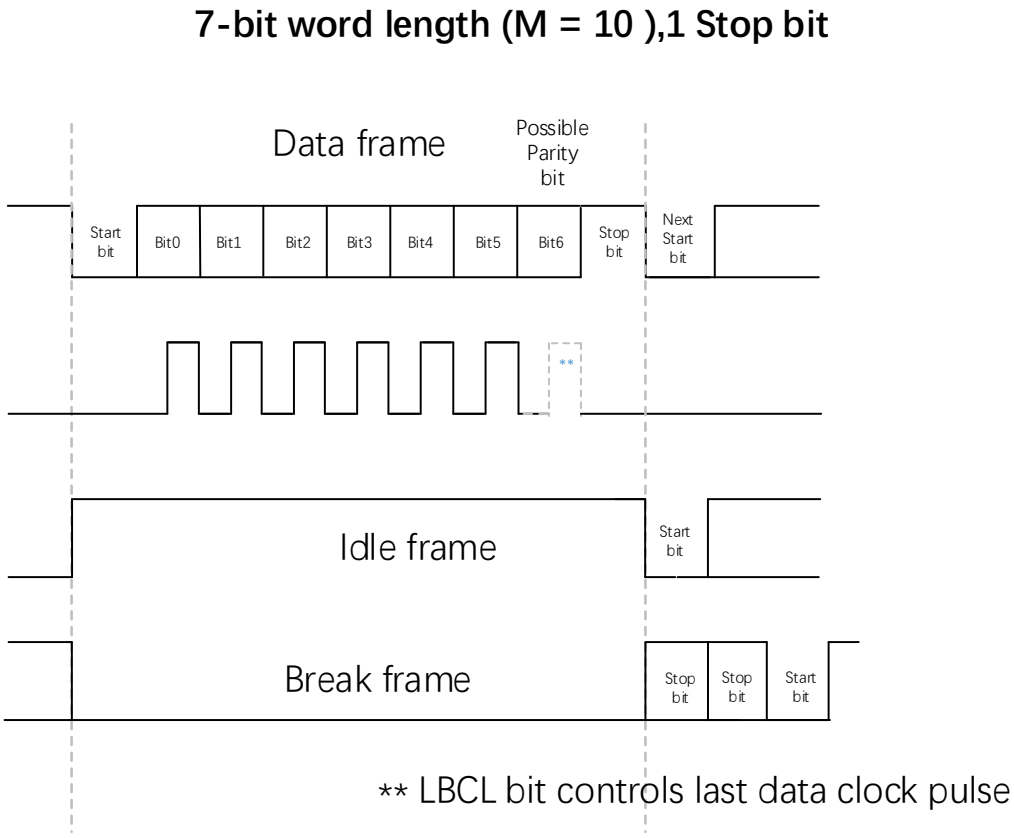


Figure 39-2 LPUART word length programming

39.2.4 LPUART transmission

The LPUART transmission steps are as follows.

- ConfigureLPUART_CR1. {M1, M0}to define the transmission data width (7/8/9bits);
- Configurethe LPUART_BRRregister and select the baud rate;
- Configurethe LPUART_CR2. STOPregister to define the numberof stop bits (1/2bits);
- ConfigureLPUART_CR1.UE = 1 toenableLPUART;
- When multi-processor transmission, configureLPUART_CR3.DMAT = 1to enableDMA;
- Configuring TE = 1, hardware inserting idle frame timing;
- Write transfers data tothe LPUART_TDR register(this operation clearsthe TXE bit). This operation may be repeated multiple times.
- After writing the last data toLPUART_TDR, the software waits forTC = 1, indicating that the last frame has been sent.
- The hardware generates the transmission timing:
  - Generate a start bit and start transmission
  - Transfer shift register output data (default LSB) to the TX pin
  - Stop bits (number sent according to configuration)
- After the last frame of data is transmitted, TC = 1 (when TXE = 1) is generated. When TCIE = 1, a TC interrupt is generated.

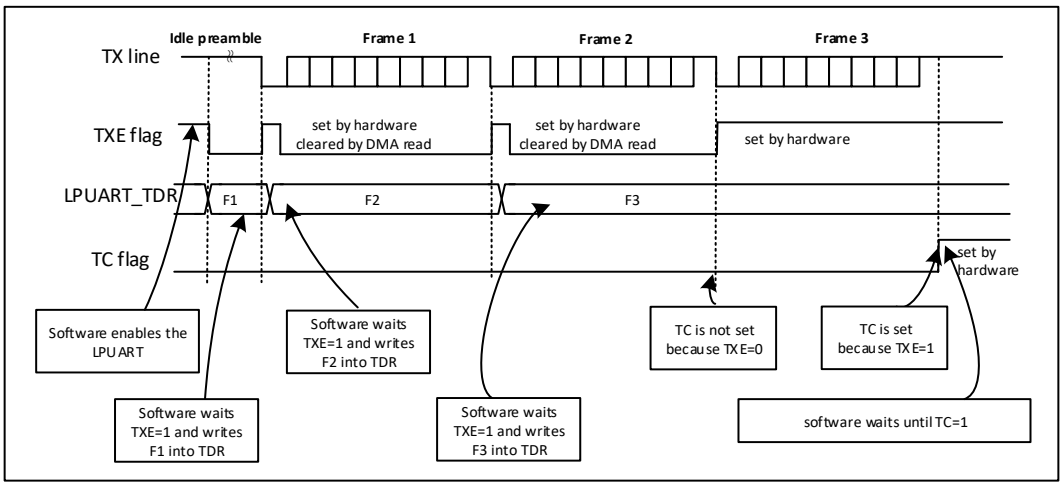


Figure 39-3 TC/TXE behavior at transmission TC/TXE timing

Special frame

Disconnect frame

Setting SBKRQ = 1 will transmit a disconnect frame after the current transmission is complete. The length of the broken frame depends on the configuration of the {M1, M0} registers. After the disconnected frame is sent, the hardware will clear the SBKRQ.

Idle line detected

Set TE = 1 to send an idle frame before the first data frame.

LPUART reception

39.2.4.1 Start bit detection

After a falling edge is detected on the RX, the hardware starts detecting the start bit. When a 0 is detected in the middle of the bit, it is determined that the bit is the start bit. If 1 is detected, the noise error flag (NE) is set, the current start bit is invalid, and the start bit is re-detected.

39.2.4.2 Data character reception

Character reception configuration steps:

- ConfigureLPUART_CR1. {M1,M0}, select a reception length;
- Configurethe LPUART_BRRregister and select the baud rate;
- ConfigureUSAR_CR2. STOPand select1-bit or2-bit stop bits;
- ConfigureLPUART_CR1.UE = 1to enableLPUART;
- When the multi-processor transmits, configuringLPUART_CR3.DMAR = 1to enableDMA reception;
- ConfigureLPUART_CR1. RE, enable reception, and start detecting the start bit;

When the character is received:

- Non-FIFO mode, set RXNE. Indicates that the contents of the shift register have been transferred to the RDR, i.e. the data can be read out;
- When RXNEIE = 1, an interrupt is generated;
- When receiving, if a frame error, a noise error or an overflow error is detected, setting a corresponding error flag bit;
- Multiprocessor communication:

- Non-FIFO mode, RXNE is set after each byte is received and cleared by a read operation of DMA on the received data register;

- In single buffer mode:
  - In non-FIFOmode, the software readsthe LPUART_RDR register toclearthe RXNE bit. The RXNE flagcan also be cleared by writingLPUART_RQR.RXFRQ = 1. The RXNE bit must be cleared before the end of the next character reception to avoid overflow errors;

39.2.4.3 Special frames and error handling

-Break character

When a broken frame is received, LPUART treats it as an error frame.

-Idle character

When an idle frame is detected, it is processed as a normal frame, and if IDLEIE = 1, an interrupt is generated.

- Overrun error

If the RXNE is not reset and another character is received, an overflow error occurs. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after each byte received. If the next data has been received or the previous DMA request has not been executed, the RXNE flag is still set and an overflow error occurs.

When an overflow error occurs, the following error handling is performed:

- SET ORE;
- RDR content is not lost. Reading the LPUART_DR registercan still read the previous data normally;
- The shift register contents are overwritten and subsequently received data is lost;
- If RXNEIE = 1, or EIE = 1, an interrupt is generated;
- Write ORECF = 1 clears ORE bit;

ORE is set to indicate that at least 1 piece of data has been lost. There are the following two situations:

- If RXNE = 1, the last valid data is still in the receive register RDR and can be read out;
- If RXNE = 0, the last valid data has been read away, and the RDR has no valid data to read. This can occur when new (lost) data is received while the last valid data is read in the RDR.

39.2.4.4 Clock source selection

The clock source ofLPUARTis configured bythe RCC_CCIPR.LPUARTxSELregister. Before configuring LPUART_CR1.UE = 1, you must configure the clock source.

The selection of the clock source determines the communication speed range.

LPUARThas dual clock domains, one for the kernel clock (LPUART_ker_ck) and the other for the system clock (LPUART_pclk). When used to wake up from low power, the kernel clock source is configured by the RCC_CCIPR.LPUARTxSELregister.

When the dual clock domain is not supported, the kernel clock is the same as the system clock.

Whenthe MCUenters low power mode and the core clock source selects low frequency,LPUARTcan receive data and wake upthe MCU, and then the software orDMAcan readthe data in LPUART_RDR.

39.2.4.5 Framing error

Frame errors occur when no stop bit is detected at reception, resulting in synchronization failure or large amounts of noise.

When the frame is wrong, do the following processing:

- The hardware sets the FE bit;
- Invalid data is transferred from the shift register tothe LPUART_RDR register;
- In the case of single-byte communication, no interrupt is generated. However, because the NE flag bit and the RXNE flag bit are set at the same time, the FE bit has a rising edge. In the case of multiprocessor communication, if LPUART_CR3. EIE = 1, an interrupt will be generated.
- WriteLPUART_ICR.FECF = 1to clearthe FE flag.

39.2.4.6 Configure the number of stop bits

Configurable bits 1/2 stop bits:

- 1 stop bit: sampling stop bit at 8/9/10 sampling point;
- 2 stop bit: sample the 2nd stop bit after the first stop bit is sampled. The RXNE and FE flags are set after detecting the end of the 2nd stop bit. The 1st stop bit does not detect frame errors.

39.2.5 LPUART baud rate generation

Transmit and receive are configured with the same baud rate.

$$\text{Tx/Rx baud rate} = \frac{256\ X\ lpuartckpres}{LPUARTDIV}$$

LPUARTDIVisconfiguredin the LPUART_BRR register. Do not change the value of the baud rate while the communication is in progress. The LPUART_BRRregister value cannot be lower than0x300. The frequency range of fCK is 3 ~ 4096 x * baud rate.

When the LPUART clock source is LSE, the maximum baud rate is 9600 baud rate. Higher baud rates can be achieved when LPUART is clocked by a different clock source than the LSE clock. For example, the LPUART clock source frequency is 100MHz, and the maximum achievable baud rate is about 33M baud.

Table 39-1 Error calculation of programmed baud rateat lpuart_ker_ck_pres = 32,768 KHz

Baud rate	**Lpuart_ker_ck_pres = 32,768 KHz**		
Expected value	**Actual value**	**Value programmed in the baud rate register**	**Error%**
0.3 KBps	0.3 KBps	0x6D3A	0
0.6 KBps	0.6 KBps	0x369D	0
1200 Bps	1200.087 Bps	0x1B4E	0.007
2400 Bps	2400.17 Bps	0xDA7	0.007
4800 Bps	4801.72 Bps	0x6D3	0.035
9600 Bps	9608.94 Bps	0x369	0.035

Table 39-2 Error calculation of programmed baud rateat lpuart_ker_ck_pres = 100 MHz

Baud rate	**Lpuart_ker_ck_pres = 100 MHz**		
Expected value	**Actual value**	**Value programmed in the baud rate register**	**Error%**
38400 baud	38400.04 baud	A2C2A	0.0001
57600 baud	38400.04 baud	6C81C	0.0001
115200 baud	115200.12 baud	3640E	0.0001
230400 baud	230400.23 baud	1B207	0.0001
460800 baud	460804.61 baud	D903	0.001
921600 baud	921625.81 Porter	6C81	0.0028

4000000 baud	4000000.00 baud	1900	0
10000000 baud	100000000.00 Baud	A00	0
20000000 baud	20000000.00 baud	500	0
33000000 baud	33032258.06 Porter	307	0.1

39.2.6 LPUART reception tolerance

The LPUART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the LPUART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

Need to meet: DTRA + DQUANT + DREC + DTCL + DWU < LPUART receiver tolerance

Where DWU is the error caused by sampling point bias when waking up using low power mode.

Table 39-3 LPUART receiver can receive data correctly at specified maximum tolerance

{M1, M0} bits	**768 < BRR < 1024**		**1024 < BRR < 2048**		**2048 < BRR < 4096**		**4096 ≤ BRR**	
9 bits ({M1, M0} = 01), 1 stop bit	NF is an error	NF doesn't care	NF is an error	NF doesn't care	NF is an error	NF doesn't care	NF is an error	NF doesn't care
	4.2% to -4.84%	4.21% to -4.97%	4.25% to -4.62%	4.21% to -4.72%	4.23% to -3.81%	4.21% to -3.97%	1.51% to -4.01%	1.43 to -4.03%

Note: When the received frame contains an idle frame that is exactly 10/11/9 bits (corresponding to M = 'b00/11/10'), the data specified in the table may vary slightly under special circumstances.

39.2.7 LPUART multiprocessor communication

Multi-processor communication can be realized through LPUART (connecting several LPUARTs in a network). For example, a certain LPUART device may be the master, and its TX output is connected to the RX input of other LPUART slaves; The respective TX output logic of the LPUART slaves is coupled together and connected to the RX input of the master.

An unaddressed device may configure the MME register to be put into silent mode. In silent mode:

- No receive status bits will be set;
- Disable all reception interruptions;
- LPUART_ISR.RWU = 1.By configuringLPUART_RQR.MMRQ, hardware and software can controlLPUART toenter silent mode;

Depending onthe valueof LPUART_CR1. WAKE, LPUARTcan enter or exit silent mode in two ways:

- WAKE = 0: Idle bus detection;
- WAKE = 1: address flag detection;

39.2.7.1 Idle bus detection

When MMRQ writes 1 and RMU automatically sets 1, LPUART enters silent mode.

When an idle frame is detected, LPUART wakes up. The RWUis then cleared by hardware, butLPUART_ISR.IDLEis not set.

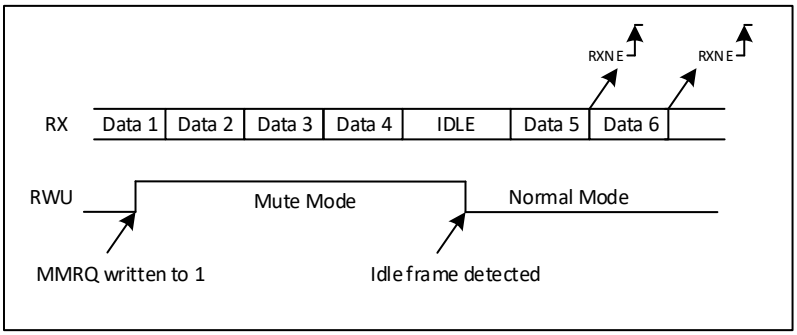


Figure 39-4 Silent mode detected with bus frame

Notes:

If the MMRQ position 1 is placed when the IDLE character has passed, the silent mode will not be entered (RWU is not set to 1).

If the LPUART is activated while the line is IDLE, the idle state is detected after the duration of one IDLE frame (not only after the reception of one character frame).

39.2.7.2 4-bit/7-bit address flag detection

In this mode, if MSB = 1, the byte is considered an address, otherwise it is considered data. In one address byte, the address of the target receiver is placed in four or seven LSBs (several selected by the ADDM7 register). This 4-bit/7-bit address is compared bythe receiver tothe address configured inthe ADDin the LPUART_CR2 register.

If the received byte does not match the configured address, LPUART enters silent mode. At this time, the hardware sets the RWU. The RXNE flag is not set for this address byte and no interrupt nor DMA request is issued as the LPUART would have entered mute mode.

When MMRQ is configured to 1, LPUART also enters silent mode and sets RWU = 1.

If the received byte matches the configured address, LPUART exits silent mode. At this time, the RWU is cleared, and then data is normally received. Since the RWU has been cleared, this situation sets the RXNE/RXFNE bit.

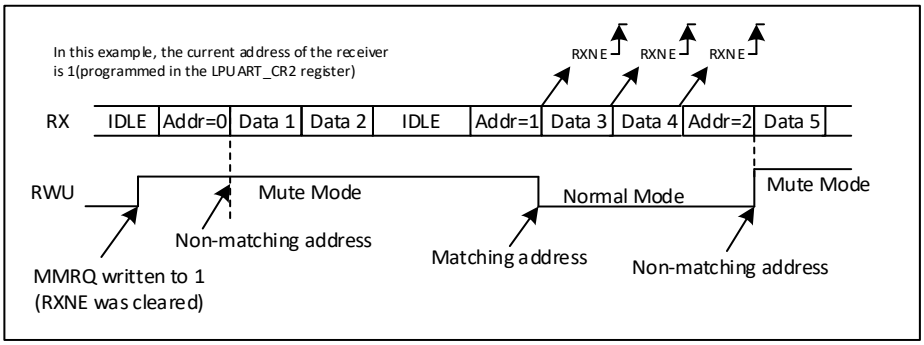


Figure 39-5 Silent mode detected by address flags

39.2.8 LPUART parity

SetLPUART_CR1.PCE = 1to enable parity control (generate a parity bit when sending and perform parity check when receiving). According to the frame length defined by {M1, M0} bits, the supported LPUART frame formats are shown in the following table:

Table 39-4 supported LPUART frame formats

{M1, M0} bits	PCE bit	LPUART frame
00	0	| Start bit | 8 bit data | Stop bit |
00	1	| Start bit | 7 bit data | Parity bit | Stop bit |
01	0	| Start bit | 9-bit data | Stop bit |
01	1	Start bit 8 bit data parity bit stop bit
10	0	| Start bit | 7-bit data | Stop bit |
10	1	Start bit 6 bit data parity bit stop bit

39.2.8.1 Even parity

The even check bit causes 6, 7, or 8 pieces of LSB data in a frame and the number of "1" in the check bit to be even.

39.2.8.2 Odd parity

The odd check bits make 6, 7, or 8 LSB data in a frame and the number of "1" in the check bits odd.

39.2.8.3 Receipt check

Set PE = 1 in case of parity error and generate interrupt when PEIE = 1. The software writes PECF = 1 to clear PE.

39.2.8.4 Transmit parity bit

The MSB is replaced by a parity bit when transmitted.

39.2.9 LPUART single-line half-duplex communication

WhenLPUART_CR3. HDSEL = 1 isconfigured,LPUARTenters single-wire half-duplex communication mode.

In this mode:

- TX and RX are connected inside the module;
- The RX pin is not used;
- Release the TX pin as standard I/O when there is no data transfer

39.2.10 DMA continuous communication

LPUART can utilize DMA continuous communication. The DMA requests for the receive buffer and the transmit buffer are generated separately.

39.2.10.1 Transmission using DMA

DMA transmissionis enabled by configuringLPUART_CR3. DMAT. When TXE = 1,the DMA transfers datafrom the designatedSRAM areatothe LPUART_DR register.

The configuration steps of DMA transmission are as follows:

- ConfigureSYSCFG_CFGR3orSYSCFG_CFGR4to selectthe DMA channelused byLPUART
- Configurethe LPUART_TDR register addressasthe destination address for the DMA transfer. After each TXE event, data will be transferred to this address;
- Configure a memory (e.g. SRAM) address as the source address for DMA transfers. Loading read data from this memory intothe LPUART_TDR registerafter each TXE event;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Depending on the application, the configuration generates a DMA interrupt when the transmission is half or all complete;
- WriteTCCF = 1 toclearthe LPUART_ISR.TC flag;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

When the DMA has written all the data to be transmitted, set TCIF = 1. Then the software needs to wait for TC = 1, indicating that LPUART has sent the last frame of data.

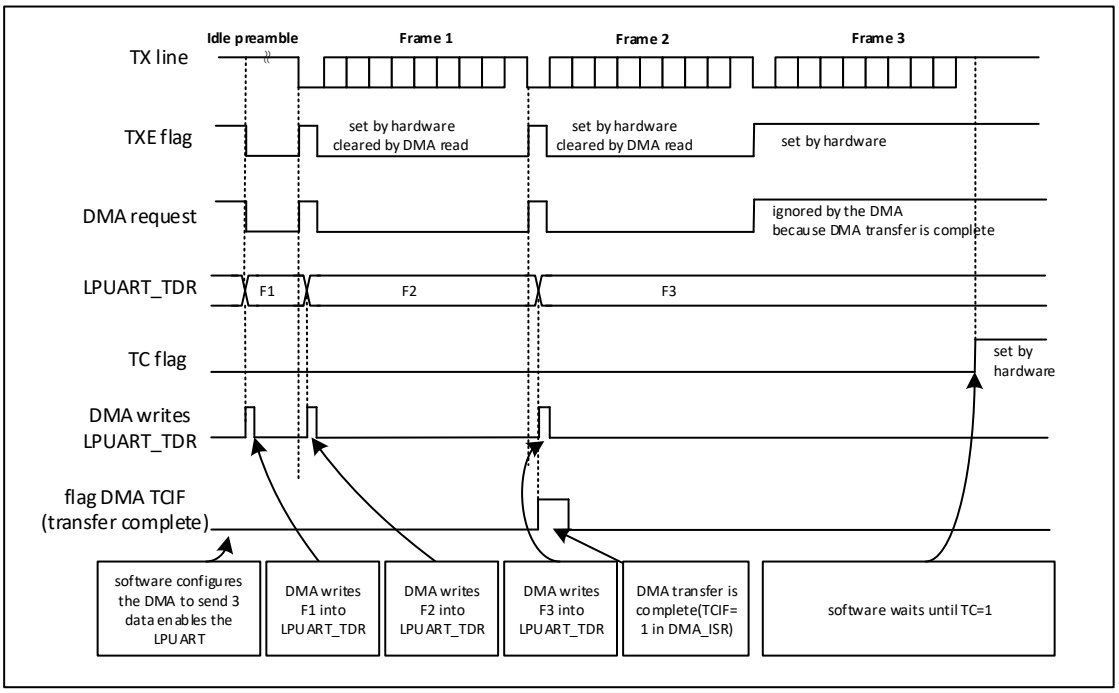


Figure 39-6 LPUART transmitted with DMA

39.2.10.2 Reception using DMA

DMA receptionis enabledby configuringLPUART_CR3. DMAR. For each byte received, the DMA transfers datafromthe LPUART_RDR registerto the designatedSRAM area.

The configuration steps of DMA reception are as follows:

- ConfigureSYSCFG_CFGR3orSYSCFG_CFGR4to selectthe DMA channelused byLPUART
- Configurethe LPUART_RDR register addressasthe source address for the DMA transfer. After each RXNE event, data willbe transferred to memory by reading datafrom this LPUART_RDR register address;
- Configure the SRAM memory address as the destination address for the DMA transfer. After each RXNE event, data will betransferred from the LPUART_RDR registerto this address;
- Write the DMA control register to configure the number of transferred bytes;
- Write the DMA register to configure channel priority;
- Depending on the application, the configuration generates a DMA interrupt when the transmission is half or all complete;
- Enable a DMA channel;

When the transmission reaches the configured number of transmission bytes, the DMA controller generates an interrupt request.

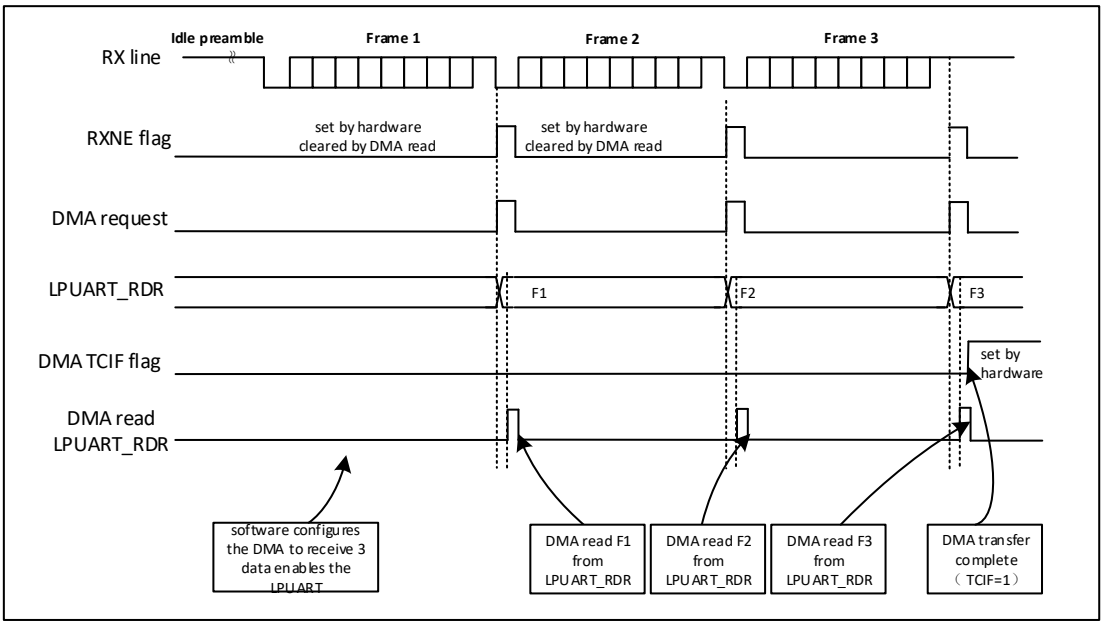


Figure 39-7 LPUART receives with DMA

39.2.10.3 Error flags and interrupts

In multiprocessor communication mode, any error generated during transmission will generate an error flag after the current byte transmission is completed, and an interrupt will be generated if the corresponding interrupt is enabled.

39.2.11 RS232 hardware flow control and RS485 driver enablement

The serial data flow of the two LPUART modules can be controlled through CTS and RTS.

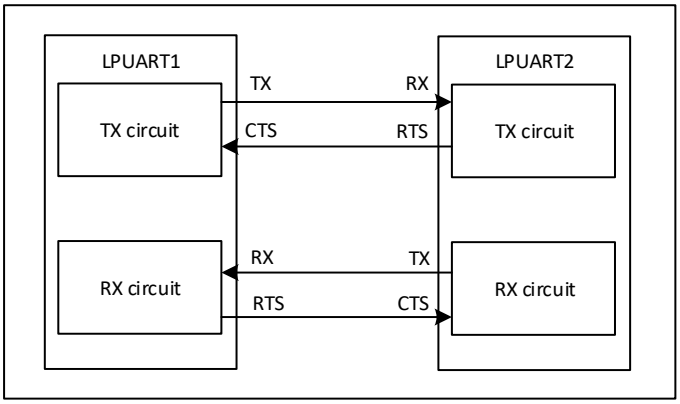


Figure 39-8 Hardware Flow Control

39.2.11.1 RS232 RTS Flow control

RTSflow control is enabledbyconfiguring LPUART_CR3. RTSE = 1. When LPUART is ready to receive data, pull RTS low. When the receive register is full, the RTS pulls high (invalid), indicating that the current frame is the last frame.

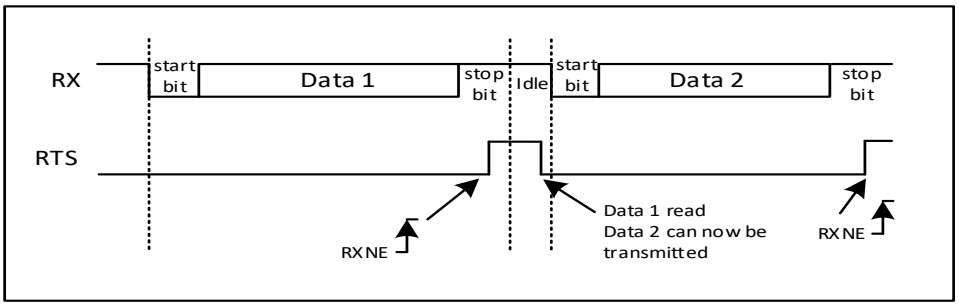


Figure 39-9 RTS flow control

39.2.11.2 RS232 CTS Flow control

CTSflow control is enabledbyconfiguring LPUART_CR3. CTSE = 1. When enabled, the CTS signal is detected before transmitting the next frame. If CTS = 0 (valid), transmit the next data (assuming there is data to be transmitted). If CTS = 1 (invalid), the current transmission is complete (before the stop bit).

When CTSE = 1, the hardware sets CTSIF as long as the CTS signal is inverted. Indicates that the receiver is ready to communicate or not. An interrupt is generated if the CTSIE bit =1.

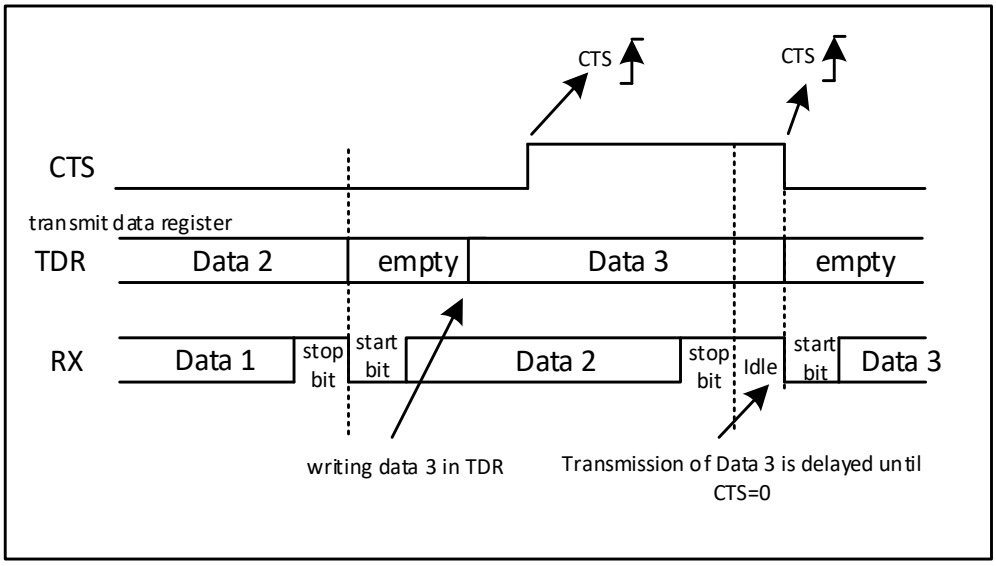


Figure 39-10 CTS flow control

Note: CTS maintains at least 3 LPUART source clocks high before the end of the current frame. CTSCF is also required to maintain at least 2 × PCLK cycles.

39.2.11.3 RS485 Driver Enable

ConfigureLPUART_CR3. DEM = 1to enablethe DE function. With the DE signal, the user can activate the external transmission control. The effective polarity of the DE signalis configured by theL-PUART_CR3.DEP register.

The valid time isconfigured by LPUART_CR1. DEATand isthe time between the valid edge of the DE signaland the start of the start bit.

The release time isconfigured by LPUART_CR1. DEDTand isthe time between the last stop bit and the DEis invalidated.

The specific time is calculated as follows, where P = BRR [20:11]:

- DE insertion time:
  - (1 + (DEAT x P)) x fCK, P ≠ 0
  - (1 + DEAT) x fCK, P = 0
- DE release time:
  - (1 + (DEDT x P)) x fCK, P ≠ 0
  - (1 + DEDT) x fCK, P = 0

39.2.12 LPUART low-power modes

LPUART supports low power mode, which allows data to be transferred when the LPUART PCLK is not present.

When UESM = 1, LPUART can wake up the MCU in low power mode.

39.2.12.1 Generate an interrupt wake-up request according to WUS

When a wake-up event is detected, the hardware generates a WUF flag. If WUFIE = 1, a WUF interrupt is generated. In this case, setting the WUF flag to 1 is sufficient to wake up the MCU from low power mode.

Notes:

- Before entering low power mode, ensure that there is no LPUART transmission;
- Regardless of whether the MCU is in low power mode or working mode, whenever a wake-up event is detected, the WUF flag will be set;
- After initializing and enabling LPUART reception, it is necessary to confirm whether the LPUART is truly enabled through the REACK bit, and then configure to enter the low power mode;
- When DMA reception is enabled, DMA reception must be turned off before entering low power mode; Then wake up from low power and re-enable DMA;

39.2.12.2 Entering low power mode from silent mode

If the LPUART is put into Mute mode before entering low-power mode:

- Silent mode cannot be awakened with idle frames because idle frame detection is not supported in low power mode;
- If the silent mode is waked up by address matching, the address matching method is also used to wake up from the low power mode. When entering low power, if RXNE = 1, LPUART will not enter the low power mode, but will remain in the silent mode;

39.2.12.3 Kernel clock off in low power mode

If the kernel clock is also turned off in low power mode,LPUARTgenerates an LPUART_ker_ck_reqsignal to enablethe kernel clockwhen it detects a falling edge onthe RX line.

If the wake-up eventLPUART_CR3. WUSis configured to address match:

- After the core clock is enabled, if the addresses match after receiving one frame, the MCU wakes up and starts normal data reception;

The following figure shows the address verification match after the wake-up event:

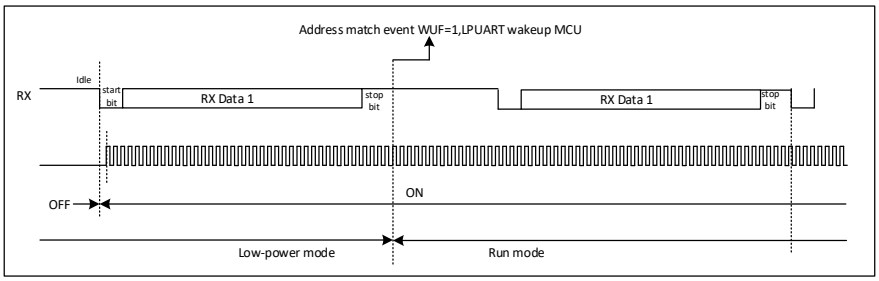


Figure 39-11 LPUART address matching wakeup

If the address validation does not match after the wake-up event, the core clock shuts down again, the MCU does not wake up, and the system remains in low power mode.

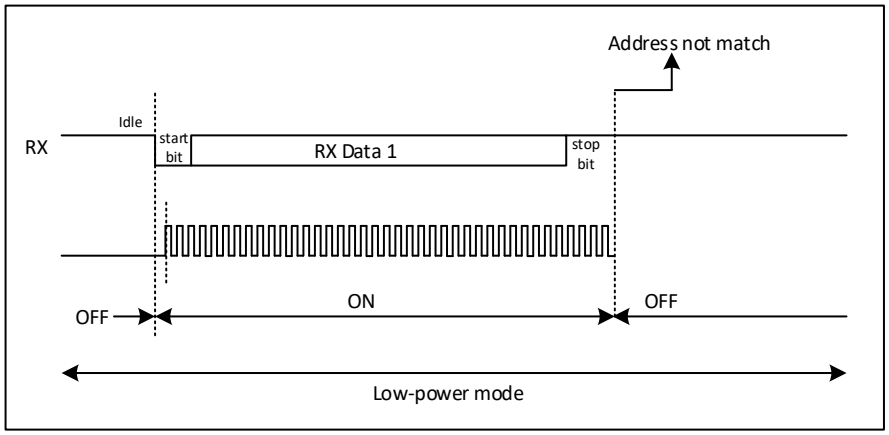


Figure 39-12 LPUART address does not match and cannot wake up

39.3 LPUART interrupts

Interrupt event	Event flag	Enable Control bit	Interrupt zero-ing	Is the interrupt signal valid	
				Interrupts and events	Wake-up
Transmit data register empty	TXE	TXEIE	Write data to TDR	YES	NO
CTS interrupts	CTSIF	CTSIE	Software configu-ration CTSCF = 1	YES	NO
Transmission is com-plete	TC	TCIE	Write data to TDR or TCCF = 1	YES	NO
Receive register is not empty (read data is ready)	RXNE (Correspond-ing to flag bit RXFNE)	RXNEIE (corresponding to inter-rupt enable RXFNEIE)	Read RDR regis-ter or RXFRQ = 1	YES	YES
Overrun error	ORE	RXNEIE/RXFNEIE	ORECF = 1	YES	NO
Idle line detected	IDLE	IDLEIE	IDLECF = 1	YES	NO
Parity error	PE	PEIE	PECF = 1	YES	NO
Noise error/Overrun er-ror/Framing Error in multibuffer communica-tion	NR/ORE/FE	EIE	NECF = 1 clears NE; ORECF = 1 clear ORE; FECF = 1 clear FE;	YES	NO
Match address byte	CMF	CMIE	CMCF = 1	YES	NO

Wake up from low power mode	WUF	WUFIE	WUCF = 1	YES	YES

39.4 LPUART registers

39.4.1 LPUART control register 1 (LPUART_CR1)

Address offset: 0x00

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFFIE	TXFEIE	FIFOEN	M1	Res.	Res.	DEAT [4: 0]					DEDT [4: 0]				
RW			RW	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	RXFFIE	RW	0	RXFIFO full interrupt enabled. 0: interrupt disabled; 1: RXFF interrupt enable; This bit can only be written when FIFO is enabled (FIFOEN = 1)
30	TXFEIE	RW	0	TXFIFO null interrupt enabled. 0: interrupt disabled; 1: TXFE interrupt enable; This bit can only be written when FIFO is enabled (FIFOEN = 1)
29	FIFOEN	RW	0	FIFO mode enable. 0: FIFO mode disabled; 1: FIFO mode enabled This bit can only be written when LPUART is not enabled (UE = 0)
28	M1	RW	0	The value of {M1, M0} configures the length and is set or cleared by the software. 2 'b00: 1 start bit, 8 data bit, n stop bit; 2 'b01: 1 start bit, 9 data bit, n stop bit; 2 'b10: 1 start bit, 7 data bit, n stop bit; This bit can only be written when LPUART is not enabled (UE = 0).
27: 26	Reserved	-	-	-
25: 21	DEAT [4: 0]	RW	5'h0	This register configures the time between the active and start bit of the drive enable (DE) signal. Use the sampling time as the unit. If the DE functionality is not supported, the bit is reserved. This bit can only be written when LPUART is not enabled (UE = 0)
20: 16	DEDT [4: 0]	RW	5'h0	This register configures the time between sending the frame stop bit and the DE signal being invalidated. Use sampling time as the unit (1/8 or 1/16 bit period, depending on the oversampling rate). If there is a write LPUART_TDR register request to send within the DEDT time, the data will only be sent when both DEDT and DEAT have ended. If the DE functionality is not supported, the bit is reserved. This bit can only be written when LPUART is not enabled (UE = 0)
15	Reserved	-	-	-
14	CMIE	RW	0	Character match interrupt enabled. This bit is set or cleared by software. 0: interrupt disabled; 1: CMF interrupt enable;
13	MME	RW	0	Silent mode enabled. 0: The receiver is operating in the working mode; 1: The receiver switches between working mode and silent mode;
12	M0	RW	0	The word length is configured in conjunction with M1. This bit can only be written when LPUART is not enabled (UE = 0)
11	WAKE	RW	0	Receive wakeup mode.

				Wake up mode from silent mode. Set and cleared by software. 0: Idle frame wake-up; 1: Address flag wake up; This bit can only be written when LPUART is not enabled (UE = 0)
10	PCE	RW	0	Parity control enable 0: parity disabled; 1: parity enabled; Parity bit: the 9th bit of 9-bit data; 8th bit of 8-bit data; The 7th bit of the 7bit data. This bit can only be written when LPUART is not enabled (UE = 0)
9	PS	RW	0	Parity selection Set and cleared by software. 0: even check; 1: odd check This bit can only be written when LPUART is not enabled (UE = 0)
8	PEIE	RW	0	PE interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: PE interrupt enable;
7	TXEIE	RW	0	Transfer register null interrupt enabled. Set and cleared by software. 0: Interrupt is disabled 1: TXE interrupt enable;
6	TCIE	RW	0	Transmission complete interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: TC interrupt enable;
5	RXNEIE	RW	0	Receive register non-empty interrupt enable; Set and zeroed by software. 0: Interrupt is disabled 1: ORE or RXNE/RXFNE interrupt enable;
4	IDLEIE	RW	0	IDLE interrupt enable Set and cleared by software. 0: Interrupt is disabled 1: IDLE interrupt enable;
3	TE	RW	0	Transmitter enable 0: Transmission prohibited 1: Transmission enabled
2	RE	RW	0	Receiver enable 0: Reception prohibited; 1: Receive enable, start detecting the start bit;
1	UESM	RW	0	Stops LPUART wake-up enable in low power mode. This register is 1, and LPUART can wake up the MCU if the LPUART clock selects LSI or LSE. 0: LPUART cannot wake up MCU; 1: LPUART mode wakes up the MCU. When enabled, the LPUART clock source selects LSI or LSE. After waking up from low power mode, this bit is cleared.
0	UE	RW	0	LPUART enable When this bit is cleared the LPUART prescalers are stopped. The configuration of LPUART remains, but all status flags, the LPUART_ISR register value changes to the default value. This bit is set and cleared by software. 0: LPUART prescaler and output inhibition, low power mode; 1: LPUART enabled The software needs to wait for LPUART_ISR.TC to be set before it can clear the UE bit and enter the low power mode; At the same time, the DMA channel needs to be disabled before the UE bit is cleared. When this bit is cleared, the LPUART configuration register value is maintained, but the LPUART_ISR status registers are all reset.

39.4.2 LPUART control register 2 (LPUART_CR2)

Address offset: 0x04

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD [7: 0]								Res.			TXOE_ALWAYS_ON	MSBFIRST	DATAINV	TXINV	RXINV
RW								-			RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	Res.	STOP [1: 0]		Res.				Res.			ADDM7	Res.	Res.	Res.	Res.
RW	-	RW	RW	-				-			RW	-			

Bit	Name	R/W	Reset Value	Function
31: 24	ADD [7: 0]	RW	8'b0	Address of the LPUART node This register is used in multiprocessor silent or stop mode and is used as an address when the address wakes up. The MSB bit sent by the transmitter is 1. In normal receive mode, this register is used for character detection. At this time, the received content is compared with ADD [7: 0], and if it matches, the CMF flag bit is set. This register can only be written if RE = 0 or UE = 0.
23: 21	Reserved	-	-	-
20	TXOE_ALWAYS_ON	RW	1	This register is used to control the LPUART output enable. 0: Hardware autonomous control output enables ON/OFF 1: Software control output enabled normally on This bit can only be written when LPUART is not enabled (UE = 0) Note: In half duplex mode, this bit needs to be turned off.
19	MSBFIRST	RW	0	The most significant bit comes first. 0: After the start bit, the 0th bit data is transmitted and received; 1: After the start bit, send and receive the 7/8/9th bit data; This bit can only be written when LPUART is not enabled (UE = 0)
18	DATAINV	RW	0	Binary data inverse processing. 0: Send and receive data register data in forward/forward logic. (1=H, 0=L) 1: Send and receive data register data with reverse/reverse logic. The parity bits are also processed inversely. (1 = L, 0 = H) This bit can only be written when LPUART is not enabled (UE = 0)
17	TXINV	RW	0	The TX pin active level is inverted. 0: The TX pin is at the standard logic level (VCC = 1/idle, Gnd = 0/mark); 1: The TX pin signal value is inverted (VCC = 0/mark, Gnd = 1/idle). This bit can only be written when LPUART is not enabled (UE = 0)
16	RXINV	RW	0	The RX pin active level is inverted. 0: The RX pin is at the standard logic level (VCC = 1/idle, Gnd = 0/mark); 1: The RX pin signal value is inverted (VCC = 0/flag, Gnd = 1/idle). This bit can only be written when LPUART is not enabled (UE = 0)
15	SWAP	RW	0	TX/RX pin interchange. 0: TX/RX pins are defined according to standard pinout; 1: TX/RX pin interchange. This is used when cross-connecting other LPUARTs.

				This bit can only be written when LPUART is not enabled (UE = 0)
14	Reserved	-	-	-
13: 12	STOP [1: 0]	RW	2'b0	Stop bit configuration. 00: 1 stop bit; 01, 11: reserved; 10: 2 stop bit; This bit can only be written when LPUART is not enabled (UE = 0)
11: 5	Reserved	-	-	-
4	ADDM7	RW	0	7-bit/4-bit address detection configuration. 0: 4-bit address detection; 1: 7 bit address detection (8 bit data mode); This bit can only be written when LPUART is not enabled (UE = 0)
3: 0	Reserved	-	-	-

39.4.3 LPUART control register 3 (LPUART_CR3)

Address offset: 0x08

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXFTCFG [2: 0]			RXFTIE	RXTFCFG			Res.	TXFTIE	WUFIE	WUS [1: 0]		Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
RW	RW	RW	RW	-	RW	RW	RW	RW	RW	-	-	RW	-		RW

Bit	Name	R/W	Reset Value	Function
31: 29	TXFTCFG [2: 0]	RW	0	TXFIFO threshold configuration. 000: TXFIFO reaches 1/8 of the total space; 001: TXFIFO reaches 1/4 of the total space; 110: TXFIFO reaches 1/2 of total space 011: TXFIFO reaches 3/4 of total space; 100: TXFIFO reaches 7/8 total space; 101: TXFIFO empty Other values: reserved;
28	RXFTIE	RW	0	RXFIFO threshold interrupt enabled. 0: interrupt disabled; 1: FIFO reaches the configured threshold interrupt enable in RXFTCFG.
27: 25	RXFTCFG [2: 0]	RW	0	RXFIFO full threshold configuration. 000: RXFIFO reaches 1/8 of the total space; 001: RXFIFO reaches 1/4 of total space; 110: RXFIFO reaches 1/2 of total space 011: RXFIFO reaches 3/4 of total space; 100: RXFIFO reaches 7/8 total space; 101: RXFIFO full Other values: reserved;
24	Reserved	-	-	-
23	TXFTIE	RW	0	TXFIFO threshold interrupt enabled. 0: interrupt disabled; 1: TXFIFO reaches TXFTCFG configuration threshold interrupt enable;
22	WUFIE	RW	0	Low power wake-up interrupt enabled. 0: interrupt disabled; 1: WUF interrupt enable;
21: 20	WUS [1: 0]	RW	0	Low power wake-up selection. 00: Address matching generates WUF 01: Reserved; 10: Start bit detection generates WUF; 11: RXNE generates WUF;
19: 16	Reserved	-	-	-
15	DEP	RW	0	DE polarity selection. 0: DE high active; 1: DE low active; This bit can only be written when LPUART is not enabled (UE = 0)
14	DEM	RW	0	DE mode enable.

				0: Interrupt is disabled 1: DE mode is enabled, and DE signal is output at RTS pin. The user controls external transmissions by enabling this bit. This bit can only be written when LPUART is not enabled (UE = 0)
13	DDRE	RW	0	Whether to disable DMA when receiving an error. 0: DMA is not disabled when receiving an error. The corresponding error flag is set, but RXNE = 0. No DMA request is generated, so the error data will not be output. 1: Disable DMA when receiving an error. Both the RXNE and error flag bits are set. Mask the DMA request before the error flag bit is cleared. This bit can only be written when LPUART is not enabled (UE = 0)
12	OVRDIS	RW	0	Overflow prohibited. 0: Set overflow error flag (PRE) when received data is not read before receiving new data; 1: Overflow function is prohibited. After receiving new data, RXNE = 1 and ORE = 0, the new data will overwrite the data in the LPUART_RDR register. This bit can only be written when LPUART is not enabled (UE = 0)
11	Reserved	-	-	-
10	CTSIE	RW	0	CTS interrupt enable 0: Interrupt is disabled 1: CTSIF interrupt enable;
9	CTSE	RW	0	CTS enable 0: CTS hardware flow control disabled 1: CTS mode enabled Data is only transmitted when the CTS input is asserted (tied to 0). If a data is written into the data register while CTS is deasserted, the transmission is postponed until CTS is asserted. This bit can only be written when LPUART is not enabled (UE = 0)
8	RTSE	RW	0	RTS enable 0: RTS hardware flow control disabled 1: RTS interrupt enabled, data is only requested when there is space in the receive buffer. The transmission of data is expected to cease after the current character has been transmitted. The RTS output is asserted (tied to 0) when a data can be received. This bit can only be written when LPUART is not enabled (UE = 0)
7	DMAT	RW	0	DMA is enabled when transmitting. 0: Interrupt is disabled 1: DMA mode is enabled for transmission
6	DMAR	RW	0	DMA enable receiver 0: Interrupt is disabled 1: DMA mode is enabled for reception
5: 4	Reserved	-	-	-
3	HDSEL	RW	0	Half-duplex selection 0: Half duplex mode is not selected 1: Half duplex mode is selected This bit can only be written when LPUART is not enabled (UE = 0)
2: 1	Reserved	-	-	-
0	EIE	RW	0	Error interrupt enable 0: Interrupt is disabled 1: Frame error FE, overflow error ORE, noise NF interrupt generation;

39.4.4 LPUARTbaud rate register (LPUART_BRR)

Address offset: 0x0C

Reset value:0x0000_0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BBR [19:16]			
-												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BBR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	-
19: 0	BBR [19: 0]	RW	0x300	LPUART baud rate. Prohibit writingof values less than0x300inthe LPUART_BRR register

39.4.5 LPUARTrequest register (LPUART_RQR)

Address offset: 0x18

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFRQ	RXFRQ	MMRQ	SBKRQ	Res.
-											RW	RW	RW	RW	-

Bit	Name	R/W	Reset Value	Function
31: 5	Reserved	-	-	-
4	TXFRQ	W	0	Transmit a data refresh request. Write 1 flushes the entire FIFO and sets TXFE (indicating that TXFIFO is empty).
3	RXFRQ	W	0	Receive a data refresh request. Writing 1 to this bit clears the RXNE flag. The overrun situation is avoided by not reading the bit energy and discarding the received data.
2	MMRQ	W	0	Quiet mode request. This bit write 1 puts LPUART into silent mode and sets the RWU flag.
1	SBKRQ	W	0	Send an abort request. This bit write 1 sets the SBKF and sends a BREAK request.
0	Reserved	-	-	-

39.4.6 LPUARTinterrupt and status register (LPUART_ISR)

Address offset: 0x1C

Reset value:0x0080_00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	TXFT	RXFT	Res	RXFF	TXFE	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
-				R		-	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
-					R	R	-	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27	TXFT	R	0	TXFIFO threshold flag. 0: TXFIFO does not reach TXFTCFG threshold; 1: TXFIFO reaches TXFTCFG threshold;
26	RXFT	R	0	RCFIFO threshold flag. 0: RXFIFO does not reach RXFTCFG threshold; 1: RXFIFO reaches RXFTCFG threshold;
25	Reserved	-	-	-
24	RXFF	R	0	RXFIFO Full logo. Indicates the received data RXFIFO size+1. 0: RXFIFO not full; 1: RXFIFO full;
23	TXFE	R	1 Turn on FIFO_EN by	TXFIFO empty. 0: TXFIFO is not empty; 1: TXFIFO empty;

			default 1, otherwise by default 0	Note: This bit is 0 when FIFO_EN is turned off
22	REACK	R	0	Receive an enable acknowledgement flag. Hardware sets/resets this register. Indicates that the hardware is ready to receive before entering low power mode. This bit is reserved if LPUART does not support wake-up from stop low power mode.
21	TEACK	R	0	Send an enable acknowledgement flag. Hardware sets/resets this register. This register may be used when writing TE = 0 in the LPUART_CR1 register to generate an idle frame request, and then writing TE = 1 to satisfy the minimum period of TE = 0.
20	WUF	R	0	Wake up flag from low power mode. The hardware writes this bit as 1 when a wake-up event is detected. The event is defined by the WUS bitfield. This bit is cleared by software by writing 1 to WUCF in the LPUART_ICR register. If WUFIE = 1 in the LPUART_CR3 register, an interrupt is generated. Note: When UESM is cleared, WUF flag is also cleared.
19	RWU	R	0	Received silent mode wake-up. The register is set when a silent mode sequence is received; If a wake-up sequence is received, the register is cleared. The specific wake-up sequence (address tag or idle frame) is controlled by the register LPUART_CR1. WAKE bit. 0: The receiver is in the working mode; 1: The receiver is in silent mode;
18	SBKF	R	0	Send abort flag. This register indicates that the disconnect character is requested to be sent. This register is set by writing 1 to the LPUART_RQR.SBKRQ register by software. The hardware clears the register after sending the stop bit of the abort character. 0: No disconnect character is transmitted; 1: Send disconnect character;
17	CMF	R	0	Address match flag. This register is set when a character matching the ADD [7: 0] value is received. Software writing 1 to the CMCF register clears the bit.
16	BUSY	R	0	Busy flag When the RX line receives data (the start bit is correctly received), the hardware sets this register. When the reception is completed, the hardware clears the register. 0: LPUART idle; 1: Reception is in progress;
15: 11	Reserved	-	-	-
10	CTS	R	0	CTS flag 0: CTS line is 1; 1: CTS line is 0;
9	CTSIF	R	0	CTS interrupt flag. 0: CTS status line has not changed; 1: CTS state line value changes;
8	Reserved	-	-	-
7	TXE	R	1	Transfer register null flag, can write data to LPUART_TDR. 0: Data register full; 1: Data register empty;
6	TC	R	1	Transmission complete This bit is set by hardware if the transmission of a frame containing data is complete and if TXE is set. An interrupt is generated if TCIE=1. The software writes 1 to the TCCF register to clear the bit. 0: Transmission not complete;

				1: Transmission completed
5	RXNE	R	0	Read data register not empty When the shift register value is transferred to the LPUART_RDR register, the hardware sets the register. Software reads the LPUART_RDR register and clears the bit. An interrupt is generated if the RXNEIE bit =1. 0: No data received; 1: Received data ready for reading;
4	IDLE	R	0	DLE line detected An idle frame is detected and the hardware sets the register. An interrupt is generated if the IDLEIE bit =1. The software writes 1 to IDLECF to clear the bit. 0: IDLE frame not detected; 1: IDLE frame detected
3	ORE	R	0	Overflow error flag. This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RXNE=1. The software writes 1 to the ORECF register to clear the bit. When RXNEIE = 1 or EIE = 1, an interrupt is generated. Note: When this bit is set, the RDR register content will not be lost, but the shift register will be overwritten. An interrupt is generated on ORE flag if the EIE bit is set.
2	NE	R	0	START bit noise flag. This bit is set by hardware when noise is detected on a received frame. Software writing NFCF = 1 will clear the bit.
1	FE	R	0	Framing error This bit is set by hardware when out-of-sync, excessive noise, or interrupt characters are detected. The software writes FECF = 1 to clear this bit. An interrupt is generated if the EIE bit =1.
0	PE	R	0	Parity error This bit is set by hardware when a parity error occurs in receiver mode. The software writes PECF = 1 to clear this bit. An interrupt is generated if the PEIE bit =1.

39.4.7 LPUARTInterrupt Flag Zero Register (LPUART_ICR)

Address offset: 0x20

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
-											W	-		W	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
-						W	-		W	-	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 21	Reserved	-	-	-
20	WUCF	W	0	Low power wakeup wakeup flag cleared. Software writing 1 will clear the WUF register.
19: 18	Reserved	-	-	-
17	CMCF	W	0	Address match flag cleared. Software writing 1 will clear the CMF register.
16: 10	Reserved	-	-	-
9	CTSCF	W	0	CTS flag cleared. The software writes 1 to clear the CTSIF register.
8: 7	Reserved	-	-	-
6	TCCF	W	0	The transfer completion flag is cleared. The software writes 1 to clear the TC register.
5	Reserved	-	-	-
4	IDLECF	W	0	Idle flag cleared. The software writes 1 to clear the IDLEF register.

3	ORECF	W	0	Overflow error flag cleared. The software writes 1 to clear the ORE register.
2	NECF	W	0	Noise flag cleared. Software writing 1 will clear the NE register.
1	FECF	W	0	Frame error flag cleared. The software writes 1 to clear the FE register.
0	PECF	W	0	The check value error flag is cleared. The software writes 1 to clear the PE register.

39.4.8 LPUARTreceive data register (LPUART_RDR)

Address offset: 0x24

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	RDR [8: 0]	R	0	Receive data register.

39.4.9 LPUARTSend Data Register (LPUART_TDR)

Address offset: 0x28

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR [8: 0]								
-							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8: 0	TDR [8: 0]	RW	0	Send data register. When the check is enabled, either bit7 or bit8 of the register is replaced by the check bit at the time of transmission. Note: It is recommended to turn on TE and UE first, and then write TDR data.

39.4.10 LPUARTPrescaler Register (LPUART_PRESC)

Address offset: 0x2C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER [3: 0]			
-												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3: 0	PRESCALER	RW	0	Input clock prescalation register. 0000: no frequency division 0001: 2 divided frequency; 0010: divider /4; 0011: 6 divided frequency; 0100: divider /8; 0101: 10 division; 0110: 12 division; 0111: divider /16; 1000: divider /32; 1001: divider /64;

				1010: divider /128; 1011: divider /256; Others: 256 frequency division.

40. Clock trimming controller (CTC)

40.1 Introduction

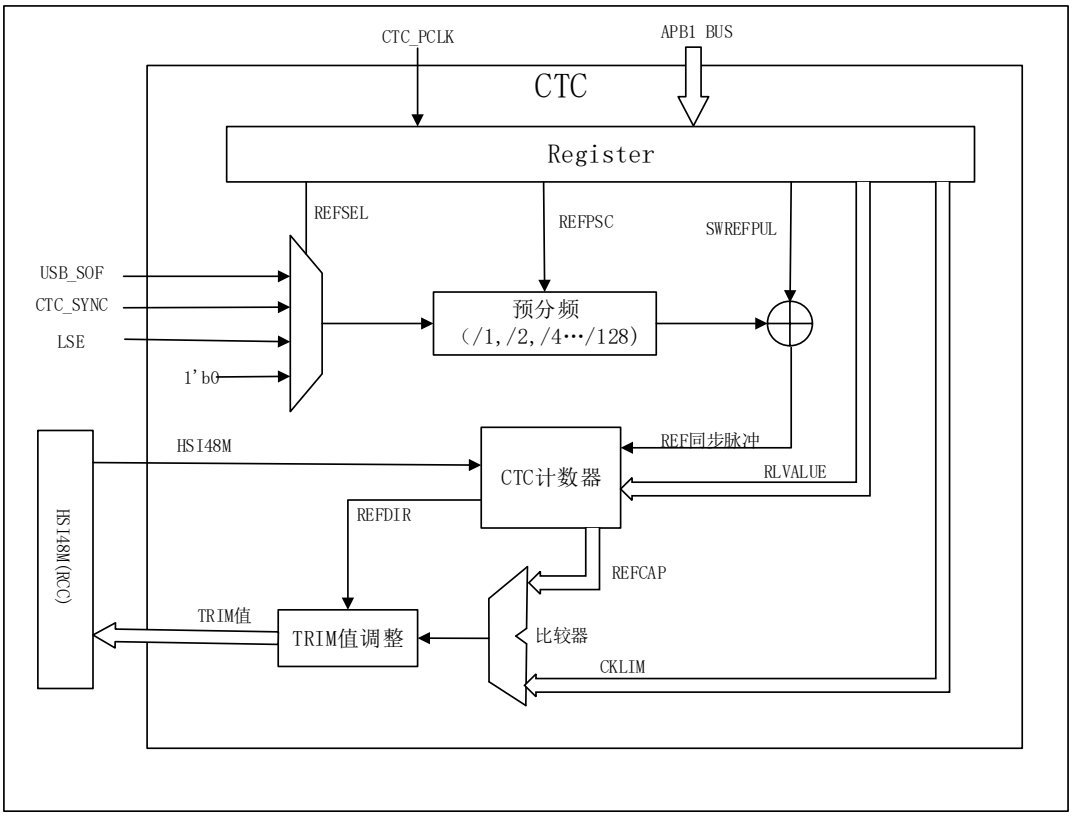
The clock calibration controller (CTC) uses hardware to automatically calibrate the internal 48 MHz RC crystal oscillator (HSI48). When the OTG_FS module uses the HSI48 clock as the clock source, the HSI48 clock frequency must be in the range of 48 MHz ± 0. 25%, but the internal crystal oscillator without calibration cannot meet such high accuracy. The CTC module calibrates the clock frequency of HSI48 based on an external high-precision reference signal source, and obtains an accurate HSI48 clock by automatically or manually adjusting the calibration value.

40.2 Main Features

The CTC module provides the following functions:

- Three external reference signal sources: GPIO, LSE clock, USB_SOF signal from OTG_FS module
- Providing software reference synchronization pulses;
- Automatic hardware calibration, no software operation required;
- 16-bits calibration counter with reference source capture and reload;
- 8-bits clock calibration base value for frequency evaluation and automatic calibration;
- Flag bits and interrupts that indicate the state of clock calibration: calibration success state (CKOKIF), warning state (CKWARNIF), and error state (ERRIF).

40.2.1 CAN block diagram



40.3 Functional description

40.3.1 REF synchronous pulse generator

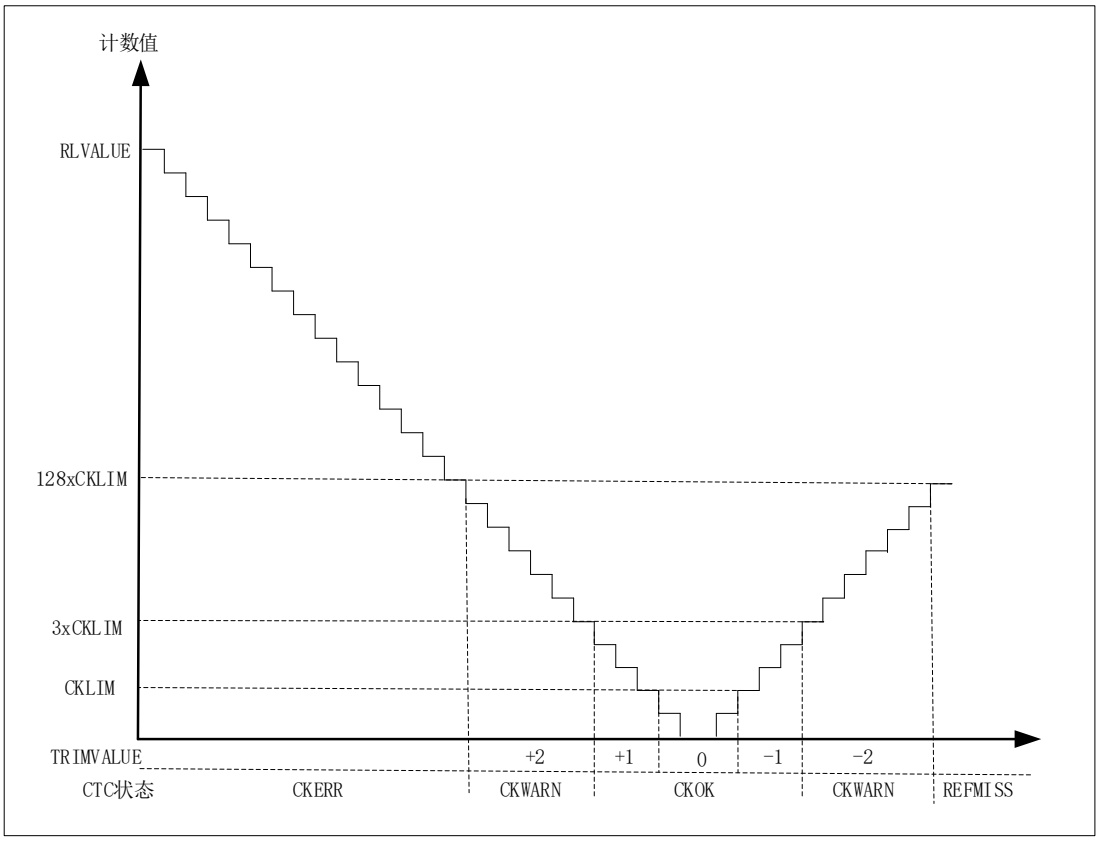
First, the reference signal source: GPIO, LSE clock output or USB_SOF is selected by setting the REFSEL bit in the CTC_CTL1 register (CTC control register 1).

The signal polarity at the time of synchronization of the reference signal source can then be configured by setting the REFPOL bit in the CTC_CTL1 register, and a suitable synchronization clock frequency signal can be generated by setting the REFPSC bit in the CTC_CTL1 register.

If you need to use a software reference pulse signal, you need to set the SWREFPUL bit in the CTC_CTL0 register (CTC control register 0) to 1. The software reference pulse signal and the external reference pulse signal are finally logically OR operated.

40.3.2 CTC calibration counter

The CTC clock calibration counter is clocked by HSI 48. After setting the CNTEN bit in the CTC_CTL0 register, when the first reference synchronization pulse signal is detected, the counter starts counting down from the RLVALUE value (RLVALUE is defined in the CTC_CTL1 register). Each time a reference synchronization pulse signal is detected, the counter reloads the RLVALUE value and resumes counting down. If the reference sync pulse signal is always not detected, the counter counts down to zero, then up to 128 x CKLIM (CKLIM is defined in CTC_CTL1), and finally stops until the next reference sync pulse signal is detected. Once the reference synchronization pulse signal is detected, the count value of the current CTC calibration counter is captured and stored in the REFCAP bit in the CTC_SR register, while the count direction of the current counter is stored in the REFDIR bit in the CTC_SR register. The details are shown in the figure below.



40.3.3 Frequency evaluation and automatic calibration process

When the reference synchronization pulse signal appears, the clock frequency evaluation function starts to be executed. If the reference synchronization pulse signal appears during the counter counting down, the current clock frequency is slower than the expected clock frequency (frequency 48M), and the TRIMVALUE value (clock calibration value) in CTC_CTL0 needs to be increased. If the reference synchronization pulse signal appears in the process of counting up the counter, it means that the current clock frequency is faster than the expected clock frequency, and the TRIMVALUE value needs to be reduced. The CKOKIF bit, CKWARNIF bit, CKERR bit, and REFMISS bit in the CTC_SR register reflect the status of the frequency evaluation.

If the AUTOTRIM position in CTC_CTL0 is 1, the hardware enables auto-calibration mode. In this mode, if the reference synchronization pulse signal appears during the counter counting down, it means that the current clock frequency is slower than the expected clock frequency, and the TRIMVALUE value in CTC_CTL0 will automatically increase to increase the current clock frequency. On the contrary, if the reference synchronization pulse signal appears during the counter counting up, it means that the current clock frequency is faster than the expected clock frequency, and the TRIMVALUE value will automatically decrease, thereby decreasing the current clock frequency.

- When the value of the counter is < CKLIM, the reference synchronization pulse signal is detected:

The CKOKIF bit (clock calibration success flag bit) in the CTC_SR register is set, and if the CKOKIE bit (clock calibration complete interrupt enable bit) in the CTC_CTL0 register is set to 1, an interrupt will be generated. If AUTOTRIM in the CTC_CTL0 register is set to 1, the TRIMVALUE value in the CTC_CTL0 register does not change.

- When the value of CKLIM ≤ counter is < 3 x CKLIM, the reference synchronization pulse signal is detected:

The CKOKIF bit in the CTC_SR register is set, and an interrupt will be generated if the CKOKIE position 1 in the CTC_CTL0 register. If the AUTOTRIM position in the CTC_CTL0 register is 1, the TRIMVALUE value in the CTC_CTL0 register will be incremented by 1 during counter down and decremented by 1 during counter up.

- When the value of 3 x CKLIM ≤ counter is < 128 x CKLIM, the reference synchronization pulse signal is detected:

The CKWARNIF bit (clock calibration warning interrupt bit) in the CTC_SR register is set, while an interrupt will be generated if the CKWARNIE bit (clock calibration warning interrupt enable bit) in the CTC_CTL0 register is set to 1. If the AUTOTRIM position in the CTC_CTL0 register is 1, the TRIMVALUE value in the CTC_CTL0 register will be incremented by 2 during counter down and subtracted by 2 during counter up.

- The value of the counter is ≥ 128 x CKLIM, and the reference synchronization pulse signal is detected during the counter counting down:

The CKERR bit (clock calibration error bit) in the CTC_SR register is set, and if the ERRIE bit (error interrupt enable bit) in the CTC_CTL0 register is set to 1, an interrupt will be generated. The TRIMVALUE value in the CTC_CTL0 register does not change.

- Value of counter = 128 x CKLIM, counter during up count:

The REFMISS bit (REF sync pulse loss bit) in the CTC_SR register is set, and an interrupt will be generated if the ERRIE position 1 in the CTC_CTL0 register. The TRIMVALUE value in the CTC_CTL0 register does not change.

If the calibration value of TRIMVALUE in the CTC_CTL0 register is greater than 127, an overflow event will occur, and if the calibration value of TRIMVALUE is less than 0, an underflow event will occur. The value range of TRIMVALUE is 0 to 127 (when an overflow event occurs, the TRIMVALUE value is 127; when an underrun event occurs, the TRIMVALUE value is 0). Then, the TRIMERR bit (calibration value error bit) in the CTC_SR register will be set, and if the ERRIE position 1 in the CTC_CTL0 register, an interrupt will be generated.

40.3.4 Software Programming Guide

The RLVALUE bit and CKLIM bit in the CTC_CTL1 register are key to clock frequency evaluation and hardware automatic calibration. Their values are calculated from the frequency of the desired clock (HSI 48: 48 MHz) and the frequency of the reference synchronization pulse signal. The ideal state is that the reference synchronization pulse signal appears when the CTC counter counts to zero, so the value of RLVALUE is:

RLVALUE = (fclock ÷ fREF)-1

The value of CKLIM is set by the user according to the accuracy of the clock. It is generally recommended to set it to half of the step size, so the value of CKLIM is:

CKLIM = (fclock ÷ fREF) × 0.12% ÷ 2

A typical step size value is 0.12%, fclock is the frequency of the desired clock (48 MHz), and fREF is the frequency of the reference synchronization pulse signal.

TRIMVALUE in the CTC_CTL0 register can be written by software when the AUTOTRIM bit is 0, but modifying TRIMVALUE will directly affect the frequency of the HSI48M clock. Therefore, TRIMVALUE should not be modified by software at will. It is recommended that the user judge and modify according to the flag bit between the two reference signals; Alternatively, if a reliable value already exists, the user can directly modify the value of HSI48 TRIM in the RCC_CFGR1 register.

40.4 Register Description (base address 0x4000_C800)

40.4.1 CTC Control Register 0 (CTC_CTL0)

Address offset: 0x00

Reset value: 0x00004000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TRIMVALUE [6: 0]							SWREFPUL	AUTOTRIM	CNTEN	Res.	EREFIE	ERRIE	CKWARNIE	CKOKIE
	RW							W	RW	RW		RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 15	Reserved	-	-	-
14: 8	TRIMVALUE [6: 0]	RW	7'h40	HSI48 calibration value. Whenthe AUTOTRIM value-inCTC_CTL0is0, this bit is set and cleared by the software, and this mode is used in the software calibration. Whenthe AUTOTRIM value in CTC_CTL0 is 1, this bit is read-only and is automatically

				modified by the hardware. This mode is used in the hardware calibration process. The intermediate value of TRIMVALUE is 64. When the TRIMVALUE value is incremented by 1, the HSI48 clock frequency increases by about 57 KHz. When the TRIMVALUE value is decreased by 1, the HSI 48 clock frequency is reduced by approximately 57 KHz.
7	SWREFPUL	W	0	The software generates synchronization reference signal pulses. This bit is set by software and provides a synchronization reference pulse signal to the CTC counter. This bit is automatically cleared by the hardware and returns 0 on read operation. 0: No effect; 1: The software generates a synchronization reference pulse signal;
6	AUTOTRIM	RW	0	Hardware auto-calibration mode. This bit is set or cleared by the software. At this position 1, the hardware auto-calibration mode is enabled, and the TRIMVALUE value in CTC_CTL0 is continuously automatically modified by hardware until the clock frequency of HSI 48 reaches 48 MHz. 0: hardware automatic calibration mode disabled 1: hardware automatic calibration mode enabled
5	CNTEN	RW	0	CTC counter enabled. This bit is set or cleared by the software to enable or disable the CTC counter. When this bit is set, the value of CTC_CTL1 cannot be modified. 0: CTC counter disabled 1: CTC counter enabled
4	Reserved	-	-	-
3	EREFIE	RW	0	The expected reference signal interrupt enable. 0: expected reference signal generation interrupt disabled 1: expected reference signal generation interrupt enabled
2	ERRIE	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled
1	CKWARNIE	RW	0	Clock calibration warning interrupt enabled. 0: clock calibration warning interrupt disabled 1: clock calibration warning interrupt enabled
0	CKOKIE	RW	0	Clock calibration completion interrupt enable. 0: clock calibration completion interrupt disabled 1: clock calibration completion interrupt enabled

40.4.2 CTC Control Register 1 (CTC_CTL1)

Address offset: 0x04

Reset value: 0x2022 BB7F

This register can only be accessed word-wise (32-bit). When CNTEN is 1, the value of this register cannot be modified.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REFPOL	Res.	REFSEL [1: 0]		Res.	REFPSC [2: 0]			CKLIM [7: 0]							
RW		RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RLVALUE [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	REFPOL	RW	0	Reference signal source polarity. This bit is set or cleared by software to select the synchronization polarity of the reference signal source. (If USB_SOF is selected for the reference source, the rising and falling edges will be valid at the same time, whether the bit is 0 or 1.) 0: Select rising edge 1: Select falling edge
30	Reserved	-	-	-
29: 28	REFSEL [1: 0]	RW	2'h2	Reference source selection. This bit is set or cleared by the software to select the reference signal source. 00: Select GPIO input signal 01: Select LSE clock 10: Select USB_SOF signal 11: Reserved, select 0
27	Reserved	-	-	-
26: 24	REFPSC [2: 0]	RW	3'h0	The reference signal source is pre-divided. This bit is set or cleared by the software. 000: Reference signal is not frequency divided 001: reference signal divided by 2 010: reference signal divided by 4 011: reference signal divided by 8 100: reference signal divided by 16 101: reference signal divided by 32 110: reference signal divided by 64 111: reference signal divided by 128
23: 16	CKLIM [7: 0]	RW	8'h22	Clock calibration time base limit. This bit is set or cleared by software and is used to define the clock calibration time base limit. This bit is used for frequency evaluation and automatic calibration processes.
15: 0	RLVALUE [15: 0]	RW	16'hBB7F	CTC counter reload value. This bit is set or cleared by the software and defines the reload value of the CTC counter, which is reloaded into the CTC calibration counter when a synchronization reference pulse is detected.

40.4.3 CTC status register (CTC_SR)

Address offset: 0x08

Reset value: 0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REFCAP [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REFDIR	Res.				TRIMERR	REFMISS	CKERR	Res.				EREFIF	ERRIF	CKWARNIF	CKOKIF
R					R	R	R					R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	REFCAP [15: 0]	R	16'h0	CTC counter capture value. When a synchronization reference pulse signal is detected, the count value in the CTC calibration counter is stored into the REFCAP bit.
15	REFDIR	R	1	The CTC calibrates the clock counting direction. When a synchronization reference pulse signal is detected, the counting direction of the CTC calibration counter is stored in the REFDIR bit. 0: up counter

				1: down counter
14: 11	Reserved	-	-	-
10	TRIMERR	R	0	Calibration value error bit. This bit is set by hardware when the TRIMVALUE in CTC_CTL0 overflows or underruns. If the ERRIE bit in CTC_CTL0 is 1, an interrupt will be generated. The TRIMERR bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No calibration value error occurred 1: Calibration value error occurred
9	REFMISS	R	0	The synchronization reference pulse signal is lost. This bit is set by hardware when the synchronization reference pulse signal is lost. REFMISS bit is set when the CTC calibration counter counts to 128 x CKLIM during the count up process and no synchronization reference pulse signal is detected. It means that the current clock is too fast to calibrate to the desired frequency value, or other errors have occurred. The REFMISS bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No synchronization reference pulse signal loss 1: Loss of synchronization reference pulse signal
8	CKERR	R	0	Clock calibration error bit. This bit is set by the hardware when a clock calibration error occurs. When the count value of the CTC calibration counter is greater than or equal to 128 x CKLIM in the process of decounting, and a synchronization reference pulse signal is detected, CKERR is set, indicating that the current clock is too slow to calibrate to the desired frequency value. When ERRIE in CTC_CTL0 is set to 1, an interrupt is generated. The CKERR bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No clock calibration error occurred 1: Clock calibration error occurred
7: 4	Reserved	-	-	-
3	EREFIF	R	0	The reference interrupt flag bit is expected. When the CTC calibration clock counter counts to 0, a reference signal is detected and this bit is set by hardware. When EREFIE in CTC_CTL0 is set to 1, an interrupt is generated. The EREFIF bit can be cleared by writing 1 to the EREFIC bit in CTC_INTC. 0: No desired reference signal generation 1: Desired reference signal generation
2	ERRIF	R	0	Error interrupt flag bit. When an error occurs, this bit is set by hardware. Whenever a TRIMERR, REFMISS or CKERR error occurs, this bit is set. When ERRIE in CTC_CTL0 is set, an interrupt is generated. The ERRIF bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No error occurred 1: Error occurred
1	CKWARNIF	R	0	Clock calibration warning interrupt flag bit. This bit is set by hardware when a clock calibration warning is generated. When the CTC calibration counter count value is greater than or equal to 3 × CKLIM and less than 128 × CKLIM, and a synchronization reference pulse signal is detected, CKWARNIF is set. This means that the current clock frequency is too slow or too fast, but the desired frequency value can be reached through calibration. When a clock calibration warning occurs, the TRIMVALUE value is incremented by 2 or subtracted by 2. When CKWARNIE in CTC_CTL0 is set to 1, an interrupt is generated. The CKWARNIF bit can be cleared by writing 1 to the CKWARNIC bit in CTC_INTC. 0: No clock calibration warning occurred

				1: A clock calibration warning occurs
0	CKOKIF	R	0	Clock calibration successful interrupt flag bit. When the clock calibration is successful, this bit is set by the hardware. If a synchronization reference pulse signal is detected when the count value of the CTC calibration counter is less than 3 x CKLIM, CKOKIF is set. It means that the current clock frequency is normal and can be used, and there is no need to use the TRIMVALUE value for clock calibration. When CKOKIE in CTC_CTL0 is set to 1, an interrupt is generated. The CKOKIF bit can be cleared by writing 1 to the CKOKIC bit in CTC_INTC. 0: Clock calibration failed 1: Clock calibration successful

40.4.4 CTC interrupt clear register (CTC_INTC)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.												EREFIC	ERRIC	CKWARNIC	CKOKIC
												W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	-
3	EREFIC	W	0	EREFIF interrupt clear bit. This bit can only be written by software and a read operation returns 0. Writing 1 can clear the EREFIF bit in CTC_STAT, but writing 0 has no effect.
2	ERRIC	W	0	ERRIF interrupt clear bit. This bit can only be written by software and a read operation returns 0. Write 1 can clear the ERRIF bit, TRIMERR bit, REFMISS bit, and CKERR bit in CTC_STAT, and write 0 has no effect.
1	CKWARNIC	W	0	CKWARNIF interrupt clear bit. This bit can only be written by software and a read operation returns 0. Writing 1 can clear the CKWARNIF bit in CTC_STAT, but writing 0 has no effect.
0	CKOKIC	W	0	CKOKIF interrupt clear bit. This bit can only be written by software and a read operation returns 0. Writing 1 can clear the CKOKIF bit in CTC_STAT, but writing 0 has no effect

41. Random Number Generator (RNG)

41.1 Overview

RNG is a random number generator. The user can enable and turn off the random source of RNG by configuring the control register. The random number needs to wait for the DATA_RDY position bit before it can be read.

41.1.1 Main features

The DMA supports:

- Has 2 independent random sources.
- The random number generated by the random source is post-processed using the linear feedback shift register (LFSR), and a 32-bit random number is generated after 32 clock cycles.

41.1.2 Top-level structure diagram

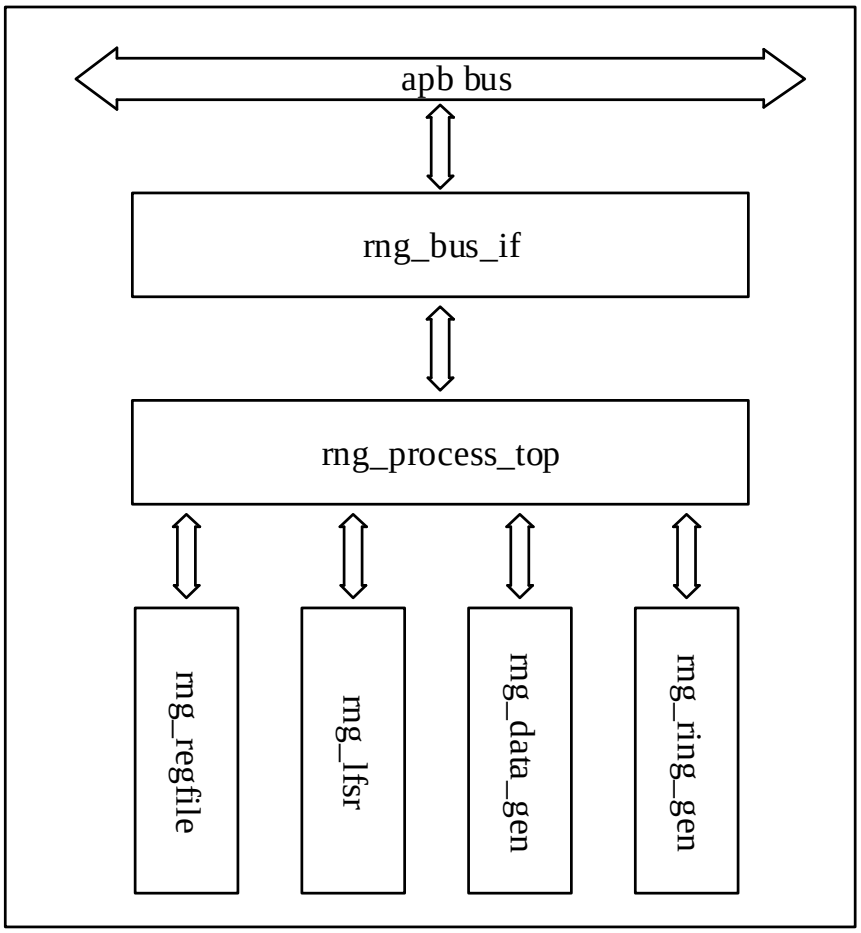


Figure 41-1 RNG top-level module block diagram

41.2 Functional description

41.2.1 Generation of random numbers

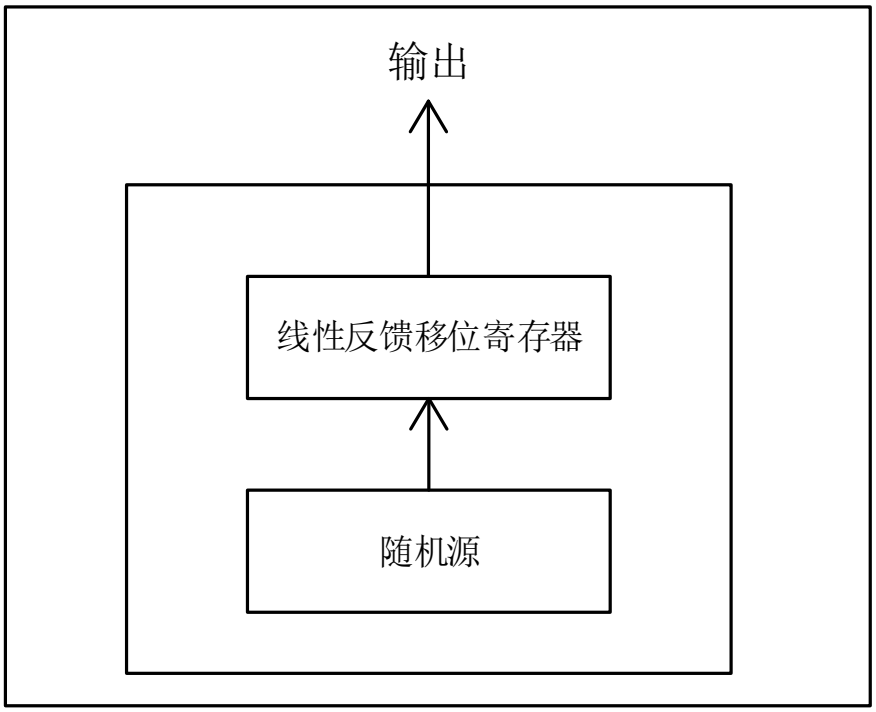


Figure 41-2 RNG random number generation model block diagram

After enabling the RNG random source (RNG_CR = 0x1/2/3), the RNG random source will generate a 1-bit random number every clock cycle, and each random number will be processed by the linear feedback shift register. After 32 clock cycles, a 32-bit random number will be generated.

41.2.2 Software operation of RNG

Take enabling only random source 0 as an example:

1. Turn on the clock of the RNG module;
2. Setting the control register RNG_CR to 0x01;
3. Waiting for the DATA_RDY position bit in the RNG_SR register;
4. Reads the random number in the RNG_DR register.

41.2.3 Random Source of RNG

RNG has two random sources:

The random source 0 is generated by XOR the output of a ring oscillator formed by two sets of multiple free-running NAND gates;

The random source 1 is generated by the exclusive OR of the output of a ring oscillator formed by a set of a plurality of free-running same-OR gates.

41.3 TIMx registers

The register width is 32bits.

41.3.1 RNG Control Register (RNG_CR)

Address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.													IE	RBG1_EN	RBG0_EN
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 3	Reserved	-	-	-
2	IE	RW	0	RNG Data Ready Complete Interrupt Enable 1: Enable interrupt. 0: Interrupt is disabled.
1	RBG1_EN	RW	0	Random Source 1 enable bit. 1: Random Source 1 enabled. 0: Random Source 1 forbidden.
0	RBG0_EN	RW	0	Random Source 0 enable bit. 1: Random Source 0 enabled. 0: Random Source 0 Forbidden.

41.3.2 RNG linear feedback shift register (RNG_LFSR)

Address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RNG_LFSR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNG_LFSR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 0	RNG_LFSR [31: 0]	RW	32'h0	It is used to store the data in the RNG linear feedback shift register, to view the data in the current linear feedback shift register, and to write any data to generate a new random number when the random source is turned on.

41.3.3 RNG status register (RNG_SR)

Address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															DATA_RDY
															R

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	DATA_RDY	R	0	Data ready status flag 1: Random data is ready. 0: Random data is not ready yet. Automatically clearafter reading RNG_DATA An RNG interrupt is generated if the IE position in RNG_CR is 1

Note: 1. The DATA_RDY bit can only be set to 1 after the random source is enabled;

2. Once the RNG_DR register is read, this bit returns 0 until a new valid value is calculated.

41.3.4 RNG Data Register (RNG_DR)

Address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RNG_DATA [31: 16]															

R															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RNG DATA [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	RNG DATA [31: 0]	R	32'h0	32-bit random number.

42. AES Hardware Accelerator (AES)

42.1 Introduction

The AES Hardware Accelerator (AES) encrypts or decrypts data using algorithms and implementations that fully comply with the Advanced Encryption Standard (AES) defined in Federal Information Processing Standards (FIPS) Publication 197.

AES supports CTR, GCM, CMAC, CCM, ECB, and CBC chain modes with 128-bit or 256-bit key lengths.

AES is an AMBA AHB slave peripheral that is only accessible via 32-bit single-time.

The peripheral supports DMA single transfer of incoming and outgoing data.

42.1.1 Main Features

- Supports 128-bit and 256-bit key lengths
- Encryption and decryption based on multiple chain modes
  - Crypto Blockchain CBC
  - Electronic codebook ECB
  - Counter CTR
  - Galois counter CCM,
  - Galois message authentication code CMAC
  - CBC-MAC Counter GCM Mode
- Data is big endian aligned
- The data input and output can be operated using DMA

42.1.2 CAN block diagram

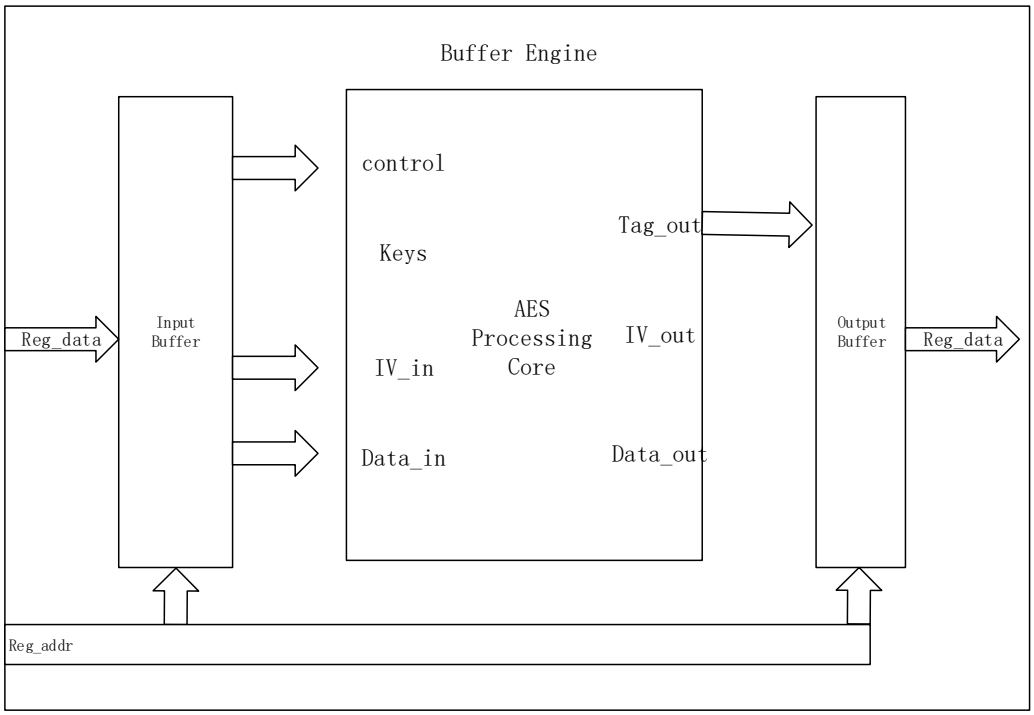
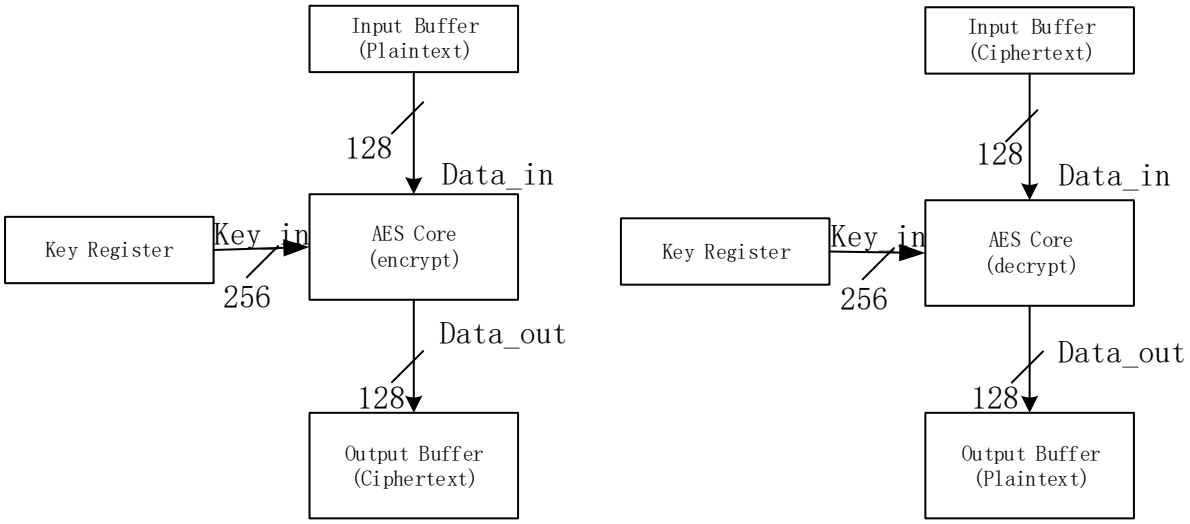


Figure 42-1 AES module block diagram

42.2 Function description

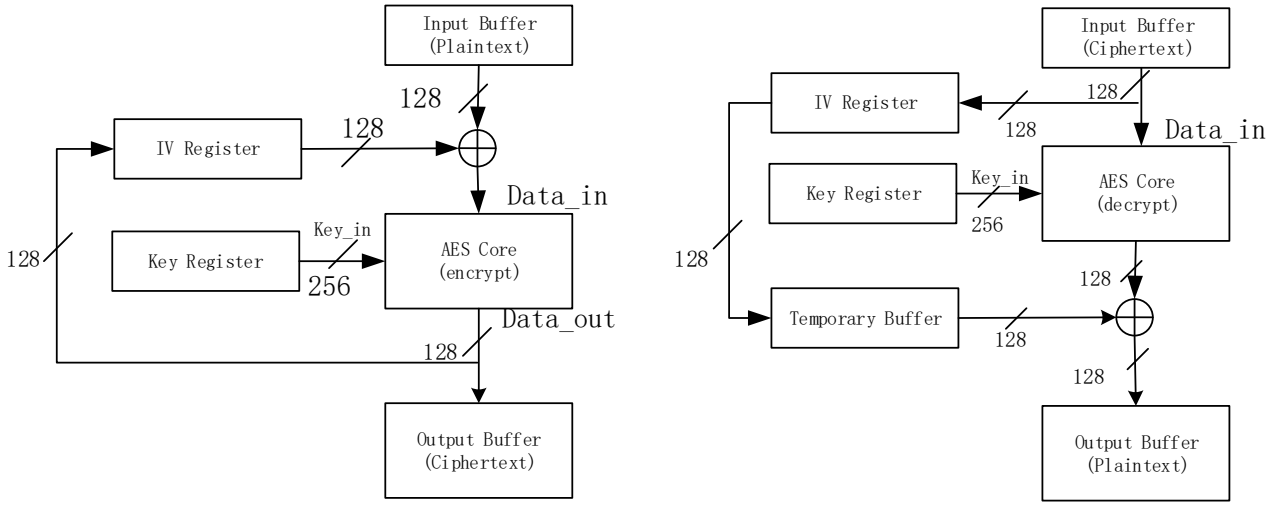
42.2.1 ECB Mode



Encryption　　Decryption

A basic electronic codebook (ECB) mode of operation in which input data is passed directly to the basic encryption core and the output of the encryption core is passed directly to the output buffer. For decryption, the encryption core operates in reverse.

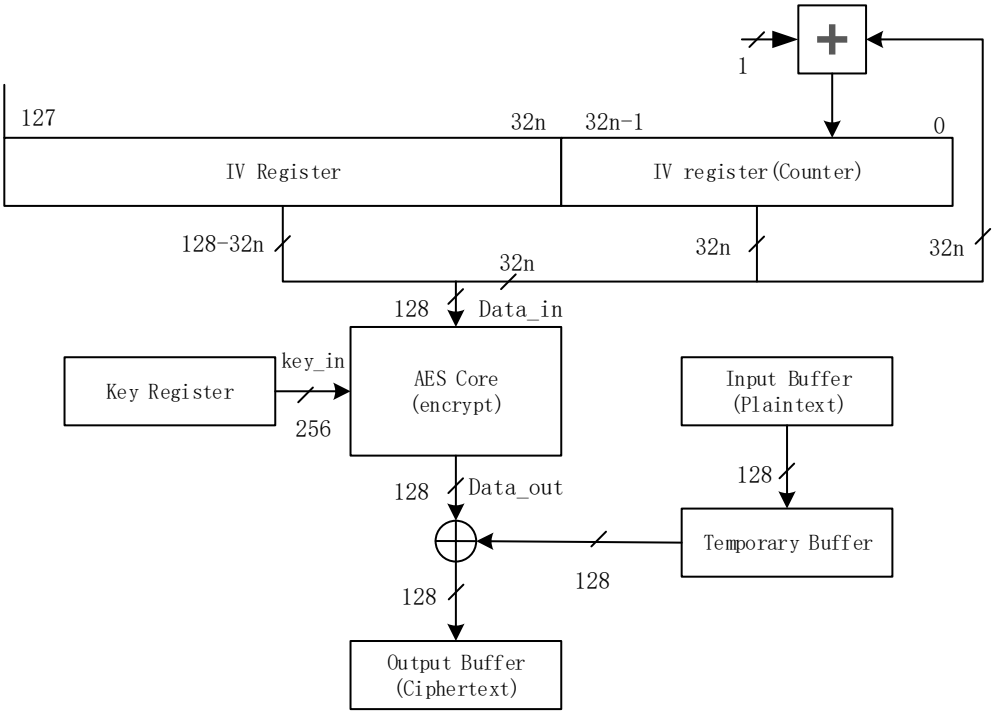
42.2.2 CBC Feedback Mode



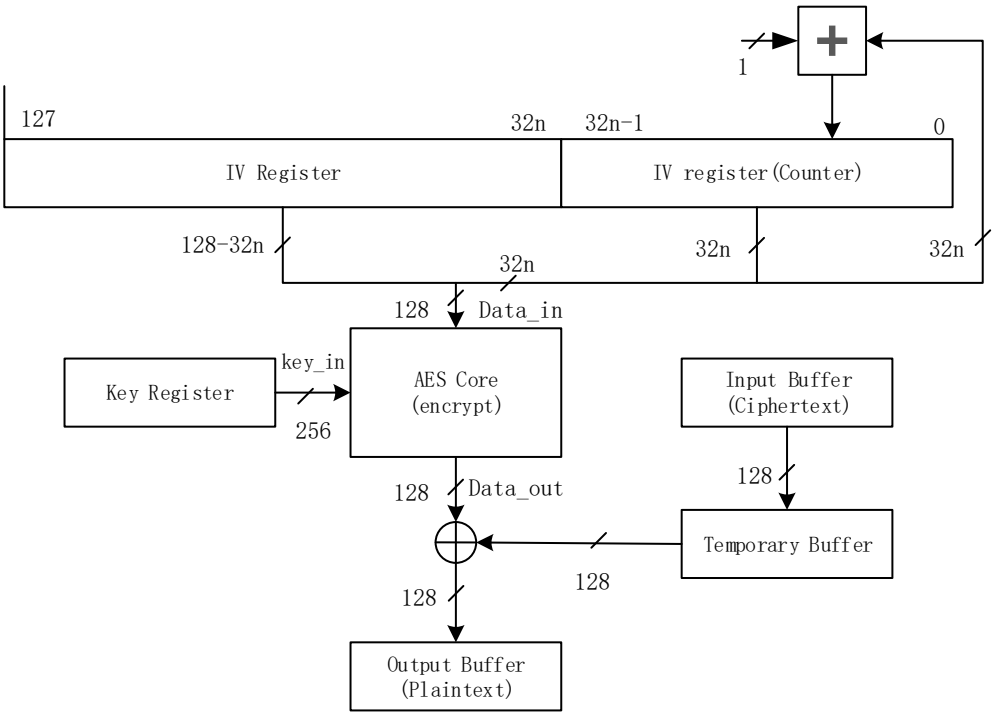
Encryption　　Decryption

A cryptographic blockchain (CBC) feedback mode of operation in which input data is XOR with an initialization vector (IV) before being passed to the base cryptographic core. The output of the encryption core passes directly to the output buffer and becomes the next IV. For decryption, the encryption operation is reversed, and the encryption core output is XOR. The input ciphertext of the current operation is the IV of the next operation.

42.2.3 CTR Mode



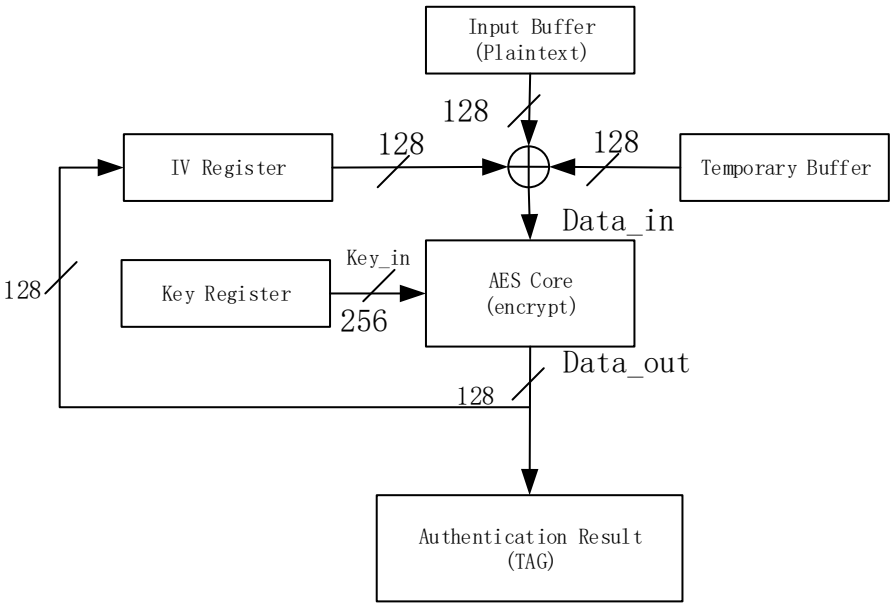
Encryption



Decryption

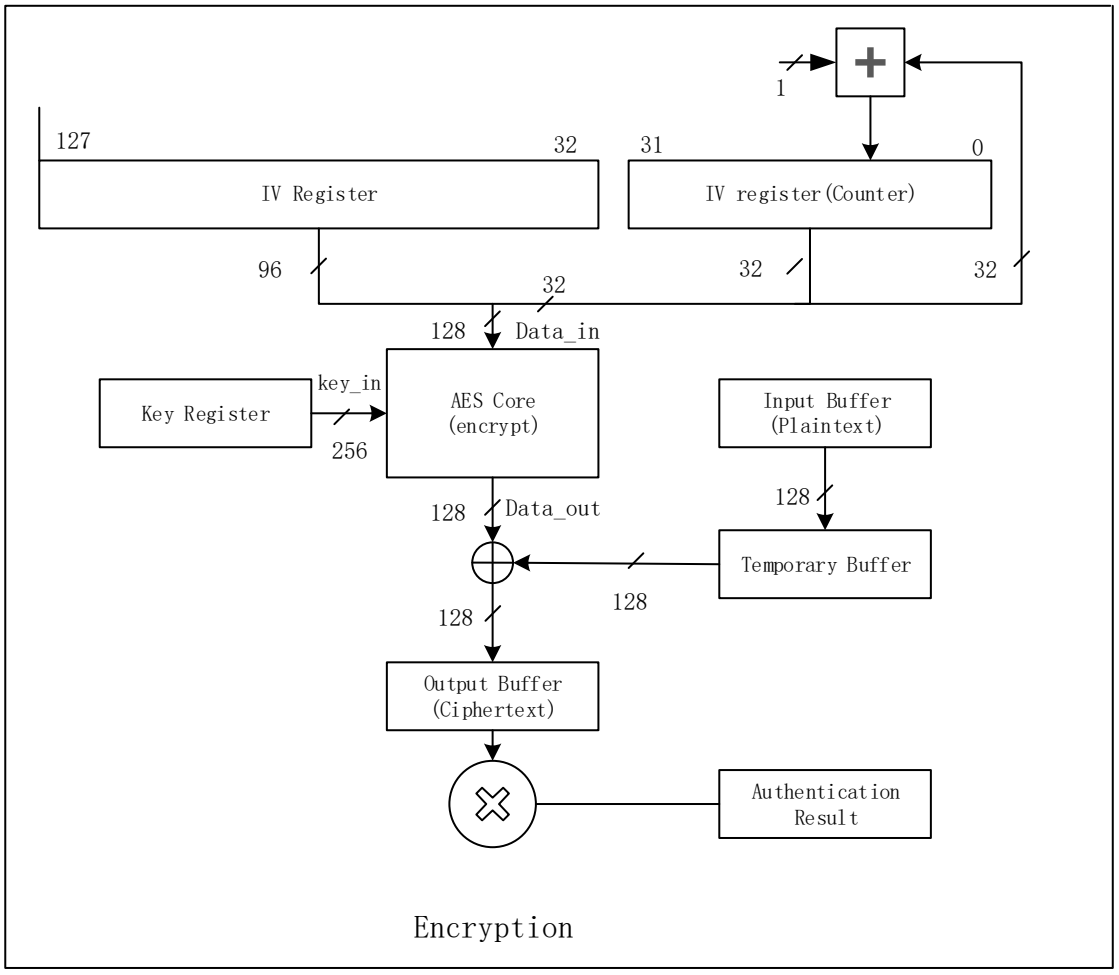
Operating mode of Counter (CTR). This operation will encrypt the IV. The output of the encryption core (encryption IV) is exclusively OR with the data to produce an output result. IV consists of two parts, a fixed part and a counter part. The counter portion increases with each block. The counter width varies depending on the context and may be 16, 32, 64, 96, or 128 bits wide. Note that in this mode, encryption and decryption use the same cryptographic operation.

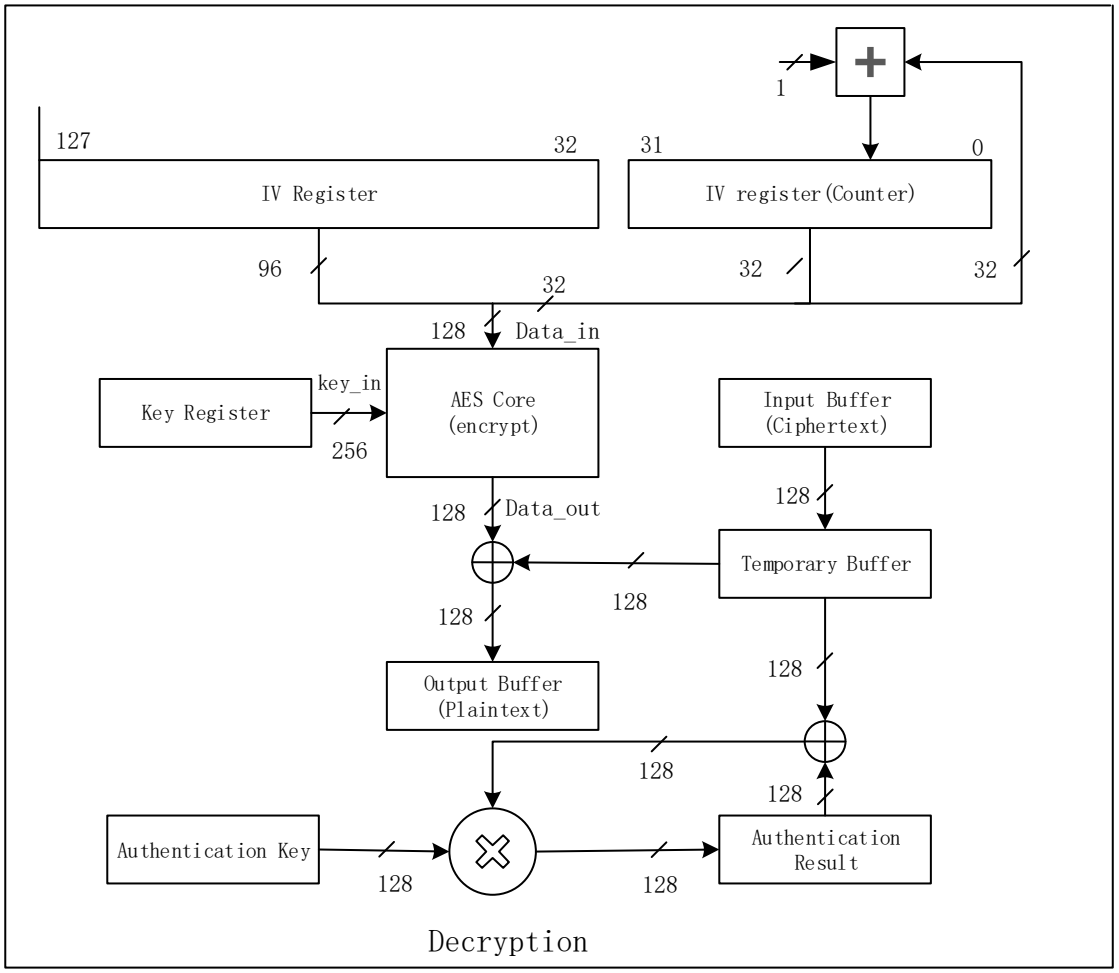
42.2.4 CMAC Authentication Mode



CMAC authentication mode in which the input of the encryption core is XOR with the IV. The output of the cryptographic core is then fed back as the IV of the next block. The last data input block and additional data (stored in the temporary buffer) and IV are XOR processed, and the XOR result is encrypted to output the result (TAG).

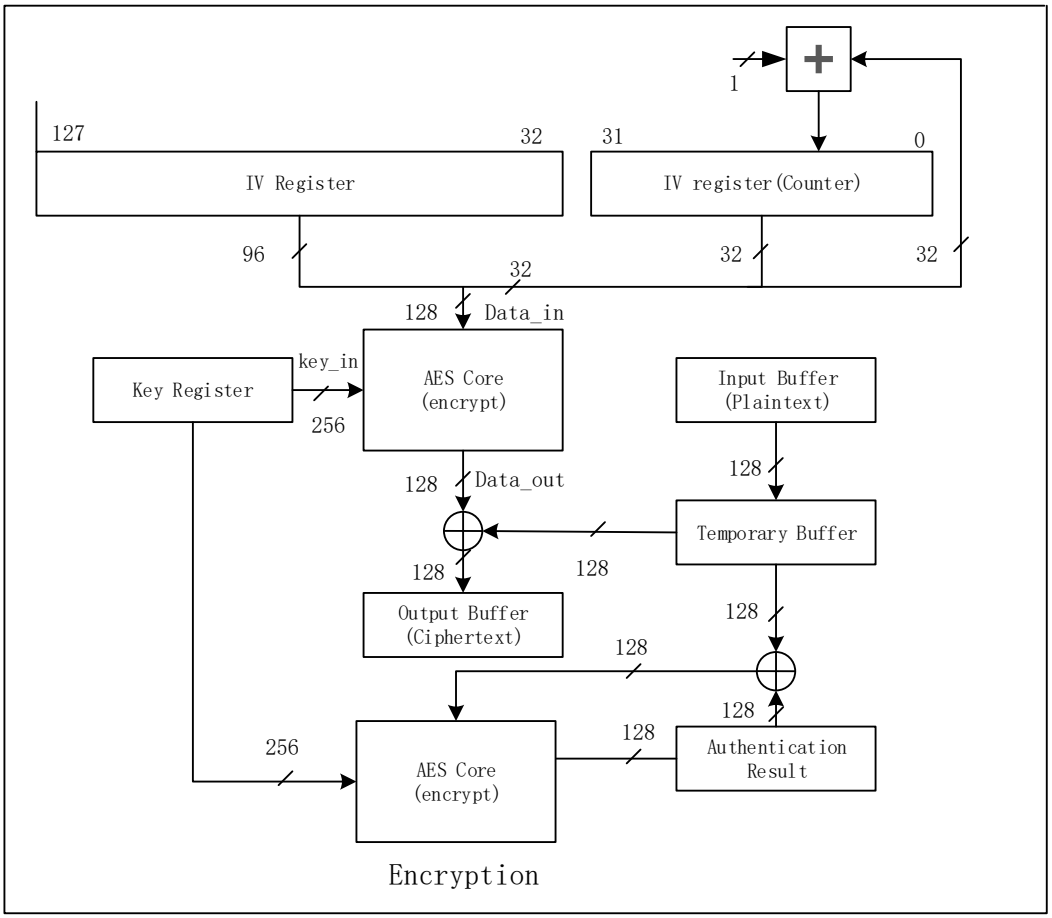
42.2.5 GCM Operation

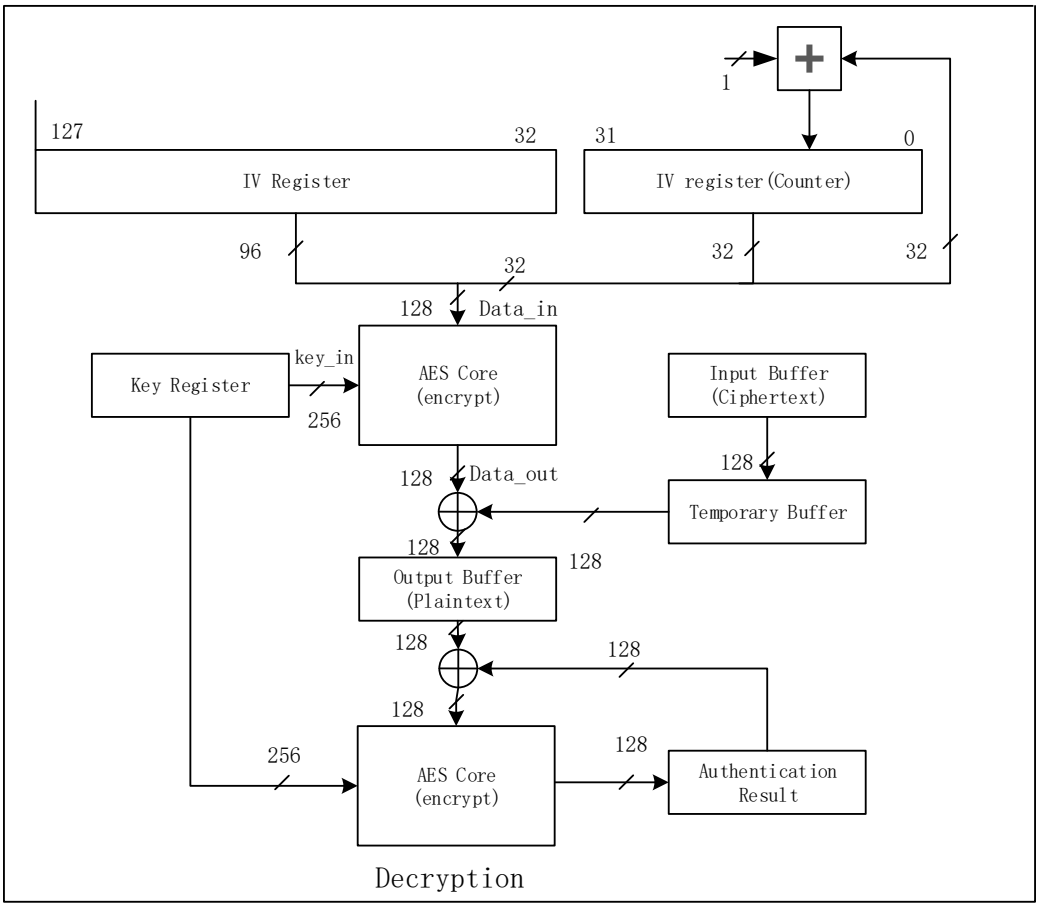




The above diagram illustrates one round of GCM operations for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). The data is encrypted in the same manner as in CTR mode by XOR the encryption core output with the input. After encryption/decryption, the ciphertext is XOR with the intermediate authentication result. The XOR result is used as input for the polynomial multiplication to create the next (intermediate) authentication result.

42.2.6 CCM Operation





The figure above shows one round of CCM (counter with CBC-MAC) operation for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). Data is encrypted in the same way as CTR mode, by XOR the encryption core output and input. The plaintext is exclusively OR with the intermediate authentication result directly after the cryptographic operation, and the exclusive OR result is used as input for the second cryptographic operation to calculate the next (intermediate) authentication result.

42.3 TIMx registers

AES register base address: 0x4800_2400

Offset address	Pin name	Description
0x00	AES_MSG_CFG	Information control register
0x04	AES_CTX_CFG	Key control register
0x08	AES_MSG_TOTAL_BYTES	Information length register
0x0C	AES_MSG_AAD_BYTES	Additional information length register
0x10	AES_GCM_AAD_INFO	GCM mode additional information length register
0x14	AES_CTX_KEY_SEL	Key selection register
0x20	AES_CTX_KEY0	Key Register 0
0x24	AES_CTX_KEY1	Key Register 1
0x28	AES_CTX_KEY2	Key Register 2
0x2C	AES_CTX_KEY3	Key Register 3
0x30	AES_CTX_KEY4	Key Register 4
0x34	AES_CTX_KEY5	Key Register 5
0x38	AES_CTX_KEY6	Key Register 6
0x3C	AES_CTX_KEY7	Key Register 7
0x40	AES_CTX_CBC_KEY0	CBC mode key register 0
0x44	AES_CTX_CBC_KEY1	CBC mode key register 1
0x48	AES_CTX_CBC_KEY2	CBC mode key register 2
0x4C	AES_CTX_CBC_KEY3	CBC mode key register 3
0x50	AES_CTX_CTR0	CTR mode operator register 0
0x54	AES_CTX_CTR1	CTR mode operator register 1
0x58	AES_CTX_CTR2	CTR mode operator register 2
0x5C	AES_CTX_CTR3	CTR mode operator register 3
0x60	AES_CTX_IV0	Initial vector register 0
0x64	AES_CTX_IV1	Initial vector register 1
0x68	AES_CTX_IV2	Initial vector register 2
0x6C	AES_CTX_IV3	Initial vector register 3
0x70	AES_CTX_MAC0	Message verification code register 0
0x74	AES_CTX_MAC1	Message Verification Code Register 1
0x78	AES_CTX_MAC2	Message Verification Code Register 2
0x7C	AES_CTX_MAC3	Message Verification Code Register 3
0x80	AES_INGRESS_FIFO	Input FIFO register
0x84	AES_ING_DBCFG	Input FIFO status register
0x88	AES_DMA_ENG_LEN	Output FIFO register
0x8C	AES_ENG_DBCFG	Output FIFO status register
0x90	AES_DMA_ING_LEN	DMA input length register
0x94	AES_INGF_STATUS	DMA input burst transfer setting register
0x98	AES_ENGRESS_FIFO	DMA output length register
0x9C	AES_ENGFIFO_STATUS	
0xA0	AES_ING_DMA_DONE	DMA output burst transfer setting register
0xA4	AES_ENG_DMA_DONE	Completion Status Register
0xA8	AES_DONE_STATUS	DMA input completion status register

42.3.1 Information Control Register (AES_MSG_CFG)

Address offset: 0x00

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Key_SIZE		MAC_LEN				ALG_MODE				DIR	MSG_BEGIN	MSG_END	Aes_GO
-	-	RW	RW	RW				RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 14	Reserved	-	-	-
13: 12	Key_SIZE [1: 0]	RW	2'h0	Set the key bit width: 0: 128 bit 1: Reserved 2: 256bit
11: 8	MAC_LEN [3: 0]	RW	4'h0	Set message verification code or initial vector length (only for CCM, CMAC, XCBC) In the standard specification, the length of the MAC is 4, 6, 8, 10, 12, 14, 16 bytes 0: 16 bytes 1: 1 byte 2: 2 bytes 3: 3 bytes 4: 4 bytes 5: 5 bytes 6: 6 bytes 7: 7 bytes 8: 8 bytes 9: 9 bytes 10:10 bytes 11: 11 bytes 12: 12 bytes 13: 13 bytes 14: 14 bytes 15: 15 bytes
7: 4	ALG_MODE [3: 0]	RW	4'h0	Select Mode: 0: ECB 1: CBC 2: CTR 3: CCM 4: CMAC 5: GCM 6 to 15: Res
3	DIR	RW	0	1: Encrypt 0: Decrypt
2	MSG_BEGIN	RW	0	Indicates that the current operation contains the first byte of information. The required key will be automatically obtained prior to the operation.
1	MSG_END	RW	0	Indicates that the current operation contains the last byte of information. If the result of the operation is a message verification code or an initial vector, it will be automatically stored in the key storage space after the operation is completed.
0	Aes_GO	RW	0	Start encryption and decryption operations. This bit and other control

				bits need to remain unchanged after starting operation until AES_DONE is set.

42.3.2 Key Control Register (AES_CTX_CFG)

Address offset: 0x04

Reset value: 0x0000_0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTX_INDEX												INV_KEY_STR	INV_KEY_RET	CTX_STR	CTX_RET
RW												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15: 4	CTX_INDEX [11: 0]	RW	12'h0	Context page number
3	INV_KEY_STR	RW	0	Store last round keys to key storage space
2	INV_KEY_RET	RW	0	Take out the previous round key
1	CTX_STR	RW	0	Store the current key in the key storage space
0	CTX_RET	RW	1	Take out the current key.

42.3.3 Information length register (AES_MSG_TOTAL_BYTES)

Address offset: 0x08

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	TOTAL_BYTES											
-	-	-	-	RW											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOTAL_BYTES															
RW															

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 0	TOTAL_BYTES [27: 0]	RW	28'h0	Set the length of the information, including all packets, excluding additional information. This is required in CCM and GCM modes.

42.3.4 Additional Information Length Register (AES_MSG_AAD_BYTES)

Address offset: 0x0C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD_LEN															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSG_LEN															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	ADD_LEN	RW	16'h0	Sets the length of additional information in the current operation. This is required in CCM and GCM modes.
15: 0	MSG_LEN	RW	16'h0	Sets the length of information in the current operation, including information and additional information. Total of all data entered into AES = Total of all data written into Ingress FIFO

42.3.5 GCM Mode Additional Information Length Register (AES_GCM_AAD_INFO)

Address offset: 0x10

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	ADD_LEN_TOT											
-	-	-	-	RW											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_LEN_TOT															
RW															

Bit	Name	R/W	Reset Value	Function
31: 28	Reserved	-	-	-
27: 0	ADD_LEN_TOT [27: 0]	RW	28'h0	Sets the total length of additional information, including all packets. This is required in GCM mode.

42.3.6 Key Register 0 (AES_CTX_KEY0)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_0															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_0															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_0	W	0	Key [31: 0]

42.3.7 Key register 1 (AES_CTX_KEY1)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_1															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_1															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_1	W	32'h0	Key [63: 32]

42.3.8 Key register 2 (AES_CTX_KEY2)

Address offset: 0x28

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_2															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_2															
W															

42.3.9 Key register 3 (AES_CTX_KEY3)

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_2	W	32'h0	Key [95: 64]

Address offset: 0x02C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_3															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_3															

W

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_3	W	32'h0	Key [127: 96]

42.3.10 Key register 4 (AES_CTX_KEY4)

Address offset: 0x30

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_4															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_4															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_4	W	32'h0	Key [159: 128]

42.3.11 Key register 5 (AES_CTX_KEY5)

Address offset: 0x34

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_5															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_5															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_5	W	32'h0	Key [191: 160]]

42.3.12 Key register 6 (AES_CTX_KEY6)

Address offset: 0x38

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_6															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_6															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_6	W	32'h0	Key [223: 192]

42.3.13 Key register 7 (AES_CTX_KEY7)

Address offset: 0x3C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY_7															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY_7															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	KEY_7	W	32'h0	Key [255: 224]

42.3.14 CBC mode key register 0 (AES_CTX_CBC_KEY0)

Address offset: 0x40

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBC_KEY_0															
W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBC_KEY_0															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CBC_KEY_0	W	32'h0	CBC Key [31: 0]

42.3.15 CBC mode key register 1 (AES_CTX_CBC_KEY1)

Address offset: 0x44

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBC_KEY_1															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBC_KEY_1															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CBC_KEY_1	W	32'h0	CBC Key [63: 32]

42.3.16 CBC mode key register 2 (AES_CTX_CBC_KEY2)

Address offset: 0x48

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBC_KEY_2															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBC_KEY_2															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CBC_KEY_2	W	32'h0	CBC Key [95: 64]

42.3.17 CBC mode key register 3 (AES_CTX_CBC_KEY3)

Address offset: 0x4C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBC_KEY_3															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CBC_KEY_3															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CBC_KEY_3	W	32'h0	CBC Key [127: 96]

42.3.18 CTR mode operator register 0 (AES_CTX_CTR0)

Address offset: 0x50

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCM_CTR_0															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCM_CTR_0															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CCM_CTR_0	W	32'h0	CTR/CCM Key [31: 0]

42.3.19 CTR mode operator register 1 (AES_CTX_CTR1)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCM_CTR_1															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCM_CTR_1															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CCM_CTR_1	W	32'h0	CTR/CCM Key [63: 32]

42.3.20 CTR mode operator register 2 (AES_CTX_CTR2)

Address offset: 0x58

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCM_CTR_2															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCM_CTR_2															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CCM_CTR_2	W	32'h0	CTR/CCM Key [95: 64]

42.3.21 CTR mode operator register 3 (AES_CTX_CTR3)

Address offset: 0x5C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCM_CTR_3															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCM_CTR_3															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	CCM_CTR_3	W	32'h0	CTR/CCM Key [127: 96]

42.3.22 Initialization Vector Register 0 (AES_CTX_IV0)

Address offset: 0x60

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV0															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV0															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	IV0	W	32'h0	Initial vector [31: 0]

42.3.23 Initialization Vector Register 1 (AES_CTX_IV1)

Address offset: 0x64

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV1															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV1															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	IV1	W	32'h0	Initial Vector [63: 32]

42.3.24 Initialization Vector Register 2 (AES_CTX_IV2)

Address offset: 0x68

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV2															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV2															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	IV2	W	32'h0	Initial Vector [95: 64]

42.3.25 Initialization Vector Register 3 (AES_CTX_IV3)

Address offset: 0x6C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV3															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV3															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	IV3	W	32'h0	Initial Vector [127: 96]

42.3.26 Message Verification Code Register 0 (AES_CTX_MAC0)

Address offset: 0x70

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAC_0															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_0															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	MAC_0	R	32'h0	Initial vector [31: 0]

42.3.27 Message verification code register 1 (AES_CTX_MAC1)

Address offset: 0x74

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAC_1															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_1															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	MAC_1	R	32'h0	Initial Vector [63: 32]

42.3.28 Message verification code register 2 (AES_CTX_MAC2)

Address offset: 0x78

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAC_2															
R															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_2															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	MAC_2	R	32'h0	Initial Vector [95: 64]

42.3.29 Message verification code register 3 (AES_CTX_MAC3)

Address offset: 0x7C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAC_3															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC_3															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	MAC_3	R	32'h0	Initial Vector [127: 96]

42.3.30 Input FIFO (AES_INGRESS_FIFO)

Address offset: 0x80

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INGRESS_FIFO															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INGRESS_FIFO															
W															

Bit	Name	R/W	Reset Value	Function
31: 0	INGRESS_FIFO	W	32'h0	AES input data FIFO entry

42.3.31 DMA input burst transfer setup register (AES_ING_DBCFG)

Address offset: 0x84

Reset value: 0xC000 0012

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLAINTEXT_REQ.	DMA_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R/W	R/W	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DST_MSIZE			DST_TR_WIDTH		
-	-	-	-	-	-	-		-	-	R/W	R/W	R/W	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	PLAINTEXT_REQ	R/W	1	This bit sets whether incoming data is required for the current AES mode
30	DMA_EN	R/W	1	This bit sets whether the input data is passed in by DMA or CPU 0: CPU 1: DMA
29: 6	Reserved	-	-	-
5: 3	DST_MSIZE	R/W	2'h2	DMA's target transmission burst transmission length: 0: 1 1: 4 2: 8 3: 16 4: 32 5: 64 6: 128
2: 0	DST_TR_WIDTH	R/W	2 'h 2	DMA target transmission width: 0: 8 bit 1: 16bit 2: 32bit

42.3.32 DMA output length register (AES_DMA_ENG_LEN)

Address offset: 0x88

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENGRESS_DMA_DATA_LENGTH															
R/W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENGRESS_DMA_DATA_LENGTH															
R/W															

Bit	Name	R/W	Reset Value	Function
31: 0	ENGRESS_DMA_DATA_LENGTH	R/W	32'h0	Sets the total length of data input from DMA to AES, which is used to generate the handshake signal.

42.3.33 DMA Output Burst Transfer Setup Register (AES_ENG_DBCFG)

Address offset: 0x8C

Reset value: 0xC000 0012

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CIPERTEXT_REQ.	DMA_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R/W	R/W	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SRC_MSIZE			SRC_TR_WIDTH		
-	-	-	-	-	-	-		-	-	R/W	R/W	R/W	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	CIPERTEXT_REQ	R/W	1	This bit sets whether outgoing data is required for the current AES mode
30	DMA_EN	R/W	1	This bit sets whether the input data is passed in by DMA or CPU 0: CPU 1: DMA
29: 6	Reserved	-	-	-
5: 3	SRC_MSIZE	R/W	2'h2	DMA's target transmission burst transmission length: 0: 1 1: 4 2: 8 3: 16 4: 32 5: 64 6: 128
2: 0	SRC_TR_WIDTH	R/W	2'h2	DMA target transmission width: 0: 8-bit 1: 16-bit 2: 32-bit

42.3.34 DMA input length register (AES_DMA_ING_LEN)

Address offset: 0x90

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INGRESS_DMA_DATA_LENGTH															
R/W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INGRESS_DMA_DATA_LENGTH															
R/W															

Bit	Name	R/W	Reset Value	Function
31: 0	INGRESS_DMA_DATA_LENGTH	R/W	32'h0	Sets the total length of data input from DMA to AES, which is used to generate the handshake signal.

				This is needed even if the CPU is used to transfer data instead of DMA.

42.3.35 Input FIFO status register (AES_INGF_STATUS)

Address offset: 0x94

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														ING_FULL	ING_EMPTY
-														R	R

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	ING_FULL	R	0	Indicates that input data FIFO is full
0	ING_EMPTY	R	1	Indicates that the input data FIFO is empty

42.3.36 Output FIFO (AES_ENGRESS_FIFO)

Address offset: 0x98

Reset value: 0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENGRESS_FIFO															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENGRESS_FIFO															
R															

Bit	Name	R/W	Reset Value	Function
31: 0	INGRESS_FIFO	R	X	AES input data FIFO entry

42.3.37 Output FIFO status register (AES_ENGFIFO_STATUS)

Address offset: 0x9C

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														ENG_FULL	ENG_EMPTY
-														R	R

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	ENG_FULL	R	0	Indicates that input data FIFO is full
0	ENG_EMPTY	R	1	Indicates that the input data FIFO is empty

42.3.38 DMA input completion status register (AES_ING_DMA_DONE)

Address offset: 0XA0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															DMA_INGRESS_DONE
-															

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	DMA_INGRESS_DONE	R	0	Indicates that the DMA input has completed transmission.

				Read clearing

42.3.39 DMA output completion status register (AES_ENG_DMA_DONE)

Address offset: 0XA4

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															DMA_ENGRESS_DONE
-															

Bit	Name	R/W	Reset Value	Function
31: 1	Reserved	-	-	-
0	DMA_NGRESS_DONE	R	0	Indicates that the DMA output has completed transmission. Read clearing

42.3.40 Completion status register (AES_DONE_STATUS)

Address offset: 0XA8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														MAC_VAILD	AES_DONE
-															

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1	MAC_VAILD	RC / W0	0	Indicates whether the MAC authentication was successful This bit is valid only after AES_DONE is set, Automatically clears after AES_GO is set Write 0 to clear the interrupt.
0	AES_DONE	RC / W0	0	Indicates that the operation has completed Automatically clears after AES_GO is set Write 0 to clear the interrupt.

43. CORDIC Coprocessor (CORDIC)

43.1 Overview

CORDIC coprocessors provide hardware acceleration of mathematical functions (mainly trigonometric, hyperbolic) commonly used in motor control, metrology, signal processing, and many other applications.

Compared to software implementations, it speeds up the computation of these functions so that the processor can use lower operating frequencies or use no processor resources to perform other tasks.

43.1.1 Main features

The main features are as follows:

- 24-bit CORDIC
- Circumferential and hyperbolic systems
- Rotation and vector modes
- Functions: sine, cosine, sinh, cosh, atan, atan2, atanh, modulus, square root, natural
- Programmable accuracy
- Low latency AHB interface
- After entering the required data, the results can be read directly (bus suspended) without polling or interrupting
- It also supports polling, interrupt, DMA and other reading results methods
- DMA read and write channel
- Multiple register reads/writes via DMA

43.2 Functional description

43.2.1 Overview

CORDIC is an efficient stepwise approximation algorithm for computing trigonometric and hyperbolic functions.

In triangular (circular) mode, the sine and cosine of the angle are determined θ by rotating the unit vector [1, 0] through a gradually decreasing angle until the cumulative sum of the rotated angles is equal to the input angle θ. The rotation vectors x and y correspond to θ the cosine and sine of, respectively. In turn, the angle of $arctangent\,(y/x)$ the vector [x, y] is that the unit vector [1, 0] is obtained by rotating [x, y] by a decreasing angle. The cumulative sum of the rotation angles is the angle of the original vector.

The CORDIC algorithm can also be used to compute hyperbolic functions (sinh, cosh, atanh) by replacing a continuous circular rotation with a rotation along a hyperbola.

Other functions can be derived from the basic functions described above.

43.2.2 CORDIC function

The first step in using a coprocessor is toselect the desired function by programmingthe CORDIC_CSRregisterFUNCaccordingly.

Table 43 -1 List of functions supported by the CORDIC coprocessor

Function	Primary argument (ARG1)	Secondary argument (ARG2)	Primary result (RES1)	Secondary result (RES2)
Cosine	Angle θ	modulus m	m * cosθ	m * sinθ
Sine	Angle θ	modulus m	m * sinθ	m * cosθ
Phase	x	y	atan2 (y, x)	$\sqrt{x^2+y^2}$
Modulus	x	y	$\sqrt{x^2+y^2}$	atan2 (y, x)
Arctangent	x	none	tan-1 x	none
Hyperbolic cosine	x	none	cosh x	sinh x
Hyperbolic sine	x	none	sinh x	cosh x
Hyperbolic arctangent	x	none	$\tanh^{-1}$ x	none
Natural logarithm	x	none	ln x	none
Square root	x	none	$\sqrt{x}$	none

There are several functions that require two input arguments (ARG1 and ARG2), and some functions generate two results simultaneously (RES1 and RES2). For example, when converting from polar coordinates to rectangular coordinates: $\sin\theta$ also generates $\cos\theta$, and $\cos\theta$ also generates $\sin\theta$. The same is true for rectangular to polar transformations (phase (x, y), modulus (x, y)) and hyperbolic functions ($\cosh\theta$, sinhθ).

Note: The exponential function, exp x, can be taken as the sum of sinh x and cosh x.

Furthermore, the logarithm of base N, logN x, can be obtained by multiplying ln x by the constant K, where K = 1/ln N.

For some functions (atan, log, sqrt), a scaling factor (see Section 43.2.4) can be applied to extend the range of the function beyond the maximum [-1, 1] range supported by the q1.31 fixed point format. The scaling factor must be set to 0 for all other circular functions and 1 for hyperbolic functions.

43.2.2.1 Cosine

Table 43-2 Cosine parameters

Parameter	Description	Range
ARG1	The angle θ in radians and has been divided by π.	[-1, 1]
ARG2	Modulus m	[0, 1]
RES1	m * cosθ	[-1, 1]
RES2	m * sinθ	[-1, 1]
SCALE	Not applicable	0

This function is used to calculate the cosine of an angle ranging from-πtoπ. It can also be used to perform transformations from polar coordinates to rectangular coordinates.

The main parameter is the angle in radiansθ. Before programming ARG1, it must be divided byπ.

The second parameter is the modulus m. If m is greater than 1, it must be scaled in software to adapt it to the q1.31 range of ARG2.

The main result, RES1, is the cosine of the angle, multiplied by the modulus m.

The secondary result, RES2, is the sine of the angle, multiplied by the modulus m.

43.2.2.2 Sine

Table 43-3 Sine parameters

Parameter	Description	Range
ARG1	The value of the angle θ in radians divided by π	[-1, 1]
ARG2	Modulus m	[0, 1]
RES1	m * sinθ	[-1, 1]
RES2	m * cosθ	[-1, 1]
SCALE	Not applicable	0

This function is used to calculate the cosine of an angle ranging from-πtoπ. It can also be used to perform transformations from polar coordinates to rectangular coordinates.

The main parameter is the angle in radiansθ. Before programming ARG1, it must be divided byπ.

The second parameter is the modulus m. If m is greater than 1, it must be scaled in software to adapt it to the q1.31 range of ARG2.

The primary result, RES1, is the sine of the angle, multiplied by the modulus m.

The secondary result, RES2, is the cosine of the angle, multiplied by the modulus m.

43.2.2.3 Phase

Table 43-4 Phase parameters

Parameter	Description	Range
ARG1	x coordinate	[-1, 1]
ARG2	y-coordinate	[-1, 1]
RES1	The value of phase angle in radians θdivided by π	[-1, 1]
RES2	Modulus m	[0, 1]
SCALE	Not applicable	0

43.2.2.4 Modulus

Table 43-5 Modulus parameters

Parameter	Description	Range
ARG1	x coordinate	[-1, 1]
ARG2	y-coordinate	[-1, 1]
RES1	Modulus m	[0, 1]
RES2	The value of phase angle expressed in radians θdivided by π	[-1, 1]
SCALE	Not applicable	0

This function calculates the size, or modulo, of the vector v = [x y]. It can also be used to perform rectangular coordinate to polar coordinate conversion.

The main parameter is the x-coordinate, i.e. the magnitude of the vector in the x-axis direction. If | x | > 1, scaling must be applied in the software to fit the q1.31 range of ARG1.

The secondary parameter is the y-coordinate, i.e. the magnitude of the vector in the y-axis direction. If | y | > 1, scaling must be applied in the software to fit the q1.31 range of ARG2.

The main result, RES1, is the modulus and is given by: $|\mathrm{v}| = \sqrt{\mathrm{x}^2 + \mathrm{y}^2}$. RES1 is limited to 1.

The secondary result, RES2, is the phase angle $\theta$of the vector v. RES2 must be πmultiplied to get the angle in radians. Note that due to the cyclic nature of the phase angle, πclose values may sometimes circle back to-π.

43.2.2.5 Arctangent

Table 43-6 Arctangent parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[-1, 1]
ARG2	Not applicable	-
RES1	The value of $2^{-n} * \tan^{-1}$ x divided by π in radians	[-1, 1]
RES2	No	-
SCALE	n	[0 7]

This function calculates the arctangent of the input argument x.

The main parameter ARG1 is the input value x = tanθ. If | x | > 1, a scaling factor of 2^{-n} must be applied in the software such that-1 < x · 2-n < 1. The scaled value $x \cdot 2^{-n}$ is programmed into ARG1, and the scaling factor n must be programmed into the SCALE parameter.

Note that the maximum input value allowed is tan = θ128, corresponding to angleθ = 89.55 degrees. For | x | > 128, a software method must be used to find tan-1 x.

Secondary parameter, ARG2, is not used.

The primary result, RES1, is angleθ = tan − 1 x. RES1 must be multiplied by 2n π to get the angle in radians.

Secondary outcome, RES2, not used.

43.2.2.6 Hyperbolic cosine

Table 43-7 Hyperbolic cosine parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[-0.559 0.559]
ARG2	Not applicable	
RES1	$2^{-n} *$ coshx	[0.5 0.846]
RES2	$2^{-n} *$ sinhx	[-0.683 0.683]
SCALE	n	1

This function calculates the hyperbolic cosine of the hyperbolic angle x. It can also be used to calculate exponential functions ex = cosh x + sinh x, and e-x = cosh x-sinh x.

The main parameter is the hyperbolic angle x. Only x values in the range of-1.118 to +1.118 are supported. Since the minimum value of cosh x is 1, which is outside the range of the q1.31 format, a scaling factor of 2^{-n} must be applied in the software. The scaling factor n = 1 must be programmed into the SCALE parameter.

Secondary parameters are not used.

The primary result, RES1, is the hyperbolic cosine, cosh x. RES1 must be multiplied by 2 to get the correct result.

The secondary result, RES2, is the hyperbolic sine, sinh x. RES2 must be multiplied by 2 to get the correct result.

43.2.2.7 Hyperbolic sine

Table 43-8 Hyperbolic sine parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[-0.559, 0.559]
ARG2	Not applicable	-
RES1	$2^{-n} * \sinh x$	[-0.683, 0.683]
RES2	$2^{-n} * \cosh x$	[0.5, 0.846]
SCALE	n	1

This function calculates the hyperbolic cosine of the hyperbolic angle x. It can also be used to calculate exponential functions ex = cosh x + sinh x, and e-x = cosh x-sinh x.

The main parameter is the hyperbolic angle x. Only x values in the range of-1.118 to +1.118 are supported. Since the minimum value of cosh x is 1, which is outside the range of the q1.31 format, a scaling factor of 2^{-n} must be applied in the software. The scaling factor n = 1 must be programmed into the SCALE parameter.

Secondary parameters are not used.

The primary result, RES1, is the hyperbolic cosine, sinh x. RES1 must be multiplied by 2 to get the correct result.

The secondary result, RES2, is the hyperbolic sine, cosh x. RES2 must be multiplied by 2 to get the correct result.

43.2.2.8 Hyperbolic arctangent

Table 43-9 Hyperbolic arctangent parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[-0.403 0.403]
ARG2	Not applicable	-
RES1	$2^{-n} * \text{atanh } x$	[-0.559 0.559]
RES2	Not applicable	-
SCALE	n	1

This function calculates the hyperbolic arc tangent of the input argument x.

The main argument is the input value x. Only x values in the range of-0.806 to +0.806 are supported. The value of x must be scaled by a factor of 2-n, where n = 1. The scaled value x * 0.5 is programmed into ARG1 and the scaling factor n = 1 must be programmed into the SCALE parameter.

Secondary parameters are not used.

The primary outcome is the hyperbolic arc tangent value atanh x. RES1 must be multiplied by 2 to get the correct value.

Secondary results were not used.

43.2.2.9 Natural logarithm

Table 43-10 Natural logarithm parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[0.054 0.875]
ARG2	Not applicable	-
RES1	$2-(n+1) * \ln x$	[-0.279 0.137]
RES2	Not applicable	-
SCALE	n	[1 4]

This function calculates the natural logarithm of the input argument x.

The main argument is the input value x. Only x values in the range of 0.107 to 9.35 are supported.

The value of x must be scaled by a factor of 2^{-n} such that $x * 2^{-n} < 1-2^{-}n$. The scaled value $x * 2^{-n}$ is programmed into ARG1 and the scaling factor n must be programmed into the SCALE parameter.

The following table lists the valid scaling factor n, along with the corresponding ranges of x and ARG1.

Table 43-11 Scaling factors in natural logarithm and their corresponding ARG1 and x ranges

n	x range	ARG1 range
1	0.107 ≤ x < 1	0.0535 ≤ ARG1 < 0.5
2	1 ≤ x < 3	0.25 ≤ ARG1 < 0.75
3	3 ≤ x < 7	0.375 ≤ ARG1 < 0.875
4	7 ≤ x ≤ 9.35	0.4375 ≤ ARG1 ≤ 0.584

Secondary parameters are not used.

The primary outcome is the natural logarithm, ln x. RES1 must be multiplied by $2-(n+1)$ to get the correct value.

Secondary results were not used.

43.2.2.10 Square root

Table 43-12 Square root parameters

Parameter	Description	Range
ARG1	$x * 2^{-n}$	[0.027 0.875]
ARG2	Not applicable	-
RES1	$\sqrt{x} * 2-n$	[0.04 1]
RES2	Not applicable	-
SCALE	n	[0 2]

This function calculates the square root of the input argument x.

The main argument is the input value x. Only x values in the range of 0.027 to 2.34 are supported.

The value of x must be scaled by a factor of 2^{-n} such that $x \cdot 2-n < (1-2^{(-n-2)})$.

The scaled value $x \cdot 2^{-n}$ is programmed into ARG1, and the scaling factor n must be programmed into the SCALE parameter.

Table 43-13 The scaling factor in the square root of the table and its corresponding ARG1 and x ranges

n	x range	ARG1 range
0	0.027 ≤ x < 0.75	0.027 ≤ ARG1 < 0.75
1	0.75 ≤ x < 1.75	0.375 ≤ ARG1 < 0.875
2	1.75 ≤ x ≤ 2.341	0.4375 ≤ ARG1 ≤ 0.585

Secondary parameters are not used.

The primary result is the square root of x. RES1 must be multiplied by 2^{-n} to get the correct value.
Secondary results were not used.

43.2.3 Fixed-point representation

CORDIC operates in a fixed-point signed integer format. The input and output values can be in q1.31 or q1.15 format.

In q1.31 format, numbers are represented by one sign bit and 31 decimal places (binary decimal places). Therefore, the range of values is-1 (0x80000000) to 1-2-31 (0x7FFFFFFF).

In the q1.15 format, the values range from-1 (0x8000) to 1-2-15 (0x7FFF). The advantage of this format is that the two q1.15 input parameters can be packed into a 32-bit write and the two results can be fetched in a 32-bit read.

43.2.4 Scaling factor

Several of the functions listed in the CORDIC functions section specify a scaling factor SCALE. This allows the input range of the function to be extended to the range supported by CORDIC without overflowing the input, output, or internal registers. If a scaling factor is required, it must be calculated in software and programmed into the SCALE fieldofthe CORDIC_CSR register. Before writing to the CORDIC_WDATA register, the input parameters must be scaled accordingly. After reading the result from the CORDIC_RDATA register, the result must also be scaled.

Note: The scaling factor causes a loss of scaling value accuracy.

43.2.5 Accuracy

The accuracy of the results depends on the number of CORDIC iterations. See picture below.

CORDIC can perform two iterations per clock cycle. For each function, the following table represents the maximum error remaining after every four iterations, and the number of clock cycles needed to achieve this accuracy. From this table, the required number of cyclescan be determined and programmed in the precision field of the CORDIC_CSR register.

Once the configured number of iterations is completed, the coprocessor stops immediately and the result can be read immediately.

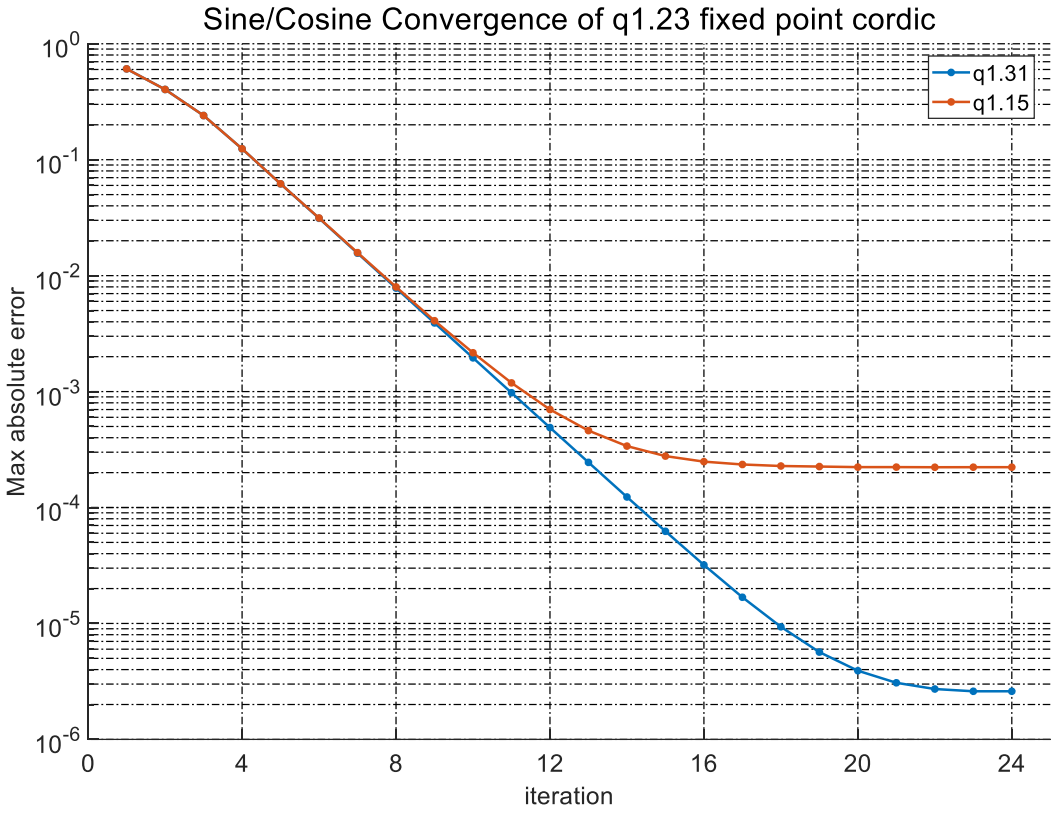


Table 43-14 precision and number of iterations

Function	Number of iterations	Number of cycles	Max residual error (1)	
			q1.31 format	q1.15 format
Sin, Cos, Phase (2), Mod, Atan (4)	4	1	2^{-3}	2^{-3}
	8	2	2^{-7}	2^{-7}
	12	3	2^{-11}	2^{-11}
	16	4	2^{-15}	2^{-15}
	20	5	2^{-18}	2^{-16}
	24	6	2^{-19}	2^{-16}
Sinh, Cosh, Atanh, Ln (3)	4	1	2^{-2}	2^{-2}
	8	2	2^{-6}	2^{-6}
	12	3	2^{-10}	2^{-10}
	16	4	2^{-13}	2^{-13}
	20	5	2^{-17}	2^{-15}
	24	6	2^{-18}	2^{-15}
Sqrt (4)	4	1	2^{-7}	2^{-7}
	8	2	2^{-14}	2^{-14}
	12	3	2^{-19}	2^{-15}

1. The maximum residual error is the maximum error that performs the same calculation as a double-precision floating-point number after a given number of iterations. There may be additional rounding errors up to 2 ^-16 for the q15 format and 2 ^-20 for the q31 format.

2. For modulus greater than 0.5, the achievable accuracy decreases proportionally with the magnitude of the modulus, as the quantization error becomes significant.
3. SCALE = 1. If a higher scaling factor is used, the accuracy that can be achieved is proportionally reduced.
4. SCALE = 0. If a higher scaling factor is used, the accuracy that can be achieved is proportionally reduced.

43.2.6 Zero-overhead mode

The fastest way to use a coprocessor is to pre-programthe CORDIC_CSR register, set the function to be executed (FUNC), the number of clock cycles required (PRECISION), the size of input and output values (ARGSIZE, RESSIZE), the number of input parameters (NARGS)and/orthe number of results (NRES), and, if needed, set the scaling factor (SCALE). The calculation is then triggered by writing the input parameters to the CORDIC_WDATA register. Once the correct number of input parameters have been written (and any ongoing calculations have been completed), a new calculation is started using these input parameters and the current CORDIC_CSR settings. If there is no change, there is no need to reprogram the CORDIC_CSR register.

If dual 32-bit input parameters are required (ARGSIZE = 0, NARGS = 1), the primary input parameter ARG1 must be written first and then the secondary input parameter ARG2. If the secondary parameters remain the same in a series of calculations, the second write can be avoided by reprogramming the number of parameters to one (NARGS = 0) after the first calculation has started. As long as the FUNC has not changed, the secondary argument retains its programmed value.

Note: After reset, ARG2 is set to +1 (0x7FFFFFFF).

If you use two 16-bit parameters (ARGSIZE = 1), you must package them into a 32-bit word, with ARG1 in the least significant half-word and ARG2 in the most significant half-word. The packed32-bit wordis then written tothe CORDIC_WDATA register. Only one write is required in this case (NARGS = 0).

For functions that take only one input parameter ARG1, it is recommended to set NARGS = 0.

IfNARGS = 1,a second write toCORDIC_WDATAmust be performed to trigger the calculation. ARG2 data is not used in this case.

Once the calculation begins, any attempt to readthe CORDIC_RDATA registeris inserted into the bus wait state until the calculation is complete, and then the result is returned. So the software can write the input and read the result immediately without polling to see if it's valid. Alternatively, the processor may wait for an appropriate number of clock cycles before reading the result. If desired, this time can be used to programthe CORDIC_CSR registerfor the next calculation and prepare the next input data. The CORDIC_CSR registercan be reprogrammed during the calculation without affecting the result of the ongoing calculation. In the same way, once the previous parameter is configured,the CORDIC_WDATA registercan be updated with the next parameter. The next parameter and settings take effect until the previous calculation is complete.

After the calculation is complete,the resultcan be read fromthe CORDIC_RDATA register.

If two 32-bit results are required (NRES = 1, RESSIZE = 0), the primary result (RES1) is read first, and then the secondary result (RES2) is read. If only one 32-bit result is expected (NRES = 0, RESSIZE = 0), only the primary result (RES1) needs to be read.

If a16-bit resultis required (RESSIZE = 1),a single read ofCORDIC_RDATAgets two results packed into a32-bit word. RES1 is located in the lower half word and RES2 is located in the upper half word. In this case, it is recommended to program NRES = 0. IfNRES = 1, a second read ofCORDIC_RDATAmust be performedin order to releaseCORDICfor the next operation. Data from the second read must be discarded.

If the expected number of parameters have been written, the next calculation begins after the expected number of results have been read. This means that at any time, there may be a calculation in progress, or waiting to read the result, and an action pending. When an operation is in the pending state, ifCORDIC_WDATAis further accessed, the pending operation will be canceled and the related data will be overwritten.

The following sequence summarizes the use of CORDIC in zero overhead mode:

1.Setthe CORDIC_CSR register.

2.Configurethe parameters of the first calculation in the CORDIC_WDATA register. This will initiate the first calculation.

3.If necessary, updatethe CORDIC_CSRregister to set the configuration for the next calculation.

4.Readthe resultfromthe CORDIC_RDATA register.

5.Configure the parameters for the next calculationin the CORDIC_WDATA register.

6. Go to step 3.

43.2.7 Polling mode

The RRDY flaginthe CORDIC_CSR registeris set whenthere is a new resultin the CORDIC_RDATA register. The flag can be polled by reading the register. Reset the CORDIC_RDATA register by reading itonce or twice, depending onthe NRES fieldofthe CORDIC_CSR register.

When the result is read immediately when it is no longer available, pollingthe RRDY flagtakes slightly longer than readingthe CORDIC_RDATAregister directly.

43.2.8 Interrupt mode

By settingthe interrupt enable(IE)bitin the CORDIC_CSR register, an interrupt is generated whenever the RRDY flagis generated. When the flag resets, the interrupt is cleared.

This mode allows the calculation result to be read under the interrupt service routine and therefore has a priority over other tasks. However, it is slower than reading results directly or polling flags due to interrupt processing delays.

43.2.9 DMA mode

Ifthe DMAwrite enable(DMAWEN)bit inthe CORDIC_CSR registeris set and there are no pending operations,a DMA write requestis generated. The DMA controllercan transfer the main input parameter(ARG1)tothe CORDIC_WDATA register.

IfNARGS = 1inthe CORDIC_CSR register,a second DMA write requestis generated to transferthe auxiliary input parameter (ARG2)intothe CORDIC_WDATA register. When all input parameters have

been written, and any ongoing calculations have been completed (by reading the results), a new calculation is started and another DMA write request is generated.

Ifthe DMA Read Enable(DMAREN)bitinthe CORDIC_CSR registeris set,a DMA read requestis generated whenthe RRDY flagis set. The DMA controllercanthen read the primary result (RES1)fromthe CORDIC_RDATA register. IfNRES = 1inthe CORDIC_CSR register,a second DMA requestis generated to read out the auxiliary result(RES2). When all results have been read, the RRDY flag is cleared.

Note: SinceDMAperformsaccess tothe CORDIC_WDATAorCORDIC_RDATAregister,each DMA requestmust be acknowledged. Therefore, when DMAread is enabled,the CPUmust be prevented from accessingthe CORDIC_RDATA register. Similarly, the processor must avoid accessingthe CORDIC_WDATA registerwhenDMA writeis enabled

43.3 TIMx registers

The register width is 32bits.

43.3.1 CORDIC Control/Status Register(CORDIC_CSR)

Address: 0x00

Reset value: 0x0000 0050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RRDY.	Res								ARGSIZE	RESSIZE	NARGS	NRES	DMAWEN	DMAREN	IEN
R									RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					SCALE [2: 0]			PRECISION [3: 0				FUNC [3: 0]			
					RW			RW				RW			

Bit	Name	R/W	Reset Value	Function
31	RRDY	R	1'b0	Result ready flag: 0: No new data in the output register. 1:The CORDIC_RDATA registercontains new data. This bit is set by the hardware when the CORDIC operation is complete. Whenthe CORDIC_RDATA registeris read(NRES+1)times, it is reset by hardware.
30: 23	Reserved	-	-	The reset value must be maintained
22	ARGSIZE	RW	1'b0	Width of input data: 0: 32 bits 1: 16 bits ARGSIZE selects the number of bits used to represent the input data. If32-bit datais selected,the CORDIC_WDATA registerrequires parameters inq1.31format. If16-bit datais selected,the CORDIC_WDATA registerrequires parameters inq1.15format. The primary argument (ARG1) writes the least significant half-word and the secondary argument (ARG2) writes the most significant half-word.
21	RESSIZE	RW	1'b0	Width of output data: 0: 32 bits 1: 16 bits RESSIZE selects the number of bits used to represent the output data. If32-bit datais selected,the CORDIC_RDATA registercontainsthe results inq1.31format. If you select 16-bit data, then The least significant half-word of CORDIC_RDATA contains the primary result (RES1)inq1.15format, and the most significant half-word containsthe secondary result (RES2)also inq1.15format.
20	NARGS	RW	1'b0	CORDIC_WDATAregister Number of parameters expected: 0: Only one 32-bit write is required for the next calculation (two 16-bit values if ARGSIZE = 1). 1: Two32-bit valuesmust be written tothe CORDIC_WDATAregister to trigger the next calculation.

				Read returns the current state of the bit.
19	NRES	RW	1'b0	Number of results inCORDIC_RDATA register: 0: Only one32-bit value (or two16-bit values ifRESSIZE = 1) is transferred tothe CORDIC_RDATA registerwhen the next calculation is complete. Reading oncefromCORDIC_RDATAresetsthe RRDY flag. 1: When the next calculation is complete, two32-bit values will be transferred tothe CORDIC_RDATA register. Two readsfromCORDIC_RDATAare required to resetthe RRDY flag. Read returns the current state of the bit.
18	DMAWEN	RW	1'b0	Enable DMA write channel: 0: Disabled. No DMA write request is generated. 1: Enable. As long as there are no pending operations, requests are generated on the DMA write channel. This bit is set and cleared by software. Read returns the current state of the bit.
17	DMAREN	RW	1'b0	Enable DMA read channel: 0: Disabled. No DMA read request is generated. 1: Enable. As long as the RRDY flag is set, a request is generated on the DMA read channel. This bit is set and cleared by software. Read returns the current state of the bit.
16	IEN	RW	1'b0	Enable interrupt: 0: Disabled. No interrupt request is generated. 1: Enable. An interrupt request is generated whenever the RRDY flag is set. This bit is set and cleared by software. Read returns the current state of the bit.
15: 11	Reserved	-	-	The reset value must be maintained
10: 8	SCALE [2: 0]	RW	3'b0	Scaling Factor The value of this field is the scaling factor applied to the parameter and/or result. The value n means that the argument has been multiplied by a factor of 2 ^ n, and the result needs to be multiplied by 2n. Refer to the relevant sections for the applicability of the scale factor for each function and the appropriate range
7: 4	PRECISION [3: 0	RW	4'b0101	Required precision (number of iterations): 1 to 6: (number of iterations)/4. Then the number of iterations is PRECISION times 4. To determine the number of iterations required for a given precision, see the relevant table. Note that the recommended range for this field is 3 to 6 for most functions In addition, when PRECISION = 0 or greater than 6, the number of iterations is 24
3: 0	FUNC [3: 0]	RW	4'b0	Function: 0: Cosine 1: Sine Phase 2 3 Modulus 4 Arctangent 5: Hyperbolic cosine 6: Hyperbolic sine 7: Arctanh Natural logarithm 9: Square Root 10 to 15: Reserved

43.3.2 CORDICparameter register(CORDIC_WDATA)

Address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARG [31: 16]															
W															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARG [15: 0]															
W															

Bit	Name	R/W	Reset Value	Function
31: 16	ARG [31: 16]	W	16h '0	Function input parameters: This register is programmed usingthe input parameters oftheselected function inthe FUNC fieldofthe CORDIC_CSR register. If you select a32-bit format(CORDIC_CSR.ARGSIZE = 0)and you need two input parameters(CORDIC_CSR.NARGS = 1), you need to make two consecutive writes to this register. The first writes the primary parameter (ARG1) and the second writes the secondary parameter (ARG2). If the 32-bit format is selected and only one input parameter is required (NARGS = 0), only one write is required to this register, which contains the main parameter (ARG1). If you select a16-bit format(CORDIC_CSR.ARGSIZE = 1), a single write to this register contains two parameters. The primary parameter (ARG1) was located in the lower half ARG [15: 0] and the secondary parameter (ARG2) was located in the upper half ARG [31: 16]. In this case, NARGS must be set to 0. Please see the CORDIC Functions section for the parameters required for each function and its allowed ranges. After writing the required number of parameters,CORDICevaluates the function specified by-CORDIC_CSR.FUNCusing the input parameters provided that the previous calculation has been completed. If the calculation is in progress, the ARG1 and ARG2 values remain pending until the calculation is complete and the result is read. During this time, writes to registers cancel pending operations and overwrite parameter data.
15: 0	ARG [15: 0]	W	16h '0	

43.3.3 CORDICresult register (CORDIC_RDATA)

Address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES [31: 16]															
R															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RES [15: 0]															
R															

Bit	Name	R/W	Reset Value	Function
31: 16	RES [31: 16]	R	16h '0	Function ResultIf a32-bit formatis selected(CORDIC_CSR.RESSIZE = 0)and two output values are required(CORDIC_CSR.NRES = 1),the register must be read twice whenthe RRDY flagis set. The first read gets the primary result (RES1). The second read gets the secondary result (RES2) and resets the RRDY. If the 32-bit format is selected and only one output value is needed (NRES = 0), this register only needs to be read once to get the primary result (RES1) and reset the RRDY flag. If the16-bit formatis selected(CORDIC_CSR.RESSIZE = 1), this register contains the primary result(RES1) RES [15: 0] for the lower half, and the secondary result(RES2) RES [31: 16] forthe upper half. In this case, NRES must be set to 0 and only one read is performed. Reading data from this register resetsthe RRDY flaginthe CORDIC_CSR register
15: 0	RES [15: 0]	R	16h '0	

44. 6800/8080 Controller (LCDC)

44.1 Overview

The LCDC is used to drive the 8080/6800 protocol LCD display.

The LCDC module has the flexibility to write instructions/data to and read data from the LCD display.

The LCDC module can configure the timing of read/write operations separately (the duration of read/write drive pulses can be configured separately) to adapt to different LCD displays.

44.2 Main features

- Programmable read-write pulse width
- Supports 8080/6800 LCD displays with data bit width less than or equal to 18 bits, including common data bit widths such as 8/9/16/18
- Supports driving up to 4 LCD displays (in time division multiplexing mode), and the timing parameters of the 4 displays, the protocol can be configured separately
- Reading and writing to a data register is equivalent to reading and writing to an LCD
- Supports massive data writing to LCD using DMA without consuming CPU resources

44.3 Functional description

44.3.1 Enable the LCDC

The EN0-3 bits of the LCDC_CSR are written by 1, that is, LCD0-3 are enabled; When 0 is written to EN0 to 3 bits of the LCDC_CSR, LCD0 to 3 are disabled.

If the LCDC is in a disabled state, writing data to the LCDx_DR/LCDx_CMDR (x = 0-3) register does not trigger a write data/instruction operation to the LCD, data read from the LCDx_DR register is an invalid value, and does not trigger a read data operation to the LCD.

Control logic of CS signal line (chip selection line): When LCDx is working, CSx is at low level (at this time, other CS lines are at high level), and when LCDx is not working, CSx is at high level.

44.3.2 Operating mode

When the MODEx bit of LCD_CSR is 1, LCDx operates in 6800 mode, otherwise it is 8080 mode. After reset, the LCDx works in 8080 mode by default.

44.3.3 Send LCD data

If LCDx is in the enabled state, you only need to write the LCDx_DR register to trigger LCDx data sending. PinLCDC _RSishigh, and D [17: 0] will be recognized as data.

When reading and writing LCDx_DR, timing parameters will be configuredaccording to LCDx_WCFGR (write operation) and LCDx_RCFGR (read operation).

When the LCD is enabled and the DMA mode is enabled, if the LCDC does not transmit data, it will send a write request to the DMA, and if the DMA responds to the write request, the LCDC will take the data sent from the DMA as LCD write data and send it to the pin. While the LCDC is transmitting data, the LCDC will stop sending write requests.

44.3.4 Send LCD command

If LCDx is in the enabled state, you only need to write the LCDx_CMDR register to trigger the LCDx instruction sending.PinLCDC_RSis low, andD [17: 0] will be recognized as the instruction.

When reading and writing LCDx_CMDR, timing parameters will be configuredaccording to LCDx_WCFGR (write operation) and LCDx_RCFGR (read operation).

If LCDx is enabled, you only need to read the LCDx_DR register to read the LCD display.

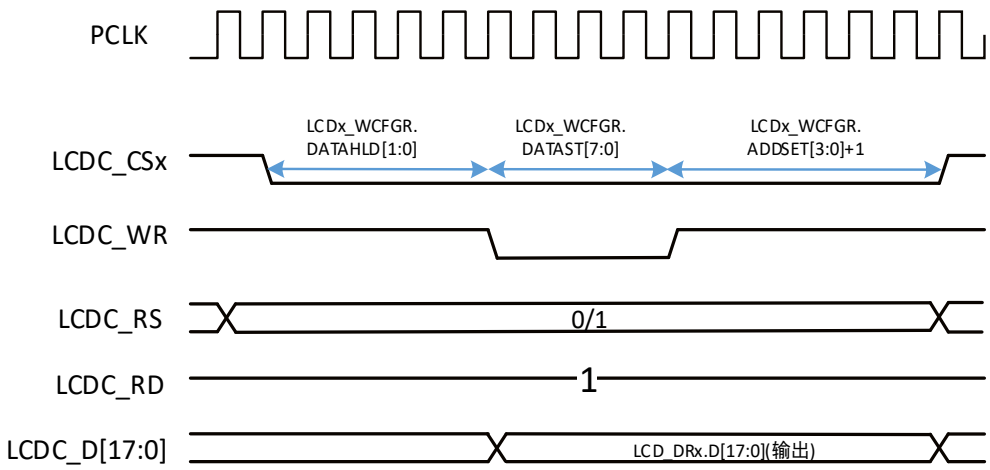
44.3.5 Read instructions from LCD

LCDC supports read instruction operation. When LCDx is in the enabled state, it only needs to read the LCDx_CMDR register totriggerthe LCDx instruction read. Different from data reading, when reading the instruction, the pin LCDC_RS is low.

Note: Some models of LCDs can read the device ID or status register directly in this way without writing instructions in advance.

44.3.6 LCDC timing

(1) LCDx 8080 write data/instruction timing

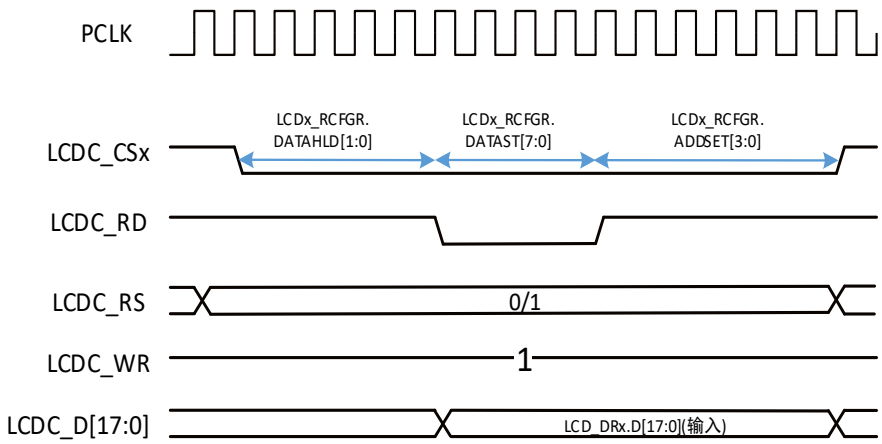


When LCDx_DR is written, an 8080/6800 write data operation is triggered; When LCDx_CMDR is written, an 8080/6800 write command operation is triggered. If the LCD_CSR registerMODEx = 0, the 8080 timing is triggered, otherwise, the 6800 timing is triggered.

When the LCDC module is writing data/commands to the LCD, busy = 1, any register of the LCDC cannot be written during this period, otherwise the write data will be ignored. Read LCDC_CSR [17] to know the busy state.

The LCDC_WRin the picture aboveis the LCDC_8WR_6Epinof the microprocessor; LCDC_RD isthe LCDC_8RD_6RW pin of the microprocessor.

(2) LCDC 8080 Read Data/Instruction Timing

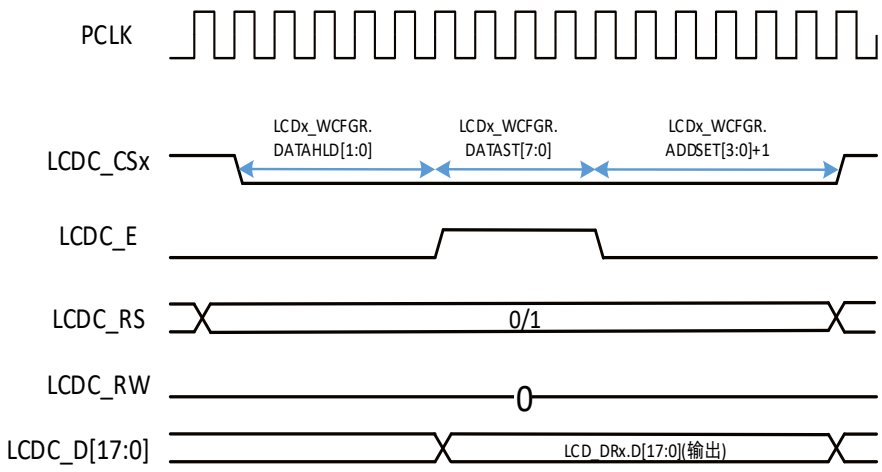


When LCDx_DR is read, an 8080/6800 data read operation is triggered; When LCDx_CMDR is read, an 8080/6800 read command operation is triggered. If the LCD_CSR registerMODEx = 0, the 8080 timing is triggered, otherwise, the 6800 timing is triggered.

When the LCDC module reads data from the LCD, the APB bus cannot access any registers of the LCDC.

The LCDC_RDin the picture aboveis the LCDC_8RD_6RWpinof the microprocessor; LCDC_WRis the LCDC_8WR_6E pinof the microprocessor.

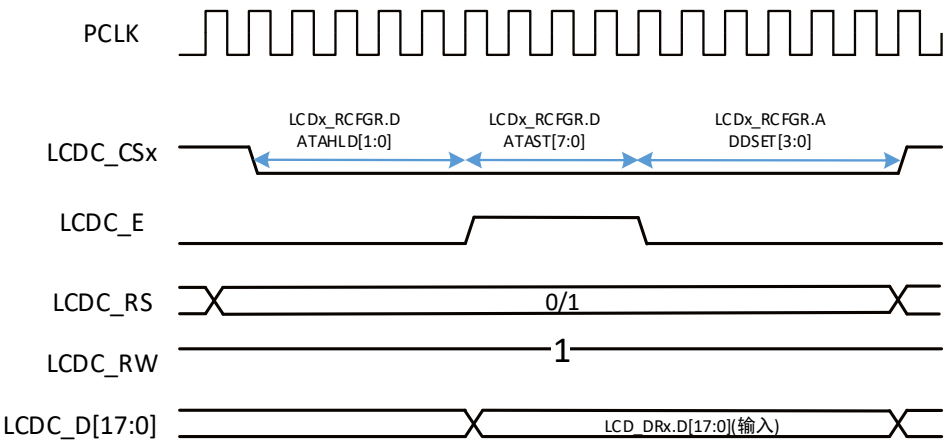
(3) LCDC 6800 write data/instruction timing



The trigger mode is the same as 8080, andthe LCD_CSR registerMODEx = 1needs to be set.

The LCDC_Ein the picture aboveis the LCDC_8WR_6Epin of the microprocessor; LCDC_RWis the LCDC_8RD _6RWpin of the microprocessor.

(4) LCDC 6800 Read Data/Instruction Timing



The trigger mode is the same as 8080, andthe LCD_CSR registerMODEx = 1needs to be set.

The LCDC_Ein the picture aboveis the LCDC_8WR_6Epin of the microprocessor; LCDC_RWis the LCDC_8RD _6RWpin of the microprocessor.

44.3.7 DMA request

After the DMA responds to the LCDC transfer request, it will send write data to the LCDC register, and the LCDC starts sending data.

(1) Peripheral address

The DMA function is only used for write data of LCD0_DR to LCD3_DR. Sending instructions, instruction parameters, and timing parameters must be used by CPU.

(2) Data width

The LCDC module supports DMA transmission of bytes, half words and words. In the LCDC registers, LCD0_DR ~ LCD3_DR support byte, half-word, and word writing, while the other registers only support word writing.

When DMA or CPU writes half-word datato LCDx_DR,pin LCDC_ D [17:16] will send low level; When DMA or CPU writes byte data to LCDx_DR,pin LCDC_ D [17: 8] sends low.

(3) Scope of application

The DMA of the LCDC has only write requests and no read requests. When reading data, it will occupy CPU resources for a long time.

When the DMA transmission is not complete, the CPU cannot access any registers of the LCDC, and the LCDC action cannot be guaranteed.

LCDC only supports 1 DMA transmission. When LCDC turns on DMA transfer, it does not support multiple DMA channels to write to LCDC, nor does DMA write to multiple LCDC registers.

44.3.8 LCDC busy mechanism

When the LCD is transmitting data (read or write data/instructions), the LCDC will be busy, LCD_CSR [17] = 1 (the busy bit is equal to 1).

When the LCD is in the busy state, no write operation is allowed in any register of the LCDC, otherwise the write data will be ignored.

If the CPU accesses LCDx_DR/LCDx_CMDR, when the LCD is in the busy state, the data register LCDx_DR is not allowed to be read, otherwise the read data is fixed to 0, but the LCDC_CSR, LCDx_WCFGR, LCDx_RCFGR registers are allowed to be read.

44.4 TIMx registers

Note: Only LCD0_DR ~ LCD3_DR support byte, half-word, and word writing, and the rest of the registers only support word writing. All registers support word reading only.

44.4.1 LCDC Control and Status Register (LCDC_CSR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														BUSY	DMAEN
-														R	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								MODE3	MODE2	MODE1	MODE0	EN3	EN2	EN1	EN0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17	BUSY	R	0	LCDC Busy Flag 0: LCDC idle 1: LCDC busy Refer to section LCDC busy mechanism
16	DMAEN	RW	0	DMA on LCDC enable 0: LCDC DMA OFF 1: LCDC DMA ON
15: 8	Reserved	-	-	-
7	MODE3	RW	0	LCD3 interface timing 0: LCD3 interface timing is 8080 timing 1: LCD3 interface timing is 6800 timing
6	MODE2	RW	0	LCD2 interface timing 0: LCD2 interface timing is 8080 timing 1: LCD2 interface timing is 6800 timing
5	MODE1	RW	0	LCD1 interface timing 0: LCD1 interface timing is 8080 timing 1: LCD1 interface timing is 6800 timing
4	MODE0	RW	0	LCD0 interface timing 0: LCD0 interface timing is 8080 timing 1: LCD0 interface timing is 6800 timing
3	EN3	RW	0	LCD3 enabled 0: LCD3 disabled 1: LCD3 enabled
2	EN2	RW	0	LCD2 enabled 0: LCD2 disabled 1: LCD2 enabled
1	EN1	RW	0	LCD1 enabled 0: LCD1 disabled 1: LCD1 enabled
0	EN0	RW	0	LCD0 enabled 0: LCD0 disabled 1: LCD0 enabled

44.4.2 LCD0 Data Register (LCD0_DR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														D [17:16]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	D [17: 0]	RW	18'h0	LCD0 data register

44.4.3 LCD0 Command Register (LCD0_CMDR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														CMD [17: 0]	
														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	CMD [17: 0]	RW	18'h0	LCD0 instruction register

44.4.4 LCD0 write timing configuration register (LCD0_WCFGR)

Address offset: 0x0C

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	rwr	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD0 write data hold time 00: Write data hold time is 1 APB cycle 01: Write data hold time is 2 APB cycles 10: Write data hold time is 3 APB cycles 11: Write data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD0 write data duration 0000 0000: Write data duration is 1 APB cycle 0000 0001: Write data duration is 2 APB cycles 0000 0010: Write data duration is 3 APB cycles … 1111 1111: Write data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD0 write address establishment duration 0000: Write address establishment time is 1 APB cycle 0001: Write address establishment time is 2 APB cycles … 1111: Write address establishment time is 16 APB cycles

44.4.5 LCD0 read timing configuration register (LCD0_RCFGR)

Address offset: 0x10

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD0 Read Data Hold Time

				00: Read data hold time is 1 APB cycle 01: Read data hold time is 2 APB cycles 10: The read data hold time is 3 APB cycles 11: Read data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD0 Read Data Duration 0000 0000: Read data duration is 1 APB cycle 0000 0001: Read data duration is 2 APB cycles 0000 0010: Read data duration is 3 APB cycles … 1111 1111: Read data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD0 Read Address Setup Duration 0000: Read address establishment time is 1 APB cycle 0001: Read address establishment time is 2 APB cycles … 1111: Read address setup time is 16 APB cycles

44.4.6 LCD1 Data Register (LCD1_DR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														D [17:16]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	D [17: 0]	RW	18'h0	LCD1 data register When writing the APB bus notebook register, the LCDC module will treat D [17: 0] as LCD data and send it to the LCD according to a specific timing (8080 or 6800, depending on the MODE1 bit of LCDC_CSR). When the APB bus reads this register, the LCDC module will read the LCD data according to a specific timing and send it to the APB bus. This register supports 8, 16, 32-bit write, but only 32-bit read In the 8080/6800 timing generated by the readbook register, only the CS1 slice line selection works (low level is active), and the rest slice line selection does not work (maintains high level).

44.4.7 LCD1 Command Register (LCD1_CMDR)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														CMD [17: 0]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	CMD [17: 0]	RW	18'h0	LCD1 instruction register

44.4.8 LCD1 write timing configuration register (LCD1_WCFGR)

Address offset: 0x1C

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													

RW	RW	-													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD1 write data hold time 00: Write data hold time is 1 APB cycle 01: Write data hold time is 2 APB cycles 10: Write data hold time is 3 APB cycles 11: Write data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD1 Write Data Duration 0000 0000: Write data duration is 1 APB cycle 0000 0001: Write data duration is 2 APB cycles 0000 0010: Write data duration is 3 APB cycles … 1111 1111: Write data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD1 Write Address Setup Duration 0000: Write address establishment time is 1 APB cycle 0001: Write address establishment time is 2 APB cycles … 1111: Write address establishment time is 16 APB cycles

44.4.9 LCD1 read timing configuration register (LCD1_RCFGR)

Address offset: 0x20

Reset value: 0x0000 FF0F

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	rwr	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD1 Read Data Hold Time 00: Read data hold time is 1 APB cycle 01: Read data hold time is 2 APB cycles 10: The read data hold time is 3 APB cycles 11: Read data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD1 Read Data Duration 0000 0000: Read data duration is 1 APB cycle 0000 0001: Read data duration is 2 APB cycles 0000 0010: Read data duration is 3 APB cycles … 1111 1111: Read data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD1 Read Address Setup Duration 0000: Read address establishment time is 1 APB cycle 0001: Read address establishment time is 2 APB cycles … 1111: Read address setup time is 16 APB cycles

44.4.10 LCD2 Data Register (LCD2_DR)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														D [17:16]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	D [17: 0]	RW	18'h0	LCD2 data register

44.4.11 LCD2 Command Register (LCD2_CMDR)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														CMD [17: 0]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	CMD [17: 0]	RW	18'h0	LCD2 instruction register

44.4.12 LCD2 write timing configuration register (LCD2_WCFGR)

Address offset: 0x2C

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD2 write data hold time 00: Write data hold time is 1 APB cycle 01: Write data hold time is 2 APB cycles 10: Write data hold time is 3 APB cycles 11: Write data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD2 Write Data Duration 0000 0000: Write data duration is 1 APB cycle 0000 0001: Write data duration is 2 APB cycles 0000 0010: Write data duration is 3 APB cycles … 1111 1111: Write data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD2 write address establishment duration 0000: Write address establishment time is 1 APB cycle 0001: Write address establishment time is 2 APB cycles … 1111: Write address establishment time is 16 APB cycles

44.4.13 LCD2 read timing configuration register (LCD2_RCFGR)

Address offset: 0x30

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD2 Read Data Hold Time 00: Read data hold time is 1 APB cycle 01: Read data hold time is 2 APB cycles 10: The read data hold time is 3 APB cycles 11: Read data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD2 Read Data Duration 0000 0000: Read data duration is 1 APB cycle 0000 0001: Read data duration is 2 APB cycles 0000 0010: Read data duration is 3 APB cycles … 1111 1111: Read data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD2 Read Address Setup Duration 0000: Read address establishment time is 1 APB cycle 0001: Read address establishment time is 2 APB cycles … 1111: Read address setup time is 16 APB cycles

44.4.14 LCD3 Data Register (LCD3_DR)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														D [17:16]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	D [17: 0]	RW	18'h0	LCD3 data register

44.4.15 LCD2 Command Register (LCD3_CMDR)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														CMD [17: 0]	
-														RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 18	Reserved	-	-	-
17: 0	CMD [17: 0]	RW	18'h0	LCD3 instruction register.

44.4.16 LCD3 write timing configuration register (LCD3_WCFGR)

Address offset: 0x3C

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD3 write data hold time 00: Write data hold time is 1 APB cycle 01: Write data hold time is 2 APB cycles 10: Write data hold time is 3 APB cycles 11: Write data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD3 Write Data Duration 0000 0000: Write data duration is 1 APB cycle 0000 0001: Write data duration is 2 APB cycles 0000 0010: Write data duration is 3 APB cycles … 1111 1111: Write data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD3 write address establishment duration 0000: Write address establishment time is 1 APB cycle 0001: Write address establishment time is 2 APB cycles … 1111: Write address establishment time is 16 APB cycles

44.4.17 LCD3 read timing configuration register (LCD3_RCFGR)

Address offset: 0x40

Reset value: 0x0000 FF0F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATAHLD [1: 0]		Res.													
RW	RW	-													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAST [7: 0								Res.				ADDSET [3: 0]			
RW	RW	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 30	DATAHLD [1: 0]	RW	2'h0	LCD3 read data retention time 00: Read data hold time is 1 APB cycle 01: Read data hold time is 2 APB cycles 10: The read data hold time is 3 APB cycles 11: Read data hold time is 4 APB cycles
29: 16	Reserved	-	-	-
15: 8	DATAST [7: 0	RW	8 'hff	LCD3 Read Data Duration 0000 0000: Read data duration is 1 APB cycle 0000 0001: Read data duration is 2 APB cycles 0000 0010: Read data duration is 3 APB cycles … 1111 1111: Read data duration is 256 APB cycles
7: 4	Reserved	-	-	-
3: 0	ADDSET [3: 0]	RW	4 'hf	LCD3 Read Address Setup Duration 0000: Read address establishment time is 1 APB cycle 0001: Read address establishment time is 2 APB cycles … 1111: Read address setup time is 16 APB cycles

45. MCU debugging interface

45.1 Introduction

The MCUDBG module provides debuggers with the following support:

- Low-power modes
- Clock control for timers, watchdogs, I2C and CAN
- The MCUDBG register also provides the encoding of the chip ID. This ID encoding can be accessed by a JTAG or SW debug interface, or by a user program.

45.2 Main features

- Supports debugging of sleep mode, stop mode and standby mode
- When the CPU enters the HALT mode, the control timer or watchdog stops counting or continues counting
- Prevent I2C's SMBUS timeout when CPU enters HALT
- Disable modification function when reading of CAN is enabled

45.3 Functional description

45.3.1 Support debugging low power mode

The user can enter the low power mode by executing the WFI and WFE instructions.

The MCU supports a variety of low-power modes, which can turn off the CPU clock or reduce CPU power consumption.

The kernel does not allow FCLK or HCLK to be turned off during debugging. As these are required for the debugger connection, during a debug, they must remain active. The MCU uses a special way that allows users to debug code in low power mode.

To achieve this function, the debugger must first set some configuration registers to change the characteristics of low power mode.

- In sleep mode, the debugger must first set the DBG_SLEEP bit. This will provide HCLK with the same clock as FCLK.
- In stop mode, the debugger must first set the DBG_STOP bit. This will activate the HSI16 clock, providing clocks for FCLK and HCLK in stop mode.
- Instandby mode, the debugger must first set the DBG_STDBY bit. This will activate the HSI16 clock, providing clocks for FCLK and HCLK in standby mode.

45.4 TIMx registers

The MCUDBG register is mapped on the external PPB bus.

45.4.1 ID coding (DBGMCU_IDCODE)

The register is reset by power-on, and the system reset does not reset the register. When the kernel is in the reset state, the debugger can access this register.

Address offset: 0x00

Reset value: 0x4444BBBB

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REV_ID [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REV_ID [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 0	REV_ID	R	32'h4444BBBB	MCU Product ID. The value is saved in the FLASH information area.

45.4.2 Debug Configuration Register (DBGMCU_CR1)

The register is reset by power-on, and the system reset does not reset the register. When the kernel is in the reset state, the debugger can access this register.

Address offset: 0x04

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	DBG_TIM11_STOP	DBG_TIM10_STOP	DBG_TIM9_STOP	DBG_TIM14_STOP	DBG_TIM13_STOP	DBG_TIM12_STOP	Res				DBG_TIM7_STOP	DBG_TIM6_STOP	DBG_TIM5_STOP	DBG_TIM8_STOP	DBG_I2C2_SMBUS_TIMEOUT
-	RW	RW	RW	RW	RW	RW	-				RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBG_I2C1_SMBUS_TIMEOUT	DBG_CAN1_STOP	DBG_TIM4_STOP	DBG_TIM3_STOP	DBG_TIM2_STOP	DBG_TIM1_STOP	DBG_WWDG_STOP	DBG_IWDG_STOP	Res					DBG_STDBY	DBG_STOP	DBG_SLEEP
RW	RW	RW	RW	RW	RW	RW	RW	-					RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30	DBG_TIM11_STOP	RW	0	When the CPU core is in the halt state, the control TIM11 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
29	DBG_TIM10_STOP	RW	0	When the CPU core is in the halt state, the control TIM10 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
28	DBG_TIM9_STOP	RW	0	When the CPU core is in the halt state, the control TIM9 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
27	DBG_TIM14_STOP	RW	0	When the CPU core is in the halt state, the control TIM14 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally;

				1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
26	DBG_TIM13_STOP	RW	0	When the CPU core is in the halt state, the control TIM13 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
25	DBG_TIM12_STOP	RW	0	When the CPU core is in the halt state, the control TIM12 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
24:21	Reserved	-	-	-
20	DBG_TIM7_STOP	RW	0	When the CPU core is in the halt state, the control TIM7 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
19	DBG_TIM6_STOP	RW	0	When the CPU core is in the halt state, the control TIM6 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
18	DBG_TIM5_STOP	RW	0	When the CPU core is in the halt state, the control TIM5 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
17	DBG_TIM8_STOP	RW	0	When the CPU core is in the halt state, the control TIM8 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
16	DBG_I2C2_SMBUS_TIMEOUT	RW	0	When the CPU core is in the halt state, the control I2C2 SMBUS timeout mode stops. 0: I2C2 is the same as normal mode when the CPU is halt; 1: When CPU is halt, I2C2 SMBUS timeout function is disabled;
15	DBG_I2C1_SMBUS_TIMEOUT	RW	0	When the CPU core is in the halt state, the control I2C1 SMBUS timeout mode stops. 0: I2C1 is the same as normal mode when the CPU is halt; 1: When CPU is halt, I2C21 SMBUS timeout function is disabled;
14	DBG_CAN1_STOP	RW	0	When the CPU core is in the halt state, start the CAN1 module's "Disable Modification on Read Function". 0: When CPU is halt, CAN1 is the same as normal mode; 1: When the CPU is halt, CAN1's "prohibit modification during read function" will be enabled. The function of inhibiting modification during read is described as follows: ECR.CEL [7: 0], PSR.PXE, PSR.RFDF, PSR.RBRS, PSR.RESI will not be reset, and PSR.LEC and PSR.DLEC signals will not be set. And these registers will not be reset or set after being read repeatedly.
13	DBG_TIM4_STOP	RW	0	When the CPU core is in the halt state, the control TIM4 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;

12	DBG_TIM3_STOP	RW	0	When the CPU core is in the halt state, the control TIM3 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
11	DBG_TIM2_STOP	RW	0	When the CPU core is in the halt state, the control TIM2 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
10	DBG_TIM1_STOP	RW	0	When the CPU core is in the halt state, the control TIM1 count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
9	DBG_WWDG_STOP	RW	0	When the CPU core is in the halt state, the control WWDG count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
8	DBG_IWDG_STOP	RW	0	When the CPU core is in the halt state, the control IWDG count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
7: 3	Reserved	-	-	-
2	DBG_STDBY	RW	0	Debug standby mode. 0: (FCLK off, HCLK off) Normal standby mode processing; 1: (FCLK ON, HCLK ON) The system clock is supplied by HSI 16. The MCU automatically generates a system reset to exit standby mode, which is the same as the power-on reset.
1	DBG_STOP	RW	0	Debug Stop mode 0: (FCLK OFF, HCLK OFF) In stop mode, both HCLK and FCLK are off. When exiting from stop mode, the clock configuration is the same as after reset (system clock is HSI16). Subsequently, the software needs to reconfigure the clock controller to enable the PLL and HSE, etc. 1: (FCLK=On, HCLK=On). When entering the stop mode, the internal clock source of the system will not be turned off, and FCLK and HCLK exist. When exiting Stop mode, the software must reprogram the clock controller if the clock control needs to be changed.
0	DBG_SLEEP	RW	0	Debugging sleep mode. 0: (FCLK ON, HCLK OFF). In Sleep mode, FCLK is provided by the previously configured system clock, and HCLK is turned off. Since sleep mode does not reset the configured clock system, the software does not need to reconfigure the clock after exiting sleep mode. 1: (FCLK=On, HCLK=On). In sleep mode, both the FCLK and HCLK clocks are provided by the previously configured system clock.

45.4.3 Debug Configuration Register (DBGMCU_CR2)

The register is reset by power-on, and the system reset does not reset the register. When the kernel is in the reset state, the debugger can access this register.

Address offset: 0x08

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							DBG_LPTIM_STOP	DBG_TIM19_STOP	DBG_TIM18_STOP	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP	DBG_CAN2_STOP	DBG_I2C4_SMBUS_TIMEOUT	DBG_I2C3_SMBUS_TIMEOUT
-							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 9	Reserved	-	-	-
8	DBG_LPTIM_STOP	RW	0	When the CPU core is in the halt state, the control LPTIM count stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
7	DBG_TIM19_STOP	RW	0	When the CPU core is in the halt state, the control TIM19 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled and counting stops;
6	DBG_TIM18_STOP	RW	0	When the CPU core is in the halt state, the control TIM18 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
5	DBG_TIM17_STOP	RW	0	When the CPU core is in the halt state, the control TIM17 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
4	DBG_TIM16_STOP	RW	0	When the CPU core is in the halt state, the control TIM16 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
3	DBG_TIM15_STOP	RW	0	When the CPU core is in the halt state, the control TIM15 counting stops. 0: When the CPU is halt, the counter clock is enabled and counts normally; 1: When the CPU is halt, the counter clock is disabled, counting is stopped, and the output is disabled;
2	DBG_CAN2_STOP	RW	0	When the CPU core is in the halt state, the CAN2 module's "prohibit modification when reading function" is started. 0: CAN2 is the same as the normal mode when the CPU is halt; 1: When the CPU is halt, CAN2's "prohibit modification during read function" will be enabled. The function of inhibiting modification during read is described as follows: ECR.CEL [7: 0], PSR.PXE, PSR.RFDF, PSR.RBRS, PSR.RESI will not be reset, and PSR.LEC and PSR.DLEC signals will not be set. And these registers will not be reset or set after being read repeatedly.
1	DBG_I2C4_SMBUS_TIMEOUT	RW	0	When the CPU core is in the halt state, the control I2C4 SMBUS timeout mode stops. 0: I2C4 is the same as normal mode when the CPU is halt; 1: When the CPU is halt, the I2C4 SMBUS timeout function is disabled;
0	DBG_I2C3_SMBUS_TIMEOUT	RW	0	When the CPU core is in the halt state, the control I2C3 SMBUS timeout mode stops.

				0: I2C3 is the same as normal mode when the CPU is halt; 1: When CPU is halt, I2C3 SMBUS timeout function is disabled;

46. Revision history

Version	Date	Descriptions
V0.2	2025.10.10	Initial version

Puya Semiconductor Co., Ltd.