PY32F031 Series

32-bit ARM® Cortex®-M0+ Microcontroller

Reference Manual

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/Clear Write0 (RC_W0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on this bit.
Read/Clear Write1 (RC_W1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on this bit.
Read/Clear Write1 (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/Clear by 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/Set by Read (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.
Reserved (Res)	Reserved bit, must be kept at reset value.

2. System architecture

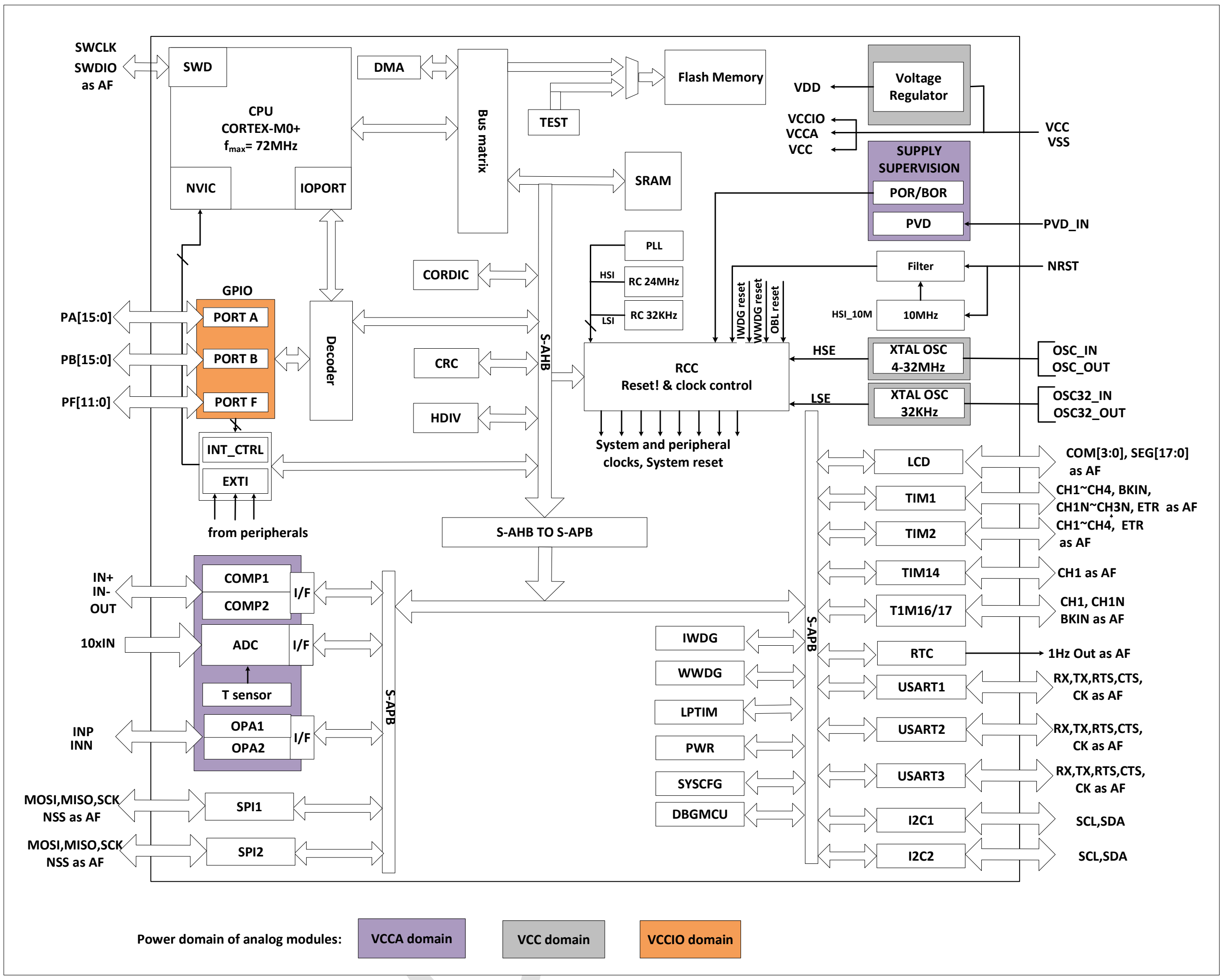


Figure 2-1 System architecture

3. Memory and bus architecture

3.1. System architecture

The system consists of:

- Two Masters
  - Cortex-M0+
  - General-purpose DMA
- Three Slaves
  - Internal SRAM
  - Internal Flash memory
  - AHB with AHB-APB bus bridge

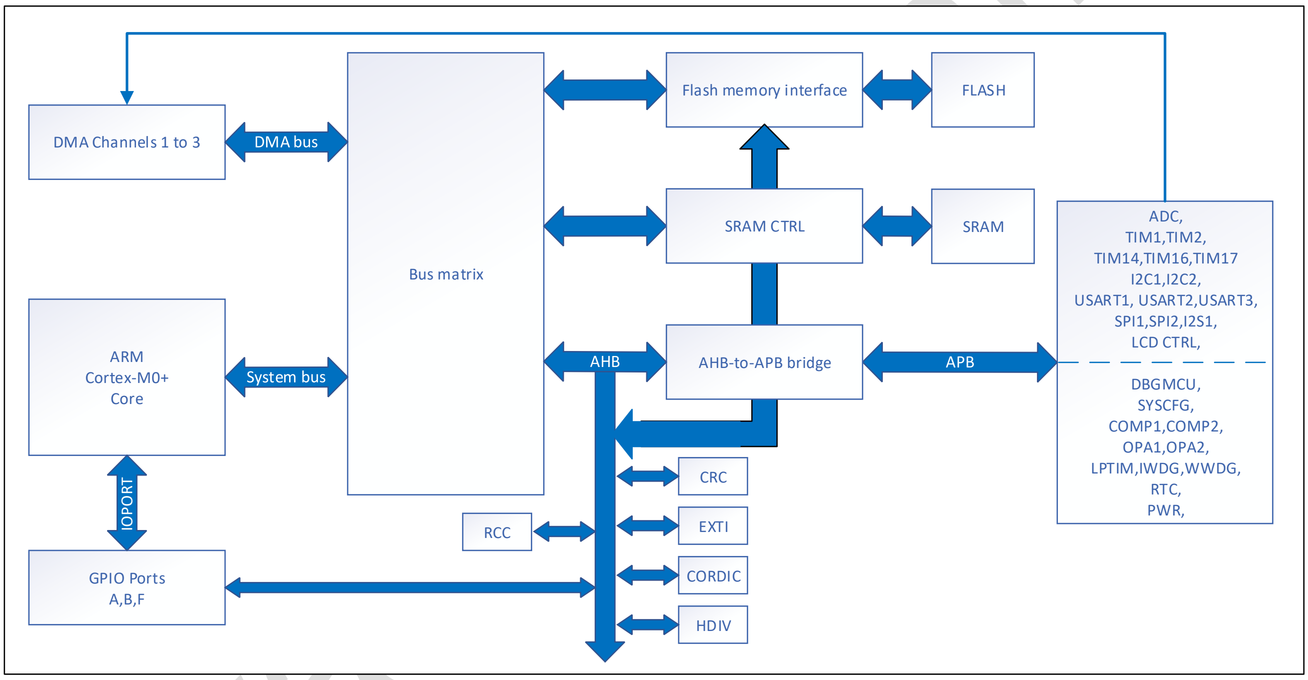


Figure 3-1 System architecture

- System bus

This bus connects the system bus of the Cortex®-M0+ core to a Bus Matrix which manages the arbitration between the CPU and the DMA.

- DMA bus

This bus connects the AHB master interface of the DMA to the Bus Matrix which manages the access of CPU and DMA to SRAM, Flash memory and AHB/APB peripherals.

- Bus Matrix

The Bus Matrix manages the access arbitration between the core system bus and the DMA master bus. This arbitration uses the Round Robin algorithm. The BusMatrix is composed of up to two masters (CPU, DMA) and three slaves (FLASH memory, SRAM, and AHB-to-APB bridge).

- AHB-to-APB bridge (APB)

The AHB to APB bridge provides full synchronous connections between the AHB and the APB bus and address mapping of the peripherals connected to this Bridge.

3.2. Memory organization

Program memory, data memory, registers and I/O ports are organized 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).

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

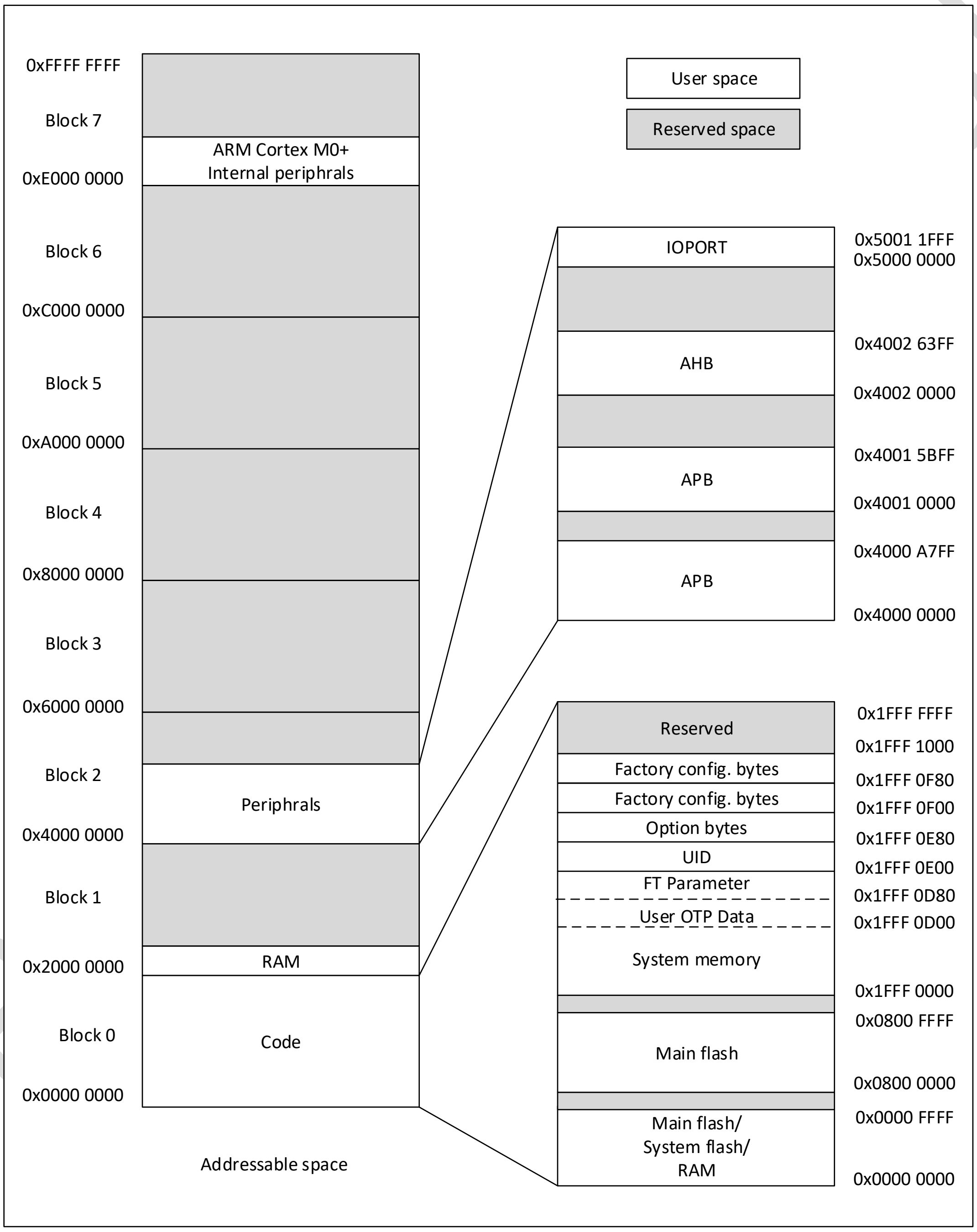


Figure 3-2 Memory map

Table 3-1 Memory Address

Type	Boundary Address	Size	Memory Area	Description
SRAM	0x2000 2000-0x3FFF FFFF	-	Reserved	-
	0x2000 0000-0x2000 1FFF	8 KB	SRAM	-
Code	0x1FFF 1000-0x1FFF FFFF	4 KB	Reserved	-
	0x1FFF 0F80-0x1FFF 0FFF	128 Bytes	Factory config. bytes	-
	0x1FFF 0F00-0x1FFF 0F7F	128 Bytes	Factory config. bytes	Store HSI triming data
	0x1FFF 0E80-0x1FFF 0EFF	128 Bytes	Option bytes	Software and hardware option
	0x1FFF 0E00-0x1FFF 0E7F	128 Bytes	UID	Unique ID
	0x1FFF 0D80-0x1FFF 0DFF	128 Bytes	Reserved	-
	0x1FFF 0D00-0x1FFF 0D7F	128 Bytes	User OTP Data	User area
	0x1FFF 0000-0x1FFF 0CFF	3.25 KB	System memory	deposit boot loader
	0x0801 0000-0x1FFE FFFF	-	Reserved	-
	0x0800 0000-0x0800 FFFF	64 KB	Main flash memory	-
	0x0001 0000-0x07FF FFFF	-	Reserved	-
	0x0000 0000-0x0000 FFFF	64 KB	Depending on the Boot	-

1. The address is marked as Reserved, which cannot be written and read as 0.

Table 3-2 Peripheral register address

Bus	Boundary Address	Size	Peripheral
	0xE000 000-0xE00F FFFF	1 MB	M0+
IOPORT	0x5000 1800-0x5FFF FFFF	-	Reserved
	0x5000 1400-0x5000 17FF	1 KB	GPIOF
	0x5000 1000-0x5000 13FF	-	Reserved
	0x5000 0C00-0x5000 0FFF	-	Reserved
	0x5000 0800-0x5000 0BFF	-	Reserved
	0x5000 0400-0x5000 07FF	1 KB	GPIOB
	0x5000 0000-0x5000 03FF	1 KB	GPIOA
AHB	0x4002 4000-0x4FFF FFFF	-	Reserved
	0x4002 3C00-0x4002 3FFF	-	Reserved
	0x4002 3800-0x4002 3BFF	1 KB	HDIV
	0x4002 3400-0x4002 37FF	1 KB	CORDIC
	0x4002 3000-0x4002 33FF	1 KB	CRC
	0x4002 2400-0x4002 2FFF	-	Reserved
	0x4002 2000-0x4002 23FF	1 KB	Flash
	0x4002 1C00-0x4002 1FFF	-	Reserved
	0x4002 1800-0x4002 1BFF	1 KB	EXTI
	0x4002 1400-0x4002 17FF	-	Reserved
	0x4002 1000-0x4002 13FF	1 KB	RCC
	0x4002 0400-0x4002 0FFF	-	Reserved
	0x4002 0000-0x4002 03FF	1 KB	DMA

Bus	Boundary Address	Size	Peripheral
APB	0x4001 5C00-0x4001 FFFF	-	Reserved
	0x4001 5800-0x4001 5BFF	1 KB	DBG
	0x4001 4C00-0x4001 57FF	-	Reserved
	0x4001 4800-0x4001 4BFF	1 KB	TIM17
	0x4001 4400-0x4001 47FF	1 KB	TIM16
	0x4001 3C00-0x4001 43FF	-	Reserved
	0x4001 3800-0x4001 3BFF	1 KB	USART1
	0x4001 3400-0x4001 37FF	-	Reserved
	0x4001 3000-0x4001 33FF	1 KB	SPI1
	0x4001 2C00-0x4001 2FFF	1 KB	TIM1
	0x4001 2800-0x4001 2BFF	-	Reserved
	0x4001 2400-0x4001 27FF	1KB	ADC
	0x4001 0400-0x4001 23FF	-	Reserved
	0x4001 0300-0x4001 03FF	1KB	OPA
	0x4001 0200-0x4001 02FF		COMP1 and COMP2
	0x4001 0000-0x4001 01FF		SYSCFG
	0x4000 B400-0x4000 FFFF	-	Reserved
	0x4000 B000-0x4000 B3FF	-	Reserved
	0x4000 8400-0x4000 AFFF	-	Reserved
	0x4000 8000-0x4000 83FF	-	Reserved
	0x4000 7C00-0x4000 7FFF	1 KB	LPTIM
	0x4000 7400-0x4000 7BFF	-	Reserved
	0x4000 7000-0x4000 73FF	1 KB	PWR
	0x4000 5C00-0x4000 6FFF	-	Reserved
	0x4000 5800-0x4000 5BFF	1 KB	I2C2
	0x4000 5400-0x4000 57FF	1 KB	I2C1
	0x4000 4C00-0x4000 53FF	-	Reserved
	0x4000 4800-0x4000 4BFF	1 KB	USART3
	0x4000 4400-0x4000 47FF	1 KB	USART2
	0x4000 3C00-0x4000 43FF	-	Reserved
	0x4000 3800-0x4000 3BFF	1 KB	SPI2
	0x4000 3400-0x4000 37FF	-	Reserved
	0x4000 3000-0x4000 33FF	1 KB	IWDG
	0x4000 2C00-0x4000 2FFF	1 KB	WWDG
	0x4000 2800-0x4000 2BFF	1 KB	RTC
	0x4000 2400-0x4000 27FF	1 KB	LCD
	0x4000 2000-0x4000 23FF	1 KB	TIM14
	0x4000 1800-0x4000 1FFF	-	Reserved

Bus	Boundary Address	Size	Peripheral
	0x4000 1400-0x4000 17FF	-	Reserved
	0x4000 1000-0x4000 13FF	-	Reserved
	0x4000 0800-0x4000 13FF	-	Reserved
	0x4000 0400-0x4000 07FF	-	Reserved
	0x4000 0000-0x4000 03FF	1 KB	TIM2

3.3. Embedded SRAM

PYF031 series feature 8 KB SRAM. It is accessed by half-word (16 bits) or word (32 bits).

The size of the SRAM can be set to 8 KB/6 KB/4 KB/2 KB by option byte to match the configurable flash size (When set to less than 8 KB, the software will generate a HardFault for reading and writing operations in the space outside the set range).

3.4. Flash memory

The Flash memory is composed of two distinct physical areas:

- The Main flash area: 64 KB, consisting of application and user data, Can be defined as 64 KB/48 KB/32 KB/16 KB depending on different products, and is used to store user programs and user data. When set to less than 64 KB capacity, software access to space outside the set range produces a HardFault.
- 4 KB of Information area:
  - Factory config. bytes 0, 128 Bytes, used to store
    - HSI frequency selection value and corresponding Trimming value
    - Configuration parameter values of erase and write time corresponding to different frequencies of HSI
  - Factory config. Bytes 1: 128 Bytes for storing:
    - Power-on verification code reading
  - UID: 128 Bytes, used to store the UID
  - Option byte: 128 Bytes, used to store configuration values for hardware and storage protection
  - User OTP Memory: 128 Bytes, used to store user data

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

3.5. Boot modes

At startup, the nBOOT0 pin and nBOOT1 (stored in Option bytes) are used to select one of the three boot options in the following table:

Table 3-3 Boot configuration

Boot mode configuration		Mode
nBOOT1 bit	BOOT0 pin	
X	0	Boot from Main flash
1	1	Boot from System memory
0	1	Boot from SRAM

After this startup delay has elapsed, the CPU fetches the top-of-stack value from address 0x0000 0000, then starts code execution from the boot memory at 0x0000 0004. Depending on the selected boot mode, Main flash memory, system memory or SRAM is accessible as follows:

- Boot from main flash: the main Flash memory is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x0800 0000). In other words, the Flash memory contents can be accessed starting from address 0x0000 0000 or 0x0800 0000.
- Boot from system memory: The system memory is aliased in the boot memory space (0x0000 0000), but it is still accessible from its original memory space (0x1FFF 0000).
- Boot from SRAM: the SRAM is aliased in the boot memory space (0x0000 0000), but it is still accessible from its original memory space (0x2000 0000).

3.5.1. Memory physical mapping

If the Boot mode is selected, the application software can modify the memory that is accessible in the program space. This modification is determined by the MEM _ MODE bit selection of SYSCFG configuration register 1 (SYSCFG _ CFGR1).

3.5.2. Embedded bootstrapper

The Boot loader is written during the chip production phase and stored in the system memory. It is used to re-write to flash memory using the following serial interface:

- USART for PA14/PA15 or PA9/PA10 or PA2/PA3

4. Embedded Flash memory

4.1. Main features of flash memory

- Main flash block: maximum 64 KB (16k x 32 bits)
- Information block: 4 KB (1k x 32 bits)
- Page size: 128 Bytes
- Sector size: 4 KB

Flash memory interface features:

- Flash memory read, write and erase
- Write protection
- Read protection

4.2. Flash memory functional description

4.2.1. Flash memory organization

The Flash memory is organized as 32-bit wide memory cells that can be used for storing both code and data constants. The Page size is 128 Bytes and the Sector size is 4 KB.

Functionally, Flash memory is divided into Main flash and information flash. The former has a capacity of 64 KB and the latter has a capacity of 8 KB.

The Page erase operation can be applied to the Main flash.

If write protection is set, Mass erase can be applied to Main flash.

Table 4-1Flash structure and boundary address

Block	Sector	Page	Base address	Size
Main memory	Sector 0	Page 0-31	0x0800 0000-0x0800 0FFF	4KB
	Sector 1	Page 32-63	0x0800 1000-0x0800 1FFF	4KB
	Sector 2	Page 64-95	0x0800 2000-0x0800 2FFF	4KB
	…	…	…	…
	Sector 14	Page 448-479	0x0800 E000-0x0800 EFFF	4KB
	Sector 15	Page 480-511	0x0800 F000-0x0800 FFFF	4KB
System memory	Sector 16	Page 0-25	0x1FFF 0000-0x1FFF 0CFF	3.25 KB
User Data		Page 26	0x1FFF 0D00-0x1FFF 0D7F	128bytes
Reserved		Page 27	0x1FFF 0D80-0x1FFF 0DFF	128bytes
UID		Page 28	0x1FFF 0E00-0x1FFF 0E7F	128bytes
Option bytes		Page 29	0x1FFF 0E80-0x1FFF 0EFF	128bytes
Factory config 0		Page 30	0x1FFF 0F00-0x1FFF 0F7F	128bytes
Factory config 1		Page 31	0x1FFF 0F80-0x1FFF 0FFF	128bytes

4.2.2. Flash read operation and access latency

The embedded Flash module can be addressed directly, as a common memory space. Through the special read control timing, the contents of the Flash memory can be read.

Both fetching and data access are carried out through the AHB bus. Read accesses can be performed through the Latency of the FLASH_ACR register, that is, number of wait states for a correct read operation is zero or one.

flash _ ACR.0 (LATENCY) bit, when it is 0, the wait state of the flash read operation is not increased; When it is 1, the flash read operation adds 1 waiting state; When it is 2, the flash read operation adds 3 wait states. This mechanism is specially designed to match the high-speed system clock and the relatively low Flash read speed.

4.2.3. Flash program and erase operations

The Flash memory can be programmed using in-circuit programming (ICP) or in-application programming (IAP).

ICP: used to update the entire contents of the Flash memory, using the SWD protocol or the boot loader to load the user application into the microcontroller. ICP provides fast and efficient design iterations and eliminates unnecessary packet processing or socketing.

IAP: using any communication interface supported by the microcontroller to download programming data into Flash memory. IAP allows the user to re-program the Flash memory while the application is running. Nevertheless, part of the application has to have been previously programmed in the Flash memory using ICP.

If a reset occurs when a Flash write or erase operation is performed, the contents of the Flash memory are not protected.

During a write or erase operation, any operation to read the flash will delay the bus. The read operation will proceed correctly once the program/erase operation has completed. This means that code or data fetches cannot be made while a program/erase operation is ongoing.

For write and erase operations, the HSI must be turned on (the software configures the corresponding parameters according to the HSI frequency to the erase and write time control register).

Write and erase operations can be implemented through the registers related to the following flash control interface:

- Acess control register (FLASH _ ACR)
- KEY register (FLASH _ KEYR)
- Option byte key register (FLASH _ OPTKEYR)
- Flash status register (FLASH _ SR)
- Flash control register (FLASH _ CR)
- Flash option register (FLASH _ OPTR)
- Flash special area address register (FLASH _ SDKR)
- Flash write protection resister (FLASH _ WRPR)
- Flash sleep time config register (FLASH _ STCR)
- Flash TS0 register(FLASH_TS0)
- Flash TS1 register(FLASH_TS1)
- Flash TS2P register(FLASH_TS2P)
- Flash TPS3 register(FLASH_TPS3)
- Flash TS3 register(FLASH_TS3)

- Flash page erase TPE register(FLASH_PERTPE)
- Flash sector/mass erase TPE register(FLASH_SMERTPE)
- Flash program TPE register(FLASH_PRGTPE)
- Flash pre-program TPE register(FLASH_PRETPE)

4.2.3.1. Unlocking the Flash memory

After reset, the Flash memory is protected against unwanted (caused by electrical interference) write or erase operations. The FLASH_CR register is not accessible in write mode, except for the OBL_LAUNCH bit, used to reload the option bits. An unlocking sequence should be written to the FLASH_KEYR register to open the access to the FLASH_CR register.

This is done in the following steps:

Step 1：write KEY1=0x4567 0123 to FLASH_KEYR register

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

Any wrong sequence locks up the FLASH_CR register until the next reset. In the case of a wrong key sequence, a bus error is detected, and a HardFault interrupt is generated. This is done after the first write cycle if KEY1 does not match, or during the second write cycle if KEY1 has been correctly written but KEY2 does not match.

The FLASH_CR register can be locked again by user software by writing the LOCK bit in the FLASH_CR register to 1.

In addition, the FLASH_CR register cannot be written when the BSY bit of the FLASH_SR register is set. Meanwhile, any attempt to write FLASH_CR register will cause the AHB bus to stall until the BSY bit is cleared.

4.2.3.2. Flash memory programming

After reset, the Flash memory is protected against unwanted (caused by electrical interference) program or erase operations. After reset, writing to the FLASH _ CR register is not allowed (except for the OBL _ LAUNCH bit used as the reload option bytes). An unlocking sequence should be written to the FLASH_KEYR register to open the access to the FLASH_CR register.

This is done in the following steps:

Step 1：write KEY1=0x4567 0123 to FLASH_KEYR register

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

Any wrong sequence locks up the FLASH_CR register until the next reset. In the case of a wrong key sequence, a bus error is detected, and a HardFault interrupt is generated. This is done after the first write cycle if KEY1 does not match, or during the second write cycle if KEY1 has been correctly written but KEY2 does not match.

The FLASH_CR register can be locked again by user software by writing the LOCK bit in the FLASH_CR register to 1.

In addition, the FLASH_CR register cannot be written when the BSY bit of the FLASH_SR register is set. Meanwhile, any attempt to write FLASH_CR register will cause the AHB bus to stall until the BSY bit is cleared.

Flash memory programming

Flash memory writes the entire page in units of 32-bit words each time (performing half-word or byte operations will produce hardfault). The program operation is started when the CPU writes a half-word into a main Flash memory address with the PG bit of the FLASH_CR register set. Any attempt to write data that are not full word long will result in a bus error generating a HardFault interrupt.

If the addressed main Flash memory location is write-protected by the FLASH_WRPR register, the program operation is skipped and a warning is issued by the WRPRTERR bit in the FLASH_SR register. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.

The Flash memory programming sequence is as follows:

1) Check that no main Flash memory operation is ongoing by checking the BSY bit in the FLASH_SR register.
2) If there is no flash erase or program operation in progress, the software reads out 32 words of the Page (this step is performed if the Page already has data stored, otherwise this step is skipped)
3) Write KEY1 and KEY2 to the FLASH _ KEYR register to unlock the protection of the FLASH _ CR register
4) Set the PG bit and EOPIE bit in the FLASH_CR register
5) Write the 1st to 31st words to the destination address (only 32-bit writes are accepted)
6) Set the PGSTRT bit in the FLASH_CR register
7) Write the 32nd word
8) Wait until the BSY bit is reset in the FLASH_SR register
9) Check the EOP flag in the FLASH_SR register (it is set when the programming operation has succeeded), and then clear it by software
10) If the program operation is end, the PG bit will be cleared by software

When the step 7 is carried out, the program operation is automatically started and the BSY bit is set simultaneously.

Flash memory erase

Flash memory can be erased according to page, or can be erased according to sector and mass.

4.2.3.3. Page erase

When a page is write-protected, it will not be erased and the WRPERR bit is set. A page erase operation needs to perform the following steps:

1) Check that no main Flash memory operation is ongoing by checking the BSY bit in the FLASH_SR register.
2) Write KEY1 and KEY2 to the FLASH _ KEYR register to unlock the protection of the FLASH _ CR register
3) Set the PER bit and EOPIE bit in 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 EOP flag bit is set
7) Clear EOP flag

4.2.3.4. Mass erase

Mass erase is used to erase the entire Main flash, but it does not work on the Information area. In addition, when WRP is enabled, the mass erase function is invalidated, no mass erase operation is generated, and the WEPERR bit is set.

1) The steps for performing mass erase are as follows:
2) Check the BSY bit to confirm if there are ongoing Flash operations
3) Write KEY1 and KEY2 to the FLASH _ KEYR register to unlock the protection of the FLASH _ CR register
4) Set the MER bit and EOPIE bit in the FLASH_CR register
5) Write any data (32-bit data) to any Main flash space
6) Wait for the BSY bit to be cleared.
7) Check that EOP flag bit is set
8) Clear EOP flag

4.2.3.5. Sector erase

Sector erase is used to erase 4 KB of Main flash, but it does not work on the information area. In addition, when a sector is protected by WRP, it will not be erased, and the WRPERR bit is set at this time.

The steps for a sector erase are as follows:

1) Check the BSY bit to confirm if there are ongoing Flash operations
2) Write KEY1 and KEY2 to the FLASH _ KEYR register to unlock the protection of the FLASH _ CR register
3) Set the SER bit and EOPIE bit in the FLASH_CR register
4) Write arbitrary data to this sector
5) Wait for the BSY bit to be cleared.
6) Check that EOP flag bit is set
7) Clear EOP flag

4.2.3.6. Write and erase time configuration

The write and erase time of Flash needs to be strictly controlled, otherwise the operation will fail. Power-on By default, the hardware design sets the time parameters of program and erase operations to the parameters with HSI of 24 MHz. When the HSI output frequency is changed, the Flash program and erase time control registers need to be properly configured according to the following table.

4.3. Unique device ID (UID)

Typical application scenarios of unique identification code:

- Used as a serial number
- When programming internal flash memory, use it as a key or encryption primitive to improve code security
- To activate secure boot processes, etc.
- The UID provides a reference number that is unique to any device.

- The user can never change these bits. Unique identifiers can also be read in different ways, such as byte/half word/word, and then concatenated using custom algorithms.

Base address : 0x1FFF 0E00

Offset Address	**Description**	**UID Bits**							
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	Lot Numer	Lot Number ASCII							
1	Lot Numer	Lot Number ASCII							
2	Lot Numer	Lot Number ASCII							
3	Lot Numer	Lot Number ASCII							
4	Wafer Number	Wafer Number							
5	Lot Numer	Lot Number ASCII							
6	Lot Numer	Lot Number ASCII							
7	Lot Numer	Lot Number ASCII							
8	Internal code	Internal code							
9	Y coordinate low order	Y coordinate low order							
10	X coordinate low order	X coordinate low order							
11	X, YCoordinate high address	Y coordinate high order				X coordinate high orde			
12	Fixed code	0x78							
13	Internal code	Internal code							
14	Internal code	Internal code							
15	Internal code	Internal code							

4.4. FLASH option bytes

4.4.1. Flash option byte description

Part of the information area of flash is used as option bytes, which are configured by the end user depending on the application requirements. 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 Option byte format

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

The software can read the option bytes from these flash memory locations or from their corresponding option registers referenced in the table.

- FLASH user option register (FLASH _ OPTR)
- FLASH SDK area address register (FLASH _ SDKR)
- FLASH WRP area address register (FLASH_WRPR)

Table 4-3 Option byte organization

Address	Description
0x1FFF 0E80	Flash user options and their inverted option bytes
0x1FFF 0E84	BOR configuration and its inverse code
0x1FFF 0E88	Flash SDK region address and option byte of its complement
0x1FFF 0E8C	Option byte and its complement of flash memory WRP address
0x1FFF 0E90	Reserved
0x1FFF 0E94	Reserved
…	Reserved
…	Reserved
…	Reserved
0x1FFF 0EFC	Reserved

- **Option bytes for Flash user options**

Flash memory address: 0x1FFF 0E80

Production value: 0x4755 B8AA

After the power-on reset (POR/BOR/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
~ nBOOT1	~NRST_ MODE	~ WWDG _SW	~ IWDG _SW	~BOR_LEV[2:0]			~BOR_ EN	~RDP[7:0]							
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**
nBOOT1	NRST_ MODE	WWDG _SW	IWDG _SW	BOR_LEV[2:0]			BOR_ EN	RDP[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~ nBOOT1	R	Complemented code of nBOOT1
30	~NRST_MODE	R	Complemented code of NRST_MODE
29	~ WWDG _ SW	R	Complemented code of WWDG _ SW
28	~IWDG_SW	R	Complemented code of IWDG_SW
27	~IWDG_STOP	R	Complemented code of IWDG _ STOP
26:24	Reserved	R	Complemented code of Bit [10: 8]
23:16	~ RDP	R	Complemented code of RDP
15	nBOOT1	R	Together with the BOOT PIN, select the boot mode
14	NRST_MODE	R	0: Reset input only 1: GPIO function
13	WWDG _ SW	R	0: Hardware watchdog 1: Software watchdog
12	IWDG_SW	R	0: Hardware watchdog 1: Software watchdog
11	IWDG_STOP	R	Independent watchdog counter freeze in Stop mode 0: Independent watchdog counter is frozen in Stop mode 1: Independent watchdog counter is running in Stop mode
10:8	Reserved	R	
7:0	RDP	R	0xAA: level 0, read protection inactive Others: level 1, read protection active

- **Option bytes for BOR configuration**

Flash memory address: 0x1FFF 0E84

Production value: 0xFFFF 0000

After the power-on reset (POR/BOR/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
											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	Res	Res	Res	Res	Res	Res	Res	BOR_LEV[2:0]			BOR_EN
												R	R	R	R

Bit	Name	R/W	Function
31:20	Reserved	R	Complemented code of Bit [15: 4]
19:17	Complemented BOR _ LEV	R	Complemented code of BOR_LEV
16	~BOR_EN	R	Complemented code of BOR_EN
15:4	Reserved	R	
3:1	BOR_LEV[2:0]	R	000：BOR rising threshold is 1.8V and falling threshold is 1.7V 001：BOR rising threhold is 2.0V and falling threhold is 1.9V 010：BOR rising threshold is 2.2V and falling threshold is 2.1V 011：BOR rising threshold is 2.4V and falling threshold is 2.3V 100：BOR rising threshold is 2.6V and falling threshold is 2.5V 101：BOR rising threshold is 2.8V and falling threshold is 2.7V 110：BOR rising threshold is 3.0V and falling threshold is 2.9V 111：BOR rising threshold is 3.2V and falling threshold is 3.1V
0	BOR_EN	R	BOR enable 0：BOR disabled 1：BOR enabled, BOR_LEV works

- **Flash SDK area address option byte**

Flash memory address: 0x1FFF 0E84

Production value: 0xFFE0 001F

After the power-on reset (POR/BOR/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.	~SDK_END[4:0]					Res.	Res.	Res.	~SDK_STRT[4:0]				
			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.	SDK_END[4:0]					Res.	Res.	Res.	SDK_STRT[4:0]				
			R	R	R	R	R				R	R	R	R	R

Bit	Name	R/W	Function
31:29	Reserved	R	Complemented code of Bit [15:13]
28:24	Complemented SDK _ END [4: 0]	R	Inverse code of SDK _ END
23:21	Reserved	R	Complemented code of Bit [7: 5]
20:16	Complemented SDK_STRT[4:0]	R	Inverse code of SDK _ STRT
15:13	Reserved	R	
12:8	SDK_END[4:0]	R	Stop address of SDK area, STEP corresponding to each bit is 2 KB
7:5	Reserved	R	
4:0	SDK_STRT[4:0]	R	Start address of SDK area, STEP corresponding to each bit is 2 KB

- **Flash WRP area address option byte**

Flash memory address: 0x1FFF 0E8C

Production value: 0x0000 FFFF

After the power-on reset (POR/BOR/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
~WRP[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**
WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	Complemented WRP	R	Complemented code of WRP
15:0	WRP	R	0: sector [y] is protected 1： sector[y] is unprotected y=0~15

4.4.2. FLASH option bytes

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 (Operate in every program or erase progress).

The following steps are used to unlock the register:

1) Unlock write protection of the FLASH _ CR register by unlocking timing

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. When the KEY timing is wrong, a bus error state is generated and a HardFault interrupt is generated.

User option (the option byte of information flash) can be protected by software writing the OPTLOCK bit of the FLASH _ CR register to prevent unwanted erase or write operations.

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

Modify user's option bytes

The write operation of the option byte is different from the operation of the 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 ~ 3 words) to the option byte register FLASH _ OPTR/FLASH _ SDKR/FLASH _ WRPR (ignore this operation if writing is not word aligned)
4) Set OPTSTRT bit
5) Write any 32-bit data to 0x1fff0e80 (triggers a formal write operation) (if writing is not word aligned, it will produce a hardfault)
6) Wait for the BSY bit to be cleared.
7) Wait for the EOP to rise, the software clears

For any changes to the option byte, the hardware will first erase the entire page corresponding to the option byte, and then write it to the option byte with the values of the FLASH _ OPTR, FLASH _ SDKR or FLASH _ WRPR registers. The hardware automatically calculates the corresponding inverse code and writes the calculated value to the corresponding area of the option byte.

Reload option 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/PDR/BOR)

The operation performed by "Load 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 _ OPTR, FLASH _ SDKR and FLASH _ WRPR). These internal registers configure the system and can be read by software. Setting the OBL _ LAUNCH bit generates a reset, so that the loading of option bytes can be performed under the system reset.

Each option bit has a corresponding complement at its same double word address (the next half word). During option byte loading, the option bit and its complement are verified, which ensures that the loading is done correctly.

If the positive complement matches, the option byte is copied into the option register.

If the positive complement does not match, the OPTVERR status bit of the FLASH _ SR register is set. The register is defined to write values as follows:

- For user options
  - BOR _ LEV is written as 000 (lowest threshold)
  - The BOR _ EN bit is written as 0 (BOR not enabled)
  - The NRST _ MODE bit is written as 0 (reset input only)
  - The RDP bit is written as 0xff (i.e. level 1)
  - The rest of the mismatched values are written as 1
- For the SDK area option, SDKR _ STRT [4: 0] = 0x00, SDKR _ END [4: 0] = 0x1F, that is, all flash space is set to the SDK
- For the WRP option, the mismatched value is the default "Unprotected"

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

- FLASH_OPTR
- FLASH _ SDKR
- FLASH _ WRPR

These registers are also used to modify option bytes. If these registers are not modified by the user, they embody the state of the system option.

4.5. Flash option bytes

Part of the flash information area in the chip (2 pages in total) is used as the Factory config. byte uses.

Factory 0 stores information for software to read (only positive code, no negative code is stored):

- HSI frequency selection value and corresponding Trimming value
- Configuration parameter values of erase and write time corresponding to different frequencies of HSI

Page 1 stores chip hardware factory information (positive and negative code storage):

- Power-on verification code reading
- Hardware Trimming configuration value

For data security, Factory config.byte1 is stored separately in the form of body and inverted code. If the hardware is powered on and reads the page and saves it to word with positive and negative codes, and the corresponding positive and negative codes are correct, the value stored in word is loaded into the corresponding input register. Conversely, the hardware maintains the default value of the input register and sets the relevant bit of the FLASH _ TRMLSR register.

Table 4-4 Factory config. byte organization

Word	Address	Contents
0	0x1FFF 0F00	Stores HSI 4 MHz frequency selection control and corresponding Triming values
1	0x1FFF 0F04	Stores HSI 8 MHz frequency selection control and corresponding Trimming values
2	0x1FFF 0F08	Stores HSI 8 MHz frequency selection control and corresponding Trimming values
3	0x1FFF 0F0C	Stores HSI 22.12 MHz frequency selection control and corresponding Trimming values
4	0x1FFF 0F10	Stores HSI 24 MHz frequency selection control and corresponding Trimming values
5	0x1FFF 0F14	Calibration value of 30 °C temperature sensor
6	0x1FFF 0F18	Calibration value of 85 °C ,105 °C temperature sensor
7	0x1FFF 0F1C	Store the configuration values of the corresponding FLASH _ TS0 and FLASH _ TS1 registers at HSI 4 MHz frequency
8	0x1FFF 0F20	Store the configuration values of the corresponding FLASH _ TS2P and FLASH _ TPS3 registers at HSI 4 MHz frequency
9	0x1FFF 0F24	Stores the configuration value of the corresponding FLASH _ PERTPE register at the frequency of HSI 4 MHz
10	0x1FFF 0F28	Stores the configuration value of the corresponding FLASH _ SMERTPE register at the frequency of HSI 4 MHz
11	0x1FFF 0F2C	Store the configuration values of the corresponding FLASH _ PRGTPE and FLASH _ PRETPE registers at HSI 4 MHz frequency
12	0x1FFF 0F30	Store the configuration values of the corresponding FLASH _ TS0 and FLASH _ TS1 registers at HSI 8 MHz frequency
13	0x1FFF 0F34	Store the configuration values of the corresponding FLASH _ TS2P and FLASH _ TPS3 registers at HSI 8 MHz frequency
14	0x1FFF 0F38	Stores the configuration value of the corresponding FLASH _ PERTPE register at the frequency of HSI 8 MHz
15	0x1FFF 0F3C	Stores the configuration value of the corresponding FLASH _ SMERTPE register at the frequency of HSI 8 MHz
16	0x1FFF 0F40	Store the configuration values of the corresponding FLASH _ PRGTPE and FLASH _ PRETPE registers at HSI 8 MHz frequency
17	0x1FFF 0F44	Store the configuration values of the corresponding FLASH _ TS0 and FLASH _ TS1 registers at HSI 16 MHz frequency
18	0x1FFF 0F48	Store the configuration values of the corresponding FLASH _ TS2P and FLASH _ TPS3 registers at HSI 16 MHz frequency
19	0x1FFF 0F4C	Stores the configuration value of the corresponding FLASH _ PERTPE register at the frequency of HSI 16 MHz

20	0x1FFF 0F50	Stores the configuration value of the corresponding FLASH _ SMERTPE register at the frequency of HSI 16 MHz
21	0x1FFF 0F54	Store the configuration values of the corresponding FLASH _ PRGTPE and FLASH _ PRETPE registers at HSI 16 MHz frequency
22	0x1FFF 0F58	Store the configuration values of the corresponding FLASH _ TS0 and FLASH _ TS1 registers at HSI 22.12 MHz frequency
23	0x1FFF 0F5C	Store the configuration values of the corresponding FLASH _ TS2P and FLASH _ TPS3 registers at HSI 22.12 MHz frequency
24	0x1FFF 0F60	Stores the configuration value of the corresponding FLASH _ PERTPE register at the frequency of HSI 22.12 MHz
25	0x1FFF 0F64	Stores the configuration value of the corresponding FLASH _ SMERTPE register at the frequency of HSI 22.12 MHz
26	0x1FFF 0F68	Store the configuration values of the corresponding FLASH _ PRGTPE and FLASH _ PRETPE registers at HSI 22.12 MHz frequency
27	0x1FFF 0F6C	Store the configuration values of the corresponding FLASH _ TS0 and FLASH _ TS1 registers at HSI 24 MHz frequency
28	0x1FFF 0F70	Store the configuration values of corresponding FLASH _ TS2P and FLASH _ TPS3 registers at HSI 24 MHz frequency
29	0x1FFF 0F74	Stores the configuration value of the corresponding FLASH _ PERTPE register at the frequency of HSI 24 MHz
30	0x1FFF 0F78	Stores the configuration value of the corresponding FLASH _ SMERTPE register at the frequency of HSI 24 MHz
31	0x1FFF 0F7C	Store the configuration values of the corresponding FLASH _ PRGTPE and FLASH _ PRETPE registers at HSI 24 MHz frequency

4.5.1. HSI_TRIMMING_FOR_USER

Address: 0x1FFF 0F00~0x1FFF 0F10

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.	HSI_FS[2:0]		
													R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	HSI_TRIM[12:0]												
			R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to read the data from this address and then write it to HSI _ FS [2: 0] and HSI _ TRIM [12: 0] corresponding to the RCC _ ICSCR register to change the HSI frequency.

4.5.2. HSI_4M/8M/16M/22.12M/24M_EPPARA0

Address: 0x1FFF 0F1C(4 MHz)、0x1FFF 0F30(8 MHz)、0x1FFF 0F44(16 MHz)、0x1FFF 0F58(22.12 MHz)、0x1FFF 0F6C(24 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1 [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**
TS3 [7:0]								TS0 [7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to read data from the corresponding address according to the required HSI clock frequency, and then write it to the FLASH _TS0, FLASH _TS1, and FLASH _TS3 registers to realize the configuration of the erasing and writing time required for the corresponding HSI frequency.

4.5.3. HSI_4M/8M/16M/22.12M/24M_EPPARA1

Address: 0x1FFF 0F20(4 MHz)、0x1FFF 0F34(8 MHz)、0x1FFF 0F48(16 MHz)、0x1FFF 0F5C(22.12 MHz)、0x1FFF 0F70(24 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	TPS3 [10:0]										
					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.	Res.	Res.	Res.	TS2P [7:0]							

								R	R	R	R	R	R	R	R

The software needs to read data from the corresponding address according to the required HSI clock frequency, and then write it to the FLASH_TS2P and FLASH_TPS3 registers to realize the configuration of the erasing and writing time required for the corresponding HSI frequency.

4.5.4. HSI_4M/8M/16M/22.12M/24M_EPPARA2

Address: 0x1FFF 0F24(4 MHz)、0x1FFF 0F38(8 MHz)、0x1FFF 0F4C(16 MHz)、0x1FFF 0F60(22.12 MHz)、0x1FFF 0F74(24 MHz)

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.	PERTPE [16]
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to choose to read data from the corresponding address according to the HSI clock frequency that needs to be set, and then write it into the FLASH _ PERTPE register to realize the configuration of the erasing and writing time required for the corresponding HSI frequency.

4.5.5. HSI_4M/8M/16M/22.12M/24M_EPPARA3

Address: 0x1FFF 0F28(4 MHz)、0x1FFF 0F3C(8 MHz)、0x1FFF 0F50(16 MHz)、0x1FFF 0F64(22.12 MHz)、0x1FFF 0F78(24 MHz)

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	SMER TPE[17:16]	
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to choose to read data from the corresponding address according to the HSI clock frequency that needs to be set, and then write it into the FLASH _ SMERTPE register to realize the configuration of the erasing and writing time required for the corresponding HSI frequency.

4.5.6. HSI_4M/8M/16M/22.12M/24M_EPPARA4

Address: 0x1FFF 0F2C(4 MHz)、0x1FFF 0F40(8 MHz)、0x1FFF 0F54(16 MHz)、0x1FFF 0F68(22.12 MHz)、0x1FFF 0F7C(24 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PRETPE[13:0]													
		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**
PRGTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The software needs to choose to read data from the corresponding address according to the required HSI clock frequency, and then write it to the FLASH _ PRGTPE and FLASH _ PRETPE registers to realize the configuration of the erasing and writing time required for the corresponding HSI frequency.

4.5.7. Flash User Data Bytes

Flash User Data is used to store user data, specifically defined as:

Table 4-5 USER Data Bytes organization

Page	Word	Address	Contents
26	832	0x1FFF 0D00	Bit[31:16]: store user data

Page	Word	Address	Contents
			Bit[15:0]: USER OTP MEMORY_LOCK USER OTP MEMORY_LOCK | User Data protection 0xAA55 | Read: Yes, Program and erase: No Other values | Read, Program and erase: Yes
	833	0x1FFF 0D04	Store user data
	834	0x1FFF 0D08	Store user data
	…	…	Store user data
	…	…	Store user data
	…	…	Store user data
	863	0x1FFF 0D7C	Store user data

Configuring the USER OTP MEMORY _ LOCK content will not be updated immediately and will not take effect until it is loaded on power.

After the power-on reset, it is determined whether the USER DATA area can be programmed according to the value of the loaded USER OTP MEMORY _ LOCK.

If programmable, the programming steps are as follows:

Modifying user data bytes

The write operation in the User data area is the same as the operation on the Main flash. The following steps are required:

1) Check the BSY bit to confirm that there are no ongoing Flash operations

2) If there is no ongoing flash erase or write operation, the software reads out the 32 words of the Page

3) Write KEY1 and KEY2 to the FLASH _ KEYR register to unlock the protection of the FLASH _ CR register

4) Set the PG bit and EOPIE bit in the FLASH_CR register

5) Write the 1st to 31st words to the User data area address (only 32-bit writes are accepted)

6) Set the PGSTRT bit in the FLASH_CR register

7) Write the 32nd word

8) Wait for the BSY bit to be cleared.

9) Wait for the EOP to rise, the software clears

When the step 7 is carried out, the program operation is automatically started and the BSY bit is set simultaneously.

4.6. Flash protection

The protection of Main flash area includes the following mechanisms:

- SDK (software design kit) protection, used to protect the access of specific program areas, the size is 2 KB.
- Read protection (RDP) blocks external access
- Write protection (WRP) prevents unintended writes (caused by confusion of program). The granularity of write protection is designed to be 4 KB.
- Option byte write protection, specialized unlocking design.

4.6.1. SDK area protection

The protection of Main flash area includes the following mechanisms:

Start address

Flash memory base address + SDK_STRT[3:0] x 0x800 (included)

End address

Flash memory base address + (SDK_END[3:0]+1) x 0x800(excluded)

When SDK_STRT [4: 0] is greater than SDK_END [4: 0], SDK protection is invalid; SDK protection is valid when SDK_STRT [4: 0] is less than or equal to SDK_END [4: 0].

When the protection is effective, when the flash _ SDKR register is unprotected (writing SDK _ STRT [4: 0] is equal to or greater than SDK _ END [4: 0]), the hardware will first trigger mass erase (the protected program in the SDK area has been written before, and mass erase plays a role in protecting the program in the SDK area), and then update the value of the SDK option in the flash option byte (the updated value at this time is that the SDK protection is invalid).

The configuration operation is ignored when the FLASH _ SDKR register is configured while the protection is active and the protection is not released (SDK _ STRT [4: 0] < = SDK _ END [4: 0]). At this time, the contents of the FLASH _ SDKR register will not be updated, and the contents of the register will not be loaded into the register from the SDK option in the flash option byte until the power-on reset (POR/BOR/PDR) or OBL reset.

4.6.2. Flash memory read protection

The read protection function can be activated by setting the RDP option byte and performing a POR/PDR/BOR or OBL reset to load a new RDP option byte. RDP protects flash main memory.

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

The Flash main memory is protected when the RDP option byte and complement exist correctly in pairs in the option byte.

Table 4-6 Flash memory read protection status

RDP byte value	RDP completed byte value	Read protection level
0xAA	0x55	Level 0
Any value except (0xAA and 0xAA55)		Level 1

Level 0: Unprotected

Read, write, and erase operations on the main flash are possible, and any operations on the option byte are also possible.

Level 1: Read Protection

If the RDP and its complement 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 that executes in User mode (started from main flash) and can do all operations on main flash, option byte.
- debug, boot from SRAM and boot from system memory (Boot loader): In debug mode, or when boot from SRAM or system memory (Boot loader), the main flash cannot be read-accessed. In these modes, a read or write access to the main flash generates a bus error and a HardFault interrupt.

When it is already at Level 1 (any number other than 0xAA), if you want to modify it to Level 0 (write 0xAA), the hardware will perform a mass erase operation on the main flash.

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

Area	READ Protection level	SDK Area Protection level	Boot From Main Flash (CPU)						Debug/ Excuted From RAM/ Excuted From System memory			DMA		
			User execution (From Non SDK Area)			User execution (From SDK Area)								
			Read	Write	Erase	Read	Write	Erase	Read	Write	Erase	Read	Write	Erase
Non SDK Area	0	Disable	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes	Yes	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
SDK Area		Disable	N/A	N/A	N/A	Yes	Yes	Yes	N/A	N/A	N/A	Yes	No	No
		Enable	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No
Non SDK Area	1	Disable	Yes	Yes	Yes	N/A	N/A	N/A	No	No	No	No	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
SDK Area		Disable	N/A	N/A	N/A	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A	N/A
		Enable	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No
System memory	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No
Option bytes area	x	Disable	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes	No	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
Factory bytes	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No
UID	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No

1. A mass erase command issued by any area will erase the SDK area.
2. Any modification to level 1 to level 0 triggers the hardware's mass erase to the main flash.
3. N/A means that when the SDK Area is disabled, since there is no SDK Area, there is no readout program in the SDK Area in the above table, and there is no access to the SDK Area by readout programs from other areas.
4. There are two situations for executing a program from SRAM or system memory: one is to start from SRAM or system memory, and the other is to start from another memory, and the program jumps to SRAM or system memory.

4.6.3. Flash write protection

Flash can be set to write-protected in response to unwanted write operations. Define that each WRP register controls a write protection (WRP) area with a size of 4 KB, i.e. 1 sector size. See the description of the WRP register for details.

When the WRP area is activated, no erase or write operation is allowed. Accordingly, even if only one area is set to write protection, the mass erase function does not function.

In addition, if an attempt is made to erase or write an area set to write protection, the write protection error identifier (WRPERR) of the FLASH _ SR register is set.

Note: Write protection only works on Main flash.

4.6.4. Option byte write protection

By default, option bytes are readable and write-protected. In order to obtain erase or write access to the option byte, the correct sequence needs to be written to the OPTKEYR register.

4.7. Flash interrupt

Table 4-8 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

Note: The following events do not have separate interrupt identifiers but produce a HardFault:

- Sequence error in unlocking FLASH _ CR register of Flash memory
- Write sequence wrong to unlock Flash option byte
- Write Flash operation fails to align 32-bit data
- Erase Flash (including page erase, sector erase, and mass erase) operation does not perform 32-bit data alignment
- Write operation to option byte register fails to align 32-bit data

4.8. Flash registers

4.8.1. FLASH access control 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.	Res.	Res.	LA-TENCY[1:0]	
															RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1:0	LATENCY	RW	0	Waiting state corresponding to Flash read operation: 0: No waiting state for Flash read operation (system clock at 24 MHz and below) 1: The flash read operation has a waiting state, that is, each flash read requires two system clock cycles (the system clock is greater than 24MHz and less than or equal to 48MHz) 2: The flash read operation has three waiting states, that is, each flash read requires four system clock cycles (the system clock is greater than 48MHz and less than or equal to 72MHz) Note: 3 is not configured. If 3 is configured for latency, it will be configured as 2

4.8.2. FLASH key register (FLASH_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

All register bits are write-only, readout 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	0x0000	The following values must be written consecutively to unlock the flash _ CR register and allow flash's program/erase operation KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.8.3. FLASH option key register (FLASH_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

All register bits are write-only, readout 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	0x0000 0000	The following values must be written consecutively to unlock the flash option register and allow program/erase operations for the option byte KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F

4.8.4. FLASH status 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	Res	BSY
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTV ERR	Res.	USRLOC K	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP ERR	Res.	Res.	Res.	EOP
RC_W 1		R									RC_W 1				RC_W 1

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	BSY	R	0	Busy bit This bit indicates that the operation of the flash 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	When the option and trimming bits and their inverses do not match, the hardware sets this bit. Load mismatched

Bit	Name	R/W	Reset Value	Function
				option bytes, forced to safe values, reference FLASH write option bytes. This bit is cleared by writing 1.
13	USRLOCK	R	0	Whether the userdata area is writable is indicated according to the value of the lower 16 bits of the first word of the power-on read userdata area 0: The value read is not 0xaa55, userdata is writable 1: The value read is 0xaa55, userdata is not writable This bit can only be reset by a power-on reset
4	WRPERR	RC_W1	0	When the address to be programmed/erased is in the write-protected flash area (WRP), the hardware sets this bit. When the UserData area is written/erased at USRLOCK 1, the hardware also sets this bit. Write 1 and clear the bit.
3:1	Reserved	-	-	Reserved
0	EOP	RC_W1	0	When the write/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. Write 1 and clear the bit.

4.8.5. FLASH control 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	Res.	ERRIE	EOPIE	Res.	Res.	Res.	Res.	PGSTRT	UPGSTRT	OPTSTRT	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.	Res.	SER	Res.	Res.	Res.	Res.	Res.	Res.	UPER.	UPG	MER	PER	PG
				RW							RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	The software can only set this bit. When set, the FLASH_CR register is locked. When the unlocking timing is successfully given, this bit is cleared by hardware, unlocking the FLASH _ CR register. [The software should set this bit after the program/erase operation is completed] When an unsuccessful unlock timing is given, the bit remains set until the next system reset.
30	OPTLOCK	RS	1	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 unlocking timing is successfully given, this bit is cleared by hardware, unlocking the FLASH _ CR register. [The software should set this bit 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	-	-	Reserved
27	OBL_LAUNCH	RC_W1	0	When set, this bit forces the system to reload option bytes. 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 reload complete

Bit	Name	R/W	Reset Value	Function
				1: Generate an option byte reload request, and the system generates a reset to reload the option byte.
26	Reserved	-	-	Reserved
25	ERRIE	RW	0	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	-	-	Reserved
19	PGSTRT	RW	0	The start bit of the program operation for the Flash main memory. This bit initiates a write operation to the Flash main memory, is set by software, and clears this bit by hardware after the BSY bit of the FLASH _ SR register is cleared.
18	UPGSTRT	RW	0	The start bit of the write operation in the user data area of the Flash information memory. This bit initiates a write operation in the user data area of the Flash information memory and is set by software. After the BSY bit of the FLASH _ SR register is cleared, the hardware clears this bit.
17	OPTSTRT	RW	0	Start of modification of option bytes This bit initiates the modification of the option byte. Software set, after the BSY bit of the FLASH _ SR register is cleared, hardware clears this bit. Note: When the flash option bytes are modified, the hardware automatically performs an erase operation on the entire 128 Bytes page, and then performs a program operation, which also includes automatic write of complemented code.
16:12	Reserved	-	-	Reserved
11	SER	RW	0	4 KB Sector erase operations 0: sector erase operation for flash is not selected 1: Select the flash sector erases operation Notes: 1) 1. Sector erase does not work on flash information memory. 2) 2. Sector erase does not work for areas set to WRP.
10:5	Reserved	-	-	Reserved
4	UPER	RW	0	User data Page erase operation 0: page erase operation for flash user data page is not selected 1: Select the page erase operation of the flash user data page Note: If the User data area cannot be erased, the configuration is invalid.
3	UPG	RW	0	User data area Program operations 0: program operation in user data area is not selected 1: Select program operation in the user data area Note: If the User data area cannot be erased, the configuration is invalid.
2	MER	RW	0	Mass erase operation 0: mass erase operation for flash not selected

Bit	Name	R/W	Reset Value	Function
				1: Select the mass erases operation of flash Notes: Mass erase will not work on flash information memory. Mass erase does not work when there is a WRP setting
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

4.8.6. FLASH option register (FLASH_OPTR)

Address offset: 0x20

Reset value: 0x0000 xxxx。

After the power-on reset (POR/BOR/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.	BOR_LEV[2:0]			BOR_EN
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	IWDG_STOP	Res.	Res.	Res.	RDP[7:0]							
RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19:17	BOR_LEV[2:0]	RW		000：BOR rising threshold is 1.8V and falling threshold is 1.7V 001：BOR rising threhold is 2.0V and falling threhold is 1.9V 010：BOR rising threshold is 2.2V and falling threshold is 2.1V 011：BOR rising threshold is 2.4V and falling threshold is 2.3V 100：BOR rising threshold is 2.6V and falling threshold is 2.5V 101：BOR rising threshold is 2.8V and falling threshold is 2.7V 110：BOR rising threshold is 3.0V and falling threshold is 2.9V 111：BOR rising threshold is 3.2V and falling threshold is 3.1V
16	BOR_EN	RW		BOR enable 0：BOR disabled 1：BOR enabled, BOR_LEV works
15	nBOOT1			Together with the BOOT PIN, select the boot mode

14	NRST_MODE	RW		0: Reset input only 1: GPIO function
13	WWDG _ SW	RW		0: Hardware watchdog 1: Software watchdog
12	IWDG_SW	RW		0: Hardware watchdog 1: Software watchdog
11	IWDG_STOP	RW		Independent watchdog counter freeze in Stop mode 0: Independent watchdog counter is frozen in Stop mode 1: Independent watchdog counter is running in Stop mode
10:8	Reserved			
7:0	RDP	RW		0xAA: level 0, read protection inactive Others: level 1, read protection active

4.8.7. FLASH SDK address register (FLASH_SDKR)

Address offset: 0x24

Reset value: 32'b0000 0000 0000 0000 000X XXXX 000X XXXX。

After the power-on reset (POR/BOR/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.	SA _ END [4: 0]					Res.	Res.	Res.	SA _ STRT [4: 0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:8	SDK_END[4:0]	RW		SDK area end addres, the corresponding STEP of each bit is 2KB
7:5	Reserved	-	-	Reserved
4:0	SDK_STRT[4:0]	RW		SDK area start address, the corresponding STEP of each bit is 2KB

4.8.8. FLASH WRP address register (FLASH_WRPR)

Address offset: 0x2C

Reset value: 0x0000 XXXX

After the power-on reset (POR/BOR/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**
WRP[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	-	-	Reserved
15	WRP[15]	RW	1	0: sector 15, write-protected, program and erase are not allowed 1: sector 15, no write protection
14	WRP[14]	RW	1	0: sector 14, write-protected, program and erase are not allowed

Bit	Name	R/W	Reset Value	Function
				1: sector 14, no write protection
13	WRP[13]	RW	1	0: sector 13, write-protected, program and erase are not allowed 1: sector 13, no write protection
12	WRP[12]	RW	1	0: sector 12, write-protected, program and erase are not allowed 1: sector 12, no write protection
11	WRP[11]	RW	1	0: sector 11, with write protection, program and erase are not allowed 1: sector 11, no write protection
10	WRP[10]	RW	1	0: sector 10, with write protection, program and erase are not allowed 1: sector 10, no write protection
9	WRP[9]	RW	1	0: sector 9, with write protection, program and erase are not allowed 1： sector 9 is unprotected
8	WRP[8]	RW	1	0: sector 8, with write protection, program and erase are not allowed 1: sector 8, with write protection
7	WRP[7]	RW	1	0: sector 7, with write protection, program and erase are not allowed 1： sector 7 is unprotected
6	WRP[6]	RW	1	0: sector 6, with write protection, program and erase are not allowed 1： sector 6 is unprotected
5	WRP[5]	RW	1	0: sector 5, with write protection, program and erase are not allowed 1： sector 5 is unprotected
4	WRP[4]	RW	1	0: sector 4, with write protection, program and erase are not allowed 1： sector 4 is unprotected
3	WRP[3]	RW	1	0: sector 3, with write protection, program and erase are not allowed 1： sector 3 is unprotected
2	WRP[2]	RW	1	0: sector 2, with write protection, program and erase are not allowed 1： sector 2 is unprotected
1	WRP[1]	RW	1	0: sector 1, with write protection, program and erase are not allowed 1： sector 1 is unprotected
0	WRP[0]	RW	1	0: sector 0, with write protection, program and erase are not allowed 1： sector 0 is unprotected

4.8.9. FLASH sleep time configuration register (FLASH _ STCR)

Address offset: 0x90

Reset value: 0x0000 5000

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**
SLEEP_TIME[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_EN
RW	RW	RW	RW	RW	RW	RW	RW								RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
15:8	SLEEP_TIME	RW	0x50	When the system clock selects LSI or LSE, in order to obtain a more optimized operating mode power consumption, the function of using this register can be selected

				(this function is only recommended when LSI or LSE is the system clock). When this feature is enabled, the time width for which Flash is in Sleep mode for every half of the system clock low cycle is: t_{HSI_10M} * SLEEP_TIME Note: t_{HSI_10M} is the period of HSI _ 10M; To ensure the function of Flash, the maximum value of this register is recommended to be set to 0x28. In addition, the set value of this register is stored in 0x1FFF 0F14 after leaving the factory to improve ease of use.
7:1	Reserved	-	-	Reserved
0	SLEEP_EN	RW	0	FLASH Sleep enable 1: enable flash sleep 0: disable flash sleep

4.8.10. FLASH TS0 register (FLASH_TS0)

Address offset: 0x100

Reset value: 0x0000 xxxx

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.	TS0							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TS0	RW	0xB4	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F6C 22.12MHz calibration value storage address: 0x1FFF 0F58 16MHz calibration value storage address: 0x1FFF 0F44 8MHz calibration value storage address: 0x1FFF 0F30 4MHz calibration value storage address: 0x1FFF 0F1C

4.8.11. FLASH TS1 register (FLASH_TS1)

Address offset: 0x104

Reset value: 0x0000 xxxx

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.	TS1								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS1	RW	0x1B0	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F6C 22.12MHz calibration value storage address: 0x1FFF 0F58 16MHz calibration value storage address: 0x1FFF 0F44 8MHz calibration value storage address: 0x1FFF 0F30

Bit	Name	R/W	Reset Value	Function
				4MHz calibration value storage address: 0x1FFF 0F1C

4.8.12. FLASH TS2P register (FLASH_TS2P)

Address offset: 0x108

Reset value: 0x0000 xxxx

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	TS2P							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TS2P	RW	0xB4	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F70 22.12MHz calibration value storage address: 0x1FFF 0F5C 16MHz calibration value storage address: 0x1FFF 0F48 8MHz calibration value storage address: 0x1FFF 0F34 4MHz calibration value storage address: 0x1FFF 0F20

4.8.13. FLASH TPS3 register (FLASH_TPS3)

Address offset: 0x10C

Reset value: 0x0000 xxxx

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.	TPS3										
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10:0	TPS3	RW	0x6C0	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F70 22.12MHz calibration value storage address: 0x1FFF 0F5C 16MHz calibration value storage address: 0x1FFF 0F48 8MHz calibration value storage address: 0x1FFF 0F34 4MHz calibration value storage address: 0x1FFF 0F20

4.8.14. FLASH TS3 register (FLASH_TS3)

Address offset: 0x110

Reset value: 0x0000 xxxx

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	TS3							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TS3	RW	0xB4	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F6C 22.12MHz calibration value storage address: 0x1FFF 0F58 16MHz calibration value storage address: 0x1FFF 0F44 8MHz calibration value storage address: 0x1FFF 0F30 4MHz calibration value storage address: 0x1FFF 0F1C

4.8.15. Flash page erase TPE register (Flash _ PERTPE)

Address offset: 0x114

Reset value: 0x0001 xxxx

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.	PERTPE
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERTPE															
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	-	-	Reserved
16:0	PERTPE	RW	0x14820	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F74 22.12MHz calibration value storage address: 0x1FFF 0F60 16MHz calibration value storage address: 0x1FFF 0F4C 8MHz calibration value storage address: 0x1FFF 0F38 4MHz calibration value storage address: 0x1FFF 0F24

4.8.16. FLASH sector/mass erase tpe register (FLASH_SMERTPE)

Address offset: 0x118

Reset value: 0x0001 xxxx

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.	SMERTPE
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMERTPE															
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	-	-	Reserved
16:0	SMERTPE	RW	0x14820	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F78 22.12MHz calibration value storage address: 0x1FFF 0F64 16MHz calibration value storage address: 0x1FFF 0F50 8MHz calibration value storage address: 0x1FFF 0F3C 4MHz calibration value storage address: 0x1FFF 0F28

4.8.17. FLASH program TPE register (FLASH_PRGTPE)

Address offset: 0x11C

Reset value: 0x0000 xxxx

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**
PRGTPE															
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	-	-	Reserved
15:0	PRGTPE	RW	0xxxxx	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F7C 22.12MHz calibration value storage address: 0x1FFF 0F68 16MHz calibration value storage address: 0x1FFF 0F54 8MHz calibration value storage address: 0x1FFF 0F40 4MHz calibration value storage address: 0x1FFF 0F2C

4.8.18. FLASH pre-program TPE register (FLASH_PRETPE)

Address offset: 0x120

Reset value: 0x0000 xxxx

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	PRETPE[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	PRETPE	RW	0x xxxx	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F7C 22.12MHz calibration value storage address: 0x1FFF 0F68 16MHz calibration value storage address: 0x1FFF 0F54 8MHz calibration value storage address: 0x1FFF 0F40 4MHz calibration value storage address: 0x1FFF 0F2C

5. Power control

5.1. Power supply

5.1.1. Power supply overview

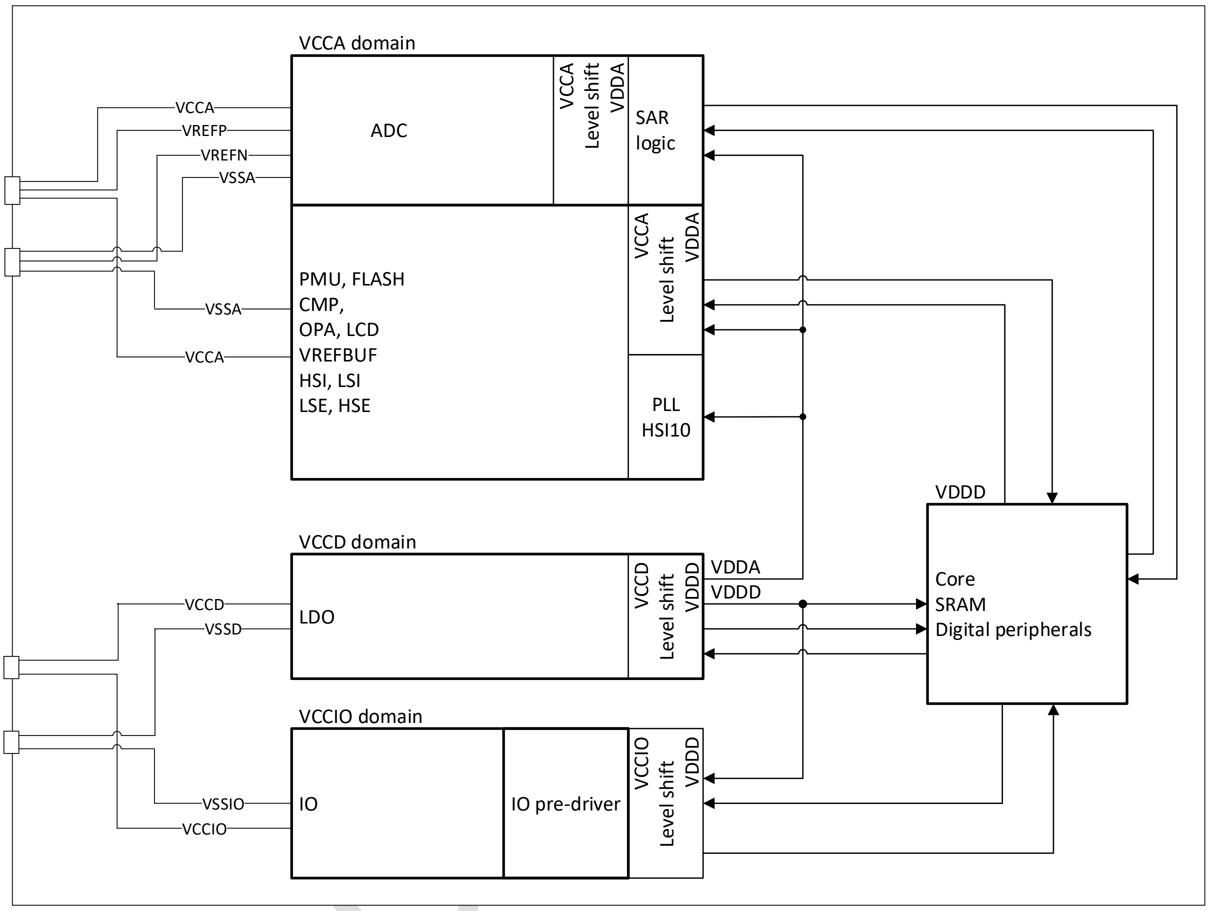


Figure 5-1 Power supply

Table 5-1 Power supply

No.	Power supply	Power value	Description
1	VCC	1.7 V ~ 5.5 V	Power supply range: 1.7V - 5.5 V. The power is supplied to the device through the power pins, with the power supply module comprising: Partial analog circuits.
2	VCCA	1.7 V ~ 5.5 V	Powers for most analog modules, sourced from the V_{CC} PAD (a dedicated power PAD can also be designed separately).
3	VCCIO	1.7 V ~ 5.5 V	Power to IO from V_{CC} PAD
4	VDD	1.2 V/1. 0 V/0. 9 V/0. 8 V	VR supplies power to the main logic circuits and SRAM inside the chip. When the MR is powered, it outputs 1.2 V. When entering the Stop mode, power can be supplied by the MR or the LPR according to the software configuration, and the LPR output is determined to be 1.2 V or 1.0 V according to the software configuration.

5.2. Voltage regulator

Chip design two VRs:

One is MR (Main regulator) and the other is LPR (low power regulator). The power supply of the VDD comes from MR or LPR depending on the operating mode of the chip.

In run mode (normal operating state): MR keeps working, outputs 1.2 v voltage, and LPR is turned off.

The software can determine whether the LDO works in MR mode and low power (Stop) mode through configuration. In low power mode, PVD and BOR work or do not work according to power-on load or register configuration; In deep low power mode, only power down detection POR1 (threshold 1.4 V) works, other PMU modules do not work (even if configured to enable).

After entering the Stop mode, the software configures whether the VDDD under the condition of LPR power supply is 1.2 V, 1.0 V, 0.9 V or 0.8 V.

The Stop mode involves the wake-up function, that is, the power supply of VDD must be switched from LPR to MR, so in addition to the extremely low power consumption of LPR, the switching time is also an important design index.

The chip system requires that the total wake-up time of Stop mode (including LPR switching to MR, HSI wake-up, Flash wake-up, CPU wake-up) be less than 5us.

5.3. Dynamic voltage scaling management

Dynamic voltage value management refers to adjusting the output VDD voltage of VR, so that the chip can run at different voltages according to application requirements, thereby obtaining corre-sponding performance and power consumption.

Two voltage ranges are defined:

■ Range 1: High performance Range

The MR provides a typical output voltage at 1.2 V (V_{DD}). It is used when the system clock fre-quency is up to 72 MHz.

■ Range 2: Low power Range

Only when the chip is in Stop mode, the setting is allowed to enter this range, and this range only works for LPR.

By default, the output of LPR mode is the default value of 1.2 V (VDDD). When the chip enters Stop mode, according to the configuration of PWR _ CR1. LPR [1: 0] and PWR _ CR1. SRAM _ RETV _ CTRL registers, the VR output is different. The details are shown in the following table:

SRAM _ RETVSEL	LPR[1:0]		SRAM voltage		Digital core voltage	Analog core voltage	System Mode	Description
			VDDP	VDDA				
0	0	0	1.2 V	SRAM _ RETV _ TRIM	1.2 V	1.2 V	-	VDDD = VDDA = 1.2 V. VDD1 = SRAM _ RETV _ TRIM VDDD and VDDA are not configured by VOS; This configuration is not recommended. When LPR = 2 'b00, SRAM _ RETVSEL should be configured to 1 to ensure that SRAM is powered normally;
0	0	1	VOS	SRAM _ RETV _ TRIM	VOS	VOS	Stop mode	VDDD = VDDA = VOS. VDD1 = SRAM _ RETV _ TRIM

SRAM _ RETVSEL	LPR[1:0]		SRAM voltage		Digital core voltage	Analog core voltage	System Mode	Description
			VDDP	VDDA				
0	1	1	1.2 V	SRAM _ RETV _ TRIM	1.2 V	1.2 V	-	Reserved If the software configu-ration LPR = 2 'b11, the hardware output LPR = 2' b00

1	0	0	1.2 V	1.2 V	1.2 V	1.2 V	Normal mode	VDDD = VDDA = 1.2 V;
								VDD1 = VDDD;
								VDDD and VDDA are not configured by VOS;

1	0	1	VOS	VOS	VOS	VOS	Stop mode	VDDD = VDDA = VOS. VDD1 = VDDD;
1	1	1	1.2 V	1.2 V	1.2 V	1.2 V	-	Reserved If the software configu-ration LPR = 2 'b11, the hardware output LPR = 2' b00

5.4. Power monitoring

5.4.1. Power-on reset (POR) / power-down reset (PDR) / brown-out reset (BOR)

The device has an integrated power-on reset (POR) / power-down reset (PDR). The module keeps working in all modes.

In addition to POR/ PDR, BOR (Brown-out reset) is also implemented. BOR can only be enabled and disabled through the Option byte.

When the BOR is turned on, the BOR threshold can be selected by the Option byte and both the rising and falling detection points can be individually configured.

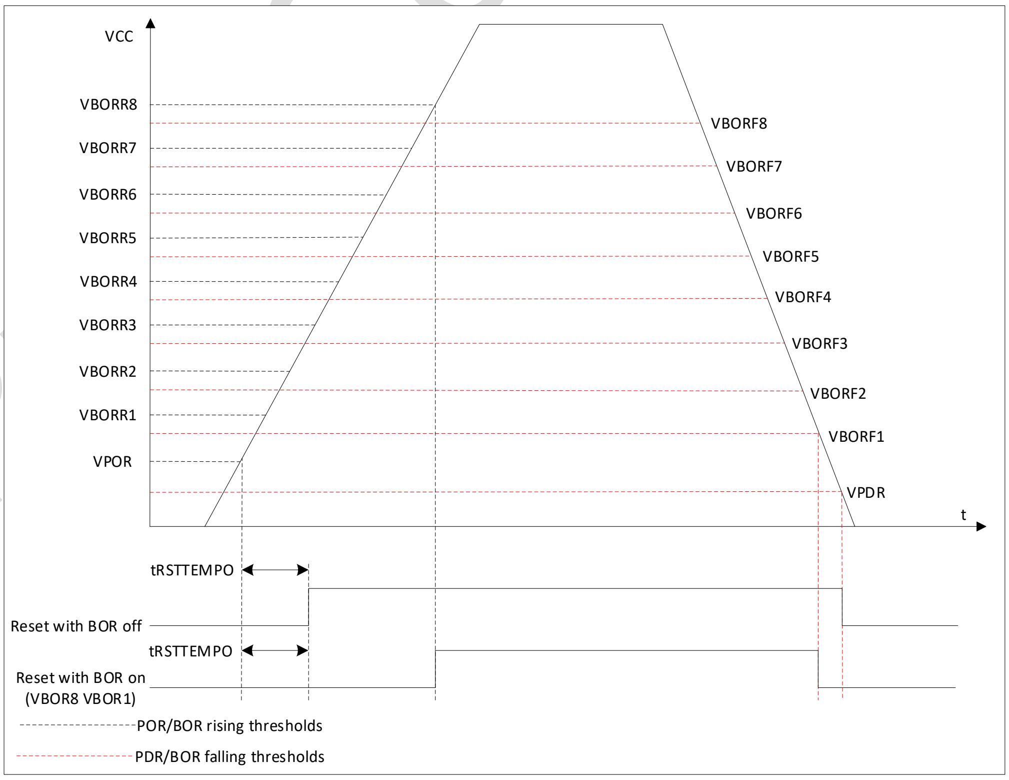


Figure 5-2 POR/PDR/BOR thresholds

6. Low power control

By default, the microcontroller is in Run mode after a system or a power Reset. When the CPU does not need to operate continuously, the chip may enter a low power mode, for example, while waiting for an external event. Software can compromise between power consumption, wake-up time, and wake-up source.

6.1. Low-power modes

6.1.1. Introduction

In addition to the run mode, the device has three low-power modes:

- Sleep mode：Peripherals can be configured to keep working when the CPU clock is off (NVIC, SysTick, etc.). It is recommended only to enable the modules that must work, and close the module after the module works.
- Stop mode：The LDO enters low-power mode. In this mode, SRAM and register contents are retained. PLL, HSI and HSE are turned off and most module clocks in the V_{DDD} domain are disabled.

In Stop mode, the LSI, LSE, RTC, LPTIM and IWDG can be kept running.

In the Stop mode, the corresponding VR state can be controlled by software and set to be powered by MR or LPR. When powered by LPR, the device consumption is reduced with longer wake-up time; When powered by MR, it has larger consumption but shorter wake-up time.

In addition, power consumption can be reduced in Run mode by:

- Reduce the system clock frequency
- For unused peripherals, Gating the peripheral clock (system clock and module clock)

To sum up, the low-power mode conversion diagram of this project is as follows.

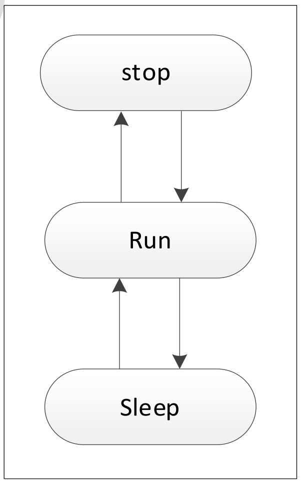


Figure 6-1 Low power mode conversion

Table 6-1 Low power entry/exit

Mode	Entry	Wake-up source	Wake-up clock	Effect	Voltage regulator	
					MR mode	LP mode
Sleep (sleep-now or sleep-on-exit)	WFI or Return from ISR	Any interrupt	Same as before entering sleep	CPU HCLK clock OFF; no effect on other clocks or analog clock sources	ON[(1)]	OFF
	WFE	Wake up event				
Stop	LDO = LP mode; SLEEPDEEP bit; 1. WFI or 2. Return from ISR or 3. WFE Note: The system clock is switched to HSI when entering, and HSISYS is not divided	Any wake up EXTI line (configured in the EXTI registers), IWDG reset and NRST	HSISYS HSI maintains the frequency configuration before entering Stop mode, without frequency division.	HSI, PLL OFF; HSE OFF; LSI and LSE can be selected as ON/OFF; LPTIM, RTC and IWDG: configured by the software Low power wake-up and some modules such as RCC keep working; The clock is off for the remaining modules.	Software Configuration Switch	Software Configuration LPR = 2 'b01

1. The software must configure the state of VR to MR mode before entering the Sleep mode.

6.1.2. Functionalities depending on the working mode

Table 6-2 Functionalities depending on the working mode(1)

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wakeup ability
CPU Core	Y	-	-	-
Flash memory	Y	Y	-(2)	-
SRAM	Y	O(3)	**-(4)**	-
Brown-out reset (BOR)	Y	Y	O	O
PVD	O	O	O	O
DMA	O	O	-	-
HSI	O	O	-	-
HSE	O	O	-	-
LSI	O	O	O	-
LSE	O	O	O	-
PLL	O	O	-	-
HSE Clock Security System (CSS)	O	O	-	-
LSE Clock Security System (CSS)	O	O	O	O
RTC	O	O	O	O
USART1/USART2/USART3	O	O	-	-
I2C	O	O	-	-
SPI1/SPI2	O	O	-	-
ADC	O	O	-	-
COMP1/COMP2	O	O	O	O
OPA1/OPA2	O	O	-	-
Temperature sensor	O	O	-	-
Timers(TIM1/TIM2/ TIM14/TIM16/TIM17)	O	O	-	-
LPTIM	O	O	O	O
IWDG	O	O	O	O
WWDG	O	O	-	-
SysTick timer	O	O	-	-
CRC	O	O	-	-

HDIV	O	O	-	-
CORDIC	O	O	-	-
LED	O	O	-	-
LCD	O	O	-	-
GPIOs	O	O	O	O

1. Legend: Y = Yes (Enable). O = Optional (Disable by default. Can be enabled by software). - = Not available
2. Flash does not power down, but no clock is provided, and enters the lowest power consumption state. At this time, FLASH requires a wake-up time of 3 us.
3. The clock of the SRAM can be turned on or off in the SLEEPING mode.
4. The SRAM is not powered down, but no clock is provided and enters the lowest power consumption state.
5. Before entering Stop mode, if LSE CSS is enabled, when there is a problem with LSE CSS, the system will wake up and enter an NMI interrupt.
6. When the LDO is in deep low power mode, the software cannot configure the module to enable. If configured, the hardware will not mask it, but it is not guaranteed to work properly.
7. When the LDO is in deep low power mode, the LSI and COMP analog modules are turned off.

6.2. Sleep mode

6.2.1. Enter Sleep mode

The Sleep mode is entered by executing the WFI (Wait For Interrupt) or WFE (Wait for Event) instructions. Two options are available to select the Sleep mode entry mechanism, depending on the SLEEPONEXIT bit in the Cortex®-M0 System Control register.

- Sleep-now: if the SLEEPONEXIT bit is cleared, the MCU enters Sleep mode as soon as WFI or WFE instruction is executed.
- Sleep-on-exit: if the SLEEPONEXIT bit is set, the MCU enters Sleep mode as soon as it exits the lowest priority ISR.

In the Stop mode, all I/O pins keep the same state as in the Run mode.

Note: When waking up with an event, HCLK needs to be configured to be undivided before entering Sleep mode.

6.2.2. Exit Sleep mode

If the WFI instruction is used to enter Sleep mode, any peripheral interrupt acknowledged by the nested vectored interrupt controller (NVIC) can wake up the device from Sleep mode.

If the WFE instruction is used to enter Sleep mode, the MCU exits Sleep mode as soon as an event occurs. Wakeup events can be spawned by:

- Enable the interrupt in the peripheral control register, not in the NVIC, and enable the SEVONPEND bit of Cortex M0 +. When the chip continues execution after waking up from the WFE, the peripheral interrupt pending bit and the peripheral NVIC IRQ channel pending bit (in the NVIC's interrupt clear pending register) must be cleared.
- configuring an external or internal EXTI line in event mode. When the CPU continues execution after waking up from the WFE, it is not necessary to clear the peripheral interrupt pending bit, or the NVIC IRQ channel pending bit corresponding to the event Line is not set.

This mode offers the lowest wake up time as no time is wasted in interrupt entry/exit.

Table 6-3 Sleep-now

Sleep-now	Description
Mode entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 and - SLEEPONEXIT = 0
Mode exit	If you enter the Sleep mode through WFI, the exit mode is: interrupt. If you enter the Sleep mode through WFE, the exit mode is: wakeup event. Reset or LSECSS NMI: - Power reset - Pin reset - IWDG Reset - LSECSS NMI Interrupt
Wakeup latency	None

Table 6-4 Sleep-on-exit

Sleep-on-exit	Description
Mode entry	WFI (wait for interrupt) while: - SLEEPDEEP = 0 and - SLEEPONEXIT = 1
Mode exit	Interrupts and events
Wakeup latency	None

6.3. Stop mode

The Stop mode is based on the Cortex®-M0+ deep Sleep mode combined with peripheral clock gating. The voltage regulator can be configured either in normal or low-power mode. In Stop mode, PLL, HSI and HSE are turned off, SRAM and register contents are preserved. LSI, LSE, LPTIM, RTC, IWDG, PVD and COMP can be configured by software, low power wake-up and some RCC logic are kept working, and the clock inputs of digital modules in the remaining V_{DDD} domain are turned off.

In the Stop mode, all I/O pins keep the same state as in the Run mode.

6.3.1. Enter Stop mode

In order to further reduce the power consumption of Stop mode, when PWR _ CR.LPR = 2 'b01 is configured, VR can enter LPR power supply.

If Flash memory programming is ongoing, the Stop mode entry is delayed until the memory access is finished.

If an access to the APB domain is ongoing, The Stop mode entry is delayed until the APB access is finished.

6.3.2. Exit Stop mode

When exiting Stop mode by issuing an interrupt or a wakeup event, the HSI oscillator is selected as system clock.

When the voltage regulator operates in low-power mode, an additional startup delay is incurred when waking up from Stop mode.

By keeping the internal regulator ON during Stop mode, the consumption is higher, although the startup time is reduced.

Table 6-5 Stop mode

Stop mode	Description
Mode entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Settings: 1） Select VR to operate under MR or LPR through the LPR register of PWR _ CR 2） The retention voltage of SRAM is selected as the VOS defined value or the value of SRAM _ RETV by the SRAM _ RETV _ CTRL bit of PWR _ CR 3） Set the wake-up time by configuring FLS _ SLPTIME bit in PWR _ CR - Set SLEEPDEEP bit in Cortex®-M0+ System Control register **Note:** In order to enter Stop mode, all EXTI line pending bits (EXTI _ PR register), all peripheral interrupt pending bits, and RTC alarm flag bits must be reset. Otherwise, the Stop mode entry procedure is ignored and program execution continues. If the application needs to turn off HSE before entering Stop mode, the system clock source must first switch to HSI and then clear the HSEON bit. In order to make the change of chip power consumption as balanced as possible, the software needs to follow the principle of step-by-step shutdown: step-by-step shutdown the clock of each module, select HSI as the system clock, turn off PLL, and turn off HSE. To shorten the wake-up time, before entering Stop mode, the system clock should be configured to select the HSI high-frequency clock, and the HPRE of the RCC _ CFGR register should be set to 0, otherwise the hardware switching clock will consume additional clock after wake-up.
Mode exit	If WFI was used for entry: - Any EXTI line configured in interrupt mode (the corresponding EXTI interrupt vector must be enabled in NVIC) If WFE was used for entry: - Any EXTI Line configured in Interrupt mode - Interrupt pending flag bit in case of CPU SEVONPEND position bit Reset or LSECSS NMI: - Power reset - Pin reset - IWDG Reset - LSECSS NMI Interrupt
Wakeup latency	LPR to MR wakeup time + HSI wakeup time + flash wakeup time Note 1: In Deep low power mode, the MR ready time is 300us ~ 750us **Note 2: Typical wake-up time in Stop mode: 3us + 2us + 3us = 8us**

6.4. Reduce the system clock frequency

In Run mode, the frequency of the system clock (SYSCLK, HCLK, PCLK) can be reduced by configuring frequency division through the prescalation register. These prescalers can also be used to reduce the frequency of peripherals before entering Sleep mode.

6.5. Peripheral clock gating

In Run mode, the AHB clock (HCLK) and APB clock (PCLK) of individual peripherals and memories can be stopped at any time to reduce power consumption.

To further reduce power consumption in Sleep mode, the clock of the peripheral may be stopped prior to execution of WFI or WFE instructions.

6.6. Power control registers

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

6.6.1. Power control register 1 (PWR_CR1)

Address offset: 0x00

Reset value: 0x0004 0000(reset by POR)

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.	SRAM _ RETV _ CTRL	Res.	Res.
													RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPR[1:0]		FLS_SLP-TIME[1:0]		Res.	Res.		DBP	Res.	Res.	Res.	Res.	Res.			
RW	RW	RW	RW				RW								

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	SRAM _ RETV _ CTRL	RW	1	SRAM retention voltage control in Stop mode 1: SRAM voltage is consistent with digital LDO output; 0: SRAM voltage is low voltage After the Stop mode wakes up, the hardware updates this register to 1. Note: After waking up from Stop mode, this register reverts to its default value.
17:16	Reserved	-	-	Reserved
15:14	LPR[1:0]	RW	0	Low power regulator under Stop mode 00: Main regulator mode 01: Low power regulator mode 10: Reserved 11: Reserved Note: After the Stop mode wakes up, the hardware updates this register to 00.
13:12	FLS_SLPTIME	RW	2’b00	In the Stop mode wake-up timing, after the HSI is stabilized, a waiting time is required before the FLASH operation. 2’b00: 5 us 2’b01: 2 us 2’b10: 3 us 2’b11: 0 us Note: When this register is set to 2'b11/2'b01, it indicates that the program is executed from SRAM after wakeup, not Flsh. And the program guarantees that Flash will not be accessed within 3us after waking up the execution program.
11:9	Reserved	-	-	Reserved
8	DBP	RW	0	RTC write protection disabled After reset, the RTC is write-protected to prevent accidental writes. To access RTC this bit must be set to 1. 0: Disable access to RTC 1: Can access RTC
7:0	Reserved	-	-	Reserved

6.6.2. Power control register 2 (PWR_CR2)

Address offset: 0x04

Reset value: 0x0000 0500(reset by POR)

Note: This register is a PVD function related 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.	FLT _ TIME [2: 0]			FLTEN	Res.	PVDT[2:0]			Res.	SRCSEL		PVDE
				RW			RW		RW				RW		RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:9	FLT_TIME	RW	3'b010	Digital filtering time configuration 110: Filtering time is approximately 30.7 ms (1024 LSI or LSE clocks) 101: Filtering time is approximately 3.8 ms (128 LSI or LSE clocks) 100: Filtering time is approximately 1.92 ms (64 LSI or LSE clocks) 011: Filtering time is approximately 480us (16 LSI or LSE clocks) 010: Filtering time is approximately 120us (4 LSI or LSE clocks) 001: Filtering time is approximately 60us (2 LSI or LSE clocks) 000: Filtering time is approximately 30us (1 LSI or LSE clock)
8	FLTEN	RW	1	Digital filter function enable control 0: Disabled 1: Enabled
7	Reserved	-	-	Reserved
6:4	PVDT[2:0]	RW	000	Voltage rising edge detection threshold (falling edge detection threshold is reduced by 0.1 V accordingly) and PVDIN detection control. 000: VPVD0 (around 1.8 V) 001: VPVD1 (around 2.0 V) 010: VPVD2 (around 2.2 V) 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: VPVD7 (around 3.2 V)
3	Reserved	-	-	Reserved
2	SRCSEL	RW	0	PVD detects power selection. 0: VCC 1: Detect PB7 pin If this position is 1, the voltage on PB7 is internally compared to VREFINT (including rise and fall thresholds). In this case, the setting of the PVDT register is invalid.
1	Reserved	-	-	Reserved
0	PVDE	RW	0	Voltage detection enable bit 0: Voltage detection not enabled 1: Voltage detection enabled PVDE write protection if SYSCFG _ CFG2.PVD _ LOCK = 1. Write protection is reset only when the system is reset.

6.6.3. PWR status register (PWR _ SR)

Address offset: 0x04

Reset value: 0x0000 0500(reset by POR)

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.	PVDO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				R											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	PVDO	R		PVD test result output. 0: The detected VCC or PB7 exceeds the comparison threshold of PVD selection 1: The detected VCC or PB7 is lower than the comparison threshold of PVD selection
10:0	Reserved	-	-	Reserved

7. Reset

Two resets are designed in the chip: power reset and system reset.

7.1. Reset source

7.1.1. Power reset

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

- Power-on/power-down reset (POR/PDR)
- Brown-out reset (BOR)

7.1.2. System reset

A system reset sets all registers to their reset values except the reset flags in the clock control/status register and the registers in the RTC domain. A system reset occurs when the following events occur:

- Reset of NRST pin
- Windowed watchdog reset (WWDG)
- Independent watchdog reset (IWDG)
- SYSRESETREQ software reset
- option byte load reset (OBL)

The reset source can be identified by checking the reset identification bit of the RCC _ CSR register.

7.1.3. NRST pin (external reset)

Through specific option bits (NRST_MODE), the NRST pin is configurable for operating as (see the option byte description for specific configuration):

- Reset input

In this mode, any valid reset signal on the NRST pin is propagated to device internal logic, but resets generated internally by the device are not visible on the pin.
In this mode, the GPIO functionality (PF2) is not available.
After this reset is generated, it is not necessary to re-perform an operation such as read Trimming, and step 6 of Table 23 is directly performed.
The pulse generator guarantees a minimum reset pulse duration of 20 µs for each internal reset source to be output on the NRST pin. For the glitch filter circuit, HSI _ 10M is used for counting processing, and the clock source is turned off by default. The low level of NRST turns on the clock, and the high level turns off the clock.

- GPIO

In this mode, the pin can be used as PF2 standard GPIO. The reset function on Pin is not valid. Reset is only possible from device internal reset sources, and it is not propagated to the pin.

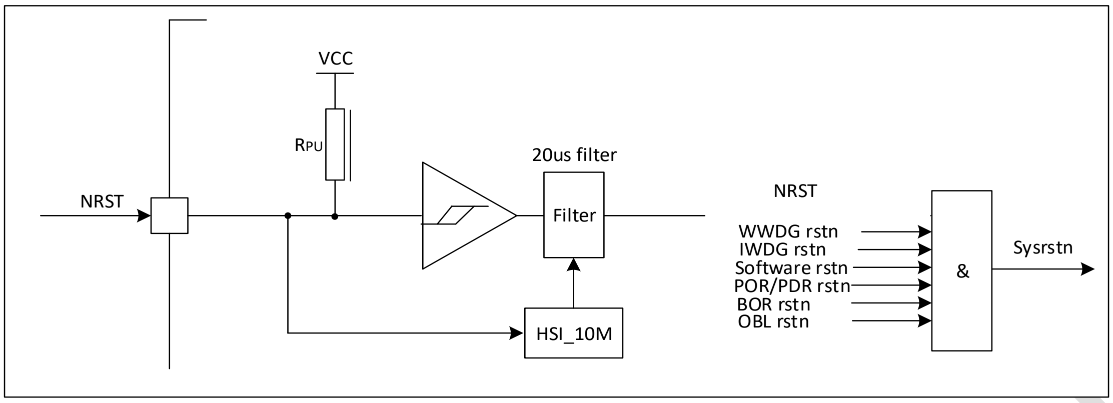


Figure 7-1 Simplified diagram of the reset circuit

7.1.4. Watchdog reset

See Watchdog. After this reset is generated, there is no need to re-perform read Trimming operations.

7.1.5. Software reset

Software reset can be achieved by setting the SYSRESETREQ bit of the ARM M0 + interrupt and reset control register. After this reset is generated, there is no need to re-perform read Trimming operations.

7.1.6. Option byte reload reset

By configuring FLASH _ CR.OBL _ LAUNCH = 1, the software generates the option byte load reset, thus starting the option byte to load again. After this reset is generated, there is no need to re-perform read Trimming operations

8. Clocks

8.1. Clock sources

8.1.1. External high speed clock HSE

The external high speed clock (HSE) comes from two sources:

- By connecting the external crystal (crystal) and cooperating with the internal vibration circuit, the clock signal of 4-32MHz is generated
- Input high-speed clock source directly from outside

Circumscribed crystal

The 4-32MHz crystal has very high accuracy. The HSERDY flag bit of RCC _ CR register indicates whether HSE is stable. The HSE may be turned on or off by the HSEON bit.

External clock source (HSE bypass)

In this mode, an external clock source is supplied directly to the device. The software selects this mode via the HSEBYP and HSEON bits of RCC _ CR. The external clock source will be input to the chip through PF0, and PF1 will be used as GPIO.

8.1.2. External low speed clock LSE

The external low speed clock (LSE) comes from two sources:

- Through external crystal oscillator and internal oscillator circuit, 32.768 kHz clock signal is generated.
- Input high-speed clock source directly from outside

The LSERDY flag bit of the RCC _ BDCR register indicates whether the LSE is stable. The LSE may be turned on or off by the LSEON bit. The driving capability can be tuned by LSEDRV [1: 0] to obtain a compromise in robustness and short start-up time.

External clock source (LSE bypass)

In this mode, an external clock source is supplied. The software selects this mode via the LSEBYP and LSEON bits of RCC _ CR. The external clock source is input to the inside of the chip through PF10, which is used as a GPIO.

8.1.3. Internal high-speed clock HSI

Internal high-speed clock, as the most important source of on-chip system clock. The center frequency of the HSI clock source is designed to be 24MHz.

8.1.4. HIS_10M

HSI _ 10M is a clock with a center frequency of 10MHz, which is used to sample the reset pin and control the flash sleep signal in the 32.768 kHz operation mode. When the NRST pin is low or enters the 32.768 kHz operation mode, the module starts to work, otherwise it is closed.

8.1.5. Internal low speed clock LSI

Internal low-speed clock, as the clock of RTC, IWDG and LPTIM, and as the system clock when the chip is running at low speed. The clock center frequency is designed at 32.768 kHz.

8.1.6. PLL

The PLL may be used to multiply the frequency of the HSI or HSE. The PLL must be configured before it can be enabled. Once the PLL is enabled, these configured registers cannot be changed.

8.2. Clock tree

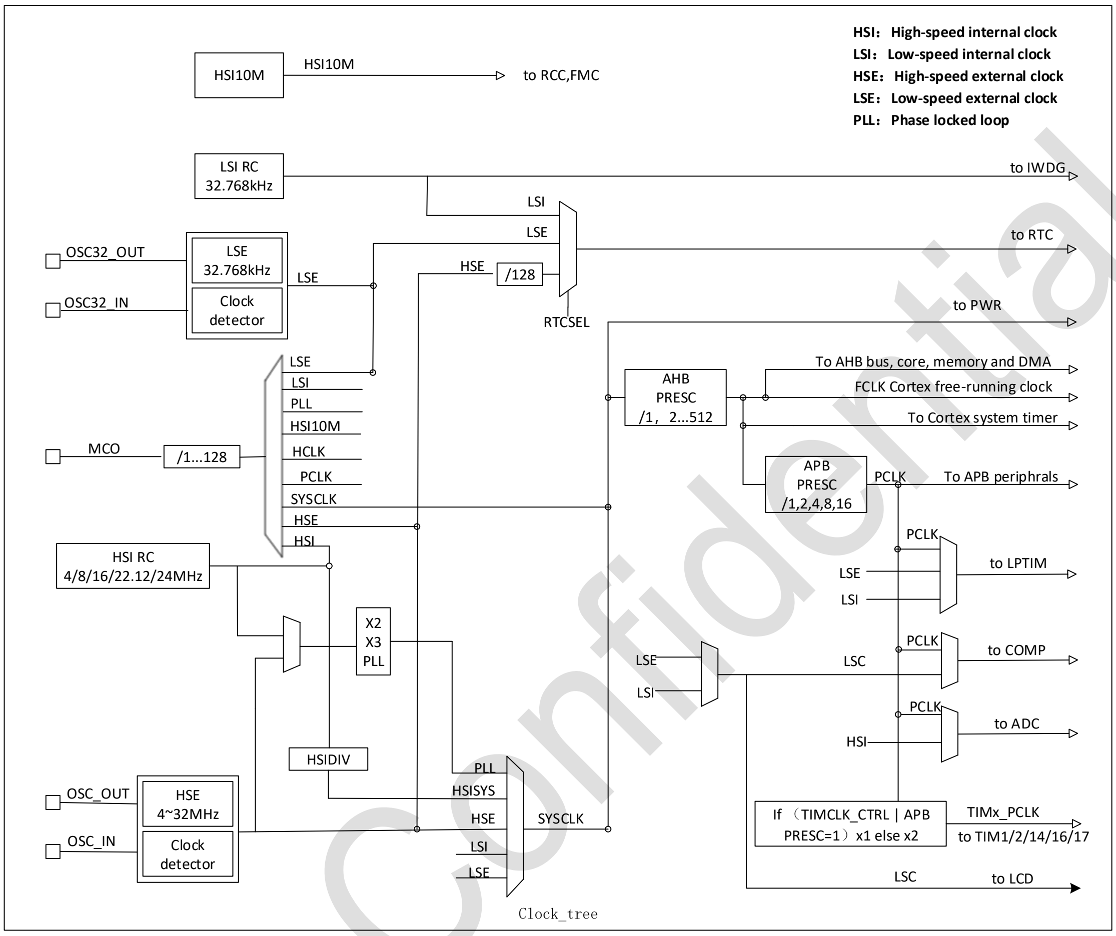


Figure 8-1 System clock structure diagram

8.3. Clock security system (CSS)

8.3.1. HSE _ CSS

The HSE clock safety system can be activated by software by configuring RCC _ CR.CSSON. In this case, after the HSE starts, the clock detection function is turned on. When the HSE is turned off, the clock detection function is turned off.

If a clock failure is found on the HSE, the HSE will be automatically shut down, and the clock failure event is sent to the brake inputs of TIM1 (advanced timer) and TIM16/TIM17 (general purpose timer), and an interrupt is generated to notify the software of the failure (Clock Security System Interrupt CSSI), which in turn allows the MCU to perform rescue operations. The CSSI is linked to the NMI (Non-maskable interrupt) of Cortex-M0 +.

Note: Once the CSS is enabled and if the HSE clock fails, the CSS interrupt occurs and an NMI is automatically generated. The NMI will be executed indefinitely unless the CSS interrupt pending bit is cleared. As a consequence, in the NMI ISR user must clear the CSS interrupt by setting the CSSC bit in the Clock interrupt register (RCC_CIR).

If the HSE is directly or indirectly used as the system clock (indirectly meaning that it is used as the input of the PLL and the PLL is used as the system clock), the clock Failure will cause the system clock to automatically switch to the HSI and shut down the HSE. If the clock failure when HSE is the input clock of the PLL, the PLL will also be turned off.

8.3.2. LSE _ CSS

The LSE clock security system can be activated by software by configuring RCC _ BDCR.LSECS-SON. In this case, after the LSE starts, the clock detection function is turned on. When the LSE is turned off, the clock detection function is turned off.

If a clock failure is found on the LSE, the LSE will be automatically shut down, and the clock failure event is sent to the brake inputs of TIM1 (advanced timer) and TIM16/TIM17 (general purpose timer), and an interrupt is generated to notify the software of the failure (Clock Security System Interrupt CSSI), which in turn allows the MCU to perform rescue operations. The CSSI is linked to the NMI (Non-maskable interrupt) of Cortex-M0 +.

Note: Once the LSECSS is enabled and if the LSE clock fails, the CSS interrupt occurs and an NMI is automatically generated. The NMI will be executed indefinitely unless the CSS interrupt pending bit is cleared. As a consequence, in the NMI ISR user must clear the CSS interrupt by setting the CSSC bit in the Clock interrupt register (RCC_CIR).

If the LSE oscillator is used as the system clock, a detected failure causes a switch of the system clock to the LSI oscillator and the disabling of the LSE oscillator. At the same time, if the LPTIM and RTC counting clocks select LSE, they will also automatically switch to LSI.

8.4. Output clock capability

In order to facilitate board-level applications, save BOM costs, and meet debugging requirements, the device needs to provide clock output functions. That is, the MCO signal (parallel frequency division) in the table below is used to realize the clock output function through the multiplexing function of GPIO.

Table 8-1 Output clock selection

Clock Source/Internal Clock	MCO output clock source
HSI	√
HSE	√
PLL	√
LSE	√
LSI	√
HSI_10M	√
SYSCLK	√
HCLK	√
PCLK	√

Note: When the MCO clock source is switched and the GPIO AF function is selected as the initial stage of the MCO, the MCO may generate glitches and need to be avoided for this period of time.

8.5. TIM14 internal and external clock calibration

Due to factors such as temperature, voltage, process and production, the frequency of internal clock sources (such as HSI, LSI, etc.) drift. Therefore, it is necessary to take some necessary measures to calibrate the frequency drift according to the changes of the external working environment of the system.

The basic idea of clock drift processing is: when the external environment of the system changes, the internal clock of the chip is dynamically measured in real time to detect and find problems. Then, the internal clock calibration parameters are fine-tuned by software, so as to achieve the purpose of dynamic calibration.

8.5.1. HSI calibration

HSI clock calibration is divided into two parts: clock detection and clock calibration.

Clock measurement

The basic concept consists in providing a relative measurement (for example, the HSI/LSE ratio): the measurement accuracy is therefore closely related to the ratio between the two clock sources. The larger the ratio, the better the measurement.

The internal clock cycle can be measured by the number of HSI clock counts between successive edges of the LSE signal. Using LSE's high accuracy (ppm level), users can measure clock frequencies at the same resolution, and can compensate for frequency deviations related to production, process, temperature and voltage by fine-tuning the clock source.

The HSI oscillator has dedicated user-accessible calibration bits for this purpose. If LSE is not available, HSE/32 can be selected to achieve the most accurate calibration possible. The frequency of the HSI is measured by inputting the capture signal through channel 1 of the TIM 14.

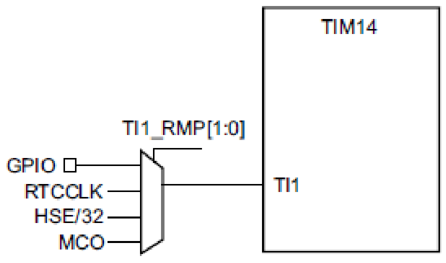


Figure 8-2 Frequency measurement vs. TIM14 capture mode

The input capture channel of the Timer 14 may be a GPIO or an in-chip clock. The selection of these clocks is achieved through the TI1 _ RMP [1: 0] register of TIM14 _ OR. The four options are shown as follows:

- TIM14 channel 1 is connected to GPIO
- TIM14 Channel 1 connects RTC Clock
- TIM14 Channel 1 connects to HSE/32 Clock
- TIM14 channel 1 is connected to MCO (Microcontroller clock output)

Clock frequency division

Once an HSI clock abnormality is detected, the software is notified for processing by interrupting. Then, the internal clock calibration parameters are fine-tuned by software, so as to achieve the purpose of dynamic calibration.

The main purpose of connecting LSE to the input capture of TIM14 channel 1 through the MCO multiplexer is to accurately measure HSI (in this case, HSI should be set to the system clock source). A count of the number of HSI clocks during changing edges of two consecutive LSE signals, such a mechanism provides a measure of the internal clock cycle.

This also makes full use of the high accuracy (ppm) of LSE when external crystal is connected, so that it is possible to determine the internal clock frequency with the same resolution, and then Trimming the clock source to compensate for frequency drift related to process, temperature, and voltage.

HSI is therefore designed with dedicated user-accessible calibration register bits.

The basic principle of this implementation mechanism is a relative measure (for example, the ratio of HSI/LSE): the accuracy will thus be closely related to the ratio of the frequencies of the two clock sources. The higher the ratio, the better the measurement effect.

8.5.2. LSI Calibration

Like HSI, the clock frequency of LSI will drift due to voltage, temperature, process and production. The calibration of LSI is performed by using HSE or HSI with a large frequency difference, and the calibration method is similar to HSI.

The calibration of the LSI is to connect the output of the LSI and the input capture of the TIM 14 The HSE is defined as the system clock source, and the number of clocks of the HSE in two consecutive LSIs provides a measure of the LSI period.

In principle, it is still the relationship of relative frequencies, that is, the frequency ratio of HSE/LSI: the calibration accuracy is closely related to this, and the larger the ratio value, the better the effect of the measurement.

8.6. Reset/clock register

The module's registers can be accessed in words (32 bits), half-words (16 bits), and bytes (8 bits).

8.6.1. Clock control register (RCC _ CR)

Address offset: 0x00

Reset value: 0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	PLL RDY	PLL ON	Res.	Res.	Res.	Res.	CSS ON	HSE BYP	HSE RDY	HSE ON
						R	RW					RS	RW	R	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	HSIDIV[2:0]			HSI RDY	Res.	HSION	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		RW			R		RW								

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	PLLRDY	R	0	PLL clock ready flag. Hardware set, indicating that the PLL clock is locked 0: PLL unlocked 1: PLL locked
24	PLLON	RW	0	PLL enable The software can be set and cleared. Cleared by hard-ware when entering Stop mode. This bit cannot be reset if the PLL clock is used as the system clock. 0: PLL OFF

Bit	Name	R/W	Reset Value	Function
				1: PLL ON
23：20	Reserved	-	-	Reserved
19	HSE_CSSON	RS	0	Clock safety system enabled. Software set enables clock safety system. When this bit is set, if HSE is ready, the hardware will perform clock detection. When the clock failure is found, the hardware will disable the clock detection. This bit can only be set, and clearing can only be reset. 0: Clock safety system OFF (clock detection OFF) 1: Clock safety system ON (clock detection ON if HSE is stable, otherwise OFF)
18	HSEBYP	RW	0	Bypass HSE external Crystal, select pin input clock. The software sets and clears, and when bypass the external crystal, connect the external pin to input the clock. The external clock must be enabled with HSEON. The HSEBYP bit is set only when the HSE external crystal is not enabled. 0: HSE external crystal is not bypassed 1: HSE external crystal is bypassed, and the external pipe pin input clock
17	HSERDY	R	0	HSE clock ready flag bit The hardware is set, indicating that HSE is stable. 0: HSE is not ready 1: HSE ready Note: When HSEON clears, HSERDY clears immediately after 6 HSE clock cycles
16	HSEON	RW	0	HSE Clock Enable Set and cleared by software. Cleared by hardware when entering Stop mode. This bit cannot be reset if the HSE oscillator is used directly or indirectly as the system clock. 0: HSE OFF 1: HSE ON
15:14	Reserved	-	-	Reserved
13:11	HSIDIV[2:0]	RW	0	HSI clock division factor. The software controls these bits to set the frequency division coefficient of the HSI, generating the HSISYS clock 000：1 001：2 010：4 011：8 100：16 101：32 110：64 111：128
10	HSIRDY	R	0	HSI clock ready flag. Set by hardware to indicate that the HSE oscillator is stable This bit is only valid if HSION = 1. 0: HSI OSC not ready; 1: HSI OSC ready; When HSION is cleared, HSIRDY will be pulled down immediately after being among the 6 HSIs.
9	Reserved	-	-	Reserved
8	HSION	RW	1	HSI clock enable Set and cleared by software. Cleared by hardware to Stop the HSI oscillator when entering Stop mode. When the HSI is directly or indirectly used as the system clock (also when exiting the Stop mode, or when the HSE is used as the system clock and a failure occurs). 0: HSI OFF 1: HSI ON Note: When HSION is cleared, HSI will generate an additional 3, and HSIRDY will be cleared after 3 HSIs.
7:0	Reserved	-	-	Reserved

8.6.2. Internal clock source calibration register (RCC _ ICSCR)

Address offset: 0x04

Reset value: 0x0100 1100, reset by POR/BOR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSI_TRIM[8:0]								
							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**
HSI_FS[2:0]			HSI_TRIM[12:0]												
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved		-	Reserved
24:16	LSI_TRIM	RW	0x100	Internal low speed clock frequency calibration. After power-on, the chip hardware will write the factory information (stored in 0x1FFF 0FB0) into this register, so that the LSI can output an accurate 32.768 KHz frequency. By rewriting the register value, the software increases (decreases) the output frequency of LSI by about 0.2% for every increase (decrease) of 1. This register reset is only a power-on reset.
15:13	HSI_FS	RW	3'b000	HSI frequency: 000: 4 MHz 001: 8 MHz 010: 16 MHz 011: 22.12 Mhz 100: 24 MHz > = 101: 4 MHz After power-on, 4MHz is selected by default. After the option byte load is completed, the hardware switches to 8MHz by loading the flash trimming page data. After the system reset, the hardware is also switched to 8MHz.
12:0	HSI_TRIM	RW	0x1100	HSI clock frequency calibration value. After power-on, the hardware uses the default calibration value of HSI 4MHz. When Trimming, the factory information (stored in 0x1FFF 0FAC) will be written to this register. The software reads out the data stored at the corresponding address in the information area and writes it to the register to realize the calibration at the specific output frequency of HSI. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0F10 22.12MHz calibration value storage address: 0x1FFF 0F0C 16MHz calibration value storage address: 0x1FFF 0F08 8MHz calibration value storage address: 0x1FFF 0F04 4MHz calibration value storage address: 0x1FFF 0F00 The register value can also be modified by writing the calibration value into the register, which is the center value. For every increase (decrease) of 1, the output frequency of the HSI increases (decreases) by about 0.1%. HSI _ TRIM [12: 9]: coarse tuning, HSI _ TRIM [8: 0] fine tuning. During trim, it needs to be controlled in two stages, such as changing the fine adjustment without changing the coarse adjustment.

8.6.3. Clock configuration register (RCC_CFGR)

Address offset: 0x08

Reset value: 0x0000 0000

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

Res.	PPRE[2:0]	HPRE[3:0]	Res.	Res.	SWS[2:0]	SW[2:0]
	RW	RW			R	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30:28	MCOPRE[2:0]	RW	0	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 It is recommended to set these bits before the MCO output is enabled.
27:24	MCOSEL[3:0]	RW	0	MCO Selection 0000: no clock, MCO output disabled0001: SYSCLK0010: HSI10M0011: HSI0100: HSE0101: PLL CLK0110: LSI0111: LSE 1000: HCLK 1001: PCLK Others: no clock Note: The output clock may be incomplete during the clock startup or switching phase.
23:15	Reserved	-	-	Reserved
14:12	PPRE[2:0]	RW	0	This bit is controlled by software. To generate the PCLK clock, it sets the division coefficients of HCLK as follows: 0xx：1 100：2 101：4 110：8 111：16
11:8	HPRE[3:0]	RW	0	AHB clock division factor. The software controls this bit. To generate the HCLK clock, it sets the division coefficients of SYSCLK as follows: 0000: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 64 1101: 128 1110: 256 1111: 512 others: reserved To ensure the normal operation of the system, the appropriate frequency needs to be configured according to the VR power supply situation. Note: It is recommended to switch the frequency division coefficients gradually.
7:6	Reserved	-	-	Reserved
5:3	SWS[2:0]	R	0	System clock switching status bit These bits are controlled by hardware and indicate which clock source is currently being used as the system clock: 000: HSISYS 001: HSE 010: PLL CLK 011: LSI

Bit	Name	R/W	Reset Value	Function
				100: LSE Others: Reserved
2:0	SW[2:0]	RW	0	System clock source select bit. These bits are controlled by software and hardware and are used to select the system clock: 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE Others: Reserved Scenarios where the hardware is configured as HSISYS include: 1) The system exits from Stop mode 2) Software configuration 001 (HSE), HSE failure occurs (HSE is the system clock source, or HSE is the PLL input, PLL is the system clock source)

8.6.4. PLL configuration register (RCC _ PLLCFGR)

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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLLMUL[1:0]		Res.	PLLSRC
												RW			RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:2	PLLMUL[1:0]	RW	2’b0	PLL frequency doubling factor 00: x2 01: x3 10: x4 (for test) 11: reserved In user mode, 2 'b10 is also used as reserved.
1	Reserved			
0	PLLSRC	RW	0	PLL clock source selection: 0: HSI 1: HSE

8.6.5. External clock source control register (RCC _ ECSCR)

Address offset: 0x10

Reset value: 0x0003 _ 0003

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.	LSE_STARTUP		Res.		LSE_DRIVER	
										RW				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.	HSE_STARTUP		Res.	HSE_DRIVER	
											RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	Res	-	Reserved

21:20	LSE_STARTUP	RW	0x0	LSE crystal oscillator stabilization time selection.
				LSEBYP=0：
				00: 4096 LSE clock cycles;
				01: 2048 LSE clock cycles;
				10: 8192 LSE clock cycles;
				11: Direct output regardless of stabilization time;
				LSEBYP=1：

Bit	Name	R/W	Reset Value	Function

				00: 2048 LSE clock cycles;
				01: 1024 LSE clock cycles;
				10: 4096 LSE clock cycles;
				11: Direct output regardless of stabilization time;

19:18	Reserved	Res	-	Reserved
17:16	LSE_DRIVER	RW	0x3	LSE drive capacity setting, default 11 00: reserved 01: Idd 315 nA, gm 3.5 uA/V 10: Idd 500nA, gm 7.5 uA/V 11: Idd 630 nA, gm 10 uA/V
15:5	Reserved		-	

4:3	HSE_STARTUP	RW	0x0	HSE stabilization time selection.
				HSEBYP=0:
				00: 4096 LSE clock cycles;
				01: 2048 LSE clock cycles;
				10: 8192 LSE clock cycles;
				11: Direct output regardless of stabilization time;
				HSEBYP=1:
				00: 2048 LSE clock cycles;
				01: 1024 LSE clock cycles;
				10: 4096 LSE clock cycles;
				11: Direct output regardless of stabilization time;

3:2	Reserved	Res	-	Reserved
1:0	HSE _ DRV	RW	0x3	HSE drive capability setting, default 11 00: reserved 01: gm 3.5 mA/V 10: gm 7.5 mA/V 11: gm 10 mA/V

8.6.6. Clock interrupt enable register (RCC _ CIER)

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.	PLLRDYIE	HSERDYIE	HSIRDYIE	Res.	LSERDYIE	LSIRDYIE
										RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	PLLRDYIE	RW	0	PLL ready interrupt enabled. 0: Disabled 1: Enabled
4	HSERDYIE	RW	0	HSE ready interrupt enable 0: Disabled 1: Enabled
3	HSIRDYIE	RW	0	HSI ready interrupt enable 0: Disabled 1: Enabled
2	Reserved	-	-	Reserved
1	LSERDYIE	RW	0	LSE ready interrupt enable 0: Disabled 1: Enabled
0	LSIRDYIE	RW	0	LSI ready interrupt enable 0: Disabled 1: Enabled

8.6.7. Clock interrupt flag register (RCC _ CIFR)

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**
Res.	Res.	Res.	Res.	Res.	Res.	LSE CSS F	CSS F	Res.	Res.	PLL RDY F	HSE RDY F	HSI RDY F	Res.	LSE RDY F	LSI RDY F
						R	R			R	R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSF	R	0	LSE Clock security system interrupt flag. Set by hardware when a failure is detected in the LSE oscillator. 0: No clock security interrupt caused by LSE clock failure 1: Clock security interrupt caused by LSE clock failure Write LSECSSC register 1 to clear this bit.
8	CSSF	R	0	HSE clock safe system interrupt identification bit. Set by hardware when a failure is detected in the HSE oscillator. 0: No clock security interrupt caused by HSE clock failure 1: Clock security interrupt caused by HSE clock failure Write CSSC register 1 to clear this bit.
7:6	Reserved	-	-	Reserved
5	PLLRDYF	R	0	PLL ready interrupt identification bit When PLL lock and PLLRDYIE position bit, the hardware sets. Cleared by software setting the PLLRDYC bit. 0: No clock ready interrupt caused by PLL lock 1: Clock ready interrupt caused by PLL lock
4	HSERDYF	R	0	HSE ready interrupt identification bit When HSE is stable and HSERDYIE is enabled, this bit is set by hardware. Cleared by software setting the HSERDYC bit. 0: No clock ready interrupt caused by HSE 1: There is a clock ready interrupt caused by HSE
3	HSIRDYF	R	0	HIS ready interrupt identification bit Set by hardware when the HSI clock becomes stable and HSIRDYIE is enabled. Cleared by software setting the HSIRDYC bit. 0: No clock ready interrupt caused by the HSI oscillator 1: Clock ready interrupt caused by the HSI oscillator
2	Res.	-	-	Reserved
1	LSERDYF	R	0	LSE ready interrupt identification bit Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSER-DYC bit. 0: No clock ready interrupt caused by the LSE oscillator 1: Clock ready interrupt caused by the LSE oscillator
0	LSIRDYF	R	0	LSI ready interrupt identification bit Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSER-DYC bit. 0: No clock ready interrupt caused by the LSI oscillator 1: Clock ready interrupt caused by the LSI oscillator

8.6.8. Clock interrupt clear register (RCC _ CICR)

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.	LSECSS C	CSS C	Res.	Res.	PLL RDY C	HSE RDY C	HSI RDY C	Res.	LSE RDY C	LSI RDY C
						W	W			W	W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSC	W	0	LSE Clock security system interrupt flag. 0：No effect； 1: Clear LSECSSF flag
8	CSSC	W	0	Clock safe interrupt clear bit. 0: No effect. 1: Clear CSSF flag bit.
7:6	Reserved	-	-	Reserved
5	PLLRDYC	W	0	PLL ready flag is cleared. 0: No effect. 1: Clear PLLRDYF bit.
4	HSERDYC	W	0	HSE ready flag cleared. 0: No effect. 1: Clear the HSERDYF bit.
3	HSIRDYC	W	0	HSI ready flag cleared. 0: No effect. 1: Clear HSIRDYE bit.
2	Reserved	-	-	Reserved
1	LSERDYC	W	0	The LSE ready flag is cleared. 0: No effect. 1: Clear LSERDYE bit.
0	LSIRDYC	W	0	The LSI ready flag is cleared. 0: No effect. 1: Clear LSIRDYE bit.

8.6.9. I/O interface reset register (RCC _ IOPRSTR)

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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIO F RST	Res.	Res.	Res.	GPIO B RST	GPIO A RST
										RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	GPIOFRST	RW	0	I/O PortF reset. 0: no effect; 1: PortF I/O restoration
4:2	Reserved	-	-	Reserved
1	GPIOBRST	RW	0	I/O PortB reset. 0: no effect; 1: I/O PortB reset.
0	GPIOARST	RW	0	I/O PortA reset. 0: no effect; 1: I/O PortA reset

8.6.10. AHB peripheral reset register (RCC _ AHBRSTR)

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.	CORDI CRST	DIVRS T	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.	CR C RST	Res.	Res.	Res.	~~FLASH RST~~	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DM A RST
			RW				~~RW~~								RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved

25	CORDICRST	RW	0	Digital coprocessor CORDIC module reset.
				0: no effect;
				1: Digital coprocessor CORDIC module reset;

24	DIVRST	RW	0	Divider module reset.
				0: no effect;
				1: Divider module reset;

23:13	Reserved	-	-	Reserved
12	CRCRST	RW	0	CRC reset 0: no effect; 1: CRC reset
11:9	Reserved	-	-	Reserved
8	Reserved	-	-	Reserved
7:1	Reserved	-	-	Reserved
0	DMARST	RW	0	DMA reset. 0: no effect; 1: DMA module reset

8.6.11. APB peripheral reset register 1 (RCC _ APBRSTR1)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM RST	OPARST	Res.	PWR RST	DBG RST	Res.	Res.	Res.	Res.	I2C2 RST	I2C RST	Res.	Res.	USART3 RST	USART2 RST	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**
Res.	SPI2 RST	Res.	Res.	WWDG RST	RTC AP-BRST	Res.	Res.	Res.	Res.	~~TIM7 RST~~	~~TIM6 RST~~	Res.	Res.	~~TIM2 RST~~	TIM2 RST
	RW			RW	RW					~~RW~~	~~RW~~			~~RW~~	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMRST	RW	0	Low Power Timer reset 0: no effect; 1: Reset
30	OPARST	RW	0	OPA reset 0: no effect; 1: Reset
29	Reserved	-	-	Reserved
28	PWRRST	RW	0	Power interface reset 0: no effect; 1: Reset
27	DBGRST	RW	0	DBG reset 0: no effect; 1: Reset
26:22	Reserved	-	-	Reserved
22	I2C2RST	RW	0	I2C2 reset 0: no effect; 1: Reset
21	I2C1RST	RW	0	I^{2}C1 reset. 0: no effect; 1: Reset
20	Reserved	-	-	Reserved
19	Reserved	-	-	Reserved
18	USART3RST	RW	0	USART3 reset 0: no effect; 1: The module is reset
17	USART2RST	RW	0	USART2 reset 0: no effect; 1: The module is reset
16:15	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
14	SPI2RST	RW	0	SPI2 reset 0: no effect; 1: Reset
13:12	Reserved	-	-	Reserved
11	WWDGRST	RW	0	WWDG reset 0: no effect; 1: Reset
10	RTCAPBRST	RW	0	RTC module APB reset. 0: no effect; 1: Reset
9:1	Reserved	-	-	Reserved
0	Reserved	-	-	Reserved

8.6.12. APB peripheral reset register 2 (RCC _ APBRSTR2)

Address offset: 0x30

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.	LCDRST	Res.	COMP2RST	COMP1RST	ADCRST	Res.	TIM17RST	TIM16RST	Res.
							RW		RW	RW	RW		RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14RST	USART1RST	Res.	SPI1RST	TIM1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGRST
RW	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	LCDRST	RW	0	LCD reset 0: no effect; 1: Reset
23	Reserved	-	-	Reserved
22	COMP2RST	RW	0	COMP2 reset 0: no effect; 1: Reset
21	COMP1RST	RW	0	COMP1 reset 0: no effect; 1: Reset
20	ADCRST	RW	0	ADC reset. 0: no effect; 1: Reset
19	Reserved	-	-	Reserved
18	TIM17RST	RW	0	TIM17 reset 0: no effect; 1: Reset
17	TIM16RST	RW	0	TIM16 reset 0: no effect; 1: Reset
16	Reserved	-	-	Reserved
15	TIM14RST	RW	0	TIM14 reset. 0: no effect; 1: Reset
14	USART1RST	RW	0	USART1 reset 0: no effect; 1: Reset
13	Reserved	-	-	Reserved
12	SPI1RST	RW	0	SPI1 reset 0: no effect; 1: Reset
11	TIM1RST	RW	0	TIM1 reset

Bit	Name	R/W	Reset Value	Function
				0: no effect; 1: Reset
10:1	Reserved	-	-	Reserved
0	SYSCFGRST	RWs	0	SYSCFG reset 0: no effect; 1: Reset

8.6.13. I/O interface clock enable register (RCC _ IOPENR)

Address offset: 0x34

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.	GPIOFEN	Res.	Res.	Res.	GPIOBEN	GPIOAEN
										RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved

5	GPIOFEN	RW	0	I/O PortF clock enabled.
				0: clock disabled;
				1: clock enabled;

4:2	Reserved	-	-	Reserved

1	GPIOBEN	RW	0	IO port B clock enable
				0: clock disabled;
				1: clock enabled;

0	GPIOAEN	RW	0	IO port A clock enable
				0: clock disabled;
				1: clock enabled;

8.6.14. AHB peripheral clock enable register (RCC _ AHBENR)

Address offset: 0x38

Reset value: 0x0000 0300

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
25	CORDICEN	RW	0	Digital coprocessor module clock enabled. 0: Interrupt is inhibited 1: enable;
24	DIVEN	RW	0	Divider module clock enabled. 0: Interrupt is inhibited 1: enable;
23:13	Reserved	-	-	Reserved
12	CRCEN	RW	0	CRC clock enable 0: Disabled 1: Enabled
11:10	Reserved	-	-	Reserved
9	SRAMEN	RW	1	In Sleep mode, the SRAM's clock enable control 0: The clock is disabled in Sleep mode 1: The clock is enabled in Sleep mode

				Note: This bit only affects the clock enable of this module in Sleep mode, and the clock of this module will not be turned off in run mode
8	FLASHEN	RW	1	In Sleep mode, FLASH's clock enables control 0: The clock is disabled in Sleep mode 1: The clock is enabled in Sleep mode Note: This bit only affects the clock enable of this module in Sleep mode, and the clock of this module will not be turned off in run mode
7:1	Reserved	-	-	Reserved
0	DMAEN	RW	0	DMA clock enable 0: Disabled 1: Enabled

8.6.15. APB peripheral clock enable register 1 (RCC _ APBENR1)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIMEN	OPAEN	Res.	PWREN	DBGEN	Res.	Res.	Res.	Res.	I2C2EN	I2C1EN	~~LPUARTEN~~	Res.	USART3EN	USART2EN	Res.
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.	SPI2EN	Res.	Res.	WWDGEN	RTCAPBEN	Res.	Res.	Res.	Res.	~~TIM7EN~~	~~TIM6EN~~	Res.	Res.	~~TIM2EN~~	TIM2EN
	RW			RW	RW		RW			~~RW~~	~~RW~~			~~RW~~	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMEN	RW	0	Low power timer 1 clock enable 0: Disabled 1: Enabled
30	OPAEN	RW	0	OPA clock enable 0: Disabled 1: Enabled
29	Reserved	-	-	Reserved
28	PWREN	RW	0	Power interface module clock enabled. 0: Disabled 1: Enabled
27	DBGEN	RW	0	DBG enable 0: Disabled 1: Enabled
26:23	Reserved	-	-	Reserved
22	I2C2EN	RW	0	I2C2 enable 0: Interrupt is inhibited 1: enable;
21	I2C1EN	RW	0	I2C1 enable 0: Disabled 1: Enabled
20:19	Reserved	-	-	Reserved
18	USART3EN	RW	0	USART3 enable 0: Interrupt is inhibited 1: enable;
17	USART2EN	RW	0	USART2 enable 0: Disabled 1: Enabled
16:15	Reserved	-	-	Reserved
14	SPI2EN	RW	0	SPI2 enable 0: Disabled 1: Enabled
13:12	Reserved	-	-	Reserved
11	WWDGEN	RW	0	Window WDG module clock enabled. 0: Disabled 1: Enabled

Bit	Name	R/W	Reset Value	Function
				This register is cleared by the hardware system reset.
10	RTCAPBEN	RW	0	RTC module APB clock enabled. 0: Disabled 1: Enabled
9:1	Reserved	-	-	Reserved
0	TIM2EN	RW	0	TIM2 enable 0: Interrupt is inhibited 1: enable;

8.6.16. APB peripheral clock enable register 2 (RCC _ APBENR2)

Address offset: 0x40

Reset value: 0x0000 00001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCDEN	Res.	COMP2EN	COMP1EN	ADCEN	Res.	TIM17EN	TIM16EN	Res.
							RW		RW	RW	RW		RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14EN	USART1EN	Res.	SPI1EN	TIM1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGEN
RW	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	LCDEN	RW	0	LCD clock enable 0: Disabled 1: Enabled
23	Reserved	-	-	Reserved
22	COMP2EN	RW	0	COMP2 enable 0: Disabled 1: Enabled
21	COMP1EN	RW	0	COMP1 enable 0: Disabled 1: Enabled
20	ADCEN	RW	0	ADC enable 0: Disabled 1: Enabled
19	Reserved	-	-	Reserved
18	TIM17EN	RW	0	TIM17 enable 0: Disabled 1: Enabled
17	TIM16EN	RW	0	TIM16 enable 0: Disabled 1: Enabled
16	Reserved	-	-	Reserved
15	TIM14EN	RW	0	TIM14 enable 0: Disabled 1: Enabled
14	USART1EN	RW	0	USART1 enable 0: Disabled 1: Enabled
13	Reserved	-	-	Reserved
12	SPIEN	RW	0	SPI1 enable 0: Disabled 1: Enabled
11	TIM1EN	RW	0	TIM1 enable 0: Disabled 1: Enabled
10:1	Reserved	-	-	Reserved
0	SYSCFGEN	RW	1	SYSCFG clock enable 0: Disabled 1: Enabled

8.6.17. Peripheral independent clock configuration register (RCC _ CCIPR)

Address offset: 0x54

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.	LPTIMSEL[1:0]		Res.	Res.
												RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.		Res.		Res.		COMP2SEL	COMP1SEL	PVDSEL	Res.	Res.	Res.	Res.	Res.	Res.	TIMCLKCTL
						RWs	RW								RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved			
19:18	LPTIMSEL[1:0]	RW	2’b00	LPTIM1 clock source selection 00: PCLK01: LSI10: No clock11: LSE
17:10	Reserved	-	-	Reserved
9	COMP2SEL	RW	0	COMP2 clock source selection 0: PCLK 1: LSC (clock selected by RCC _ BDCR.LSCOSEL) Note: Configure the select LSC clock before enabling FLTEN.
8	COMP1SEL	RW	0	COMP1 clock source selection 0: PCLK 1: LSC (clock selected by RCC _ BDCR.LSCOSEL) Note: Configure this register selection clock before enabling COMP2 _ FR2. FLTEN.
7	PVDSEL	RW	0	PVD detect clock source selection. 0: PCLK 1: LSC (clock selected by RCC _ BDCR.LSCOSEL) Note: Configure this register selection clock before enabling COMP1 _ FR1. FLTEN.
6:1	Reserved	-	-	Reserved
0	TIMCLKCTRL	RW	0	TIMER PCLK Frequency Control. 0: TIMER PCLK is system PCLK * 2, but the frequency will not exceed HCLK; 1: TIMER PCLK is system PCLK * 1;

8.6.18. RCC domain control register (RCC _ BDCR)

Address offset: 0x5C

Reset value: 0x0000 0000, via POR/BOR

This register is only allowed to be written when PWR _ CR1. DBP is 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	LSCOSEL	LSCOEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDRST
						RW	RW								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.	LSECSSD	LSECSSON	Res.		LSEBYP	LSERDY	LSEON
RW						RW			RW	RW			RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	LSCSEL	RW	0	Low speed clock output selection 0: LSI 1: LSE
00:17	Reserved	-	-	Reserved
16	BDRST	RW	0	RTC domain soft reset. 0: no effect 1: Reset
15	RTCEN	RW		RTC clock enabled. 0: Disabled 1: Enabled In addition to POR/BOR, BDRST may also reset this bit.
14:10	Reserved	-	-	Reserved
9:8	RTCSEL[1:0]	RW	0	RTC clock source selection. 00: No clock01: LSE10: LSI11: HSE divided by 128 Once the RTC clock source is selected, it cannot be changed unless: ● RTC is reset to 00 ● Select as LSE (LSECSSD = 1) but no LSE ● BDRST soft reset to 00
7	Reserved	-	-	Reserved
6	LSECSSD	R	0	LSE CSS (clocksecurity system) detection failed. Set by hardware to indicate when a failure has been detected by the Clock Security Systemon the external 32.768 kHz oscillator (LSE). 0: No failure detected on LSE 1: Failure detected on ISE
5	LSECSSON	RW	0	CSS on LSE enable 0: Disabled 1: Enabled LSECSSON must be enabled after the LSE oscillator is enabled (LSEON=1) and ready (LSERDY=1). Once enabled this bit cannot be disabled, except after an LSE failure detection (LSECSSD=1).
4:3	Reserved	-	-	Reserved
2	LSEBYP	RW	0	LSE oscillator bypass 0: Not Bypassed, low-speed external clock selection crystal oscillator 1: Bypassed, low-speed external clock selection external interface input clock Note: This bit can only be written when external 32.768 KHz OSC is disabled (LSEON = 0 and LSERDY = 0).
1	LSERDY	R	0	LSE OSC ready bit. Set and cleared by hardware to indicate that the LSE oscillator is stable. 0: not ready 1: ready
0	LSEON	RW	0	LSE oscillator enable 0: Disabled 1: Enabled

8.6.19. Control/status register (RCC_CSR)

Address offset: 0x60

Reset value: 0x0000 0000

Reset by POR (flag bit), reset by system reset (LSION), reset by system reset excluding NRST (NRST _ FLTIDS)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	WWDG RSTF	IWDG RSTF	SFT RSTF	PWR RSTF	PIN RSTF	OBL RSTF	Res.	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	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.	NRST_FLT-DIS	Res.	Res.	Res.	Res.	Res.	Res.	LSIRDY	LSION
							RW							R	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	WWDGRSTF	R	0	Window WDG reset flag. Cleared by writing to the RMVF bit.
29	IWDGRSTF	R	0	Independent watchdog reset flag Cleared by writing to the RMVF bit.
28	SFTRSTF	R	0	Soft reset flag. Cleared by writing to the RMVF bit.
27	PWRRSTF	R	0	BOR/POR/PDR reset flag Cleared by writing to the RMVF bit.
26	PINRSTF	R	0	PIN reset flag Cleared by writing to the RMVF bit.
25	OBLRSTF	R	0	Option byte loader reset flag Cleared by writing to the RMVF bit.
24	Reserved	-	-	Reserved
23	RMVF	RW	0	After the software is set, the reset flag will be cleared.
8	NRST_FLTDIS	RW	0	NRST filtering disabled 0: HSI _ 10M enabled, and filtering 20us width function enabled 1: Filtering function is disabled and HSI _ 10M remains off
7:2	Reserved	-	-	Reserved
1	LSIRDY	R	0	LSI oscillator stable flag 0: LSI oscillator unstable 1: LSI oscillator stable

9. General-purpose I/Os (GPIO)

9.1. Introduction

Each general-purpose I/O port has:

- Four 32-bit configuration registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR and GPIOx_PUPDR)
- Two 32-bit data registers (GPIOx_IDR and GPIOx_ODR)
- A 32-bit set/reset register (GPIOx_BSRR)
- A 32-bit locking register (GPIOx_LCKR)
- 2 multiplexing function selection registers (GPIOx _ AFRH and GPIOx _ AFRL).

9.2. GPIO main features

- Output status: push-pull or open drain + pull up/pull down
- Output data from output data register (GPIOx_ODR) or peripheral (alternate function output)
- Speed selection for each I/O
- Input status: floating, pull-up/down, analog
- Input data to input data register (GPIOx_IDR) or peripheral (alternate function input)
- Bit set and reset register (GPIOx_BSRR) for bitwise write access to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configurations
- Analog function
- Alternate function selection registers (Max. 16 alternate functions for each IO)
- Fast toggle capable of changing every clock cycles
- Highly flexible pin multiplexing allows the use of I/O pins as GPIOs or as one of several peripheral functions

9.3. GPIO functional description

Each port bit of the general-purpose I/O (GPIO) ports can be individually configured by software in several modes:

- Input float
- Input pull-up
- Input drop-down
- Analog input
- Open drain output with pull-up or pull-down
- push-pull output with pull-up or pull-down
- push-pull with pull-up or pull-down multiplexing function
- – Multiplexed open drain with pull-up or pull-down

Each I/O port can be freely programmed, but the I/0 port registers must be accessed in 32-bit words, half-words, or bytes. The purpose of the GPIOx_BSRR register is to allow atomic read/modify accesses to any of the GPIOx_ODR registers. In this way, there is no danger when an IRQ is generated between read and change access.

The following figure shows the basic structure of an I/O port (1bit):

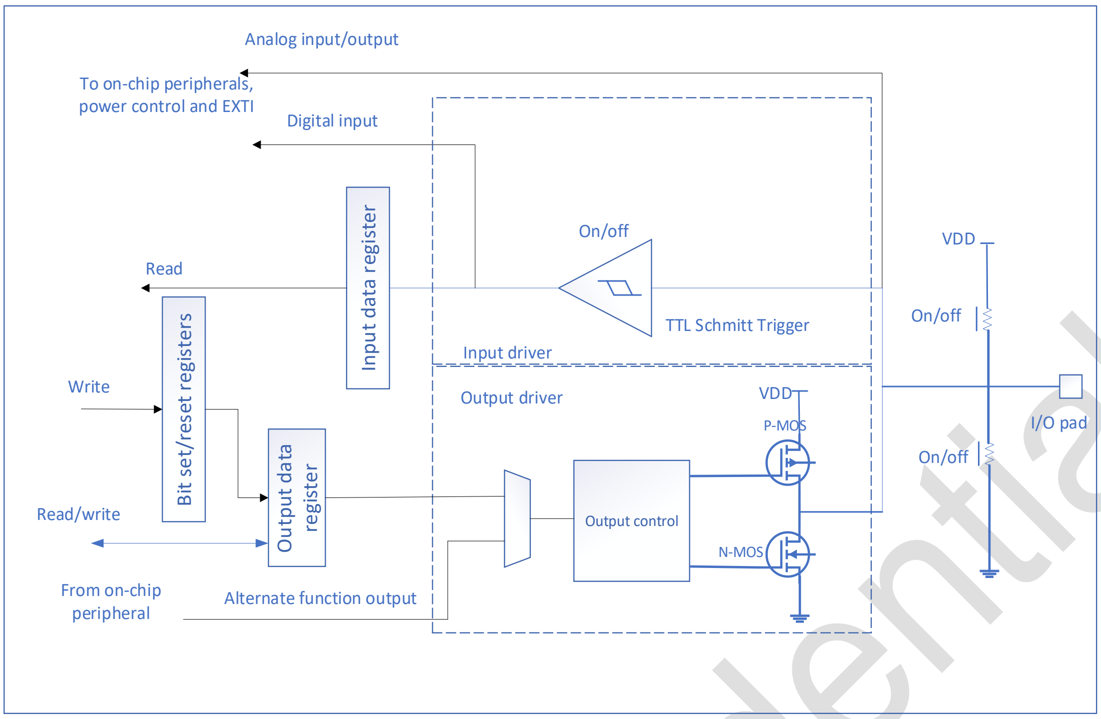


Figure 9-1 Basic structure of an I/O port bit

9.3.1. General-purpose I/Os (GPIO)

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in analog mode. The debug pins are in AF pull-up/pull-down after reset:

– PA14-SWCLK: put in pull-down mode

– PA13-SWDIO: put in pull-up mode

The Boot pin is placed in input mode by default, pull-down mode

– PF7-Boot: put in drop-down mode

When the pin is configured as output, the value written to the output data register (GPIOx_ODR) is output on the I/O pin. It is possible to use push-pull or open-drain mode outputs (low level is output, high level is HI-Z).

The input data register (GPIOx_IDR) captures the data present on the I/O pin at every AHB clock cycle.

All GPIO pins have weak internal pull-up and pull-down resistors, which can be activated ornot depending on the value in the GPIOx_PUPDR register.

9.3.2. I/O pin alternate function multiplexer and mapping

The device I/O pins are connected to on-board peripherals/modules through a multiplexer that allows only one peripheral alternate function (AF) connected to an I/O pin at a time. In this way, there can be no conflict between peripherals available on the same I/O pin.

The multiplexer on each I/O port has up to 16 multiplexing function inputs (AF0 to AF15), which can be configured through the registers GPIOx _ AFRL (for pins 0 to 7) and GPIOx _ AFRH (for pins 8 to 15).

■ After reset the multiplexer selection is alternate function 0 (AF0). The I/Os are configured in alternate function mode through GPIOx_MODER register

- The multiplexing functions of each leg are distributed on the corresponding data hands as explained
- In addition to this flexible I/O multiplexing architecture, each peripheral has alternate functions mapped onto different I/O pins to optimize the number of peripherals available in smaller packages.

To use an I/O in a given configuration, the user has to proceed as follows:

- Debug function: after each device reset these pins are assigned as alternate function pins immediately usable by the debugger host
- GPIO: configure the desired I/O as output, input or analog in the GPIOx_MODER register.
- Peripheral alternate function:
  - The I/O corresponding to the register GPIOx _ AFRL or GPIOx _ AFRH configuration is multiplexing function x (x = 0... 15)
  - Select the type, pull-up/pull-down and output speed via the GPIOx_OTYPER, GPIOx_PUPDR and GPIOx_OSPEEDER registers, respectively.
  - Configure the desired I/O as an alternate function in the GPIOx_MODER register.
- Additional functions:
  - Regardless of any mode the IO port is configured into, the ADC, OPA, functions are enabled in the registers of the ADC and OPA modules. When the IO port is used as ADC or OPA, it is recommended to configure the port to analog mode through the register GPIOx _ MODER;
  - The LCD and COMP functions, in addition to being enabled in the registers of the LCD and COMP modules, also need to configure the ANA2EN register in the SYSCFG module. When the IO port is used for LCD and COMP functions, it is recommended to configure the port to analog mode through the register GPIOx _ MODER;
  - For crystal oscillator additional functions, the respective functions are configured in the corresponding PWR and RCC module registers. These functions have priority over the configuration in the standard GPIO registers.

9.3.3. I/O port control registers

Each of the GPIO ports has four 32-bit memory-mapped control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) to configure up to 16 I/Os. The GPIOx_MODER register is used to select the I/O mode (input, output, AF, analog). The registers GPIOx _ OTYPER and GPIOx _ OSPEEDR are used to select the output type (push-pull or open-drain) and speed. Register GPIOx _ PUPDR is used to select pull-up/pull-down

9.3.4. I/O port data registers

Each GPIO has two 16-bit memory-mapped 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. The input data register (GPIOx _ IDR) is used to store the level status on the I/O port and is read-only.

9.3.5. I/O data bitwise handling

The set/reset register (GPIOx _ BSRR) is a 32-bit register that can bit and reset the individual positions of the output data register (GPIOx _ ODR). The bit set reset register has twice the size of GPIOx_ODR.

To each bit in GPIOx_ODR, correspond two control bits in GPIOx_BSRR: BS(i) and BR(i). When written to 1, bit BS(i) sets the corresponding ODR(i) bit. When written to 1, bit BR(i) resets the ODR(i) corresponding bit.

Writing any bit to 0 in GPIOx_BSRR does not have any effect on the corresponding bit inGPIOx_ODR. If there is an attempt to both set and reset a bit in GPIOx_BSRR, the set action takes priority.

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_ODRat bit level: it is possible to modify one or more bits in a single atomic AHB write access.

9.3.6. GPIO locking mechanism

Register GPIOx _ LCKR is a control register that freezes IO through a series of special write timings, including GPIOx _ MODER, GPIOx _ OTYPER, GPIOx _ OSPEEDR, GPIOx _ PUPDR, GPIOx _ AFRL, and GPIOx _ AFRH.

To write the GPIOx_LCKR register, a specific write / read sequence has to be applied. When the right LOCK sequence is applied to bit 16 in this register, the value of LCKR [15:0] is used to lock the configuration of the I/Os (during the write sequence the LCKR [15:0] value must be the same). When a LOCK procedure is executed on a port bit, the configuration of the port bit can no longer be changed until the next MCU or peripheral reset. Each bit of GPIOx _ LCKR freezes a bit corresponding to the control register (GPIOx _ MODER, GPIOx _ OTYPER, GPIOx _ OSPEEDR, GPIOx _ PUPDR, GPIOx _ AFRL, and GPIOx _ AFRH).

The LOCK sequence can only be performed using a word (32-bit long) access to the GPIOx_LCKR register due to the fact that GPIOx_LCKR bit 16 has to be set at the same time as the [15:0] bits.

9.3.7. I/O alternate function input/output

Two registers are provided to select one of the alternate function inputs/outputs available for each I/O. With these registers, the user can connect an alternate function to some other pin as required by the application.

Many possible peripheral functions can be multiplexed at each GPIO port using the registers GPIOx _ AFRL and GPIOx _ AFRH, so the application lets each I/O select one of the functions. The AF selection signal being common to the alternate function input and alternate function output, a single channel is selected for the alternate function input/output of a given I/O.

9.3.8. External interrupt/wakeup lines

All ports have external interrupt capability. In order to use the external interrupt line, the port must be disabled to be configured in analog mode or crystal pin, and the input enable needs to be triggered.

9.3.9. Input configuration

When the I/O port is programmed 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
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state

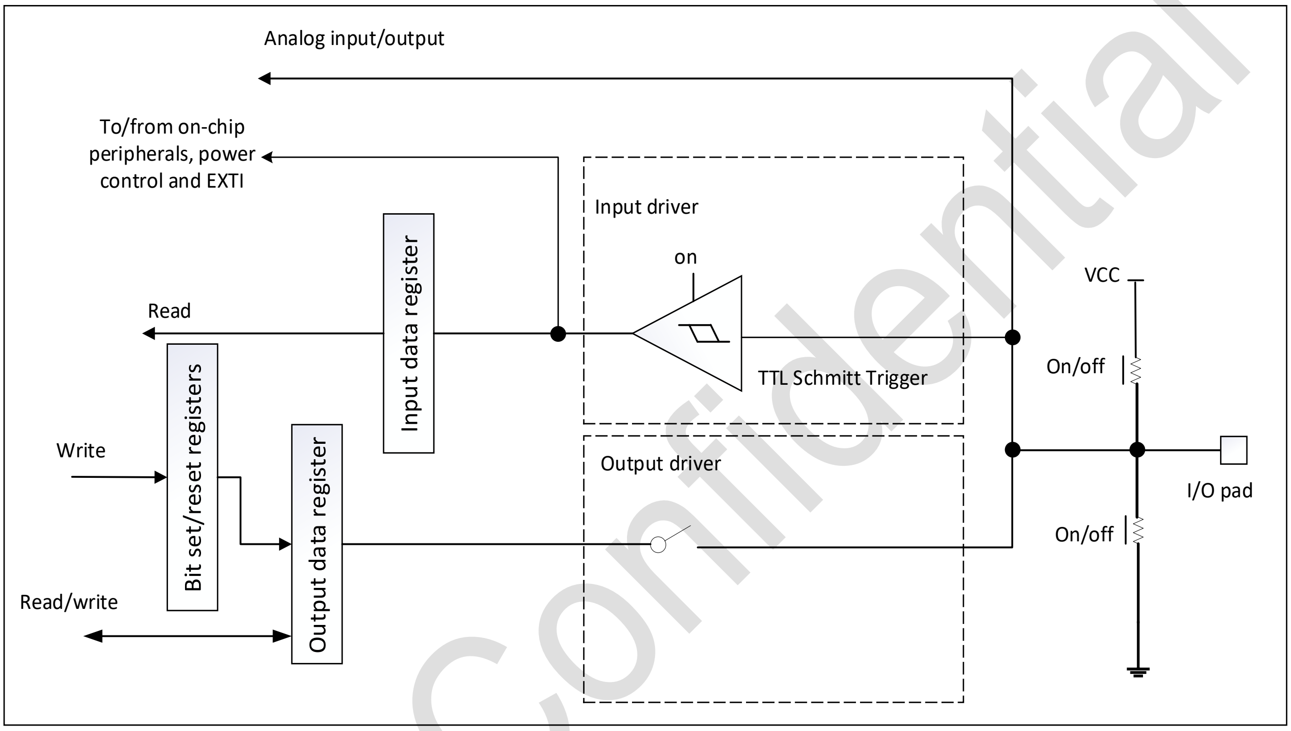


Figure 9-2 Input float/pull-up/pull-down configuration

9.3.10. Output configuration

When the I/O port is programmed as output:

- The output buffer is disabled
  - — Open drain mode: '0' on the output register activates the N-MOS, while '1' on the output register puts the port in a high impedance state (PMOS is never activated).
  - — Push-pull mode: A “0” in the Output register activates the N-MOS whereas a “1” in the Output register activates the P-MOS.
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state
- A read access to the output data register gets the last written value

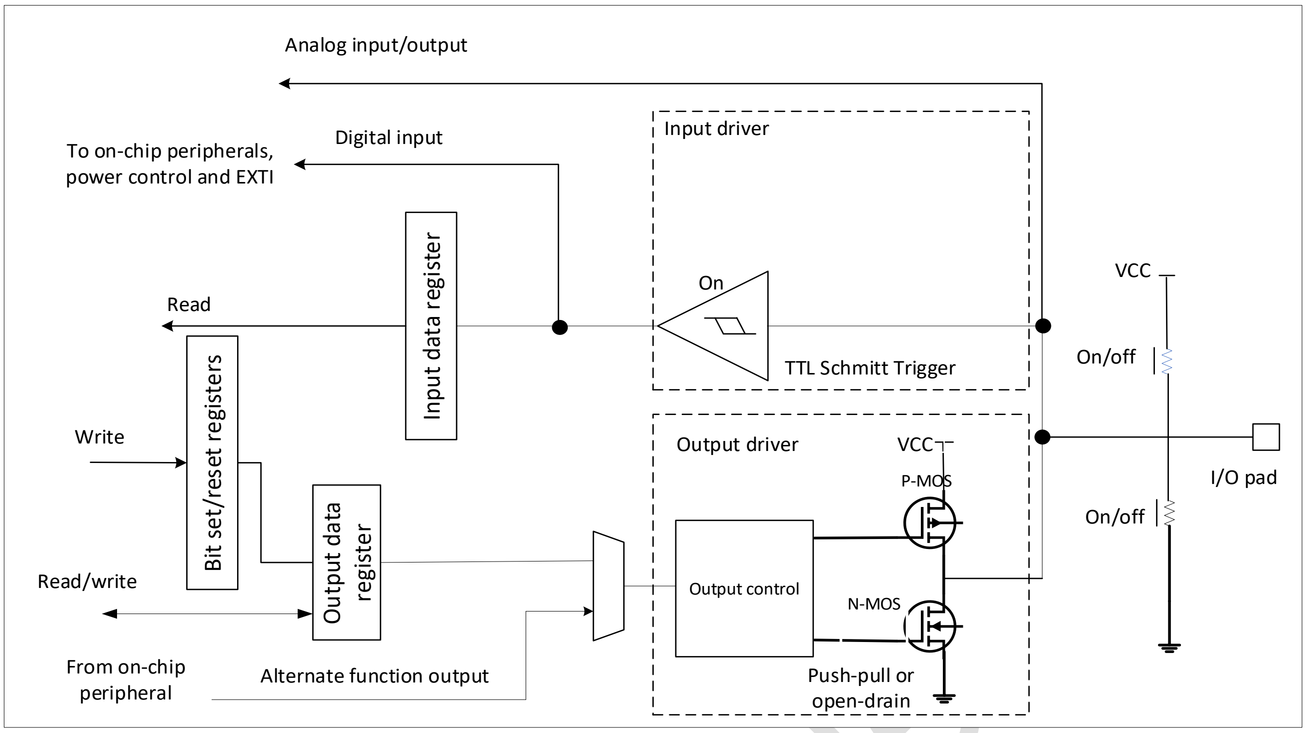


Figure 9-3 Output configuration

9.3.11. Alternate function configuration

When the I/O port is programmed as alternate function:

- The output buffer can be configured in open-drain or push-pull mode
- The output buffer is driven by the signals coming from the internal peripheral (alternate function output)
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state

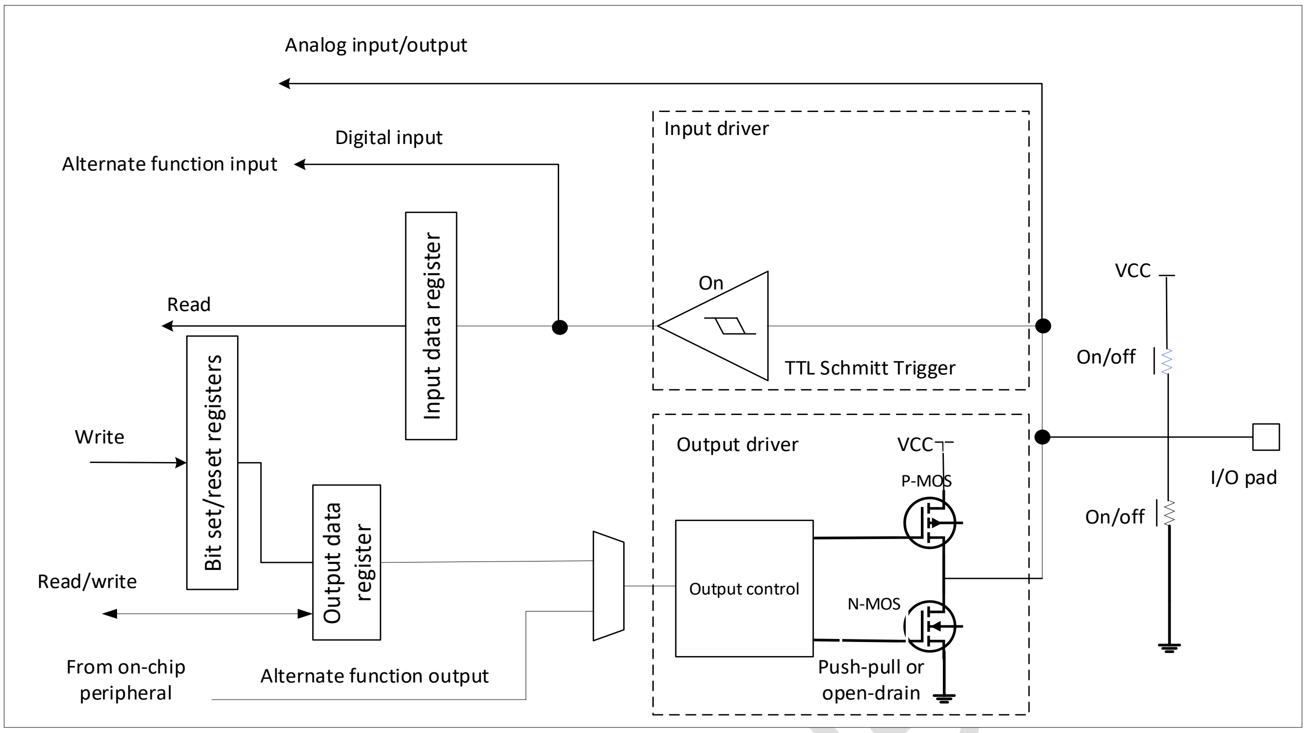


Figure 9-4 Alternate function configuration

9.3.12. Analog configuration

When the I/O port is programmed as analog configuration:

- The output buffer is disabled;
- The Schmitt trigger input is deactivated, providing zero consumption for every analog value of the I/O pin. The Schmidt trigger output value is forcibly set to '0';
- Weak pull-up and pull-down resistors are disabled (software setting is required);
- The value is' 0 'when the input data register is read.

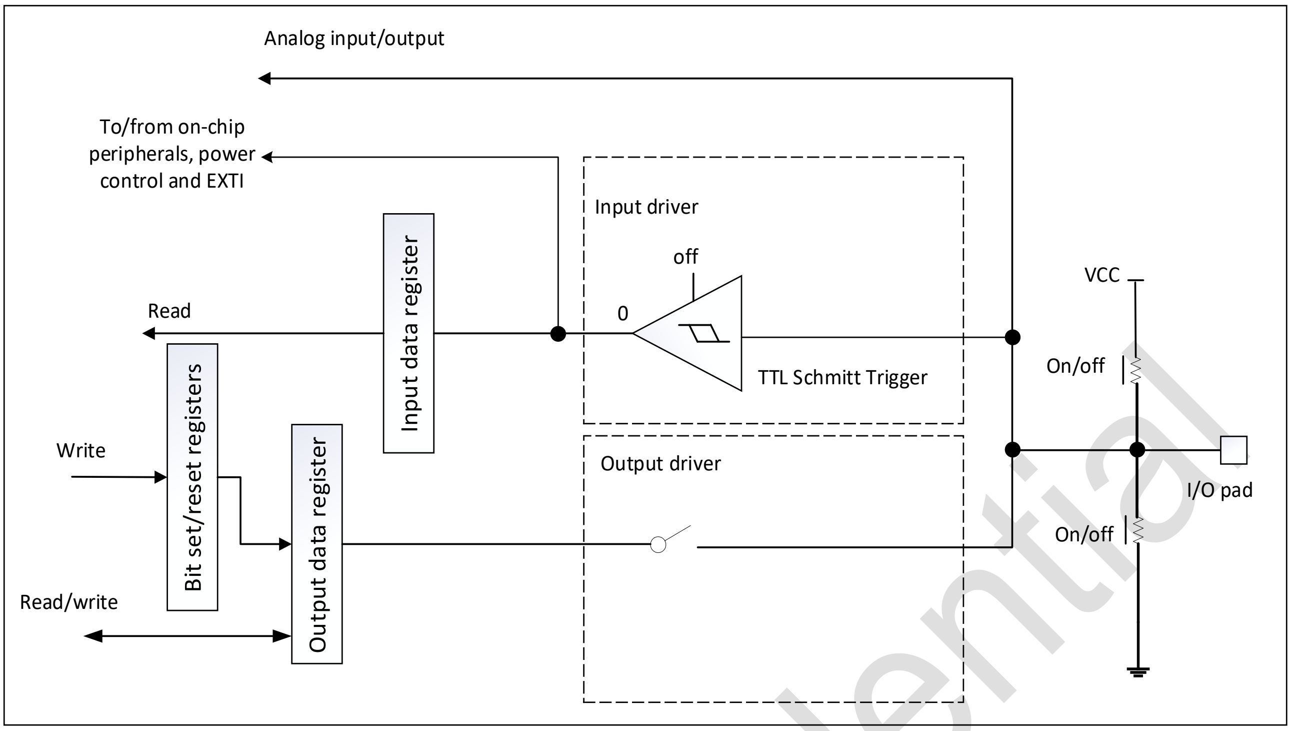


Figure 9-5 High impedance-analog configuration

9.3.13. Use LSE or HSE pins as GPIO

When the HSE or LSE function is turned off (the default after reset), the corresponding pin can be used as a normal GPIO.

When the HSE or LSE function is turned on (HSEON or LSEON is set in the RCC _ CSR register), the software needs to configure the corresponding port as an analog port.

When the oscillator is configured in a user external clock mode, only the pin OSC_IN or OSC32_IN is reserved for clock input and the OSC_OUT or OSC32_OUT pin can still be used as normal GPIO.

9.4. GPIO registers

The peripheral registers can be written in word, half word or byte mode.

9.4.1. GPIO port mode register (GPIOx _ MODER) (x = A, B, F)

Address offset: 0x00

Reset value:

a) 0xEBFF FFFF for GPIOA

b) 0xFFFF FFFF for GPIOB

c) 0x00FF 3FFF For GPIOF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15 [1:0]		MODE14 [1:0]		MODE13 [1:0]		MODE12 [1:0]		MODE11 [1:0]		MODE10 [1:0]		MODE9 [1:0]		MODE8 [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
MODE7 [1:0]		MODE6 [1:0]		MODE5 [1:0]		MODE4 [1:0]		MODE3 [1:0]		MODE2 [1:0]		MODE1 [1:0]		MODE0 [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		y = 15..0

				These bits are written by software to configure the I/O mode. 00: Input mode 01: General purpose output mode 10: Alternate function mode 11: reset state

9.4.2. GPIO port output type register (GPIOx _ OTYPER) (x = A, B, 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.															
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	-	-	Reserved
15:0	MODE[1:0]	RW		These bits are written by software to configure the I/O output type. 0: Output push-pull (reset state) 1: Output open-drain

9.4.3. GPIO port output speed register (GPIOx _ OSPEEDR) (x = A, B, F)

Address offset: 0x08

Reset value: 0x0C00 0000 (for port A)

Reset value: 0x0000 0000 (for other ports)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15		OSPEED14		OSPEED13		OSPEED12		OSPEED11		OSPEED10		OSPEED9		OSPEED8	
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		OSPEED6		OSPEED5		OSPEED4		OSPEED3		OSPEED2		OSPEED1		OSPEED0	
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		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, B, F)

Address offset: 0x0C

Reset value:

0x2400 0000(for port A)

0x0000 0000 (for port B)

0x0000 8000 (for port F)

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		Y = 15.. 0 These bits are written by software to configure the I/O pull-up or pull-down 00: No pull-up, pull-down 01: Pull-up 10: Pull-down 11: Reserved11: Reserved

9.4.5. GPIO port input data register (GPIOx _ IDR) (x = A, B, F)

Address offset: 0x10

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
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	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	-	-	Reserved
15:0	Idy	R		y = 15..0 This is read-only, and the read value bit corresponds to the state of the I/O port

9.4.6. GPIO port output data register (GPIOx _ ODR) (x = A, B, F)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
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 [1: 0]	RW		y = 15..0 These bits can be read and written by software. Description: For GPIOx _ BSRR or GPIOx _ BRR registers. (x = A, B, F), each ODR bit can be set/cleared independently.

9.4.7. GPIO port bit set/reset register (GPIOx _ BSRR) (x = A, B, 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		y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: does not affect the corresponding ODRy bit 1: Clear the corresponding ODRy bit Note: If the corresponding bits of both Bsy and Bry are set, the Bsy bit works
15:0	BSy	W		y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: does not affect the corresponding ODRy bit 1: Set the corresponding ODRy bit

9.4.8. GPIO port configuration lock register (GPIOx_LCKR) (x = A, B, F)

This register is used to lock the configuration of the port bits when a correct write sequence is applied to bit 16 (LCKK). The value of bits [15:0] is used to lock the configuration of the GPIO. During the write sequence, the value of LCKR [15:0] must not change. When the LOCK sequence has been applied on a port bit, the value of this port bit can no longer be modified until the next system reset.

Note: A specific write sequence is used to write to the GPIOx_LCKR register. Only word access (32-bit long) is allowed during this locking sequence.

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		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 write sequence: The write timing of the lock key: write 1-> write 0-> write 1-> read 0-> read 1. The last read can be omitted, but it can be used to confirm that the lock key has been activated. Note: During the LOCK key write sequence, the value of LCK[15:0] must not change. Any error in the lock sequence aborts the lock. After the first lock sequence on any bit of the port, any read access on the LCKK bit returns ‘1’ until the next MCU reset or peripheral reset.
15:0	LCKy	RW		y = 15..0 These bits are readable and writable but can only be written when the LCKK bit is 0. 0: Port configuration not locked 1: Port configuration locked

9.4.9. GPIO alternate function low register (GPIOx_AFRL) (x = A, B, 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] ((y = 7 to 0))	RW		These bits are written by software to configure alter-nate function I/Os. AFSELy selection: 0000: AF01000: AF80001: AF11001: AF9 0010: AF21010: AF10 0011: AF31011: AF11 0100: AF41100: AF12 0101: AF51101: AF13 0110: AF61110: AF14 0111: AF71111: AF15

9.4.10. GPIO alternate function high register (GPIOx_AFRH) (x = A, B, F)

Address offset: 0x28

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] ((y = 8 to 15))	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 0000: AF01000: AF80001: AF11001: AF9 0010: AF21010: AF10 0011: AF31011: AF11 0100: AF41100: AF12 0101: AF51101: AF13 0110: AF61110: AF14 0111: AF71111: AF15

9.4.11. GPIO port bit reset register (GPIOx _ BRR) (x = A, B, 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	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	-	-	Reserved
15:0	Bry	RW		y = 15..0 These bits are write-only. A read to these bits re-turns the value 0. 0: does not affect the corresponding Ody bit 1: Clear the corresponding Ody bit

10. System configuration controller (SYSCFG)

The F002B-C Series devices feature a set of configuration registers. The main purposes of the system configuration controller are the following:

- IO pin interface control
- DMA peripheral channel selection control
- Remap the memory located at the beginning of the code interval (Boot)
- Manage TIMERs ETR or brake inputs

10.1. SYSCFG registers

10.1.1. SYSCFG configuration register 1 (SYSCFG_CFGR1)

This register serves as a specific configuration for memory and DMA request remap and control of special IO functions.

Two bits are used to configure the type of memory accessible at address 0x0000 0000. These two bits are used to select the physical remap of the software and bypass the hardware BOOT selection. After reset these bits take the value selected by the actual boot mode configuration.

Address offset: 0x00

Reset value: 0x0000 000x (x is the memory mode selected by the actual boot mode configuration)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.							GPIO _ _ AHB _ SEL	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
Res.	Res.	ETR_SRC_TIM2 [1:0]		Res.	Res.	ETR_SRC_TIM1 [1:0]		Res.	Res.	TIM2_IC4_SRC		TIM1_IC1_SRC		Mem _ MODE [1:0]	
		RW	RW			RW	RW			RW RW		RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	RW	-	Readable and writable

24	GPIO _ AHB _ SEL	RW	0	CPU FASTIO or CPU AHB bus access GPIO register control.
				0: CPU FASTIO bus accesses GPIO register;
				1: CPU AHB bus accesses GPIO register;
				Note: DMA can access the GPIO register via the AHB bus by default and is not controlled by this bit

23:14	Reserved	RES	-	Reserved

13:12	ETR_SRC_TIM2 [1:0]	RW	2'b00	TIMER2 ETR input source selection.
				2 'b00: ETR is derived from GPIO;
				2 'b01: ETR derived from COMP1 output;
				2 'b10: ETR derived from COMP2 output;
				2 'b11: ETR is derived from ADC analog watchdog output;

11:10	Reserved	RES	-	Reserved

Bit	Name	R/W	Reset Value	Function

9:8	ETR_SRC_TIM1 [1:0]	RW	3'b00	TIMER1 ETR input source selection.
				2 'b00: ETR derived from GPIO;
				2 'b01: ETR derived from COMP1 output;
				2 'b10: ETR derived from COMP2 output;
				2 'b11: ETR is derived from ADC analog watchdog output;

7:6	Reserved	RES	-	Reserved
5:4	TIM2_IC4_SRC	RW	2'b00	TIM2 CH4 input source 00: from TIM2 _ CH4 10; 01: from comp1 output; 10: from comp2 output 11: reserved
3:2	TIM1_IC1_SRC	RW	2' b00	TIM1 CH1 Input Source 00: from TIM1 _ CH1 10; 01: from comp1 output; 10: from comp2 output 11: reserved
1:0	Mem _ MODE [1:0]			Memory mapping select bit Set and cleared by software. They control mapping of the 0x0000 0000 address of the memory. After reset (including power-on reset and pin reset only), these bits resample the values defined in the Boot pin and option byte, taking the actual boot mode configuration values. (Sampling time is the release edge of reset) X0: Main flash, mapped at 0x0000 0000 01: System flash, mapped at 0x0000 0000 11: SRAM, mapped at 0x0000 0000

10.1.2. SYSCFG configuration register 2 (SYSCFG_CFGR2)

Address offset: 0x04

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**
Res	Res	Res	COMP2_Ocref_CLR_TIM2	COMP2_Oc-ref_CLR_TIM1	COMP1_Oc-ref_CLR_TIM2	COMP1_Oc-ref_CLR_TIM1	COMP2_BRK_TIM17	COMP1_BRK_TIM17	COMP2_BRK_TIM16	COMP1_BRK_TIM16	COMP2_BRK_TIM1	COMP1_BRK_TIM1	PVD_LOCK	Res	LOCKUP_LOCK
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	-
12	COMP2_Ocref_CLR_TIM2	RW	1' b0	1: COMP2 output as TIM2 ocref _ clr input; 0: COMP2 output is not input as TIM2 ocref _ clr
11	COMP2_Ocref_CLR_TIM1	RW	1' b0	1: COMP2 output as TIM1 ocref _ clr input; 0: COMP2 output is not input as TIM1 ocref _ clr
10	COMP1_Ocref_CLR_TIM2	RW	1' b0	1: COMP1 output as TIM2 ocref _ clr input; 0: COMP1 output is not input as TIM2 ocref _ clr
9	COMP1_Ocref_CLR_TIM1	RW	1' b0	1: COMP1 output as TIM1 ocref _ clr input; 0: COMP1 output is not input as TIM1 ocref _ clr
8	COMP2_BRK _ TIM17	RW	0	COMP2 is enabled as a TIMx break input. 0: COMP2 output is not used as TIM17 break input 1: COMP2 output as TIM17 break input
7	COMP1_BRK	RW	0	COMP1 is enabled as a TIMx break input.

Bit	Name	R/W	Reset Value	Function
	_ TIM17			0: COMP1 output is not used as TIM17 break input 1: COMP1 output as TIM17 break input
6	COMP2_BRK _ TIM16	RW	0	COMP2 is enabled as a TIMx break input. 0: COMP2 output is not used as TIM16 break input 1: COMP2 output as TIM16 break input
5	COMP1_BRK _ TIM16	RW	0	COMP1 is enabled as a TIMx break input. 0: COMP1 output is not used as TIM16 break input 1: COMP1 output as TIM16 break input
4	COMP2_BRK _TIM1	RW	0	COMP2 is enabled as a TIMx break input. 0: COMP2 output is not used as TIM1 break input 1: COMP2 output as TIM1 break input
3	COMP1 _ BRK _ TIM1	RW	0	COMP1 is enabled as a TIMx break input. 0: COMP1 output is not used as TIM1 break input 1: COMP1 output as TIM1 break input
2	PVD _ LOCK	RW	0	PVD Lock Enable bit Software set, system reset and clear. It can be used as a brake input to enable and lock the PVD connection to TIM1/TIM16/TIM17, and also lock the PVDE and PLS bits of the PWR _ CR register. 0: The PVD interrupt is not connected to the brake input of TIM1/TIM16/TIM17. The PVDE and FLS bits can be written by the application. 1: PVD interrupts the brake input connection to TIM1/TIM16/TIM17. PVDE and FLS bits are read-only.
1	Reserved	-	-	-
0	LOCKUP_ LOCK	RW		Enable Cortex-M0+ LOCKUP Software set, system reset and clear. It can enable and lock the LOCKUP (hardfault) output of Cortex-M0 + to the brake input of TIM1/TIM16/TIM17. 0: The LOCKUP output of Cortex-M0 + is not connected to the brake input of TIM1/TIM16/TIM17 1: LOCKUP output of Cortex-M0 + is connected to brake input of TIM1/TIM16/TIM17

10.1.3. SYSCFG configuration register 3 (SYSCFG_CFGR3)

Address offset: 0x08

Reset value: 0x003F _ 3F3F

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.	DMA3_MAP					
											RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	DMA2_MAP						Res.	Res.	DMA1_MAP					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:16	DMA3_MAP	RW	6’b111111	000000: ADC 000001: SPI1 _ TX 000010: SPI1 _ RX 000011: SPI2 _ TX 000100: SPI2 _ RX 000101: USART1 _ TX 000110: USART1 _ RX 000111: USART2 _ TX 001000: USART2 _ RX 001001: I2C1 _ TX

Bit	Name	R/W	Reset Value	Function
				001010: I2C1 _ RX 001011: TIM1 _ CH1 001100: TIM1 _ CH2 001101: TIM1 _ CH3 001110: TIM1 _ CH4 001111: TIM1 _ COM; 010000: TIM1 _ UP 010001: TIM1 _ TRIG 010010: TIM2 _ CH1 010011: TIM2 _ CH3 010100: TIM2 _ CH4 010101: TIM2 _ TRG 010110: TIM2 _ UP 010111: TIM2 _ CH2 011000: TIM16 _ CH1 011001: TIM16 _ UP 011010: TIM17 _ CH1 011011: TIM17 _ UP 011100: USART3 _ TX 011101: USART3 _ RX 011110: I2C2 _ TX 011111: I2C2 _ RX 100000: LCD Others: reserve
15:14	Reserved	-	-	Reserved
13:8	DMA2_MAP	RW	6'b111111	000000: ADC 000001: SPI1 _ TX 000010: SPI1 _ RX 000011: SPI2 _ TX 000100: SPI2 _ RX 000101: USART1 _ TX 000110: USART1 _ RX 000111: USART2 _ TX 001000: USART2 _ RX 001001: I2C1 _ TX 001010: I2C1 _ RX 001011: TIM1 _ CH1 001100: TIM1 _ CH2 001101: TIM1 _ CH3 001110: TIM1 _ CH4 001111: TIM1 _ COM; 010000: TIM1 _ UP 010001: TIM1 _ TRIG 010010: TIM2 _ CH1 010011: TIM2 _ CH3 010100: TIM2 _ CH4 010101: TIM2 _ TRG 010110: TIM2 _ UP 010111: TIM2 _ CH2 011000: TIM16 _ CH1 011001: TIM16 _ UP 011010: TIM17 _ CH1 011011: TIM17 _ UP 011100: USART3 _ TX 011101: USART3 _ RX

Bit	Name	R/W	Reset Value	Function
				011110: I2C2 _ TX 011111: I2C2 _ RX 100000: LCD Others: reserve
7:6	Reserved	-	-	Reserved
5:0	DMA1_MAP	RW	6'b111111	000000: ADC 000001: SPI1 _ TX 000010: SPI1 _ RX 000011: SPI2 _ TX 000100: SPI2 _ RX 000101: USART1 _ TX 000110: USART1 _ RX 000111: USART2 _ TX 001000: USART2 _ RX 001001: I2C1 _ TX 001010: I2C1 _ RX 001011: TIM1 _ CH1 001100: TIM1 _ CH2 001101: TIM1 _ CH3 001110: TIM1 _ CH4 001111: TIM1 _ COM; 010000: TIM1 _ UP 010001: TIM1 _ TRIG 010010: TIM2 _ CH1 010011: TIM2 _ CH3 010100: TIM2 _ CH4 010101: TIM2 _ TRG 010110: TIM2 _ UP 010111: TIM2 _ CH2 011000: TIM16 _ CH1 011001: TIM16 _ UP 011010: TIM17 _ CH1 011011: TIM17 _ UP 011100: USART3 _ TX 011101: USART3 _ RX 011110: I2C2 _ TX 011111: I2C2 _ RX 100000: LCD Others: reserve

10.1.4. GPIOA filter enable register (PA _ ENS)

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
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[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed noise filter enabled

10.1.5. GPIOB filter enable register (PB _ ENS)

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
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[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed noise filter enabled

10.1.6. GPIOF filter enable register (PF _ ENS)

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.	PF _ ENS [11: 0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:12	Reserved			
11:0	PF _ ENS [x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed noise filter enabled

10.1.7. I2C type IO configuration register (SYSCFG _ IOCFG)

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.	PF _ PU _ IIC [1: 0]		PF _ EIIC [3: 0]				Res.	Res.	PB _ EHS [5: 0]					
		rw		rw						rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PA _ EHS [1: 0]		PB _ EIIC [4: 0]					PA _ EIIC [7: 0]							
rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:30	Reserved			
29:28	PF _ PU _ IIC	RW	0	I2C _ PU type IO 4.7 K pull-up resistor control enabled. Bit0: Control PF5 Bit1: Control PF6
27:24	PF _ EIIC [3: 0]	RW	0	PF EIIC Signal Control. Used as _ I2C type IO. Bit0: Control PF0

Bit	Name	R/W	Reset Value	Function
				Bit1: Control PF1 Bit2: Control PF5 Bit3: Control PF6
23:22	Reserved			
21:16	PB _ EHS [5: 0]	RW	0	PB EHS signal control. Used as 80mA LED IO control. Bit0: Control PB2 Bit1: Control PB3 Bit2: Control PB4 Bit3: Control PB5 Bit4: Control PB6 Bit5: Control PB7
15	Reserved			
14:13	PA _ EHS [1: 0]	RW	0	PA EHS Signal Control. Used as 80mA LED IO control. Bit0: Control PA0 Bit1: Control PA15
12:8	PB _ EIIC [4: 0]	RW	0	PB IIC Signal Control. Used as _ I2C type IO. Bit0: Control PB6 Bit1: Control PB7 Bit2: Control PB8 Bit3: Control PB10 Bit4: Control PB11
7:0	PA _ EIIC [7: 0]	RW	0	PA EIIC Signal Control. Used as _ I2C type IO. Bit0: Control PA2 Bit1: Control PA3 Bit2: Control PA7 Bit3: Control PA8 Bit4: Control PA9 Bit5: Control PA10 Bit6: Control PA11 Bit7: Control PA12

10.1.8. GPIOA analog 2 enable register (PA _ ANA2EN)

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
PA_ANA2EN [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 _ ANA2EN [x]	RW	0x0000	IO PORTA PAD _ ANA2 enable 0: PAD _ ANA2 disable 1: PAD _ ANA2 enable Note 1: This register needs to be enabled when IO is configured in analog mode. If IO is configured in non-analog mode, this register needs to be configured to 0. Note 2: Mapped to COMP1 and COMP2, the PAD of the LCD module must be configured with the corresponding bit of this register as 1.

10.1.9. GPIOB analog 2 enable register (PB _ ANA2EN)

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

PB_ANA2EN [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	-	-	Reserved
15:0	PB _ ANA2EN [x]	RW	0x0000	IO PORTB PAD _ ANA2 enable 0: PAD _ ANA2 disable 1: PAD _ ANA2 enable Note 1: This register needs to be enabled when IO is configured in analog mode. If IO is configured in non-analog mode, this register needs to be configured to 0. Note 2: Mapped to COMP1 and COMP2, the PAD of the LCD module must be configured with the corresponding bit of this register as 1.

10.1.10. GPIOF analog 2 enable register (PF _ ANA2EN)

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.	PF_ANA2EN [11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	PF _ ANA2EN [x]	RW	0x000	IO PORTF PAD _ ANA2 enable 0: PAD _ ANA2 disable 1: PAD _ ANA2 enable Note 1: This register needs to be enabled when IO is configured in analog mode. If IO is configured in non-analog mode, this register needs to be configured to 0. Note 2: Mapped to COMP1 and COMP2, the PAD of the LCD module must be configured with the corresponding bit of this register as 1.

11. DMA

11.1. Introduction

Direct memory access (DMA) is used to provide a high-speed data transfer between peripherals and memory as well as from memory to memory. Without CPU intervention, data can be moved quickly through DMA, saving CPU resources for other operations.

The DMA controller have 3 channels in total, each one dedicated to manage memory access requests from one or more peripherals. The DMA controller includes an arbiter to handle DMA requests, which manages the priority of each DMA request.

11.2. Main features

- 3 configurable channels
- Each channel is associated either with a DMA request signal coming from a peripheral, or with a software trigger in memory-to-memory transfers. These functions are configured by software
- On the same DMA module, the priority among multiple requests can be set by software programming (there are four levels: very high, high, medium and low). When the priority setting is equal, it is determined by hardware (request 1 takes precedence over request 2, and so on)
- The transfer sizes of the source and destination are independent (byte, half word and word), simulating packing and unpacking. Source and destination addresses must be aligned to the transfer width.
- Loop-enabled buffer management
- Each channel has 3 event flags (DMA Half Transfer, DMA Transfer Complete, and DMA Transfer Error), which are logically OR as a separate interrupt request
- Memory and inter-memory transfer
- Data transmission of peripherals and memory, memory and peripherals, peripherals and peripherals
- FLASH, SRAM, APB, and AHB peripherals can all be accessed as sources and destinations
- Programmable transmission quantity 0 ~ 65535

11.3. DMA functional description

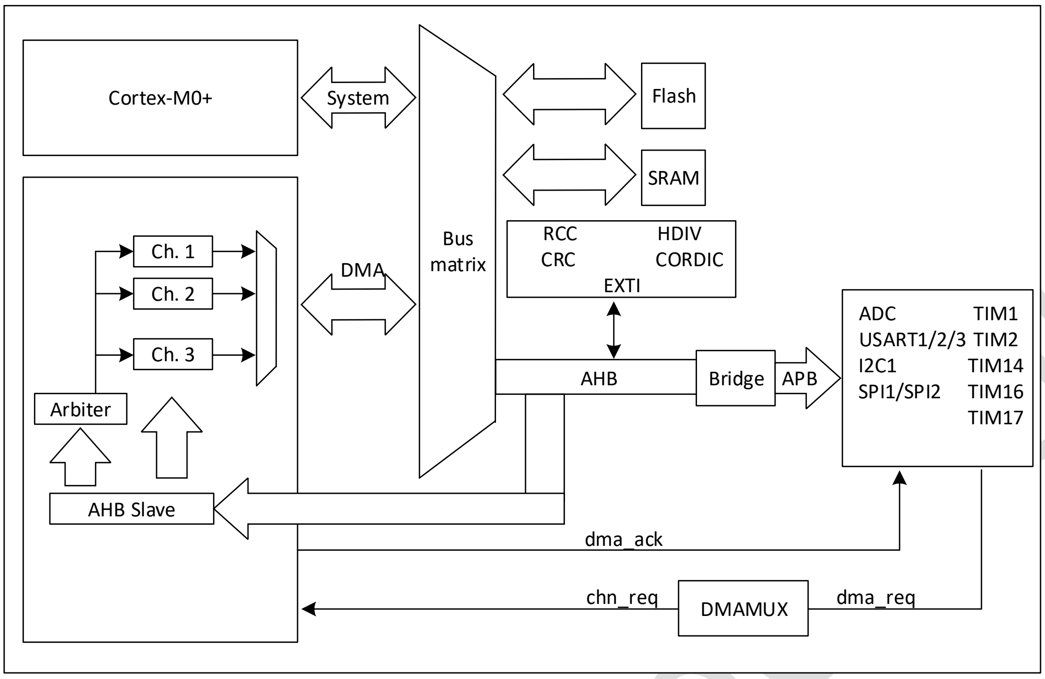


Figure 11-1 DMA Block Diagram

11.3.1. DMA transmission

After completing an event, the peripheral sends a request signal to the DMA controller. The DMA controller processes the request according to the priority of the channel. When the DMA controller starts to access the requesting peripheral, the DMA controller sends it an answer signal after the transmission is completed. When a response signal is received from the DMA controller, the peripheral immediately releases its request. Once the peripheral releases the request, the DMA controller revokes the acknowledgement signal. If there are more requests, the peripheral can initiate the next transfer.

In summary, each DMA transfer consists of three operations:

■ From the peripheral data register or from the current peripheral/memory address register, the start address of the first transfer is the peripheral base address or memory cell specified by the DMA _ CPARx or DMA _ CMARx register.

■ The data is stored to the peripheral register or the memory address indicated by the current peripheral/memory address register, and the start address at the first transfer is the peripheral and address or memory cell specified by the DMA _ CPARx or DMA _ CMARx register.

■ Performs a decrement of the DMA _ CNDTRx register, which indicates the number of outstanding operations.

11.3.2. Arbiter

The arbiter initiates access to the peripheral/memory according to the priority of the channel request. Priority management is divided into two stages:

■ Software: The priority of each channel can be set in the DMA _ CCRx register, with 4 levels

— Highest priority

  - High priority
  - Medium priority
  - Low priority
- Hardware: If 2 requests have the same software priority, the lower numbered channel has higher priority than the numbered channel. For example, channel 2 takes precedence over channel 4.

11.3.3. DMA channel

Each channel can perform DMA transfers between peripheral registers with fixed addresses and memory addresses. The amount of data transferred by DMA is programmable, up to 65535. A register containing the amount of data to be transferred, decremented after each transfer.

Programmable data sizes

The amount of transfer of peripherals and memory can be programmed by the PSIZE and MSIZE bits in the DMA _ CCRx register.

Pointer increment

By setting the PINC and MINC flag bits in the DMA _ CCRx register, the pointers of peripherals and memory can selectively complete auto-increment after each transfer. When the bit increment mode is set, the next address to be transferred will be the previous address plus the increment value depending on the selected data width bit 1, 2, or 4.

The first address transferred is the address stored in the DMA _ CPARx/DMA _ CPARx register. During transfer, these registers hold their initial values and the software cannot change and read out the address currently being transferred (it is in the internal current peripheral/memory address register).

When the channel is configured in acyclic mode, 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 _ CNDTRx register with the DMA channel closed.

In loop mode, at the end of the last transfer, the contents of the DMA _ CNDTRx register are automatically reloaded to its initial value, and the internal current peripheral/memory address register is also reloaded to the initial base address set by the DMA _ CPARx/DMA _ CMARx register.

Circular mode

Circular mode is used to handle circular buffers and continuous data transfers (such as the scan mode of ADC). The CIRC bit in the DMA _ CCRx register is used to turn on this function. When the cyclic mode is started and the number of data transmissions becomes 0, the initial value set when the channel is configured will be automatically restored, and the DMA operation will continue.

Memory to memory mode

The operation of the DMA channel can be carried out without peripheral requests. This operation is memory-to-memory mode.

When the MEM2MEM bit in the DMA _ CCRx register is set, the DMA transfer will begin as soon as the software sets the EN bit in the DMA _ CCRx register to start the DMA channel. When the DMA _ CNDTRx register becomes 0, the DMA transfer ends. The memory-to-memory mode cannot be used simultaneously with the loop mode.

Channel configuration procedure

Configure the DMA channel as follows:

- Set the address of the peripheral register in the DMA _ CPARx register. When a peripheral data transfer request occurs, this address will be the source or destination of the data transfer.
- Set the address of the data memory in the DMA _ CMARx register. When a peripheral data transfer request occurs, the transferred data will be read from or written to this address.
- Set the amount of data to be transferred in the DMA _ CNDTRx register. After each data transfer, this value is decremented.
- The priority of the channel is set in the PL [1: 0] bit of the DMA _ CCRx register.
- In the DMA _ CCRx register, the direction of data transfer, the cyclic mode, the incremental mode of the peripheral and memory, the data width of the peripheral and memory, the interrupt generated by half of the transfer or the interrupt generated by completion of the transfer are set.
- Set the ENABLE bit of the DMA _ CCRx register to start the channel.

Once a DMA channel is initiated, a DMA request to a peripheral connected on the channel can be responded.

When half of the data is transmitted, the Half Transmission Flag (HTIF) is set to 1. When the Allow Half Transmission Interrupt bit (HTIE) is set, an interrupt request will be generated. After the data transmission is completed, the Transmission Complete Flag (TCIF) is set to 1, and when the Allow Transmission Complete Interrupt bit (TCIE) is set, an interrupt request will be generated.

11.3.4. Programmable data width, data alignment, and internal code

When the memory data width MSIZE and the peripheral data width PSIZE are different, the DMA aligns the data according to the following table:

Table 11-1 data width and size ends (PINC = MINC = 1)

Source breadth	Destination breadth	transmission Number	Source: Address/Data	Transmission operations	Target: Address/Data
8	8	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: Read B0 [7: 0] at 0x0, write B0 [7: 0] at 0x0 2: Read B1 [7: 0] at 0x1, write B1 [7: 0] at 0x1 3: Read B2 [7: 0] at 0x2, write B2 [7: 0] at 0x2 4: Read B3 [7: 0] at 0x3, write B3 [7: 0] at 0x3	0x0/B0 0x1/B1 0x2/B2 0x3/B3
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: Read B0 [7: 0] at 0x0, write 00B0 [7: 0] at 0x0 2: Read B1 [7: 0] at 0x1, write 00B1 [7: 0] at 0x2 3: Read B2 [7: 0] at 0x2, write 00B2 [7: 0] at 0x4 4: Read B3 [7: 0] at 0x3, write 00B3 [7: 0] at 0x6	0x0/00B0 0x2/00B1 0x4/00B2 0x6/00B3
8	32	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: Read B0 [7: 0] at 0x0, write 000000B0 [31: 0] at 0x0 2: Read B1 [7: 0] at 0x1, write 000000B1 [31: 0] at 0x4 3: Read B2 [7: 0] at 0x2, write 000000B2 [31: 0] at 0x8	0x0/000000B0 0x4/000000B1 0x8/000000B2 0xC/000000B3

Source breadth	Destination breadth	transmission Number	Source: Address/Data	Transmission operations	Target: Address/Data
				4: Read B3 [7: 0] at 0x3, write 000000B3 [31: 0] at 0xC	
16	8	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [15: 0] at 0x0, write B0 [7: 0] at 0x0 2: Read B3B2 [15: 0] at 0x2, write B2 [7: 0] at 0x1 3: Read B5B4 [15: 0] at 0x4, write B4 [7: 0] at 0x2 4: Read B7B6 [15: 0] at 0x6, write B6 [7: 0] at 0x3	0x0/B0 0x1/B2 0x2/B4 0x3/B6
16	16	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [15: 0] at 0x0, write B1B0 [15: 0] at 0x0 2: Read B3B2 [15: 0] at 0x2, write B3B2 [15: 0] at 0x2 3: Read B5B4 [15: 0] at 0x4, write B5B4 [15: 0] at 0x4 4: Read B7B6 [15: 0] at 0x6, write B7B6 [15: 0] at 0x6	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6
16	32	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: Read B1B0 [7: 0] at 0x0, write 0000B1B0 [31: 0] at 0x0 2: Read B3B2 [7: 0] at 0x2, write 0000B3B2 [31: 0] at 0x4 3: Read B5B4 [7: 0] at 0x4, write 0000B5B4 [31: 0] at 0x8 4: Read B7B6 [7: 0] at 0x6, write 0000B7B6 [31: 0] at 0xC	0x0/0000B1B0 0x4/000B3B2 0x8/0000B5B4 0xC/0000B7B6
32	8	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: Read B3B2B1B0 [31: 0] at 0x0, write B0 [7: 0] at 0x0 2: Read B7B6B5B4 [31: 0] at 0x4, write B4 [7: 0] at 0x1 3: Read BBBAB9B8 [31: 0] at 0x8, write B8 [7: 0] at 0x2 4: Read BFBEBDBC [31: 0] at 0xc, write BC [7: 0] at 0x3	0x0/B0 0x1/B4 0x2/B8 0x3/BC
32	16	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: Read B3B2B1B0 [31: 0] at 0x0, write B1B0 [7: 0] at 0x0 2: Read B7B6B5B4 [31: 0] at 0x4, write B5B4 [7: 0] at 0x2 3: Read BBBAB9B8 [31: 0] at 0x8, write B9B8 [7: 0] at 0x4 4: Read BFBEBDBC [31: 0] at 0xc, write BDBC [7: 0] at 0x6	0x0/B1B0 0x2/B5B4 0x4/B9B8 0x6/BDBC
32	32	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: Read B3B2B1B0 [31: 0] at 0x0, write B3B2B1B0 [7: 0] at 0x0 2: Read B7B6B5B4 [31: 0] at 0x4, write B7B6B5B4 [7: 0] at 0x2 3: Read BBBAB9B8 [31: 0] at 0x8, write BBBAB9B8 [7: 0] at 0x4 4: Read BFBEBDBC [31: 0] at 0xc, write BFBEBDBC [7: 0] at 0x6	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC

Access not support byte/halfword slave

If DMA writes in bytes or half words to an AHB device that does not support byte or half word writing operation (i.e. HSIZE is not suitable for this module), no error occurs, DMA will write 32-bit HWDATA data according to the following two examples:

- When HSIZE = halfword, write halfword '0xABCD' and DMA will set the HWDATA bus to '0xABCDABCD'.

- When HSIZE = byte, write byte '0xAB' and DMA will set the HWDATA bus to '0xABABABAB'.

Assuming that the AHB/APB bridge is a 32-bit slave device of an AHB, it does not process the HSIZE parameter, it will transfer any byte or half-word on the AHB to the APB in 32 bits as follows:

- A write byte data '0xB0' operation on an AHB to address 0x0 (or 0x1, 0x2, or 0x3) will be translated to a write data '0xB0B0B0B0' operation on an APB to address 0x0.
- A write half-word data '0xB1B0' operation on an AHB to address 0x0 (or 0x2) will be translated to a write data '0xB1B0B1B0' operation on an APB to address 0x0.

11.3.5. Error Management

Reading and writing a reserved address area will produce a DMA transfer error. During DMA read and write operations, when a DMA transfer error occurs, the hardware automatically clears the EN bit of the channel configuration register (DMA _ CCRx) corresponding to the channel where the error occurred, and the channel operation is stopped. At this time, the Transfer Error Interrupt Flag bit (TEIF) corresponding to the channel in the DMA _ IFR register will be set, and an interrupt will be generated if the Transfer Error Interrupt Allow bit is set in the DMA _ CCRx register.

11.3.6. DMA interrupts

Each DMA channel can generate interrupts when DMA transmission is halfway through, transmission completion, and transmission errors. For application flexibility, these interrupts are opened by setting different bits of the register.

Table 11-2 DMA Interrupt Request

Interrupt event	Event flag bit	Enable control bit
Transmission more than half	HTIF	HTIE
Transmission complete	TCIF	TCIE
Transmission error	TEIF	TEIE

When the DMA _ CNDTRx register is 1, the HTIFx bit will not be set, and the TCIFx bit will be set when the transfer is completed.

11.3.7. DMA peripheral request mapping

Table 11-3 DMA requests per channel

Peripherals	**Channel 1**	**Channel 2**	**Channel 3**
ADC	ADC	ADC	ADC
SPI	SPI _ RX SPI _ TX	SPI _ RX SPI _ TX	SPI _ RX SPI _ TX
USART	USART1_RX USART1_TX USART2_RX USART2_TX USART3_RX USART3_TX	USART1_RX USART1_TX USART2_RX USART2_TX USART3_RX USART3_TX	USART1_RX USART1_TX USART2_RX USART2_TX USART3_RX USART3_TX
I2C1	I2C1_RX I2C1_TX	I2C1_RX I2C1_TX	I2C1_RX I2C1_TX
I2C2	I2C2_RX I2C2_TX	I2C2_RX I2C2_TX	I2C2_RX I2C2_TX
TIM1	TIM1_CH1 TIM1_CH2 TIM1_CH3 TIM1_CH4	TIM1_CH1 TIM1_CH2 TIM1_CH3 TIM1_CH4	TIM1_CH1 TIM1_CH2 TIM1_CH3 TIM1_CH4

Peripherals	Channel 1	Channel 2	Channel 3
	TIM1_UP TIM1_TRIG TIM1_COM	TIM1_UP TIM1_TRIG TIM1_COM	TIM1_UP TIM1_TRIG TIM1_COM
TIM2	TIM2_CH1 TIM2_CH2 TIM2_CH3 TIM2_CH4 TIM2_UP TIM2_TRIG	TIM2_CH1 TIM2_CH2 TIM2_CH3 TIM2_CH4 TIM2_UP TIM2_TRIG	TIM2_CH1 TIM2_CH2 TIM2_CH3 TIM2_CH4 TIM2_UP TIM2_TRIG
TIM16	TIM16_CH1 TIM16_UP	TIM16_CH1 TIM16_UP	TIM16_CH1 TIM16_UP
TIM17	TIM17_CH1 TIM17_UP	TIM17_CH1 TIM17_UP	TIM17_CH1 TIM17_UP
LCD	LCD	LCD	LCD

11.4. DMA registers

11.4.1. DMA interrupt status register (DMA_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.	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.	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
				R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31：12	Reserved	-	-	Reserved

11	TEIF3	R	0	Channel 3 transmission error flag.
				The hardware is set, and the software writes DMA _ IFCR = 1 to clear.
				0: No transmission error (TE);
				1: Channel 3 transmission error (TE);

10	HTIF3	R	0	Channel 3 Half Transmission Flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No half-transmission event; 1: Half transmission event occurs on channel 3;

9	TCIF3	R	0	Channel 3 Transmission Complete Flag.
				0: No transmission complete (TC);
				1: Channel 3 transmission complete (TC);

8	GIF3	R	0	Channel 3 global interrupt flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No TE/HT/TC event; 1: TE/HT/TC event occurs in channel 3;

7	TEIF2	R	0	Channel 2 transmission error flag.
				The hardware is set, and the software writes DMA _ IFCR = 1 to clear.
				0: No transmission error (TE);
				1: Channel 2 transmission error (TE);

6	HTIF2	R	0	Channel 2 Half Transmission Flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No half-transmission event; 1: Half transmission event occurs on channel 2;
5	TCIF2	R	0	Channel 2 Transmission Complete Flag.

Bit	Name	R/W	Reset Value	Function

				The hardware is set, and the software writes DMA _ IFCR = 1 to clear.
				0: No transmission complete (TC);
				1: Channel 2 transmission complete (TC);

4	GIF2	R	0	Channel 2 global interrupt flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No TE/HT/TC event; 1: TE/HT/TC event occurs on channel 2;

3	TEIF1	R	0	Channel 1 transmission error flag.
				The hardware is set, and the software writes DMA _ IFCR = 1 to clear.
				0: No transmission error (TE);
				1: Channel 1 transmission error (TE);

2	HTIF1	R	0	Channel 1 Half Transmission Flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No half-transmission event; 1: A half-transmission event occurs on channel 1;

1	TCIF1	R	0	Channel 1 Transmission Complete Flag.
				The hardware is set, and the software writes DMA _ IFCR = 1 to clear.
				0: No transmission complete (TC);
				1: Channel 1 transmission complete (TC);

0	GIF1	R	0	Channel 1 global interrupt flag. The hardware is set, and the software writes DMA _ IFCR = 1 to clear. 0: No TE/HT/TC event; 1: TE/HT/TC event occurs in channel 1;

11.4.2. DMA interrupt flag clear register (DMA_IFCR)

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.	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
				W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31：12	Reserved	-	-	Reserved
11	CTEIF3	W	0	The channel 3 transmission error flag is cleared. 0：No effect； 1: TEIF3 cleared;
10	CHTIF3	W	0	Channel 3 half transmission flag cleared. 0：No effect； 1: Zero HTIF3;
9	CTCIF3	W	0	The channel 3 transmission completion flag is cleared. 0：No effect； 1: Zero TCIF3;
8	CGIF3	W	0	Channel 3 global interrupt flag cleared. 0：No effect； 1: Clear GIF/TEIF/HTIF/TCIF of channel 3;
7	CTEIF2	W	0	The channel 2 transmission error flag is cleared. 0：No effect； 1: TEIF2 is cleared
6	CHTIF2	W	0	Channel 2 half transmission flag cleared. 0：No effect； 1: Zero HTIF2;

Bit	Name	R/W	Reset Value	Function
5	CTCIF2	W	0	The channel 2 transmission completion flag is cleared. 0：No effect； 1: Zero TCIF2;
4	CGIF2	W	0	Channel 2 global interrupt flag cleared. 0：No effect； 1: Clear GIF/TEIF/HTIF/TCIF of channel 2;
3	CTEIF1	W	0	The channel 1 transmission error flag is cleared. 0：No effect； 1: TEIF1 is cleared
2	CHTIF1	W	0	Channel 1 half transmission flag cleared. 0：No effect； 1: Zero HTIF1;
1	CTCIF1	W	0	The channel 1 transmission completion flag is cleared. 0：No effect； 1: Zero TCIF1;
0	CGIF1	W	0	Channel 1 global interrupt flag cleared. 0：No effect； 1: Zero GIF/TEIF/HTIF/TCIF of channel 1;

11.4.3. DMA channel 1 configuration register (DMA _ CCRx)

Address offset: 0x08 + 0x14 * (x-1) (x = 1 ~ 3)

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.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	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	-	-	Reserved
14	MEM2MEM	RW	0	Channel 1 memory to memory mode. 0: Interrupt is inhibited 1: memory to memory mode enabled;
13：12	PL[1:0]	RW	0	Channel 1 priority configuration. 00: low 01: medium; 10: high 11: Very high;

11：10	MSIZE[1:0]	RW	0	Channel 1 memory data width.
				00: 8 bits;
				01: 16 bits;
				10: 32 bits;
				11: Reserved.

9：8	PSIZE[1:0]	RW	0	Channel 1 peripheral data width.
				00: 8 bits;
				01: 16 bits;
				10: 32 bits;
				11: Reserved.

7	MINC	RW	0	Channel 1 memory address increment mode. 0: Interrupt is inhibited 1: Memory address increment mode enabled;
6	PINC	RW	0	Channel 1 peripheral address increment mode. 0: Interrupt is inhibited

Bit	Name	R/W	Reset Value	Function
				1: Peripheral address increment mode enabled;
5	CIRC	RW	0	Channel 1 Loop Mode. 0: Interrupt is inhibited 1: Loop mode enabled;
4	DIR	RW	0	Channel 1 Data Transmission Direction. 0: read from peripheral; 1: read from memory
3	TEIE	RW	0	Channel 1 Transmission Error Interrupt (TE) enabled. 0: Interrupt is inhibited 1: TE interrupt enable;
2	HTIE	RW	0	Channel 1 Half Transmission Interrupt (HT) enabled. 0: Interrupt is inhibited 1: HT interrupt enable;
1	TCIE	RW	0	Channel 1 Transfer Completion Interrupt (TC) enabled. 0: Interrupt is inhibited 1: TC interrupt enable;
0	EN	RW	0	Channel 1 enabled. 0: Interrupt is inhibited 1: channel 1 enabled;

11.4.4. DMA channel 1 data transfer quantity register (DMA _ CNDTRx)

Address offset: 0x0C + 0x14 * (x-1) (x = 1 ~ 3)

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**
NDT[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	-	-	Reserved
15：0	NDT[15:0]	RW	0	Channel 1 data transmission quantity. The number of data transmissions is 0 ~ 65535. This register is written only when the channel is not working (DMA _ CCR1.EN = 0). This register is read-only when the channel is enabled, indicating the number of bytes remaining to be transferred. This register value is decremented after each DMA transfer. After the data transmission is completed, the contents of the register may become 0; When this channel is configured in loop mode, the contents of the registers are automatically reloaded to the values they were previously configured. When the value of this register is 0, no data is transferred even if the DMA channel starts.

11.4.5. DMA channel 1 peripheral address register (DMA _ CPAR1)

Address offset: 0x10 + 0x14 * (x-1) (x = 1 ~ 3)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PA[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
PA[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	PA[31:0]	RW	0	Channel 1 peripheral address. The base address of the channel 1 peripheral data register as the source or destination of data transmission. When PSIZE = 2 'b01, the PA [0] bit is not used. The operation is automatically aligned with the half-word address. When PSIZE = 2 'b10, the PA [1: 0] bit is not used. The operation is automatically aligned with the word address.

11.4.6. DMA channel 1 memory address register (DMA _ CMARx)

Address offset: 0x14 + 0x14 * (x-1) (x = 1 ~ 3)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MA[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**
MA[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	MA[31:0]	RW	0	Channel 1 memory address. Channel 1 memory address, as the source or destination of data transfer. When MSIZE = 2 'b01, the MA [0] bit is not used. The operation is automatically aligned with the half-word address. When MSIZE = 2 'b10, the MA [1: 0] bit is not used. The operation is automatically aligned with the word address.

12. Interrupts and events

12.1. Nested vectored interrupt controller (NVIC)

12.1.1. Main features

- Support 32 maskable external interrupts (not including the sixteen CPU interrupt lines)
- 4 programmable priority levels (2 bits of interrupt priority are used)
- Low latency exception and interrupt handling
- Power management control
- Implementation of system control registers

The NVIC and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts. Including CPU exception, all interrupts are managed by NVIC.

12.1.2. SysTick calibration value register

The SysTick calibration value is set to 6000, and the SysTick clock is set to 6MHz (max fHCLK/8), giving a reference time base of 1ms.

Before entering Sleep or Stop low power consumption, you need to turn off the interrupt of the systick.

12.1.3. Interrupt and exception vectors

Position	Priority	Types of Priority	Acronym	Description	Address
-	-	-	-	Reserved	0x0000_0000
-	-3	fixed	Reset	Reset	0x0000_0004
-	-2	fixed	NMI_Handler	NMI. RCC Clock Security System (CSS) corresponds to the NMI vector	0x0000_0008
-	-1	fixed	HardFualt_Handler	All class of fault	0x0000_000C
-	3	settable	SVCall	System service via SWI Instruction	0x0000_002C
-	5	settable	PendSV	Pendable request for system service	0x0000_0038
	6		SysTick	System tick timer	0x0000_003C
0	7	settable	WWDG	Window watch dog	0x0000_0040
1	8	settable	PVD	Power voltage detector interrupt (EXTI line 16)	0x0000_0044
2	9	Settable	RTC	RTC interrupts (combined EXTI lines 19)	0x0000_0048
3	10	Settable	Flash	Flash global interrupt	0x0000_004C
4	11	Settable	RCC	RCC global interrupt	0x0000_0050
5	12	Settable	EXTI0_1	EXTI line [1: 0] interrupt	0x0000_0054
6	13	Settable	EXTI2_3	EXTI line [3: 2] interrupt	0x0000_0058
7	14	Settable	EXTI4_15	EXTI line [15: 4] interrupt	0x0000_005C
8	15	Settable	LCD	LCD global interrupt	0x0000_0060

Position	Priority	Types of Priority	Acronym	Description	Address
9	16	Settable	DMA_Channel1	DMA channel 1 interrupt	0x0000_0064
10	17	Settable	DMA_Channel 2_3	DMA channel 2 & 3 interrupt	0x0000_0068
11	18	-	Reserved	Reserved	0x0000_006C
12	19	Settable	ADC_COMP	ADC and COMP interrupts (COMP combined with EXTI 17 & 18)	0x0000_0070
13	20	Settable	TIM1_BRK_UP_TRG_COM	TIM1 break, update, trigger and commutation interrupts	0x0000_0074
14	21	Settable	TIM1_CC	TIM1 Capture Compare interrupt	0x0000_0078
15	22	Settable	TIM2	TIM2 global interrupt	0x0000_007C
16	23	-	Reserved	Reserved	0x0000_0080
17	24	Settable	LPTIM	LPTIM interrupt	0x0000_0084
18	25	-	Reserved	Reserved for TIM7	0x0000_0088
19	26	Settable	TIM14	TIM14 global interrupt	0x0000_008C
20	27	-	Reserved	Reserved for TIM15	0x0000_0090
21	28	Settable	TIM16	TIM16 global interrupt	0x0000_0094
22	29	Settable	TIM17	TIM17 global interrupt	0x0000_0098
23	30	Settable	I2C1	I2C1 global interrupt	0x0000_009C
24	31	Settable	I2C2	I2C2 global interrupt	0x0000_00A0
25	32	Settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	Settable	SPI2	SPI2 global interrupt	0x0000_00A8
27	34	Settable	USART1	USART1 global interrupt	0x0000_00AC
28	35	Settable	USART2	USART2 global interrupt	0x0000_00B0
29	36	Settable	USART3	USART3 global interrupt	0x0000_00B4
30	37	Settable	SQRT	Square Root interupt	0x0000_00B8
31	38	Settable	CORDIC	CORDIC global interrupt	0x0000_00BC

1. The grayed cells (the address less than 0x0000 0040) responded to the Cortex ®-M0 + interrupts.

12.2. Extended interrupts and events controller (EXTI)

The extended interrupt and event controller (EXTI) manages the CPU and system wake-up through configurable and direct event inputs (Lines). It outputs following signals:

- Interrupt request, sent to int _ ctrl module to generate CPU IRQ
- Event request, event input to CPU (RXEV)
- The wake-up request is sent to the power consumption management control module

The EXTI wake-up requests allow the system to be woken up from Stop modes. The interrupt request and event request generation can also be used in Run mode.

EXTI allows the management of up to 21 configurable/direct event lines (19 configurable event lines and 2 direct event lines).

12.2.1. EXTI main features

- The system can wake up via GPIO and specified module (COMP/LPTIM) input events
- Configurable events (from I/Os, peripherals not having an associated interrupt pending status bit, or peripherals generating a pulse)
  - Selectable active trigger edge
  - Interrupt pending status bits
  - Individual interrupt and event generation mask
  - SW trigger possibility
- Direct events (peripherals with associated flags and interrupt pending status bits)
  - Fixed rising edge active trigger
  - There is no interrupt pending bit in EXTI module
  - Individual interrupt and event generation mask
  - No SW trigger possibility
- I/O port selector

12.2.2. EXTI block diagram

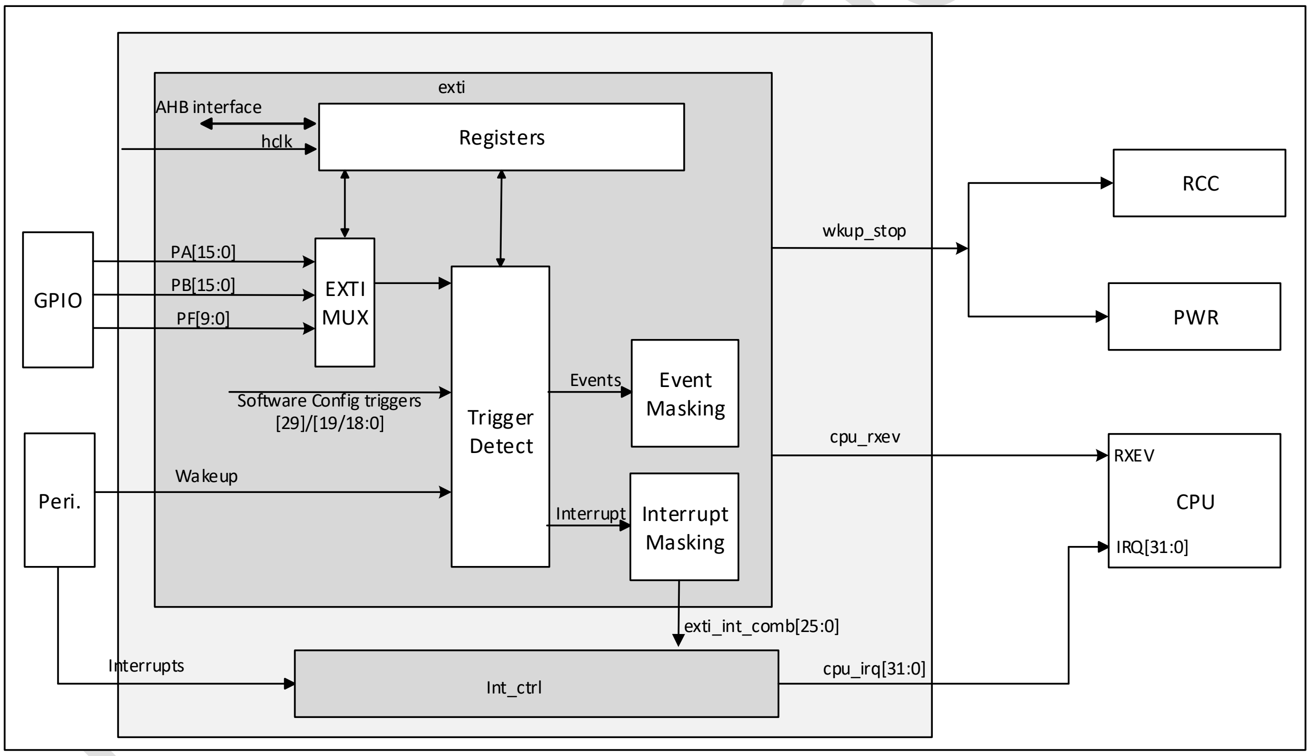


Figure 12-1 EXTI block diagram

12.2.3. EXTI connections between peripherals and CPU

Peripherals that can generate wake-up or interrupt event signals in Stop mode are connected to the EXTI module.

- Generating a pulse, or wake-up signal (a peripheral wake-up signal without an interrupt status bit inside the peripheral) is connected to the configurable line of the EXTI module. At this time, the EXTI module generates an interrupt suspend bit (this bit needs to be cleared), and the EXTI interrupt will serve as an interrupt signal for the CPU.

- The interrupt and wake-up signal of the peripheral with the associated status bit (which is cleared at the peripheral) is connected to the wake-up trigger signal line of the EXTI module. At this time, in addition to the peripheral module, the EXTI module also has a status suspend flag bit and generates an interrupt to the CPU.
- All GPIO ports are input to the EXTI MUX module, and through the configuration of configurable, it is allowed to be selected as a system wake-up signal.

12.2.4. EXTI configurable event trigger wake-up

By configuring the EXTI _ SWIER1 register, the software can trigger the wake-up function.

There is a corresponding register configuration to trigger a rising edge or falling edge or double edge to trigger a configurable type event. The hardware detects a configurable type event input signal according to the configuration to generate a corresponding wake-up event or interrupt signal.

The CPU has its dedicated interrupt mask register and a dedicated event mask registers. An event generated to the CPU after an event is enabled. The unique event input signal rxev is output to the CPU after all events' OR 'operation to the CPU.

Configurable type events have a unique interrupt suspend request register, which is shared with the CPU. The suspend register is set only if the CPU interrupt mask register (EXTI _ IMR) is configured to be unmasked. Each configurable type event will correspond to an external CPU interrupt signal (some will be multiplexed to the same external CPU interrupt signal). Configurable type event interrupts require CPU acknowledgement through the EXTI _ PR register (write 1 clear).

Note: When a bit of the interrupt pending register (EXTI _ PR) remains active (not cleared), the system cannot enter low power mode.

12.2.5. EXTI direct event input wake-up

direct type events will generate interrupts in the EXTI module and will generate event signals that wake up the system and CPU subsystems. When the CPU handles the interrupt generated by this type of trigger event, it needs to clear the interrupt status bit of the peripheral module.

12.2.6. EXTI multiplexer

The GPIO is connected to 16 external interrupt/event lines in the following manner:

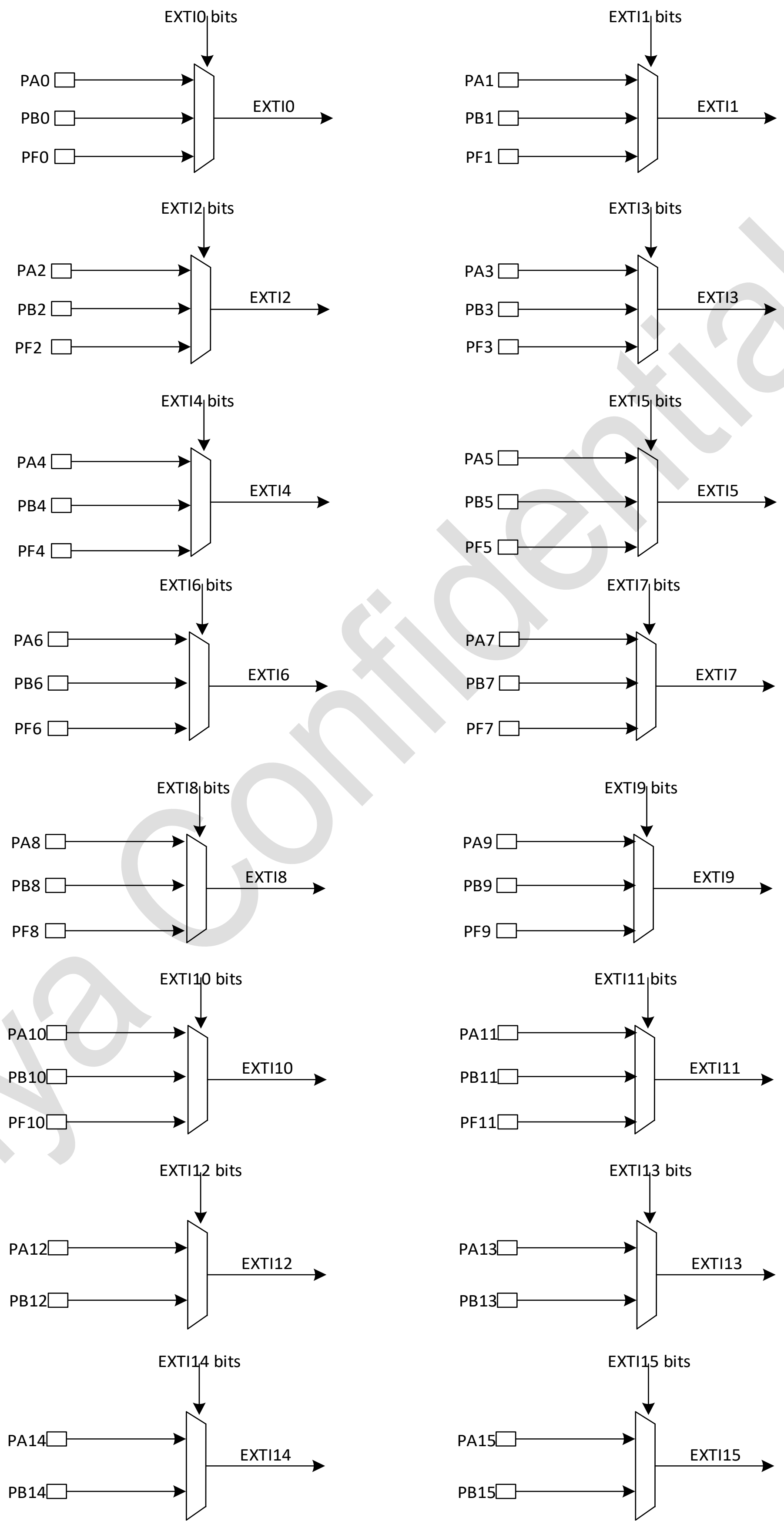


Figure 12-2 External interrupt/event GPIO mapping

The EXTI lines are connected as shown as follows:

EXTI line	Line source	Line type
Line 0-15	GPIO	configurable
Line 16	PVD output	Configurable
Line 0-7	COMP 1 output	Configurable
Line 18	COMP2 output	Configurable
Line 19	RTC	Direct
Line 20	Reserved	-
Line 21	Reserved	-
Line 22	Reserved	-
Line 23	Reserved	-
Line 24	Reserved	-
Line 25	Reserved	-
Line 26	Reserved	-
Line 27	Reserved	-
Line 28	Reserved	-
Line 29	LPTIM	Direct

12.3. EXTI register

The peripheral registers have to be accessed by word (32-bit), half-word (16 bits), and byte (8 bits).

12.3.1. EXTI rising trigger selection register (EXTI_RTSR)

Address offset: 0x00

Reset value: 0x0000 0000

Contains only register bits for configurable events.

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	RT18	RT17	RT16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved			
18	RT18	RW	0	Rising trigger event configuration bit of line18 0: Disabled 1: Enabled
17	RT17	RW	0	Rising trigger event configuration bit of line17 0: Disabled 1: Enabled
16	RT16	RW	0	Rising trigger event configuration bit of line16 0: Disabled 1: Enabled
15	RT15	RW	0	Rising trigger event configuration bit of line15 0: Disabled 1: Enabled

Bit	Name	R/W	Reset Value	Function
14	RT14	RW	0	Rising trigger event configuration bit of line14 0: Disabled 1: Enabled
13	RT13	RW	0	Rising trigger event configuration bit of line13 0: Disabled 1: Enabled
12	RT12	RW	0	Rising trigger event configuration bit of line12 0: Disabled 1: Enabled
11	RT11	RW	0	Rising trigger event configuration bit of line11 0: Disabled 1: Enabled
10	RT10	RW	0	Rising trigger event configuration bit of line10 0: Disabled 1: Enabled
9	RT9	RW	0	Rising trigger event configuration bit of line9 0: Disabled 1: Enabled
8	RT8	RW	0	Rising trigger event configuration bit of line8 0: Disabled 1: Enabled
7	RT7	RW	0	Rising trigger event configuration bit of line7 0: Disabled 1: Enabled
6	RT6	RW	0	Rising trigger event configuration bit of line6 0: Disabled 1: Enabled
5	RT5	RW	0	Rising trigger event configuration bit of line5 0: Disabled 1: Enabled
4	RT4	RW	0	Rising trigger event configuration bit of line4 0: Disabled 1: Enabled
3	RT3	RW	0	Rising trigger event configuration bit of line3 0: Disabled 1: Enabled
2	RT2	RW	0	Rising trigger event configuration bit of line2 0: Disabled 1: Enabled
1	RT1	RW	0	Rising trigger event configuration bit of line1 0: Disabled 1: Enabled
0	RT0	RW	0	Rising trigger event configuration bit of line0 0: Disabled 1: Enabled

The configurable wakeup lines are edge-triggered. No glitches must be generated on these lines.

If a rising edge on a configurable interrupt line occurs during a write operation to the EXTI_RTSR register, the pending bit is not set.

Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

12.3.2. EXTI falling trigger selection register (EXTI_FTSR)

Address offset: 0x04

Reset value: 0x0000 0000

Contains only register bits for configurable events.

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.	FT18	FT17	FT16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	FT18	RW	0	Falling trigger event configuration bit of line18 0: Disabled 1: Enabled
17	FT17	RW	0	Falling trigger event configuration bit of line17 0: Disabled 1: Enabled
16	FT16	RW	0	Falling trigger event configuration bit of line16 0: Disabled 1: Enabled
15	FT15	RW	0	Falling trigger event configuration bit of line15 0: Disabled 1: Enabled
14	FT14	RW	0	Falling trigger event configuration bit of line14 0: Disabled 1: Enabled
13	FT13	RW	0	Falling trigger event configuration bit of line13 0: Disabled 1: Enabled
12	FT12	RW	0	Falling trigger event configuration bit of line12 0: Disabled 1: Enabled
11	FT11	RW	0	Falling trigger event configuration bit of line11 0: Disabled 1: Enabled
10	FT10	RW	0	Falling trigger event configuration bit of line10 0: Disabled 1: Enabled
9	FT9	RW	0	Falling trigger event configuration bit of line9 0: Disabled 1: Enabled
8	FT8	RW	0	Falling trigger event configuration bit of line8 0: Disabled 1: Enabled
7	FT7	RW	0	Falling trigger event configuration bit of line7 0: Disabled 1: Enabled
6	FT6	RW	0	Falling trigger event configuration bit of line6 0: Disabled 1: Enabled
5	FT5	RW	0	Falling trigger event configuration bit of line5 0: Disabled 1: Enabled
4	FT4	RW	0	Falling trigger event configuration bit of line4 0: Disabled 1: Enabled
3	FT3	RW	0	Falling trigger event configuration bit of line3 0: Disabled 1: Enabled
2	FT2	RW	0	Falling trigger event configuration bit of line2 0: Disabled 1: Enabled
1	FT1	RW	0	Falling trigger event configuration bit of line1 0: Disabled 1: Enabled
0	FT0	RW	0	Falling trigger event configuration bit of line0 0: Disabled 1: Enabled

The configurable wakeup lines are edge-triggered. No glitches must be generated on these lines. If a falling edge on a configurable interrupt line occurs during a write operation to the EXTI_FTSR register, the pending bit is not set.

Rising and falling edge triggers can be set for the same interrupt line. In this case, both generate a trigger condition.

12.3.3. Software interrupt event register (EXTI_SWIER)

Address offset: 0x08

Reset value: 0x0000 0000

Contains only register bits for configurable events.

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.	SW18	SW17	SW16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SW15	SW14	SW13	SW12	SW11	SW10	SW9	SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	SWI18	RW	0	Rising trigger event configuration bit of line18 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
17	SWI17	RW	0	Rising trigger event configuration bit of line17 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is self-cleared by the hardware. Read returns 0.
16	SWI16	RW	0	Rising trigger event configuration bit of line16 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
15	SWI15	RW	0	Rising trigger event configuration bit of line15 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
14	SWI14	RW	0	Rising trigger event configuration bit of line14 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is self-cleared by the hardware. Read returns 0.
13	SWI13	RW	0	Rising trigger event configuration bit of line13 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is self-cleared by the hardware. Read returns 0.
12	SWI12	RW	0	Rising trigger event configuration bit of line12 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
11	SWI11	RW	0	Rising trigger event configuration bit of line11 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
10	SWI10	RW	0	Rising trigger event configuration bit of line10 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
9	SWI9	RW	0	Rising trigger event configuration bit of line9 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is self-cleared by the hardware. Read returns 0.
8	SWI8	RW	0	Rising trigger event configuration bit of line8

Bit	Name	R/W	Reset Value	Function
				0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
7	SWI7	RW	0	Rising trigger event configuration bit of line7 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
6	SWI6	RW	0	Rising trigger event configuration bit of line6 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
5	SWI5	RW	0	Rising trigger event configuration bit of line5 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
4	SWI4	RW	0	Rising trigger event configuration bit of line4 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
3	SWI3	RW	0	Rising trigger event configuration bit of line3 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
2	SWI2	RW	0	Rising trigger event configuration bit of line2 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
1	SWI1	RW	0	Rising trigger event configuration bit of line1 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
0	SWI0	RW	0	Rising trigger event configuration bit of line0 0: No effect 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)

12.3.4. Pending register (EXTI _ PR)

Address offset: 0x0C

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.													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
31:19	Reserved	-	-	Reserved
18	PR18	RC_W1	0	Rising trigger event configuration bit of line18 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred

Bit	Name	R/W	Reset Value	Function
				1: Generate rising edge/falling edge/software trigger event request;
17	PR17	RC_W1	0	Rising trigger event configuration bit of line17 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
16	PR16	RC_W1	0	Rising trigger event configuration bit of line16 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
15	PR15	RC_W1	0	Rising trigger event configuration bit of line15 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
14	PR14	RC_W1	0	Rising trigger event configuration bit of line14 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
13	PR13	RC_W1	0	Rising trigger event configuration bit of line13 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
12	PR12	RC_W1	0	Rising trigger event configuration bit of line12 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
11	PR11	RC_W1	0	Rising trigger event configuration bit of line11 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
10	PR10	RC_W1	0	Rising trigger event configuration bit of line10 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
9	PR9	RC_W1	0	Rising trigger event configuration bit of line9 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
8	PR8	RC_W1	0	Rising trigger event configuration bit of line8 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
7	PR7	RC_W1	0	Rising trigger event configuration bit of line7 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;

Bit	Name	R/W	Reset Value	Function
6	PR6	RC_W1	0	Rising trigger event configuration bit of line6 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
5	PR5	RC_W1	0	Rising trigger event configuration bit of line5 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
4	PR4	RC_W1	0	Rising trigger event configuration bit of line4 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
3	PR3	RC_W1	0	Rising trigger event configuration bit of line3 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
2	RPIF2	RC_W1	0	Rising trigger event configuration bit of line2 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
1	PR1	RC_W1	0	Rising trigger event configuration bit of line1 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
0	PR0	RC_W1	0	Rising trigger event configuration bit of line0 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;

12.3.5. EXTI external interrupt selection register 1 (EXTI_EXTICR1)

Address offset: 0x60

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.	EXTI3 [1:0]		Res	Res.	Res.	Res.	Res.	Res.	EXTI2 [1:0]	
						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.	EXTI1 [1:0]		Res	Res.	Res	Res.	Res.	Res.	EXTI0 [1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI3 [1:0]	RW	0	EXIT3 GPIO port selection 2'b00: PA[3] pin 2'b01: PB[3] pin 2'b10: PF[3] pin 2 'b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI2 [1:0]	RW	0	EXTI2 corresponds to GPIO port selection. 2'b00: PA[2] pin 2'b01: PB[2] pin 2'b10: PF[2] pin 2 'b11: reserved

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	-	Reserved
9:8	EXTI1 [1:0]	RW	0	EXTI1 corresponds to GPIO port selection. 2'b00: PA[1] pin 2'b01: PB[1] pin 2'b10: PF[1] pin 2 'b11: reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI0 [1:0]	RW	0	EXTI0 corresponds to GPIO port selection. 2'b00: PA[0] pin 2'b01: PB[0] pin 2'b10: PF[0] pin 2 'b11: reserved

12.3.6. EXTI external interrupt selection register 2 (EXTI_EXTICR2)

Address offset: 0x64

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.	EXTI7		Res	Res.	Res.	Res.	Res.	Res.	EXTI6	
						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.	EXTI5		Res	Res.	Res	Res.	Res.	Res.	EXTI4 [1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI7 [1:0]	RW	0	EXTI7 corresponds to GPIO port selection. 2'b00: PA[7] pin 2'b01: PB[7] pin 2'b10: PF[7] pin 2 'b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI6 [1:0]	RW	0	EXTI6 corresponds to GPIO port selection. 2'b00: PA[6] pin 2'b01: PB[6] pin 2'b10: PF[6] pin 2 'b11: reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI5 [1:0]	RW	0	EXTI5 corresponds to GPIO port selection. 2'b00: PA[5] pin 2'b01: PB[5] pin 2'b10: PF[5] pin 2 'b11: reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI4 [1:0]	RW	0	EXTI4 corresponds to GPIO port selection. 2'b00: PA[4] pin 2'b01: PB[4] pin 2'b10: PF[4] pin 2 'b11: reserved

12.3.7. EXTI external interrupt selection register 3 (EXTI_EXTICR3)

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.	EXTI11		Res	Res.	Res.	Res.	Res.	Res.	EXTI10	
						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.	EXTI9		Res	Res.	Res	Res.	Res.	Res.	EXTI8 [1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI11 [1:0]	RW	0	EXTI11 corresponds to GPIO port selection.

Bit	Name	R/W	Reset Value	Function
				2'b00: PA[11] pin 2'b01: PB[11] pin 2'b10: PF[11] pin 2 'b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI10 [1:0]	RW	0	EXTI10 corresponds to GPIO port selection. 2'b00: PA[10] pin 2'b01: PB[10] pin 2'b10: PF[10] pin 2 'b11: reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI9 [1:0]	RW	0	EXTI9 corresponds to GPIO port selection. 2'b00: PA[9] pin 2'b01: PB[9] pin 2'b10: PF[9] pin 2 'b11: reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI8 [1:0]	RW	0	EXTI8 corresponds to GPIO port selection. 2'b00: PA[8] pin 2'b01: PB[8] pin 2'b10: PF[8] pin 2 'b11: reserved

12.3.8. EXTI external interrupt selection register 3 (EXTI_EXTICR3)

Address offset: 0x6C

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.	EXTI15	Res	Res.	Res.	Res.	Res.	Res.	Res.	EXTI14
							RW								RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTI13	Res	Res.	Res	Res.	Res.	Res.	Res.	EXTI12
							RW								RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	EXTI15	RW	0	EXTI15 corresponds to GPIO port selection. 1'b0: PA[12] pin 1'b1: PB[12] pin
23:17	Reserved	-	-	Reserved
16	EXTI14	RW	0	EXTI14 corresponds to GPIO port selection. 1'b0: PA[12] pin 1'b1: PB[12] pin
15:9	Reserved	-	-	Reserved
8	EXTI13	RW	0	EXTI15 corresponds to GPIO port selection. 1'b0: PA[12] pin 1'b1: PB[12] pin
7:1	Reserved	-	-	Reserved
0	EXTI12	RW	0	EXTI12 corresponds to GPIO port selection. 1'b0: PA[12] pin 1'b1: PB[12] pin

12.3.9. Interrupt mask register (EXTI _ IMR)

Address offset: 0x80

Reset value: 0x2008 0000

Note: The reset value is set such as to, by default, enable interrupt from direct lines, and disablein-terrupt from configurable lines.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IM29	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IM19	IM18	IM17	IM16
		RW										RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
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	IM29	RW	1	Interrupt mask on line 29. 0: Interrupt request is masked 1: Interrupt request is not masked
28:20	Reserved			
19	IM19	RW	1	EXTI line19 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
18	IM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	IM17	RW	0	Interrupt mask on line 17. 0: Interrupt request is masked 1: Interrupt request is not masked
16	IM16	RW	0	EXTI line16 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
15	IM15	RW	0	EXTI line15 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
14	IM14	RW	0	EXTI line14 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
13	IM13	RW	0	EXTI line13 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
12	IM12	RW	0	EXTI line12 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
11	IM11	RW	0	EXTI line11 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
10	IM10	RW	0	EXTI line10 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
9	IM9	RW	0	EXTI line9 wakes up CPU mask control as an in-terrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
8	IM8	RW	0	EXTI line8 wakes up CPU mask control as an in-terrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
7	IM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	IM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	IM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked 1: Interrupt request is not masked
4	IM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked

Bit	Name	R/W	Reset Value	Function
3	IM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	IM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	IM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	IM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

12.3.10. Event mask register (EXTI_EMR)

Address offset: 0x84

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EM29	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EM19	EM18	EM17	EM16
		RW										RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
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	EM29	RW	0	Interrupt mask on line 29. 0: Interrupt request is masked 1: Interrupt request is not masked
28:20	Reserved			
19	EM19	RW	0	EXTI line19 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
18	EM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	EM17	RW	0	Interrupt mask on line 17. 0: Interrupt request is masked 1: Interrupt request is not masked
16	EM16	RW	0	EXTI line16 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
15	EM15	RW	0	EXTI line15 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
14	EM14	RW	0	EXTI line14 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
13	EM13	RW	0	EXTI line13 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
12	EM12	RW	0	EXTI line12 wakes up CPU mask control as an event.

Bit	Name	R/W	Reset Value	Function
				0: Interrupt request is masked 1: Interrupt request is not masked
11	EM11	RW	0	EXTI line11 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
10	EM10	RW	0	EXTI line10 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
9	EM9	RW	0	EXTI line9 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
8	EM8	RW	0	EXTI line8 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
7	EM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	EM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	EM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked 1: Interrupt request is not masked
4	EM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked
3	EM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	EM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	EM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	EM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

13. Cyclic redundancy check calculation unit (CRC)

13.1. Introduction

The CRC (cyclic redundancy check) calculation unit is used to get a CRC code from 32-bit data word and a generator polynomial.

13.2. CRC main features

- The default polynomial value is the CRC-32 (Ethernet) polynomial: 0x4C11DB7.
- X32 + X26 + X23 +X22 +X16 + X12 + X11 + X10 +X8 + X7 + X5 + X4 + X2 + X +1
- Supports 32-bit data input
- Single input/output 32-bit data register
- General purpose 8-bit register (can be used as temporary storage)
- CRC computation done in 4 AHB clock cycles (HCLK) for the 32-bit data size

13.3. CRC functional description

13.3.1. CRC block diagram

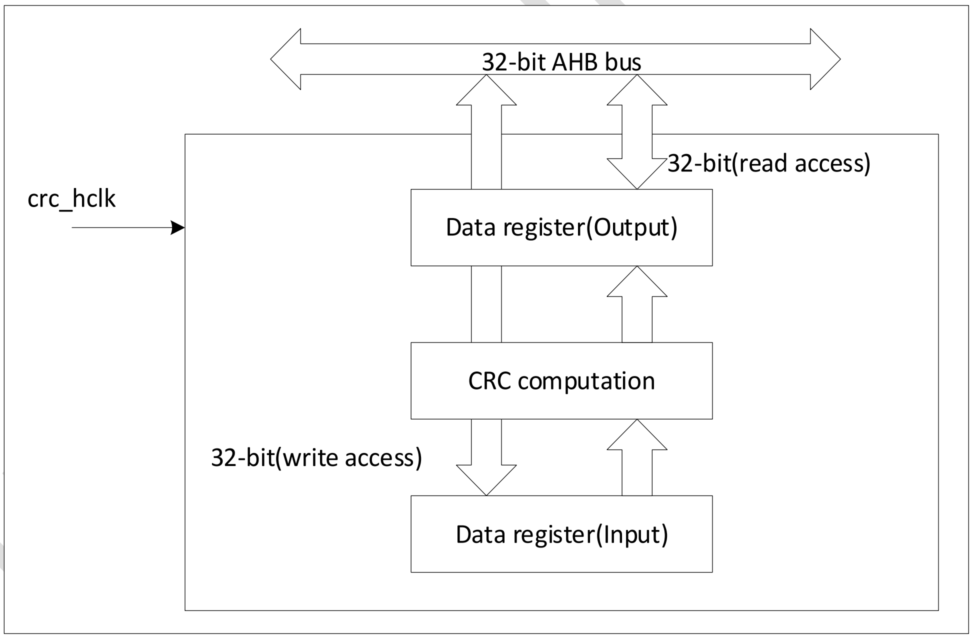


Figure 13-1 CRC calculation unit block diagram

The CRC calculation unit contains a 32-bit data register:

- When writing to this register, it serves as an input register, allowing you to input new data for CRC calculation.
- When reading from this register, it returns the result of the previous CRC calculation.

Each time data is written to the register, the calculation result is a combination of the previous CRC calculation result and the new one (CRC calculation is performed on the entire 32-bit word rather than byte by byte).

While the CRC is being calculated, the write operation is blocked until the CRC calculation ends.

You can reset the register CRC_DR to 0xFFFFFFFF by setting the RESET bit in the register CRC_CR. This operation does not affect the data in the register CRC_IDR

13.4. CRC registers

13.4.1. 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															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:0	DR	RW	32’hFFFFFFFF	Data register. As an input register when new data is written. It holds the previous CRC calculation result when it is read.

13.4.2. CRC 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
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.	IDR[7:0]							
								RW							

Bit	Name	R/W	Reset Value	Function
31:8	Reserved		-	
7:0	IDR[7:0]	RW	0	General-purpose 8-bit data register These bits can be used as a temporary storage location for bytes. This register is not affected by CRC resets generated by the RESET bit in the CRC_CR register.

13.4.3. 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.	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.	RESET
															W

Bit	Name	R/W	Reset Value	Function
31:1	Reserved		-	
0	RESET		0	This bit is set by software to reset the CRC calculation unit. This bit can only be set, it is automatically cleared by hardware.

14. Analog-to-digital converters (ADC)

14.1. Introduction

The chip has a 12-bit SARADC (successive approximation analog-to-digital converter). The module has a total of up to 15 channels to be measured, including 10 external and 5 internal channels.

A/D conversion of the various channels can be performed in single, continuous, scan (single channel, all channels) or discontinuous mode. The continuous mode combined with the scanning mode may implement a cyclic mode. The result of the ADC is stored in a left-aligned or right-aligned 16-bit data register.

The analog watchdog feature allows the application to detect if the input voltage goes outside the user-defined higher or lower thresholds.

An efficient low-power mode is implemented to allow very low consumption at low frequency.

14.2. ADC main features

- High performance
  - 12 bits, 10 bits, 8 bits, and 6 bits resolutions are configurable
  - ADC conversion time: 1us @ 12bit (1 MHz)
  - Support self - calibration (initiated by software)
  - Programmable sampling time
  - Programmable data alignment mode (left or right alignment)
  - Support DMA function
- Low-power
  - Reduce PCLK frequency for low power operation while still maintaining proper ADC performance
  - Auto off mode: prevents ADC overrun in applications with low frequency PCLK
- Analog input channels
  - 10 external analog input channels: PA [7: 0] and PB [1: 0]
  - 1 internal temperature sensor channel
  - 1 channel for internal reference voltage (VREFINT)
  - 1 internal channel (VCCA/3)
  - 2 OPA channels
- The conversion operation can be initiated by
  - By software
  - Hardware boot with configurable polarity (TIM1, TIM2, or GPIO via EXTI11)
- Conversion modes
  - Single mode: converts a single channel or can scan a sequence of channels
  - Continuous mode: converts selected inputs continuously
  - Discontinuous mode: converts selected inputs once per trigger
- Interrupt generation
  - At the end of sampling

— End of conversion

— End of sequence conversion

— Analog watchdog

— Overflow Event

■ Analog watchdog

14.3. ADC functional descriptionADC functional description

14.3.1. ADC block diagram

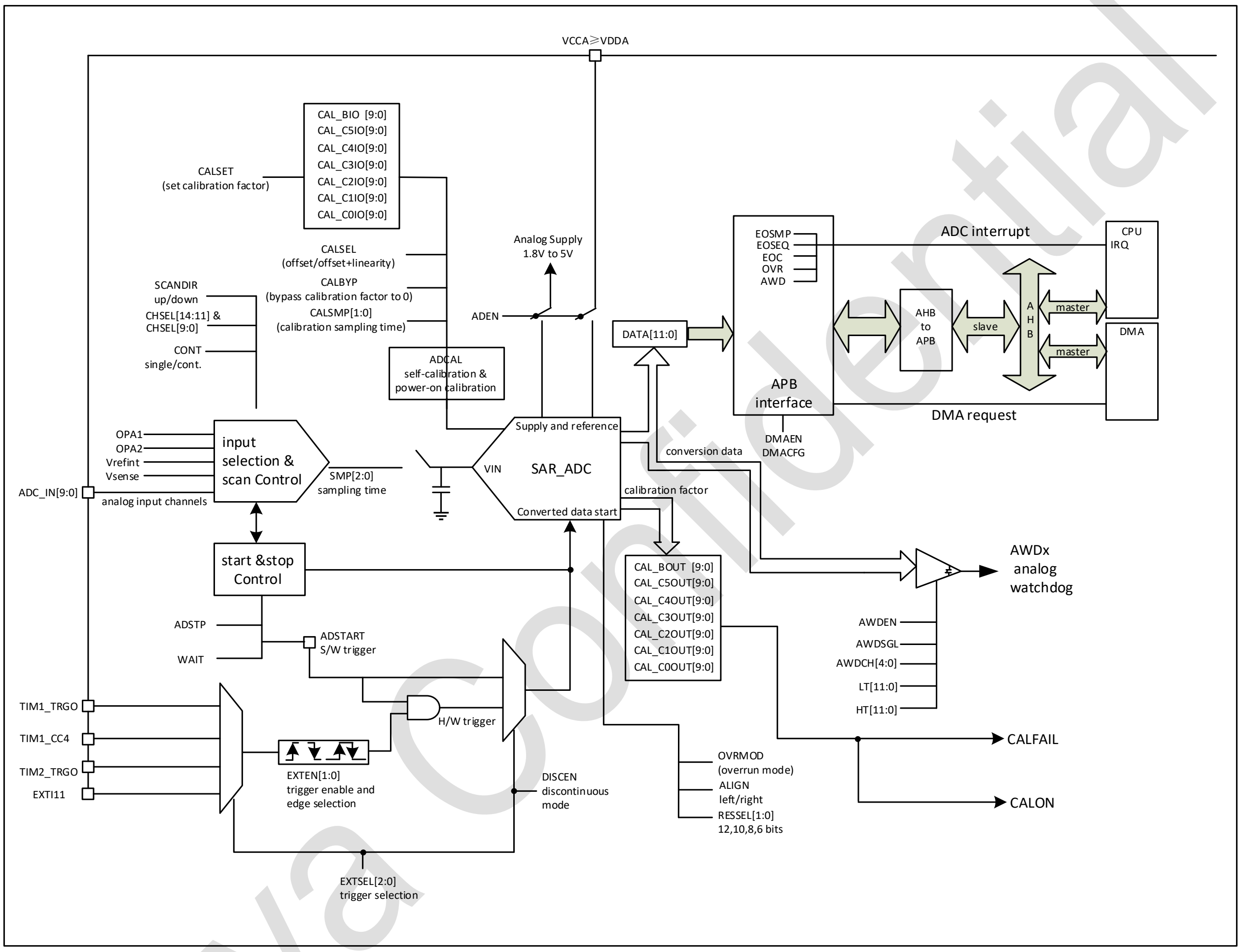


Figure 14-1 ADC system architecture

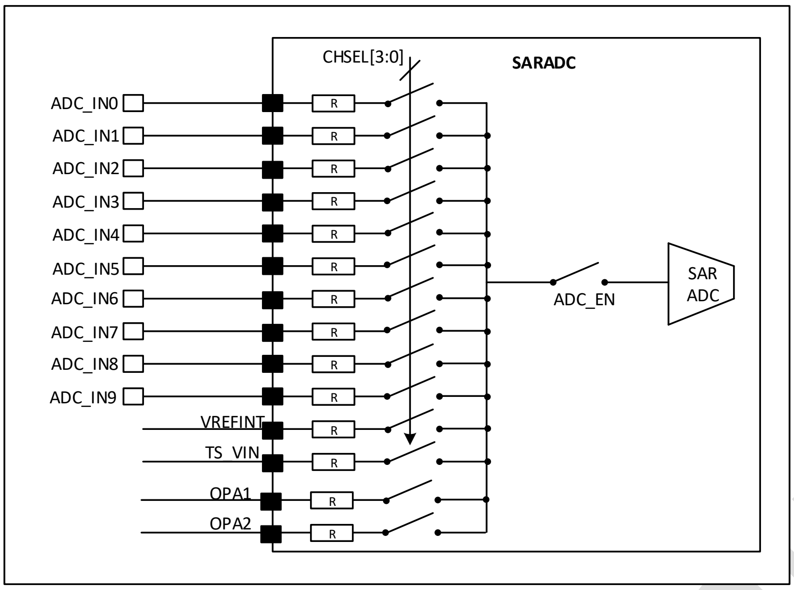


Figure 14-2 ADC channel with analog switch

14.3.2. Calibration (ADCAL)

This ADC has a calibration function (software start). During the procedure, the ADC calculates a calibration factor which is internally applied to the ADC until the next ADC power-off (lost after the ADC is powered down). The application cannot use the ADC module during the ADC calibration and until the calibration is complete.

Calibration is preliminary to any ADC operation. Calibration is used to eliminate offset errors from chip to chip due to process variations.

ADC software calibration

The software sets ADCAL = 1 to start the calibration. The calibration can only be started when the ADC is not enabled (ADEN = 0), and only the system clock is supported as the clock of the ADC. When the calibration is complete, the ADCAL is cleared to 0 by the hardware. After calibration is complete, the calibration factor can be read out from the ADC _ CALFACTOR register. The calibration factor is maintained until a system reset occurs.

When the operating conditions of the ADC change (VCC change is the main factor of ADC offset shift, followed by temperature change), it is recommended to perform a recalibration operation.

Software procedure to calibrate the ADC:

- Confirm ADEN = 0, CKMODE Select System Clock
- Set ADCAL=1
- Wait until ADCAL=0

14.3.3. ADC on-off control (ADEN)

At MCU power-up, the ADC is disabled and put in power-down mode (ADEN=0).

The ADEN bit is used to enable or disable the ADC.

The ADC transformation is also initiated by setting ADSTRART or, if hardware triggered start, by an external triggered event.

The ADSTART, ADSTP, ADEN and ADDIS registers are configured in accordance with the following principles:

- When the software is not configured with ADEN = 1, any bit of ADSTART/ADSTP/ADDIS cannot be configured to be 1 (hardware masking), that is, when ADEN = 1 is configured, ADSTART/ADSTP/ADDIS must be 0, or when ADSTART/ADSTP/ADDIS is configured to be 1, ADEN = 1;
- When ADSTART is 0, the software cannot set ADSTP (hardware masking);
- When ADEN = 1 and ADSTART is 0, the software can configure ADDIS = 1 (hardware shielding);
- If ADEN = 1, setting ADSTP (ADSTART = 1 is the premise) or ADDIS will clear ADEN; After ADEN is cleared, ADDIS is also cleared;
- The software clears ADEN and then re-enables ADEN and ADSTART at 4 ADC _ CLK cycles.

Caution: ADEN bit cannot be set during four ADC clock cycles after the ADCAL bit is cleared by hardware and when ADCAL = 1 (end of calibration).

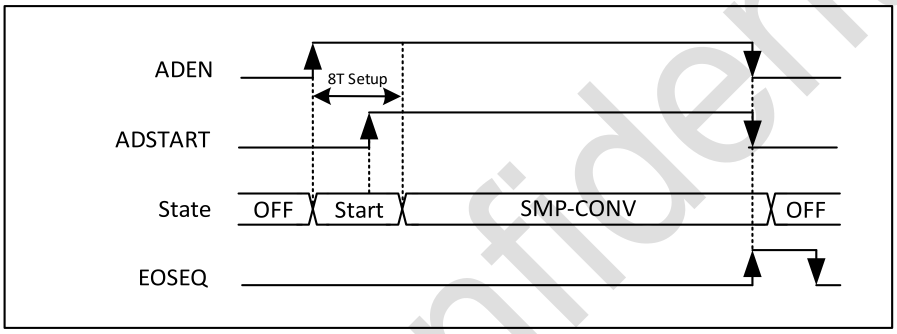


Figure 14-3 Enable/Disable ADC

14.3.4. ADC clock

The ADC has a dual clock-domain architecture, so that the ADC clock (ADC_CLK) is independent from the APB clock (PCLK). ADC _ CLK may be generated by two possible clock sources.

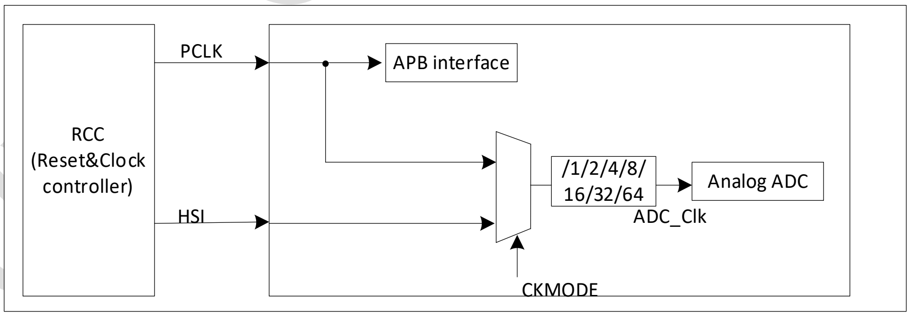


Figure 14-4 ADC clock scheme

Table 14-1 Delay between table trigger and start of transition

ADC clock source	CKMODE[3:0]	dividing factor	Latency between the trigger event and the start of conversion (T Is the clock cycle)
PCLK	0000	1	0
	0001	2	0
	0010	4	0
	0011	8	0
	0100	16	0
	0101	32	0

	0110	64	0
	0111	/	/
HSI	1000	1	0
	1001	2	0
	1010	4	0
	1011	8	0
	1100	16	0
	1101	32	0
	1110	64	0
	1111	/	/

14.3.5. Configuring the ADC

Software must write to the ADCAL and ADEN bits in the ADC_CR register if the ADC is disabled (ADEN must be 0). Software must only write to the ADSTART bit in the ADC_CR register only if the ADC is enabled and there is no pending request to disable the ADC (ADEN = 1).

For all the other control bits in the ADC_IER, ADC_CFGRi, ADC_SMPR, ADC_TR, ADC_CHSELR and ADC_CCR registers, software must only write to the configuration control bits if the ADC is enabled (ADEN = 1) and if there is no conversion ongoing (ADSTART = 0). ADC _ CHSELR is written with ADEN = 0 and ADSTART = 0.

Software must only write to the ADSTP bit in the ADC_CR register only if the ADC is enabled and there is no pending request to disable the ADC (ADSTART = 1).

14.3.6. Channel selection (CHSEL, SCANDIR)

There are up to 15 multiplexed channels:

- 10 analog inputs from GPIO pins (ADC_IN0...ADC_IN9)
- 5 internal analog inputs (temperature sensing, internal reference voltage, VCCA/3, OPA1 output and OPA2 output)

It is possible to convert a single channel or to automatically scan a sequence of channels.

The sequence of the channels to be converted must be programmed in the ADC_CHSELR channel selection register: each analog input channel has a dedicated selection bit. The ADC _ SEQRx register controls the priority of the transition channel.

The order in which the channels will be scanned can be configured by programming the bit SCANDIR bit in the ADC_CFGR1 register:

- SCANDIR = 0: forward scan: from SQ0 to channel SQ14
- SCANDIR = 1: Fallback scan: from SQ14 to SQ0

The temperature sensing is connected to the ADC _ IN10 (TS _ VIN) channel, the internal reference voltage is connected to the ADC _ IN11 channel (VREFINT), VCCA/3 is connected to ADC _ IN12, the OPA1 output is connected to ADC _ IN13, and the OPA2 output is connected to ADC _ IN14. The ADC _ CHSELR register controls the transition priority (but the scan order is still from small to large according to the channel number), and the ADC _ SEQRx register controls the transition channel.

For example, when the priority conversion ADC _ IN14, the second conversion ADC _ IN3, and the last conversion ADC _ IN0 are selected, different channel priority conversion can be realized only by enabling the values of CHSEL0-2 in the ADC _ CHESELR register, the ADC _ SEQR0.SQ0 register to write channel 14, the ADC _ SEQR0.SQ1 register to write channel 3, and the ADC _ SEQR0.SQ2 to write channel 0.

Index	Register0	Register1	Register2	Register3	...	Register13	Register14
0	chsel0	chsel1	chsel2	chsel3	...	chsel13	chsel14
1	SMP0	SMP1	SMP2	SMP3	...	SMP13	SMP14
2	Seq0	Seq1	Seq2	Seq3	...	Seq13	Seq14

14.3.7. Programmable sampling time (SMP)

Before starting a conversion, the ADC needs to establish a direct connection between the voltage source to be measured and the embedded sampling capacitor of the ADC. This sampling time must be enough for the input voltage source to charge the sample and hold capacitor to the input voltage level.

Having a programmable sampling time allows to trim the conversion speed according to the input resistance of the input voltage source.

The number of ADC clocks used by the ADC to sample the input voltage can be modified by SMP [2: 0] bits in the ADC _ SMPR1 and ADC _ SMPR2 registers, and each channel can be independently configured with different sampling times. If required by the application, the software can change and adapt this sampling time between each conversions.

The total conversion time is calculated as follows:

t_{CONV} = (Sampling time + (conversion resolution + 0.5) x ADC clock cycles

For example:

With ADC_CLK = 16 MHZ, resolution is 12 bit, and a sampling time of 3.5 ADC clock cycles:

tCONV = (3.5 + 12.5) x ADC clock cycles = 16 x ADC clock cycles = 1 μ s

The EOSMP flag bit is used to indicate the end of the sampling phase.

14.3.8. Single conversion mode (CONT = 0, DISCEN = 0)

In Single conversion mode, the ADC performs a single sequence of conversions, converting all the channels once. This mode is selected when CONT=0 and DISCEN = 0 in the ADC_CFGR1 register.

Conversion is started by either:

- Setting the ADSTART bit in the ADC_CR register
- Hardware trigger event

Inside the sequence, after each conversion is complete:

- The converted data are stored in the 16-bit ADC_DR register.
- The EOC (end of conversion) flag is set
- An interrupt is generated if the EOCIE bit is set.

After the sequence of conversions is complete:

- The EOSEQ (end of sequence) flag is set
- An interrupt is generated if the EOSIE bit is set.

Then the ADC stops until a new external trigger event occurs or the ADSTART bit is set again.

Note: To convert a single channel, program a sequence with a length of 1. ADC _ CHSEL simply selects the current transition channel enabled. Channel transition priority ADC _ SEQRx control. If the switch channel CHSEL0 = 1, CHSEL12 = 1 are selected, and the switch channel 12 is preferred, the switch channel 12 needs to be written to the register SQ0.

14.3.9. Continuous conversion mode (CONT=1)

In continuous conversion mode, when a software or hardware trigger event occurs, the ADC performs a sequence conversion. Converts all channels once and automatically restarts performing the same sequence conversion. When CONT = 1 in the register ADC _ CFGR1, the ADC is selected to the continuous conversion mode. Conversion is started by either:

- Setting the ADSTART bit in the ADC_CR register
- Hardware trigger event

Inside the sequence, after each conversion is complete:

- The converted data are stored in the 16-bit ADC_DR register.
- The EOC (end of conversion) flag is set
- An interrupt is generated if the EOCIE bit is set.

After the sequence of conversions is complete:

- The EOSEQ (end of sequence) flag is set
- An interrupt is generated if the EOSEQIE bit is set

(A new sequence restarts immediately and the ADC continuously repeats the conversion sequence.)

Note: To convert a single channel, program a sequence with a length of 1.

The ADC cannot be in a discontinuous conversion mode and a continuous conversion mode at the same time, in which case (DISCEN = 1, CONT = 1), it behaves as a single conversion mode. The ADC _ CHSELR register simply selects the current transition channel enabled. The channel transition priority is controlled by the ADC _ SEQRx register. If the switch channel CHSEL0 = 1 and CHSEL12 = 1 are selected and the switch channel 12 is preferred, the switch channel 12 needs to be written to the register SQ0.

14.3.10. Discontinuous conversion mode (DISCEN = 1)

This mode is turned on by setting the DISCEN bit in the ADC _ CFGR1 register.

In this mode (DISCEN = 1), a hardware or software trigger event is required to initiate each transition defined in a sequence.

On the contrary, when DISCEN = 0, a hardware or software trigger event can start all transformations defined in a sequence.

For example:

DISCEN = 1, the channels to be converted are: 0, 3, 7, 10, and SQ0 = 0, SQ1 = 3, SQ2 = 7, SQ3 = 10 in ADC _ SEQ0:

-1st trigger: Channel 0 is converted and an EOC event is generated

-2nd trigger: Channel 3 is switched and an EOC event is generated

-3rd Trigger: Channel 7 is switched and an EOC event is generated

-4th Trigger: Channel 10 is switched and EOC and EOSEQ events are generated

-5th trigger: Channel 0 is switched and an EOC event is generated

-6th trigger: Channel 3 is switched and an EOC event is generated

- ...

DISCEN = 0, the channels to be converted are: 0, 3, 7, 10, and SQ0 = 0, SQ1 = 3, SQ2 = 7, SQ3 = 10 in ADC _ SEQ0:

- 1st trigger: The entire complete sequence transition, followed by channels 0, 3, 7 and 10.

Each time the conversion is completed, an EOC event is generated, and the conversion to the last channel generates an EOSEQ event in addition to the EOC.

-Any trigger event will restart the complete sequence transition.

Note: It is not possible to have the ADC in both continuous conversion mode and non-continuous conversion mode. In this configuration (DISCEN = 1, CONT = 1), it behaves as a single conversion mode. The channel transition priority is controlled by the ADC _ SEQRx register. If the switch channel CHSEL0 = 1 and CHSEL12 = 1 are selected, if the switch channel 12 is preferred, the switch channel 12 needs to be written to the register SQ0.

14.3.11. Starting conversions (ADSTART)

Software starts ADC conversions by setting ADSTART=1.

When ADSTART is set, the conversion:

- Starts immediately if EXTEN = 00 (software trigger)
- At the next active edge of the selected hardware trigger if EXTEN ≠ 00

The ADSTART bit is also used to indicate whether an ADC operation is currently ongoing. When the ADC is idle, this bit can be reconfigured to ADSTART = 0.

The ADSTART bit is cleared by hardware.

- Single transition mode is triggered by software (CONT = 0, EXTSEL = 0x0)

-after completion of sequence conversion (EOSEQ = 1)

- Discontinuous conversion mode is triggered by software (CONT = 0, DISCEN = 1, EXTSEL = 0x0)

  -after completion of conversion (EOC = 1)

- In all cases (CONT = X, EXTSEL = X)

-After the software calls and executes the ADSTP procedure

NOTE: In continuous mode (CONT = 1), the ADSTART bit cannot be cleared by the hardware caused by EOSEQ because the sequence conversion is automatically restarted. When hardware trigger is selected in single mode (CONT=0 and EXTEN = 0x01), ADSTART is not cleared by hard-ware when the EOSEQ flag is set. This avoids the need for software to reset the ADSTART bit and ensures that no hardware trigger events are missed.

14.3.12. ADC timing

The elapsed time between the start of a conversion and the end of conversion is the sum of the configured sampling time plus the successive approximation time depending on data resolution:

tADC = tSMPL + tSAR = [3.5 min + 12.5 12bit] x tADC _ CLK

tADC = tSMPL + tSAR = 218.75 ns min + 781.25 ns 12bit = 1 µ s min for fADC _ CLK = 16 MHz

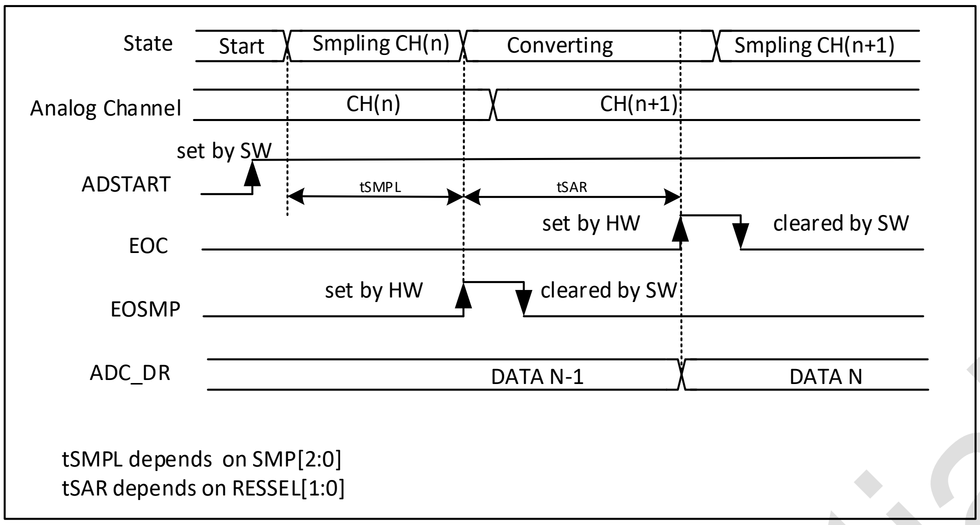


Figure 14-5 Analog-to-digital conversion timing

14.3.13. Stopping an ongoing conversion (ADSTP)

The software can decide to stop any ongoing conversions by setting ADSTP=1 in the ADC_CR register. This will reset the ADC operation and the ADC will be idle, ready for a new operation.

When the ADSTP bit is set by software, any ongoing conversion is aborted and the result is discarded (ADC_DR register is not updated with the current conversion).

The scan sequence is also aborted and reset (meaning that restarting the ADC would restart a new sequence).

Once this procedure is complete, the ADSTP and ADSTART bits are both cleared by hardware.

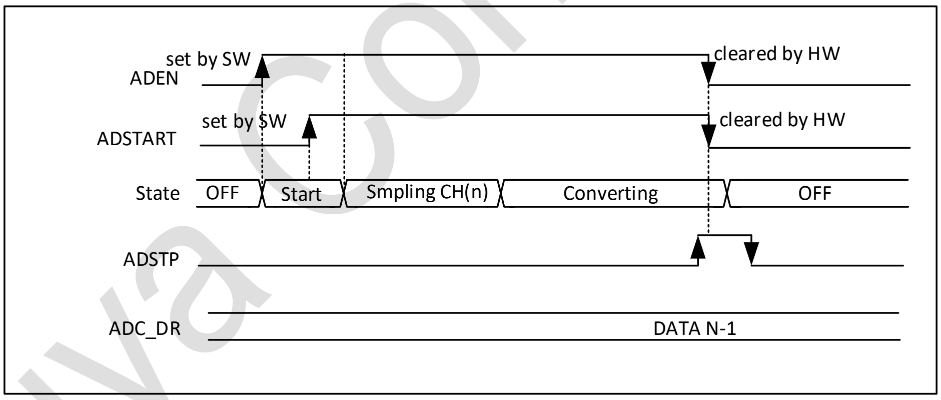


Figure 14-6 Stop timing

14.4. Conversion on external trigger and trigger polarity (EXTSEL, EXTEN)

A conversion or a sequence of conversion can be triggered either by software or by an external event (for example timer and input pin). If the EXTEN[1:0] ≠ “00”, then external events are able to trigger a conversion with the selected polarity. The trigger selection is effective once software has set bit ADSTART=1.

Any hardware triggers which occur while a conversion is ongoing are ignored.

If bit ADSTART=0, any hardware triggers which occur are ignored.

Source	EXTEN[1:0]
Trigger detection disabled	00
Detection on rising edge	01
Detection on falling edge	10
Detection on both rising and falling edges	11

Note: External trigger polarity cannot be changed during transition. The EXTSEL[2:0] control bits are used to select possible events that can trigger conversions.

The following table gives possible external triggers for rule transformations. Software source trigger events can be generated by setting the ADSTART bit in the ADC_CR register.

Table 14-2 External triggers

Name	source	EXTSEL[2:0]
EXT0	TIM1_TRGO	000
EXT1	TIM1_CC4	001
EXT2	TIM2_TRGO	010
EXT3	Reserved	011
EXT4	Reserved	100
EXT5	Reserved	101
EXT6	Reserved	110
EXT7	EXTI11	111

Note: The external trigger source cannot be changed during conversion.

14.4.1. Quick transition mode

A faster transition time (t_{SAR}) can be obtained by reducing the transition resolution. The conversion resolution can be configured to 12/10/8/6 bits by setting RES[1: 0] in the ADC_CFGR1 register. When applications do not require high-precision data, low conversion resolution can be used to speed up conversion times. The conversion result is also 12 bits wide, and the lower bits are supplemented by 0.

Resolution mode reduces the conversion time of successive approximations, such as:

RESSEL [1:0]	t_{SAR} (ADC Clock Cycle)	tSAR (ns) @ fADC = 24 MHz	t_{SMP} (ADC Clock Cycle)	t_{ADC}(t_{SMP} = 3.5) (ADC Clock Cycle)	tSAR (ns) @ fADC = 24 MHz
12	12.5	521 ns	3.5	16	667 ns
10	10.5	438 ns	3.5	14	583 ns
8	8.5	396 ns	3.5	12	500 ns
6	6.5	271 ns	3.5	10	417 ns

14.4.2. End of conversion / end of sampling phase

The ADC notifies the application of an end of each transition (EOC) event.

The ADC sets the EOC flag in the ADC_ISR register as soon as a new conversion data result is available in the ADC_DR register. An EOC interrupt can be generated if the EOCIE bit is set in the ADC_IER register. The EOC flag is cleared by software either by writing 1 to it, or by reading the ADC_DR register.

The ADC also indicates the end of sampling phase by setting the EOSMP flag in the ADC_ISR register. The EOSMP flag is cleared by software by writing1 to it. An EOSMP interrupt can be generated if the EOSMPIE bit is set in the ADC_IER register.

14.4.3. End of conversion sequence (EOSEQ flag)

The ADC notifies the application of the End of Each Sequence Conversion (EOSEQ) event.

The ADC sets the EOSEQ flag in the ADC_ISR register as soon as the last data result of a conversion sequence is available. An interrupt can be generated if the EOSEQIE bit is set in the ADC_IER register. The EOSEQ flag is cleared by software by writing 1 to it.

14.4.4. Sampling time diagram

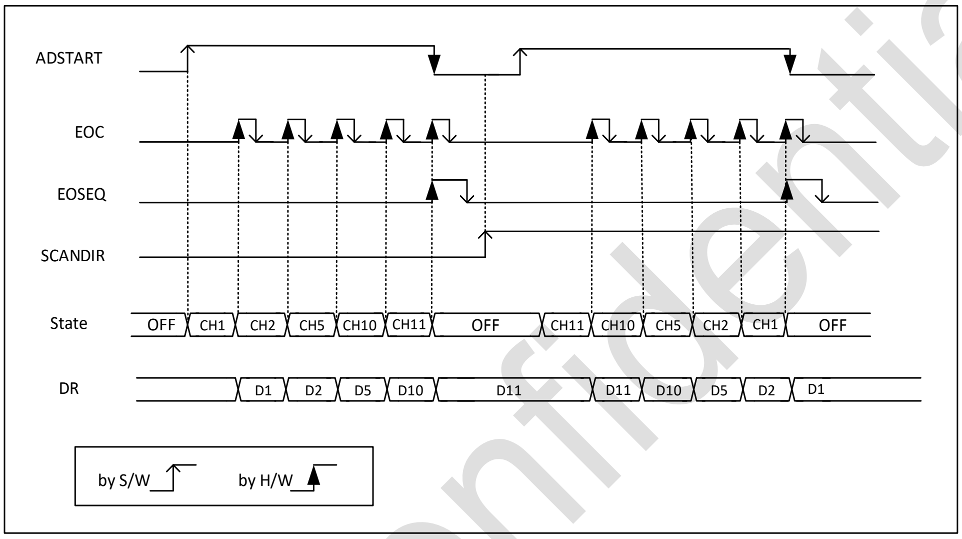

ADSTART
EOC
EOSEQ
SCANDIR
State
OFF CH1 CH2 CH5 CH10 CH11 OFF CH11 CH10 CH5 CH2 CH1 OFF
DR
D1 D2 D5 D10 D11 D11 D10 D5 D2 D1
by S/W
by H/W

Figure 14-7 Single conversions of a sequence, software trigger

1. EXTEN = 0x0, CONT = 0
2. CHSEL = 0x20601, WAIT = 0, AUTOFF = 0

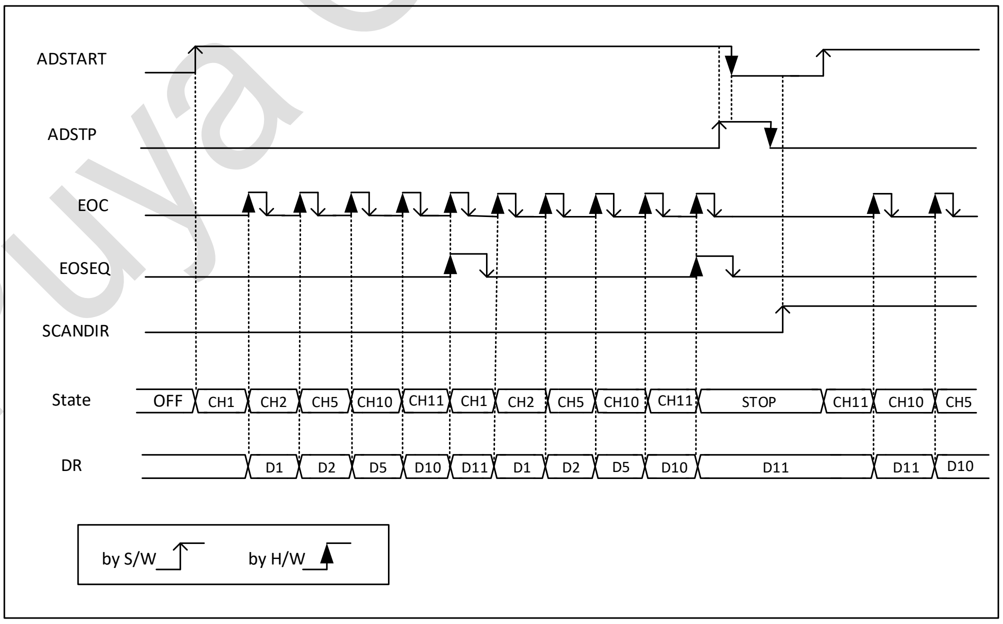

ADSTART
ADSTP
EOC
EOSEQ
SCANDIR
State
OFF CH1 CH2 CH5 CH10 CH11 CH1 CH2 CH5 CH10 CH11 STOP CH11 CH10 CH5
DR
D1 D2 D5 D10 D11 D1 D2 D5 D10 D11 D11 D10
by S/W
by H/W

Figure 14-8 Continuous conversion of a sequence, software trigger

1. EXTEN = 0x0, CONT = 1,
2. CHSEL = 0x20601, WAIT = 0, AUTOFF = 0

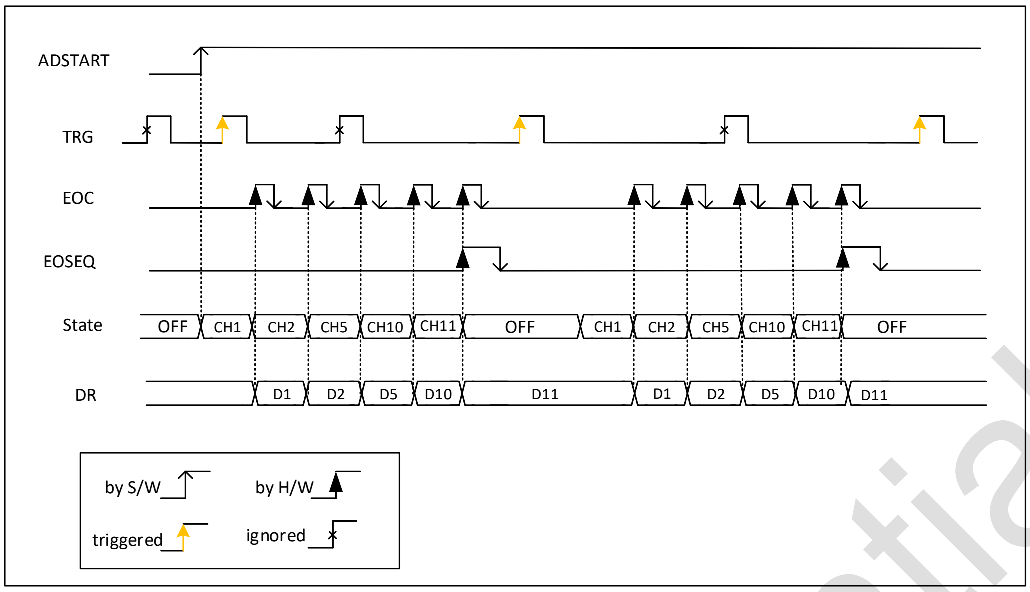


Figure 14-9 Single conversions of a sequence, hardware trigger

1. EXTSEL = TRGx, EXTEN = 0x1 (rising edge), CONT = 0
2. CHSEL = 0xF, SCANDIR = 0, AUTDLY = 0, AUTOFF = 0

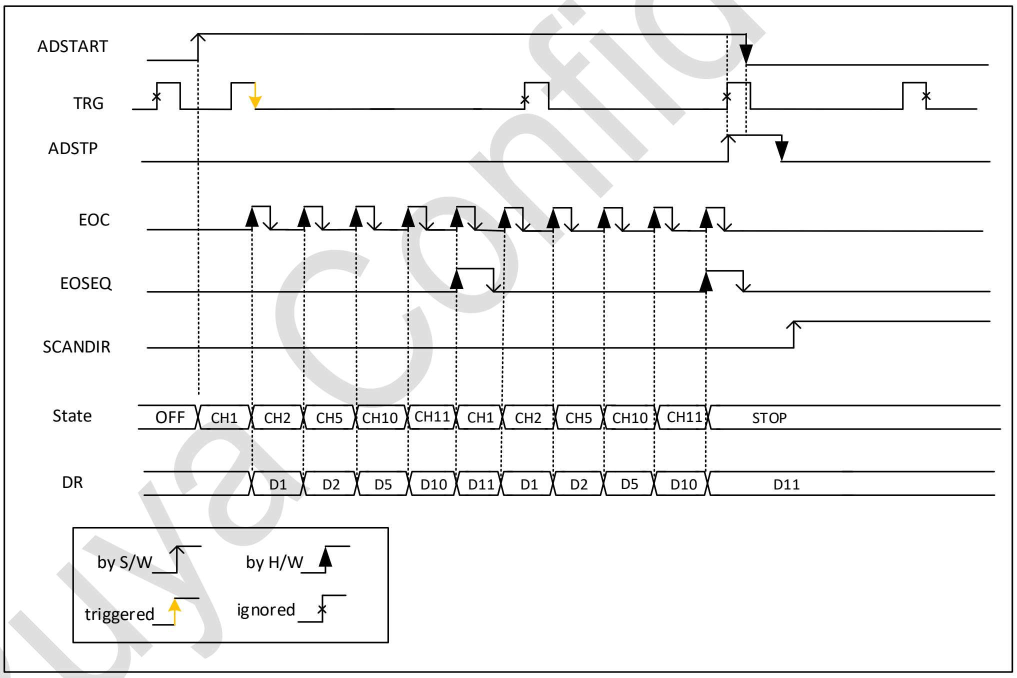


Figure 14-10 Continuous conversions of a sequence, hardware trigger

1. EXTSEL = TRGx, EXTEN = 0x2 (falling edge), CONT = 1
2. CHSEL = 0xF, SCANDIR = 0, WAIT = 0, AUTOFF = 0

14.5. Data management

14.5.1. Data register and data alignment (ADC_DR, ALIGN)

At the end of each conversion (when an EOC event occurs), the result of the converted data is stored in the ADC_DR data register which is 16-bit wide.

The format of the ADC_DR depends on the configured data alignment and resolution. The ALIGN bit in the ADC_CFGR1 register selects the alignment of the data stored after conversion. Data can be right-aligned (ALIGN=0) or left-aligned (ALIGN=1).

ALIGN	RESSEL	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0X0	0X0				DATA[11:0]											
	0X1	0X0				DATA[9:0]										0X0	
	0X2	0X0				DATA[7:0]								0x0			
	0X3	0X0				DATA[6:0]						0X0					
1	0X0	DATA[11:0]												0X0			
	0X1	DATA[9:0]										0X0		0X0			
	0X2	DATA[7:0]								0x0				0X0			
	0X3	DATA[6:0]						0X0						0X0			

14.5.2. ADC overrun (OVR, OVRMOD)

The overrun flag (OVR) indicates a data overrun event, when the converted data was not read in time by the CPU or the DMA, before the data from a new conversion is available.

If EOC is still '1', a new conversion has been completed at this time, then the CPU will set the OVR flag in the ADC _ ISR register, indicating that the ADC overshoot. An interrupt can be generated if the OVRIE bit is set in the ADC_IER register.

When an overshoot event occurs, the ADC will continue to operate and continue to transition unless the software decides to stop and reset this sequence transition. The software can set ADSTP in the ADC _ CR register to 1 to stop the ADC transition. The OVR flag can be cleared by software write 1.

It is possible to configure if the data is preserved or overwritten when an overrun event occurs by programming the OVRMOD bit in the ADC_CFGR1 register:

- OVRMOD=0

An overrun event preserves the data register from being overwritten: the old data is maintained, and the new conversion is discarded. If OVR remains at 1, further conversions can be performed, but the resulting data is discarded.

- OVRMOD=1

-The data register is overwritten with the last conversion result but previously unread data is lost, if OVR remains 1, the subsequent conversion is performed and the ADC _ DR register stores the last conversion result value.

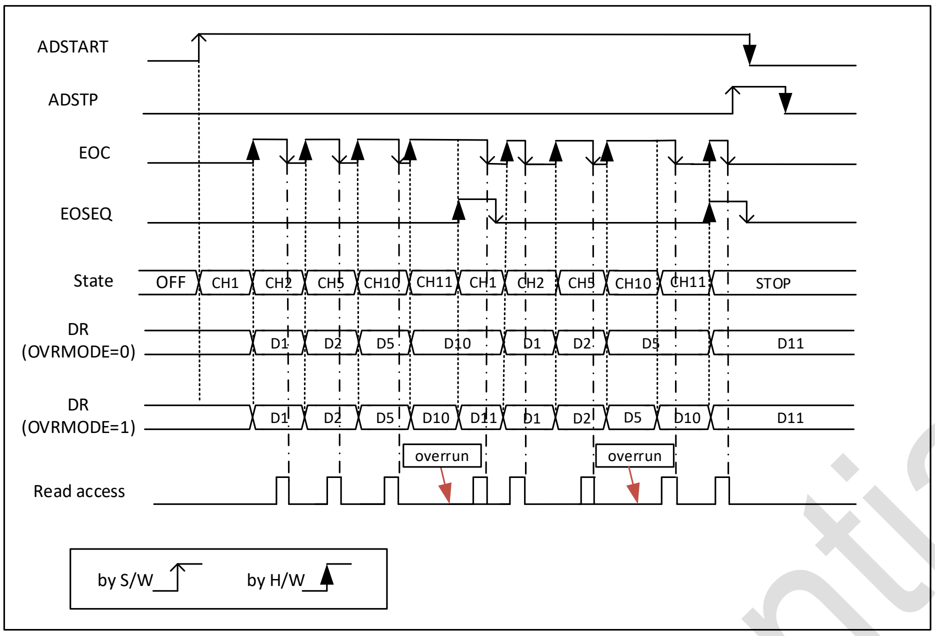


Figure 14-11 Overload

14.5.3. Managing conversion sequences without DMA

If the conversion of the ADC is slow enough, the conversion sequence can be controlled by software. In this case, the software applies the EOC flag and its associated interrupt to process each translation data. At the end of each conversion, at the EOC position bit in the ADC _ ISR register, the converted value of the ADC _ DR register can be read at this time. The OVRMOD bit in the ADC _ CFGR1 register may be configured to 0 to manage overshoot events.

14.5.4. Convert without DMA and overflow detection

There are applications that convert one or more channels without having to read each conversion result. In this case, the OVRMOD bit must be set to 1 and the software should ignore the OVR flag. When OVRMOD = 1, the overshoot event cannot prevent the ADC from continuing conversion and the data in the ADC _ DR register is always the last converted data.

14.6. Managing conversion sequences with DMA

Because the conversion result data of all channels is stored in a single data register, it is more efficient to use DMA mode when there are more than one conversion channel. This avoids loss of the conversion result stored in the ADC _ DR register. When DMA mode is on (DMAEN = 1 in the ADC _ CFGR1 register), a DMA request is generated at the end of each transition. This allows the translation data in the ADC _ DR register to be transferred to the target address specified by the software.

Nevertheless, in the event of an overshoot (OVR = 1) due to the failure of the DMA to serve the DMA request in time, the ADC stops generating the DMA request and as a result, the new conversion data is no longer transmitted by the DMA (when OVR = 0, the transmission is continued). It can also be considered that all data transferred to RAM is valid (because invalid data is never transferred again).

Depending on the configuration of the OVRMOD bit, the data in the ADC _ DR register can be selected as either Hold or Override.

The DMA transfer request is blocked until the software clears the OVR bit.

There are two different DMA modes, which depend on the configuration of the DMACFG bit in the ADC _ CFGR1 register:

- DMA one shot mode (DMACFG = 0)

This mode can be selected when DMA is programmed to transmit fixed length data.

- DMA cycle mode (DMACFG = 1)

This mode can be selected when the DMA is programmed to cycle mode.

DMA one shot mode (DMACFG = 0)

In this mode, the ADC generates a DMA request each time the converted data is valid. Once the DMA has reached the last DMA transmission, the ADC stops generating DMA requests even if the ADC conversion has started again. (When the DMA _ EOT interrupt occurs, the next ADC conversion may have started)

When the DMA transfer is complete (all transfers configured in the DMA controller have been completed):

- Contents of ADC data register freeze
- Any conversion in progress is terminated. And the result value is discarded
- No new DMA requests are issued to the DMA controller. This method avoids an ADC overshoot error if there is still an ADC conversion activation
- ADC scan sequence stops and resets
- DMA Stop

DMA circular mode (DMACFG = 1)

In this mode, even if the DMA reaches the last DMA transmission, the ADC will generate a DMA request each time the converted data is valid. This allows the DMA to be configured in a cyclic mode to process a continuous analog input data stream.

14.7. Low-power features

14.7.1. Wait mode conversion

Wait mode conversion can be used to simplify the software as well as optimizing the performance of applications clocked at low frequency where there might be a risk of ADC overrun occurring.

When the WAIT bit is set to 1 in the ADC_CFGR1 register, a new conversion can start only if the previous data has been treated, once the ADC_DR register has been read or if the EOC bit has been cleared. This is a way to automatically adapt the speed of the ADC to the speed of the system that reads the data.

Notes:

Any hardware triggers which occur while a conversion is ongoing or during the wait time preceding the read access are ignored.

2. After polling EOC = 1 in WAIT mode, you need to wait for 1 ADC _ CLK clock before reading DR.

3. It is not recommended to configure DMA in WAIT mode. If DMA is used, ADC _ CLK frequency division > 1/32 needs to be met (HCLK and PCLK are at the same frequency).

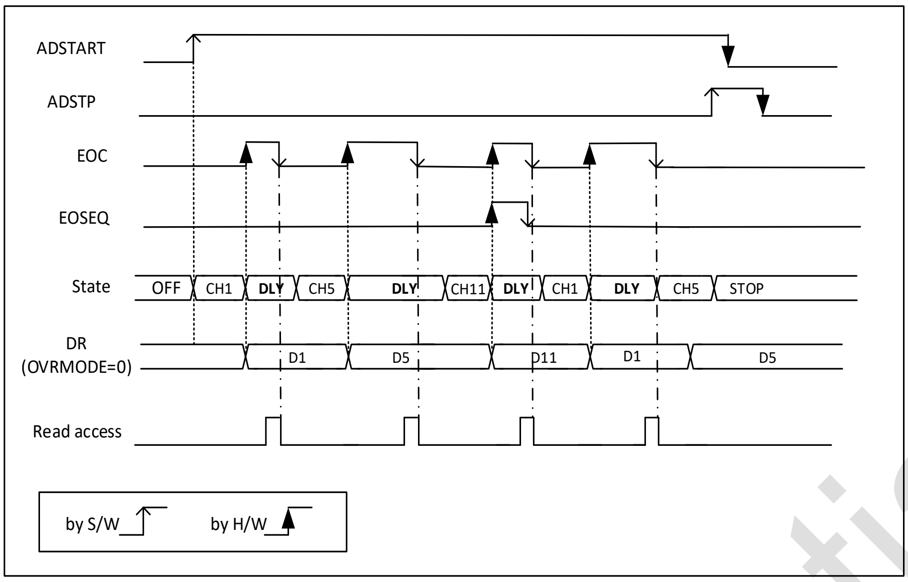


Figure14-12 Wait mode conversion

1. EXTEN = 0x0, CONT = 1
2. CHSEL = 0x3, SCANDIR = 0

14.8. Analog watchdog

The AWD analog watchdog feature is enabled by setting the AWDEN bit in the ADC_CFGR1 register. It is used to monitor that either one selected channel or all enabled channels.

The AWD analog watchdog status bit is set if the analog voltage converted by the ADC is below a lower threshold or above a higher threshold. The threshold values are programmed into the ADC _ HTR and ADC _ LTR 16-bit registers with up to 12 bits of valid data. An interrupt can be enabled by setting the AWDIE bit in the ADC_IER register. The AWD flag is cleared by software by writing 1 to it. When converting a data with a resolution of less than 12-bit (according to bits RES [1:0]), the LSB of the programmed thresholds must be kept cleared because the internal comparison is always performed on the full 12-bit raw converted data (left aligned).

Table 14-3 Analog watchdog comparison

Resolution digits	Analog watchdog comparison		Description
	Raw converted data, left aligned	Thresholds	
00: 12-bit	DATA[11:0]	LT[11:0] and HT[11:0]	
01: 10-bit	DATA[11:2],00	LT[11:0] and HT[11:0]	The user must configure LT[1:0] and HT[1:0] to “00”
10: 8-bit	DATA[11:4],0000	LT[11:0] and HT[11:0]	The user must configure LT[3:0] and HT[3:0] to “0000”
11: 6-bit	DATA[11:6],000000	LT[11:0] and HT[11:0]	The user must configure LT[5:0] and HT[5:0] to “000000”

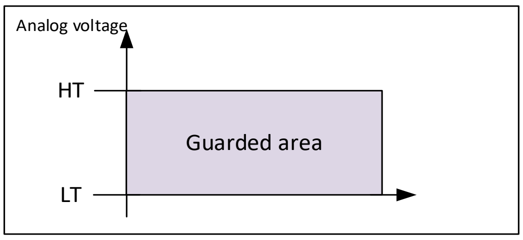


Figure 14-13 Analog watchdog guarded area

Table 14-4 Analog watchdog channel selection

Channels guarded by the analog watchdog	AWDSGL bit	AWDEN bit
None	x	0
All channels	0	1
Single channel	1	1

14.8.1. ADC_AWD_OUT signal output generation

Each analog watchdog is associated to an internal hardware signal ADC_AWD_OUT which is directly connected to the ETR input (external trigger) of some on-chip timers.

ADC_AWD_OUT is activated when the associated analog watchdog is enabled:

- ADC_AWD_OUT is set when a guarded conversion is outside the programmed thresholds.
- After the transition of the next channel selected by AWDCH is completed, ADC _ AWD _ OUT is reset within the programmed threshold. It remains at 1 if the next guarded conversions are still outside the programmed thresholds.
- ADC_AWD_OUT is also reset when disabling the ADC (when setting ADDIS to 1). Note that stopping conversions (ADSTP set to 1), might clear the ADC_AWDx_OUT state.
- A channel not selected as an analog watchdog does not affect the ADC _ AWD _ OUT status bit.

AWD flag is set by hardware and reset by software: AWD flag has no influence on the generation of ADC_AWD_OUT (ex: ADC_AWD_OUT can toggle while AWDx flag remains at 1 if the software did not clear the flag).

The ADC_AWD_OUT signal is generated by the PCLK domain.

The AWD comparison is performed at the end of each ADC conversion.

14.9. Temperature sensor and internal reference voltage

The temperature sensor can be used to measure the junction temperature (T_J) of the device.

The temperature sensor is internally connected to the ADC input channel which is used to convert the sensor′ s output voltage to a digital value. The sampling time for the temperature sensor analog pin must be greater than the minimum T_{S_temp} value specified in the datasheet. When not in use, the sensor can be put in power down mode.

The temperature sensor output voltage changes linearly with temperature, however its characteristics may vary significantly from chip to chip due to the process variations. To improve the accuracy of the temperature sensor, calibration values are individually measured for each part during production test and stored in the system memory area.

The internal voltage reference (V_{REFINT}) provides a stable voltage output for the ADC and Comparators.

Note: The TSVREF bit must be set to enable the internal channels: temperature sensor, V_{REFINT}.

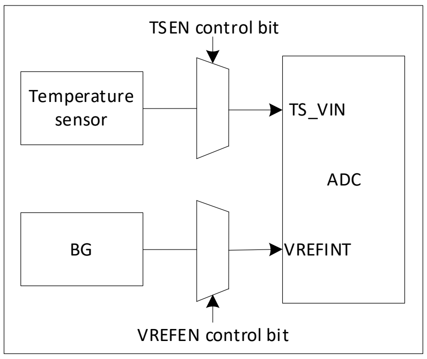


Figure 14-14 Temperature sensor and reference voltage channel

Read temperature

1. How to use a temperature sensor:
2. Select ADC1 _ IN10 input channel
3. Select an appropriate sampling time specified in the device datasheet (T_{S_temp}).
4. Set the TSEN bit in the ADC_CCR register to wake up the temperature sensor from power down mode and wait for its stabilization time.
5. Start the ADC conversion by setting the ADSTART bit in the ADC_CR register (or by external trigger).
6. Read VSENSE conversion data from ADC _ DR register

Calculate the actual temperature using the following formula（x6 version）:

$$Temperature(in\ ℃) = \frac{85℃ - 30℃}{TS_{CAL2} - TS_{CAL1}} \times (TS_{DATA} - TS_{CAL1}) + 30℃$$

TSCAL2 represents the calibration value of the 85 ℃ temperature sensor

TSCAL1 represents the calibration value of the 30 ℃ temperature sensor

Calculate the actual temperature using the following formula（x7 version）:

$$Temperature(in\ ℃) = \frac{105℃ - 30℃}{TS_{CAL2} - TS_{CAL1}} \times (TS_{DATA} - TS_{CAL1}) + 30℃$$

TSCAL2 represents the calibration value of the 105 ℃ temperature sensor

TSCAL1 represents the calibration value of the 30 ℃ temperature sensor

T_{S_DATA} is the actual temperature sensor output value converted by ADC

Note: The sensor has a startup time after waking from power down mode before it can output V_{SENSE} at the correct level. The ADC also has a startup time after power-on, so to minimize the delay, the ADEN and TSEN bits should be set at the same time.

Calculating the actual V_{CC} voltage using the internal reference voltage

The following formula can give the voltage value of the true VCC:

$$VREFINT = 1.2V = \frac{ADC\_DATAx}{4095} \times VCC$$

VREFINT is fixed at 1.2 V;

Convert the relative value of channel voltage measured by an ADC into absolute voltage value

The ADC gives a digital value based on the ratio of the voltage on the analog power input and conversion channel. Most applications need to convert this ratio into a voltage value. For the case where the VCC is known and the ADC conversion value is right-aligned, this absolute voltage value can be obtained by the following formula:

$$VCHANNEL = \frac{ADC\_DATAx}{4095} \times VCC$$

VCHANNEL is the channel voltage;

ADC _ DATA is the conversion data in ADC _ DR;

4096 is represented as 12 bits.

14.10. ADC interrupts

An interrupt can be generated by any of the following events:

- End of any conversion (EOC flag)
- End of conversion sequence (EOS flag)
- When an analog watchdog detection occurs (AWD flag)
- When the end of sampling phase occurs (EOSMP flag)
- When a data overrun occurs (OVR flag)

Separate interrupt enable bits are available for flexibility.

Table 14-5 ADC interrupts

Interrupt event	Event flag	Enable control bit
End of conversion	EOC	EOCIE
End of sequence conversion	EOS	EOSIE
Analog watchdog status bit is set	AWD	AWDIE
End of sampling phase	EOSMP	EOSMPIE
Overshoot	OVR	OVRIE

14.11. ADC registers

14.11.1. ADC interrupt and status register (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	Res	Res	Res	Res	Res	AWD	Res	Res	OVR	EOSEQ	EOC	EOSMP	Res.
								rc_w1			rc_w1	rc_w1	rc_w1	rc_w1	

Bit	Name	R/W	Reset Value	Function
31:8	Reserved			
7	AWD	RC_W1	0	Analog watchdog This bit is set by hardware when the converted voltage crosses the values programmed in the ADC_LTR and ADC_HTR registers. This bit is cleared by writing 1

Bit	Name	R/W	Reset Value	Function
				0: No analog watchdog event occurred (or the flag event was already cleared by software) 1: Analog watchdog event occurred
6:5	Reserved			
4	OVR	RC_W1	0	ADC overrun This bit is set by hardware when an overrun occurs. When the EOC flag is set, a new transition has been completed. It is cleared by software writing 1 to it. 0: No overrun occurred (or the flag event was already acknowledged and cleared by software) 1: Overrun has occurred
3	EOSEQ	RC_W1	0	End of sequence flag This bit is set by hardware at the end of the conversion of a sequence of channels selected by the CHSEL bits. This bit is cleared by writing 1 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 This bit is set by hardware at the end of each conversion of a channel when a new data result is available in the ADC_DR register. It is cleared by software writing 1 to it or by reading the ADC_DR register. 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	This bit is set by hardware during the conversion, at the end of the sampling phase. It is cleared by software by programming it to ‘1’. 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			

14.11.2. ADC interrupt enable register (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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWDIE	Res.	Res.	OVRIE	EOSE-QIE	EO-CIE	EOSMPIE	Res.
								rw			rw	rw	rw	rw	

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	AWDIE	RW	0	Analog watchdog interrupt enable This bit is set and cleared by software to enable/disable the analog watchdog interrupt. 0: Analog watchdog interrupt disabled 1: Analog watchdog interrupt enabled
6:5	Reserved			
4	OVRIE	RW	0	Overrun interrupt enable This bit is set and cleared by software to enable/disable the overrun interrupt. 0: Overrun interrupt disabled 1: Overrun interrupt enabled.
3	EOSEQIE	RW	0	End of conversion sequence interrupt enable This bit is set and cleared by software to enable/disable the end of sequence of conversions interrupt. 0: EOSEQ interrupt disabled 1: EOSEQ interrupt enabled.
2	EOCIE	RW	0	End of conversion interrupt enable

Bit	Name	R/W	Reset Value	Function
				This bit is set and cleared by software to enable/disable the end of conversion interrupt. 0: EOC interrupt disabled 1: EOC interrupt enabled.
1	EOSMPIE	RW	0	End of sampling flag interrupt enable This bit is set and cleared by software to enable/disable the end of sampling phase interrupt. 0: EOSMP interrupt disabled 1: EOSMP interrupt enabled.
0	Reserved	-	-	Reserved

Note: Software is allowed to write this bit only when ADSTART=0 (which ensures that no conversion is ongoing).

14.11.3. ADC control register (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
AD-CAL	Res														
rs															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								VREF-BUF_SEL		VREF-BUF_EN	AD-STP	MSBS EL	AD-START	AD-DIS	ADE N
								rw	rw	RW	rs	rw	rs	rs	rs

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RS	0	This bit is set by software to start the calibration of the ADC. It is cleared by hardware after calibration is complete. 0: Calibration complete 1: Write 1 to calibrate the ADC. Read at 1 means that a calibration is in progress. Note: The software cannot write 0 when writing 1, and the hardware is cleared.
30:8	Reserved	-	-	Reserved
7:6	VREFBUF_SEL	RW	0	VREFBUF module output voltage selection. 00: 1.024 V 01: 1.5 V 10: 2.048 V 11: 2.5 V
5	VREFBUF_EN	RW	0	VREFBUF enabled. 0: V_{REFBUF} disabled 1: V_{REFBUF} enabled
4	ADSTP	RS	0	ADC stop conversion command This bit is set by software to stop and discard an ongoing conversion (ADSTP Command). It is cleared by hardware when the conversion is effectively discarded and the ADC is ready to accept a new start conversion command. 0: No ADC stop conversion command ongoing 1: Write 1 to stop the ADC. Read 1 means that an ADSTP command is in progress. Software writing that the bit is 0 is an invalid operation.
3	MSBSEL	RW	0	Resolution highest bit conversion time control bit. For analog circuit debugging. 0: transition time is 1 TCLK; (default) 1: Conversion time is 2TCLK;
2	ADSTART	RS	0	ADC start command. This bit is set by software to start ADC conversion. Depending on the EXTEN [1:0] configuration bits, a conversion either starts immediately (software trigger configuration) or once a hardware trigger event occurs (hardware trigger configuration). If this bit is cleared by hardware:

Bit	Name	R/W	Reset Value	Function
				In single conversion mode (CONT=0, DISCEN=0), when software trigger is selected (EXTEN=00):at the assertion of the end of Conversion Sequence (EOSEQ) flag. In discontinuous conversion mode(CONT=0, DISCEN=1), when the software trigger is selected(EXTEN=00): at the assertion of the end of Conversion (EOC) flag. In all other cases: after the execution of the ADSTP command, at the same time as the ADSTP bit is cleared by hardware. 0: No ADC stop conversion command ongoing 1: Write 1 to start the ADC. Read 1 means that the ADC is operating and may be converting. Note: The software can only configure ADSTART = 1 if ADEN = 1 and ADDIS = 0. Software writing that the bit is 0 is an invalid operation.
1	ADDIS	RS	0	ADEN disabled. Software set disables ADC. When ADC is disabled (while ADEN is cleared by hardware), the hardware clears this bit. Software writing that the bit is 0 is an invalid operation. 0: No ADDIS command ongoing 1: Write 1 to disable the ADC. Read 1 means that an ADDIS command is in progress. Note: Setting ADDIS to ‘1’ is only effective when ADEN=1 and ADSTART=0 (which ensures that no conversion is ongoing)
0	ADEN	RS	0	ADC enable command Software sets this bit to enable the ADC and the ADC will be ready for operation. Software writing that the bit is 0 is an invalid operation. 0: Disable ADC (OFF state) 1: ADC is enabled

14.11.4. ADC configuration register 1 (ADC_CFGR1)

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.	AWDCH				Res	Res.	AWDEN	AWDSGL	Res.	Res	Res.	Res.	Res	DISCEN
		RW	RW	RW	RW			RW	RW						RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	WAIT	CONT	OVRMOD	Res.	Res.	Res.	EXTSEL			ALIGN	RES _SEL		SCANDIR	DMAFG	DMAEN
	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:26	AWDCH[3:0]	RW	0000	Analog watchdog channel selection. These bits are set and cleared by software. They select the input channel to be guarded by the analog watchdog. 0000: ADC analog input Channel 0 0001: ADC analog input Channel 1 … 1011: ADC analog input Channel 11 1100: ADC analog input Channel 12 1101: ADC analog input Channel 13 Others: Reserved *Description:* 1. The channel configured by the AWDCH [3: 0] bit also needs to be set to the CHSELR register, and the CHSELR register configuration channel 1 corresponding to AWDCH needs to be configured to 0, the CHSELR register configuration channel 2 corresponding to AWDCH needs to be configured to 1, and so

				on; CHSELR register configuration channel 14 corresponding to AWDCH needs to be configured to 13; 2. AWDCH cannot be configured for channel 0 corresponding to CHSELR, that is, channel 0 has no watchdog function; 3. Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing);
25:24	Reserved	-	-	Reserved
23	AWDEN	RW	0	Analog watchdog enable This bit is set and cleared by software. 0: Analog watchdog disabled 1: Analog watchdog enabled Note: Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
22	AWDSGL	RW	0	Enable the watchdog on a single channel or on all channels By setting and clearing this bit, the software can enable or disable the analog watchdog on the channel set by the AWDCH [3: 0] bit or on all channels. 0: Analog watchdog enabled on all channels 1: Enable analog watchdog on one channel (AWDCH [3: 0] configure which channel) Note: Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
21:17	Reserved	-	-	Reserved
16	DISCEN	RW	0	Discontinuous mode enabled. The software can set and clear this bit, enabling/disabling discontinuous mode. 0: Discontinuous mode disabled 1: Discontinuous mode enabled It is not possible to enable both discontinuous and continuous modes, i.e. disable setting DISCEN = 1 and CONT = 1. NOTE: The software is allowed to write these bits only if ADSTART = 0 (make sure there is no conversion in progress).
15	Reserved	-	-	Reserved
14	WAIT	RW	0	Waiting for conversion mode. This bit can be set and cleared by software, enabling/disabling waiting transition mode. 0: Wait for conversion mode to close 1: Wait for conversion mode to turn on Note: Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
13	CONT	RW	0	Single/continuous conversion mode. This bit is set and cleared by software. If set to 1, until the bit is cleared, otherwise the transition will continue to occur when a trigger occurs. It is not possible to have both discontinuous mode and continuous mode enabled: it is forbidden to set both bits DISCEN=1 and CONT=1. Note: Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
12	OVRMOD	RW	0	Overload management mode. The software can set and clear this bit to configure the way data overload is managed. 0: ADC_DR register is preserved with the old data when an overrun is detected. 1: When overload occurs, the ADC _ DR register will be overwritten by the last conversion result NOTE: The software is allowed to write these bits only if ADSTART = 0 (make sure there is no conversion in progress).
11:10	EXTEN[1:0]	RW	00	External drive enabling and polarity selection.

				The software can set and clear this bit, select the drive polarity and enable the drive. 00: Hardware driver detection is not enabled (software start transition) 01: Rising edge hardware driver detection 10: Falling edge hardware driver detection 11: Rising and falling edge hardware drive detection NOTE: The software is allowed to write these bits only if ADSTART = 0 (make sure there is no conversion in progress).
9	Reserved	-	-	Reserved
8:6	EXTSEL[2:0]	RW	000	External drive selection. This bit selects the external event that triggers the start of the transition. 000：TRG0(TIM1_TRGO) 001：TRG1(TIM1_CC4) 010：TRG2(TIM2_TRGO) 011 TRG3 (Reserved) 100 TRG4 (Reserved) 101 TRG5 (Reserved) 110 TRG6 (Reserved) 111：TRG7(EXTI11)
5	ALIGN	RW	0	Data alignment. The software sets and clears this bit to select right align or left align. 0: Right alignment 1: Left alignment Note: Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
4:3	RESSEL[1:0]	RW	00	Data resolution. The software sets this bit to select the conversion resolution. 00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit Note: These bits can be software manipulated only if ADEN = 0
2	SCANDIR	RW	0	Scan sequence direction. The software can set and clear this bit to select the scan sequence direction. 0: Up (from SQ0 to SQ14) 1: Down (from SQ14 to SQ0) NOTE: The software is allowed to write these bits only if ADSTART = 0 (make sure there is no conversion in progress).
1	DMACFG	RW		DMA access configuration. This bit can be set and cleared by software, selected in both DMA mode operations and valid at DMAEN = 1. 0: DMA single mode selection 1: DMA cycle mode selection NOTE: The software is allowed to write these bits only if ADSTART = 0 (make sure there is no conversion in progress).
0	DMAEN	RW	0	DMA access enabled. The software can set and clear this bit to enable the generation of DMA requests. Utilize DMA controller to manage automatic conversion data. 0: Disable DMA 1: Enable DMA

14.11.5. ADC configuration register 2 (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
CKMODE				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:28	CKMODE [3:0]:	RW	0000	ADC clock mode, the software can set and clear this bit, defining the clock source of the analog ADC. 0000: PCLK 0001：PCLK/2 0010：PCLK/4 0011：PCLK/8 0100：PCLK/16 0101：PCLK/32 0110：PCLK/64 1000: HSI 1001：HSI/2 1010：HSI/4 1011：HSI/8 1100：HSI/16 1101：HSI/32 1110：HSI/64 Others: ADCAL = 0, ADSTART = 0, ADSTP = 0 and ADEN = 0 only when the ADC is not enabled). The software is allowed to manipulate these bits Note: When selecting the HSI source, configure this register so that the ADC _ CLK frequency cannot be higher than the PCLK division of 2.
27:0	Reserved	-	-	Reserved

14.11.6. ADC sample time register 0 (ADC _ SMPR0)

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.	SMP9			SMP8			SMP7			SMP6			SMP5	
		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**
	SMP4			SMP3			SMP2			SMP1			SMP0		
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	-	-	Reserved
2:0	SMPx [2: 0]	RW	000	Sampling clock selection. The software can configure the sample time for this bit to select channel x. 000: 3.5 ADC clock cycles 001：5.5 ADC clock cycles 010: 7.5 ADC clock cycles 011：13.5 ADC clock cycles 100：28.5 ADC clock cycles 101：41.5 ADC clock cycles 110：134.5 ADC clock cycles

				111：239.5 ADC clock cycles The software is allowed to write these bits only if ADSTART = 0 (making sure there is no conversion in progress).

14.11.7. ADC sample time register 1 (ADC _ SMPR1)

Address offset: 0x18

Reset value: 0x0FFF 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.	SMP14			SMP13			SMP12			SMP11			SMP10		
	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	-	-	Reserved
2:0	SMPx [2: 0]	RW	000	Sampling clock selection. The software can configure the sample time for this bit to select channel x. 000: 3.5 ADC clock cycles 001：5.5 ADC clock cycles 010: 7.5 ADC clock cycles 011：13.5 ADC clock cycles 100：28.5 ADC clock cycles 101：41.5 ADC clock cycles 110：134.5 ADC clock cycles 111：239.5 ADC clock cycles The software is allowed to write these bits only if AD-START = 0 (making sure there is no conversion in progress).

14.11.8. ADC watchdog threshold register (ADC_TR)

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.	HT											
				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.	Res.	Res.	Res.	LT											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:16	HT[11:0]	RW	0xFFF	Analog watchdog higher threshold These bits are written by software to define the higher threshold for the analog watchdog. The software is allowed to write these bits only if AD-START = 0 (making sure there is no conversion in progress).
15:12	Reserved	-	-	Reserved
11:0	LT[11:0]	RW	0x000	Analog watchdog lower threshold These bits are written by software to define the lower threshold for the analog watchdog. The software is allowed to write these bits only if AD-START = 0 (making sure there is no conversion in progress).

14.11.9. ADC channel selection register (ADC_CHSELR)

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.	CHSEL14	CHSEL13	CHSEL12	CHSEL11	CHSEL10	CHSEL 9	CHSEL 8	CHSEL 7	CHSEL 6	CHSEL 5	CHSEL 4	CHSEL 3	CHSEL 2	CHSEL 1	CHSEL 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	-	-	Reserved
14:0	CHSELx	RW	0x0000	Channel selection is enabled (corresponding to SEQ0 ~ 14 configuration). These bits are written by software and define which channels are part of the sequence of channels to be converted. 0: Input channel x not selected 1: Select input channel x Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress) Note: When changing the conversion mode, you need to configure ADDIS = 1 to clear ADEN to synchronize the CHESEL generated by the hardware to be consistent with the configuration of this register.

14.11.10. ADC sequence select register 0 (ADC _ SEQR0)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SQ7 [3:0]				SQ6 [3:0]				SQ5 [3:0]				SQ4 [3:0]			
RW				RW				RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ3 [3:0]				SQ2 [3:0]				SQ1 [3:0]				SQ0 [3:0]			
RW				RW				RW				RW			

Bit	Name	R/W	Reset Value	Function
31:28	SQ7	RW	0	Channel selection. The software configures the values of these bits. The eighth transition in the regular sequence. These bits define the number (0-14) of the eighth transition channel in the transition sequence. See SQ0 definition for corresponding channels.
27:24	SQ6	RW	0	Channel selection. The software configures the values of these bits. The sev-enth transition in the regular sequence. These bits define the number (0-14) of the seventh transition channel in the transition sequence. See SQ0 definition for corresponding channels.
23：20	SQ5	RW	0	Channel selection. The software configures the values of these bits. The sixth transition in the regular sequence. These bits define the number (0-14) of the sixth transition channel in the transi-tion sequence. See SQ0 definition for corresponding channels.
19:16	SQ4	RW	0	Channel selection. The software configures the values of these bits. The fifth transition in the regular sequence. These bits define the number (0-14) of the fifth transition channel in the transition sequence. See SQ0 definition for corresponding channels.

15:12	SQ3	RW	0	Channel selection. The software configures the values of these bits. The 4th transition in the regular sequence. These bits define the number (0-14) of the 4th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
11:8	SQ2	RW	0	Channel selection. The software configures the values of these bits. The 3rd transition in the regular sequence. These bits define the number (0-14) of the 3rd transition channel in the transition sequence. See SQ0 definition for corresponding channels.
7:4	SQ1	RW	0	Channel selection. The software configures the values of these bits. The sec-ond transition in the regular sequence. These bits define the number (0-14) of the second transition channel in the transition sequence. See SQ0 definition for corresponding channels.
3:0	SQ0	RW	0	Channel selection. The software configures the values of these bits. The first transition in the regular sequence. These bits define the number (0-14) of the first transition channel in the transition sequence. The corresponding channel is as follows: 0000: External channel 0 0001: External channel 1 … 1000: external channel 8 1001: external passage 9 1010: internal TS 1011 Internal VREFINT 1100: internal VCCA/3 1101 Internal OPA1 output 1110 Internal OPA2 output Others: reserved

14.11.11. ADC sequence select register 1 (ADC _ SEQR1)

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	SQ14 [3:0]				SQ13 [3:0]				SQ12 [3:0]			
				RW				RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ11 [3:0]				SQ10 [3:0]				SQ9 [3:0]				SQ8 [3:0]			
RW				RW				RW				RW			

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:24	SQ14	RW	0	Channel selection. The software configures the values of these bits. The 15th transition in the regular sequence. These bits define the number (0-14) of the 15th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
23：20	SQ13	RW	0	Channel selection. The software configures the values of these bits. The 14th transition in the regular sequence. These bits define the number (0-14) of the 14th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
19:16	SQ12	RW	0	Channel selection. The software configures the values of these bits. The 13th transition in the regular sequence.

				These bits define the number (0-14) of the 13th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
15:12	SQ11	RW	0	Channel selection. The software configures the values of these bits. The 12th transition in the regular sequence. These bits define the number (0-14) of the 12th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
11:8	SQ10	RW	0	Channel selection. The software configures the values of these bits. The 11th transition in the regular sequence. These bits define the number (0-14) of the 11th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
7:4	SQ9	RW	0	Channel selection. The software configures the values of these bits. The 10th transition in the regular sequence. These bits define the number (0-14) of the 10th transition channel in the transition sequence. See SQ0 definition for corresponding channels.
3:0	SQ8	RW	0	Channel selection. The software configures the values of these bits. The ninth transition in the regular sequence. These bits define the number (0-14) of the ninth transition channel in the transition sequence. See SQ0 definition for corresponding channels.

14.11.12. ADC data register (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
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**
DATA[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	-	-	Reserved
15:0	DATA[15:0]	R	0	Transform data. This bit is read-only. The conversion result of the last converted channel is placed in this register. The data are left- or right-aligned.

14.11.13. ADC calibration configuration and status register (ADC_CCSR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CALON.	CAP-SUC	OFFSUC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R	RC_W1	RC_W1													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALSET	CAL-BYP	CALSMP		CALSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW		RW											

Bit	Name	R/W	Reset Value	Function
31	CALON	R	0	Calibration flag, indicates that ADC calibration is in progress. 1: ADC calibration in progress

Bit	Name	R/W	Reset Value	Function
				0: ADC calibration has ended or has not started
30	CAPSUC	RC_W1	0	Capacitance calibration status bit. Indicates whether the ADC capacitance calibration is successful. Hardware set 1; The software writes 1 and sets 0; CALON = 0, CALSEL = 0, CAPSUC = 1: Invalid status CALON = 0, CALSEL = 0, CAPSUC = 0: CAPs calibration was not performed CALON = 0, CALSEL = 1, CAPSUC = 1: ADC CAPs calibrated successfully CALON = 0, CALSEL = 1, CAPSUC = 0: ADC CAPs calibration failed Note: The capacitance calibration is only calibrated once, regardless of whether the C11 ~ C6 calibration is successful or not.

29	OFFSUC	RC_W1	1' b0	Offset calibration status bit.
				Indicates whether the ADC offset calibration was successful. Hardware set 1; The software writes 1 and sets 0;
				CALON = 0, CALSEL = 0, OFFSUC = 0: ADC OFFSET Calibration failed
				CALON = 0, CALSEL = 0, OFFSUC = 1: ADC OFFSET calibration successful
				CALON = 0, CALSEL = 1, OFFSUC = 1: ADC OFFSET Calibration successful
				CALON = 0, CALSEL = 1, OFFSUC = 0: ADC OFFSET Calibration failed

28:16	Reserved	-	-	Reserved
15	CALSET	RS	0	Calibration factor selection Software set (before ADCAL = 0, i.e. before calibration), hardware cleared. 1: Set CAL _ CXIN data as final calibration data 0: Close the path from CAL _ CXIN to CAL _ CXOUT, and select the result generated internally by the calibration circuit.
14	CALBYP	RS	0	Calibration factor bypass Software set (before ADCAL = 0, i.e. before calibration), hardware cleared. 1: Mask automatic calibration and set the output result of CALSET calibration to CAL _ CXOUT 0: Select automatic calibration or set the output result of CALSET calibration to CAL _ CXOUT
13:12	CALSMP	RW	0	Calibration sample time selection Configure the number of clock cycles in the sampling phase of calibration based on the following information: 00: 1 ADC clock cycles 01： 2 ADC clock cycles 10： 4 ADC clock cycles 11： 8 ADC clock cycles The longer the SMP is configured during calibration, the more accurate the calibration result is, but this configuration will bring the problem of longer calibration cycle
11	CALSEL	RW	0	Calibration content selection bit to select what needs to be calibrated 1: Calibrate OFFSET and Linearity 0: Calibrate OFFSET only
10:0	Reserved	-	-	Reserved

14.11.14. ADC common configuration register (ADC_CCR)

Address offset: 0x308

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.	TSVREFIN-TEN	Res.	Res.	Res.	Res.	Res.	Res.	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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	TSVREFINTEN	RW	0	Temperature sensor and VREFINT enable bit, which the software can set and clear to enable/disenable the temperature sensor or VREFINT. 0: Disabled 1: Enabled Allow software to write these bits only if ADSTART = 0 (make sure there is no conversion in progress)
21:0	Reserved	-	-	Reserved

15. Comparator (COMP)

15.1. Introduction

The device integrates two general-purpose comparators (COMP), namely COMP1 and COMP2.

The two modules can be used as separate modules or in combination with timer.

Comparators can be used as：

- Triggered by analog signal to wake-up function from low-power mode
- Analog signal conditioning
- Cycle by cycle current control loop when comparators are connected with PWM output from timer.

15.2. COMP main features

- Each comparator has configurable positive or negative input for flexible voltage selection:
  - Multiple I/O pins
  - Power supply VCC
  - Temperature sensor The output of
  - Supports 64-step voltage division of internal reference voltages VREFBUF and VCCA
  - OPA output as INP input
- Configurable hysteresis function
- Programmable speed and consumption
- The output can be triggered by a connection to the I/O or timer input
  - OCREF_CLR event (cycle by cycle current control)
  - Brakes for fast PWM shutdown
  - timer IC Input
- COMP1 and COMP2 comparators can be combined in a window comparator.
- Each COMP has interrupt generation capability and is used as a wake-up of the chip from low power modes (Sleep and Stop modes) (via EXTI)
- Configurable digital filter

15.3. COMP functional description

15.3.1. COMP block diagram

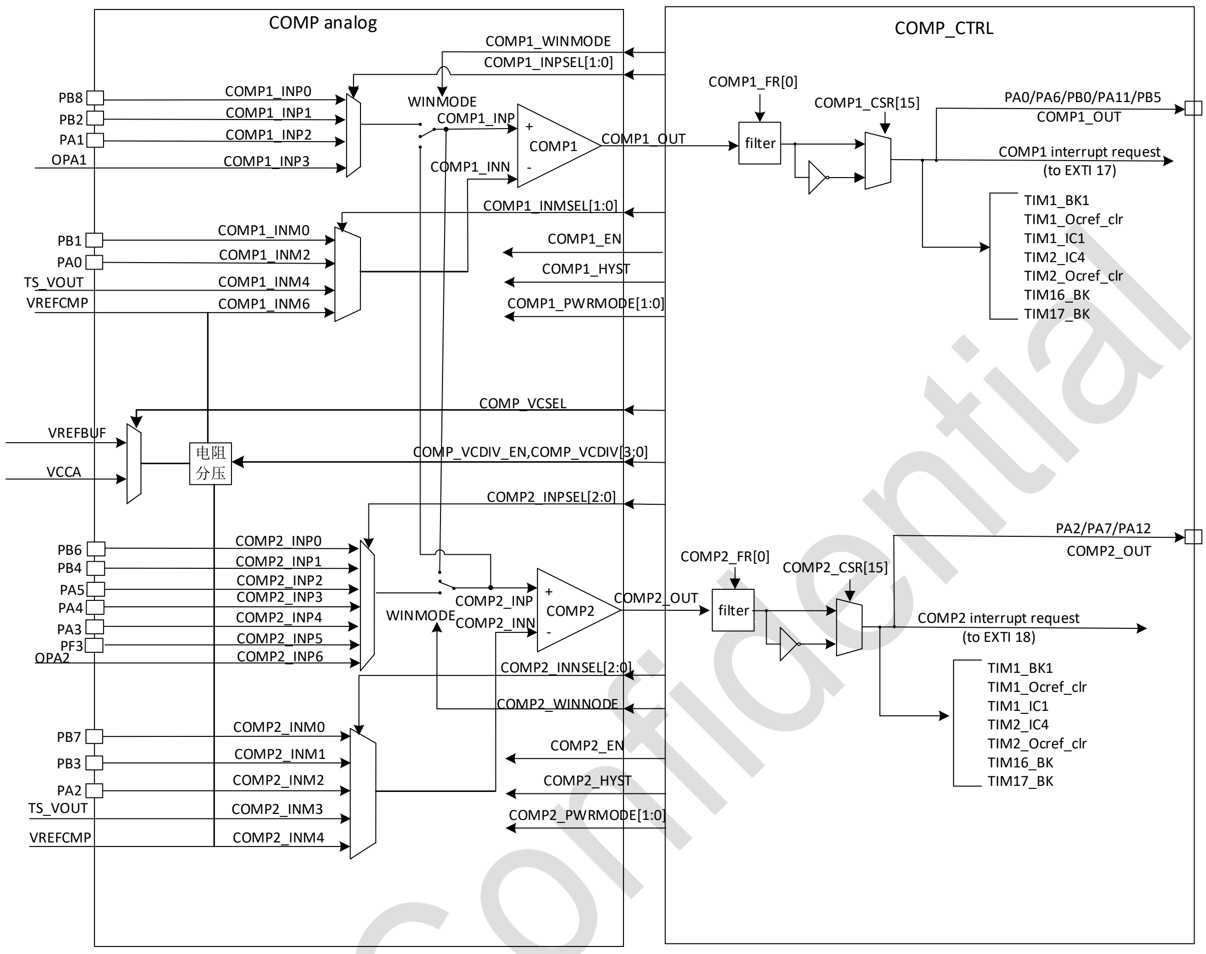


Figure 15-1 Comparator architecture diagram

15.3.2. COMP pins and internal signals

I/O used as input to the comparator must be configured in analog mode in the GPIO register and the corresponding PAD in the SYSCFG.P * _ ANA2EN register needs to be enabled.

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 of PWM signal when brake input is connected
- OCREF_CLR event (cycle by cycle current control)
- Input capture for timing measures

15.3.3. COMP reset and clocks

The COMP module has two clock sources:

1) PCLK (APB clock), system bus clock
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 you need to work in stop mode, select LSE or LSI.

The reset signal sources of the COMP module are:

1. APB reset, system reset

2. Reset of circuits after analog comparator output (latch circuit of analog output, glitch filter circuit, etc.). The reset signal includes APB reset source and COMP module software reset source (RCC _ APBRSTR2.COMP1RST and RCC _ APBRSTR2.COMP2RST)

15.3.4. Comparator lock device

The comparator can be used for safety purposes, such as overcurrent and temperature protection. For applications with specific functional security requirements, it is necessary to ensure that the programming of the comparator cannot be overwritten in the event 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 Lock bit is set to 1, which makes the entire register read-only, including the COMPx Lock bit.

The write protection can only be reset by the reset signal of the chip.

15.3.5. Window comparator

The role of the Window comparator is to monitor whether the analog voltage is within the low and high threshold ranges.

You can create a window comparator using two comparators. The monitored analog voltage is simultaneously connected to the non-inverting (+ terminal) inputs of both comparators, and the high threshold and the low threshold are connected to the inverting inputs (-terminal) of both comparators, respectively.

By enabling the WINMODE bit, the non-inverting (+ input terminals) of the two comparators can be connected together to save an I/O pin.

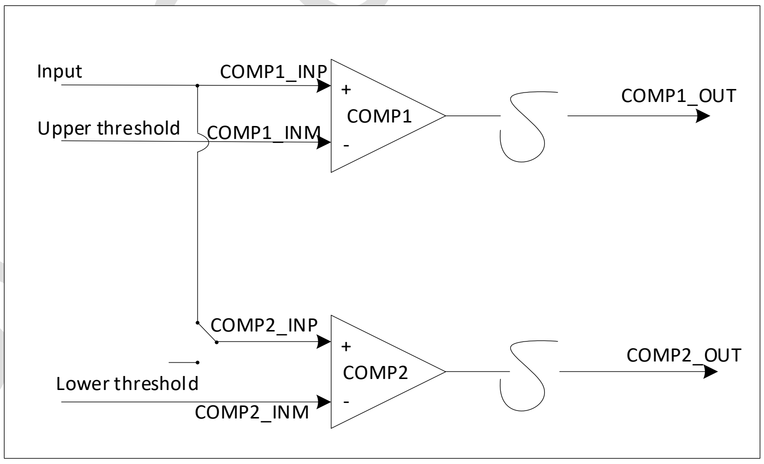


Figure 15-2 Window comparator

15.3.6. Hysteresis

In order to avoid false output transitions in the case of noisy signals, the comparator can enable a function with hysteresis (by enabling the HYST bits of COMP1 _ CSR and COMP2 _ CSR, the hysteresis functions of COMP1 and COMP2, respectively, can be turned on).

15.3.7. Low-power modes

The power consumption and transmission delay of the comparator can be adjusted to achieve the most suitable trade-off for a specific application. The PWRMODE [1: 0] bit of the COMPx _ CSR register may be used as an option for this function.

Note that before entering stop, if you select the PWR _ CR2 register LPR = 2 'b01 (i.e. choose to power with low power regulator), you need to first set COMP at Medium speed (PWRMODE = 01). And the COMP1 _ CSR.VCSEL register cannot be configured to 0 (select VREFBUF). In addition, if the PWR _ CR2 register LPR = 2 'b10 is selected, then comparator wake-up is not supported (the analog PMU turns off the analog quantity of the comparator).

In addition, to reduce power consumption, the APB clock and the COMP clock are controlled by RCC _ APBENR2.COMP1EN (and RCC _ APBENR2.COMP2EN), and the software can only enable this register when using the COMP module.

15.3.8. Comparator filtering

The output filtering function of COMP and the corresponding filter width can be enabled by setting the COMP _ FR register. Note that this setting should be completed before COMP _ EN is enabled.

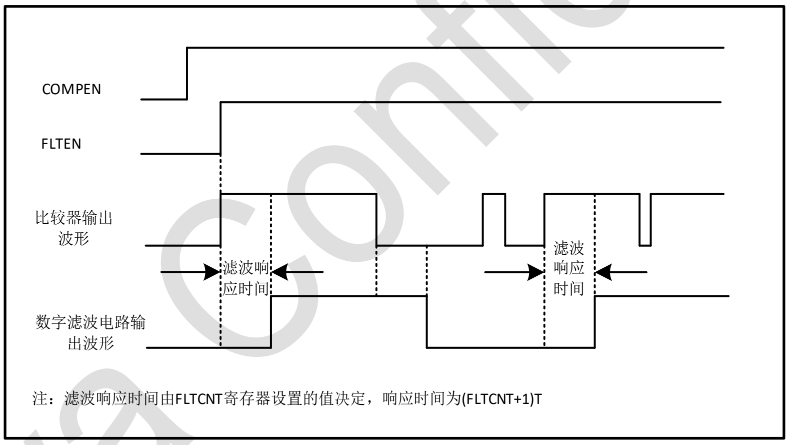


Figure 15-3 comparator filtering

15.3.9. COMP interrupts

The comparator outputs are internally connected to the Extended interrupts and events controller. Each comparator has its own EXTI line （17 and 18） and can generate either interrupts or events. The same mechanism is used to exit from low-power modes.

15.4. COMP registers

15.4.1. Comparator 1 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	COMP_OUT	Res.	Res.	COMP_VCSEL	COMP VCDIV _ en	COMP _ VCDIV				Res.	Res.	PWR-MODE		Res.	HYST

RW	R			RW	RW	R W	R W	R W	R W			R W	R W		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLAR-ITY	Res.	Re s.	Re s.	WINMODE	Res.	INPSEL [1; 0]		INMSEL[2:0]			Re s.	Re s.	Re s.	Re s.	COM P1 _EN
RW		-	-	RW	-	R W	R W	R W	R W	R W					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	COMP_OUT	R		COMP1 output This read-only flag reflects the level of the comparator 1 output before the polarity selector.
29:28	Reserved	-	-	Reserved
27	VCSEL	RW	0	COMP1, COMP2 Reference voltage Vref Select 0: VREFBUF 1: VCC Note: When configuring VREFBUF, you need to configure ADC _ CR.VREFBUF _ EN = 1 first
26	VCDIV _ en	RW	0	COMP1, COMP2 Partial Voltage Enable 1: Enable 0: Disenable
25:20	VCDIV	RW	0	COMP1, COMP2 partial pressure selection 000000: 1/64 Vref000001: 2/64 Vref000010: 3/64 Vref... 111110: 63/64 Vref111111: Vref
19:18	PWRMODE [1: 0]	RW	0	COMP1 power consumption mode selection The software is readable and writable, with selected power consumption and consequent speed of COMP1 00: High speed 01: Medium speed 10: High speed 11: High speed Note: This bit is not controlled by the LOCK function.
17	Reserved	-	-	Reserved
16	HYST	RW	0	COMP1 hysteresis function enable control 0: Hysteresis function turned off 1: Hysteresis function enabled
15	POLARITY	RW	0	COMP1 polarity selection Software readable and writable (if not locked) 0: Non-inverted 1: Inverted
14:12	Reserved	-	-	Reserved
11	WINMODE	RW	0	COMP1 does not reverse output selection (window mode) Software readable and writable (if not locked) 0: Signal selected by INPSEL [1: 0] of COMP1 1: COMP2 _ INP signal of COMP2 Note that the WINMODE mode of both COMPs cannot be enabled simultaneously.
10	Reserved			
9:8	INPSEL[1:0]	RW	00	COMP1 does not reverse input signal selection, software is readable and writeable (if not locked) 00: INP0, corresponding to PAD PB8 01: INP1, corresponding to PAD PB2 10: INP2, corresponding to PAD PA1 11: INP3, corresponding to OPA1 _ VOUT inside chip
7:5	INMSEL[1:0]	RW	00	000: INM0, corresponding to PAD PB1 010: INM2, corresponding to PAD PA0 100: INM4, corresponding to TS _ VOUT inside chip 110: INM6, corresponding to VREFCMP inside chip others: reserved
4:1	Reserved	-	-	Reserved
0	COMP1_EN	RW	0	Comparator 1 enable Software readable and writable (if not locked)

Bit	Name	R/W	Reset Value	Function
				0: Disable 1: Enable

15.4.2. Comparator 1 filtering 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 [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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1	RW	0x0	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	-	-	Reserved
0	FLTEN1	RW	0x0	Comparator 1 digital filter function configuration 0: Disable digital filtering function 1: Enable digital filtering function Note: This bit must be set when COMP1 _ EN is 0

15.4.3. 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	COMP_OUT	Res.	Res.	Res.	Res.	Res.				Res.	Res.	PWR-MODE		Res.	HYST
RW	R											RW	RW		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLAR-ITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL [1; 0]		INMSEL[2:0]			Res.	Res.	Res.	Res.	COMP2 _EN
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	COMP_OUT	R		COMP2 output This read-only flag reflects the level of the comparator 2 output before the polarity selector.
29:20	Reserved	-	-	Reserved
19:18	PWRMODE [1: 0]	RW	0	COMP2 power consumption mode selection The software is readable and writable, with selected power consumption and consequent speed of COMP2 00: High speed 01: Medium speed 10: High speed 11: High speed Note: This bit is not controlled by the LOCK function.
17	Reserved	-	-	Reserved
16	HYST	RW	0	COMP2 hysteresis function enable control 0: Hysteresis function turned off 1: Hysteresis function enabled
15	POLARITY	RW	0	COMP2 polarity selection

Bit	Name	R/W	Reset Value	Function
				Software readable and writable (if not locked) 0: Non-inverted 1: Inverted
14:12	Reserved	-	-	Reserved
11	WINMODE	RW	0	COMP2 does not reverse output selection (window mode) Software readable and writable (if not locked) 0: Signal selected by INPSEL [1: 0] of COMP2 1: COMP1 _ INP signal of COMP1 Note that the WINMODE mode of both COMPs cannot be enabled simultaneously.
10:8	INPSEL[2:0]	RW	000	COMP2 does not reverse input signal selection, soft-ware is readable and writeable (if not locked) 000: INP0, corresponding to PAD PB6 001: INP1, corresponding to PAD PB4 010: INP2, corresponding to PAD PA5 011: INP3, corresponding to PAD PA4 100: INP4, corresponding to PAD PA3 101: INP5 corresponding to PAD PF3 110: INP6, corresponding to OPA2 _ VOUT inside chip
7:5	INMSEL[2:0]	RW	000	000: INM0, corresponding to PAD PB7 001: INM1, corresponding to PAD PB3 010: INM2, corresponding to PAD PA2 011: INM3, corresponding to TS _ VOUT inside chip 100: INM4, corresponding to VREFCMP inside chip others: Reserved
4:1	Reserved	-	-	Reserved
0	COMP2_EN	RW	0	Comparator 2 enable Software readable and writable (if not locked) 0: Disable 1: Enable

15.4.4. Comparator 2 filtering 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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2	RW	0x0	Comparator 2 Sampling Filter Counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of sam-ples reaches the filter count value, the results are output consistently. Sample Count Period = FLTCNT [15: 0]
15:1	Reserved	-	-	Reserved
0	FLTEN2	RW	0x0	Comparator 2 digital filter function configuration 0: Disable digital filtering function 1: Enable digital filtering function Note: This bit must be set when COMP2 _ EN is 0

16. Operational amplifier (OPA)

16.1. Introduction

The OPA module is suitable for simple amplifier applications. The two internal OPA can be cascaded using external resistors. The input range of OPA is 0V to AVCC and the output range is 0.1 V to AVCC-0. 2V.

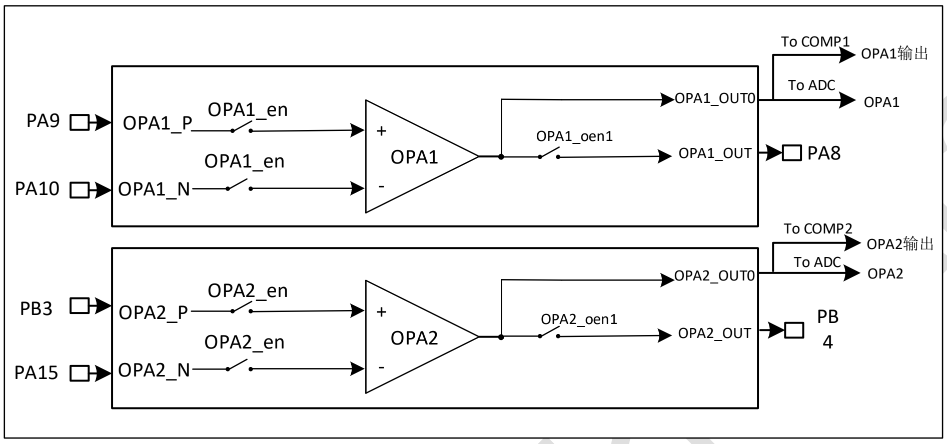


Figure 16-1 OPA architecture block diagram

16.2. OPA main features

- Two independently configured operational amplifier
- The input range of OPA is 0 to VCCA, and the output range is 0.1 V to VCCA-0. 2V programmable gain
- Can be configured as universal operational amp mode

16.3. OPA functional description

OPA can amplify a small-signal analog input signal by using external components to form an amplifier, and the output is an amplified signal.

16.4. OPA registers

16.4.1. OPA output enable register (OPA _ OENR)

Address: 0x30

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	OPA2OEN	Res	Res	Res	Res	OPA1OEN	Res
									RW					RW	

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	OPA2OEN	RW	0	OPA2 output enabled, high active
5:2	Reserved	-	-	Reserved
1	OPA1OEN	RW	0	OPA1 output enabled, high active

16.4.2. OPA control register (OPA _ CR)

Address offset: 0x34

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	OPA2EN	OPA1EN	Res	Res	Res	Res	Res
									RW	RW					

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6	OPA2EN	RW	0	OPA2 enabled, high active
5	OPA1EN	RW	0	OPA1 enabled, high active
4:0	Reserved	-	-	Reserved

17. Liquid crystal display (LCD) controller

17.1. Introduction

The LCD controller is a digital controller/driver for monochrome passive liquid crystal displays (LCD), with up to 8 common terminals (COM) and 18 segment terminals (SEG) to drive 72 (4 * 18) or 122 (8 * 14) LCD pixels. The exact number of terminals depends on the device pins described in the data manual. The LCD is composed of several segments (pixels or complete symbols), which can all be lit or off. Each segment contains a layer of liquid crystal molecules aligned between the two electrodes. When a voltage above the threshold voltage is applied to the liquid crystal, the corresponding segment is visible. The segment voltage must be AC to avoid electrophoretic effects in the liquid crystal (which will affect the display effect). After that, waveforms must be generated across the section to avoid DC.

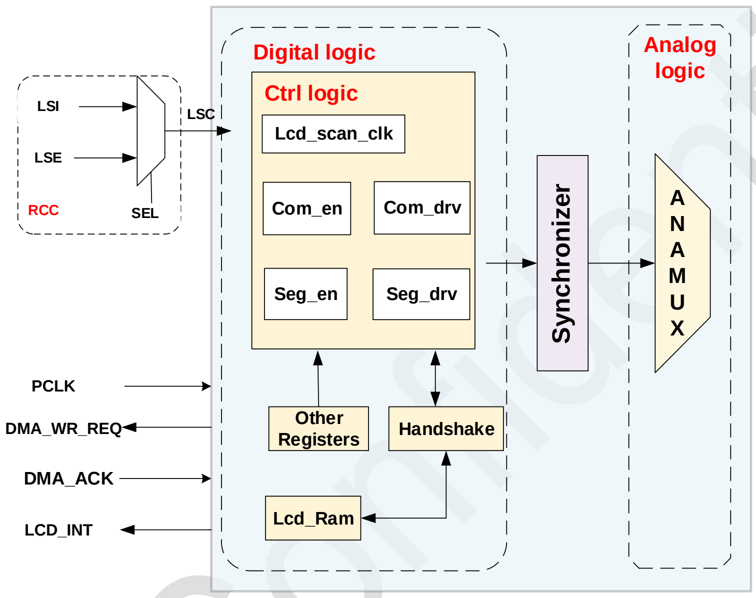


Figure 17-1 LCD control block diagram

17.2. LCD main features

- Highly flexible frame rate control
- Support static, 1/2, 1/3, 1/4, 1/6, and 1/8 of a duty ratio
- Supports static, 1/2, 1/3 bias voltage.
- Up to 16 registers LCD data RAM
- By software configuration of LCD contrast
- 3 kinds of waveform generation
  - Internal resistance divider, external resistance divider
  - By way of internal resistance of the software configuration partial pressure power consumption, so as to match the capacitance charge needed for the LCD panel
- Support low power consumption mode: LCD controller can display in run, Sleep, Stop modes.
- Configurable frame interrupts.
- Support LCD flashing function and configuration of multiple flicker frequency configuration
- Unused LCD segments and public pin can be configured to digital or analog functions

17.3. LCD function description

17.3.1. LCD function description

For the I/O of the LCD, it must be configured in analog mode in the GPIO register, and it is necessary to enable the PAD mapped by the LCD in the SYSCFG.P * _ ANA2EN register.

The specific functions of the LCD module are as follows:

- An external low-speed crystal oscillator or an internal low-speed RC clock can be selected as the working clock (generated by the RCC module)
- Generating a com port switch waveform according to the LCD clock;
- Generates the original waveform of the segment switch based on the current register settings
- The LCD RAM is used to store the display data of the LCD, and can also be used as a general data register
- After the data in the LCD RAM is rewritten, the data displayed on the LCD changes

17.3.2. LCD clock

The input clock is the LSC (32kHz) output by RCC, and the frequency division coefficient is selected according to the lcd scanning frequency. When the value of the counter is 0, the clock signal is pulled up; When counting to half of the frequency division coefficient value, pulling the clock signal low; After counting to overflow, the counter is reset to zero and the clock signal is pulled up again.

17.3.3. LCD flushing frequency

The LCDCR0.LCD _ FRAME register can select the LCD refresh frequency, that is, the transmission time of each frame. There are the following four frequencies to choose from:

- 00: 64 Hz
- 01: 128 Hz
- 10: 256 Hz
- 11: 512 Hz

When the clock (external low-speed crystal oscillator/internal low-speed RC, 32.768 kHz) is input, the frequency is divided according to the selected frequency.

17.3.4. LCD display mode

In different display modes, the available space of the LCD display data storage register (LCDRAM) is as follows:

- Maximum storage space of 14 × 8 bits in 1/8 duty cycle mode
- Maximum storage space of 16 × 6 bits in 1/6 duty cycle mode
- Maximum storage space of 18 × 4 bits in 1/4 duty cycle mode
- In 1/3 duty cycle mode, the maximum storage space is 18 × 3 bits
- In 1/2 duty cycle mode, the maximum storage space is 18 × 2 bits
- In static display mode, the maximum storage space is 18 × 1 bit

Mode 1

The data of the LCDRAM is read out synchronously with the common signal and output to the segment pin. For example, in the 1/2 duty cycle mode, when COM0 is active, ram0 bit0 to 17 of the

LCDRAM are read together and assigned to SEG0 to SEG17 respectively; When COM1 is active, the LCDRAM ram0 bit 18-31 and ram1 bit 0-3 are read together and assigned to SEG0 to SEG17, respectively. The SEG corresponding to writing "1" in LCDRAM will be displayed on the screen, and the SEG corresponding to writing "0" in LCDRAM will not be displayed.

After the LCD controller starts to work, it will not be interfered by the CPU. Pins that are not used as LCD functions can be used as general-purpose input and output ports. At this time, LCDRAM can be used as ordinary data registers without use.

The figure below is the relationship diagram between common/segment and LCDRAM bits under different duty cycle conditions. Each LCDRAM corresponds to a SEG. Taking the 1/8 duty cycle mode as an example, all bits of LCDRAM00 to LCDRAM03 and bits 0 to 15 of LCDRAM04 are valid.

Mode 0

The data of the LCDRAM is read out synchronously with the common signal and output to the segment pin. For example, in the 1/8 duty cycle mode, when COM0 is active, ram0-3 bit0, 8, 16, 24 and ram4 bit0, 8 of the LCDRAM are read together and assigned to SEG0 to SEG17, respectively; When COM1 is active, ram0-3 bits 1, 9, 10, and 23 and ram4 bits 1, 9 of the LCDRAM are read together and assigned to SEG0 to SEG17, respectively. The SEG corresponding to writing "1" in LCDRAM will be displayed on the screen, and the SEG corresponding to writing "0" in LCDRAM will not be displayed.

After the LCD controller starts to work, it will not be interfered by the CPU. Pins that are not used as LCD functions can be used as general-purpose input and output ports. At this time, LCDRAM can be used as ordinary data registers without use.

The figure below is the relationship diagram between common/segment and LCDRAM bits under different duty cycle conditions. Each LCDRAM corresponds to a SEG. Taking the 1/8 duty cycle mode as an example, all bits of LCDRAM00 to LCDRAM03 and bits 0 to 15 of LCDRAM04 are valid.

17.3.5. LCD bias circuit

The Bias voltage of the LCD has three sources: internal resistor division and external resistor division. When the internal resistor voltage divider is selected, the chip automatically switches the internal circuit to produce a voltage that complies with Bias and Duty. When choosing external resistor voltage division or external capacitor voltage division, users need to build related circuits on the peripheral pins of the chip.

Internal Mode

The internal resistance modes VLCDH, VLCD1 to VLCD3 can be used as an LCD SEG output or an IO port.

In internal resistance mode, the driving voltage of the LCD is controlled by CR0. Contrast, as shown in the following table:

Table 17-1 Internal resistance mode

Cr0.contrast	VLCD (1/3 bias)	VLCD (1/2 bias)
0	1.00 * VCC	1.00 * VCC
1	0.94* VCC	0.92* VCC
2	0.9 * VCC	0.85 * VCC
3	0.85* VCC	0.8* VCC
4	0.81 * VCC	0.75 * VCC
5	0.78 * VCC	0.7 * VCC
6	0.75 * VCC	0.66 * VCC
7	0.72 * VCC	0.63 * VCC
8	0.70 * VCC	0.61 * VCC
9	0.67 * VCC	0.58 * VCC
10	0.65 * VCC	0.55 * VCC
11	0.63 * VCC	0.53 * VCC
12	0.61 * VCC	0.51 * VCC
13	0.59 * VCC	0.48 * VCC
14	0.57 * VCC	0.47 * VCC
15	0.55 * VCC	0.45 * VCC

External Mode

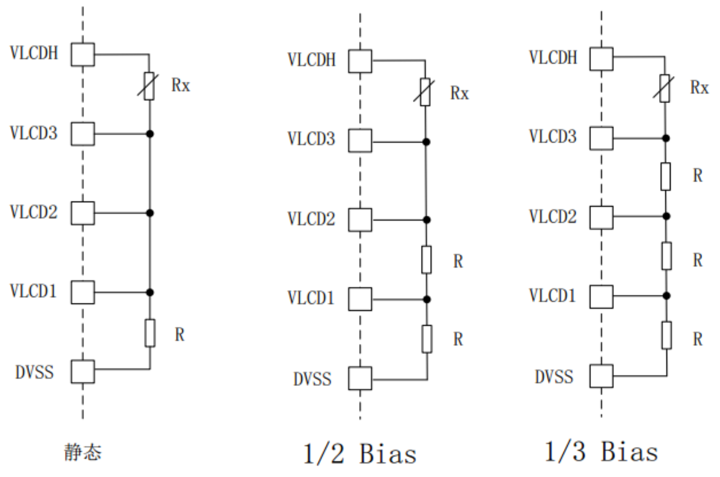


Figure 17-2 External resistance mode

17.3.6. DMA mode

Working with DMA can be activated by setting the DMA _ EN bit of LCD _ CR1. The steps to assign a DMA channel to the reception of the LCD are as follows (x represents the channel number):

- Select LCD on the DMAMUX control register to activate a DMA channel.
- The LCDRAM address is configured as the destination address of transmission through the DMA control register, and the value of the memory address is regarded as the source address. After each INTF event, data will be transferred from the memory to the corresponding LCDRAM address.
- Configure the total number of bytes to be transferred to the DMA control register.
- Configure the channel priority in the DMA register

- Depending on the requirements of the application, configure whether the DMA interrupt is generated when the transfer is half or all completed.

When the transmission amount specified by the DMA controller is received, the DMA controller generates an interrupt on the interrupt vector of the DMA channel.

17.3.7. LCD interrupt

Frame interrupt flag (LCD _ INTF): When the interrupt enable bit is turned on (IE = 1), this flag is set after the end of single frame transmission. At this time, the LCD can send a DMA request.

Interrupt generation condition: After all the data of a single frame is read in, and after the data of the last state machine has been read in, the hardware sets the frame interrupt flag.

17.4. LCD register

17.4.1. LCD configuration register 0 (LCD_CR0)

Address offset: 0x00

Reset value: 0x00C2

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**
CONTRAST[3:0]				BSEL[2:0]			DUTY[3:0]			BIAS	RES		LCD _ CLK [1: 0]		EN
RW				RW			RW			RW	RES		RW		RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	Contrast	RW	000	LCD contrast adjustment Note: Only valid if the Bias voltage source selects internal resistor voltage divider. The larger the Constrast value, the smaller the amplitude of the LCD waveform. At 0X0, the LCD waveform has the largest amplitude and the largest contrast ratio; …… At 0XF, the LCD waveform has the smallest amplitude and the smallest contrast ratio;
11:9	BSEL	RW	000	Bias Voltage Source Selection 111 Reserved 110 internal resistor voltage division, large power consumption mode 101 Reserved 100 internal resistor voltage division, low power consumption mode 011 Reserved 010: Internal resistor voltage division, medium power consumption mode 000: External resistance mode, requires external circuit cooperation
8:6	DUTY	RW	011	LCD duty Configuration 000: static 001: 1/2 duty 010: 1/3 duty 011: 1/4 duty 100: Reserved 101: 1/6 duty 110: Reserved 111: 1/8 duty
5	BIAS	RW	0	0: 1/3 bias (initial value) 1: 1/2 bias
4:3	Reserved	-	-	Reserved
2:1	LCDCLK	RW	2'b01	LCD scanning frequency selection

				00: 64 Hz 01: 128 Hz 10: 256 Hz 11: 512 Hz Note: LCD frame frequency = LCD scanning frequency × Duty
0	EN	RW	0	LCD enable control 1: Enabled 0: Disabled

17.4.2. LCD configuration register 1 (LCD _ CR1)

Address offset: 0x04

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**
Res				INTF	DMA EN	IE	MODE	Res	BLIN KE	BLINKCNT [5: 0]					
RES				RO	RW	RW	RW	RW	RW	RW					

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	INTF	RO	0	LCD interrupt flag 1: Interrupt 0: No interruption
10	DMAEN	RW	0	DMA hardware trigger enable 1: Enable LCD interrupt trigger DMA 0: Disable LCD interrupt trigger DMA
9	IE	RW	0	Interrupt enable 1: Enabled 0: Disabled
8	MODE	RW	0	LCD RAM Display Mode Selection 0: Mode 0 1: Mode 1
7	Reserved	-	-	Reserved
6	BLINKEN	RW	0	LCD splash screen configuration 1: Enabled 0: Disabled
5:0	BLINKCNT	RW	0	Splash screen frequency and LCD interruption interval setting Note: LCD blinking frequency is = LCD frame fre-quency/(BlinkCnt+1) LCD interrupt interval = (BlinkCnt+1) * (1/LCD frame fre-quency)

17.4.3. Interrupt clear register (LCD _ INTCLR)

Address offset: 0x08

Reset value: 0x0400

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					INTF _ CLR	Res									
RW					RW	RW									

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	INTF _ CLR	R1W0	1	Interrupt flag clear, write 0 clear, write 1 invalid
9:0	Reserved	-	-	Reserved

17.4.4. Output configuration register (LCD _ POEN0)

Address offset: 0x0C

Reset value: 0xFFFFFFF

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
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	Res	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13:0	Sx	RW	0x3FFF	Segx output control bit 0: SEG output enabled 1: SEG output is off, other functions can be used, such as IO, analog input and output

17.4.5. Output configuration register 1 (LCD _ POEN1)

Address offset: 0x10

Reset value: 0x1FFF

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	c3	c2	c1	c0	S14 c7	S15 c6	S16 c5	S17 c4	Res	Res	Res	Res
				RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:8	Cx	RW	F	COMx output control bit (com0 _ com3) 0: COM output enabled 1: COM output is off, other functions can be used, such as IO, analog input and output
7:4	SxCy	RW	F	Segx/COMy output control bit 0: SEG14 ~ 17/COM7 ~ 4 output enabled 1: SEG14 ~ 17/COM7 ~ 4 outputs are off, other functions can be used, such as IO, analog input and output SEG COM pin function selection is determined by CR0. DUTY
3:0	Reserved	-	-	Reserved

17.4.6. LCD_RAM0~3

Address offset: 0x14 ~ 0x20

Reset value: 0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
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**
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
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	Dx	RW	-	LCD dot output, displaying reference LCD display mode The SEG COM crossing point corresponding to 0 is not lit; 1 corresponding SEG COM cross-illumination;

Note: When configuring LCD _ RAMx, ensure that it is completed within 2 LCD clocks (LSI or LSE). And the interval between the two writes satisfies 2 LCD clocks.

17.4.7. LCD_RAM4

Address offset: 0x24

Reset value: 0x00

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**
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
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	Dx	RW	-	LCD dot output, displaying reference LCD display mode The SEG COM crossing point corresponding to 0 is not lit; 1 corresponding SEG COM cross-illumination;

Note: When configuring LCD _ RAMx, ensure that it is completed within 2 LCD clocks (LSI or LSE). And the interval between the two writes satisfies 2 LCD clocks.

18. Advanced-control timers (TIM1)

18.1. Introduction

The advanced-control timer (TIM1) is consist of a 16-bit auto-reload counter driven by a programmable prescaler. It can be used in various scenarios, including pulse length measurement of input signals (input capture) or generating output waveforms (output compare, output PWM, complementary PWM with dead-time insertion).

The pulse length and waveform period can be modulated from microseconds to milliseconds using a timer divider and an RCC clock control divider. The advanced (TIM1) and generic (TIMx) timers are completely independent and do not share any resources. They can sync up.

18.2. TIM1 main features

- 16-bit up, down, up/down auto-reload counter
- 16-bit programmable frequency divider that allows the clock frequency of the counter to be divided from 1 to 65535 on the fly
- Up to 4 independent channels
  - Input capture
  - Output Compare
  - PWM generation (edge or center-aligned mode)
  - One-pulse mode output
  - Retriggerable single-pulse mode output
- Complementary outputs with programmable dead time
- Synchronization circuit using external signals to control timers and interconnect timers
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- Output signal of the timer can be set as reset and know state by break input.
- Interrupt/DMA occurs on the following events
  - Update: Counter overflow (up/down), counter initialization (via software or internal/external trigger)
  - Trigger event
  - Input capture
  - Output Compare
  - Break input
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

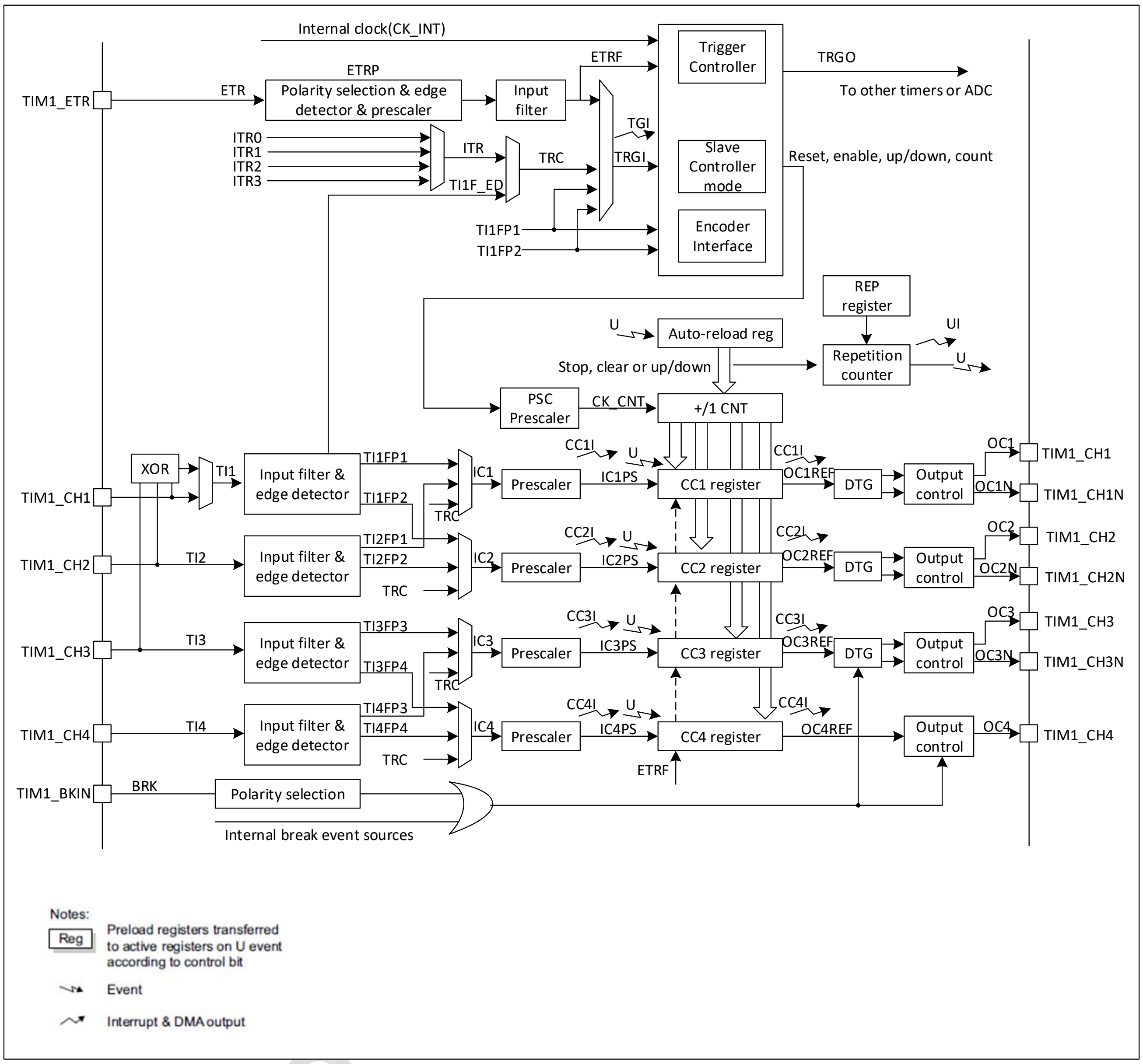


Figure 18-1 Advanced-control timer

18.3. TIM1 functional description

18.3.1. 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 counter clock can 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 includes:

- Counter register (TIM1_CNT)
- Prescaler register (TIM1_PSC)
- Auto-reload register (TIM1_ARR)
- Repetition counter register (TIM1_RCR)

The auto-load register is preloaded. Writing to or reading from the auto-load register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM1_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM1_CR register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

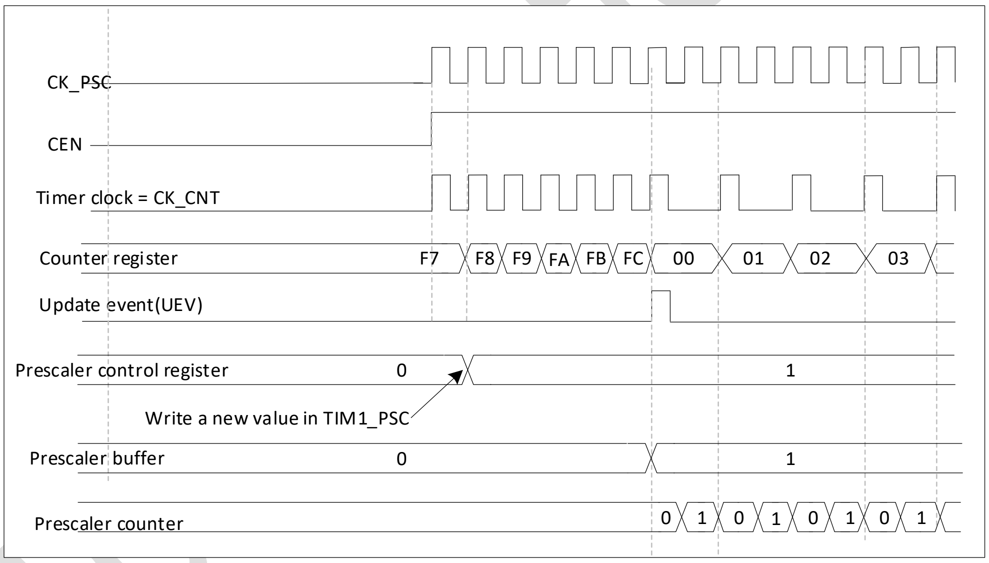


Figure 18-2 Counter timing diagram with prescaler division change from 1 to 2

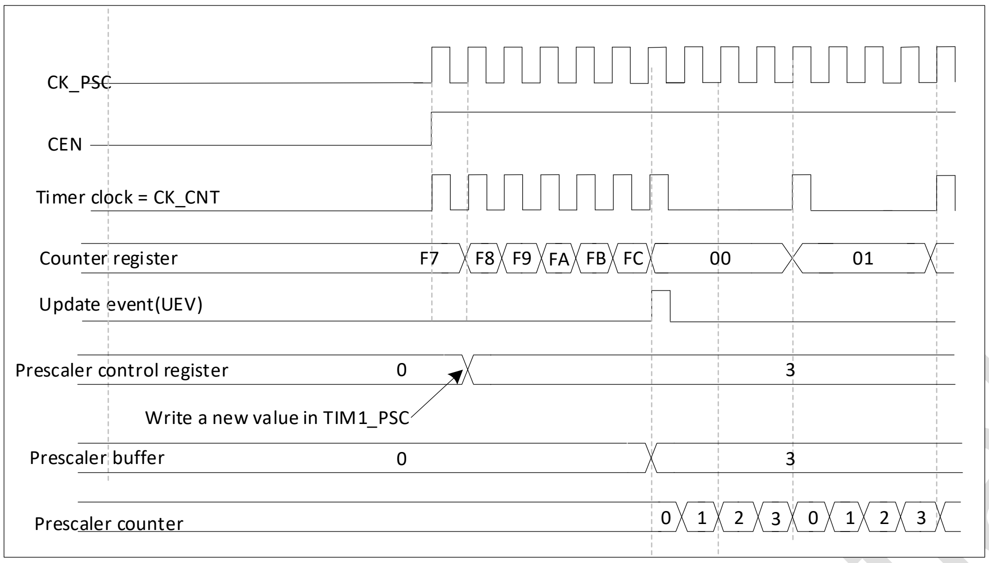


Figure 18-3 Counter timing diagram with prescaler division change from 1 to 4

18.3.2. Timer enable

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value, then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register. Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates 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 the UDIS bit has been written to 0. Even then, when an update event should occur, the counter will still be cleared '0' and the count of the prescaler will be called 0 (but the value of the prescaler will not change). In addition, if the URS bit in the TIMx _ CR1 register is set (Select Update Request), setting the UG bit will generate an update event UEV, but the hard-ware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on 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.

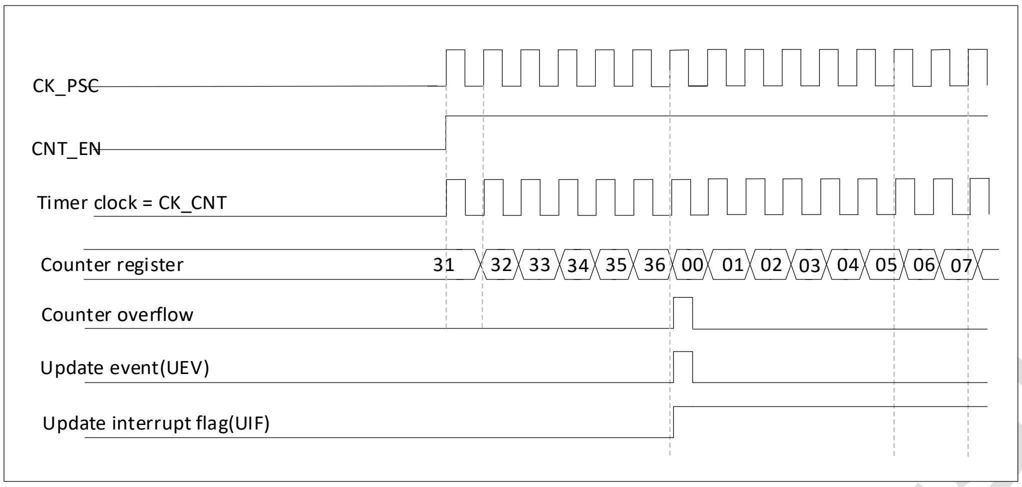


Figure 18-4 Counter timing diagram, internal clock divided by 1

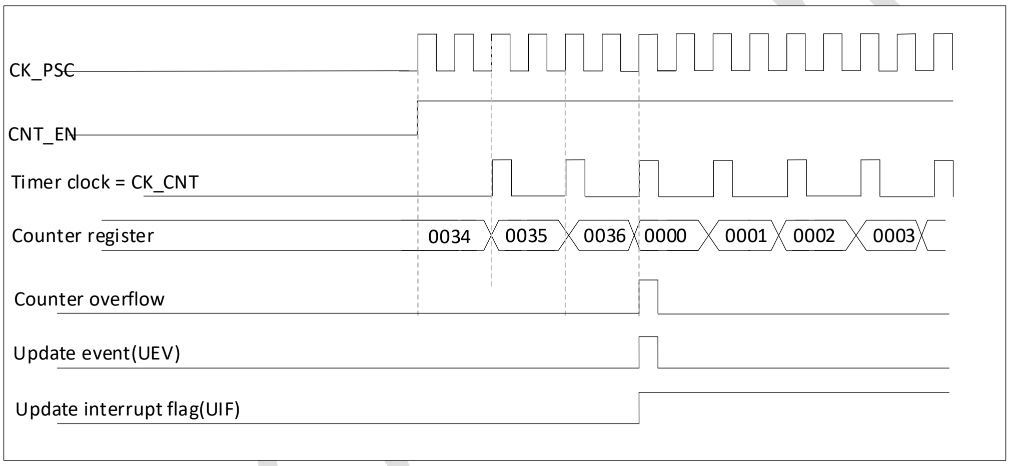


Figure 18-5 Counter timing diagram, internal clock divided by 2

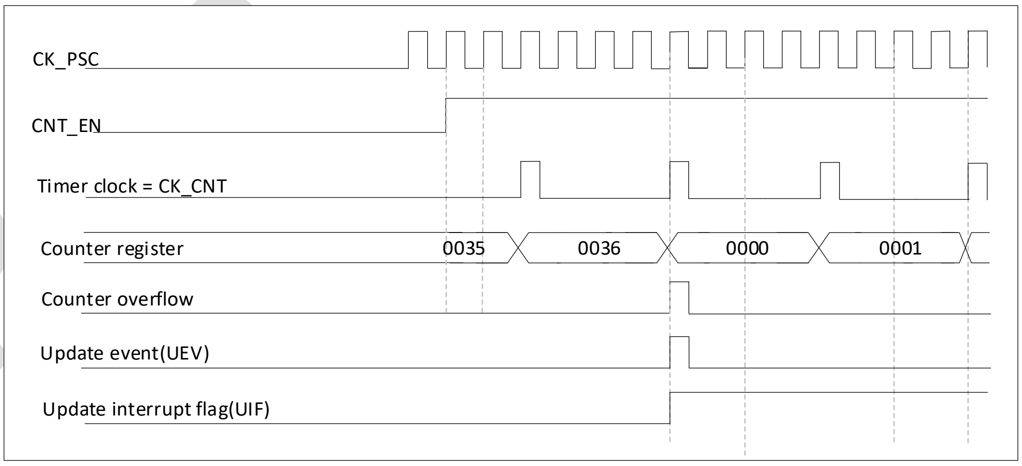


Figure 18-6 Counter timing diagram, internal clock divided by 4

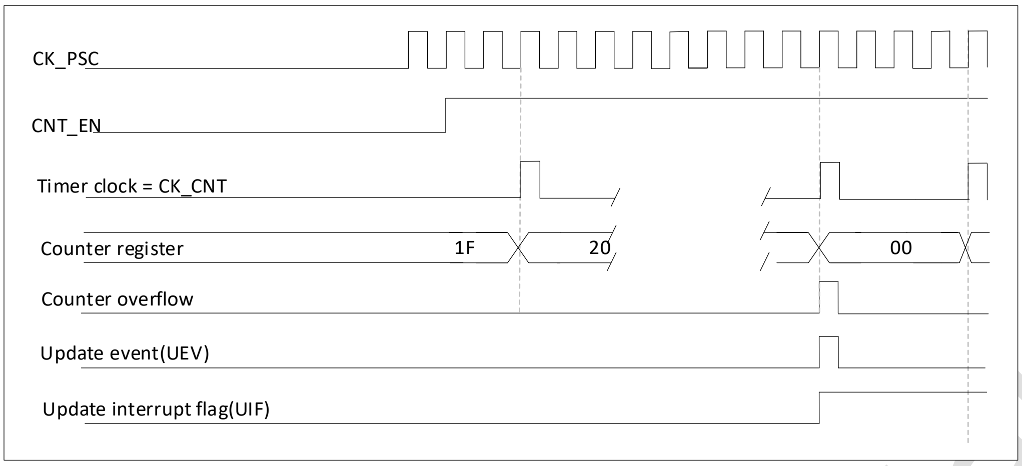


Figure 18-7 Counter timing diagram, internal clock divided by N

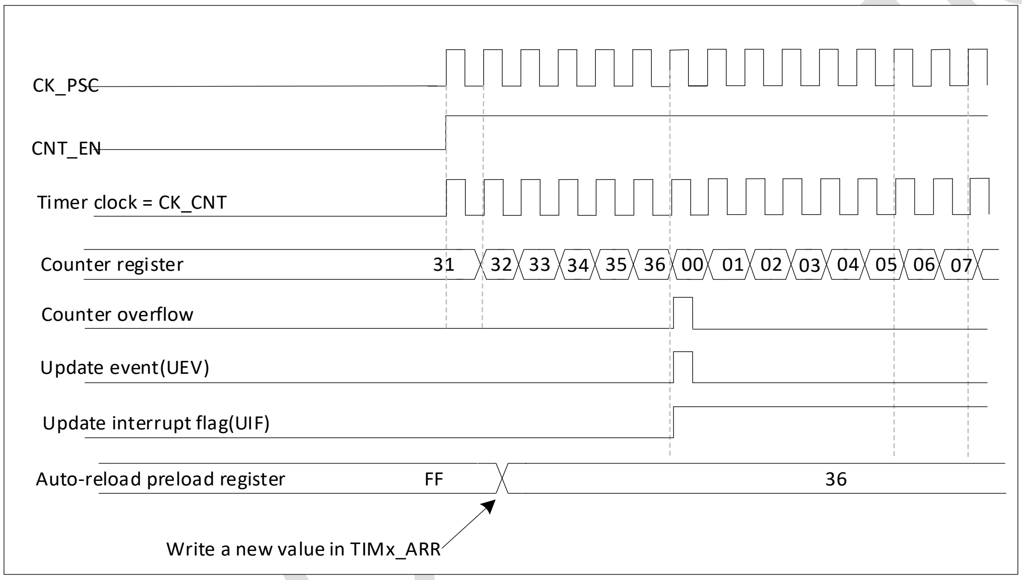


Figure 18-8 Counter timing diagram, update event when ARPE=0 (TIM1_ARR not preloaded)

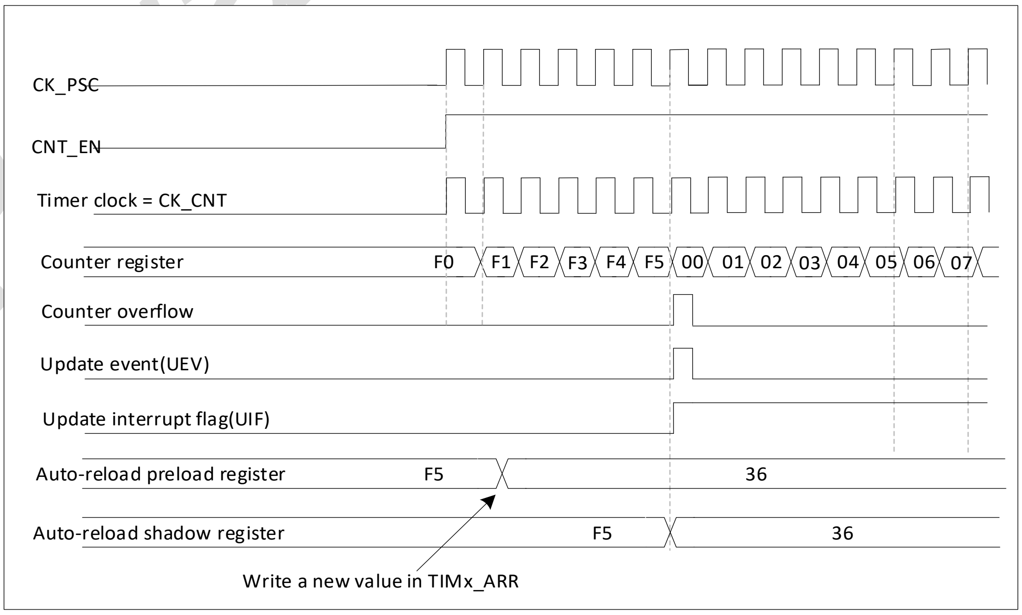


Figure 18-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times programmed in the repetition counter register(TIMx_RCR). Else the update event is generated at each counter underflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. 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. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescaler rate doesn’t change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and DMA requests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- 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 current autoload register is updated to the preload value (the contents of the TIMx _ ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

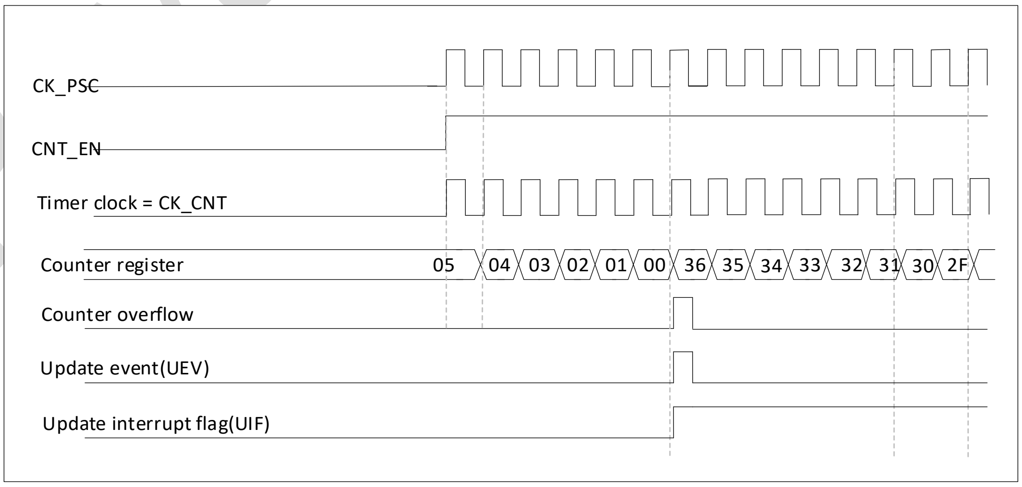


Figure 18-10 Counter timing diagram, internal clock divided by 1

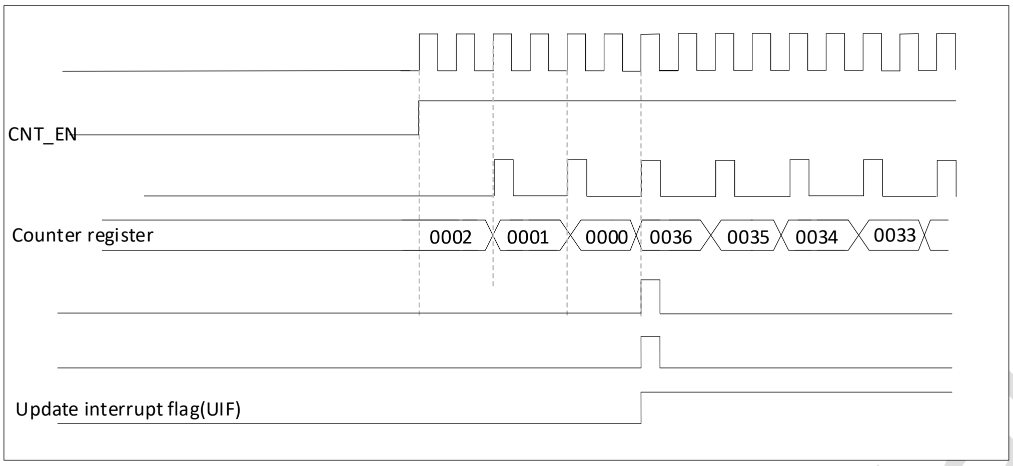


Figure 18-11 Counter timing diagram, internal clock divided by 2

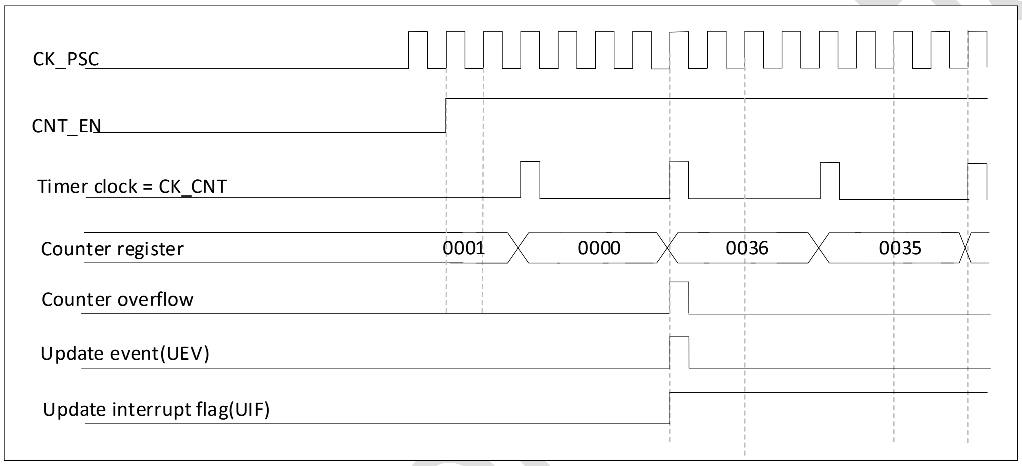


Figure 18-12 Counter timing diagram, internal clock divided by 4

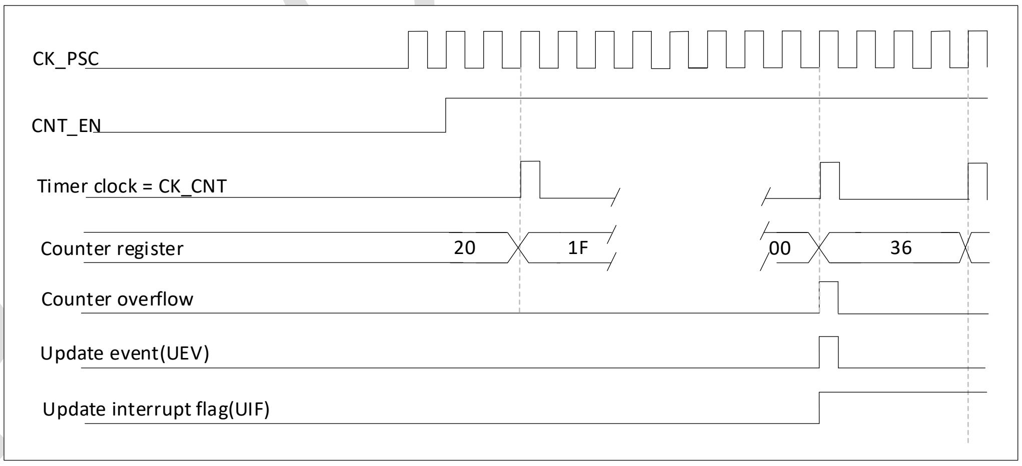


Figure 18-13 Counter timing diagram, internal clock divided by N

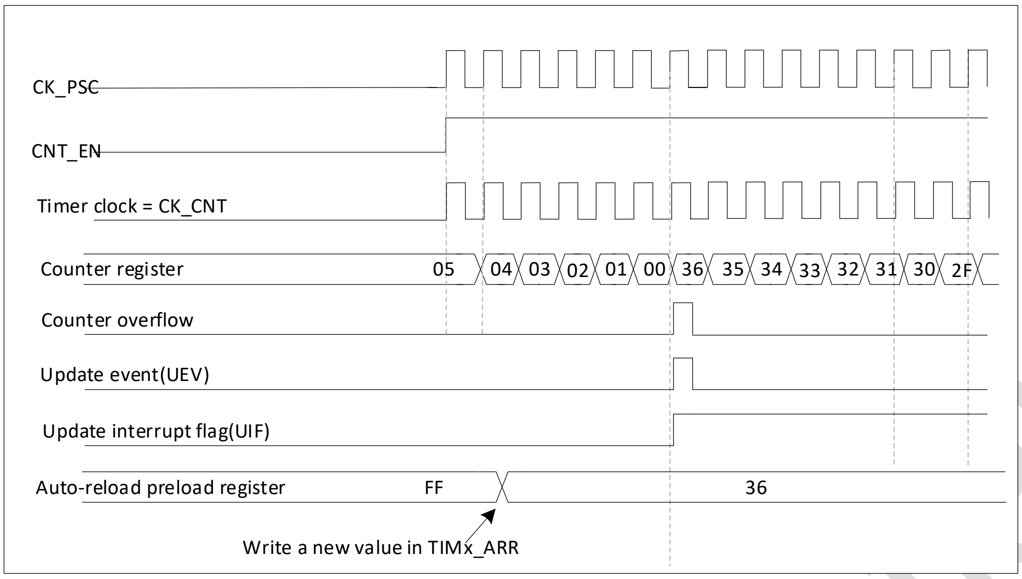


Figure 18-14 Counter timing diagram, update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) – 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are not equal to '0'. The Output compare interrupt flag of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and down (Center aligned 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. In this case, the counter restarts counting from0, as well as the counter of the prescaler.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. 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. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and DMA re-quests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

- 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 that if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

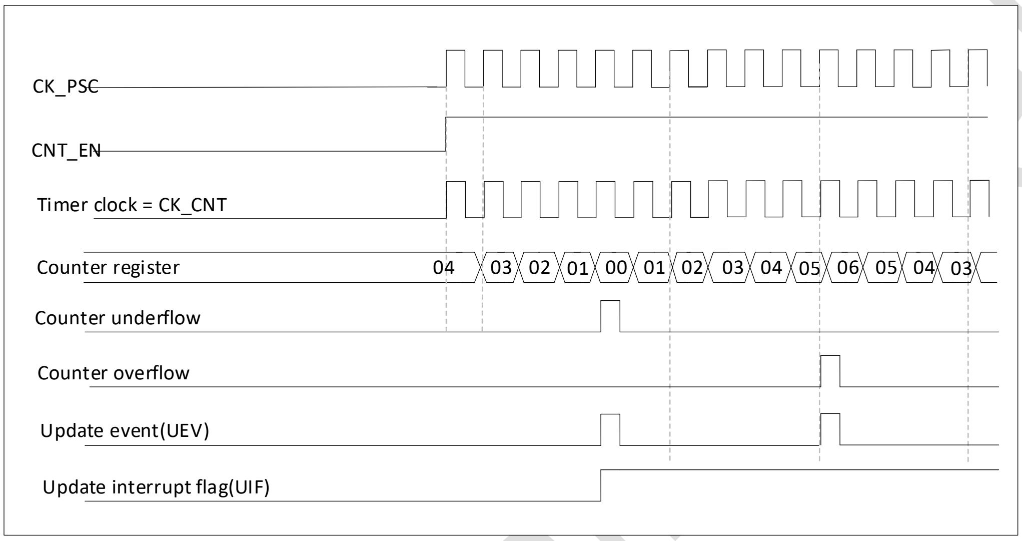


Figure 18-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

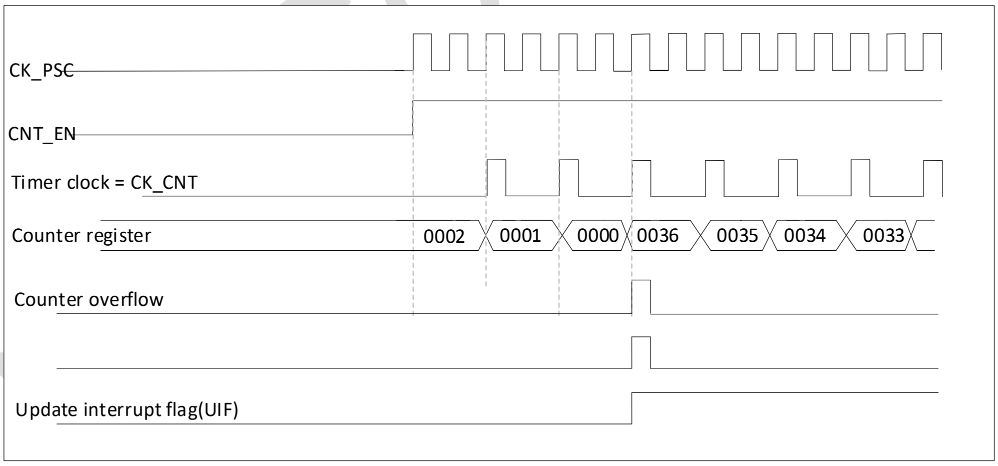


Figure 18-16 Counter timing diagram, internal clock divided by 2, TIMx_ARR=0x36

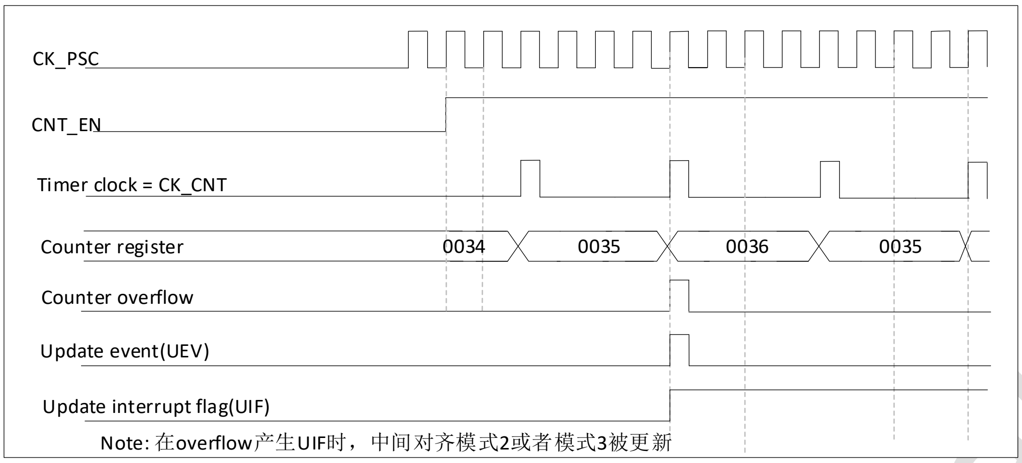


Figure 18-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x36

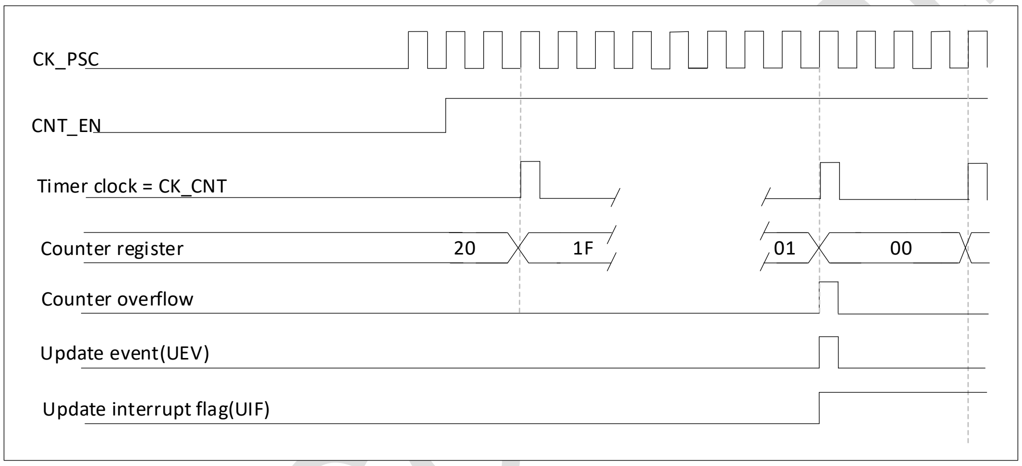


Figure 18-18 Counter timing diagram, internal clock divided by N

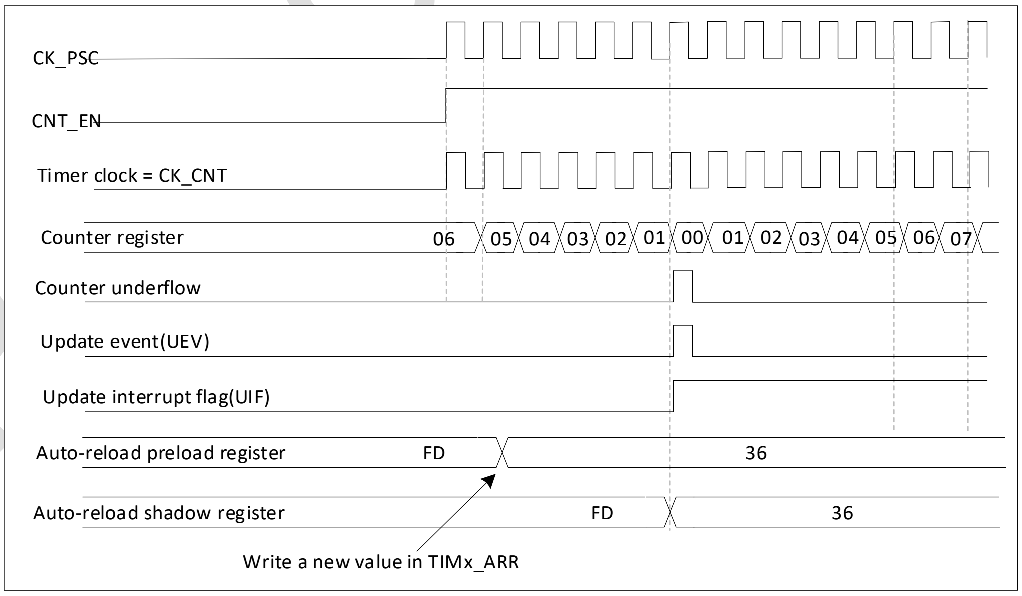


Figure 18-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

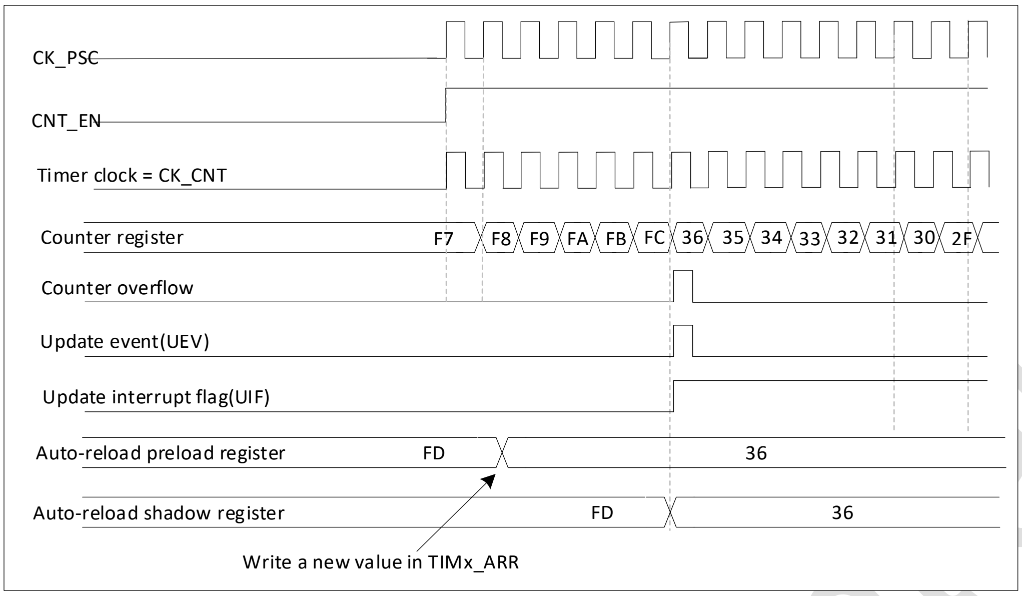


Figure 18-20 Counter timing diagram, Update event with ARPE=1 (counter overflow)

18.3.3. Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N + 1 counter overflows or underflows, where N is the value in the TIMx_RCR repetition counter register.

The repetition counter is decremented:

- At each counter overflow in upcounting mode
- At each counter underflow in downcounting mode
- At each counter overflow and at each counter underflow in center-aligned mode. Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. In center alignment mode, because the waveform is symmetrical, the maximum resolution is 2xTck if the comparison register is flushed only once per PWM cycle.

The repetition counter is an auto-reload type; the repetition rate is maintained as defined by the TIMx_RCR register value. When the update event is generated by software (by setting the UG bit in TIMx_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is, and the repetition counter is reloaded with the content of the TIMx_RCR register.

In the central alignment mode, for the odd value of the RCR, an update event occurs depending on whether an overflow or underflow occurs when the RCR register is written and when the counter starts. If the RCR was written before starting the counter, the UEV occurs on the overflow. For example for RCR = 3, the UEV is generated on each 4th overflow or underflow event depending on when RCR was written.

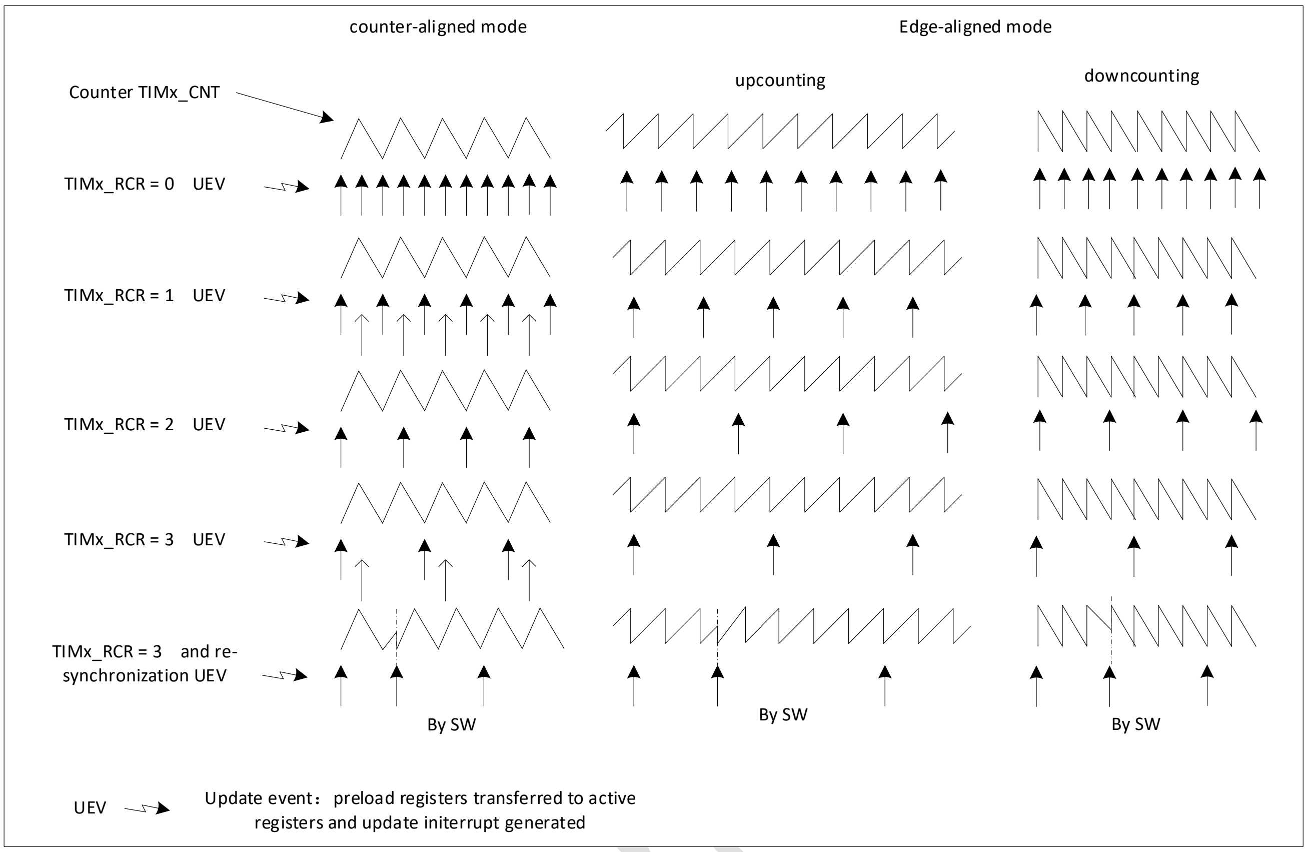


Figure 18-21 Update rate examples depending on mode and TIM1_RCR register settings

18.3.4. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, one timer Timer1 may be configured as a prescaler for another timer Timer3.

Internal clock source (CK_INT)

If the slave mode controller is disabled, then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software. As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

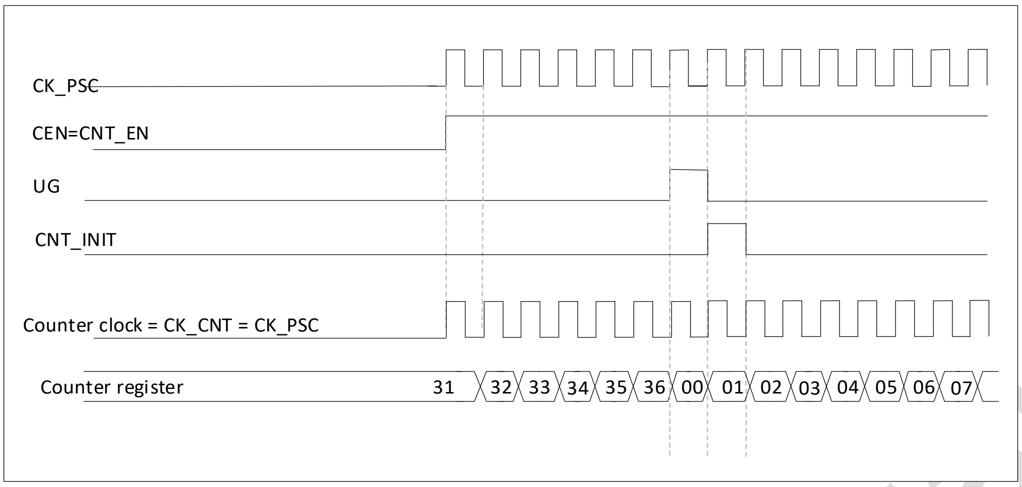


Figure 18-22 Control circuit in normal mode, internal clock divided by 1

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.

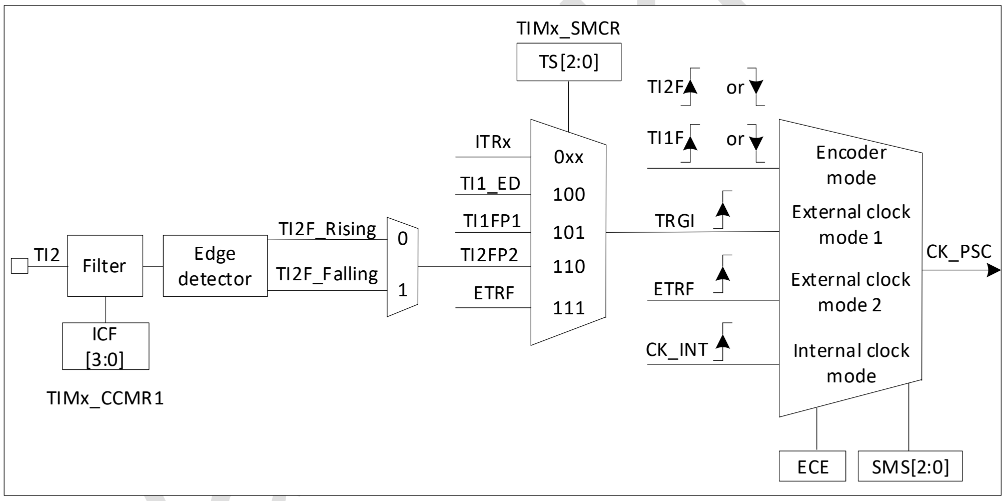


Figure 18-23 TI2 external clock connection example

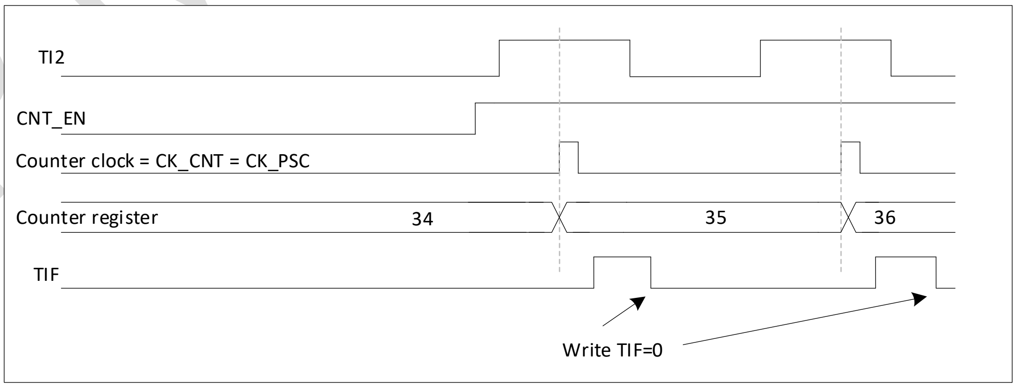


Figure 18-24 Control circuit in external clock mode 1

External clock source mode 2

This mode is selected by writing ECE=1 in the TIMx_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

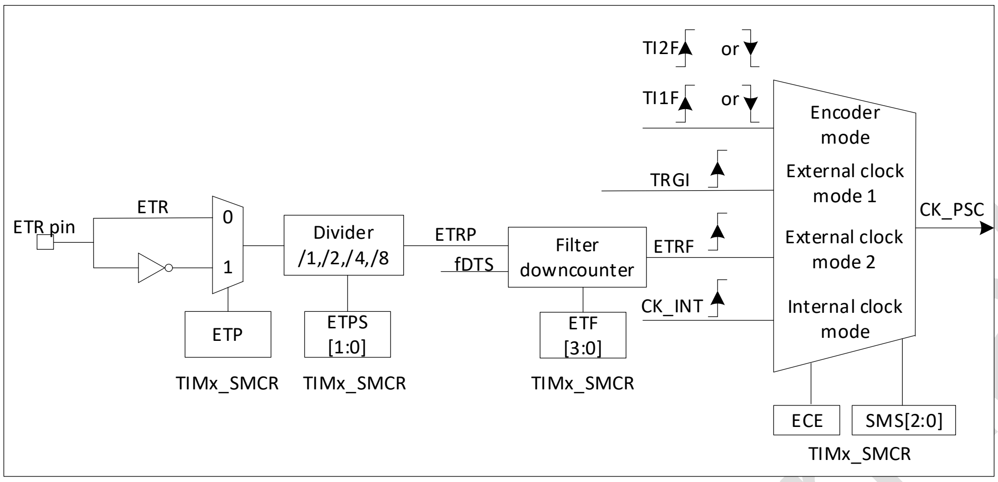


Figure 18-25 TI2 external trigger input block

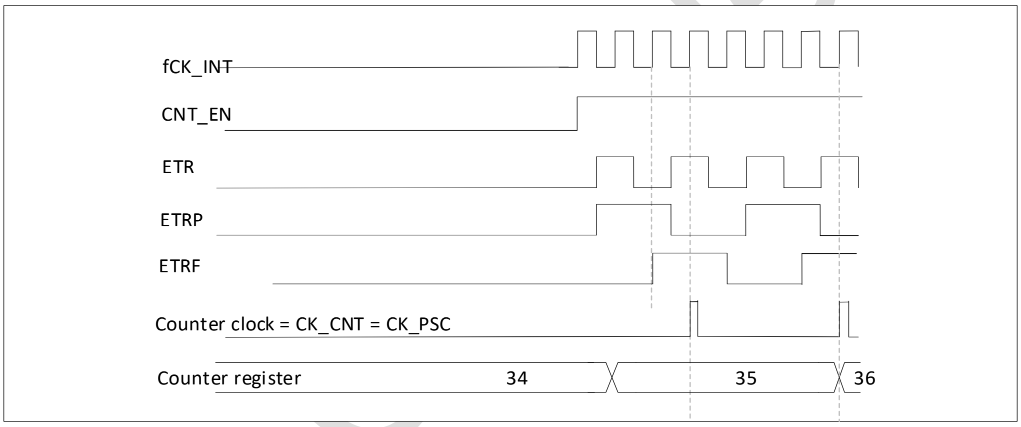


Figure 18-26 Control circuit in external clock mode 2

18.3.5. Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

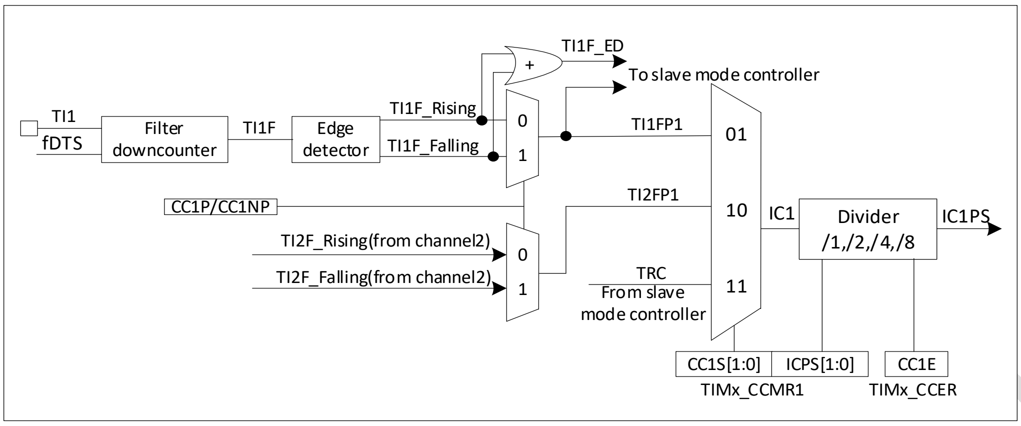


Figure 18-27 Capture/compare channel (example: channel 1 input stage)

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.

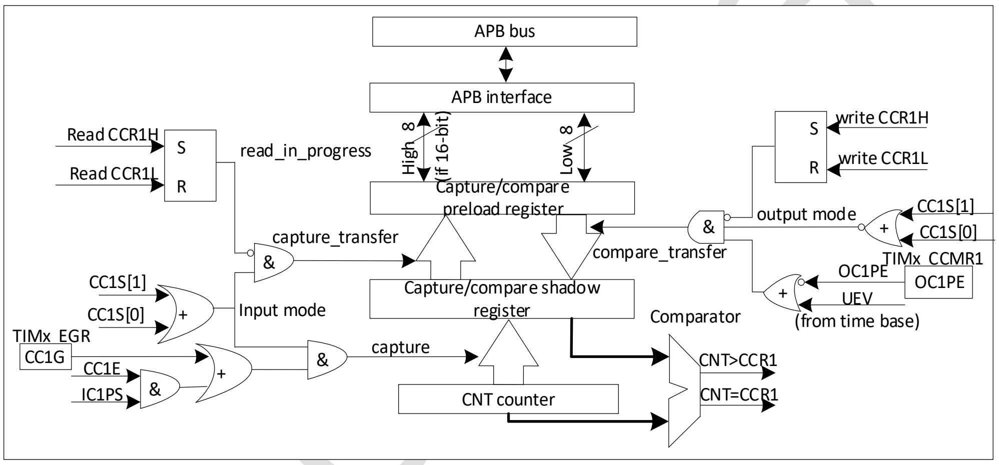


Figure 18-28 Capture/compare channel 1 main circuit

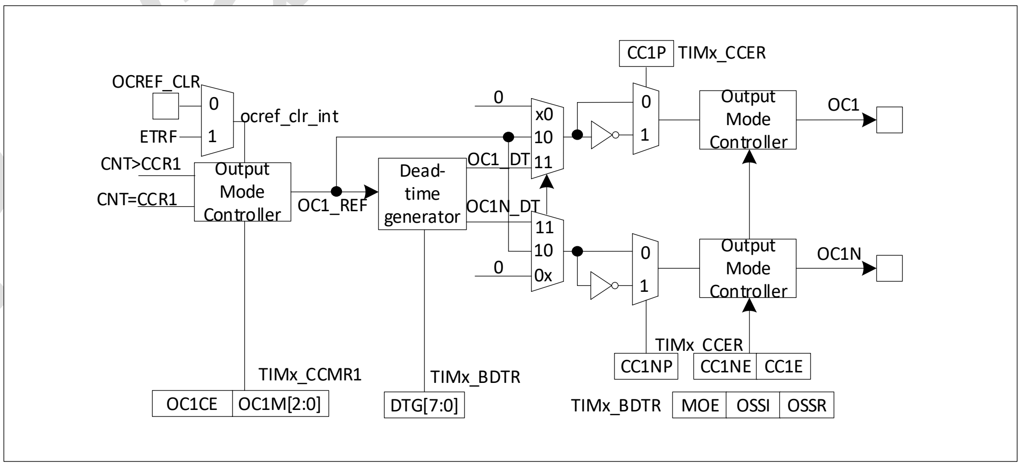


Figure 18-29 Output stage of capture/compare channel (channel 1 to 3)

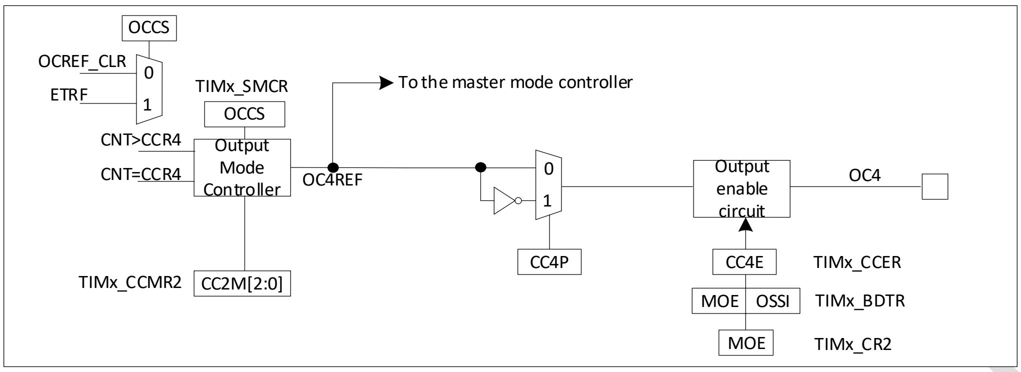


Figure 18-30 Output stage of capture/compare channel (channel 4)

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.

18.3.6. Input capture mode:

In Input capture mode, the capture/compare registers are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture event occurs, the corresponding CCxIF flag (TIMx _ SR register) is set to 1, and if interrupt and DMA operations are turned on, an interrupt or DMA request will be generated. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to ‘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 the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in 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 Fck_int frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- 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 necessary, associated interrupt requests are allowed by setting the CC1IE bit in the TIMx _ DIER register, and DMA requests are 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.
- CC1IF flag 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.
- If the CC1DE bit is set, a DMA request will also be generated.

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: An input capture interrupt and/or DMA request can be generated by software by setting the corresponding CCxG bit in the TIMx _ EGR register.

18.3.7. Input capture mode (PWM input mode)

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on 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, you can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- 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.

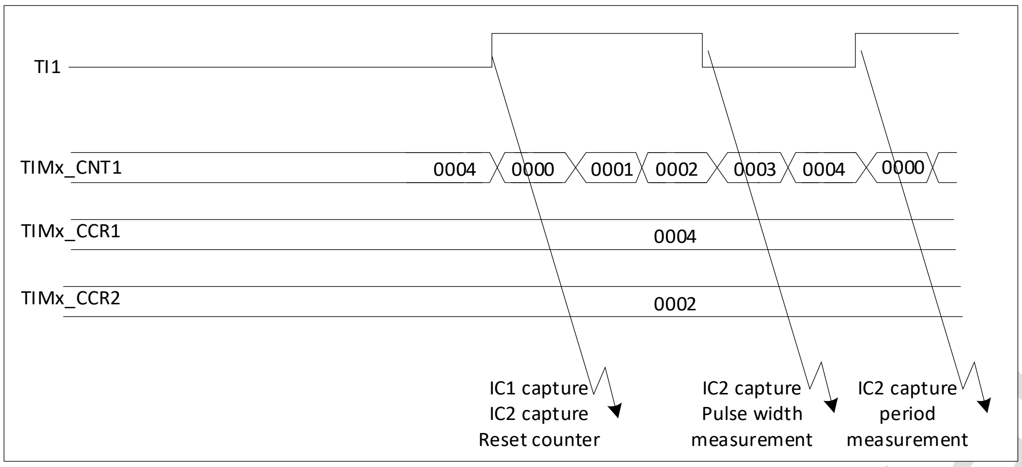


Figure 18-31 PWM input mode timing

18.3.8. 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. 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 and DMA requests can be sent accordingly. This is described in the output compare mode section below.

18.3.9. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. 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).
- 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.

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register. 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 at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE= ’0’, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the figure below.

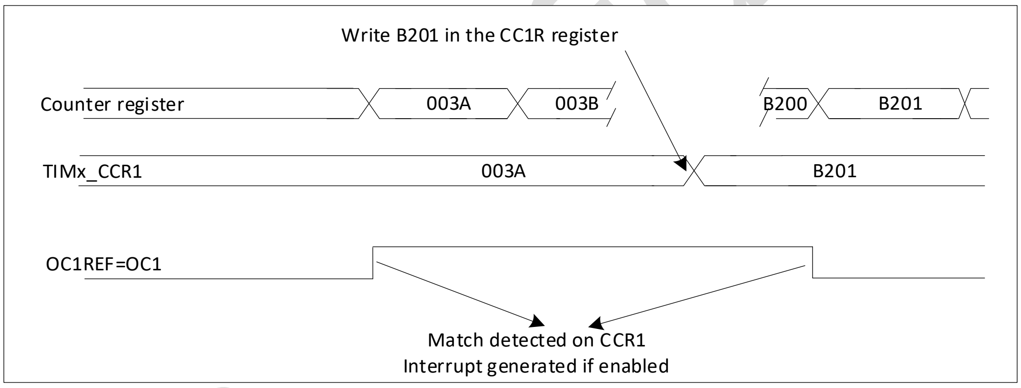


Figure 18-32 Output compare mode, toggle on OC1

18.3.10. PWM mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

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 compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

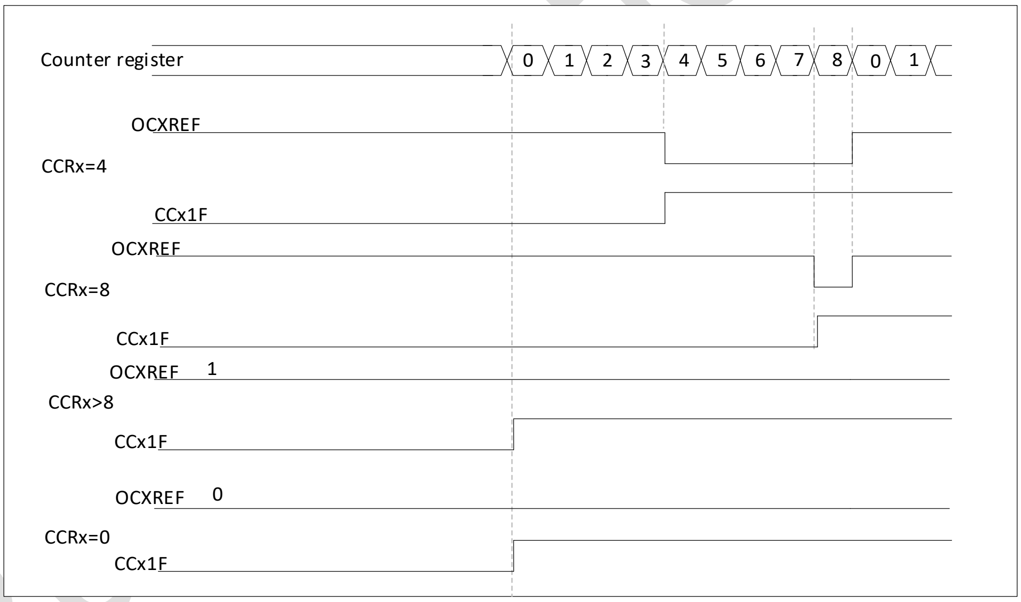


Figure 18-33 edge alignment PWM output, up (ARR = 8)

■ **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 compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at '1'. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from'00' (all the remaining configurations having the same effect on the OCxRef/OCx signals). 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.

Figure below shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in TIMx_CR1 register.

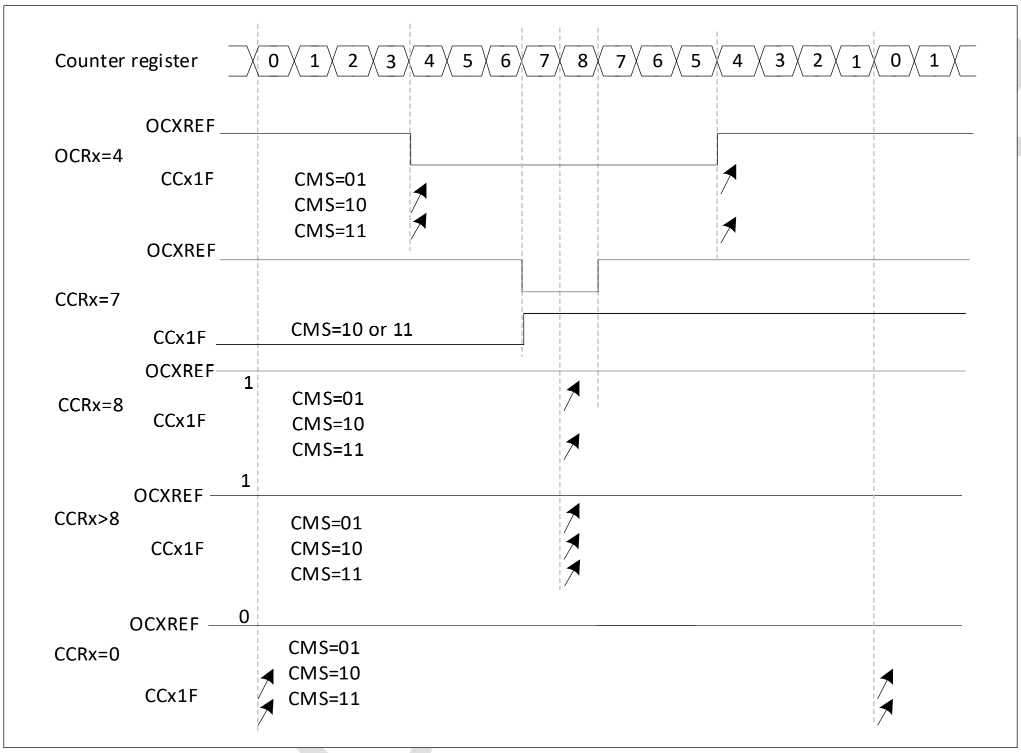


Figure 18-34 Center-aligned PWM waveforms (ARR=8)

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. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular: The direction is not updated if you write a value in the counter that is greater than the auto-reload value (TIMx_CNT>TIMx_ARR). For example, if the counter was counting up, it continues to count up. The direction is updated if you write 0 or write the TIMx_ARR value in the counter 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.

18.3.11. 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 activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. 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. There is one 8-bit dead-time generator for each channel. 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)

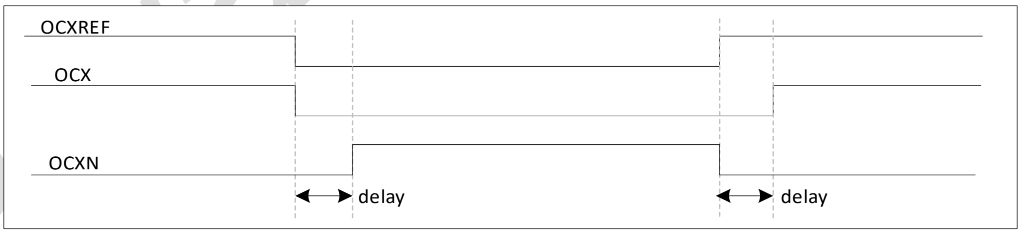


Figure 18-35 Complementary output with dead-time insertion

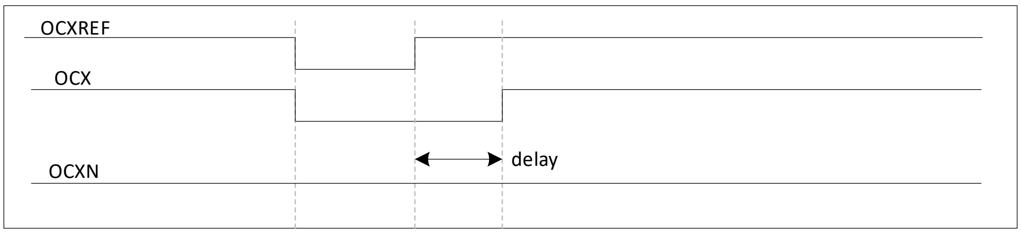


Figure 18-36 Dead-time waveforms with delay greater than the negative pulse

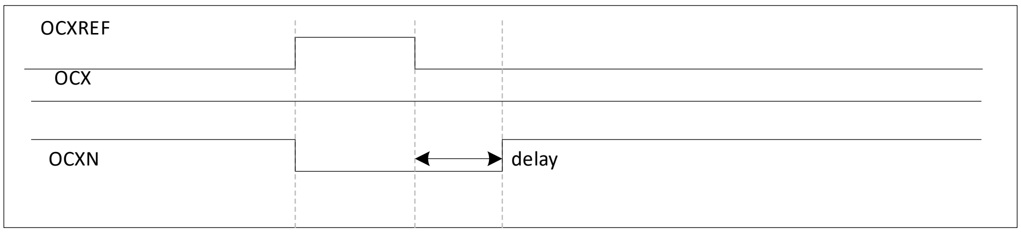


Figure 18-37 Dead-time waveforms with delay greater than the positive pulse.

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

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 is not complemented and becomes active as soon as OCxREF is high. 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.

18.3.12. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits. In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- the CPU LOCKUP output
- the PVD output
- Clock failure event generated by CSS monitoring
- the output from a comparator

When exiting from reset, the break circuit is disabled and the MOE bit is low. You can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. 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.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the output into an invalid state, an idle state, or a reset state (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 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that due to the resynchronization of the MOE, the dead time is longer than usual (approximately 2 clock cycles of ck _ tim).
  - If OSSI=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx _ BDTR register is set, the MOE bit is automatically set at the next update event UEV; For example, this can be used for shaping. Otherwise, the MOE remains low until it is set '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: 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.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR Register.

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 to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The application can choose from 3 levels of protection selected by the LOCK bits in the TIMx_BDTR register. 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.

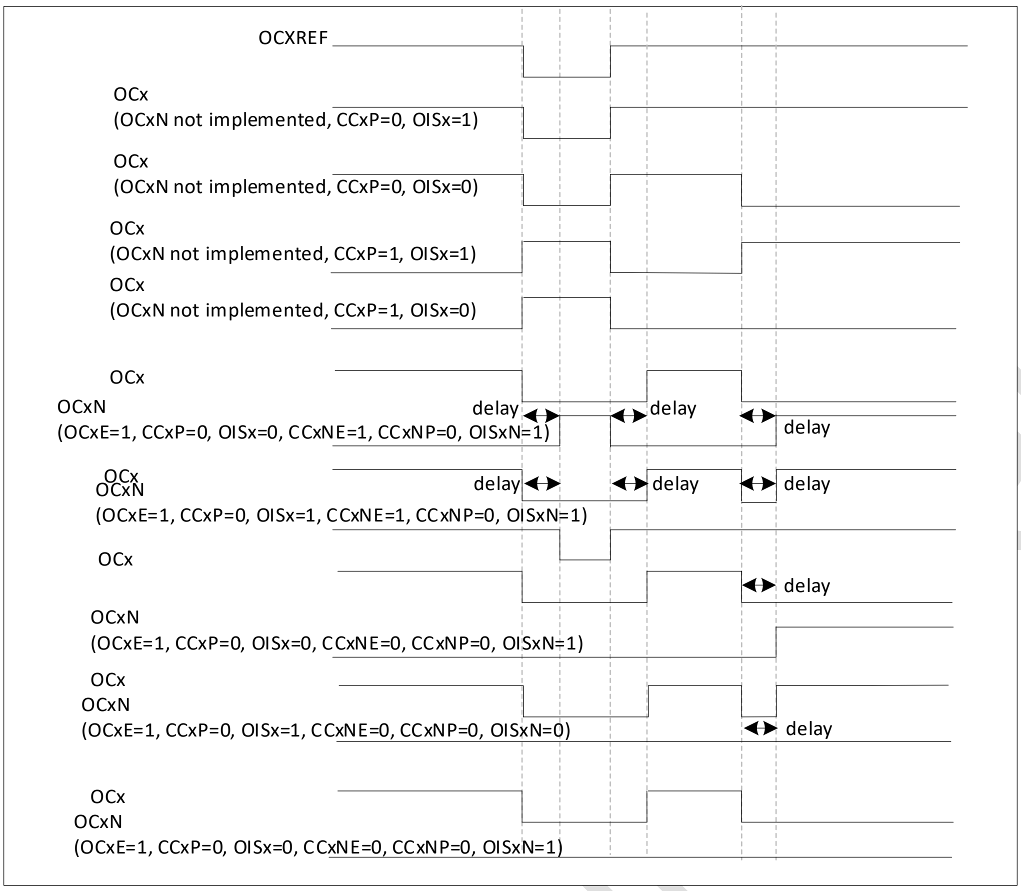

OCXREF
OCx
(OCxN not implemented, CCxP=0, OISx=1)
OCx
(OCxN not implemented, CCxP=0, OISx=0)
OCx
(OCxN not implemented, CCxP=1, OISx=1)
OCx
(OCxN not implemented, CCxP=1, OISx=0)
OCx
OCxN
(OCxE=1, CCxP=0, OISx=0, CCxNE=1, CCxNP=0, OISxN=1)
OCx
OCxN
(OCxE=1, CCxP=0, OISx=1, CCxNE=1, CCxNP=0, OISxN=1)
OCx
OCxN
(OCxE=1, CCxP=0, OISx=0, CCxNE=0, CCxNP=0, OISxN=1)
OCx
OCxN
(OCxE=1, CCxP=0, OISx=1, CCxNE=0, CCxNP=0, OISxN=0)
OCx
OCxN
(OCxE=1, CCxP=0, OISx=0, CCxNE=0, CCxNP=0, OISxN=1)
delay

Figure 18-38 Output behavior in response to a break

18.3.13. Clearing the OCxREF signal on an external event

For a given channel, setting the corresponding OCxCE bit in the TIMx _ CCMRx register to 1 can pull the OCxREF signal low with the high level at the ETRF input, and the OCxREF signal will remain low until the next update event UEV.

This function can only be used in output compare and PWM modes. It does not work in Forced mode.

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 according to the user needs.

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 programmed in PWM mode.

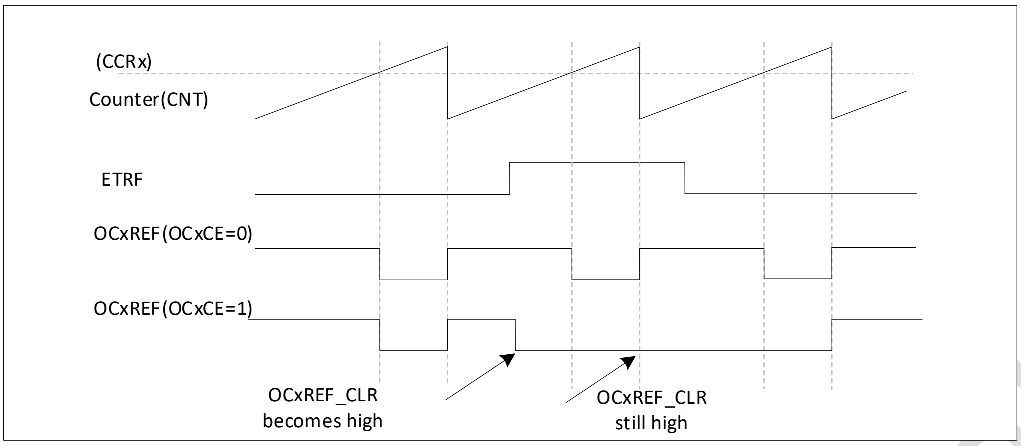


Figure 18-39 OCxREF for clearing TIM1

Note: If the PWM duty cycle is 100% (if CCRx > ARR), enable OCxREF again on the next counter overflow.

18.3.14. 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. Thus you can program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM 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.

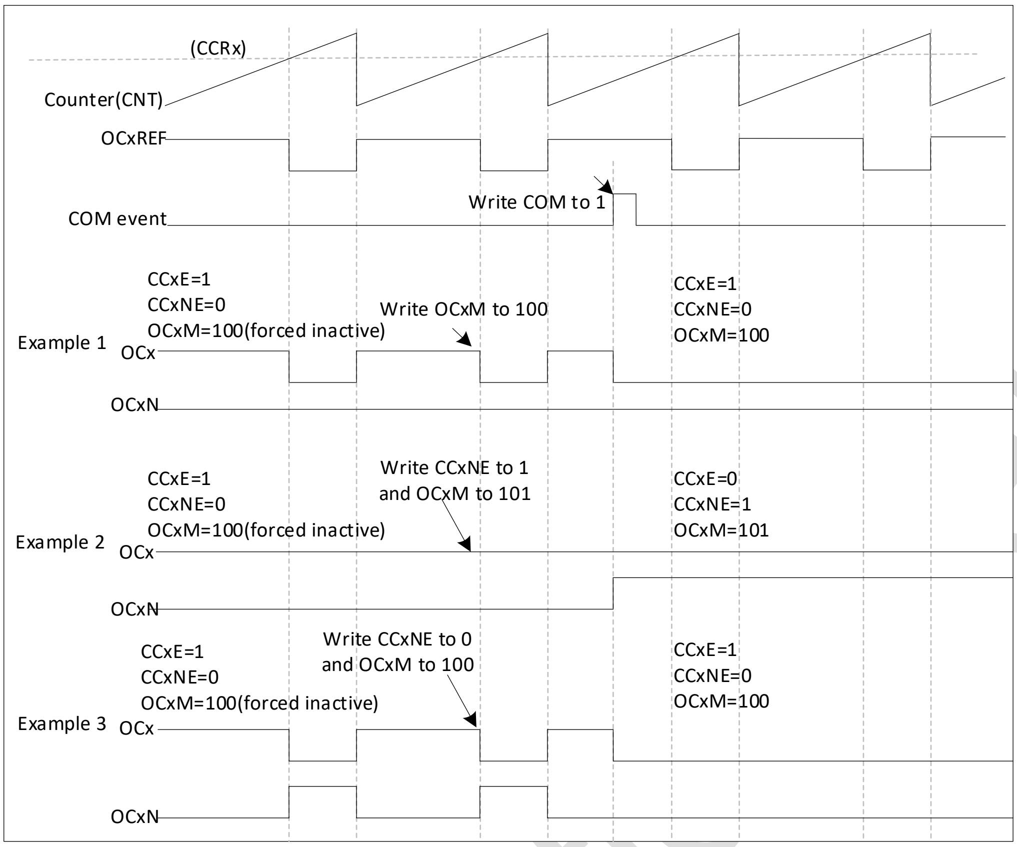


Figure 18-40 6-step generation, COM example (OSSR=1)

18.3.15. 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:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

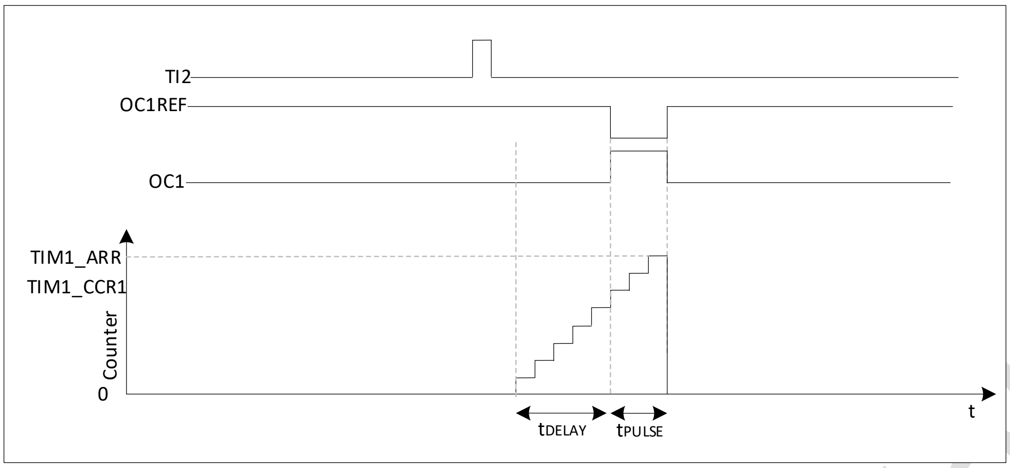


Figure 18-41 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let’s use TI2FP2 as trigger 1:

- 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 t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} 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 ‘1’ to ‘0’ 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)

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 t_{DELAY} 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.

18.3.16. Retriggerable one pulse mode (OPM)

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: The OCxM[3:0] and SMS[3:0] bit fields are split into two parts for compatibility reasons, the most significant bit are not contiguous with the 3 least significant ones.

This mode must not be used with center-aligned PWM modes. It is mandatory to have CMS[1:0] = 00 in TIMx_CR1.

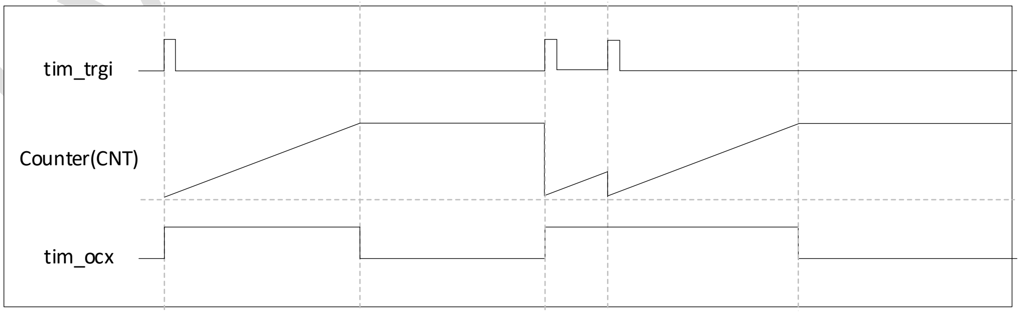


Figure 18-42 Example of a retriggerable monopulse mode

18.3.17. Encoder interface mode

To select Encoder Interface mode: write SMS='001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS=010 if it is counting on TI1 edges only and SMS=011 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, 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 and TI2FP2 are the signals of TI1 and TI2 after passing through the input filter and polarity control; If there is no filtering and disguised phase, then TI1FP1 = TI1 and TI2FP2 = TI2. 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 18-1 Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Counting on TI2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

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 on TI1).
- CC2S='01' (TIMx_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P and CC2NP = ‘0’ (TIMx_CCER register, TI1FP1 non-inverted, TI1FP1=TI1)
- CC2P and CC2NP = ‘0’ (TIMx_CCER register, TI1FP2 non-inverted, TI1FP2=TI2)

- SMS= 011 (TIMx_SMCR register, both inputs are active on both rising and falling edges)
- CEN='1' (TIMx_CR1 register, Counter enabled)

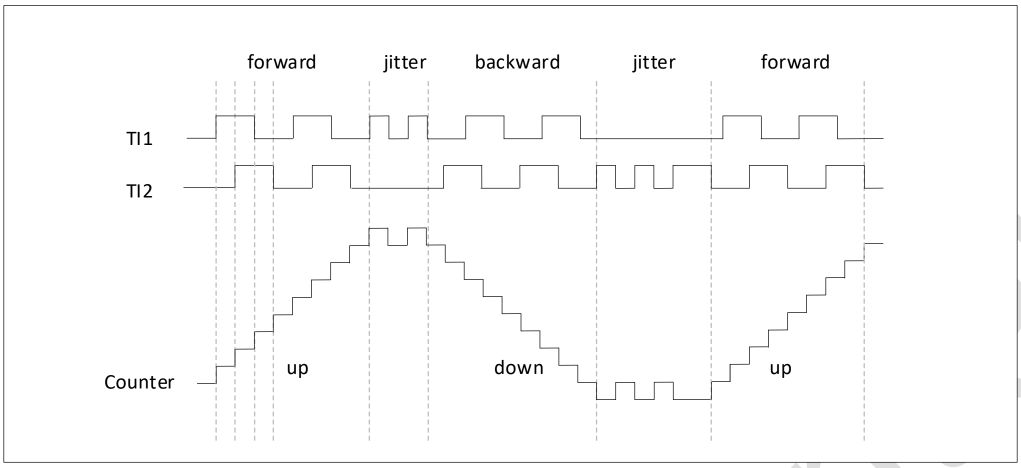


Figure 18-43 Example of counter operation in encoder interface mode

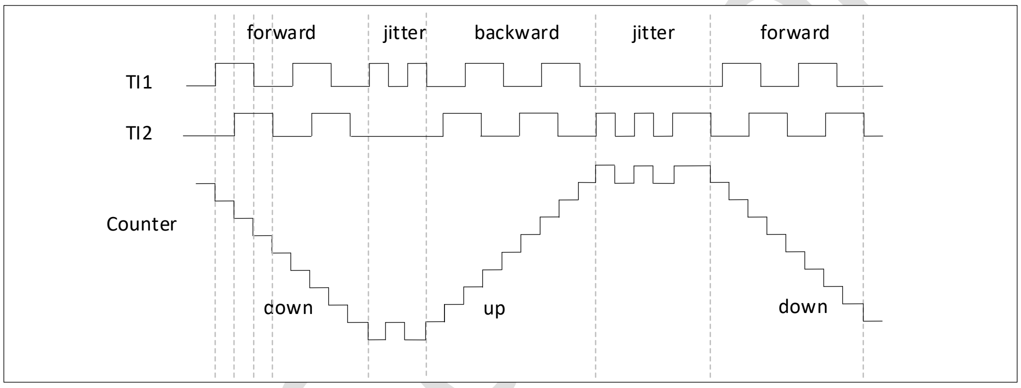


Figure 18-44 TI1P1 Example of encoder interface mode with polarity inversion

The timer, when configured in Encoder Interface mode provides information on the sensor's current position. Dynamic information can be obtained (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. If possible, the value of the counter can be latched 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.

18.3.18. Timer input XOR function

The TI1S bit in the TIM_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins 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.

18.3.19. Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIM2 referred to as "interfacing timer". The "interfacing timer" captures the 3 timer input pins (CC1, CC2, CC3) connected through an XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

The slave mode controller is configured in reset mode; 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.

On the “interfacing timer”, capture/compare channel 1 is configured in capture mode, capture signal is TRC. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1) 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.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIMx_CR2 register to ‘1’.
- 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 get a maximum counter period longer than the time between 2 changes on the sensors.
- Program the channel 1 in capture mode (TRC selected): write the CC1S bits in theTIMx_CCMR1 register to ‘01’. The digital filter can also be programmed if needed,
- Program channel 2 in PWM 2 mode with the desired delay: write the OC2M bits to ‘111’and the CC2S bits to ‘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).

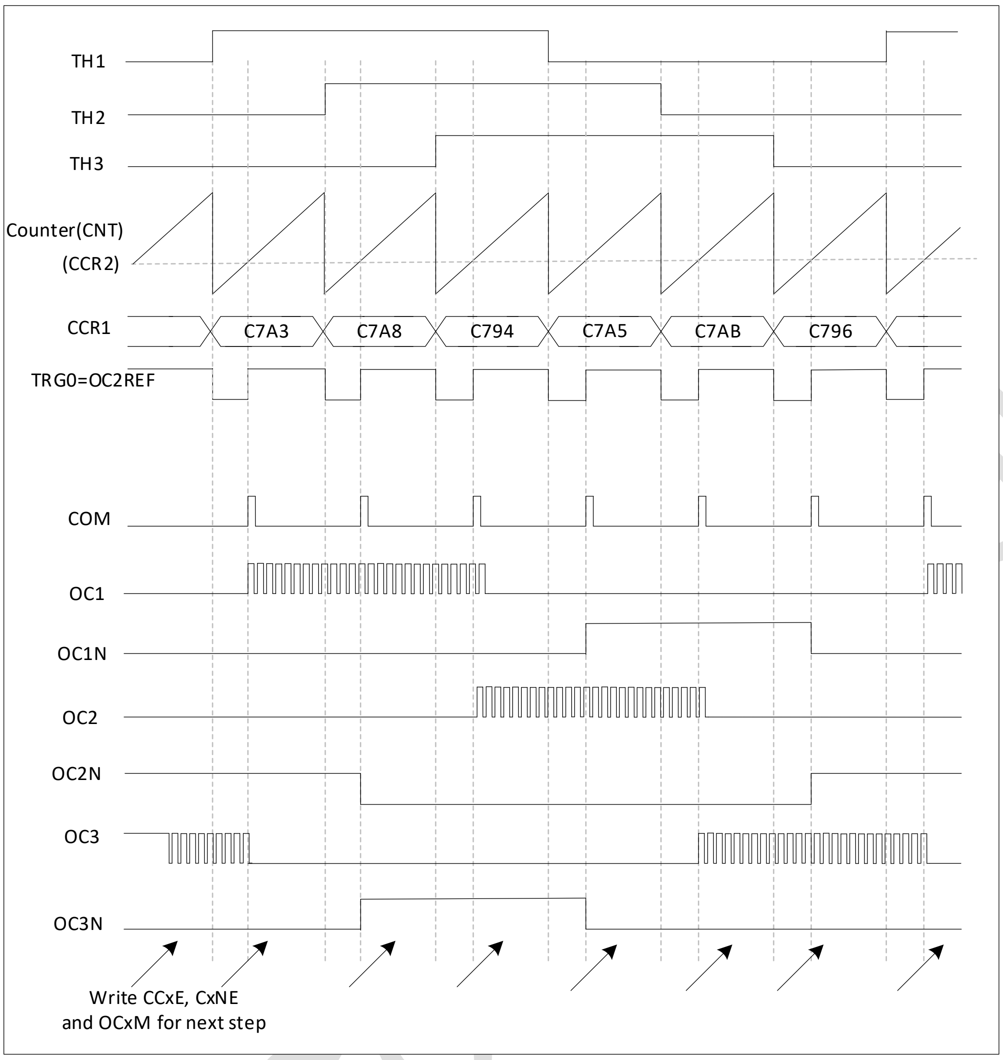


Figure 18-45 Example of Hall sensor interface

18.3.20. TIM and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are 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=101 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.

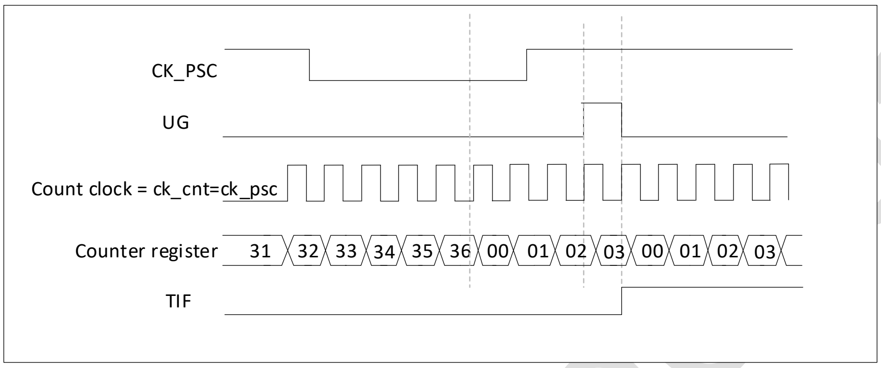


Figure 18-46 Control circuit in reset mode

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.

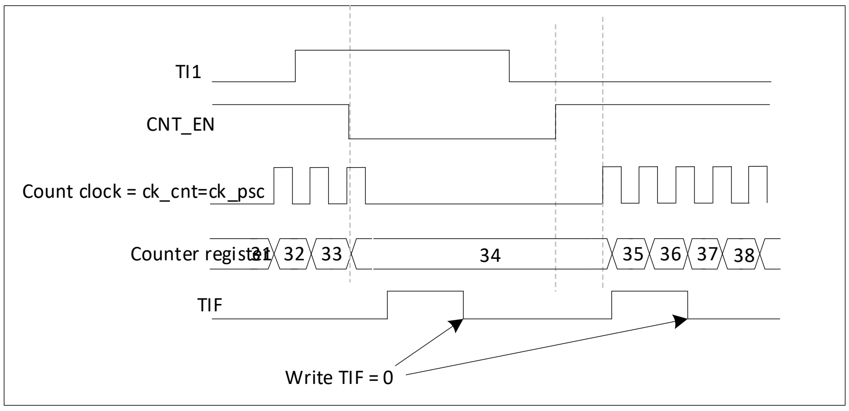


Figure 18-47 Control circuit in Gated mode

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 re-synchronization circuit on TI2 input.

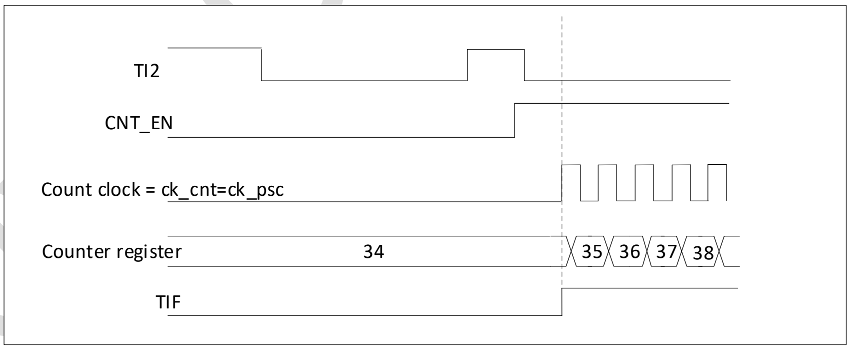


Figure 18-48 Control circuit in Gated mode

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 by programming the TIMx_SMCR register as follows:
   - 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. Configure the channel 1 as follows, to detect rising edges on TI:
   - 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
   - Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
3. Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. 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.

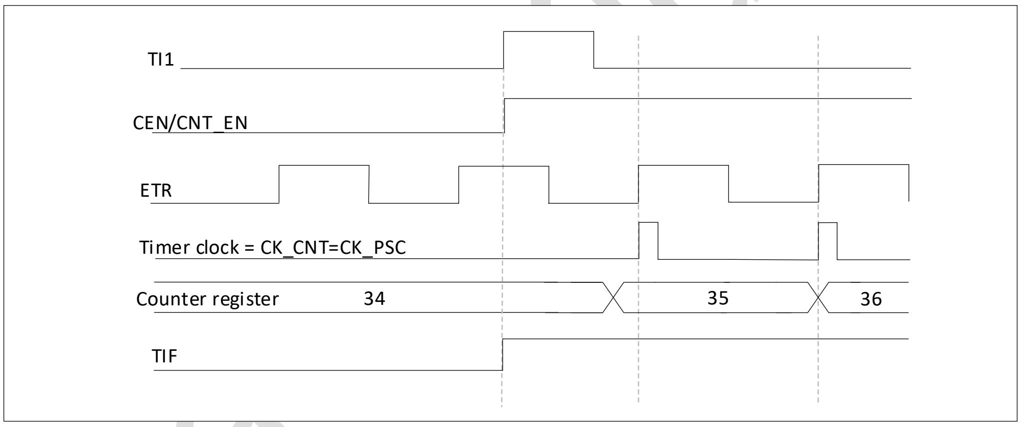


Figure 18-49 Control circuit in external clock mode 2 + trigger mode

18.3.21. Timer synchronization

The TIM timer is connected internally and is used for the synchronization or linking function of the timer. When one timer is in master mode, it can reset, start, stop, or clock the counter of another timer in slave mode.

18.3.22. Debug mode

When the microcontroller enters debug mode, the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

18.4. TIM1 registers

18.4.1. TIM1 control register 1 (TIM1_CR1)

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	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock Division Factor These 2 bits define the division ratio between the timer clock (CK _ INT) frequency, the dead time and the sampling clock used by the dead generator and the digital filter (ETR, Tix) 00: tDTS = tCK _ INT01: tDTS = 2 x tCK _ INT10: tDTS = 4 x tCK _ INT11: Reserve, do not use this configuration
7	ARPE	RW	0	Automatic reload preload allowed bit 0: TIM1 _ ARR register is not buffered 1: TIM1 _ ARR register is loaded into buffer
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge alignment 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. Configured as output channel The output of (CCxS = 00 in the TIM1 _ CCMRx register) compares the interrupt flag bit and is set only when the counter counts down. 10: Center-aligned mode 2. The counter counts up and down alternatively. The counter counts up and down alternatively. Output comparison interrupt mark of channel configured as output (CCxS = 00 in TIM1 _ CCMRx register) Bit, set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The counter counts up and down alternatively. Output comparison interrupt mark of channel configured as output (CCxS = 00 in TIM1 _ CCMRx register) 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 used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	RW	0	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 This bit is set and cleared by software to select the UEV event sources. 0: If an update interrupt or DMA request is allowed, either of the following events produces 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 allowed to be generated, only a counter overflow/underflow generates an update OFF or DMA request
1	UDIS	RW	0	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.

Bit	Name	R/W	Reset Value	Function
				1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). 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: Counter disabled 1: Counter enabled 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.

18.4.2. TIM1 control register 2 (TIM1_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	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	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
31:15	Reserved	-	-	Reserved
14	OIS4	RW		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 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BKR register).
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BKR register).
7	TI1S	RW	0	TI1 selection 0: The TIM1_CH1 pin is connected to TI1 input 1: The TIM1_CH1, CH2 and CH3 pins are connected to the TI1 input.
6:4	MMS[2:0]	RW	000	Master mode selection These two bits are used to select synchronization information (TRGO) to be sent to the slave timer in master mode. Possible combinations like Below: 000: Reset - the UG bit from the TIM1_EGR register is used as trigger output (TRGO). 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. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). Sometimes require Starts multiple timers at the same time or controls one window from the timer. 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

Bit	Name	R/W	Reset Value	Function
				When the input is triggered, there is a delay on the TRGO unless the master/slave mode is selected (see description of the MSM bit in the TIM1 _ 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: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	RW	0	DMA Selection for Capture/Comparison 0: When a CCx event occurs, a DMA request for CCx is sent. 1: When an update event occurs, a DMA request for CCx is sent.
2	CCUS	RW	0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit only. 1: If the capture/compare control bit is preloaded (CCPC = 1), it can be controlled by setting the COM bit or TRGI A rising edge of updates them. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	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; Once this bit is set, they are only set when COM Bits are updated. Note: This bit acts only on channels that have a complementary output.

18.4.3. TIM1 slave mode control register (TIM1_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	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SMS[3]
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	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		

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	SMS[3]	RW	0	See SMS description for details
15	ETP	RW	0	External trigger polarity. This bit selects whether ETR or inverted 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. This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.
13:12	ETPS[1:0]	RW	00	External trigger prescaler The frequency of the external trigger signal ETRP must be at most 1/4 of the frequency of TIM1CLK. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2

Bit	Name	R/W	Reset Value	Function
				10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
11:8	ETF[3:0]	RW	0000	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, downsampling at fDTS 0001: fSAMPLING = fCK _ INT, N = 2 0010: fSAMPLING = fCK _ INT, N = 4 0011: fSAMPLING = fCK _ INT, N = 8 0100: fSAMPLING = fCK _ INT/2, N = 6 0101: fSAMPLING = fCK _ INT/2, N = 8 0110: fSAMPLING = fCK _ INT/4, N = 6 0111: fSAMPLING = fCK _ INT/4, N = 8 1000: fSAMPLING = fCK _ INT/8, N = 6 1001: fSAMPLING = fCK _ INT/8, N = 8 1010: fSAMPLING = fCK _ INT/16, N = 5 1011: fSAMPLING = fCK _ INT/16, N = 6 1100: fSAMPLING = fCK _ INT/16, N = 8 1101: fSAMPLING = fCK _ INT/32, N = 5 1110: fSAMPLING = fCK _ INT/32, N = 6 1111: fSAMPLING = fCK _ INT/32, N = 8 It must be noted that when ETF [3: 0] = 1 or 2 or 3, fDTS is re-placed by CK _ INT in the equation
7	MSM	RW	0	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	000	Trigger selection. This bit-field selects the trigger input to be used to synchronize the counter. 000: Reserved (ITR0) 001: Reserved (ITR1) 010： TIM2(ITR2) 011： TIM17(ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF) Note: These bits must be changed only when they are not used to avoid wrong edge detections at the transition.
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF _ CLR _ INT is connected to OCREF _ CLR input 1: OCREF _ CLR _ INT is connected to ETRF
2:0	SMS[2:0]	RW	000	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: Encoder mode 1 Counter counts up/down on TI1FP2 edge depending on TI2FP1 level. If SMS [3] = 0: 010: Encoder mode 2 Counter counts up/down on TI2FP1 edge depending on TI1FP2 level. 011: Encoder mode 3 The counter counts up/down on the edges of TI1FP1 and TI2FP2 depending on the level of the other inputs. 100: Reset mode Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated mode

Bit	Name	R/W	Reset Value	Function
				The counter clock is enabled when the trigger input (TRGI) is high. 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 at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. If SMS [3] = 1, SMS [2: 0] must be configured to 0. 000: Combined "reset + trigger" mode-the rising edge of the selected trigger input (tim _ trgi) reinitializes the counter, generates a register update and starts the counter. Note: In encoder mode, do not use uev as trgo output signal, (i.e. mms cannot be configured to 010)

Table 18-2 TIM1 internal trigger connection

Slave TIM	ITR0 (TS=000)	ITR1 (TS=001)	ITR2 (TS=010)	ITR3 (TS=011)
TIM1	Reserved	TIM2_TRGO	Reserved	TIM17_OC1

18.4.4. TIM1 DMA/interrupt enable register (TIM1 _ 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	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	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
31：15	Reserved	-	-	Reserved
14	TDE	RW	0	TDE: Triggering of DMA requests enabled 0: Trigger DMA request disabled. 1: Triggering of DMA request enabled
13	COMDE	RW	0	COMDE: DMA requests for COM enabled 0: Disable COM DMA request 1: COM DMA requests enabled
12	CC4DE	RW	0	CC4DE: DMA requests enabled capture/compare 4 0: DMA request disabled to capture/compare 4 1: DMA request enabled to capture/compare 4
11	CC3DE	RW	0	CC3DE: DMA requests enabled to capture/compare 3 0: DMA request for capture/compare 3 disabled 1: DMA request enabled to capture/compare 3
10	CC2DE	RW	0	CC2DE: DMA requests that enable capture/compare 2 0: DMA request for capture/compare 2 disabled 1: DMA request enabled to capture/compare 2
9	CC1DE	RW	0	CC1DE: DMA request enabled to capture/compare 1 0: DMA request for capture/compare 1 is disabled 1: DMA request enabled to capture/compare 1
8	UDE	RW	0	UDE: Updated DMA requests enabled 0: Updated DMA requests disabled 1: Updated DMA requests enabled
7	BIE	RW	0	BIE: Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	TIE	RW	0	TIE: Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled

Bit	Name	R/W	Reset Value	Function
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	RW	0	CC4IE: Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

18.4.5. TIM1 status register (TIM1_SR)

Address offset: 0x010

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	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	IC2IR	IC1IR
-	-	-	-	-	-	-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	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	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	-	Rc_w0	Rc_w0	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	-	-	Reserved
23	IC4IF	RC_W0	0	Falling Edge Capture 4 Flag Refer to IC1IF description
22	IC3IF	RC_W0	0	Falling Edge Capture 3 Flag Refer to IC1IF description
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 can be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the falling edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A falling edge capture event occurs.
19	IC4IR	RC_W0	0	Rising Edge Capture 4 Flags Refer to IC1IR description
18	IC3IR	RC_W0	0	Rising Edge Capture 3 Flag Refer to IC1IR description
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 may be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the rising edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A rising edge capture event occurs.
12	CC4OF	Rc_w0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	Rc_w0	0	Capture/Compare 3 overcapture flag

Bit	Name	R/W	Reset Value	Function
				Refer to CC1OF description
10	CC2OF	Rc_w0	0	Capture/Compare 2 overcapture flag Refer to CC1OF description
9	CC1OF	Rc_w0	0	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 detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1OF flag was already set.
8	Reserved	-	-	Reserved
7	BIF	Rc_w0	0	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.
6	TIF	Rc_w0	0	Trigger interrupt flag When a trigger event occurs (when the slave mode controller is in a mode other than the gated mode, it is detected at the TRGI input that Effect edge, or either edge in gated mode) by hardware to the position 1. 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 CCxE, CCxNE, OCxM have been updated) this bit is set to 1 by the hardware. It is cleared by software. 0: No update occurred. 1: COM interrupt pending.
4	CC4IF	Rc_w0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	Rc_w0	0	Capture/Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	Rc_w0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set to 1 by hardware when the counter value matches the comparison value, except in centrosymmetric mode (refer to TIM1 _ CR1 register Device's CMS bit). It is cleared by software. 0: No match; 1: The content of the counter TIM1_CNT matches the content of the TIM1_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM1_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM1_CCR1 register (An edge has been detected on IC1 which matches the selected polarity) Note: When CEN is on, this bit will also be set.
0	UIF	Rc_w0	0	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 by hardware when the registers are updated: – At overflow or underflow regarding the repetition counter value (update if REP_CNT = 0) and if the UDIS=0 in the TIM1_CR1 register. If UDIS of the TIM1 _ CR1 register = 0 and URS = 0, an update event occurs when UG of the TIM1 _ EGR register = 1 Software (software re-initializes CNT);

Bit	Name	R/W	Reset Value	Function
				− If UDIS = 0 and URS = 0 of the TIM1 _ CR1 register, an update event occurs when the CNT is reinitialized by the trigger event Pieces. (Refer to slave mode control register (TIM1 _ SMCR))

18.4.6. TIM1 event generation register (TIM1_EGR)

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	Res	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	-	-	-	-	-	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	Reserved
7	BG	W	0	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. At this time, MOE = 0 and BIF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated.
6	TG	W	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: TIF of the TIM1 _ SR register = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated.
5	COMG	W	0	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.
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 by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. If channel CC1 is configured as input: The current counter value is captured in the TIM1 _ CCR1 register, and CC1IF = 1 is set. 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 can be set by software, it is automatically cleared by hardware. 0: No action

Bit	Name	R/W	Reset Value	Function
				1: Reinitialize the counter and generate an update of the registers. Note: The counter of the prescaler is also cleared to 0 (but the prescaler The coefficient remains unchanged). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reloadvalue (TIM1_ARR) if DIR=1 (downcounting).

18.4.7. TIM1 capture/compare mode register 1 (TIM1_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	OC2 M[3]	Res	Res	Res	Res	Res	Res	Res	OC1 M[3]
-	-	-	-	-	-	-	RW	-	-	-	-	-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2 CE	OC2M [2:0]			OC2 PE	CO2 FE	CC2S [1:0]		OC1 CE	OC1M [2:0]			OC1 PE	OC1 FE	CC1S [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
31:25	Reserved	-	-	Reserved
24	OC2M[3]	RW	0	See OC2M description
23:17	Reserved	-		Reserved, must be kept at reset value.
16	OC1M[3]	RW	0	See OC1M description
15	OC2CE	RW	0	Output Compare 2 clear enable
14:12	OC2M [2:0]	RW	000	Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 preload enable
10	OC2FE	RW	0	Output Compare 2 fast enable
9:8	CC2S [1:0]	RW	00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIM1_CCER).
7	OC1CE	RW	0	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 [2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 andOC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen. The comparison between the output comparison register TIM1 _ CCR1 and the counter TIM1 _ CNT does not work for OC1REF With; 0001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the cap-ture/compare register 1 (TIMx_CCR1). 0010: Set channel 1 to inactive level on match. When the value of the counter TIMx _ CNT is compared with the capture/com-pare register When 1 (TIMx _ CCR1) is the same, OC1REF is forced to be low. 0011: Toggle OC1REF toggles when TIMx_CNT=TIMx_CCR1. 0100: Force inactive level OC1REF is forced low.

Bit	Name	R/W	Reset Value	Function
				0101: Force active level OC1REF is forced high. 0110: PWM mode 1 - In upcounting, channel 1 is active as long as TIM1_CNT<TIM1_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0') as long as TIM1_CNT>TIM1_CCR1 else active (OC1REF='1'). 0111: PWM Mode 2-On up count, channel 1 is invalid level once TIM1 _ CNT < TIM1 _ CCR1, active level otherwise; When counting down, channel 1 is active once TIM1 _ CNT > TIM1 _ CCR1, otherwise it is invalid. 1000: Resumable 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: Recoverable OPM Mode 2-In incremental count mode, the channel is invalid 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 down 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. 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. Note 3: When using recoverable OPM mode, do not configure the counting mode to central counting mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIM1_CCR1 disabled. TIM1_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Turn on the preload function of the TIM1 _ CCR1 register. Read and write operations only operate on the preload register. The preload value of TIM1 _ 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: The PWM mode can be used without validating the preload register only in one pulse mode. Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 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. Therefore, OC is set to a comparison level and It is not related to the comparison results. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 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 works when the internal trigger input is selected

Bit	Name	R/W	Reset Value	Function
				(selected by TS bit of TIM1 _ SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIM1_CCER).

Input capture 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**
IC2F [3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F [3:0]				IC1PSC [1:0]		CC1S [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
31:16	Reserved	-	-	Reserved
15:12	IF2F	RW	0000	Input capture 2 filter
11:10	IC2PSC [1:0]	RW	00	Input/capture 2 prescaler
9:8	CC2S [1:0]	RW	0	Capture/Compare 2 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIM1_CCER).
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 in-put and the length of the digital filter applied to TI1. 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, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is de-tected 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 [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1.

Bit	Name	R/W	Reset Value	Function
				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 works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIM1_CCER).

18.4.8. TIM1 capture/compare mode register 2 (TIM1_CCMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	OC4M[3]	Res	Res	Res	Res	Res	Res	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	CO4FE	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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	OC4M[3]	RW	0	Refer to OC4M description
23:17	Reserved	-	-	Reserved
16	OC3M[3]	RW	0	Refer to OC3M description
15	OC4CE	RW	0	Output Compare 4 clear enable
14:12	OC4M [2:0]	RW	000	Output compare 4 mode
11	OC4PE	RW	0	Output Compare 4 preload enable
10	OC4FE	RW	0	Output Compare 4 fast enable
9:8	CC4S [1:0]	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 is working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIM1_CCER).
7	OC3CE	RW	0	Output Compare 3 clear enable
6:4	OC3M [2:0]	RW	00	Output compare 3 mode
3	OC3PE	RW	0	Output Compare 3 preload enable
2	OC3FE	RW	0	Output Compare 3 fast enable
1:0	CC3S [1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIM1_CCER).

Input capture mode:

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:12	IC4F	RW	0000	Input capture 4 filter
11:10	IC4PSC	RW	00	Input/capture 4 prescaler
9:8	CC4S	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 is working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIM1_CCER).
7:4	IC3F	RW	0000	Input capture 3 filter
3:2	IC3PSC	RW	00	Input/capture 3 prescaler
1:0	OC3S	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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, IC1 is mapped on TRC. This mode only works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIM1_CCER).

18.4.9. TIM1 capture/compare enable register (TIM1_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	Res	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

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	Reserved
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 output enable Refer to CC1E description.
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
10	CC3NE	RW	0	Input/Capture 3 complementary output enable. Refer to CC1NE description.
9	CC3P	RW	0	Input/Capture 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable Refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	CC2NE	RW	0	Input/Capture 2 complementary output enable. Refer to CC1NE description.
5	CC2P	RW	0	Input/Capture 2 output polarity Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity.

Bit	Name	R/W	Reset Value	Function
				0: OC1N active high. 1: OC1N active low. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register) and CC1S='00' (channel configured as output).
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1Nand CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The CC1NP/CC1P bits select the polarities of TI1FP1 and TI2FP1 as trigger or capture signals. 00: Non-reverse/rising edge: TIxFP1 rising edge active (capture, trigger in reset mode, external clock, or trigger mode); TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: Reverse/falling edge: TIxFP1 falling edge active (capture, trigger in reset mode, 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-reverse/double edge TIxFP1 both rising and falling edges are valid (capture, trigger in reset mode, 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 in the TIMx _ CR2 register is set, the actual valid bit of CC1P will only load the preload value when the com event occurs. Note: Once the LOCK level (LCCK bit in the TIM1 _ BDTR register) is set to 3 or 2, this bit cannot be modified
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. The CC1 channel is configured as input: This bit determines whether the value of the counter can be captured into the TIM1 _ CCR1 register. 0: Capture disabled. 0: Capture enabled. Notes: For complementary output channels, this bit is preloaded. If the CCPC bit in the TIMx _ CR2 register is set, the actual valid bit of CC1E will only load the preload value when the com event occurs.

Table 18-3 Output control bits for complementary OCx and OCxN channels with break feature

Control bit					Output status	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx output state	OCxN output state
1	X	0	0	0	Output disabled (not driven by the timer), OCx=0, OCx_EN=0	Output disabled (not driven by the timer), OCxN=0, OCxN_EN=0
		0	0	1	Output disabled (not driven by the timer), OCx=0, OCx_EN=0	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1

		0	1	0	OCxREF + Polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not driven by the timer), OCxN=0, OCxN_EN=0
		0	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
		1	0	0	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
		1	0	1	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=1	OCxREF+Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+Polarity OCx=OCxREF xor CCxP, OCx_EN=1	Off-State (output enabled with inactive state), OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCN_EN=1
0	0	X	0	0	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
	0		0	1	Output disabled (not driven by the timer anymore). Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 If the clock ispresent: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	0		1	0		
	0		1	1		
	1		0	0	Output disabled (not driven by the timer anymore). OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 If the clock ispresent: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	1		1	0		
	1		1	1		

18.4.10. TIM1 Calculator (TIM1 _ 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	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															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

18.4.11. TIM1 Prescaler (TIM1 _ PSC)

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**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value

				The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). The PSC contains the value loaded into the current pre-scaler register when an update event occurs; Update event includes counter Clear 0 by the UG bit of TIM _ EGR or by the slave con-troller operating in reset mode.

18.4.12. TIM1 automatic reload register (TIM1 _ 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	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**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

18.4.13. TIM1 Repetition counter register (TIM1_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	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	REP[7:0]							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	0	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 send from the preload Register transfer to the current register); If an update inter-rupt is allowed, it will also affect the rate at which the up-date interrupt is generated. Each time the REP_CNT related downcounter reaches zero, an update event is generated, and it restarts count-ing from REP value. Since REP _ CNT overloads the REP value only when the cycle update event U _ RC occurs, the new value written to the TIM1 _ RCR register only takes effect when the next cycle update event occurs. It means 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.

18.4.14. TIM1 capture/compare register 1 (TIM1_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	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**

CCR1 [15:0]
RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). 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 TIM1_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

18.4.15. TIM1 capture/compare register 2 (TIM1 _ 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	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**
CCR2 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR2 [15:0]	RW	0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR2 register (bit OC2PE). 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 TIM1_CNT and signaled on OC output. CC2 channel configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

18.4.16. TIM1 capture/compare register 3 (TIM1_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	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**
CCR3 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:0	CCR3 [15:0]	RW	0	Capture/Compare 3 value If channel CC3 is configured as output:

Bit	Name	R/W	Reset Value	Function
				CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR3 register (bit OC3PE). 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 TIM1_CNT and signaled on OC output. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

18.4.17. TIM1 capture/compare register 4 (TIM1 _ 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	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**
CCR4 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31： 16	Reserved	-	-	Reserved
15:0	CCR4 [15:0]	RW	0	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR4 register (bit OC4PE). 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 TIM1_CNT and signaled on OC output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

18.4.18. TIM1 break and dead-time register (TIM1_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	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**
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
31:16	Reserved	-	-	Reserved
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is cleared by software or automatically setting depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIM1_CCER register).

Bit	Name	R/W	Reset Value	Function
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
12	BKE	RW	0	Break enable 0: Disable brake input (BRK and BRK _ ACTH); 1: Turn on the brake input (BRK and BRK _ ACTH). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
11	OSSR	RW	0	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 enabling (Section 12.5.9, Capture/Compare Enable Register (TIM1 _ CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
10	OSSI	RW	0	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 enabling (Section 12.5.9, Capture/Compare Enable Register (TIM1 _ CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are first forced to their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: LOCK Level 1, DTG, BKE, BKP, AOE bits in TIM1_BDTR register and OISx and OISxN bits in TIM1_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIM1_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIM1_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIM1_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	0000 0000	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] × T_{dtg}， $T_{dtg} = T_{DTS}$；

Bit	Name	R/W	Reset Value	Function
				DTG[7:5]=10x => DT=(64+DTG[5:0]) × T_{dtg}， T_{dtg} = 2 × T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) × T_{dtg}， T_{dtg} = 8 × T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) × T_{dtg}， T_{dtg} = 16 × T_{DTS}； Example: If TDTS = 125ns (8MHZ), the possible dead time is: 0 to 15875 ns by 125 ns steps; 16us to 31750ns, if the step time is 250ns; 32us to 63us, if the step time is 1us; 64 us to 126 us by 2 us steps Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).

18.4.19. TIM1 DMA control register (TIM1_DCR)

Address offset: 0x48

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	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	-	-	Reserved
12:8	DBL[4:0]	RW	0 0000	DMA continuous transfer length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the address of the TIM1 _ DMAR register When, the timer performs a continuous transmission), that is, defines the number of bytes to be transmitted: 00000: 1 transmission 00001: 2 transmissions 00010: 3 transmissions .... 10001: 18 transmissions
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	0 0000	DBA [4: 0]: DMA Base address These bits define the base address of the DMA in continuous mode (when reading or writing to the address of the TIM1 _ DMAR register 00000: TIM1 _ CR1 00001: TIM1 _ CR2 00010: TIM1 _ SMCR ......

18.4.20. DMA address of TIM1 continuous mode (TIM1 _ DMAR)

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	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	DMAB[31:0]	RW	0	DMA continuous transfer register A read or write to the TIM1 _ DMAR register results in an access operation to a register at the following address: TIM1 _ CR1 address + DBA + DMA pointer, where: The "TIM1 _ CR1 address" is the address of the control register 1; "DBA" is the base address defined in the TIM1 _ DCR register; The "DMA pointer" is an offset automatically controlled by the DMA, which depends on the DBL defined in the TIM1 _ DCR register.

Note: When using the DMA continuous transfer function, the value of the CNDTR register of the corresponding channel in the DMA must be corresponded to the value of the DBL in the TIMx _ DCR register, otherwise it will not be used properly.

19. General-purpose timers (TIM2)

19.1. Introduction to TIM2

The TIM2 universal timer is composed of a 32-bit automatic reload counter driven by a programmable frequency divider.

It is suitable for a variety of occasions, including measuring the pulse length of input and output signals (input capture) or generating output waveforms (output comparison and PWM).

Using a timer prescaler and an RCC clock controller prescaler, the pulse length and waveform period can be adjusted from a few subtle to a few milliseconds.

Each timer is completely independent and does not share any resources with each other. They can operate in sync together.

19.2. TIM2 Main Features

General purpose TIM2 timer features include:

- 32-bit (TIM2) up, down, up-down auto-load counter
- 16-bit programmable (on the fly) prescaler, the division coefficient of the counter clock frequency is any value between 1 and 65536
- 4 independent channels
  - Input capture
  - Output Compare
  - PWM generation (edge or middle alignment mode)
  - One-pulse mode output
- Synchronization circuitry that can synchronously control the timer and other internally connected timers using external signals
- Interrupt/DMA is generated when the following events occur
  - Update: Counter upflow/downflow, counter initialization (via software or internal/external trigger)
  - Trigger event (counter start, stop, initialize or count by internal/external trigger)
  - Input capture
  - Output Compare
- Trigger input for external clock or cycle-by-cycle current management

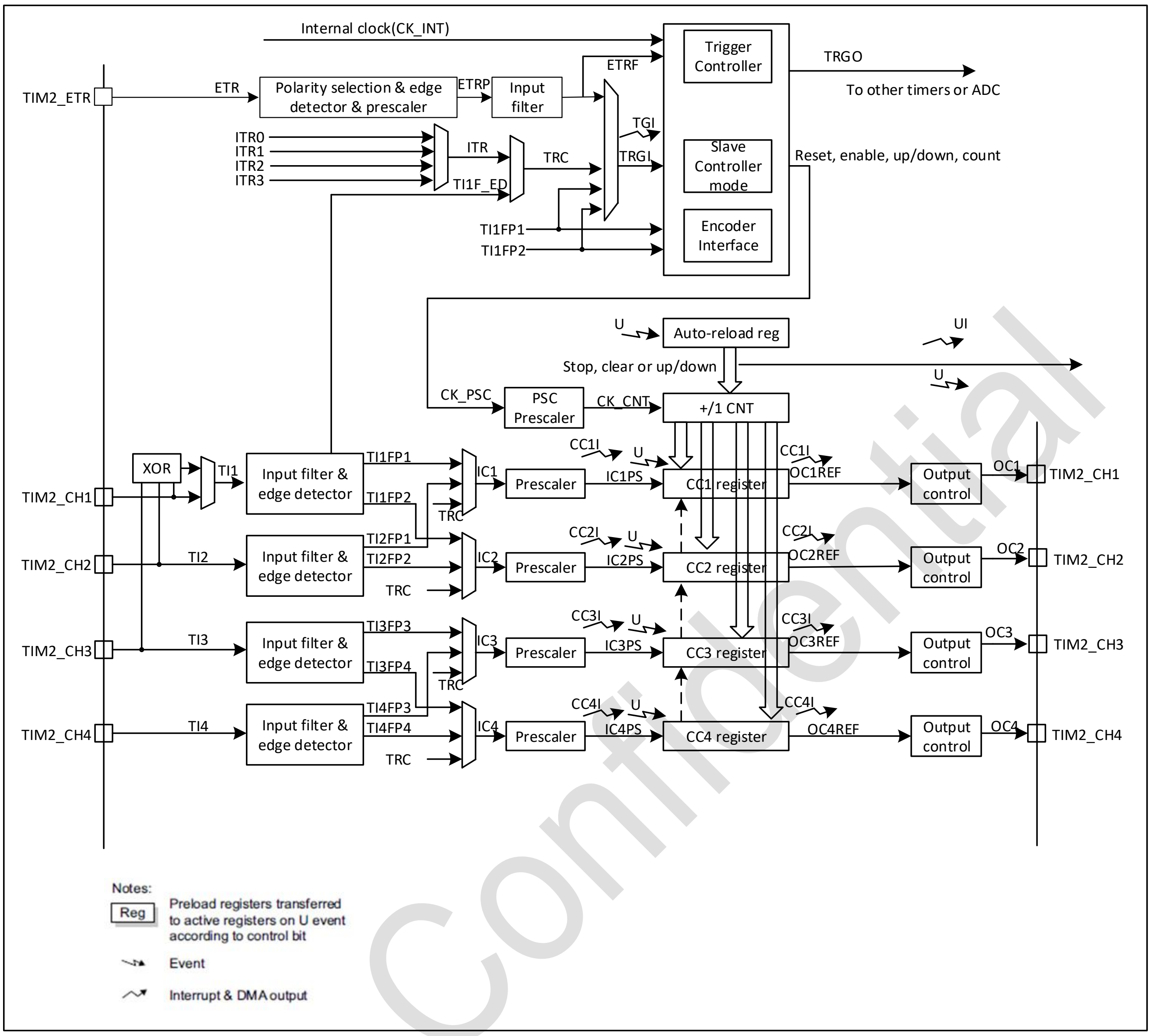


Figure 19-1 General-purpose timer architecture (TIM2)

19.3. TIM2 functional description

19.3.1. Time-base unit

The main part of the TIM2 timer is a 32-bit counter and its associated autoload register. The counter can count up, down or both up and down. The counter clock can 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 includes:

- Counter register (TIM2_CNT)
- Prescaler register (TIM2_PSC)
- Auto-reload register (TIM1_ARR)

The auto-load register is preloaded. Writing to or reading from the auto-load register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM2_CR1 register. The update event is sent when the counter reaches the overflow (or under-flow when downcounting) and if the UDIS bit equals 0 in the TIM2_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in 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 counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM2_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

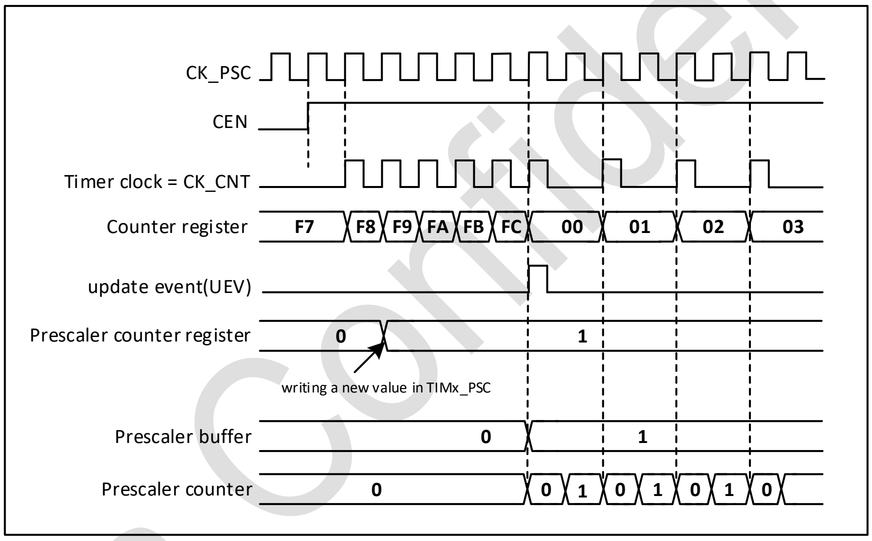


Figure 19-2 Counter timing diagram with prescaler division change from 1 to 2

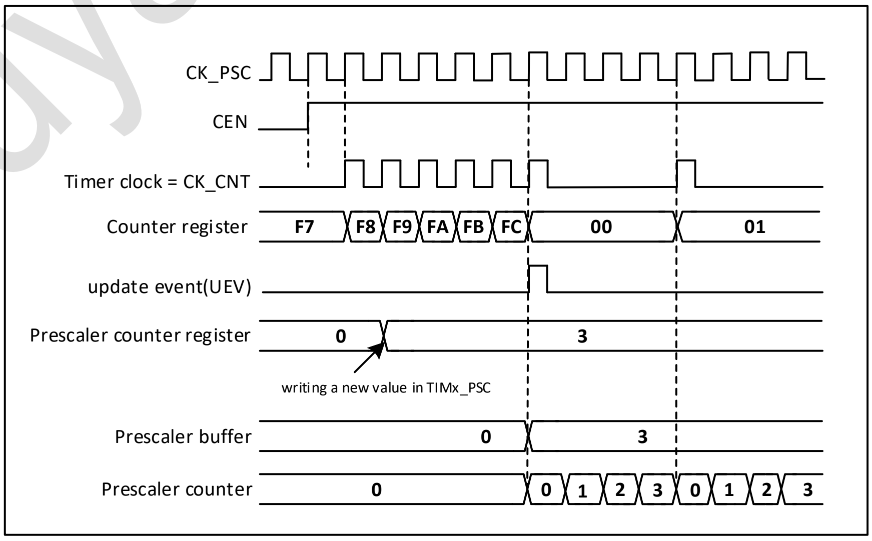


Figure 19-3 Counter timing diagram with prescaler division change from 1 to 4

19.3.2. Timer enable

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value, then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM2_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, when an update event should occur, the counter will still be cleared '0', and the count of the prescaler will also be called 0 (but the value of the prescaler will not change). In addition, if the URS bit in the TIM2 _ CR1 register is set (Select Update Request), setting the UG bit will generate an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on 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 figure below gives some examples of the action of the counter at different clock frequencies when TIM2 _ ARR = 0x36.

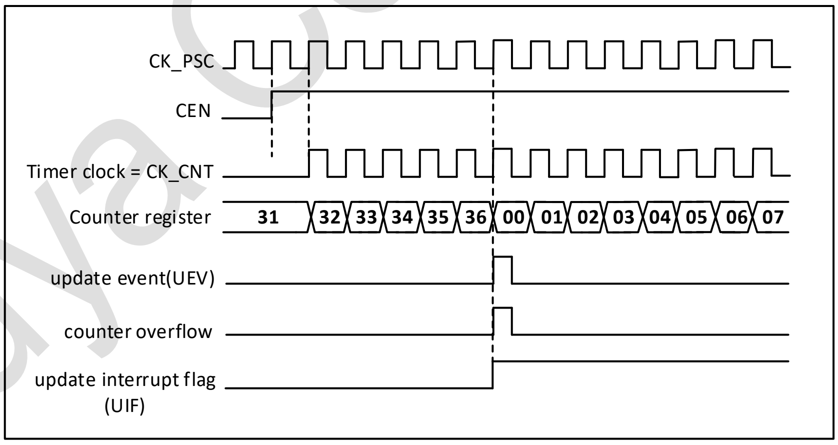


Figure 19-4 Counter timing diagram, internal clock divided by 1

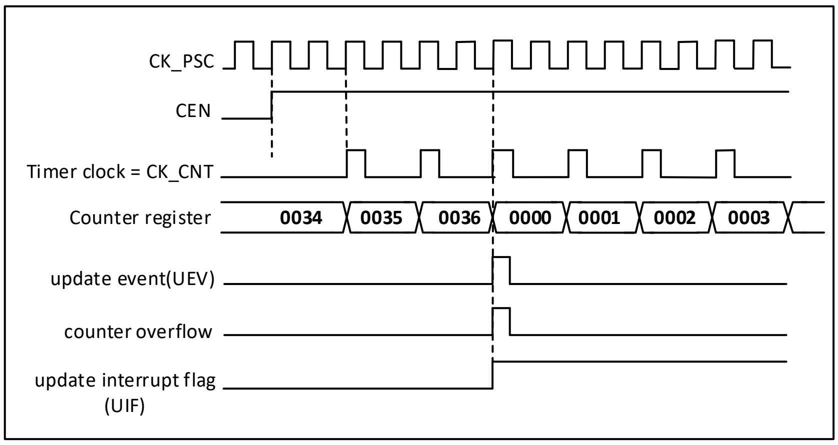


Figure 19-5 Counter timing diagram, internal clock divided by 2

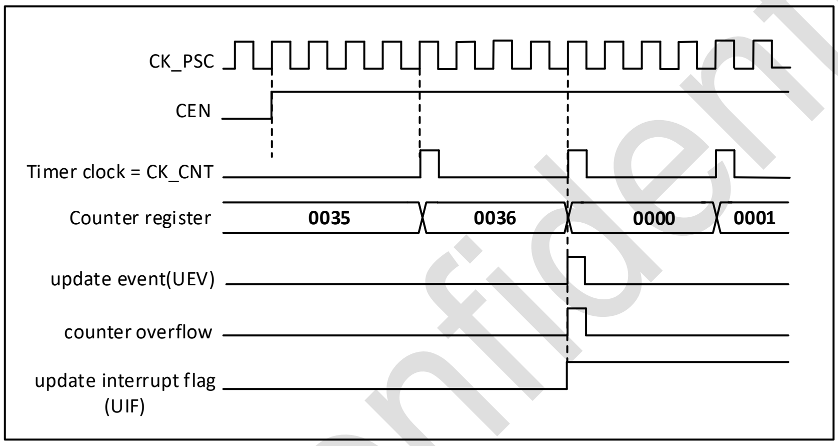


Figure 19-6 Counter timing diagram, internal clock divided by 4

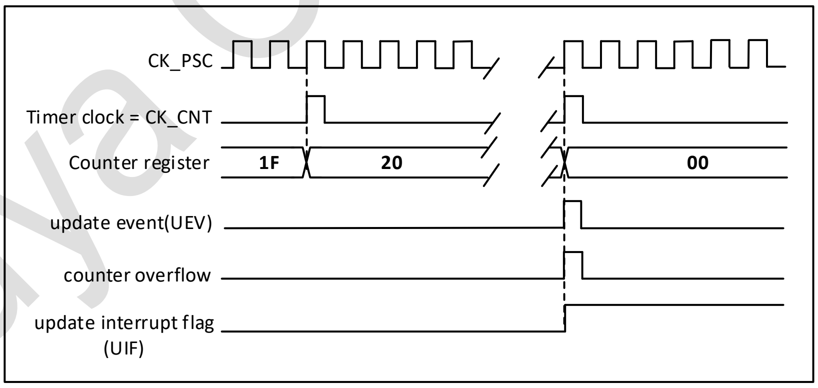


Figure 19-7 Counter timing diagram, internal clock divided by N

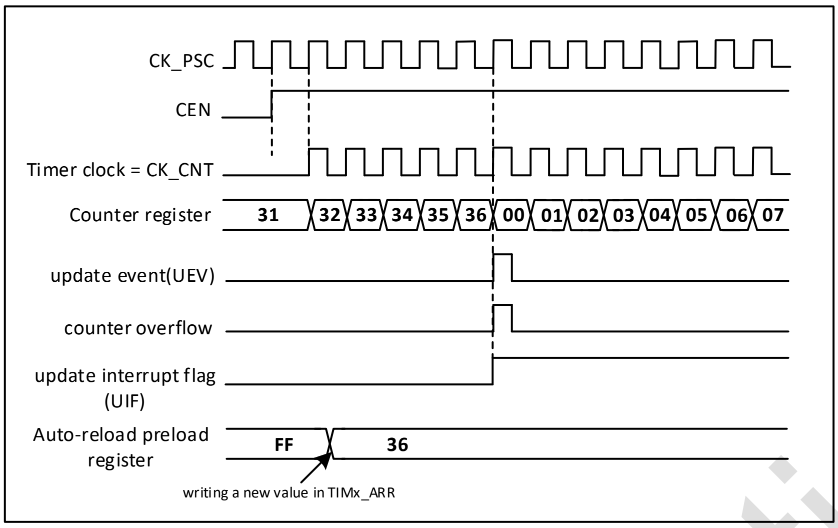


Figure 19-8 counter timing diagram, update event when ARPE = 0 (TIMx _ ARR is not pre-loaded)

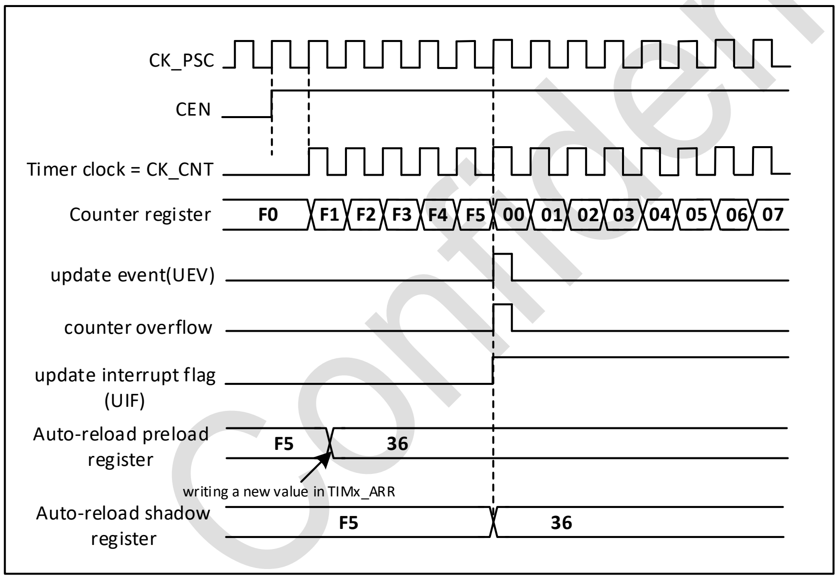


Figure 19-9 counter timing diagram, update event when ARPE = 1 (pre-loaded with TIM2 _ ARR)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value down to 0, then re-starts from the auto-reload value and generates a counter underflow event.

Setting the UG bit in the TIM2 _ EGR register (either by software or using a slave mode con-troller) can also generate an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM2_CR1 regis-ter. 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. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler re-starts from 0 (but the prescaler rate doesn’t change).

In addition, if the URS bit (update request selection) in TIM2_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and

DMA requests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is loaded with the preloaded value (the value of the TIM2 _ PSC register).
- The current autoload register is updated to the preload value (the contents of the TIM2 _ ARR register).

Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

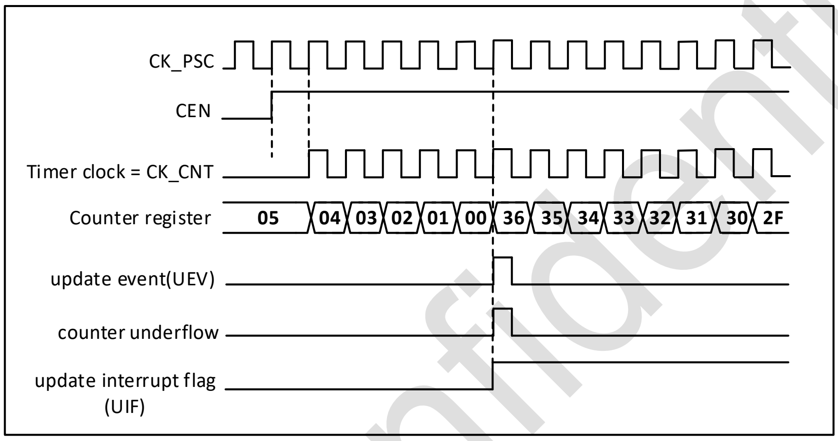


Figure 19-10 Counter timing diagram, internal clock divided by 1

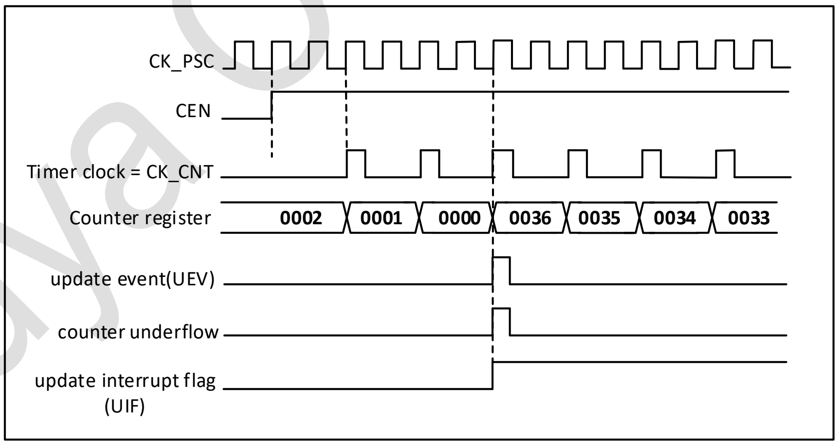


Figure 19-11 Counter timing diagram, internal clock divided by 2

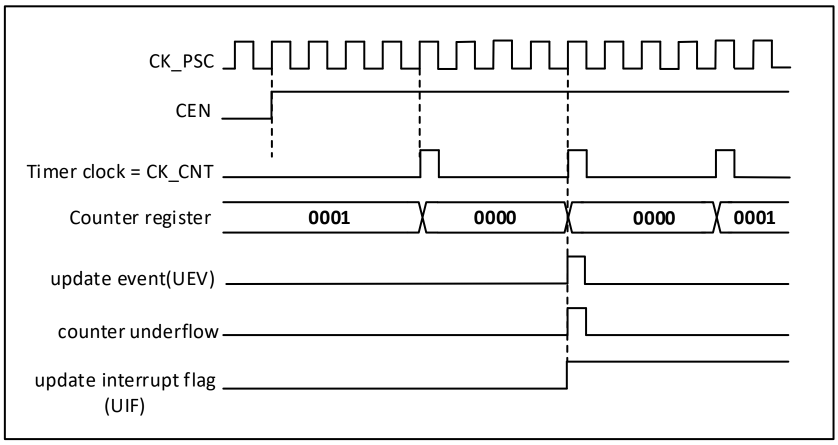


Figure 19-12 Counter timing diagram, internal clock divided by 4

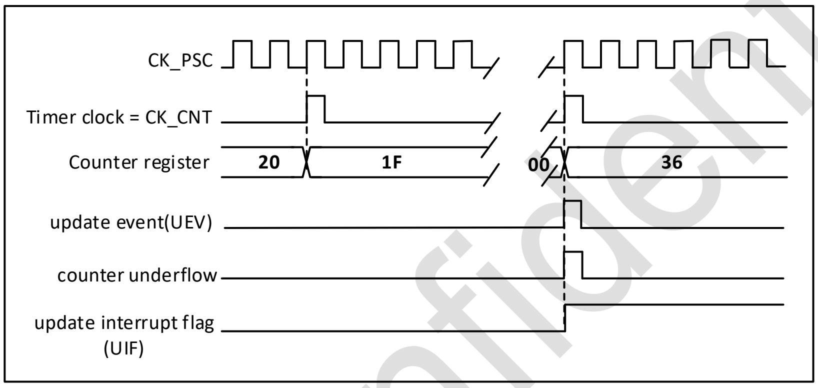


Figure 19-13 Counter timing diagram, internal clock divided by N

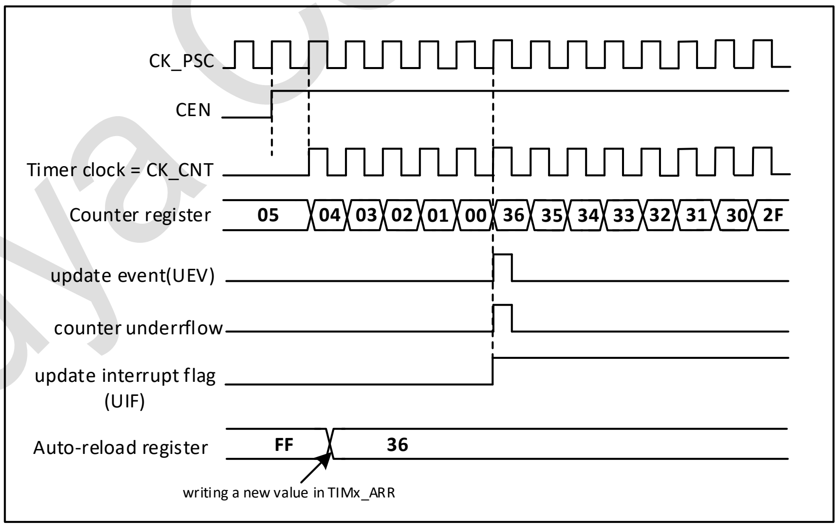


Figure 19-14 Counter timing diagram, update event when repetition counter is not used

Central alignment mode (up/down count)

In central alignment mode, the counter counts from 0 to the auto-loaded value (TIM2 _ ARR register) − 1, generating a counter overflow event, then counting down to 1, generating a counter underflow event, and then counting again from 0.

Center-aligned mode is active when the CMS bits in TIM2_CR1 register are not equal to '0'. The Output compare interrupt flag of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and down (Center aligned mode 3,CMS = “11”).

In this mode, the DIR direction bit in the TIM2_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

An update event can be generated on each count overflow and each count underflow; The update event can also be generated by setting the UG bit in the TIM2 _ EGR register (either software or using a slave mode controller). In this case, the counter restarts counting from0, as well as the counter of the prescaler.

The UEV event can be disabled by software by setting the UDIS bit in the TIM2_CR1 register. 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. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIM2_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt and DMA requests are sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The current autoload register is updated to the preload value (TIM2 _ ARR)

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).

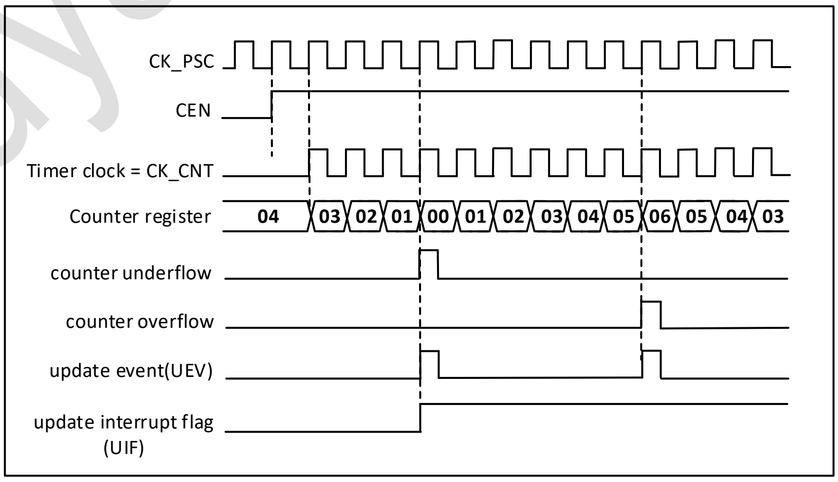


Figure 19-15 counter timing diagram with internal clock division factor of 1, TIM2 _ ARR = 0x6

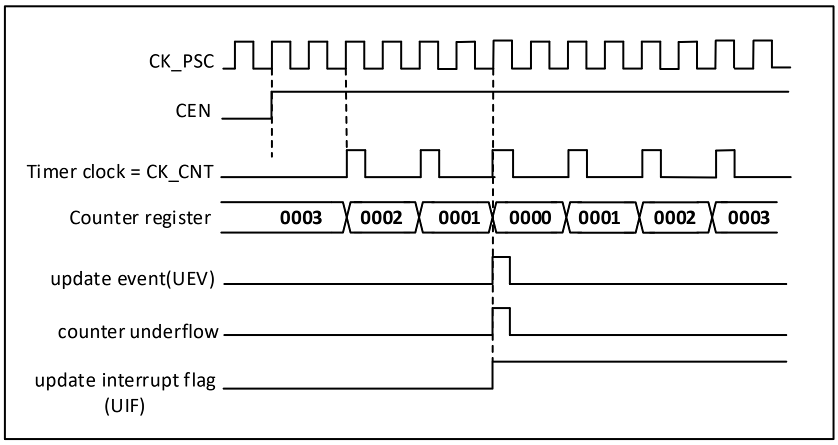


Figure 19-16 Counter timing diagram, internal clock divided by 2

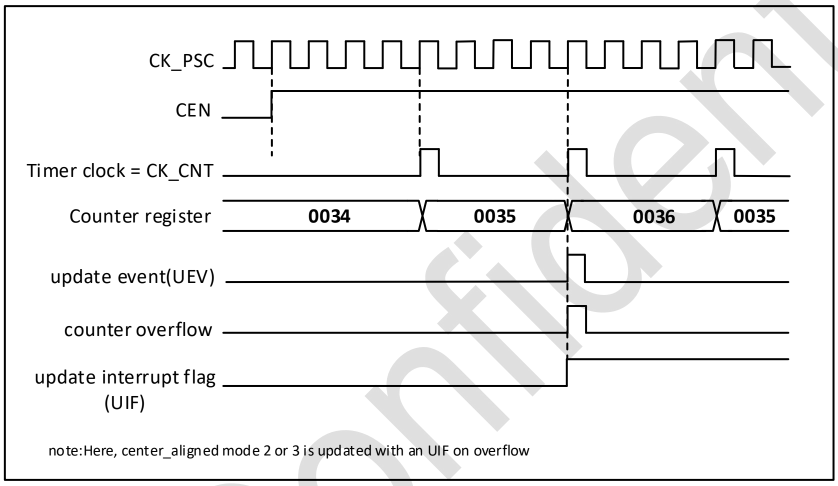

CK_PSC
CEN
Timer clock = CK_CNT
Counter register
0034
0035
0036
0035
update event(UEV)
counter overflow
update interrupt flag (UIF)
note:Here, center_aligned mode 2 or 3 is updated with an UIF on overflow

Figure 19-17 Counter timing diagram, internal clock division factor is 4, TIM2 _ ARR = 0x36

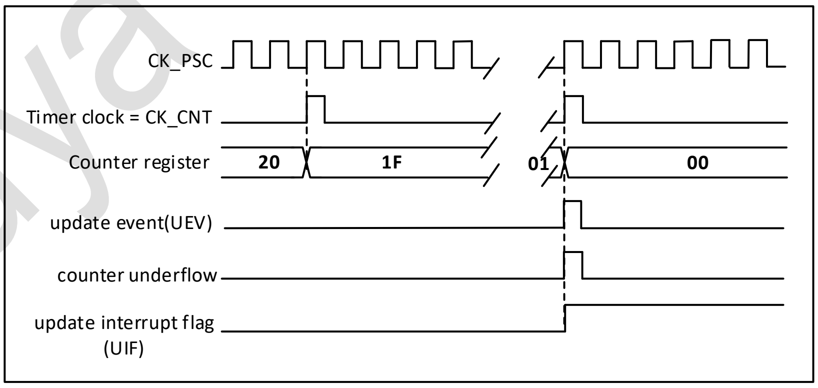


Figure 19-18 Counter timing diagram, internal clock divided by N

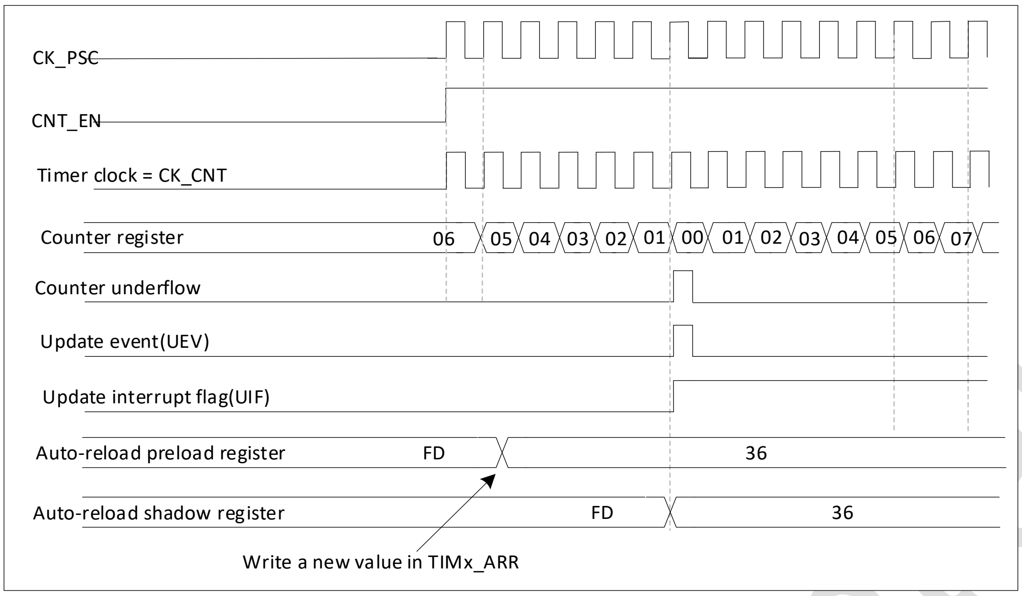


Figure 19-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

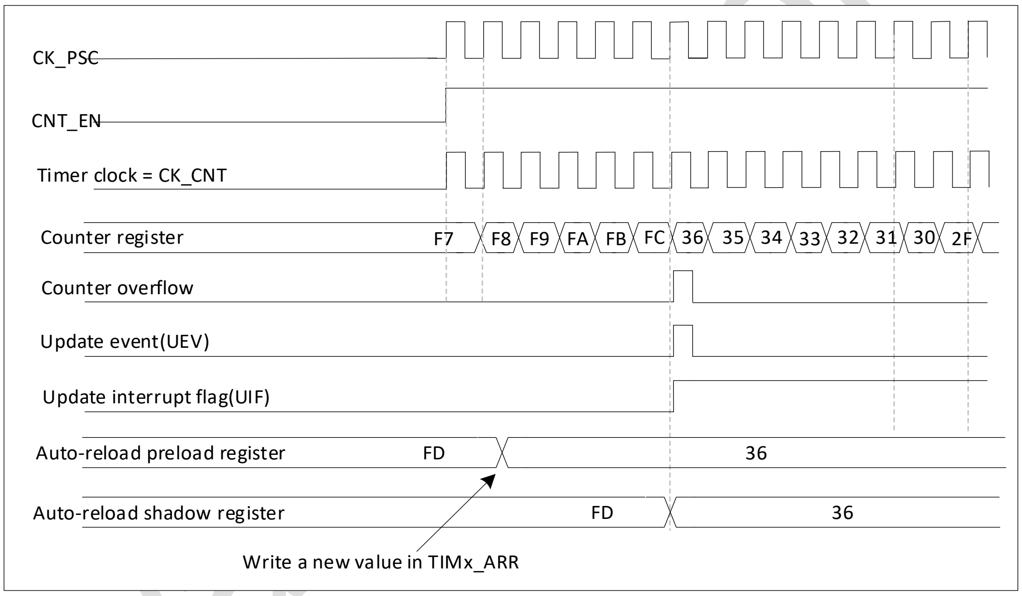


Figure 19-20 Counter timing diagram, Update event with ARPE=1 (counter overflow)

19.3.3. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, one timer Timer1 may be configured as a prescaler for another timer Timer3.

Internal clock source (CK_INT)

If the slave mode controller is disabled, the CEN, DIR (TIM2 _ CR1 register) and UG bits (TIM2 _ EGR register) are de facto control bits and can only be modified by software. As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

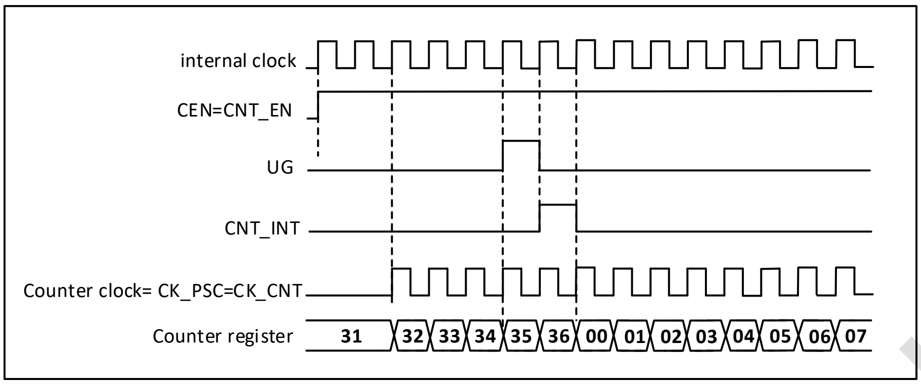


Figure 19-21 Control circuit in general mode with internal clock division factor of 1

External clock source mode 1

This mode is selected when SMS of the TIM2 _ SMCR register = 111. The counter can count at each rising or falling edge on a selected input.

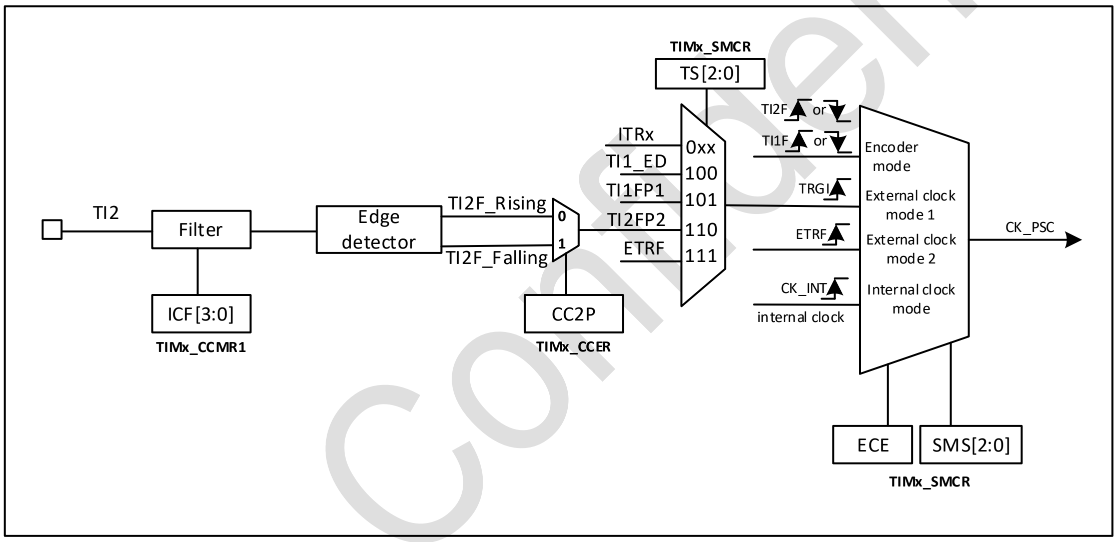


Figure 19-22 TI2 Example of external clock connection

For example, to configure the up counter to count on the rising edge of the T12 input, use the following steps:

1. Configure TIM2 _ CCMR1 register CC2S = 01, configure channel 2 to detect rising edge of TI2 input
2. Configure IC2F [3: 0] of TIM2 _ CCMR1 register, select input filter bandwidth (if no filter is needed, keep IC2F = 0000) Configure CC2P = 0 of TIM2 _ CCER register, select rising edge polarity
3. Configure SMS = 111 of TIM2 _ SMCR register, select timer external clock mode 1
4. Configure TS = 110 in the TIM2 _ SMCR register and select TI2 as the trigger input source
5. Set CEN = 1 of the TIM2 _ CR1 register to start the counter

Note: The capture prescaler is not used as a trigger, so it does not need to be configured

When the rising edge occurs at 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 re-synchronization circuit at the input of TI2.

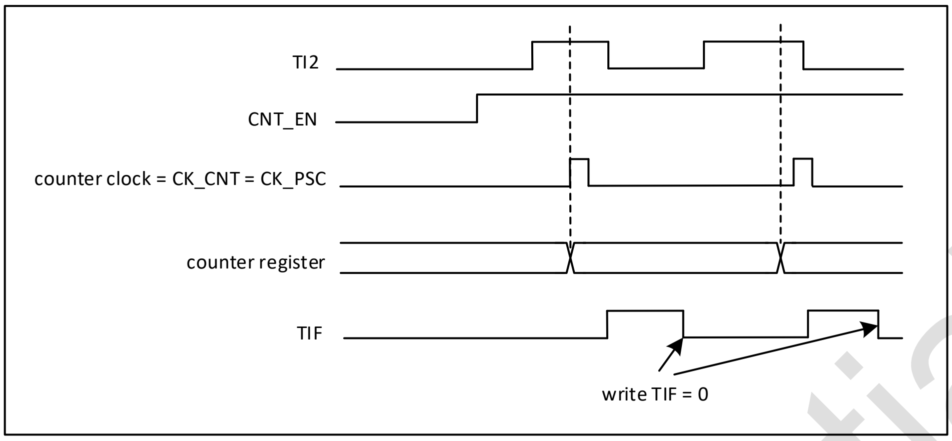


Figure 19-23 Control circuit in external clock mode 1

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:

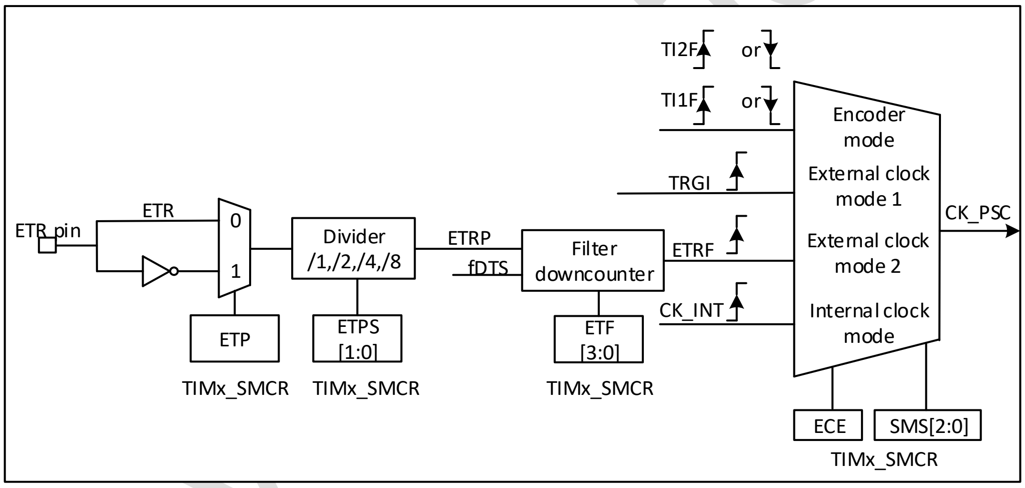


Figure 19-24 ETR External Clock Connection Example

For example, to configure an up-counter that counts every 2 rising edges under an ETR, use the following steps:

1. In this example, no filter is needed, 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 the counter, write CEN = 1 in the 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 resynchronization circuit of the ETRP signal.

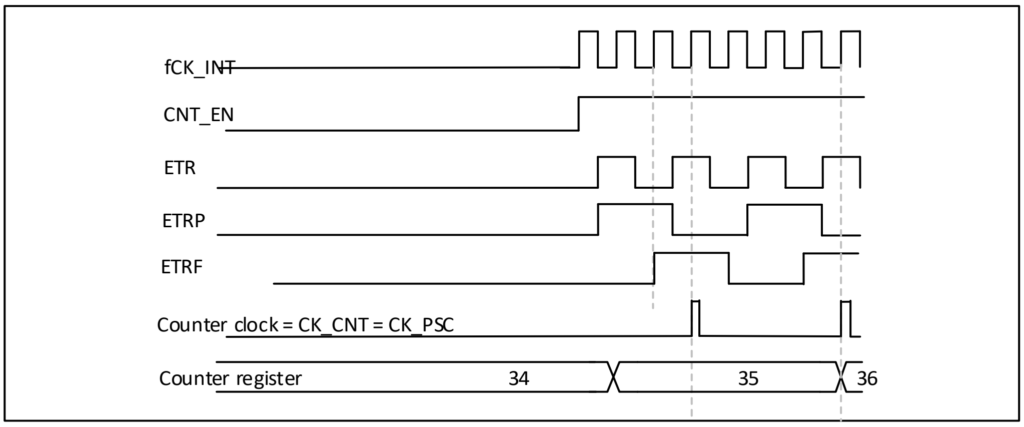


Figure 19-25 Control circuit in external clock mode 2

19.3.4. Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), a input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

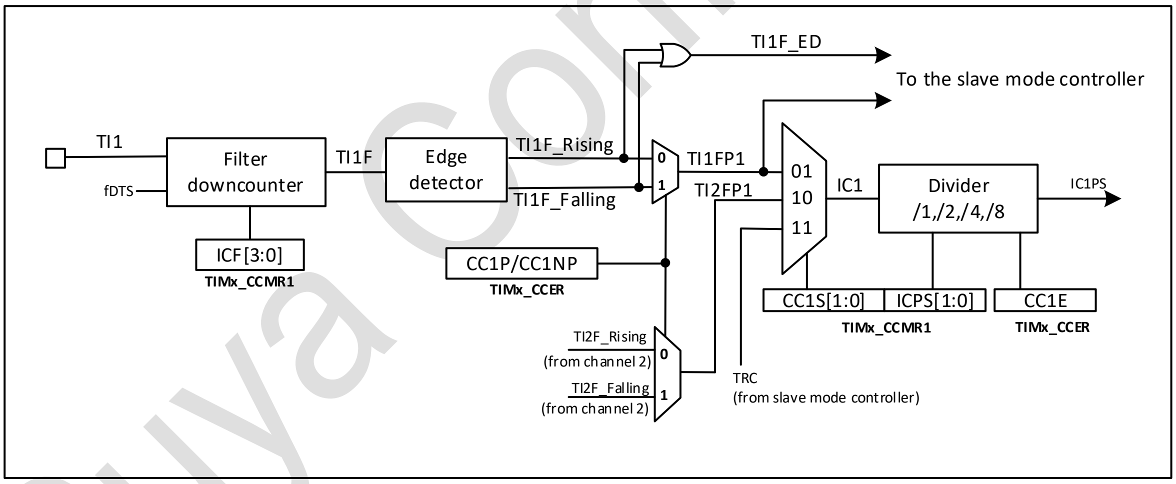


Figure 19-26 Capture/compare channel (example: channel 1 input stage)

The output section produces an intermediate waveform OCxRef (high significant) as a reference, and the end of the chain determines the polarity of the final output signal.

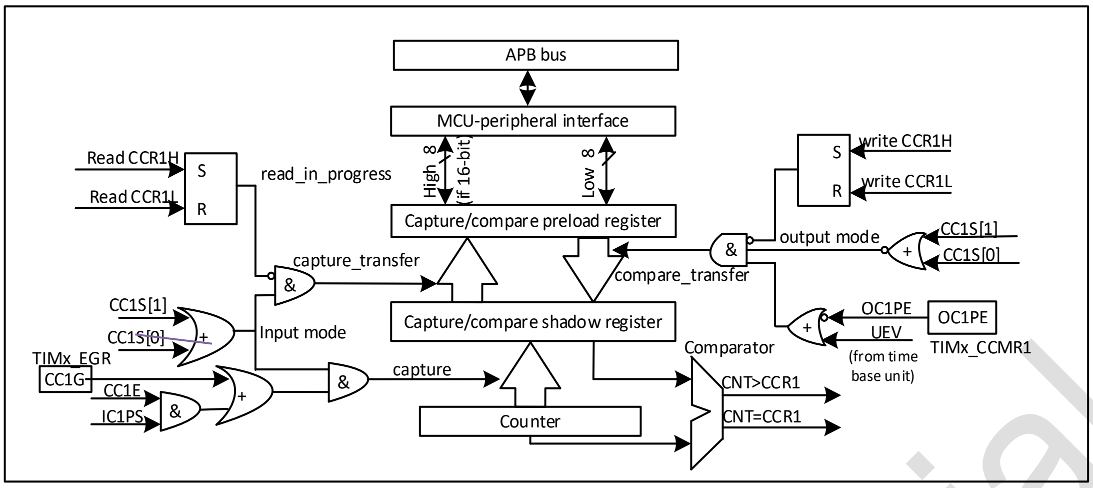


Figure 19-27 Capture/compare channel 1 main circuit

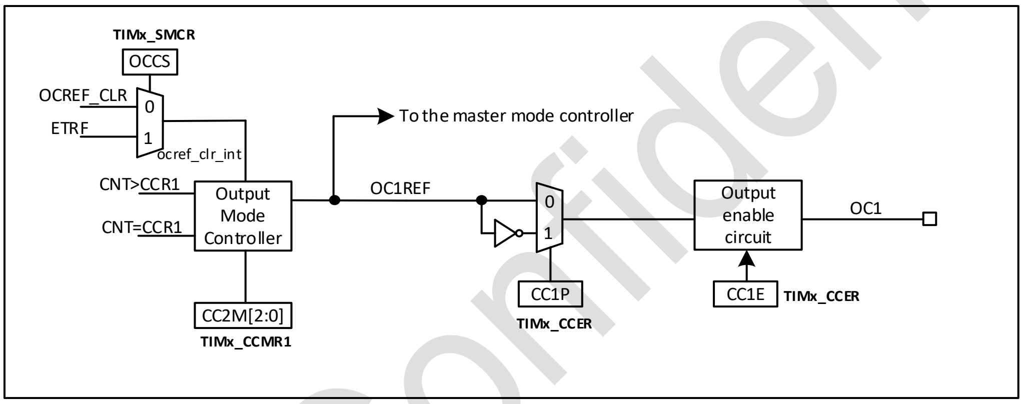


Figure 19-28 Capture/Compare the output portion of the channel (channel 1)

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.

19.3.5. 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. When a capture event occurs, the corresponding CcxIF flag (TIM2 _ SR register) is set to 1, and if interrupt and DMA operations are turned on, an interrupt or DMA request will be generated. If the CcxIF flag is already high when the capture event occurs, the repeat capture flag CcxOF (TIMx _ SR register) is set to 1. Writing CcxIF = 0 clears CcxIF, or reading captured data stored in the TIM2 _ CCRx register also clears CcxIF. Writing CcxOF = 0 clears CcxOF.

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: TIM2 _ CCR1 must be connected to the TI1 input, so write CC1S = 01 in the TIMx _ CCR1 register, as long as CC1S is not '00', the channel is configured as input, and the TIMx _ CCR1 register becomes read-only.
- According to the characteristics of the input signal, the input filter is configured to the required bandwidth (i.e., when the input is Tix, the input filter control bit is the IcxF bit in the TIM2 _ 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 TIM2_CCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (rising edge) (and CC1NP = 0) in the TIM2 _ 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 TIM2_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If necessary, associated interrupt requests are allowed by setting the CC1IE bit in the TIMx _ DIER register, and DMA requests are allowed by setting the CC1DE bit in the TIMx _ DIER register.

When an input capture occurs:

- The TIM2_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag 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.
- If the CC1DE bit is set, a DMA request will also be generated.

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: An input capture interrupt and/or DMA request can be generated by software by setting the corresponding CCxG bit in the TIM2 _ EGR register.

19.3.6. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Both Icx signals are mapped to the same Tix input.
- These two Icx signals are edge active, but have opposite polarities.
- One of the TixFP signals is used as the trigger input signal, and the slave mode controller is configured to reset mode.

For example, you can measure the period (in TIM2_CCR1 register) and the duty cycle (in TIM2_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIM2_CCR1: write the CC1S bits to 01 in the TIM2_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIM2_CCR1 and counter clear): write the CC1P bit to ‘0’ (active on rising edge).
- Select the active input for TIM2_CCR2: write the CC2S bits to 10 in the TIM2_CCMR1 register (TI1 selected).
- Select the effective polarity of TI1FP2 (capture data to TIM2 _ CCR2): set CC2P = 1 (falling edge active).
- Select a valid trigger input signal: Set TS = 101 in the TIM2 _ SMCR register (select TI1FP1).
- Configure the slave mode controller to reset mode: Set SMS in TIM2 _ SMCR = 100.
- Enable capture: Set CC1E = 1 and CC2E = 1 in the TIM2 _ CCER register.

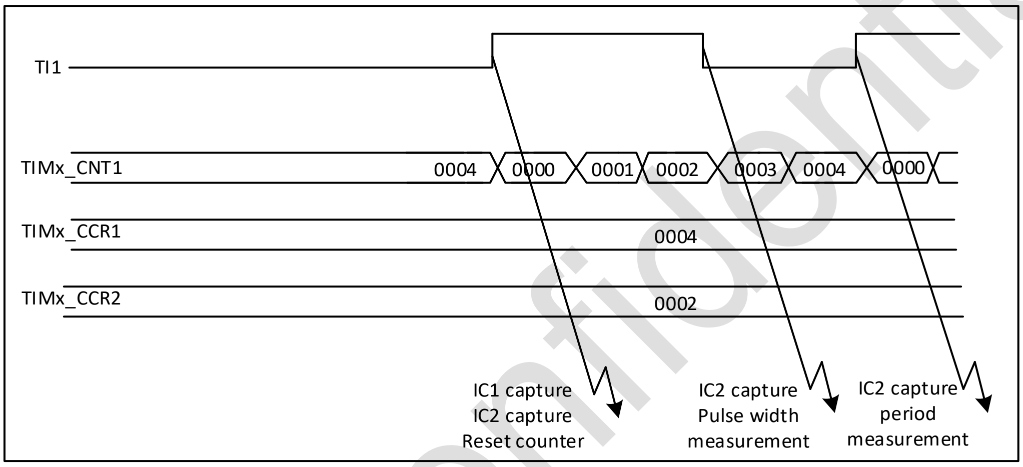


Figure 19-29 PWM input mode timing

19.3.7. Forced output mode

In the output mode (CCxS = 00 in the TIM2 _ CCMRx register), the output comparison signal (OCxREF and corresponding OCx) can be directly forced into an active or invalid state by the software, regardless of the result of the comparison between the output comparison 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 and DMA requests can be sent accordingly. This is described in the output compare mode section below.

19.3.8. 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:

- The values defined by the output comparison mode (OCxM bit in the TIM2 _ CCMRx register) and the output polarity (CCxP bit in the TIM2 _ CCER register) are output to the corresponding pins. 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 the flag bit in the interrupt status register (CcxIF bit in the TIM2 _ SR register).
- If the corresponding interrupt mask (CcxIE bit in the TIM2 _ DIER register) is set, an interrupt is generated.
- If the corresponding enable bit is set (CcxDE bit in the TIM2 _ DIER register, CCDS bit in the TIM2 _ CR2 register selects the DMA request function), a DMA request is generated.

The TIM2_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM2_CCMRx register.

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 corresponding data into the TIM2 _ ARR and TIM2 _ CCRx registers.
3. If you want to generate an interrupt request, set the CcxIE bit.
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 TIM2 _ 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.

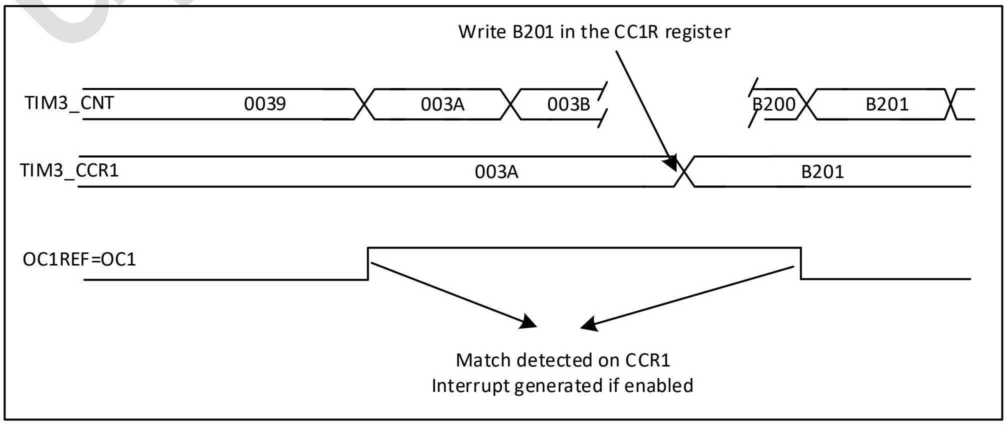


Figure 19-30 Output compare mode, toggle on OC1

19.3.9. Pulse width modulation (PWM) mode

The PWM pulse width modulation mode may allow generation of a signal having a frequency determined by the TIMx _ ARR register and a duty cycle determined by the TIMx _ CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIM2_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIM2_EGR register.

The polarity of OCx can be set by software in the CCxP bit in the TIM2 _ CCER register, which can be set to high active or low active. The output of OCx is enabled to be controlled by a combination of CCxE, CCxNE, MOE, OSSI, and OSSR bits (in the TIM2 _ CCER and TIM2 _ BDTR registers). See the description of the TIM2 _ CCER register for details.

In the PWM mode (Mode 1 or Mode 2), TIM2 _ CNT and TIM2 _ CCRx are always compared (depending on the counting direction of the counter) to determine whether TIM2 _ CCRx ≤ TIM2 _ CNT or TIM2 _ CNT ≤ TIM2 _ CCRx is compliant.

Nevertheless, in order to comply with the OCREF CLR function, OCREF CLR may only be given as follows:

1. When the comparison results change.
2. In the output comparison mode (configured by frozen (OCxM = 000) to switch to any PWM mode (OCxM = "110" or "111") by setting (OCxM of the TIM2 _ CCMRx register).

This is a software forced switch to PWM mode while the timer is running.

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 TIM2_CR1 register is low. In the following example, we consider PWM mode 1. When TIMx _ CNT < TIM2 _ CCRx, the PWM reference signal OCxREF is high, otherwise it is low. If the comparison value in TIM2 _ CCRx is greater than the auto-reload value (TIMx _ ARR), OCxREF remains' 1 '. If the compare value is 0 then OCxRef is held at ‘0’. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

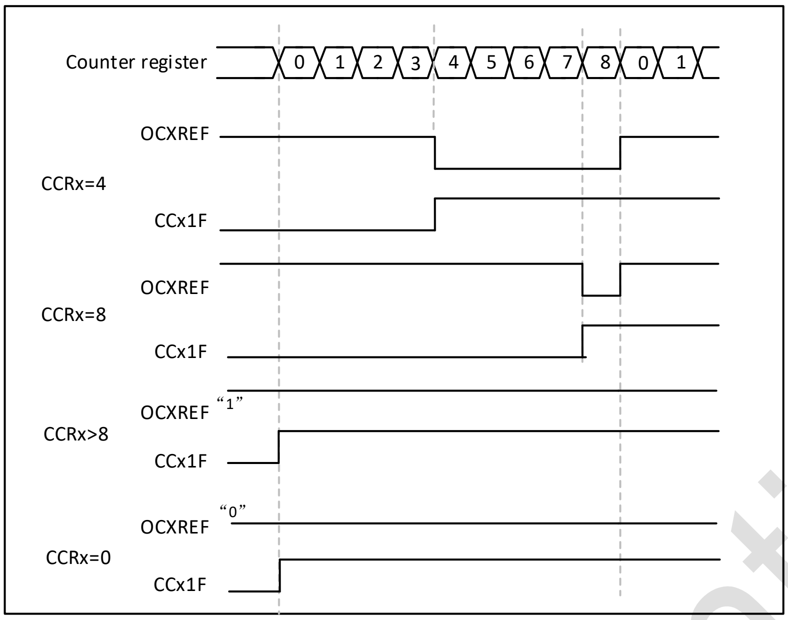


Figure 19-31 edge-aligned PWM waveform (ARR = 8)

- **Down count configuration**

Downcounting is active when DIR bit in TIM2_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low when TIM2 _ CNT > TIM2 _ CCRx, and high otherwise. If the comparison value in TIMx _ CCRx is greater than the auto-reload value in TIM2 _ ARR, OCxREF remains' 1 '. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIM2_CR1 register are different from‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals). 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 TIM2_CR1 register is up-dated by hardware and must not be changed by software.

Figure below shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 01 selected for CMS=01 in TIM2_CR1 register.

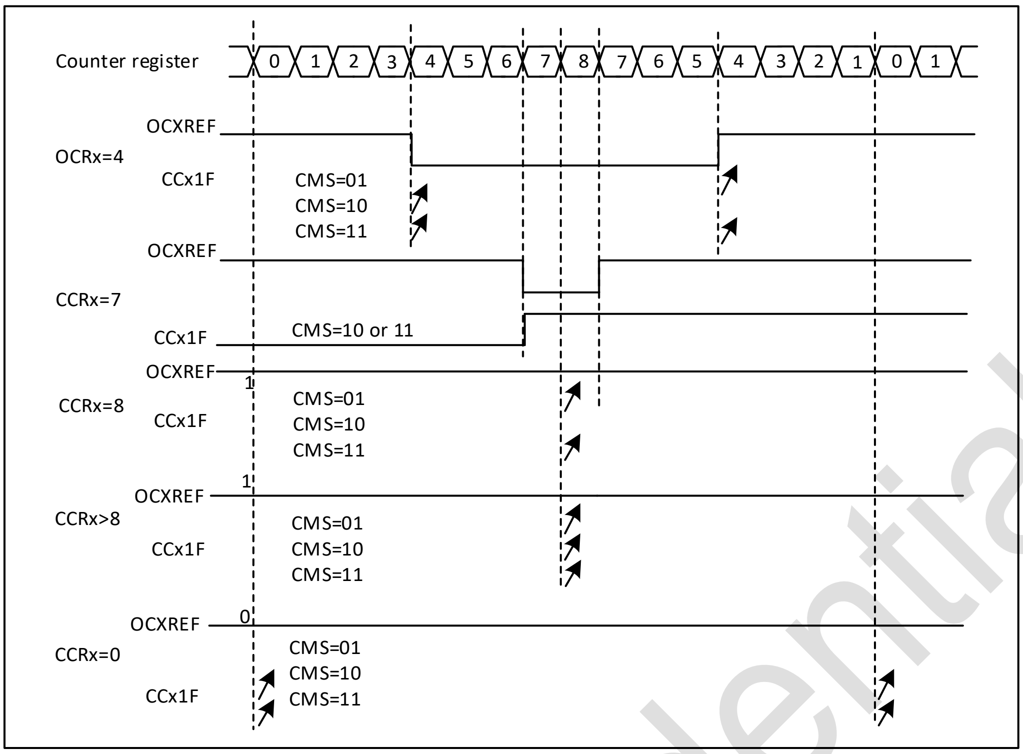


Figure 19-32 Center-aligned PWM waveforms (ARR=8)

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 TIM2_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. Specifically:
  - If the value of the write counter is greater than the value of the auto-reload (TIM2 _ CNT > TIM2 _ 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 TIM2 _ ARR is written to the counter, the direction is updated, but no update event UEV is generated.
- The safest way to use the central alignment mode is to generate a software update (set the UG bit in the TIM2 _ EGR bit) before starting the counter, and do not modify the value of the counter while the count is in progress.

19.3.10. 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. Setting the OPM bit of the TIM2 _ CR1 register will select the monopulse mode, which will allow the counter to automatically stop when the next update event UEV is generated.

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:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)
- In downcounting: CNT > CCRx

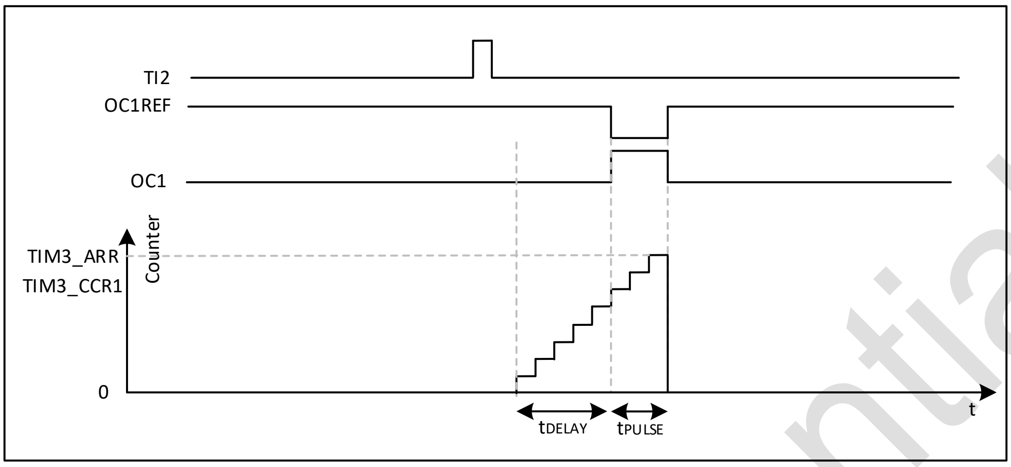


Figure 19-33 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIM2_CCMR1 register.
- Set CC2P = 0 in the TIM2 _ CCER register to enable TI2FP2 to detect rising edges.
- Set TS = 110 in the TIM2 _ SMCR register, and TI2FP2 is the trigger of the slave mode controller (TRGI).
- Set SMS = 110 (trigger mode) in the TIM2 _ SMCR register, and TI2FP2 is used to start the counter.

  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 TIM2_CCR1 register.
- tPULSE is defined by the difference between the auto-load value and the comparison value (TIM2 _ ARR-TIM2 _ CCR1).
- It is assumed that a waveform from 0 to 1 is to be generated when a comparison match occurs, and a waveform from 1 to 0 is to be generated when the counter reaches the preload value; First, set OC1M = 111 of the TIM2 _ CCMR1 register to enter PWM mode 2; Selectively enable preload registers as needed: set OC1PE = 1 in TIM2 _ CCMR1 and ARPE in TIM2 _ CR1 register; Then fill in the comparison value in the TIM2 _ CCR1 register, fill in the autoload value in the TIM2 _ ARR register, set the UG bit to generate an update event, and then wait for an external trigger event on TI2. CC1P is written to '0' in this example.

  In our example, the DIR and CMS bits in the TIM2_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIM2_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

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 t_{DELAY} min we can get.

If you want to output the waveform with minimum delay, you can set the OCxFE bit in the TIM2 _ CCMRx register; At this time, OCxREF (and OCx) responds directly to the excitation and no longer depends on the result of the comparison, and the output waveform is the same as the waveform when the comparison is matched. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

19.3.11. Encoder interface mode

To select Encoder Interface mode: write SMS='001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS=010 if it is counting on TI1 edges only and SMS=011 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, 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 and TI2FP2 are the signals of TI1 and TI2 after passing through the input filter and polarity control; If there is no filtering and disguised phase, then TI1FP1 = TI1 and TI2FP2 = TI2. 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 only counts continuously between 0 and the autoload value of the TIM2 _ ARR register (either 0 to ARR count or ARR to 0 count depending on the direction). Therefore, TIMx _ ARR must be configured before starting counting; Similarly, the trappers, comparators, prescalers, trigger output characteristics, etc. remain operating as usual. 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 19-1 Counting direction versus encoder signals

Effective edge	The level of the relative signal (the relative signal of TI1FP1 is TI2 and the relative signal of TI2FP2 is TI1)	TI1FP1 signal		TI2FP2 signal	
		rise	descend	rise	descend
Count only at TI1	tall	Count down	Count up	Not counting	Not counting
	Low	Count up	Count down	Not counting	Not counting
Count only at TI2	tall	Not counting	Not counting	Count up	Count down
	Low	Not counting	Not counting	Count down	Count up
Count on TI1 and TI2	tall	Count up	Count up	Count up	Count down
	Low	Count down	Count down	Count down	Count up

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' (TIM2_CCMR1 register, IC1FP1 mapped on TI1).
- CC2S='01' (TIM2_CCMR2 register, IC2FP2 mapped on TI2).
- CC1P = '0' (TIM2 _ CCER register, IC1FP1 is not inverted, IC1FP1 = TI1)
- CC2P = '0' (TIM2 _ CCER register, IC2FP2 is not inverted, IC2FP2 = TI2)
- SMS = '011' (TIM2 _ SMCR register, all inputs are valid on rising and falling edges).
- CEN='1' (TIM2_CR1 register, Counter enabled)

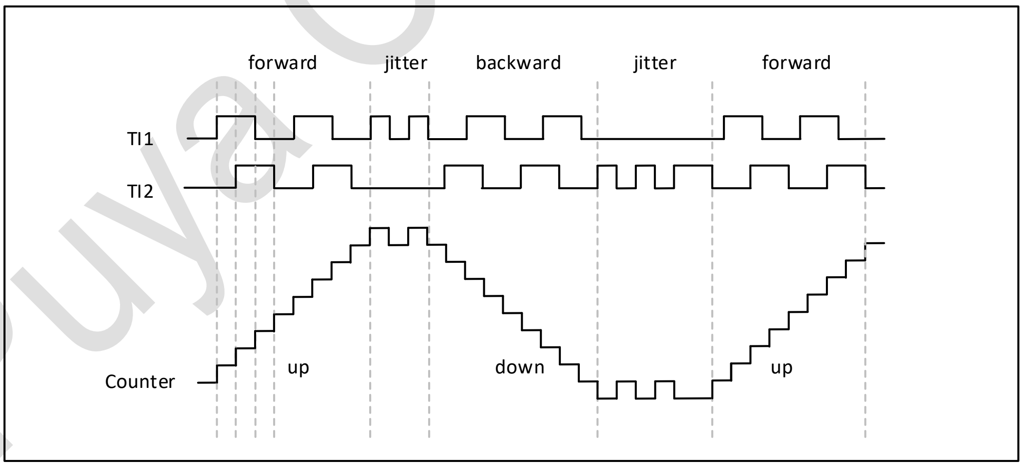


Figure 19-34 Example of counter operation in encoder interface mode

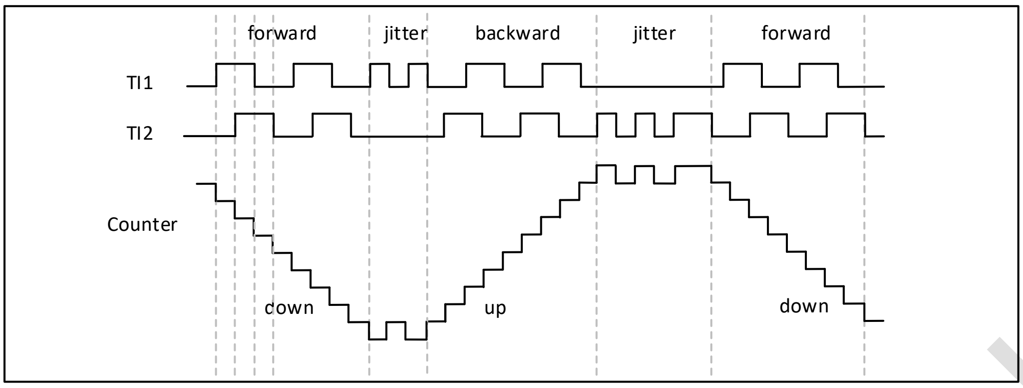


Figure19-35 TI1P1 inverted encoder interface mode example

The timer, when configured in Encoder Interface mode provides information on the sensor's current position. Dynamic information can be obtained (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the time between two events, the counter can also be read at regular times. If possible, the value of the counter can be latched 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.

19.3.12. Timer input XOR function

The TI1S bit in the TIM2_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM2_CH1, TIM2_CH2 and TIM2_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

One application example is the Hall sensor interface.

19.3.13. Synchronization of timer and externally triggered

The TIM2 timer can be synchronized with an external trigger in multiple modes: reset mode, gating mode and trigger mode.

Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; Meanwhile, if the URS bit of the TIM2 _ CR1 register is low, an update event UEV is also generated; All preload registers (TIM2 _ ARR, TIM2 _ 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. Set CC1P = 0 (and CC1NP = 0) in the TIMx _ CCER register to determine polarity (only the rising edge is detected).
- Set SMS = 100 in the TIM2 _ SMCR register and configure the timer to reset mode; Set TS = 101 in the TIMx _ SMCR register and select TI1 as the input source.

- Enable the counter by writing CEN=1 in the TIM2_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 TIM2 _ 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 TIM2 _ DIER register.

The following diagram shows the action when the register TIM2 _ ARR = 0x36 is automatically reloaded. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

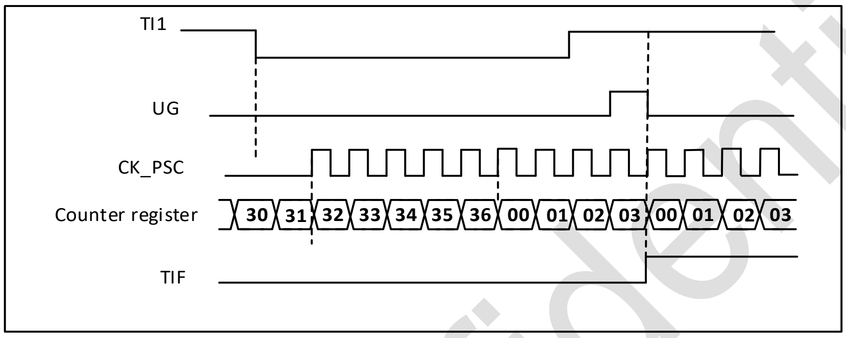


Figure 19-36 Control circuit in reset mode

Slave mode: Gated mode

Enables the counter according to the level of the 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. Set CC1P = 1 (and CC1NP = 0) in the TIM2 _ CCER register to determine polarity (only detect low level).
- Set SMS = 101 in the TIM2 _ SMCR register, and configure the timer to gating mode; Set TS = 101 in the TIM2 _ SMCR register and select TI1 as the input source.
- Enable the counter by writing CEN=1 in the TIM2_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 standard in TIM2 _ SR is set 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.

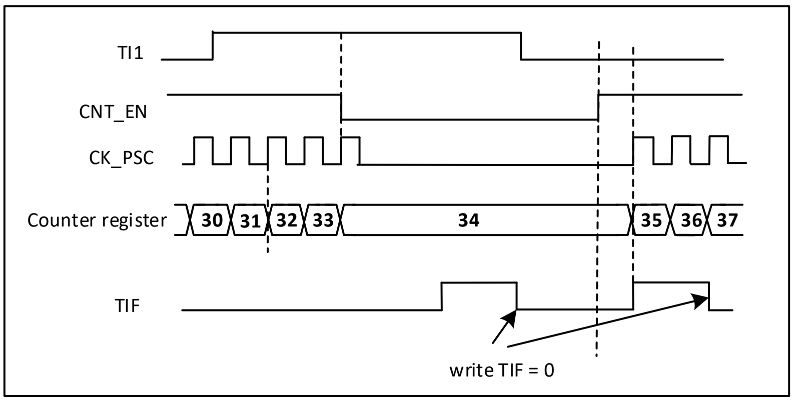


Figure 19-37 Control circuit in Gated mode

From 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 TIM2x_CCMR1 register. Set CC2P = 1 (and CC2NP = 0) in the TIM2 _ CCER register to determine polarity (only detect low level).
- Set SMS = 110 in the TIM2 _ SMCR register, and configure the timer to trigger mode; Set TS = 110 in the TIM2 _ SMCR register and select TI2 as the input source.

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.

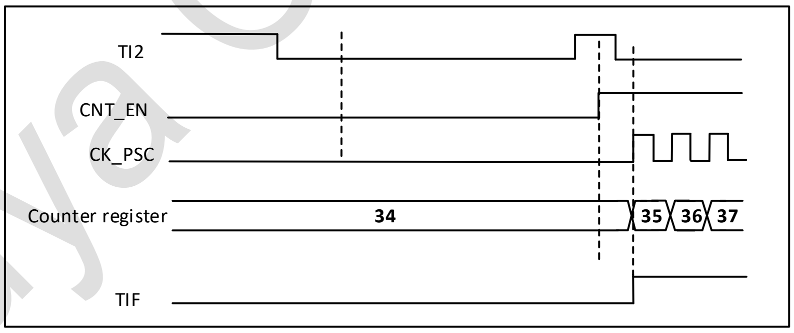


Figure 19-38 Control circuit in flip-flop mode

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 not recommended to select ETR as TRGI using the TS bit of the TIM2 _ 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. Memory configuration external trigger input circuit:
   - 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. Configure the channel 1 as follows, to detect rising edges on TI:
   - IC1F = 0000: no filter
   - The capture prescaler is not used for triggering, so it does not need to be configured.
   - CC1S=01 in TIM2_CCMR1 register to select only the input capture source
   - Set CC1P = 0 in the TIM2 _ CCER register to determine polarity (only the rising edge is detected)
3. Set SMS = 110 in the TIM2 _ SMCR register, and configure the timer to trigger mode. Set TS = 101 in the TIM2 _ SMCR register and select TI1 as the input source.

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.

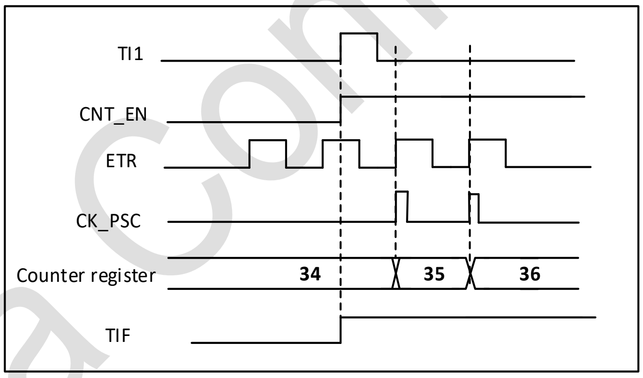


Figure 19-39 Control circuit in external clock mode 2 + trigger mode

19.3.14. Timer synchronization

The TIM timer is connected internally and is used for the synchronization or linking function of the timer. When one timer is in master mode, it can reset, start, stop, or clock the counter of another timer in slave mode.

The following diagram shows an overview of the Trigger Select and Master Mode Select modules.

Use one timer as a prescaler for the other

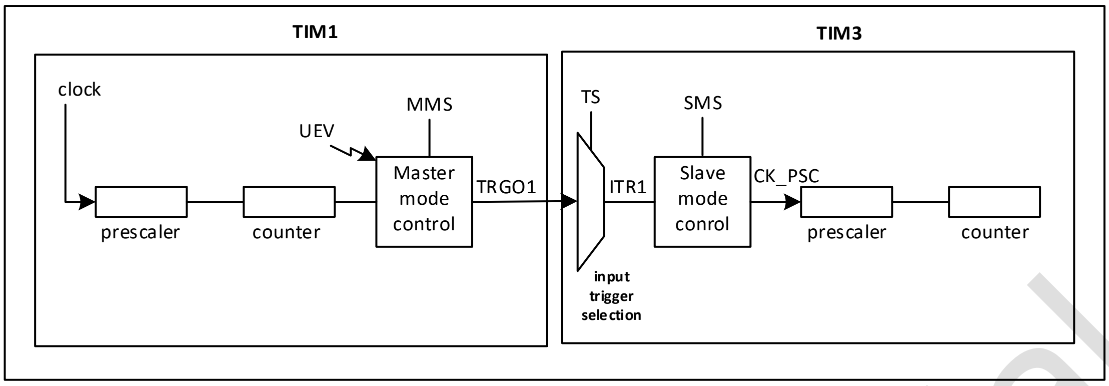


Figure 19-40 Master/Slave Timer

For example, timer 1 can be configured as a prescaler for timer 2. Perform the following operations:

- The timer 1 is configured as the main mode, and it can output a periodic trigger signal at each update event UEV. When MMS = '010' of the TIM1 _ CR2 register, a rising edge signal is output on TRGO1 every time an update event occurs.
- TRGO1 connected to timer 1 is output to timer 2, TS = '000' of the TIM2 _ SMCR register is set, and timer 2 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 TIM2 _ SMCR register); In this way, the timer 2 can be driven by the periodic rising edge of the timer 1 (i.e., the counter of the timer 1 overflows) signal.
- Finally, the CEN bit of the corresponding (TIM2 _ CR1 register) must be set to start two timers respectively, ensuring that Timer2 is started first and Timer1 is started later.

Note: If OCx has been selected as the trigger output of timer 1 (MMS = 1xx), its rise is used to drive the counter of timer 2.

Use one timer to enable another timer

In this example, the enablement of timer 2 is controlled by the output comparison of timer 1. Refer to the connection of FIGS. 19-40 above. Timer 2 counts the divided internal clock only when the OC1REF of timer 1 is high. The clock frequencies of both timers are obtained by dividing CK _ INT by 3 by the prescaler (fCK _ CNT = fCK _ INT/3).

- Timer 1 is configured as the main mode, and its output comparison reference signal (OC1REF) is sent out as the trigger output (MMS of TIM1 _ CR2 register = 100)
- OC1REF waveform of configuration timer 1 (TIM1 _ CCMR1 register)
- Configure Timer 2 to get input trigger from Timer 1 (TS = 000 of TIM2 _ SMCR register)
- Configure timer 2 in gated mode (SMS of TIM2 _ SMCR register = 101)
- Set CEN = 1 of the TIM2 _ CR1 register to enable timer 2
- Set CEN = 1 of the TIM1 _ CR1 register to start timer 1

Note: The clock of Timer 2 is not synchronized with the clock of Timer 1, this mode only affects the enable signal of the Timer 2 counter.

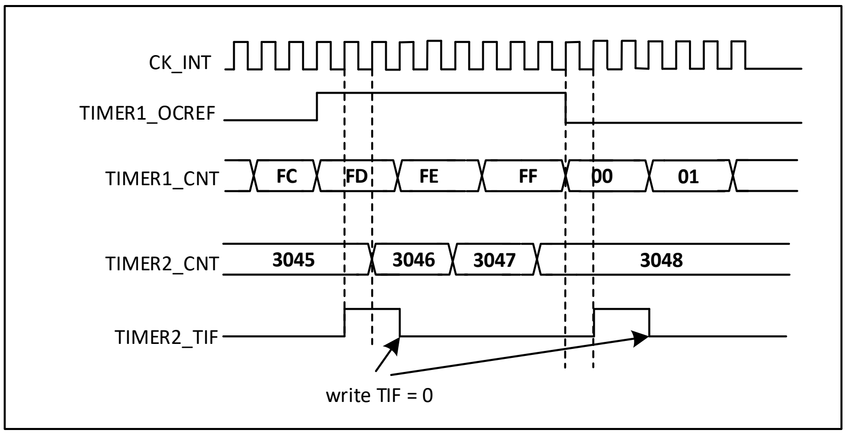


Figure 19-41 OC1REF control timer 2 for timer 1

In the example of FIGS. 19-41, their counters and prescalers are not initialized before timer 2 is started, so they start counting from the current value. You can reset the 2 timers before starting timer 1 so that they start with a given value, i.e. write any value as needed in the timer counter. Write the UG bit of the TIMx _ EGR register to reset the timer.

In the next example, Timer 1 and Timer 2 need to be synchronized. Timer 1 is master mode and starts from 0, Timer 2 is slave mode and starts from 0xE7; The prescaler coefficients of the two timers are the same. Writing '0' to the CEN bit of TIM1 _ CR1 disables timer 1, and timer 2 stops.

- Timer 1 is configured as the main mode, and the timer enable signal (CNT _ EN) is sent as the trigger output (TIM1 _ CR2 register MMS = 001).
- The OC1REF waveform (TIM1 _ CCMR1 register) of timer 1 is configured.
- Configure Timer 2 to get input trigger from Timer 1 (TS = 000 of TIM2 _ SMCR register)
- Configure timer 2 in gated mode (SMS of TIM2 _ SMCR register = 101)
- Set UG = '1' of the TIM1 _ EGR register to reset timer 1.
- Set UG = '1' of the TIM2 _ EGR register to reset timer 2.
- Write '0xE7' to the counter of timer 2 (TIM2 _ CNT), initializing it to 0xE7.
- Set CEN = '1' of the TIM2 _ CR1 register to enable timer 2.
- Set CEN = '1' of the TIM1 _ CR1 register to start timer 1.
- Set CEN = '0' of the TIM1 _ CR1 register to stop timer 1.

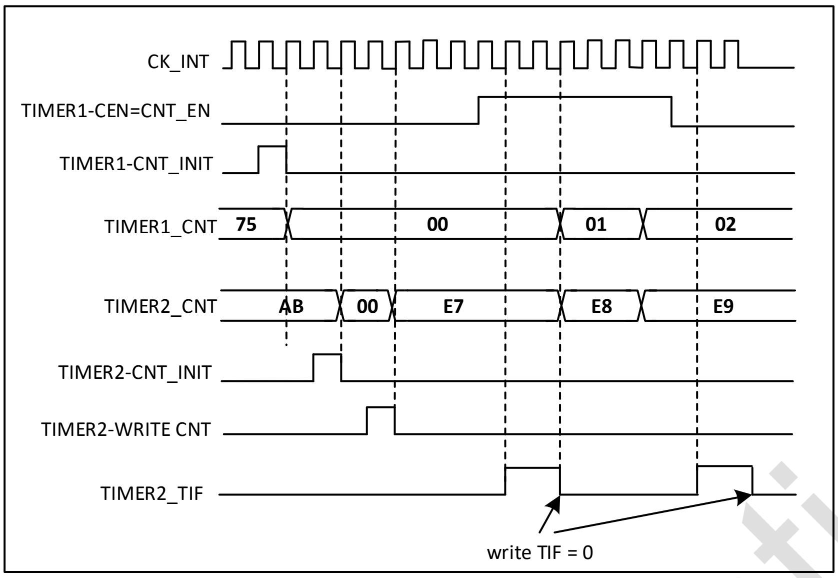


Figure 19-42 Timer 2 can be controlled by enabling Timer 1

Use one timer to start another timer

In this example, Timer 2 is enabled using the update event of Timer 1. Refer to the connection in the diagram above. Once timer 1 generates an update event, timer 2 starts counting from its current value (which may be non-zero) according to the divided internal clock. Upon receipt of the trigger signal, the CEN bit of timer 2 is automatically set to '1' and the counter starts counting until '0' is written to the CEN bit of the TIM2 _ 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 timer 1 to main mode and send its update event (UEV) as trigger output (MMS of TIM1 _ CR2 register = 010)
- Configure the period of timer 1 (TIM1 _ ARR register)
- Configure Timer 2 to get input trigger from Timer 1 (TS = 000 of TIM2 _ SMCR register)
- Configure timer 2 in trigger mode (SMS = 110 for TIM2 _ SMCR register)
- Set CEN = 1 of the TIM1 _ CR1 register to start timer 1

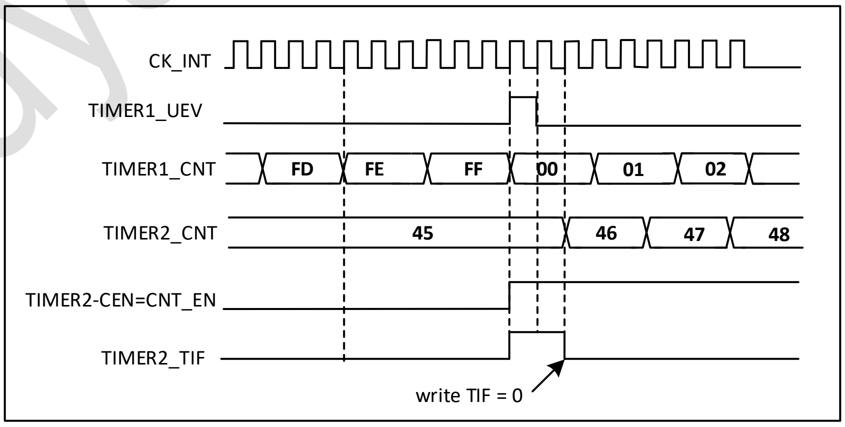


Figure 19-43 Trigger timer 2 with an update of timer 1

In the previous example, you can initialize two counters before starting counting. The following figure shows the action of using the triggered mode instead of the gated mode (SMS = 110 of the TIM2 _ SMCR register) in the same configuration as the above figure.

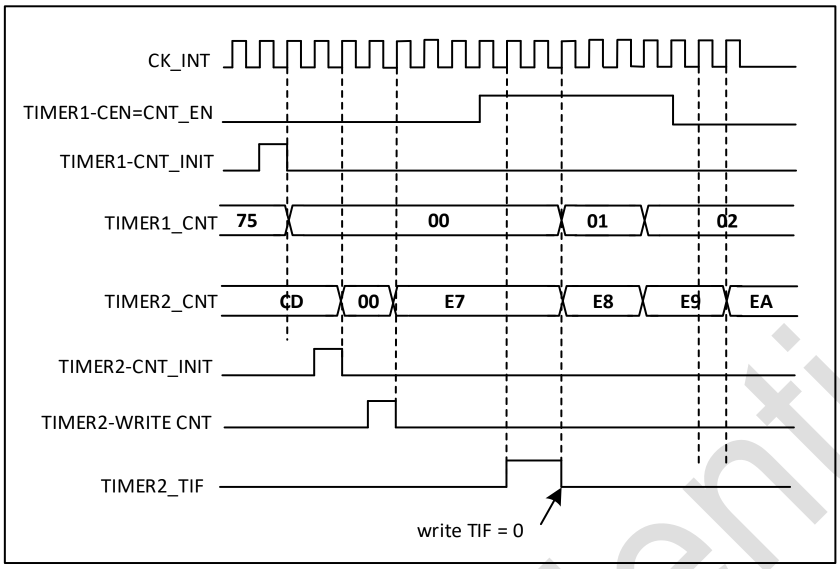


Figure 19-44 Trigger timer 2 with enabling of timer 1

Use an external trigger to start 2 timers synchronously

In this example, when the TI1 input of timer 1 rises, timer 1 is enabled, and timer 2 is enabled at the same time. See the connection mode in the above figure. To ensure the alignment of counters, timer 1 must be configured in master/slave mode (corresponding to TI1 as slave, corresponding to timer 2 as master):

- Timer 1 is configured in the main mode, sending its enable as the trigger output (MMS = '001' of the TIM1 _ CR2 register).
- Timer 1 is configured in slave mode and an input trigger is obtained from TI1 (TS = '100' of the TIM1 _ SMCR register).
- Timer 1 is configured in trigger mode (SMS = '110' of TIM1 _ SMCR register).
- Configure timer 1 in master/slave mode with MSM = '1' for TIM1 _ SMCR register.
- Configure Timer 2 to get input trigger from Timer 1 (TS = 000 of TIM2 _ SMCR register)
- Timer 2 is configured in trigger mode (SMS = '110' of TIM2 _ SMCR register).

When a rising edge appears on TI1 of timer 1, the two timers start counting synchronously according to the internal clock, 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 timer 1 in master/slave mode.*

19.3.15. Debug mode

When the microcontroller enters debug mode, the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

19.4. Register Description

19.4.1. TIM2 control register 1 (TIM2 _ CR1)

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	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31： 10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock Division Factor These 2 bits define the division ratio between the timer clock (CK _ INT) frequency, the dead time and the sampling clock used by the dead generator and the digital filter (ETR, Tix) 00: tDTS = tCK _ INT01: tDTS = 2 x tCK _ INT10: tDTS = 4 x tCK _ INT11: Re-serve, do not use this configuration
7	ARPE	RW	0	Automatic reload preload allowed bit 0: TIM2 _ ARR regis-ter is not buffered 1: TIM2 _ ARR register is loaded into buffer
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge alignment 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. Configured as output channel The output of (CCxS = 00 in the TIM2 _ CCMRx register) compares the interrupt flag bit only when the counter is down The number of hours is set. 10: Center-aligned mode 2. The counter counts up and down alternatively. The counter counts up and down alter-natively. Output compare interrupt flags of channels con-figured in output (CCxS=00 in TIM2_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The counter counts up and down alter-natively. The output of the channel configured as output (CCxS = 00 in the TIM2 _ 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 used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clear-ing the bit CEN).
2	URS	RW	0	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 allowed to be generated, any of the following events generates an up-date interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller

Bit	Name	R/W	Reset Value	Function
				1: If an update interrupt or DMA request is allowed, only a counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	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. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). 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: Counter disabled 1: Counter enabled 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.

19.4.2. TIM2 control register 2 (TIM2_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.	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.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	-	-	-

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	Reserved
7	TI1S	RW	0	TI1 selection 0: The TIM2_CH1 pin is connected to TI1 input 1: The TIM2 _ CH1, TIM2 _ CH2, and TIM2 _ CH3 pins are connected to the TI1 input via XOR.
6:4	MMS[2:0]	RW	000	Master mode selection These two 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 from the TIM1_EGR register is used as trigger output (TRGO). 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. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is 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 TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM1_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.

Bit	Name	R/W	Reset Value	Function
				011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	RW	0	DMA Selection for Capture/Comparison 0: When a CCx event occurs, a DMA request for CCx is sent. 1: When an update event occurs, a DMA request for CCx is sent.
2:0	Reserved	-	-	Reserved

19.4.3. TIM2 slave mode control register (TIM2_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.	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**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	ETP	RW	0	External trigger polarity The bit selects whether to use the ETR or the inversion of the ETR as the trigger operation 0: ETR is not inverted, high level or rising edge is active; 1: ETR is inverted, low or falling edge active.
14	ECE	RW	0	External clock enable bit This bit enables external clock mode 2 0: Disable external clock mode 2; 1: Enable external clock mode 2. The counter is driven by any valid edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 and connecting the TRGI to the ETRF (SMS = 111 and TS = 111). 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 '111'). Note 3: When external clock mode 1 and external clock mode 2 are simultaneously enabled, the input of the external clock is ETRF.
13:12	ETPS	RW	0	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	0	External trigger filter These bits define the frequency at which the ETRP signal is sampled and the bandwidth at which the ETRP is digitally filtered. In fact, the digital filter is an event counter,

Bit	Name	R/W	Reset Value	Function
				which will produce an output jump after recording N events. 0000: No filter, sampled in fDTS 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1000: Sampling frequency fSAMPLING = fDTS/8, N = 6 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 1111: Sampling frequency 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 al-low perfect synchronization between the current timer (via TRGO) and its slave timers. It is useful if we want to syn-chronize several timers on a single external event.
6:4	TS[2:0]	RW	0	Trigger selection These bit-fields select the trigger input to be used to syn-chronize the counter. 000: Internal Trigger 0 (ITR0) Internal Trigger 1 (ITR1) 010 Internal Trigger 2 (ITR2) 011 Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111： Reserved Note: These bits must be changed only when they are not used to avoid wrong edge detections at the transition.
3	OCCS	RW	0	OCREF clear Selection 0: OCREF _ CLR _ INT is connected to OCREF _ CLR in-put 1: OCREF _ CLR _ INT is connected to ETRF
2:0	SMS[2:0]	RW	0	Select from Mode 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: Off slave mode-if CEN = 1, the prescaler is driven di-rectly by the internal clock. 001: Encoder Mode 1-The counter counts up/down on the edge of TI2FP2 depending on the level of TI1FP1. 010: Encoder Mode 2-The counter counts up/down on the edge of TI1FP1 depending on the level of TI2FP2. 011: Encoder Mode 3-The counter counts up/down on the edges of TI1FP1 and TI2FP2 depending on the level of the other inputs. 100: Reset mode-The rising edge of the selected trigger input (TRGI) reinitializes the counter and generates a sig-nal 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-counter starts on the rising edge of trig-ger input TRGI (but not reset), only counter Start-up of is controlled. 111: External clock mode 1-rising edge of selected trigger input (TRGI) drives counter.

Bit	Name	R/W	Reset Value	Function
				Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 19-2 TIMx internal trigger connection

Slave timer	ITR0 (TS=000)	ITR1 (TS=001)	ITR2 (TS=010)	ITR3 (TS=011)
TIM2	TIM1	Reserved	Reserved	TIM14 OC1

19.4.4. TIM2DMA/interrupt enable register (TIM2 _ 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	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	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-	RW	-	RW	RW	RW	RW	RW	-	RW	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	TDE	RW	0	TDE: Triggering of DMA requests enabled 0: Trigger DMA request disabled. 1: Triggering of DMA request enabled
13	Reserved	-	-	Reserved
12	CC4DE	RW	0	CC4DE: DMA requests enabled capture/compare 4 0: DMA request disabled to capture/compare 4 1: DMA request enabled to capture/compare 4
11	CC3DE	RW	0	CC3DE: DMA requests enabled to capture/compare 3 0: DMA request for capture/compare 3 disabled 1: DMA request enabled to capture/compare 3
10	CC2DE	RW	0	CC2DE: DMA requests that enable capture/compare 2 0: DMA request for capture/compare 2 disabled 1: DMA request enabled to capture/compare 2
9	CC1DE	RW	0	CC1DE: DMA request enabled to capture/compare 1 0: DMA request for capture/compare 1 is disabled 1: DMA request enabled to capture/compare 1
8	UDE	RW	0	UDE: Updated DMA requests enabled 0: Updated DMA requests disabled 1: Updated DMA requests enabled
7	Reserved	-	-	Reserved
6	TIE	RW	0	TIE: Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
5	Reserved	-	-	Reserved
4	CC4IE	RW	0	CC4IE: Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled.

Bit	Name	R/W	Reset Value	Function
				1: Update interrupt enabled

19.4.5. TIM2 status register (TIM2_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	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	IC2IR	IC1IR
-	-	-	-	-	-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	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	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	Res	TIF	Res	CC4IF	CC3IF	CC2IF	CC1F	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	-	-	Reserved
23	IC4IF	RC_W0	0	Falling Edge Capture 4 Flag Refer to IC1IF description
22	IC3IF	RC_W0	0	Falling Edge Capture 3 Flag Refer to IC1IF description
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 can be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the falling edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A falling edge capture event occurs.
19	IC4IR	RC_W0	0	Rising Edge Capture 4 Flags Refer to IC1IR description
18	IC3IR	RC_W0	0	Rising Edge Capture 3 Flag Refer to IC1IR description
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 may be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the rising edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A rising edge capture event occurs.
15:13	Reserved	-	-	Reserved
12	CC4OF	Rc_w0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	Rc_w0	0	Capture/Compare 3 overcapture flag Refer to CC1OF description
10	CC2OF	Rc_w0	0	Capture/Compare 2 overcapture flag Refer to CC1OF description
9	CC1F	Rc_w0	0	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 detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1IF flag was already set.
8:7	Res.	-	0	Reserved, must be kept at reset value.
6	TIF	Rc_w0	0	Trigger interrupt flag When a trigger event occurs (when the slave mode con-troller is in a mode other than the gated mode, it is de-tected at the TRGI input that Effect edge, or either edge in gated mode) by hardware to the position 1. It is cleared by software.

Bit	Name	R/W	Reset Value	Function
				0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved	-	-	Reserved
4	CC4IF	Rc_w0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	Rc_w0	0	Capture/Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	Rc_w0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set to 1 by hardware when the counter value matches the comparison value, except in centrosymmetric mode (refer to TIM2 _ CR1 register Device's CMS bit). It is cleared by software. 0: No match; 1: The content of the counter TIM2_CNT matches the content of the TIM2_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM2_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM2_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	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 by hardware when the registers are updated: – At overflow or underflow regarding the repetition counter value (update if REP_CNT = 0) and if the UDIS=0 in the TIM2_CR1 register. If UDIS of the TIM2 _ CR1 register = 0 and URS = 0, an update event occurs when UG of the TIM2 _ EGR register = 1 − If UDIS = 0 and URS = 0 of the TIM2 _ CR1 register, an update event occurs when the CNT is reinitialized by the trigger event (Ref: Slave Mode Control Register (TIM2 _ SMCR))

19.4.6. TIM2 event generation register (TIM2_EGR)

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	Res	Res	TG	Res	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	-	-	-	-	-	-	W	-	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	Reserved
7	Reserved	-	-	Reserved
6	TG	W	0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: TIF of the TIM2 _ SR register = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated.
5	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
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 by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. If channel CC1 is configured as input: The current counter value is captured in the TIM1 _ CCR1 register, and CC1IF = 1 is set. If the corresponding inter-rupt 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	Generate an update event This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is se-lected or if DIR=0 (upcounting), else it takes the auto-re-loadvalue (TIM2_ARR) if DIR=1 (downcounting).

19.4.7. TIM2 capture/compare mode register 1 (TIM2 _ CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare 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**
OC2CE	OC2M [2:0]			OC2PE	CO2FE	CC2S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
IC2F [3:0]				IC2PSC [1:0]				IC1F [3:0]				IC1PSC [1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	OC2CE	RW	0	Output Compare 2 clear enable
14:12	OC2M [2:0]	RW	000	Output Compare 2 mode
11	OC2PE	RW	0	Output Compare 2 preload enable
10	OC2FE	RW	0	Output Compare 2 fast enable
9:8	CC2S [1:0]	RW	00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM1 _ SMCR register).

Bit	Name	R/W	Reset Value	Function
				Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIM1_CCER).
7	OC1CE	RW	0	Output Compare 1 clear enable 0: OC1REF is not affected by ETRF input; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6:4	OC1M [2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 andOC1N 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 comparison register TIM1 _ CCR1 and the counter TIM1 _ CNT does not work for OC1REF 001: Set channel 1 to active level on match. When the value of the counter TIMx _ CNT is compared with the capture/compare register When 1 (TIMx _ CCR1) is the same, OC1REF is forced to be high. 010: Set channel 1 to inactive level on match. When the value of the counter TIMx _ CNT is compared with the capture/compare register When 1 (TIMx _ CCR1) is the same, OC1REF is forced to be low. 011: Toggle OC1REF toggles when TIMx_CNT=TIM1_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 TIM2 _ 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-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. 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 Compare 1 preload enable 0: Preload register on TIM2_CCR1 disabled. TIM2_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM2_CCR1 enabled. Read/Write operations access the preload register. TIM2_CCR1 preload value is loaded in the active register at each update event. Note 1: Once the LOCK level is set to 3 (the LOCK bit in the TIMx _ BDTR register) and CC1S = 00 (the channel is configured to Output) then the bit cannot be modified. Note: The PWM mode can be used without validating the preload register only in one pulse mode. Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 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. Therefore, OC is set to a comparison level and

Bit	Name	R/W	Reset Value	Function
				It is not related to the comparison results. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM2 _ SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIM2_CCER).

Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IF2F	RW	0000	Input capture 2 filter
11:10	IC2PSC [1:0]	RW	00	Input/capture 2 prescaler
9:8	CC2S [1:0]	RW	0	Capture/Compare 2 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM2 _ SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIM2_CCER).
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 in-put and the length of the digital filter applied to TI1. 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, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is de-tected 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 [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection

Bit	Name	R/W	Reset Value	Function
				This bit-field defines the direction of the channel (in-put/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: CC1 channel configured as input, IC1 mapped on TI2; 11: The CC1 channel is configured as an input, and the IC1 is mapped on the TRC. This mode only works when the internal trigger input is selected (selected by TS bit of TIM2 _ SMCR register). Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIMx _ CCER register).

19.4.8. TIM2 capture/compare mode register 2 (TIM2 _ CCMR2)

Address offset: 0x1C

Reset value: 0x0000 0000

Output compare 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**
OC4CE	OC4M [2:0]			OC4PE	CO4FE	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

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	OC4CE	RW	0	Output Compare 4 clear enable
14:12	OC4M [2:0]	RW	000	Output compare 4 mode
11	OC4PE	RW	0	Output Compare 4 preload enable
10	OC4FE	RW	0	Output Compare 4 fast enable
9:8	CC4S [1:0]	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIMx _ SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).
7	OC3CE	RW	0	Output Compare 3 clear enable
6:4	OC3M [2:0]	RW	00	Output compare 3 mode
3	OC3PE	RW	0	Output Compare 3 preload enable
2	OC3FE	RW	0	Output Compare 3 fast enable
1:0	CC3S [1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIMx _ SMCR register).

Bit	Name	R/W	Reset Value	Function
				Note: CC3S is writable only when the channel is closed (CC3E = 0 in the TIMx _ CCER register).

Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IF4F	RW	0000	Input capture 4 filter
11:10	IC4PSC [1:0]	RW	00	Input/capture 4 prescaler
9:8	CC4S [1:0]	RW	0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/out-put) as well as the used input. 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 works when the internal trigger input is selected (selected by TS bit of TIM2 _ SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER).
7:4	IC3F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI3 input and the length of the digital filter applied to TI1. The digital fil-ter is made of an event counter in which N consecutive events are needed to validate a tran-sition on the output: 0000: No filter, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC3PSC [1:0]	RW	00	Input/capture 3 prescaler This bit-field defines the ratio of the prescaler acting on CC3 input (IC1). Once CC13E = 0 (in TIMx _ CCER register), the prescaler is reset. 00: no prescaler, capture is done each time an edge is de-tected 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	CC3S [1:0]	RW	00	CC3S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/out-put) as well as the used input. 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 TI3; 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode only works when the internal trigger input is selected (selected by TS bit of TIM2 _ SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIM2_CCER).

19.4.9. TIM2 capture/compare enable register (TIM2 _ 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	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**
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	-	-	Reserved
15	CCNP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
14	Reserved	-	-	Reserved
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	Reserved	-	-	Reserved
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
10	Reserved	-	-	Reserved
9	CC3P	RW	0	Input/Capture 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable Refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	Reserved	-	-	Reserved
5	CC2P	RW	0	Input/Capture 2 output polarity Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. This bit is used in conjunction with CC1P to define that polarity of TI1FP1/TI2FP1, refer to CC1P description
2	Reserved	-	-	Reserved
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The two-bit CC1NP/CC1P selects whether the polarity signal of TI1FP1 or TI2FP1 is used as the trigger or capture signal. 00: non-inverting/rising edge: capture occurs on the rising edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 is not inverted (gate triggered mode, encoded mode). 01: inverting/falling edge: non-inverting/rising edge: capture occurs on the falling edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 inverted (gate triggered mode, encoded mode). 10: Reserved, invalid configuration. 11: non-inverted/both edges
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: OFF-OC1 disables output, 1: ON-OC1 signal is output to the corresponding output pin, The CC1 channel is configured as input: This bit determines whether the value of the counter can be captured into the TIMx _ CCR1 register. 0: Capture disabled; 0: Capture enabled.

Output Control for Standard OCx Channels

CCxE Bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx=OCxREF+Polarity,OCx_EN=1

19.4.10. TIM2 Counter (TIM2 _ CNT)

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	RW	0	Counter value

19.4.11. TIM2 Prescaler (TIM2 _ PSC)

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**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). The PSC contains the value loaded into the current pre-scaler register when an update event occurs;

19.4.12. TIM2 auto-reload register (TIM2_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16]															
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
31:0	ARR[31:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer in detail to the updates and actions of the time base unit regarding the ARR. The counter is blocked while the auto-reload value is null.

19.4.13. TIM2 capture/compare register 1 (TIM2_CCR1)

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 [31:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR1 register (bit OC1PE). 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 TIM2_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

19.4.14. TIM2 capture/compare register 2 (TIM2_CCR2)

Address: 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															
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
31:0	CCR2 [31:0]	RW	0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The current capture/compare register contains the value compared to the counter TIM2 _ CNT and outputs a signal on the OC port. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

19.4.15. TIM2 capture/compare register 3 (TIM2_CCR3)

Address: 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 [31:0]	RW	0	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The current capture/compare register contains the value compared to the counter TIM2 _ CNT and outputs a signal on the OC port. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

19.4.16. TIM2 capture/compare register 4 (TIM2_CCR4)

Address: 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 [15:0]	RW	0	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM2_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The current capture/compare register contains the value compared to the counter TIM2 _ CNT and outputs a signal on the OC port. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

19.4.17. TIM2 DMA control register (TIM2 _ DCR)

Address: 0x48

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.	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	-	-	Reserved
12:8	DBL[4:0]	RW	0 0000	DMA continuous transfer length These bits define the transfer length of the DMA in continuous mode (when reading or writing to the address of the TIMx _ DMAR register When, the timer performs a continuous transmission), that is, defines the number of bytes to be transmitted: 00000: 1 byte

Bit	Name	R/W	Reset Value	Function
				00001: 2 bytes 00010: 3 bytes .... 10001: 18 bytes
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	0 0000	DBA [4: 0]: DMA Base address These bits define the base address of the DMA in continuous mode (when reading or writing to the address of the TIM2 _ DMAR register DBA is defined as the offset from the address where the TIM1 _ CR1 register is located: 00000: TIM2 _ CR1 00001: TIM2 _ CR2 00010: TIM2 _ SMCR ......

19.4.18. DMA address of TIM2 continuous mode (TIM2 _ DMAR)

Address offset: 0x4C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA continuous transfer register A read or write to the TIM2 _ DMAR register results in an access operation to a register at the following address: TIM2 _ CR1 address + (DBA + DMA index) * 4, where: The "TIM2 _ CR1 address" is the address of the control register 1; "DBA" is the base address defined in the TIM2 _ DCR register; The "DMA pointer" is an offset automatically controlled by the DMA, which depends on the DBL defined in the TIM2 _ DCR register.

20. General-purpose timer (TIM14)

20.1. TIM14 introduction

The general-purpose timer TIM14 is consist of a 16-bit auto-reload 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).

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 TIM14 timers are completely independent and do not share any resources with each other. They can operate synchronously together, see the Synchronization section of TIM2.

20.2. TIM14 main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536 (can be changed “on the fly”).
- One independent channel for:
  - Input capture
  - Output Compare
  - PWM generation (edge-aligned mode)
- Interrupt generation on the following events:
  - Update: counter overflow, counter initialization (by software)
  - Input capture
  - Output Compare

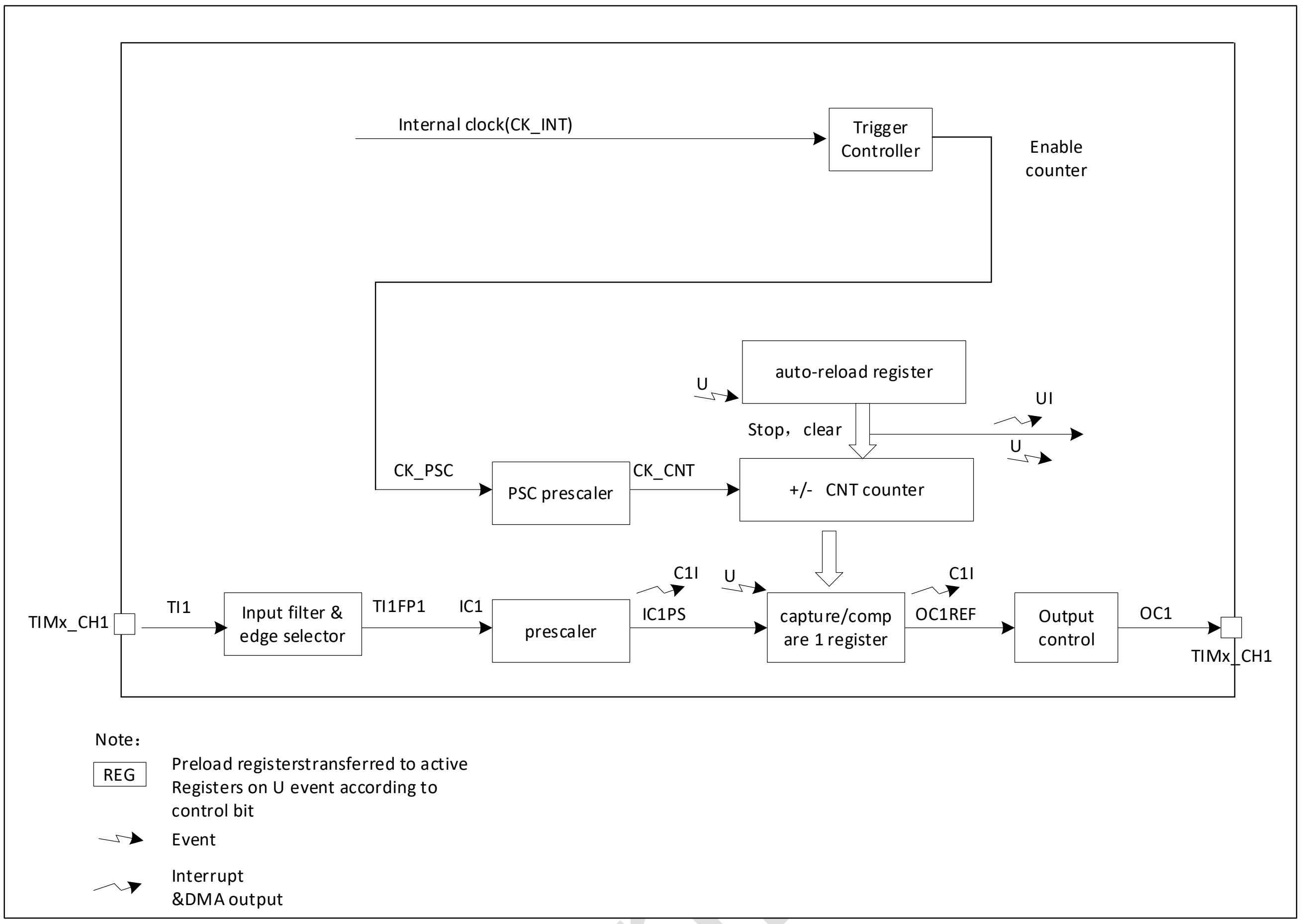


Figure 20-1 General-purpose timer diagram (TIM14)

20.3. TIM14 functional description

20.3.1. Time-base unit

The main block of the timer is a 16-bit up-counter with its related auto-reload register. The counter clock can 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 includes:

- Counter register (TIM14_CNT)
- Prescaler register (TIM14_PSC)
- Auto-reload register (TIM14_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM14_CR1 register. The update event is sent when the counter reaches the overflow value and if the UDIS bit equals 0 in the TIM14_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM14_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM14_CR1 register.

Prescalation Description:

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM14_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Tables below give some examples of the counter behavior when the prescaler ratio is changed on the fly.

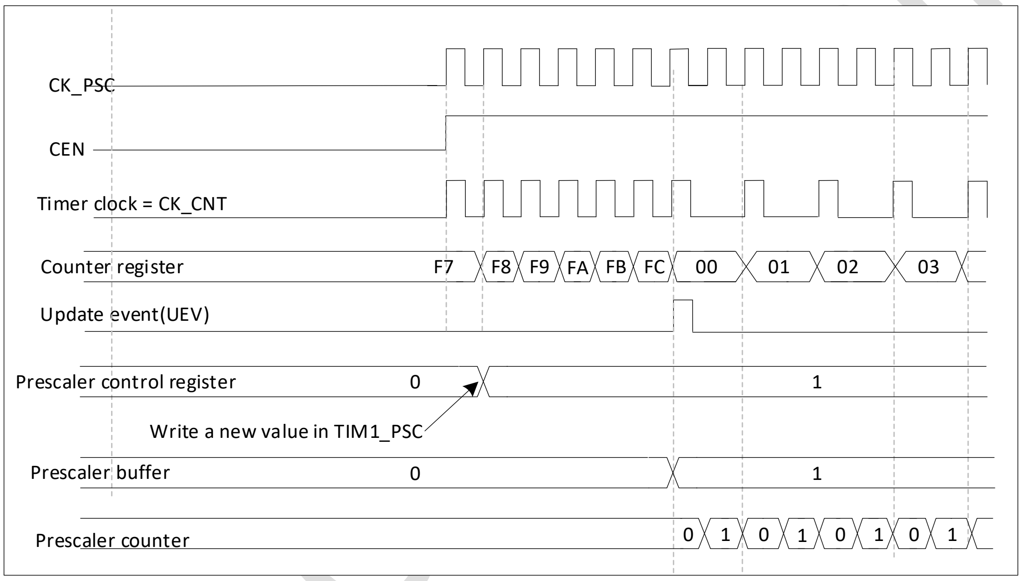


Figure 20-2 Counter timing diagram with prescaler division change from 1 to 2

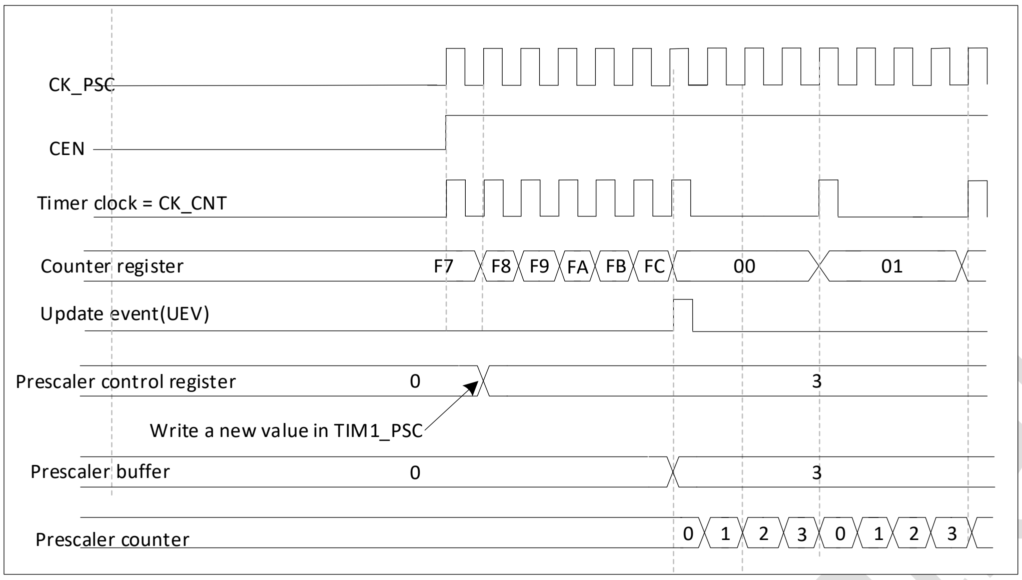


Figure 20-3 Counter timing diagram with prescaler division change from 1 to 4

20.3.2. Counting pattern

Upcounting mode

The counter counts from 0 to the auto-reload value (contents of the TIM14_ARR register), then restarts from 0 and generates a counter overflow event.

Else the update event is generated at each counter overflow. Setting the UG bit in the TIM14_EGR register (by software) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter re-starts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit in the TIM14 _ CR1 register is set (Select Update Request), setting the UG bit will generate an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM14_SR register) is set (depending on the URS bit).

- The auto-reload shadow register is updated with the preload value (TIM14_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM14_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

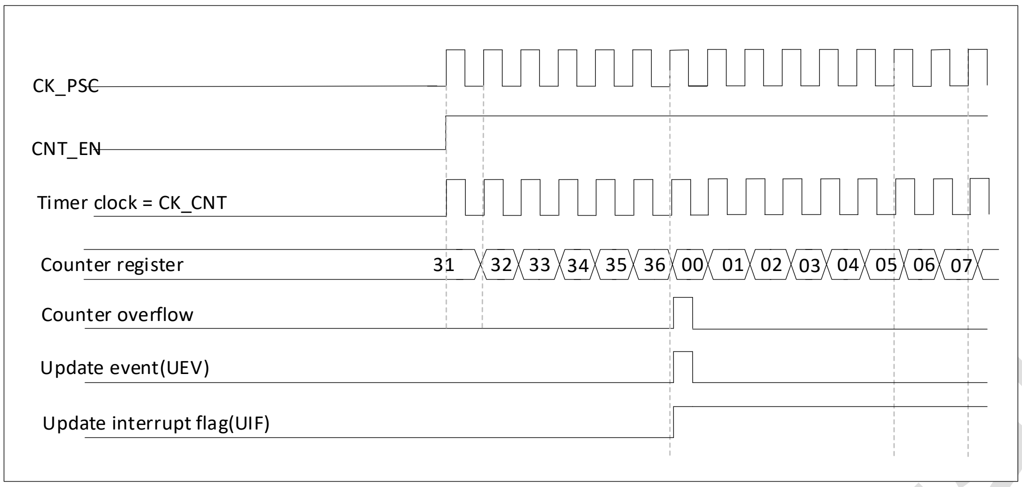


Figure 20-4 Counter timing diagram, internal clock divided by 1

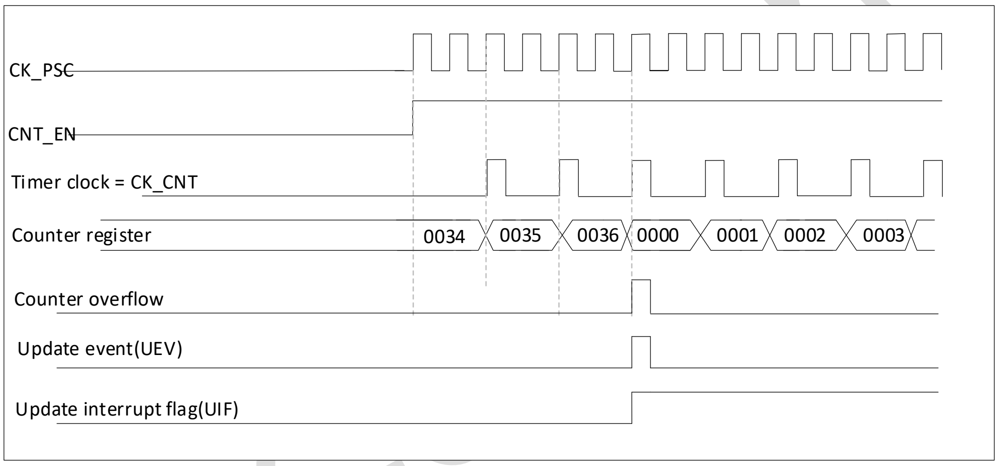


Figure 20-5 Counter timing diagram, internal clock divided by 2

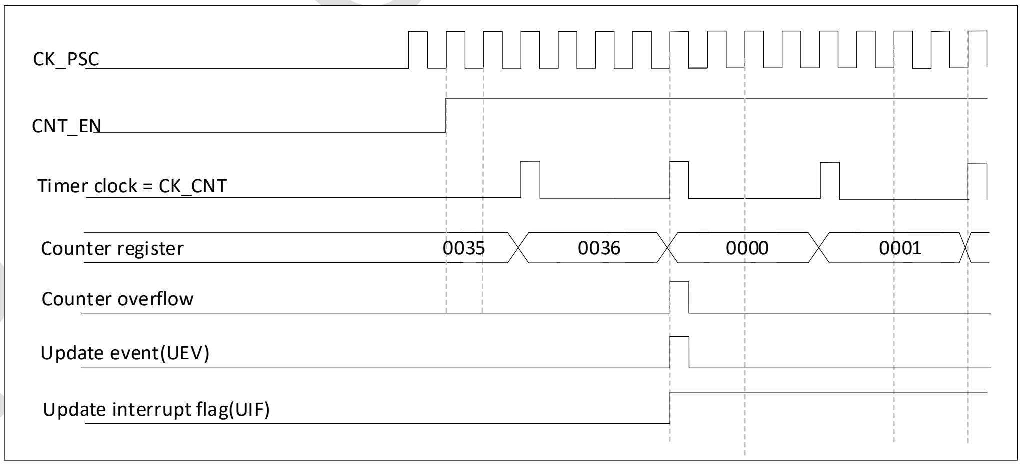


Figure 20-6 Counter timing diagram, internal clock divided by 4

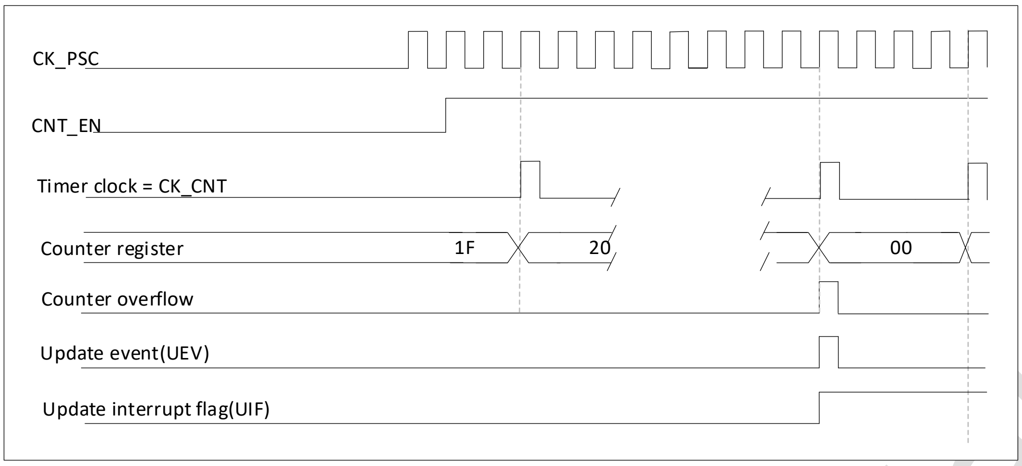


Figure 20-7 Counter timing diagram, internal clock divided by N

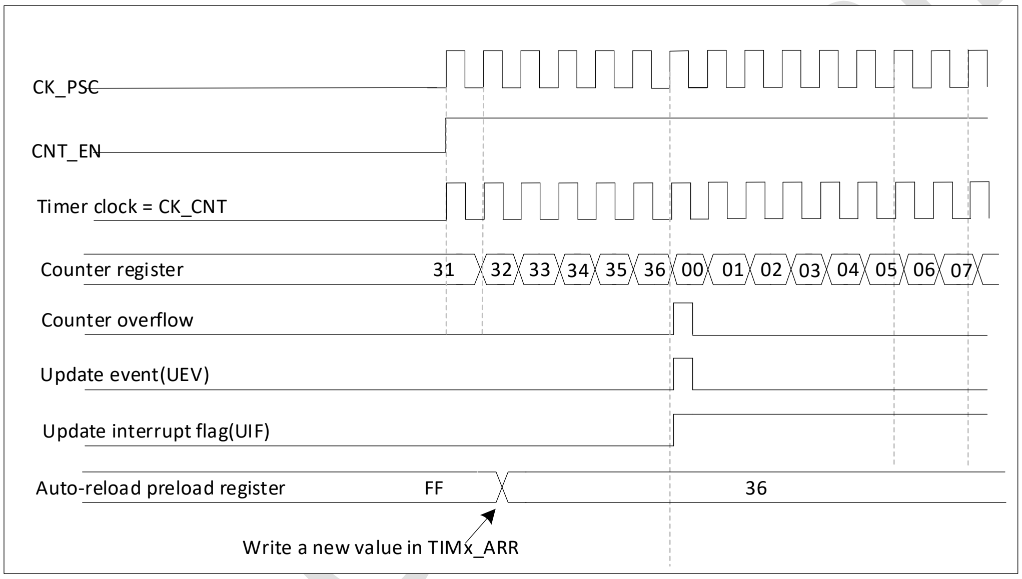


Figure 20-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

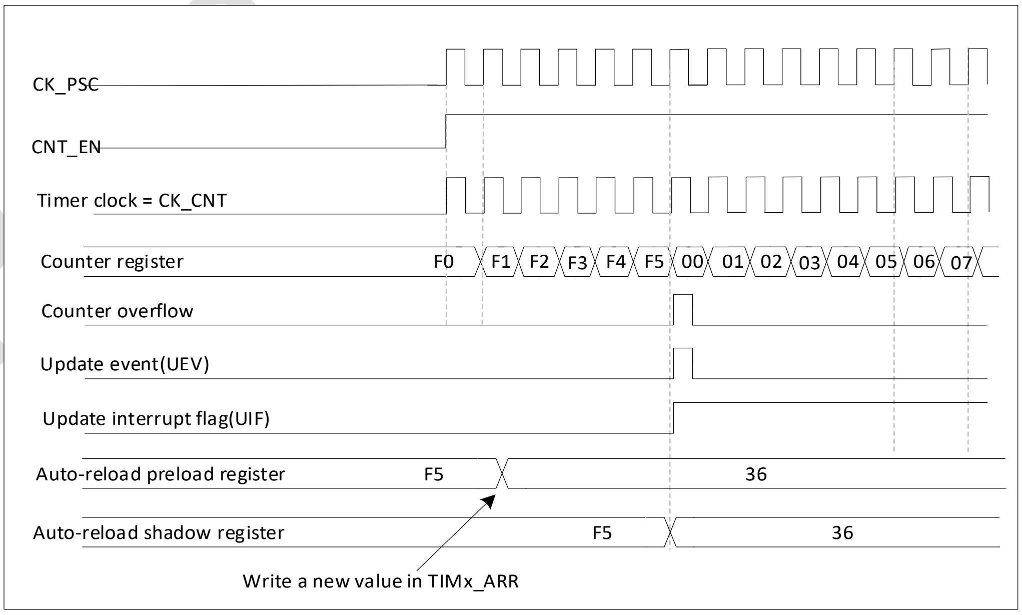


Figure 20-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

20.3.3. Clock sources

The counter clock is provided by the Internal clock (CK_INT) source. The CEN bit of the TIMx _ CR1 register and the UG bit of the TIM14 _ EGR register are the actual control bits (except that the UG bit is automatically cleared) and can only be changed by software. Once the CEN bit is set to 1, the internal clock provides clock to the frequency divider.

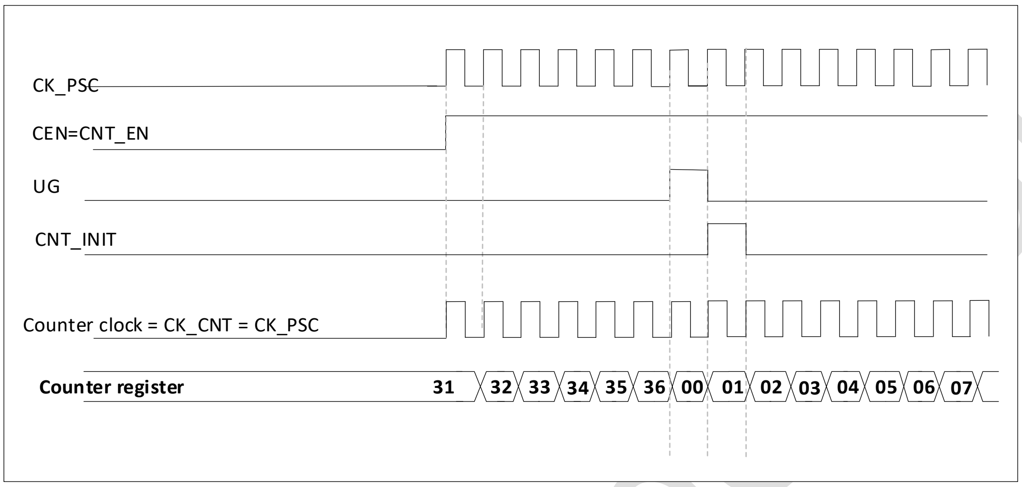


Figure 20-10 Control circuit in normal mode, internal clock divided by 1

20.3.4. 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).

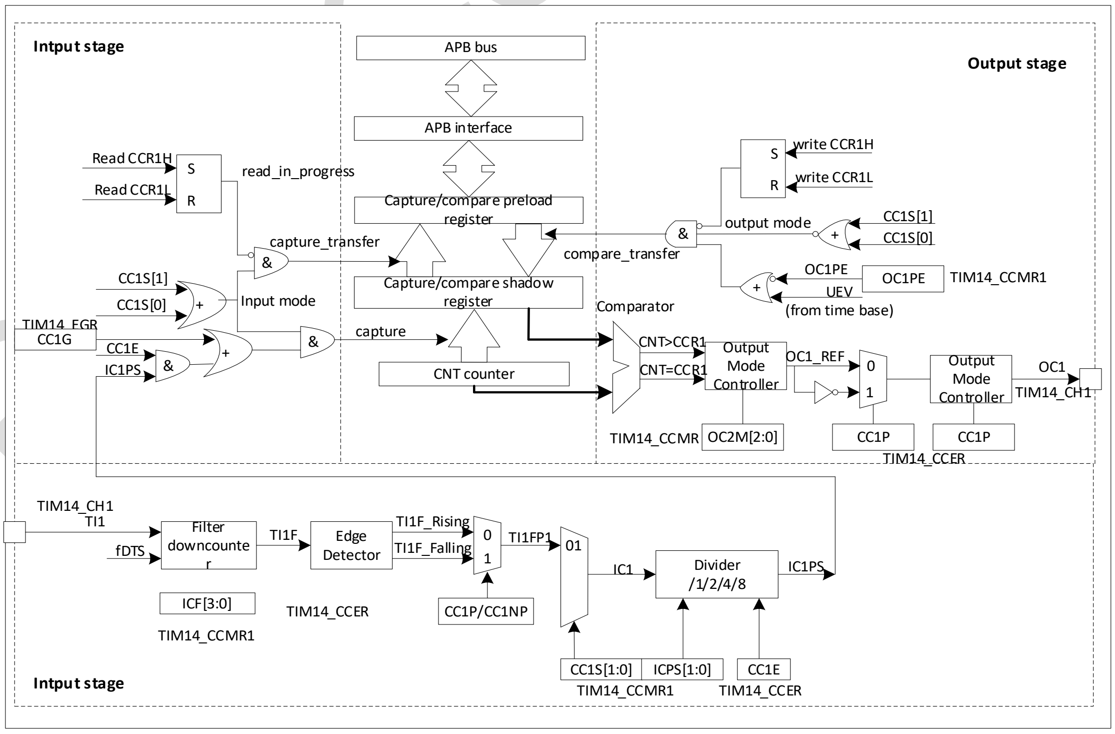


Figure 20-11 TIM1 block diagram

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge detector with polarity selection then generates a signal (TixFPx) that can be triggered as an input from the mode controller or as a capture control. The signal enters the capture register (IcxPS) by pre-division.

The output stage generates an intermediate waveform (active high) which is then used for reference. The polarity acts at the end of the chain.

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.

20.3.5. Input capture mode:

In input capture mode, when the corresponding edge of the Icx signal is detected, the current value of the counter is latched into the capture/compare register (TIM14 _ CCRx). When a capture event occurs, the corresponding CcxIF flag (TIM14 _ SR register) is set to 1. If the CcxIF flag is already high when the capture event occurs, the repeat capture flag CcxOF (TIMx _ SR register) is set to 1. Writing to CcxIF clears the CcxIF, or reading captured data in storage also clears the CcxIF. Writing CcxOF = 0 clears CcxOF.

The following example shows how to capture the counter value in TIM14_CCR1 when TI1input rises. To do this, use the following procedure:

- Select a valid output: TIM14 _ CCR1 must be connected to the Tl1 input, so write CC1S = 01 in the TIM14 _ CCMR1 register, as long as CC1S is not 00, the channel is configured as input and the TIM14 _ CCR1 register becomes read-only.
- Depending on the characteristics of the input signal, configure the input filter to the desired width (that is, when the input is Tix, the input filter control bit is the IcxF bit in the TIM14 _ CCMRx register). Let's imagine that, when toggling, the input signal is not stable during at least 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 f_{DTS} frequency). Then write IC1F bits to'0011' in the TIM14_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by programming CC1P and CC1NP bits to '00' in the TIMx_CCER register (rising edge in this case).
- 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.

When an input capture occurs:

- The TIM14_CCR1 register gets the value of the counter on the active transition.

- CC1IF flag is set (interrupt flag). When at least 2 consecutive captures occur and CC1IF has not been cleared, CC1OF is also set to 1.
- 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 requests can be generated by software by setting the corresponding CCxG bit inthe TIMx_EGR register.

20.3.6. Forced output mode

In this mode (CCxS bits = 00 in the TIM14 _ CCMRx register), the output comparison signals (OCxREF and corresponding OCx) can be directly set to active or invalid by the software, regardless of the comparison result between the output comparison 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 TIM14_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 TIM14_CCMRx register.

Anyway, the comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. This is described in the output compare mode section below.

20.3.7. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

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 TIM14_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 the flag bit in the interrupt status register (the CcxIF bit in the TIMx _ SR register).
- If the corresponding interrupt mask (CcxIE bit in the TIM14 _ DIER register) is set, an interrupt is generated.

The TIM14_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM14_CCMRx register.

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).

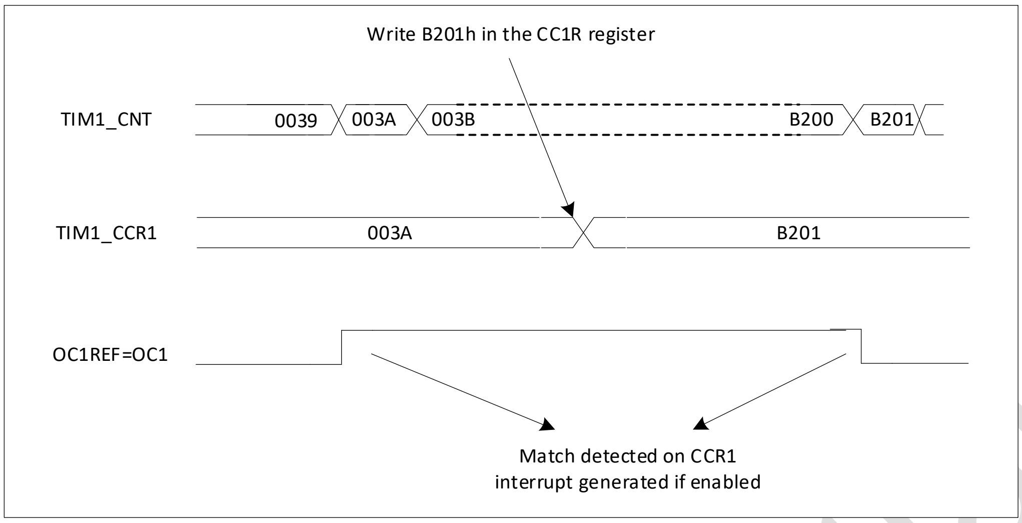


Figure 20-12 Output compare mode, toggle on OC1

20.3.8. Pulse width adjustment (PWM) mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIM14_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIM14_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIM14_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIM14_CCER register. It can be programmed as active high or active low. The CCxE bit in the TIM14 _ CCER register controls the OCx output enable.

In PWM mode (1 or 2), TIM14x_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT ≤ TIM14_CCRx.

The timer is able to generate PWM in edge-aligned mode only since the counter is upcounting.

PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIM14_CNT < TIMx_CCRx else it becomes low. If the compare value in TIM14_CCRx is greater than the auto-reload value (in TIM14_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’.

Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

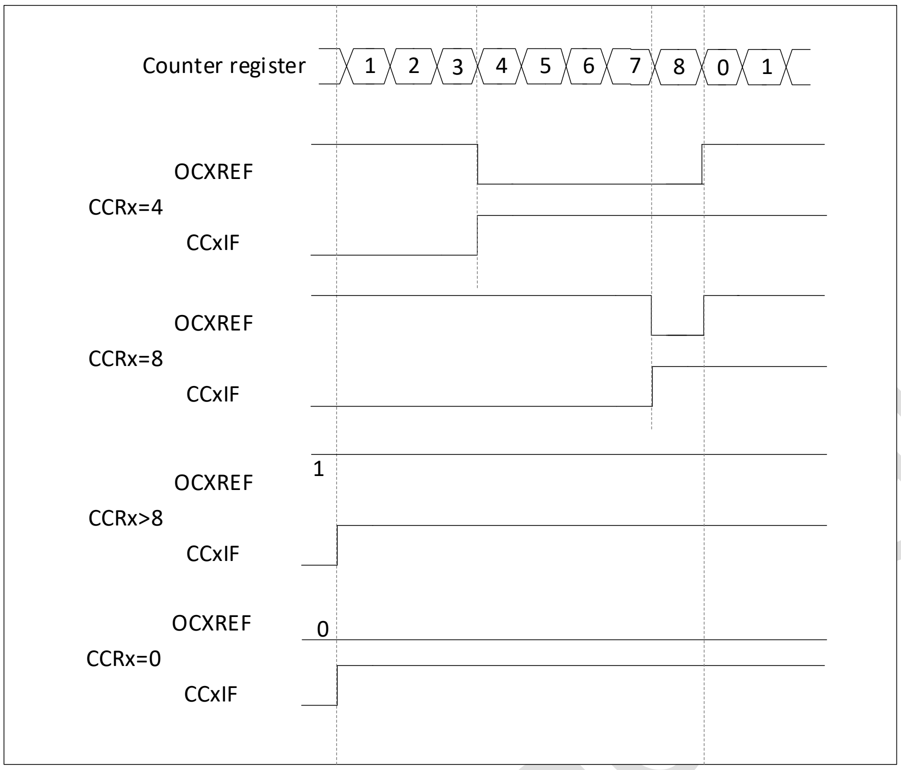


Figure 20-13 Edge-aligned PWM waveforms (ARR=8)

20.3.9. Debug mode

When the microcontroller enters debug mode (Cortex®-M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

20.4. TIM14 registers

20.4.1. TIM14 control register 1 (TIM14_CR1)

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.	CKD[1:0]		ARPE	Res.		Res	Res	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	-		-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock division factor, these 2 bits define the division ratio between the sampling clocks used at the timer clock (CK _ INT) frequency 00: tDTS = tCK _ INT01: tDTS = 2 x tCK _ INT10: tDTS = 4 x tCK _ INT11: Reserve, do not use this configuration
7	ARPE	RW	0	Automatic reload preload allowed bit 0: TIM14 _ ARR register is not buffered 1: TIM14 _ ARR register is loaded into buffer

Bit	Name	R/W	Reset Value	Function
6:3	Reserved	-	-	Reserved
2	URS	RW	0	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 allowed to be generated, any of the following events generates an up-date interrupt or DMA request: – Counter overflow/underflow − Setting the UG bit 1: If an update interrupt or DMA request is allowed, only a counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	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 Buffered registers are then loaded with their preload val-ues. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). If the UG bit is set or a hardware reset is issued from the mode controller, the counter and prescaler are Re-initialize.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: external clock, gated mode and encoder mode can work only if the CEN bit has been previously set by soft-ware. However, trigger mode can set the CEN bit auto-matically by hardware.

20.4.2. TIM14 DMA/interrupt enable register (TIM14 _ 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.	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.														CC1IE	UIE
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

20.4.3. TIM14 status register (TIM14_SR)

Address offset: 0x010

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	IC1IF	Res	Res	Res	IC1IR
-	-	-	-	-	-	-	-	-	-	-	RC_W0	-	-	-	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						**CC1OF**	**Res**							**CC1F**	**UIF**

-	Rc_w0	-	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	Reserved
20	IC1IF	Rc_w0	0	Falling Edge Capture 1 Flag This flag can be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the falling edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A falling edge capture event occurs.
19:17	Res.	-	0	Reserved, must be kept at reset value.
16	IC1IR	Rc_w0	0	Rising Edge Capture 1 Flag This flag may be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the rising edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A rising edge capture event occurs.
15:10	Reserved	-	-	Reserved
9	CC1OF	Rc_w0	0	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 detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1IF flag was already set.
8:2	Reserved	-	-	Reserved
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag **If channel CC1 is configured as output:** When the counter value matches the comparison value, this bit is set to 1 by the hardware and it is cleared to 0 by the software. 0: No match; 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. **If channel CC1 is configured as input:** This bit is set by hardware on a capture. It is cleared by software or by reading the TIM14_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM14_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	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 by hardware when the registers are updated: – At overflow and if UDIS '0' in the TIMx_CR1 register. If UDIS of the TIMx _ CR1 register = 0 and URS = 0, an update occurs when UG of the TIMx _ EGR register = 1 Software (software re-initializes CNT);

20.4.4. TIM14 event generation register (TIM14_EGR)

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.														**CC1G**	**UG**
-														W	W

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved

1	CC1G	W	0	Capture/compare 1 generation. This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: **If channel CC1 is configured as input:** Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt request will be generated. **If channel CC1 is configured as input:** The current value of the counter is captured in TIM14_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation. This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected).

20.4.5. TIM14 capture/compare mode register 1 (TIM14_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare 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	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
-								-	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6:4	OC1M [2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF Is high active, and the active levels of OC1, OC1N depend on the CC1P, CC1NP bits. 000: Frozen. The comparison between the output comparison register TIM1 _ CCR1 and the counter TIMx _ CNT does not work for OC1REF With; 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. When the value of the counter TIMx _ CNT is compared with the capture/compare register When 1 (TIMx _ CCR1) is the same, 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- In upcounting, channel 1 is active as long as TIM1_CNT<TIM1_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF=‘0’) as long as TIM1_CNT>TIM1_CCR1 else active (OC1REF=‘1’). 111: PWM mode 2 Channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the

Bit	Name	R/W	Reset Value	Function
				output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIM14_CCR1 disabled. TIM14_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM14_CCR1 enabled. Read/Write operations access the preload register. TIM14_CCR1 pre-load value is loaded in the active register at each update event.
2	OC1FE	RW	0	The output comparison 1 is fast enabled, which is used to speed up the response of the CC output to the flip-flop in-put event. 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. Therefore, OC is set to a compari-son level and It is not related to the comparison results. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OC1FE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10 Reserved 11: Reserved. Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIM14_CCER).

Input capture 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								IC1F [3:0]				IC1PSC [1:0]		CC1S [1:0]	
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 in-put and the length of the digital filter applied to TI1. 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, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6

Bit	Name	R/W	Reset Value	Function
				0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 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 [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved 11: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM14_CCER).

20.4.6. TIM14 capture/compare enable register (TIM14_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	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	CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. CC1 channel configured as output: CC1NP must be kept cleared. CC1 channel configured as input: CC1NP bit is used in conjunction with CC1P to defineTI1FP1 polarity (refer to CC1P description).
2	Reserved	-	-	Reserved
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The two-bit CC1NP/CC1P selects whether the polarity signal of TI1FP1 or TI2FP1 is used as the trigger or capture signal. 00: non-inverting/rising edge: capture occurs on the rising edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 is not inverted (gate triggered mode, encoded mode). 01: inverting/falling edge: capture occurs on the falling edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 inverted (gate triggered mode, encoded mode). 10: Reserved, invalid configuration. 11: non-inverted/both edges
0	CC1E	RW	0	Input/Capture 1 output enable **CC1 channel configured as output:** 0: Off - OC1 is not active 1: ON-OC1 signal output to corresponding output pin **The CC1 channel is configured as input:**

Bit	Name	R/W	Reset Value	Function
				This bit determines whether the value of the counter can be captured into the TIMx _ CCR1 register. 0: Capture disabled. 1: Capture enabled.

CcxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx=OCxREF+Polarity,OCx_EN=1

20.4.7. TIM14 Counter (TIM14 _ 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	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															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

20.4.8. TIM14 Prescaler (TIM14 _ PSC)

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**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). The PSC contains the value loaded into the current pre-scaler register when an update event occurs; Update event includes counter Clear 0 by the UG bit of TIM _ EGR or by the slave con-troller operating in reset mode.

20.4.9. TIM14 auto-reload register (TIM14_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.	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**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0	Auto-reload value

				ARR is the value to be loaded in the actual auto-reload register. For details, refer to 12.4.1: Time Base Unit Updates and Actions Relating to ARR. The counter is blocked while the auto-reload value is null.

20.4.10. TIM14 capture/compare register 1 (TIM14_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	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**
CCR1 [15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). 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 is the counter value transferred by the last input capture 1 event (IC1).

20.4.11. TIM14 option register (TIMx _ OR)

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														TI1_RMP	
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1:0	TI1_RMP	RW	0	Timer input 1 remap Set and cleared by software. 00: TIM14 channel 1 is connected to the GPIO. Refer to the multiplexing function in the datasheet for details. 01: TIM14 channel 1 is connected to RTCCLK. 10: TIM14 channel 1 is connected to HSE/32 clock 11: TIM14 channel 1 is connected to the MCU clock output (MCO). This configuration is determined by setting the MCO [2: 0] of the RCC _ CFG register.

21. General purpose timer (TIM16/17)

The TIM16 and TIM17 of this product have exactly the same functions.

21.1. Main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536 (can be changed "on the fly").
- A channel as:
  - Input capture
  - Output Compare
  - PWM generation (edge-aligned 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:
  - UPDATE: Counter overflow
  - Input capture
  - Output Compare
  - Brake signal input

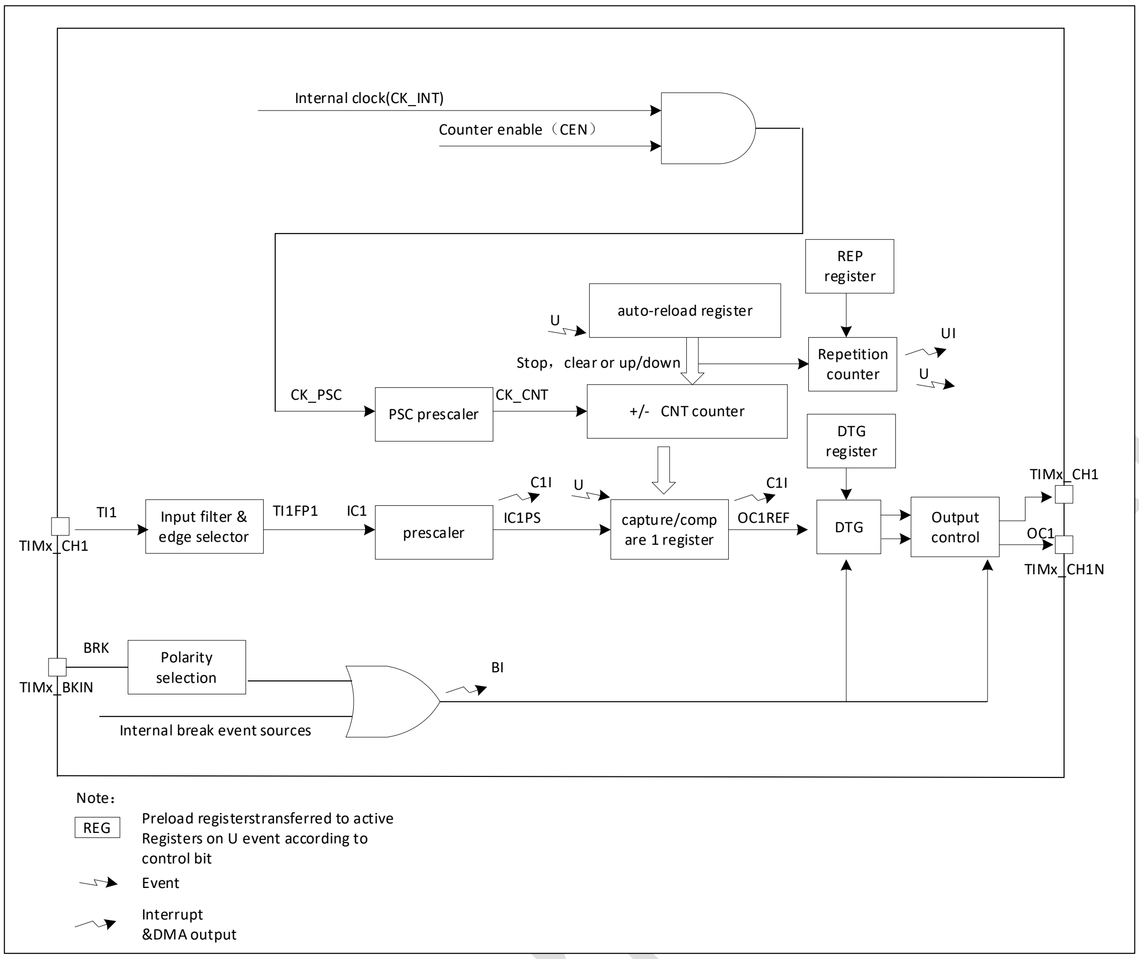


Figure 21-1 TIM16 and TIM17 architecture

21.2. Functional Description

21.2.1. Time-base unit

The main part of this programmable timer is a 16-bit up-counter with automatic reload. The clock of the counter is obtained 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 includes:

- Counter register (TIMx _ CNT)
- Prescalation register (TIMx _ PSC)
- Automatic reload register (TIMx _ ARR)
- Repeat count register (TIMx _ RCR)

The autoload register is preloaded, and writing or reading the autoload register (TIMx _ ARR) will access the preload register. Depending on the setting of the Automatic Load Enable Bit (ARPE) in the TIMx _ CR1 register, the contents of the preload register may be transferred to the shadow register at all times or at each update event UEV. The update event is sent when the counter reaches the overflow value and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescalation Description:

The prescaler can divide the counter clock frequency by any factor between 1 and 65535. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

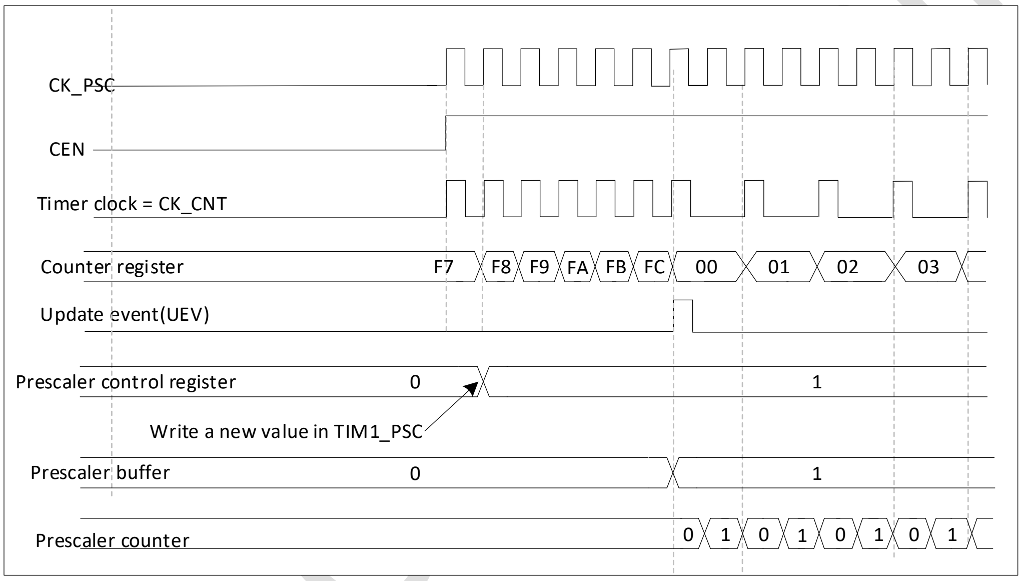


Figure 21-2 Counter timing diagram with prescaler division change from 1 to 2

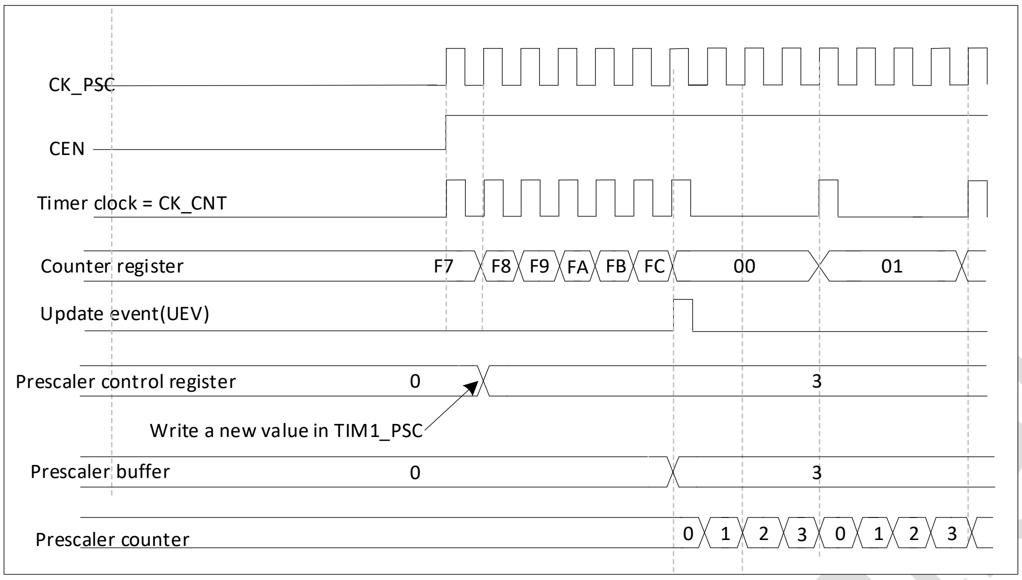


Figure 21-3 Counter timing diagram with prescaler division change from 1 to 4

21.2.2. Timer enable

The counter counts from 0 to the auto-load value (the value registered by TIMx _ ARR), and then starts counting again from 0, generating a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register. Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates 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 the UDIS bit has been written to 0. Despite this, the counter still starts at 0, and the count of the prescaler is also called 0 (but the value of the prescaler remains un-changed). In addition, if the URS bit in the TIMx _ CR1 register is set (Select Update Request), setting the UG bit will generate an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt or DMA request is generated). This is to avoid generating both update and cap-ture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded by the value in the 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 regis-ter).

The figure below shows several counter behaviors at different frequencies when TIMx _ ARR = 0X36.

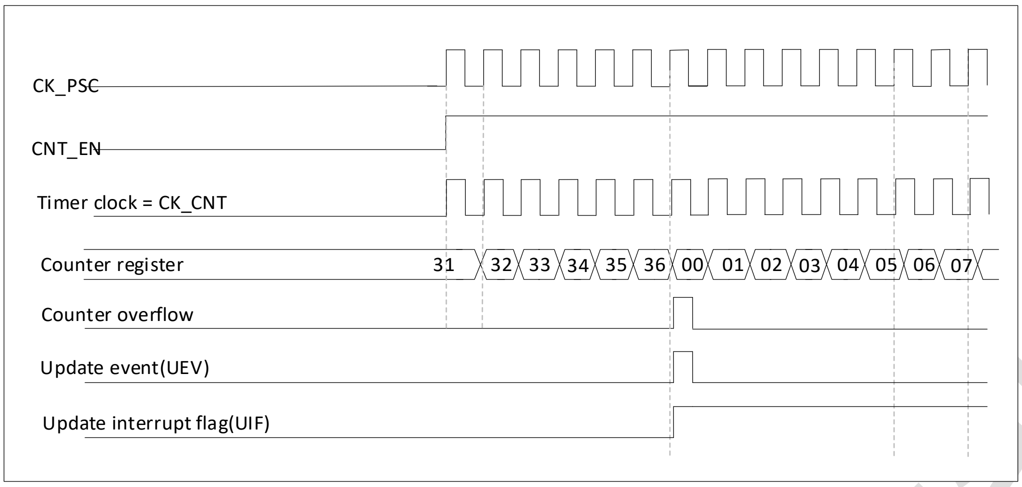


Figure 21-4 Counter timing diagram, internal clock divided by 1

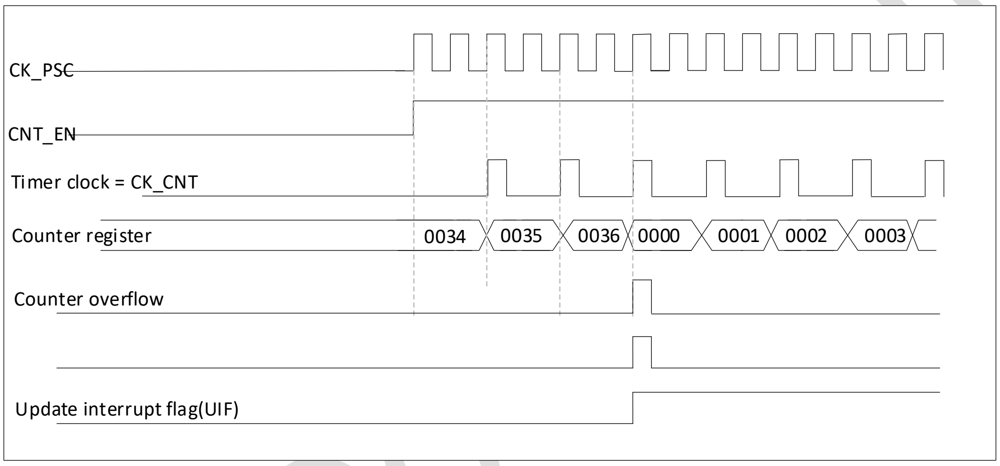


Figure 21-5 Counter timing diagram, internal clock divided by 2

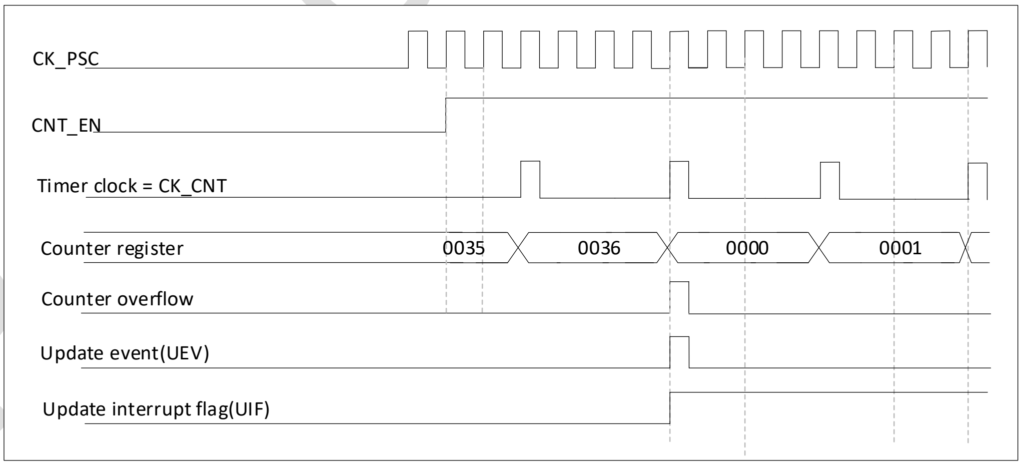


Figure 21-6 Counter timing diagram, internal clock divided by 4

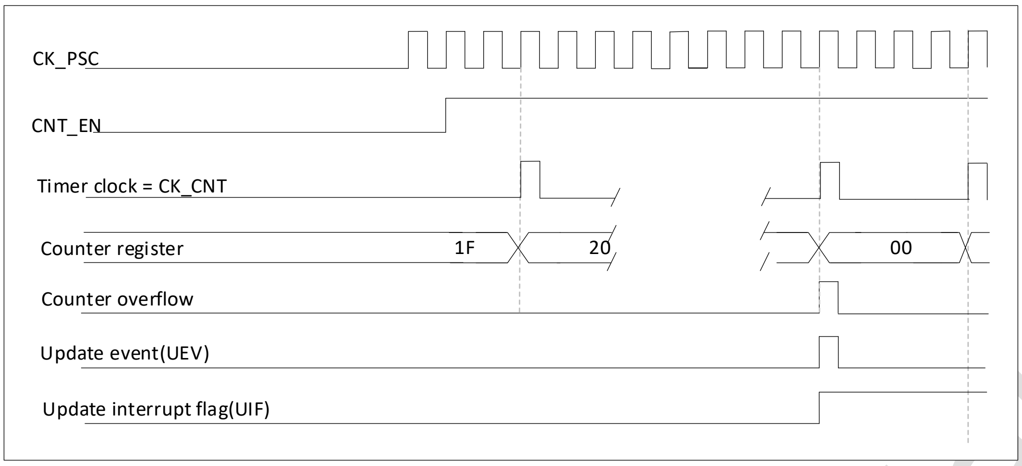


Figure 21-7 Counter timing diagram, internal clock divided by N

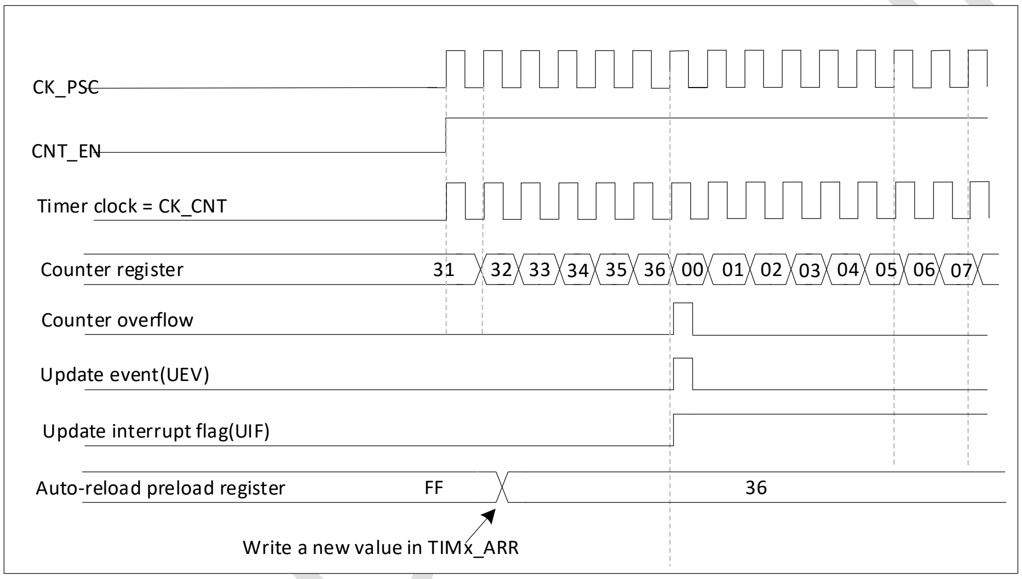


Figure 21-8 counter timing diagram, update event when ARPE = 0 (TIMx _ ARR is not pre-loaded)

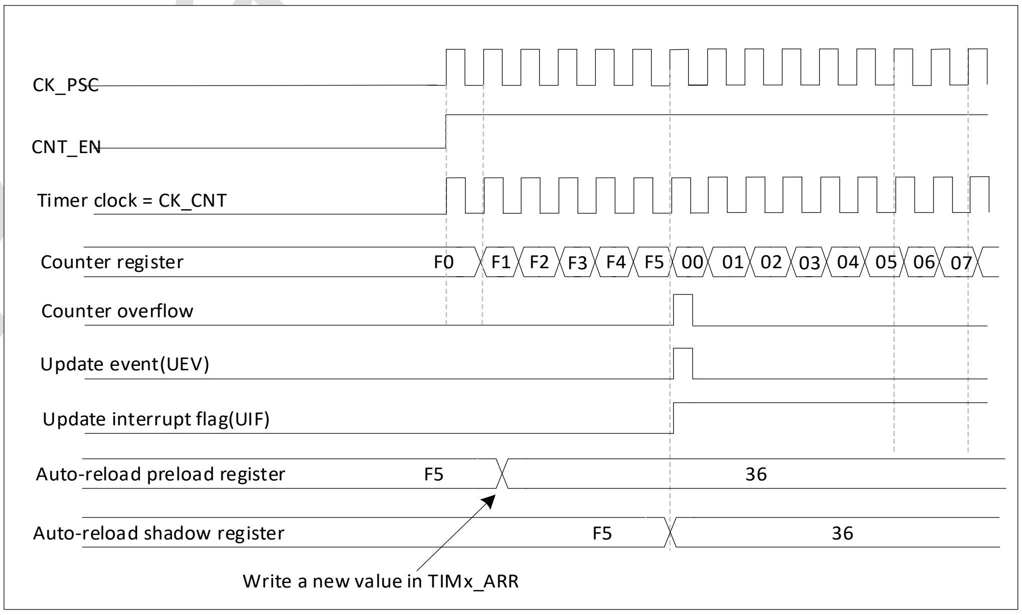


Figure 21-9 counter timing diagram, update events when ARPE = 1

21.2.3. Repetition counter

The Time-base unit describes how an update event (UEV) is generated when the counter over-flows, however in practice it can only be generated when the repetition counter reaches 0. This feature is useful for generating PWM signals.

This means that at every N count overflows, data is transferred from the preload register to the shadow register (TIMx _ ARR auto-reload register, TIMx _ PSC prescalation register, and also the capture/compare register TIMx _ CCRx in comparison mode), N being the value in the TIMx _ RCR repeat counter register.

Repeat counter, decremented as each count overflows in the up counter mode.

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 into the repetition counter.

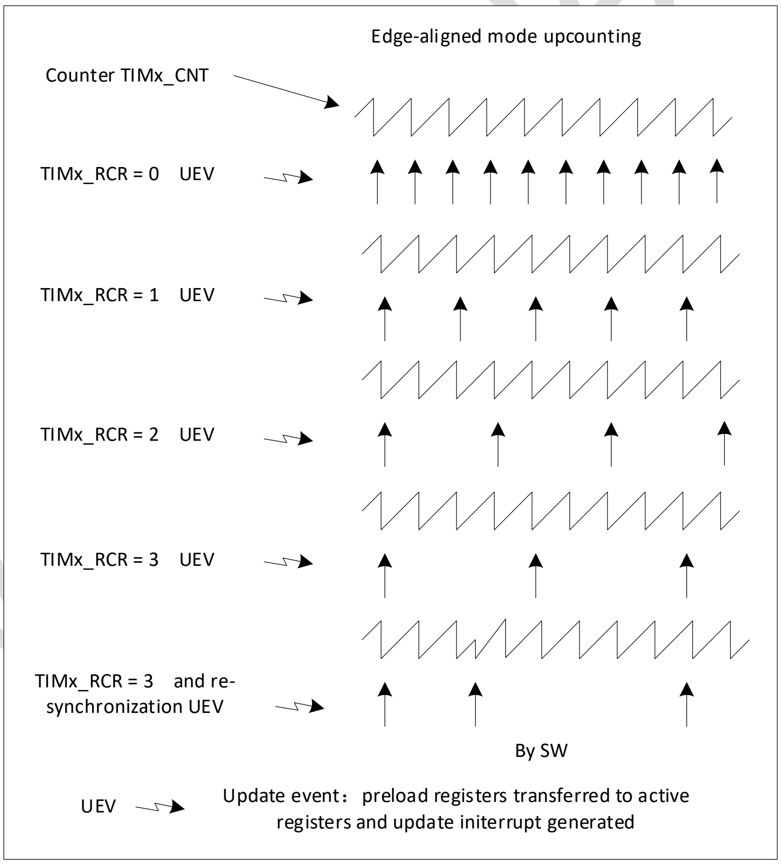

Edge-aligned mode upcounting
Counter TIMx_CNT
TIMx_RCR = 0 UEV
TIMx_RCR = 1 UEV
TIMx_RCR = 2 UEV
TIMx_RCR = 3 UEV
TIMx_RCR = 3 and re-synchronization UEV
By SW
UEV
Update event：preload registers transferred to active registers and update initerrupt generated

Figure 21-10 Examples of update rates in different modes, and register settings for TIMx_RCR

21.2.4. Clock sources

The counter clock is provided by the Internal clock (CK_INT) source. The CEN bit of the TIMx _ CR1 register and the UG bit of the TIMx _ EGR register are the actual control bits (except that the UG bit is automatically cleared) and can only be changed by software. Once the CEN bit is set to 1, the internal clock provides clock to the frequency divider.

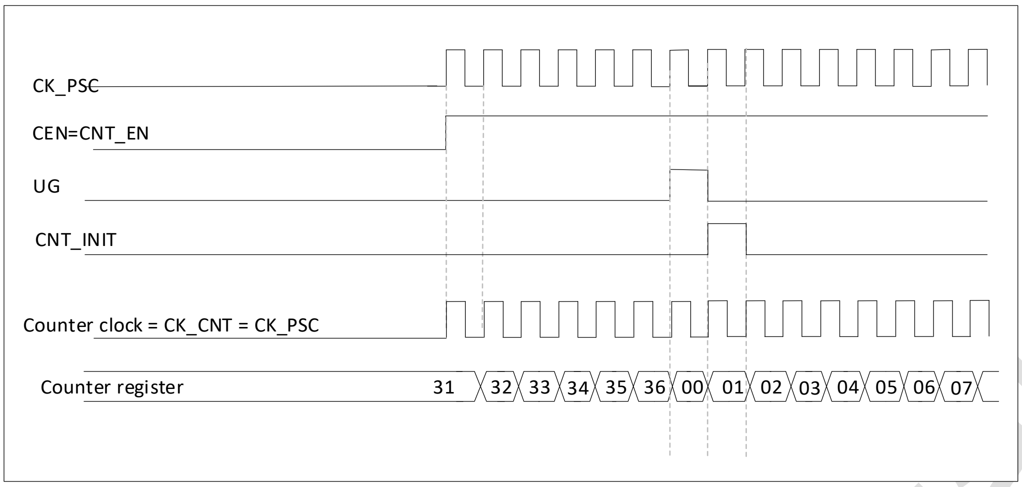


Figure 21-11 Control circuit in normal mode, internal clock divided by 1

21.2.5. 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 a capture/compare channel overview.

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. An edge detector with polarity selection then generates a signal (TixFPx) that can be triggered as an input from the mode controller or as a capture control. The signal enters the capture register (IcxPS) by pre-division.

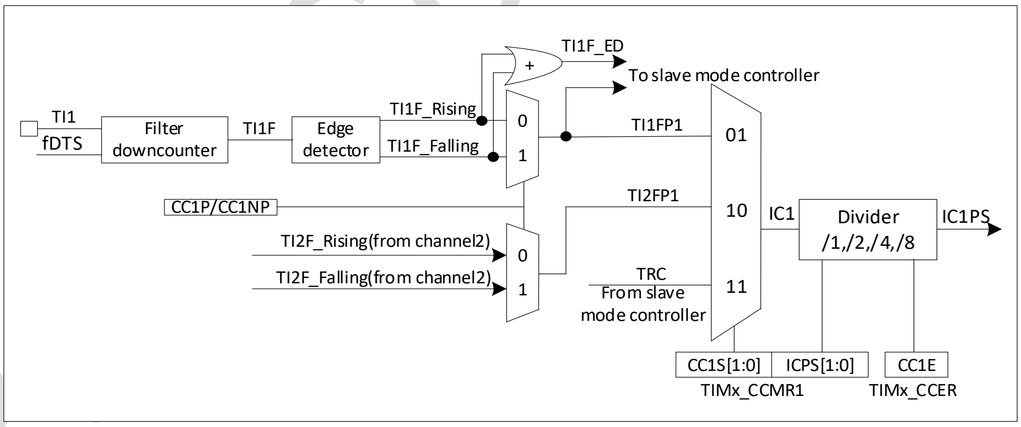


Figure 21-12 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform (active high) which is then used for reference. The polarity acts at the end of the chain.

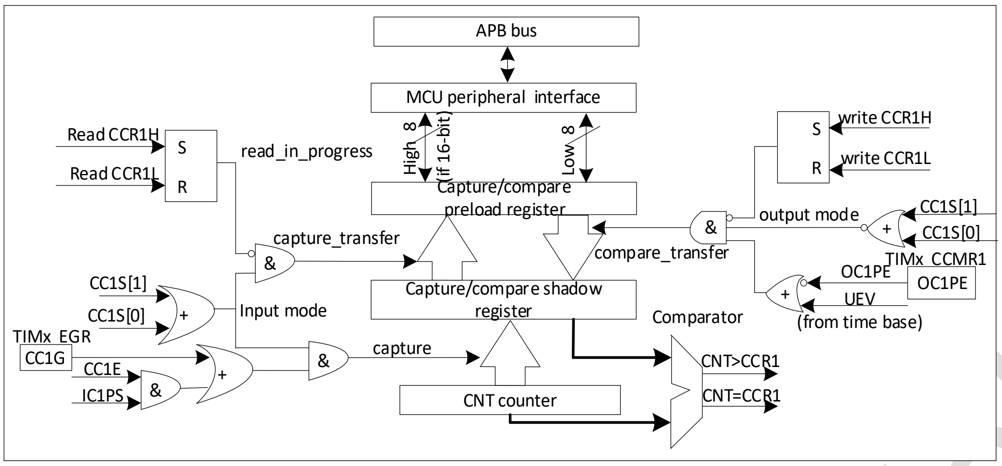


Figure 21-13 Capture/compare channel 1 main circuit

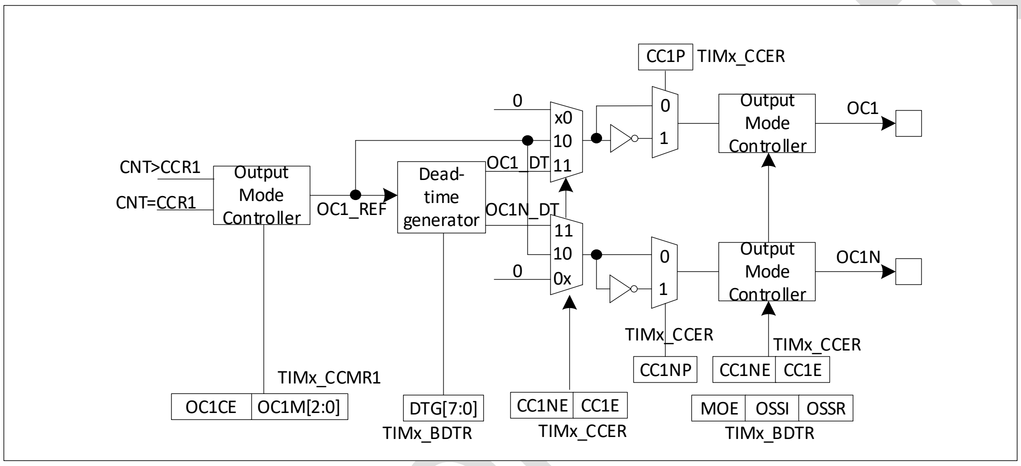


Figure 21-14 Capture/Compare the output stage of channel 1

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.

21.2.6. Input capture mode:

In input capture mode, when the corresponding edge of 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 repeat capture flag CcxOF (TIMx _ SR register) is set to 1. Writing to CcxIF clears the CcxIF, or reading captured data in storage also clears the CcxIF. Writing CcxOF = 0 clears CcxOF.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TMx_CCR1 register becomes read-only.
- Depending on the characteristics of the input signal, configure the input filter to the desired width (that is, when the input is Tix, the input filter control bit is the IcxF bit in the TIMx _ CCMRx register). Let's imagine that, when toggling, the input signal is not stable during at least 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 TIMxCCMR1 register.
- Select the valid transition edge of the TI1 channel and write CC1P = 0 (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 necessary, associated interrupt requests are allowed by setting the CC1IE bit in the TIMx _ DIER register, and DMA requests are 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.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- If the CC1IE bit is set, an interrupt request will be generated.
- If the CC1DE bit is set, a DMA request is generated

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 requests can be generated by software by setting the corresponding CCxG bit inthe TIMx_EGR register.

21.2.7. Forced output mode

In this mode (CCxS bits = 00 in the TIMx _ CCMRx register), the output comparison signals (OCxREF and corresponding OCx) can be directly set to active or invalid by the software, regardless of the comparison result between the output comparison register and the counter.

Write the corresponding OCxM = 101 in the TIMx _ CCMRx register to force the output comparison signal (OCxREF/OCx) to be valid. 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 and DMA requests 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 indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- The values defined by the output comparison mode (OCxM bit in the TIMx _ CCMRx register) and the output polarity (CCxP bit in the TIMx _ CCER register) are output to the corresponding pins. 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 the flag bit in the interrupt status register (the CcxIF bit in the TIMx _ SR register).
- If the corresponding interrupt mask (CcxIE bit in the TIMx _ DIER register) is set, an interrupt is generated.

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

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. If you want to generate an interrupt request, set the CcxIE bit.
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 at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE= ’0’, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the figure below.

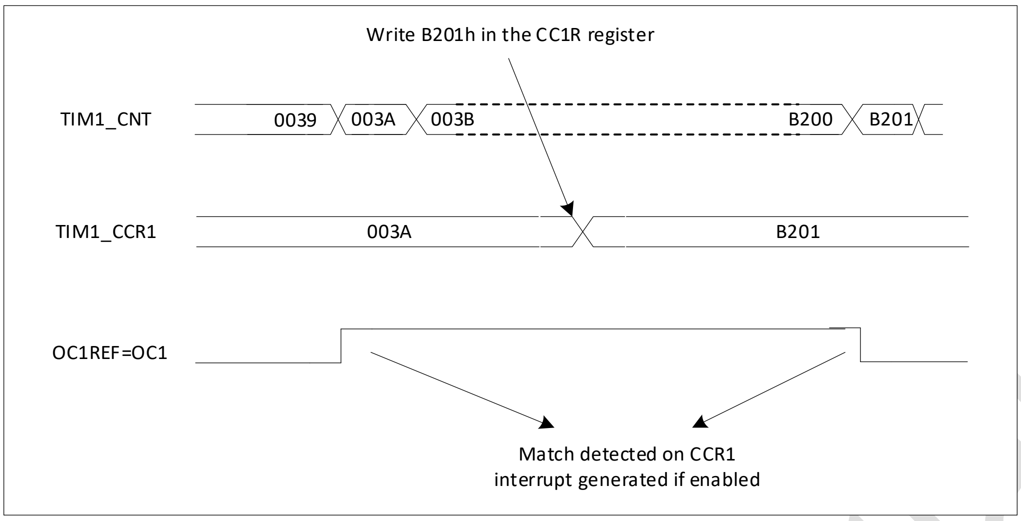


Figure 21-15 Output compare mode, toggle on OC1

21.2.9. PWM mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. The TIMx _ CCMRx register OCxPE bit must be set to enable the corresponding preload register, and finally the ARPE bit in the TIMx _ CR1 register (in up-count or centrosymmetric mode) to enable the preload register for auto-reload.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

The polarity of OCx can be set by software in the CCxP bit in the TIMx _ CCER register, which can be set to high active or low active. The CCxE bit in the TIMx _ CCER register controls the OCx output enable.

In PWM mode (Mode 1 or Mode 2), TIMx _ CNT and TIMx _ CCRx are always compared to determine whether TIMx _ CNT ≤ TIMx _ CCRx is compliant.

The timer is able to generate PWM in edge-aligned mode only since the counter is upcounting.

PWM edge-aligned mode

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 compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

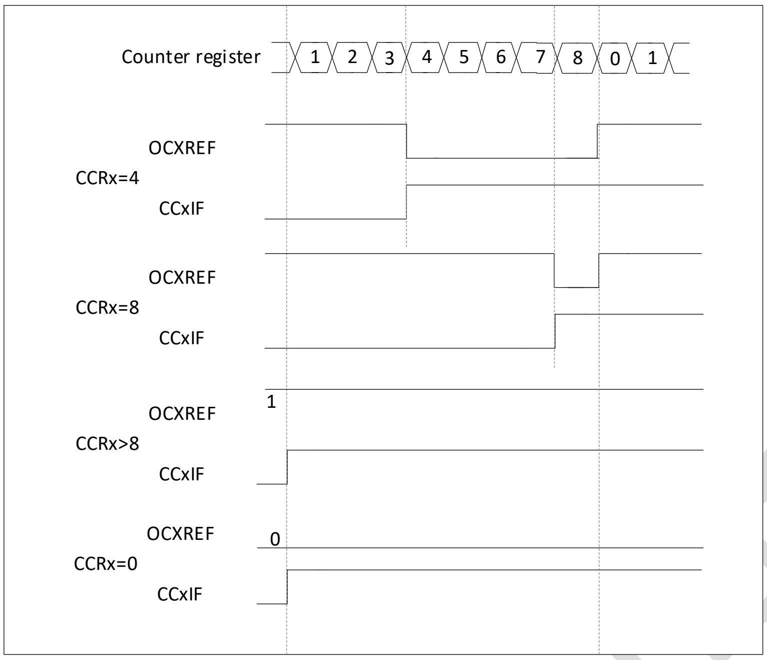


Figure 21-16 Edge-aligned PWM waveform (ARR = 8)

21.2.10. Complementary outputs and dead-time insertion

The TIM16/17 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 activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. 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. There is one 8-bit dead-time generator for each channel. 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)

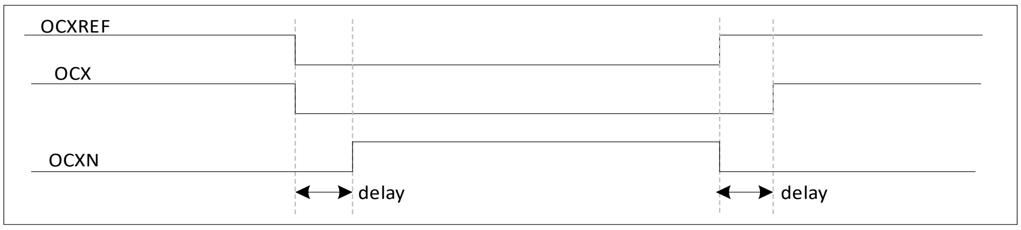


Figure 21-17 Complementary output with dead-time insertion

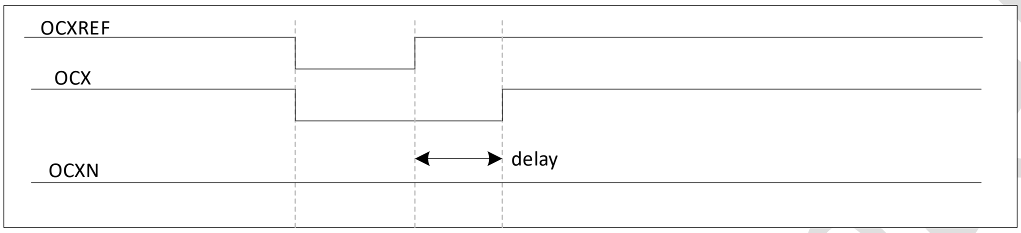


Figure 21-18 Dead-time waveforms with delay greater than the negative pulse

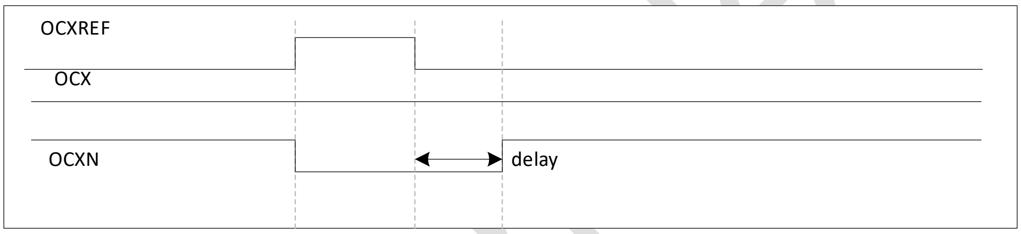


Figure 21-19 Dead-time waveforms with delay greater than the positive pulse.

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

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 is not complemented and becomes active as soon as OCxREF is high. 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.

21.2.11. Using the break function

When the brake function is used, both the output enable signal and the invalid level signal are modified according to additional control bits (MOE, OSSI and OSSR in the TIMx _ BDTR register, OISx and OISxN in the TIMx _ CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The brake source (BRK) signal channel may be an external input signal (BKIN) or an internal signal as follows:

- Kernel LOCKUP output
- the PVD output
- Clock failure event signal generated by CSS detection
- the output from a comparator

When exiting from reset, the break circuit is disabled and the MOE bit is low. You can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. 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.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the output into an invalid state, an idle state, or a reset state (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 then the timer releases the enable output else the enable output remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that due to the resynchronization of the MOE, the dead time is longer than usual (approximately 2 clock cycles of ck _ tim).
  - If OSSI=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set. If the BDE bit in the TIM1 _ 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 at the next update event UEV; For example, this can be used for shaping. Otherwise, the MOE remains

low until it is set '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: 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.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR Register.

Here are two ways to generate brakes:

- BKE is simultaneously enabled in the TIMx _ BDTR register through the BKR input of programmable polarity
- The BG bit in TIMx _ EGR is set by software.

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 to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The user can select one of the three levels of protection through the LOCK bit in the TIMx _ BDTR register, see TIM1 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.

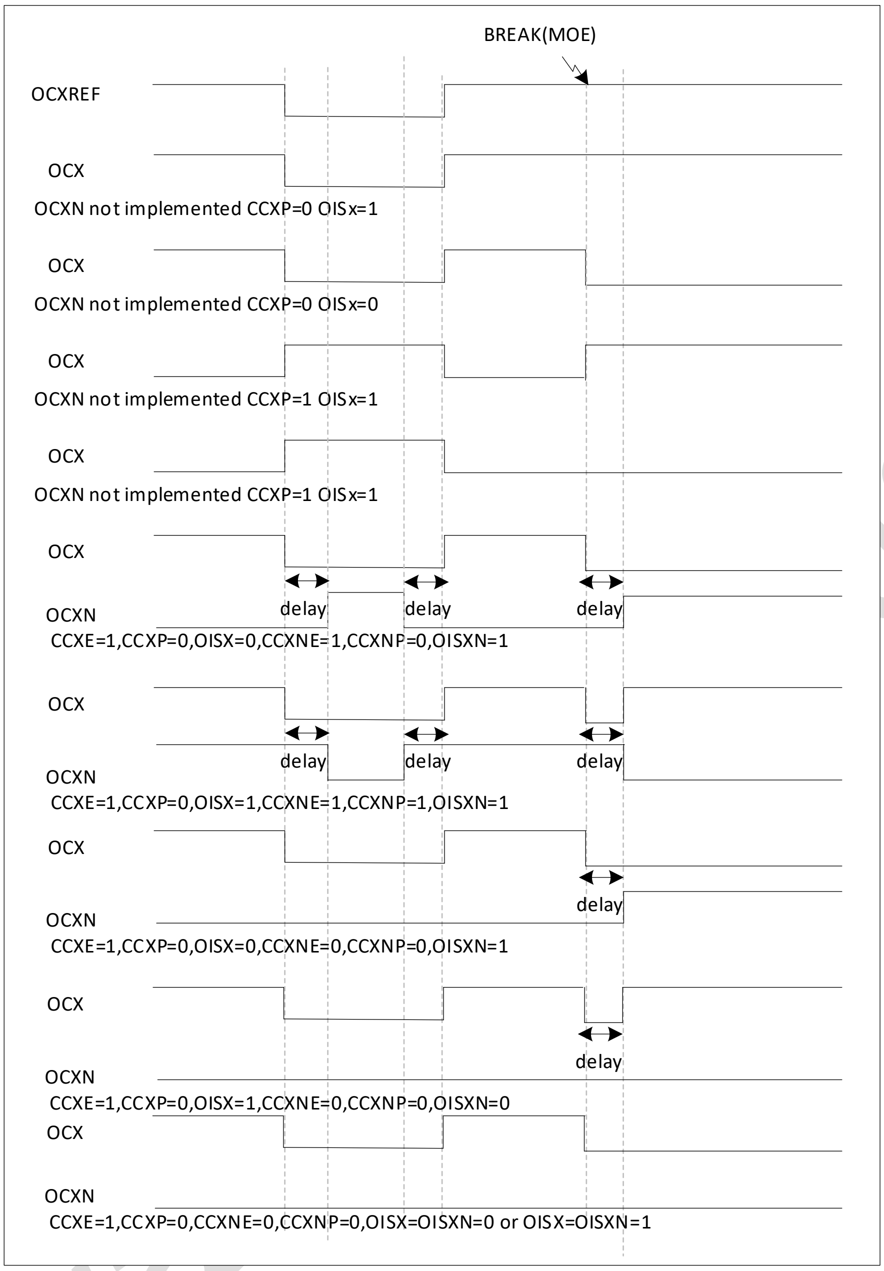

BREAK(MOE)
OCXREF
OCX
OCXN not implemented CCXP=0 OISx=1
OCX
OCXN not implemented CCXP=0 OISx=0
OCX
OCXN not implemented CCXP=1 OISx=1
OCX
OCXN not implemented CCXP=1 OISx=1
OCX
delay
delay
delay
OCXN
CCXE=1,CCXP=0,OISX=0,CCXNE=1,CCXNP=0,OISXN=1
OCX
delay
delay
delay
OCXN
CCXE=1,CCXP=0,OISX=1,CCXNE=1,CCXNP=1,OISXN=1
OCX
delay
OCXN
CCXE=1,CCXP=0,OISX=0,CCXNE=0,CCXNP=0,OISXN=1
OCX
delay
OCXN
CCXE=1,CCXP=0,OISX=1,CCXNE=0,CCXNP=0,OISXN=0
OCX
OCXN
CCXE=1,CCXP=0,CCXNE=0,CCXNP=0,OISX=OISXN=0 or OISX=OISXN=1

Figure 21-20 Output behavior in response to a break

21.2.12. 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:

- In up-count mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx)

- In down count mode: CNT > CCRx

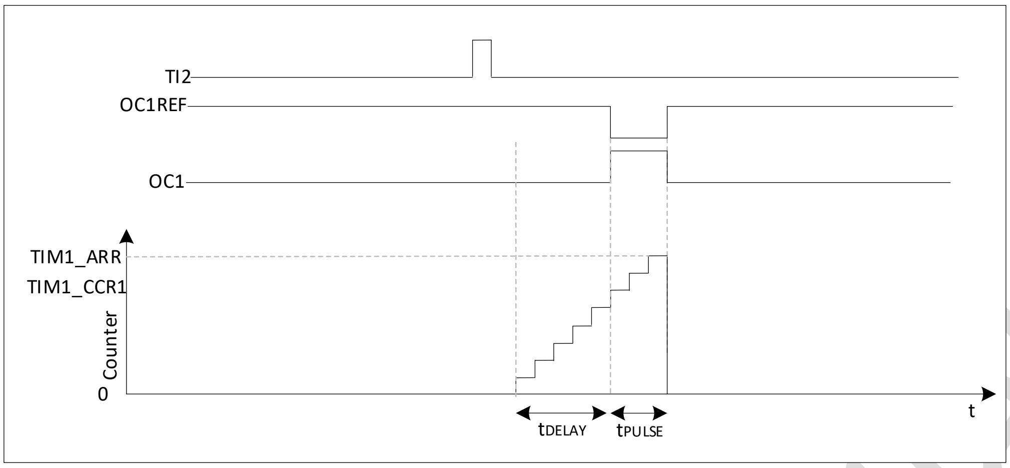


Figure 21-21 Example of one pulse mode

21.3. TIM16/TIM17 register

21.3.1. TIM16/17 control register 1 (TIMx _ CR1)

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	CKD[1:0]		ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
						RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock Division Factor These 2 bits define the division ratio between the timer clock (CK _ INT) frequency, the dead time and the sampling clock used by the dead generator and the digital filter (ETR, Tix) 00: tDTS = tCK _ INT01: tDTS = 2 x tCK _ INT10: tDTS = 4 x tCK _ INT11: Re-serve, do not use this configuration
7	ARPE	RW	0	Automatic reload preload allowed bit 0: TIM1 _ ARR regis-ter is not buffered 1: TIM1 _ ARR register is loaded into buffer
6:4	Reserved	-	-	Reserved
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clear-ing the bit CEN).
2	URS	RW	0	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 allowed, either of the following events produces 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 allowed, only a counter overflow/underflow generates an update interrupt or DMA request
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable UEV event generation.

Bit	Name	R/W	Reset Value	Function
				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. 1: UEV disabled. The update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). 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: Counter disabled 1: Counter enabled 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. TIM16/17 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	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	OIS1N	OIS1	Res	Res	Res	Res	CCDS	Res	Res	CCPC
-	-	-	-	-	-	RW	RW	-	-	-	-	RW	-	-	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: This bit cannot be modified after the LOCK (TIM1 _ BKR register) level 1, 2, or 3 has been set.
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 Note: This bit cannot be modified after the LOCK (TIM1 _ BKR register) level 1, 2, or 3 has been set.
7:4	Reserved	-	-	Reserved
3	CCDS	RW	0	DMA Selection for Capture/Comparison 0: When a CCx event occurs, a DMA request for CCx is sent. 1: When an update event occurs, a DMA request for CCx is sent.
2: 1	Reserved	-	-	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; After this bit is set, they are only updated after the COMG bit is set. Note: This bit acts only on channels that have a complementary output.

21.3.3. TIM16/17 DMA/interrupt enable register (TIM16/17 _ 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	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**
						CC1DE	UDE	BIE	Res	COMIE	Res	Res	Res	CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9	CC1DE	RW	0	CC1DE: DMA request enabled to capture/compare 1 0: DMA request for capture/compare 1 is disabled 1: DMA request enabled to capture/compare 1
8	UDE	RW	0	UDE: Updated DMA requests enabled 0: Updated DMA requests disabled 1: Updated DMA requests enabled
7	BIE	RW	0	BIE: Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	Reserved	-	-	Reserved
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4:2	Reserved	-	-	Reserved
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

21.3.4. TIM16/17 status register (TIM16/17 _ SR)

Address offset: 0x010

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	IC1IF	Res	Res	Res	IC1IR
											RC_W0	-	-	-	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CC1OF	Res	BIF	Res	COMIF	Res	Res	Res	CC1F	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	Reserved
20	IC1IF	Rc_w0	0	Falling Edge Capture 1 Flag This flag can be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the falling edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A falling edge capture event occurs.
19:17	Reserved	-	-	Reserved
16	IC1IR	Rc_w0	0	Rising Edge Capture 1 Flag This flag may be set to 1 by the hardware only when the respective channel is configured as input capture and the capture event is triggered by the rising edge. It is cleared by software or reading TIMx_CCR1. 0: No duplicate capture generation; 1: A rising edge capture event occurs.
15:10	Reserved	-	-	Reserved
9	CC1OF	Rc_w0	0	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’.

Bit	Name	R/W	Reset Value	Function
				0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	Reserved	-	-	Reserved
7	BIF	Rc_w0	0	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.
6	Reserved	-	-	Reserved
5	COMIF	Rc_w0	0	COM interrupt flag Once a COM event is generated (when CCxE, CCxNE, OCxM have been updated) this bit is set to 1 by the hardware. It is cleared by software 0. 0: No update occurred. 1: COM interrupt pending.
4:2	Reserved	-	-	Reserved
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set to 1 by hardware when the counter value matches the comparison value, except in centrosymmetric mode (refer to TIM1 _ CR1 register Device's CMS bit). It is cleared by software. 0: No match; 1: The value of TIMx _ CNT matches the value of TIMx _ CCR1. If channel CC1 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 in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	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 by hardware when the registers are updated: − If the UDIS of the TIMx _ CR1 register = 0, an update event (counter overflow) is generated; If UDIS of the TIMx _ CR1 register = 0 and URS = 0, an update occurs when UG of the TIMx _ EGR register = 1 Software (software re-initializes CNT);

21.3.5. TIM16/17 event generation register (TIM16/17 _ EGR)

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	Res	BG	Res	COMG	Res	Res	Res	CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	Reserved
7	BG	W	0	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action

Bit	Name	R/W	Reset Value	Function
				1: A break event is generated. At this time, MOE = 0 and BIF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated.
6	Reserved	-	-	Reserved
5	COMG	W	0	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC = 1, CCxE, CCxNE, OCxM bits are allowed to be updated. Note: This bit is only valid for channels with complementary outputs.
4:2	Reserved	-	-	Reserved
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: Set CC1IF = 1. If the corresponding interrupt and DMA are turned on, the corresponding interrupt and DMA will be generated. If channel CC1 is configured as input: The current counter value is captured to the TIMx _ CCR1 register, and CC1IF = 1 is set. 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	Generate an update event This bit can be set by software, it is automatically cleared by 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 The coefficient remains unchanged). If in centrosymmetric mode or DIR = 0 (count up), the counter is cleared, and if DIR = 1 (count down), the counter takes the value of TIMx _ ARR.

21.3.6. TIM16/17 capture/compare mode register 1 (TIM16/17 _ CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare 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	Res	Res	Res	Res	Res	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
Res	Res	Res	Res	Res	Res	Res	Res	IC1F [3:0]				IC1PSC [1:0]			
								RW	RW	RW	RW	RW	RW	RW	

Output compare mode

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	-	Reserved
6:4	OC1M [2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 andOC1N 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 comparison register TIM1 _ CCR1 and the counter TIM1 _ CNT does not work for OC1REF With;

Bit	Name	R/W	Reset Value	Function
				001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 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-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. 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 Compare 1 preload enable 0: Preload register on TIM1_CCR1 disabled. TIM1_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Turn on the preload function of the TIM1 _ CCR1 register. Read and write operations only operate on the preload register. The preload value of TIM1 _ 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: The PWM mode can be used without validating the preload register only in one pulse mode. Else the behavior is not guaranteed.
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 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. Therefore, OC is set to a comparison level and It is not related to the comparison results. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OC1FE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10 Reserved 11 Reserved Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIM16/17 _ CCER register).

Input capture mode:

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 in-put and the length of the digital filter applied to TI1. 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, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING = fCK _ INT, N = 2 1001: Sampling frequency fSAMPLING = fDTS/8, N = 8 0010: Sampling frequency fSAMPLING = fCK _ INT, N = 4 1010: Sampling frequency fSAMPLING = fDTS/16, N = 5 0011: Sampling frequency fSAMPLING = fCK _ INT, N = 8 1011: Sampling frequency fSAMPLING = fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is de-tected 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 [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (in-put/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10 Reserved 11 Reserved Note: CC1S is writable only when the channel is closed (CC1E = 0 in the TIM16/17 _ CCER register).

Table 21-22 Output control of complementary OCx and OCxN channels with interrupt function

Control bits					Output state	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
1	X	0	0	0	Output disabled (not driven by timer), OCx = 0, OCx _ EN = 0	Output disabled (not driven by timer), OCxN = 0, OCxN _ EN = 0
		0	0	1	Output disabled (not driven by timer), OCx = 0, OCx _ EN = 0	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + Polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (and not driven by timer), OCxN = 0, OCxN _ EN = 0
		0	1	1		
		1	0	0		
		1	0	1		
		1	1	0		
		1	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementarity of OC-REF (not OCREF)
0	0	X	0	0		
	0		0	1		
	0		1	0		

Control bits					Output state	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
	0		1	1		
	1		0	0		
	1		0	1		
	1		1	0		
	1		1	1		

21.3.7. TIM16/17 Capture/Compare enable register (TIM16/17 _ 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	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	CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：4	Reserved	-	-	Reserved
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. Note: Once the LOCK level (LCCK bit in the TIMx _ BDTR register) is set to 3 or 2 and CC1S = 00 (channel configu-ration as output), this bit cannot be modified.
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1Nand 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, CC1E bits.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: The two-bit CC1NP/CC1P selects whether the polarity sig-nal of TI1FP1 or TI2FP1 is used as the trigger or capture signal. 00: non-inverting/rising edge: capture occurs on the rising edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 is not inverted (gate triggered mode, encoded mode). 01: inverting/falling edge: non-inverting/rising edge: cap-ture occurs on the falling edge of TixFP1 (capture, reset trigger, external clock or trigger mode); TixFP1 inverted (gate triggered mode, encoded mode). 10: Reserved, invalid configuration. 11: non-inverted/both edges Note: Once the LOCK level (LOCK bit in the TIMx _ BDTR register) is set to 3 or 2 and CC1S = 00 (channel configu-ration as output), this bit cannot be modified.

Bit	Name	R/W	Reset Value	Function
0	CC1E	RW	0	Input/Capture 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, 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, CC1NE bits. **The CC1 channel is configured as input:** This bit determines whether the value of the counter can capture the TIMx _ CCR1 register. 0: Capture disabled. 1: Capture enabled.

21.3.8. TIM16/17 Calculator (TIM16/17 _ 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	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															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

21.3.9. TIM16/17 Prescaler (TIM16/17 _ PSC)

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**
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). The PSC contains the value loaded into the current prescaler register when an update event occurs; The update event includes the counter being cleared 0 by the UG bit of TIMx _ EGR or cleared 0 by a slave controller operating in reset mode.

21.3.10. TIM16/17 automatic reload register (TIM16/17 _ 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	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**
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0xFFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

21.3.11. TIM16/17 repetition counter register (TIM16/17 _ 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	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	REP[7:0]							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	0	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 send from the preload Register transfer to the current register); If an update interrupt is allowed, it will also affect the rate at which the update interrupt is generated. Each time the REP_CNT related downcounter reaches zero, an update event is generated, and it restarts counting from REP value. Since REP _ CNT overloads the REP value only when the cycle update event U _ RC occurs, the new value written to the TIMx _ RCR register only takes effect when the next cycle update event occurs. It means 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.

21.3.12. TIM16/17 capture/compare register 1 (TIM16/17 _ 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	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**
CCR1 [15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value **If channel CC1 is configured as output:** CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). 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 is the counter value transferred by the last input capture 1 event (IC1).

21.3.13. TIM16/17 brake and dead time register (TIM16/17 _ 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	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**
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
31：16	Reserved	-	-	Reserved
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is cleared by software or automatically setting depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIM1_CCER register). OCN Output。 Refer to the detailed description of the OC/OCN enable (Capture/Compare Enable Register (TIMx _ CCER)).
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). 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: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
12	BKE	RW	0	Break enable 0: Disable brake input (BRK and BRK _ ACTH); 1: Turn on the brake input (BRK and BRK _ ACTH). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
11	OSSR	RW	0	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 enabling. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0).

Bit	Name	R/W	Reset Value	Function
				1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1. 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 for Idle mode This bit is used when MOE=0 and on channels configured as outputs. Refer to the detailed description of OC/OCN enabling. 0: When inactive, OC/OCN outputs are disabled (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. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: Lock level 1, cannot write to the DTG/BKE/BKP/AOE bit of the TIM1 _ BDTR register and the OISx/OISxN bit of the TIM1 _ CR2 register; 10: Lock level 2, cannot write bits in lock level 1, nor can you write CC polarity bits (CCxP/CCNxP bits of TIM1 _ CCER register once the relevant channel is set as output through CCxS bits) and OSSR/OSSI bits; 11: Lock level 3, cannot write bits in lock level 2, nor can you write CC control bits (OCxM/OCxPE bits of TIM1 _ CCMRx register once the relevant channel is set as output through CCxS bits); Note: After system reset, the LOCK bit can only be written once. Once written to the TIMx _ BDTR register, its contents freeze straight To reset.
7:0	DTG[7:0]	RW	0000 0000	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] × T_{dtg}， T_{dtg} = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG[5:0]) × T_{dtg}， T_{dtg} = 2 × T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) × T_{dtg}， T_{dtg} = 8 × T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) × T_{dtg}， T_{dtg} = 16 × T_{DTS}； Example: If TDTS = 125ns (8MHZ), the possible dead time is: 0 to 15875 ns by 125 ns steps; 16us to 31750ns, if the step time is 250ns; 32us to 63us, if the step time is 1us; 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 repaired Change.

21.3.14. TIM16/17 DMA control register (TIM16/17 _ DCR)

Address offset: 0x48

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	DBL[4:0]					Res	Res	Res	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：13	Reserved	-	-	Reserved

12:8	DBL[4:0]	RW	0 0000	DMA continuous transfer length These bits define the transfer length of the DMA (when reading or writing to the address of the TIM16/17 _ DMAR register When, the timer performs a continuous transmission), the transmission can be half-word or byte mode: 00000: 1 transmission 00001: 2 transmissions 00010: 3 transmissions ...... 10001: 18 transmissions
7:5	Reserved	-	-	Reserved
4:0	DBA[4:0]	RW	0 0000	DBA [4: 0]: DMA Base address These bits define the base address of the DMA in continu-ous mode (when reading or writing to the address of the TIM16/17 _ DMAR register), and the DBA is defined as the offset from the address of the TIM1 _ CR16/17 regis-ter: 00000: TIM16/17 _ CR1, 00001: TIM16/17 _ CR2, 00010: TIM16/17 _ SMCR, ...

21.3.15. DMA address for TIM16/17 continuous mode (TIM16/17 _ DMAR)

Address offset: 0x4C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	DMAB[31:0]	RW	0	DMA continuous transfer register A read or write to the TIM16/17 _ DMAR register results in an access operation to a register at the following address: TIM16/17 _ CR1 address + DBA + DMA pointer, where: "TIM16/17 _ CR1 address" is the address of the control register 1; "DBA" is the base address defined in the TIM1 _ DCR reg-ister; The "DMA pointer" is an offset automatically controlled by the DMA, which depends on the DBL defined in the TIM16/17 _ DCR register.

22. 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. 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 infra-red 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 gener-ate 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. LEDs require large sink current driving capabilities (for PA0, PA15, PB2 ~ PB7), which can be activated through the PA _ EHS register and PB _ EHS register of the SYSCFG _ IOCFG register, which is sufficient to support direct control of the large sink current of infrared LEDs.

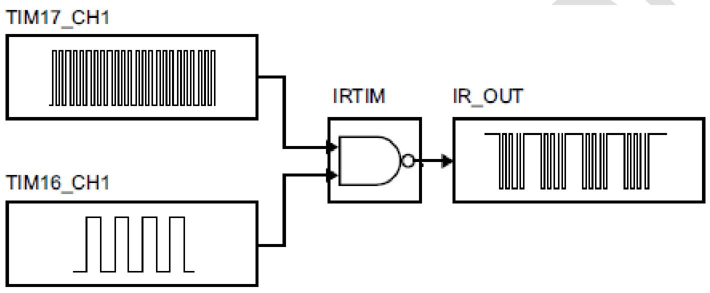


Figure 22-1 IRTIM implementation

23. Low-power timer (LPTIM)

23.1. Introduction

The LPTIM is a 16-bit timer. The ability of LPTIM to wake up the system from low power mode makes it suitable for implementing low power applications.

LPTIM introduces a flexible clocking scheme that delivers the required functionality and performance while minimizing power consumption.

23.2. Main features

- 16 bit upcounter
- 3-bit prescaler with 8 possible dividing factors (1, 2, 4, 8, 16, 32, 64, 128)
- Selectable clock
  - Internal clock sources: LSE, LSI or APB clock
- 16 BIT ARR reloadable register
- Continous mode

23.3. Functional Description

23.3.1. LPTIM block diagram

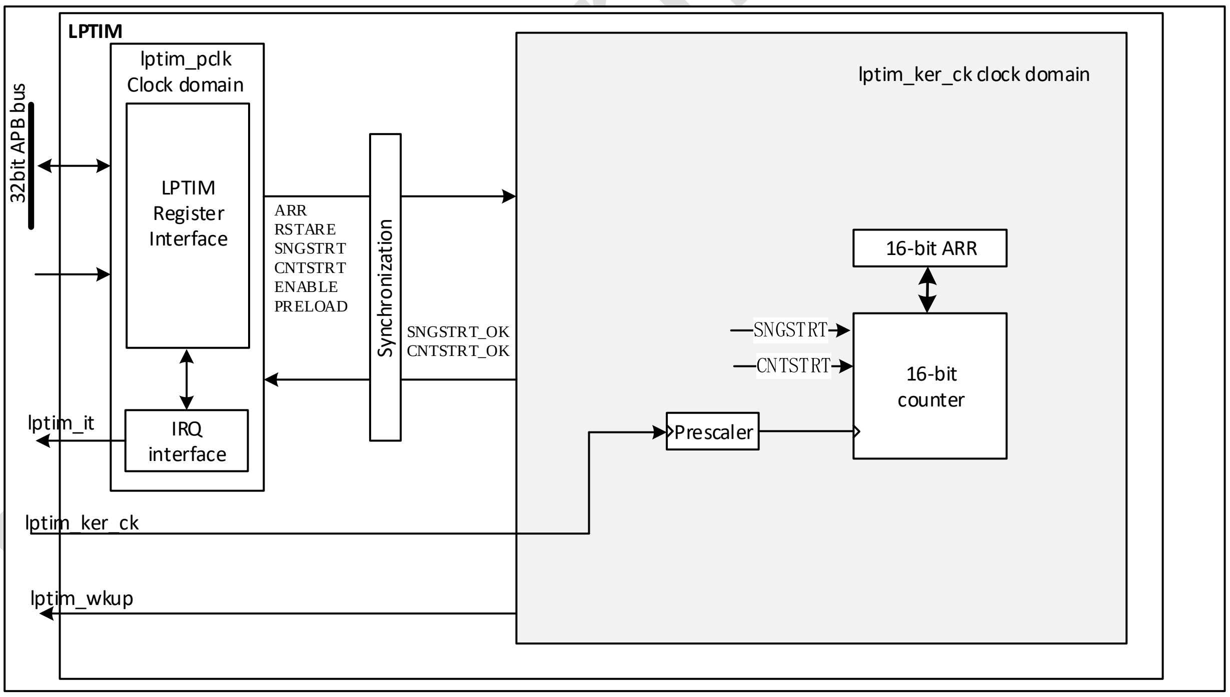


Figure 23-1 Block diagram of low power timer

23.3.2. LPTIM pins and internal signals

Table 23-1 LPTIM internal signals

Names	Signal type	Description
Lptim _ pclk	Digital input	LPTIM APB clock domain

Names	Signal type	Description
Lptim _ ker _ ck	Digital input	LPTIM kernel clock
Lptim _ it	Digital output	LPTIM global interrupt
Lptim _ wakeup	Digital output	LPTIM wakeup event

23.3.3. LPTIM reset and clocks

The LPTIM can be clocked using several clock sources.

Through the RCC module, it can be clocked using an internal clock signal (which can be selected among APB, LSI, LSE sources.

23.3.4. Prescaler

The LPTIM 16-bit counter is preceded by a configurable power-of-2 prescaler. The prescaler division ratio is controlled by the PRESC[2:0].

The table below lists all the possible division ratios:

Table 23-2 Pre-division coefficients

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

23.3.5. Working Mode

LPTIM has two operating modes as follows (only single count is supported):

- 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.

Single count mode

To enable single count, the LPTIM _ CR.SNGSTRT register bit must be set to 1.

Setting SNGSTRT will start the counter for a single count. 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.

The single count waveform is as follows:

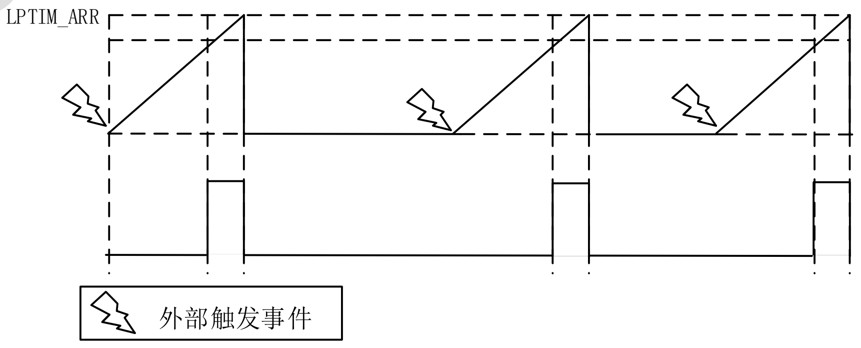


Figure 23-2 Low power timer output waveform, single count mode

Continuous counting mode

To enable the continuous counting, the LPTIM_CR.CNTSTRT bit must be set.

LPTIM_CR.CNTSTRT is set will start the counter for continuous counting.

The continuous counting waveform is as follows:

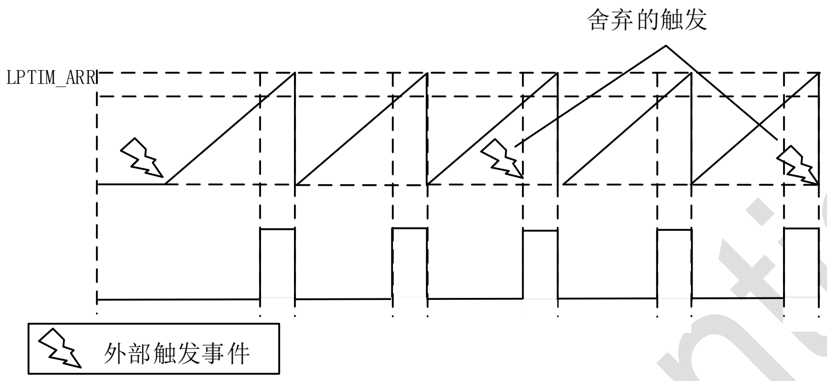


Figure 23-3 Low power timer output waveform, continuous counting mode

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 transition from single-shot mode to continuous mode "on the fly":

- If the Continuous mode was previously selected, setting SNGSTRT will switch the LPTIM to the One-shot mode. The counter (if active) will stop 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 active) will restart as soon as it reaches ARR.

23.3.6. Register update

The LPTIM _ ARR register is updated immediately after the APB bus write operation, and if the timer has been started, it is updated in synchronization with the next LPTIM update event.

The PRELOAD bit controls how the LPTIM_ARR register is updated:

- When the PRELOAD bit is reset to ‘0’, the LPTIM_ARR and the LPTIM_CMP registers are immediately updated after any write access.
- When the PRELOAD bit is set to "1": If the timer has been started, LPTIM _ ARR will be updated at the end of the current cycle.

The LPTIM APB interface and the LPTIM Kernel logic use different clocks, so there is a certain delay when the APB writes, and the written values are applied to the counter comparator. Within this latency period, any additional write into these registers must be avoided.

The ARROK flag in the LPTIM_ISR register indicates when the write operation is completed to the LPTIM_ARR register.

After a write to the LPTIM_ARR register, a new write operation to the same register can only be performed when the previous write operation is completed. Any successive write before respectively the ARROK flag be set, will lead to unpredictable results.

23.3.7. Enable timer

The ENABLE bit located in the LPTIM_CR register is used to enable/disable the 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.

23.3.8. Timer counter reset

In order to reset the content of LPTIM_CNT register to zero, reset mechanism is implemented:

Asynchronous reset mechanism:

The asynchronous reset is controlled by the RSTARE bit of the LPTIM _ CR register. When this bit is set to 1, any access to the LPTIM _ CNT register resets its contents to zero.
It should be noted that in order to reliably read the LPTIM _ CNT register, two read accesses must be performed and the results are compared. If the results are consistent, it is considered that the read value is reliable.
It is important to note that:

- When asynchronous reset is enabled, the results of two reads of the LPTIM _ CNT register are different (the first read resets LPTIM _ CNT).
- When PCLK is selected by the LPTIM counting clock, two consecutive accesses cannot guarantee that the read value is reliable.

Synchronous reset mechanism

- The synchronous reset is controlled by the COUNTRST bit in the LPTIM_CR register. After the COUNTRST position is 1, the reset signal is sent to the Kernel clock domain of LPTIM. Therefore, it should be noted that the logic circuit of the LPTIM Kernel clock domain has passed several clock cycles before the reset takes effect. This will make the LPTIM counter count few extra pluses between the time when the reset is trigger, and it becomes effective.
- Since COUNTRST is in the APB clock domain and the LPTIM counter is in the LPTIM Kernel clock domain, a delay of 3 clock cycles of the Kernel clock is required to synchronize the reset signal from the APB clock domain when writing 1 to the COUNTRST bit.

23.3.9. 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.

23.4. LPTIM Description

Table 23-3Differences between different low power consumption modes of LPTIM

Mode	Description
Sleep	No effect, an LPTIM interrupt causes the device to exit Sleep mode.
Stop	No effect, when LPTIM is controlled by LSE or LSI clock. An LPTIM interrupt causes the device to exit stop.

- Continuous counting mode, it takes at least one LSI clock cycle (about 30us) from waking up to entering the next WFI/WFE before the interrupt signal can be normally generated during the second waking up;

- In single count mode, single count needs to be started. After configuring the registers, 3 lptim _ ker _ ck are required to complete synchronization; Then enter the next low power consumption mode through the WFI/WFE instruction;

23.5. LPTIM interrupts

The following events generate an interrupt/wake event if they are enabled in the LPTIM_IER register:

- Automatically reload matches

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 is generated.

Interrupt event	Description
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).
Auto-reload register update OK	Interrupt flag is raised when the write operation to the LPTIM_ARR register is complete.

23.6. Register Description

23.6.1. LPTIM interrupt and status register (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	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	ARROK	Res	Res	ARRM	Res
											~~R~~			r	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	Reserved
4	ARROK	R	0	Automatic reload register update OK. ARROK is set by hardware to inform application that the APB bus write operation to the LPTIM_ARR register has been successfully completed. ARROK flag can be cleared by writing 1 to the ARROKCF bit in the LPTIM_ICR register.
3：2	Reserved	-	0	
1	ARRM	R	0	Auto-reload match ARRM is set by hardware to inform application that LPTIM_CNT register's value reached the LPTIM_ARR register's value. ARRM flag can be cleared by writing 1 to the ARRMCF bit in the LPTIM_ICR register.
0	Reserved	-	-	Reserved

23.6.2. LPTIM interrupt clear register (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	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	ARROKCF	Res	Res	ARRM CF	Res
											W			w	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	Reserved
4	ARROKCF	W	0	Automatic reload register updates OK clear flag. Writing 1 to this bit clears the ARROK flag in the LPTIM _ ISR register.
3：2	Reserved	-	-	Reserved
1	ARRMCF	W	0	Auto-reload match clear flag Writing 1 to this bit clears the ARRM flag in the LPTIM_ISR register
0	Reserved	-	-	Reserved

23.6.3. LPTIM interrupt enable register (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	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	ARROKIE	Res	Res	ARRM IE	Res
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	Reserved
4	ARROKIE	RW	0	Auto-reload register update OK interrupt enable 0: ARROK interrupt disabled 1: ARROK interrupt enabled
3：2	Reserved	-	0	
1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: ARRM interrupt disabled 1: ARRM interrupt enabled
0	Reserved	-	-	Reserved

23.6.4. LPTIM configuration register (LPTIM_CFGR)

Address offset: 0x00C

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	Re s	Re s	Re s	Re s	PRE-LOAD	Re s	Re s	Re s	Re s	Re s	Re s
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	PRESC[2:0]	Re s	Re s	Re s	Re s	Res	Re s	Re s	Re s	Re s		
				rw	rw	rw									

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	PRELOAD	RW	0	Register update mode Preload bit control LPTIM _ ARR and LPTIM _ CMP regis-ter update modes 0: Registers are updated after each APB bus write access 1: Registers are updated at the end of the current LPTIM period

Bit	Name	R/W	Reset Value	Function
21:12	Reserved	-	-	Reserved
11:9	PRESC[2:0]	RW	0	Clock prescaler The PRESC bits configure the prescaler division factor. It can be one among the following division factors: 000:/1 001:/2 010:/4 011:/8 100:/16 101:/32 110:/64 111:/128
8:0	Reserved	-	-	Reserved

23.6.5. LPTIM control register (LPTIM_CR)

Address offset: 0x010

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	RSTARE	COUN-TRST	COUN-TRST	SNGSTRT	ENA-BLE
											rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	RSTARE	RW	0	Reset after read enable This bit is set and cleared by software. When RSTARE is set to '1', any read access to LPTIM_CNT register will asynchronously reset LPTIM_CNT register content.
3	COUNTRST	RW	0	Timer counter reset This bit is set by software and cleared by hardware. When set to '1' this bit will trigger a synchronous reset of the LPTIM_CNT counter register. Due to the synchronous nature of this reset, it only takes place after a synchronization delay of 3 LPTIM core clock cycles (LPTIM core clock may be different from APB clock). Caution: COUNTRST must never be set to '1' by software before it is already cleared to '0' by hardware. Software should consequently check that COUNTRST bit is already cleared to '0' before attempting to set it to '1'.
2	CNTSTRT	RW	0	Timer start in Continuous mode This bit is set by the software. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between the LPTIM_ARR and LPTIM_CNT registers. The LPTIM counter keeps counting in continuous mode. Note: This bit can be set only when the LPTIM is enabled. It will be automatically reset by 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 and cleared by software. 0: LPTIM is disabled 1: LPTIM is enabled

23.6.6. LPTIM autoreload register (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	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
ARR[15:0]															
rw															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15：0	ARR	RW	0x0001	Auto reload value ARR is the auto-reload value for the LPTIM. The LPTIM_ARR register must only be modified when the LPTIM is enabled.

23.6.7. LPTIM counter register (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	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]															
R															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15：0	CNT	R	0	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 con-secutive read accesses and verify that the two returned values are identical. A read access can be considered reli-able when the values of the two consecutive read ac-cesses are equal.

24. Independent watchdog (IWDG)

24.1. Introduction

An Independent watchdog (IWDG for short) is integrated into the device, which has the characteristics of improving security level, accurate timing and flexible use. The IWDG detects and resolves functional clutter due to software failure and triggers a system reset when the counter reaches the specified timeout value.

The IWDG is clocked by LSI, so even if the main clock fails, it can keep working.

IWDG is the best suited for applications that require the watchdog as a standalone process outside of the main application and do not have high timing accuracy constraints.

24.2. IWDG main features

- Free-running down counter
- Clock provided by LSI (also works in Stop mode)
- Support hardware mode
- Configurable to stop counting in Stop mode
- Reset is generated when the down counter value is 0x000

24.3. IWDG functional description

24.3.1. IWDG block diagram

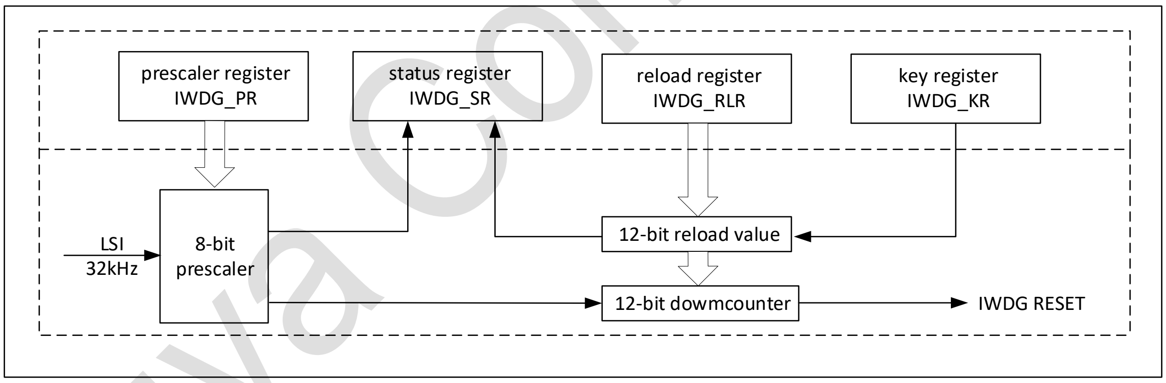


Figure 24-1 IWDG block diagram

When the independent watchdog is started by writing the value 0x0000 CCCC in the IWDG key register (IWDG_KR), the counter starts counting down from the reset value of 0xFFF. When it reaches the end of count value (0x000) a reset signal is generated (IWDG reset).

Whenever 0x0000 AAAA is written to the IWDG key register, the value of IWDG _ RLR (reload register) is reloaded into the counter, and IWDG does not generate a reset.

Once running, the IWDG cannot be stopped.

24.3.2. Hardware watchdog

If the option byte of the power-on load (option byte) is set to turn on the hardware watchdog, the IWDG power-on is automatically enabled and a reset signal is generated if the IWDG key register is not overwritten by software before the counter counts to the final value.

24.3.3. Register access protection

Write access to the registers IWDG Prescaler, IWDG Reload, and IWDG Window is protected. In order to modify them, the user must first write 0x0000 5555 to the IWDG Key register. Writing other numbers to these registers will destroy the timing, such as writing 0x0000AAAA load, and the registers will be protected again.

If the values of the Prescaler register, Reload register, and Window register are being updated, the status register will be reflected.

24.3.4. Debug mode

This function only exists when the system supports DBG _ MCU.

If the CPU enters debug mode, whether IWDG continues to count or enters Stop mode depends on the configuration of DBG _ IWDG _ stop in the DBG module.

24.3.5. Stop mode

There is an IWDG _ stop bit in the option byte in the flash information area, which can control whether IWDG continues to count normally or stops counting when the system enters the Stop mode. The default IWDG _ STOP is' 1 '.

24.4. IWDG registers

24.4.1. IWDG key register (IWDG_KR)

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
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	RES	-	Reserved
15:0	KEY[15:0]	W	0x00	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. Writing the key value 0x5555 to enable access to the IWDG_PR and IWDG_RLR registers; 0xCCCC: indicates that IWDG is started (not restricted by this command word if hardware watchdog is selected).

24.4.2. 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	-	-	Reserved

2:0	PR[2:0]	RW	0	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.
				000: divider /4;
				001: divider /8;
				010: divider /16;
				011: divider /32;
				100: divider /64;
				101: divider /128;
				110: divider /256;
				111: divider /256;

24.4.3. IWDG reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF

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]											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	RL[11:0]	RW	0	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. Registers can only be modified if IWDG _ SR.RVU = 0.

24.4.4. Status register (WWDG_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		RVU	PVU
														R	R

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
1	RVU	R	0	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the win-dow value is ongoing. 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 by hardware to indicate that an update of the pre-scaler value is ongoing. It is reset by hardware when the pre-scaler update operation is completed.

Note: Wait for IWDG _ PVU and IWDG _ SR.RVU to be 0 before updating IWDG _ PR and IWDG _ SR.RLR, 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 execute the following code.

25. Window watchdog (WWDG)

25.1. Introduction

System window watchdog (WWDG) is used to detect the occurrence of a software fault. Usually, this kind of software fault is caused by external interference or unpredicted logical conditions. This kind of fault causes the application program to terminate the normal operation sequence. Watchdog generates a reset signal when a programmed time period expires, unless the program refreshes the contents of the down counter before the T6 bit is cleared.

A reset will also occur if the value of the 7-bit down counter is flushed before the down counter has reached the window register value. This implies that the counter must be refreshed within a limited window.

The WWDG clock comes from the frequency division of the APB clock and has a configurable time window (this window is programmable to detect abnormal early or late application behavior). WWDG is most suitable for scenarios that require Watchdog to respond in a precise timing window.

25.2. WWDG main features

- Programmable free-running decrement counter
- Conditional reset
  - When the value of the decrement counter is less than 0x40, a reset is generated (if the watchdog is activated).
  - When the decrement counter is reloaded outside the window, a reset occurs (if the watchdog is activated).
- Early wakeup interrupt: triggered when the down counter is equal to 0x40 (if the function is enabled and watchdog is activated)

25.3. WWDG functional description

If WWDG is active (WDGA bit is set), and when the 7-bit down counter (T [6: 0] bit) decreases from 0x40 to 0x3F (T6 is cleared), a reset is caused. If the counter value is greater than the value of the memory window register when the software reloads the counter, a reset is generated.

The application must write to the WWDG _ CR register at routine intervals during normal operation to prevent reset. The above write operation must occur only if the counter value is less than the value of the window register and higher than 0x3F. The value of the WWDG _ CR register must be between 0xFF and 0xC0.

25.3.1. WWDG architecture block diagram

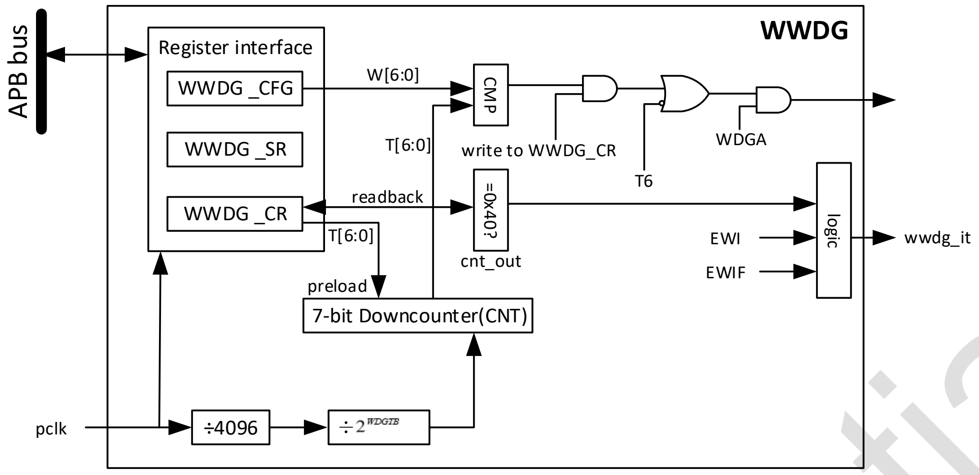


Figure 25-1 Window watchdog architecture

25.3.2. Start the watchdog

When the user selects "Software WWDG" for the WWDG _ SW bit in the option byte, the watch-dog is usually disabled after reset. Then by setting the WDGA bit of the WWDG _ CR register, the WWDG module is enabled and then cannot be disabled unless a reset occurs.

When WWDG _ SW selects "Hardware WWDG" in the user option byte, the module is usually en-abled after reset and cannot be disabled.

25.3.3. Control decrement counter

When the down counter is free-running, it counts down even if the watchdog is disabled. When watchdog is enabled, the T6 bit must be set to prevent an immediate reset.

The T [5: 0] bit contains the count value incremented before the WWDG generates a reset, and this data represents the time delay before the watchdog generates a reset.

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. When the decrement counter reaches 0x40, this interrupt is generated, and the corresponding interrupt service routine (ISR) can be used to load the counter to prevent WWDG from resetting. This interrupt can be cleared by writing '0' in the WWDG _ SR register.

25.3.4. Advanced watchdog interrupt function

Early wake-up interrupt (EWI) can be used for specific security operations, or in the case of data logging before reset. The EWI function can be enabled by configuring the WWDG _ CFR.EWI reg-ister. When the value of the decrement counter reaches 0x40, an EWI interrupt is generated, and the corresponding interrupt handler ISR can perform a specific operation before generating a re-set.

25.3.5. How to write a watchdog timeout program

The WWDG _ CR register is typically avoided by writing 1 to the T6 bit.

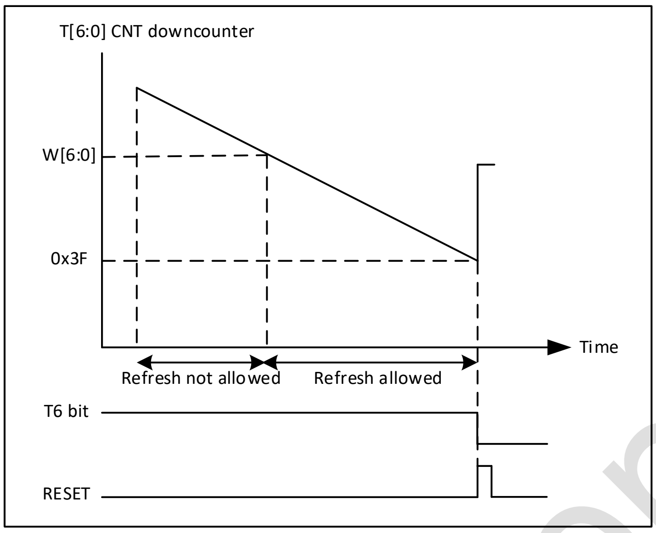


Figure 25-2 Window watchdog timing diagram

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)

25.4. WWDG registers

25.4.1. 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	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	WDGA	T [6: 0]						
								RS	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	WDGA	RS	0	WDGA: Activation bit. This bit is set '1' by software, but can only be cleared '0' by hardware after reset. When WDGA = 1, the watchdog can generate a reset. 0: Interrupt is inhibited 1: enable;
6:0	T [6: 0]	RW	32'h7F	7-bit counter (MSB to LSB). This register is used to store the count value of the watchdog. Every (4096x2WDGTB) PCLK cycle is subtracted by 1. When the count value changes from 40h to 3Fh (T [6] becomes 0), a watchdog reset occurs.

25.4.2. 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	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	EWI	WDGTB[1:0]		T [6: 0]						
						RS	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9	EWI	RS	0	Early wake-up interruption. At this position 1, when the counter value reaches 40h, an interrupt will be generated. This interrupt can only be cleared by the hard-ware after reset.

8：7	WDGTB[1:0]	RW	2’b0	— Time base (timer base).
				The time base of the prescaler is set as follows:
				00: CK counter clock (PCLK divided by 4096) di-vided by 1;
				01: CK counter clock (PCLK divided by 4096) di-vided by 2;
				10: CK counter clock (PCLK divided by 4096) di-vided by 4;
				11: CK counter clock (PCLK divided by 4096) di-vided by 8;

6：0	W [6: 0]	RW	7’h7F	7-bit window value. This register contains a window value for com-parison with a decrement counter.

25.4.3. Status register (WWDG _ SR)

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	Res	Res	Res	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31：1	Reserved	-	-	Reserved
0	EWIF	RC_W0	0	Early wake-up interrupt flag. When the counter value reaches 40h, this bit is set to 1 by the hardware. The software clears it by writing 0, and writing 1 is invalid. This bit is also set to 1 when the interrupt is not enabled.

26. Real-time clock (RTC)

26.1. Introduction

The real time clock (real time clock) is an independent timer. It has a set of continuous counting counters, which can provide a clock calendar function under the corresponding software configuration. Modifying the value of the counter can reset the current time and date of the system.

26.2. Main features

- Programmable pre-division factor: division factor up to 220
- 32-bit programmable counter for measurements over longer periods
- 2 separate clocks: PCLK1 and RTC clock for APB1 interface (the frequency of the RTC clock must be less than more than a quarter of the PCLK1 clock frequency)
- You can choose the following three RTC clock sources:
  - HSE clock divided by 128
  - LSI oscillator clock
  - LSE oscillator clock
- 3 specialized maskable interrupts: ─ Alarm interrupt, used to generate a software programmable alarm interrupt ─ Second interrupt, used to generate a programmable periodic interrupt signal (up to 1 second) ─ Overflow interrupt, indicating that the internal programmable counter overflows and turns back to 0

26.3. RTC functional description

26.3.1. Overview

The RTC consists of two main parts (see figure below). The first part (APB interface) is used to connect to the APB bus. The other part (the RTC core) consists of a set of programmable counters divided into two main modules:

The first module is the prescalation module of the RTC, which is programmable to produce an RTC time reference TR _ CLK of up to 1 second. The RTC prescaler module includes a 20-bit programmable frequency 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 module 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.

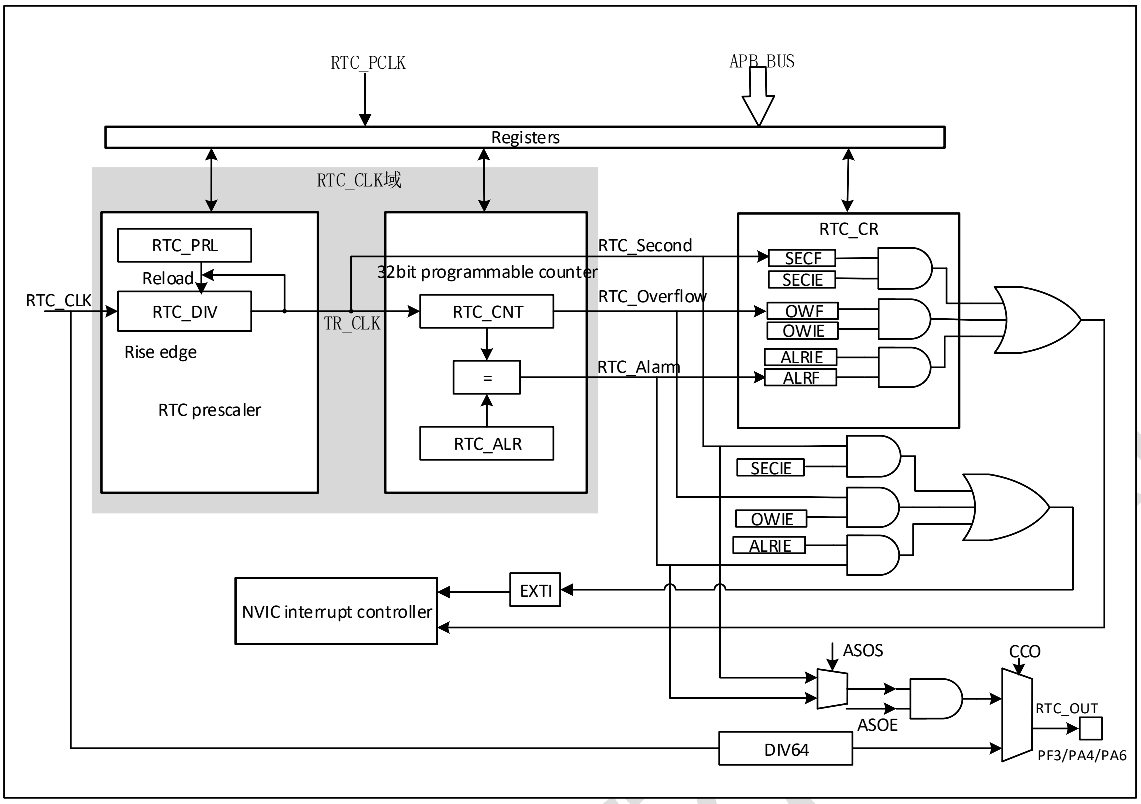


Figure 26-1 RTC block diagram

The DBP bit of the PWR _ CR1 register is used to control write protection inhibition of the RTC register. By default, DBP = 0, and write access to the RTC register cannot be performed. Only after the software sets DBP can the write operation to the RTC register be realized.

26.3.2. Reset RTC register

All PCLK domain register resets of the RTC come from the system reset and the software reset of the RTC (RCC _ APBRSTR1 [10]). All RTC _ CLK domain register resets for RTC come from power-on resets (POR/PDR/BOR) and software resets for RTC (RCC _ BDCR [16]).
Note that the clock source enable signal and the clock source selection control signal of the RTC module in the RCC _ BDCR register are reset by the power-on reset (POR/PDR/BOR) and the RTC software reset signal (RCC _ BDCR [16]).

26.3.3. Read RTC register

The RTC core is completely independent of the APB clock. The software accesses the values of the RTC _ DIV, RTC _ CNT, and RTC _ ALR registers through the APB interface, but the updates of the related readable registers first pass through the rising edge of the RTC clock internally, and then are synchronized again using the APB clock. The same is true for the flag register of the RTC.
If the APB interface has been previously disabled, a read operation is performed immediately after the APB interface is enabled (and in this case before the first register update), and the read operation may be corrupted (typically read returns 0). This happens in the following situations:

1. System reset or power-on reset
2. Exiting Stop mode

In all the above-mentioned cases, the RTC core circuit remains running all the time, while the APB interface is turned off (reset or no clock).

Accordingly, when reading the RTC register, the software must wait for the RSF bit (register synchronization flag) of the RTC _ CRL register to be set by the hardware after the RTC APB interface has been closed.

26.3.4. Configure the RTC register

You must enter configuration mode by setting the CNF bit of the RTC _ CRL register before writing to the RTC _ PRL, RTC _ CNT, RTC _ ALR, BKP _ RTCCR registers.

In addition, the write operation to any register of RTC must be performed after the previous write operation has been completed. It is possible to determine whether the RTC register is being updated by querying the RTOFF status bit in the RTC _ CR register. The RTC register may be written only if the RTOFF status bit is' 1 '.

Configuration process:

(1) Querying the RTOFF bit until the value of RTOFF becomes' 1 '; (indicates that the previous configuration has been completed)

(2) Set the CNF value to 1 to enter the configuration mode; (At this time, the RTOFF bit is still 1, ensuring that RTOFF = 1 when the CPU writes to the register)

(3) 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)

(4) 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)

(5) The RTOFF is queried until the RTOFF bit becomes' 1 'to confirm that the write operation has completed. (The buffer register is written to the RTC _ CLK field after setting RTOFF)

Note: The write operation can only be performed when the CNF flag bit is cleared, and this process requires at least 3 RTC _ CLK cycles. (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))

26.3.5. Setting of RTC flags

The hardware sets the RTC Second Flag (SECF) before changing the RTC counter for each RTC CLOCK clock cycle. The RTC Overflow Flag (OWF) is set at the last RTC clock cycle before the counter reaches 0x0000.

The RTC _ Alarm and the RTC alarm flag (ALRF) are set for the RTC clock cycle before the value of the counter reaches the value of the alarm register plus 1 (RTC _ ALR+1). Writes to RTC alarms must be synchronized with the RTC second flag using one of the following procedures:

(1) Use the RTC alarm interrupt and modify the RTC alarm and/or the RTC counter in the interrupt handler.

(2) Wait for the SECF bit in the RTC control register to be set before changing the RTC alarm and/or the RTC counter.

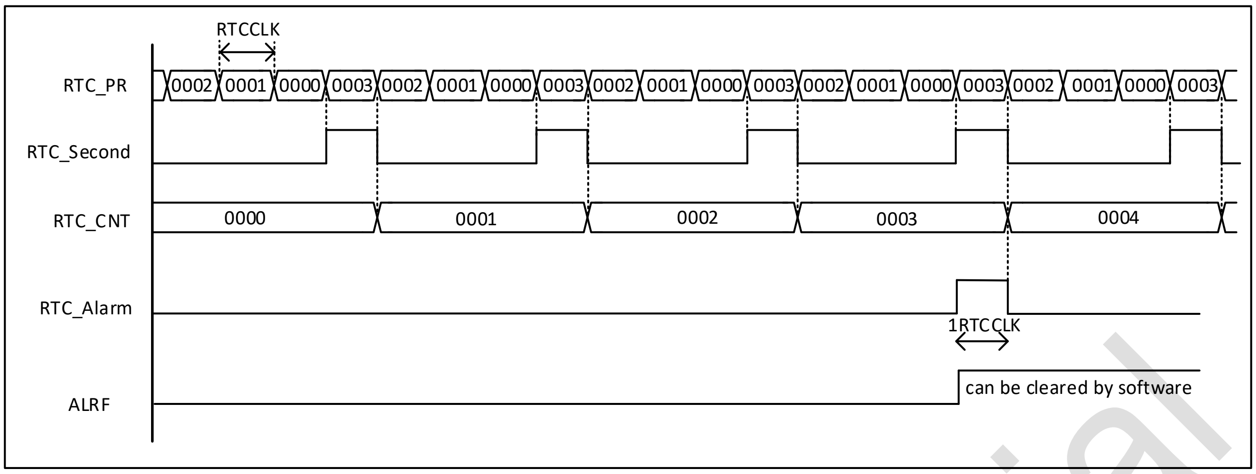


Figure 26-2 RTC second and ALARM waveform diagram example, PR = 0003, ALARM = 00004

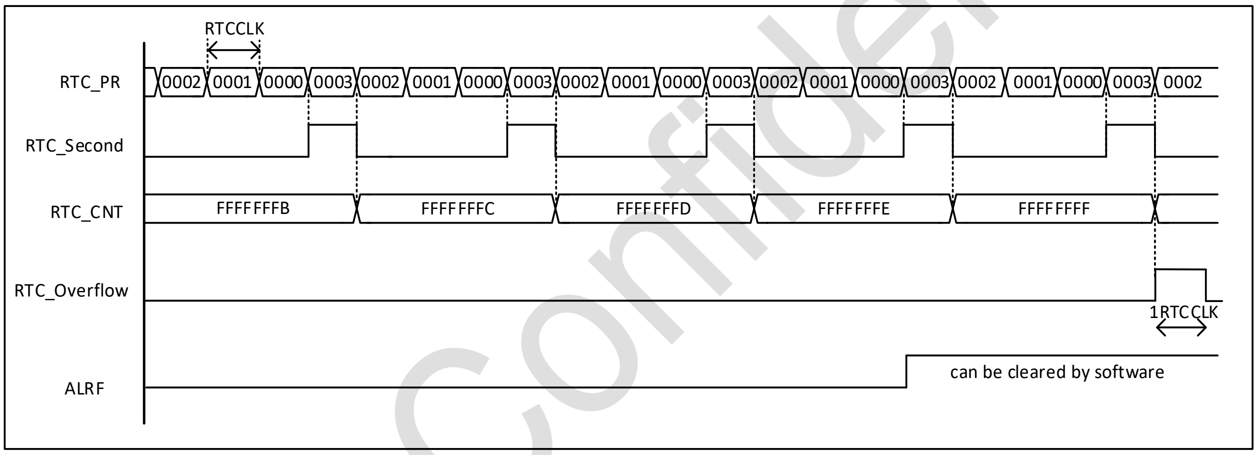


Figure 26-3 Example of RTC overflow waveform diagram, PR = 0003

26.3.6. RTC calibration

For measurement purposes, the 64 division of the RTC clock can be output to the IO pin (PF5).

This function is achieved by setting the CCO bit (RTCCR register).

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

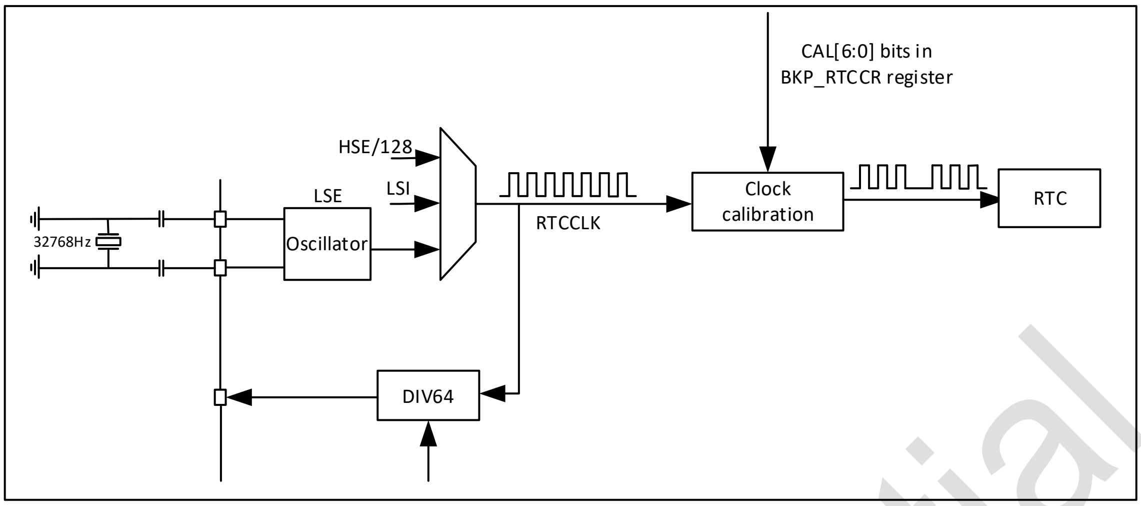


Figure 26-4 RTC calibration chart

26.4. RTC registers

26.4.1. RTC control register (RTC _ CRH)

Address offset: 0x00
Reset value: 0x0000

This register is reset by a system reset.

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	ALR IE	SEC IE
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2	OWIE	RW	0	Overflow interrupt allowance bit 0: Overflow interrupt disabled 1: Overflow interrupt enabled
1	ALRIE	RW	0	Alarm clock interrupt allowance bit 0: Alarm clock interrupt disabled 1: Alarm interruption enabled
0	SECIE	RW	0	Second interrupt allowance bit 0: Second interrupt disabled 1: Second interrupt enabled

These bits are used to mask interrupt requests. Note: After reset, all interrupts are not enabled, so after initialization, it is possible to write the RTC register to ensure that there are no pending interrupt requests. However, when the peripheral is completing the previous write operation (RTOFF = 0), it cannot write the RTC _ CRH register.

This control register controls the function of the RTC. Some bits must use a dedicated configuration process to be written.

26.4.2. RTC control register (RTC _ CRL)

Address offset: 0x04

Reset value: 0x0020

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	-	-	Reserved
5	RTOFF	R	1	RTC operation OFF, this bit is read-only. The RTC module utilizes this bit to indicate the status of the last operation performed on its registers (indicating whether the operation is completed or not). If this bit is' 0 ', it means that no write operation can be performed to any RTC register. 0: The last write to the RTC register is still in progress 1: The last write operation to RTC register has been completed
4	CNF	RW	0	Configuration flag This bit must be set '1' by the software to enter configuration mode to allow new values to be written to the RTC _ CNT, RTC _ ALR or RTC _ PRL registers. The write operation will only be performed when this bit is set to '1' and cleared again by '0' by the software. See "Configurating RTC registers" 0: Exit configuration mode (start updating RTC register) 1: Enter configuration mode
3	RSF	RC_W0	0	Registers synchronized flag When the RTC _ CNT register and the RTC _ DIV register are updated, the hardware sets the bit '1' and the software clears the bit. 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: Register has not been synchronized 1: Registers have been synchronized
2	OWF	RC_W0	0	Overflow flag This bit is set '1' by the hardware when a 32-bit programmable counter overflows. An interrupt is generated if OWIE = 1 in the RTC _ CRH register. This bit can only be cleared by the software '0', and writing '1' is invalid. 0: no spill; 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. This bit can only be cleared by the software '0', and writing '1' is invalid. 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. An interrupt is generated if SECIE = 1 in the RTC _ CRH register. This bit can only be cleared by software, write '1' is

Bit	Name	R/W	Reset Value	Function
				invalid. 0: Second flag condition does not hold 1: Second flag condition holds

The function of the RTC is controlled by the control register. When the peripheral is continuing the last write operation (RTOFF = 0), the RTC _ CR register cannot be written.

26.4.3. RTC prescaler load register high (RTC _ PRLH)

The PRL register holds the count value of the RTC prescaler periodicity. This register is write-protected by the RTOFF bit of the RTC _ CR register. Only when RTOFF = 1 is allowed to write the CPU (write to the buffer register).

Address offset: 0x08

Write only

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**
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	-	-	Reserved
3：0	PRL[19:16]	W	0	RTC prescaler reload value high (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: 0 value is not recommended, otherwise RTC interrupt and flag bits cannot be generated correctly

26.4.4. RTC reload register low bit (RTC _ PRLL)

Address offset: 0x0C

Write only

Reset value: 0x8000

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	-	-	Reserved
15：0	PRL	W	0x8000	RTC prescaler reload value high (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: 0 value is not recommended, otherwise RTC interrupt and flag bits cannot be generated correctly

26.4.5. RTC prescaler divider register high (RTC _ DIVH)

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 read-only, 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

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	RTC _ DIV [19:16]			
												R	R	R	R

Bit	Name	R/W	Reset Value	Function
31：4	Reserved	-	-	Reserved
3：0	RTC _ DIV [19:16]	R	0	RTC clock divider remainder high bit.

26.4.6. RTC prescaler divider register low (RTC _ DIVL)

Address offset: 0x14

Reset value: 0x8000

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	-	-	Reserved
15：0	DIV[15:0]	R	0x8000	RTC clock divider

26.4.7. RTC counter register high (RTC _ CNTH)

The RTC module has a 32-bit programmable counter, which is accessed through two 16-bit registers. The count is based on the TR _ CLK time reference generated by the prescaler.

The RTC _ CNT register holds the count value of this counter. The register is write-protected and can only be written if 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 frequency division. When a read operation occurs, the current value of the counter (system time) is returned.

Address offset: 0x18

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**

RTC _ 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	-	-	Reserved
15: 0	RTC _ CNT [31: 16]	RW	0x0000	The upper 16 bits of the RTC core counter When the RTC _ CNTH register is read, the upper 16 bits of the current value of the RTC counter register are returned. Write operations to this register can only be performed when you enter configuration mode.

26.4.8. RTC counter register low (RTC _ CNTL)

Address offset: 0x1C

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**
RTC _ 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	-	-	Reserved
15: 0	RTC _ CNT [15: 0]	RW	0x0000	RTC kernel counter 16 bits lower When the RTC _ CNTL register is read, the lower 16 bits of the current value of the RTC counter register are returned. Write operations to this register can only be performed when you enter configuration mode.

26.4.9. RTC alarm register high (RTC _ ALRH)

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 write access is only allowed if RTOFF = 1.

Address offset: 0x20

Reset value: 0xFFFF

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**
RTC _ ALR [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	-	-	Reserved
15: 0	ALR[31:16]	RW	0xFFFF	RTC alarm 16 bits higher The software can write the upper 16 bits of the Alarm time. Writing to this register must enter configuration mode.

26.4.10. RTC alarm register high (RTC _ ALRL)

Address offset: 0x24

Reset value: 0xFFFF

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**
RTC _ 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	-	-	Reserved
15：0	ALR[15:0]	RW	0xFFFF	RTC alarm lower 16 bits The software can write the lower 16 bits of the Alarm time. Writing to this register must enter configuration mode.

26.4.11. RTC clock calibration and output configuration register (BKP _ RTCCR)

Address offset: 0x2C

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**
Res	Res	Res	Res	Res	Res	ASOS	ASOE	CCO	CAL[6:0]						
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9	ASOS	RW	0	Second/Alarm Pulse Output Select Bit When that ASOE bit is set, the ASOS bit can be used to select whether the output on the Pin is an RTC second pulse signal or an Alarm pulse signal 0: RTC Alarm pulse Signal 1: RTC second pulse Signal
8	ASOE	RW	0	Second/alarm pulse output enable bit When this bit is set, the ASOS bit determines whether an RTC second pulse signal or an Alarm pulse signal is output on the pin.
7	CCO	RW	0	Calibration clock output 0: No action 1: When this bit is set, the 64-division of RTC clock is output on the pin. Note: ASOE priority is higher when ASOE and CCO are set at the same time.
6：0	CAL[6:0]	RW	0	Calibration value This value shows the negligible number of clock pulses per 220 clocks, which allows the RTC to calibrate to slow down the clock in steps of 1000000/220 PPM. The RTC clock can be slowed down from 0 to 121 PPM.

Note: PA4, PA6 and PF3 can be configured to output second interrupts, alarm interrupts or 64-divided clocks of RTC clocks. Depending on the configuration of ASOE, ASOS and CCO, the output conditions of PA4, PA6 and PF3 are as follows:

ASOE	ASOS	CCO	RTC_OUT
0	x	0	-
0	x	1	RTC _ CLK/64
1	0	x	RTC alarm interrupt
1	1	x	RTC second interrupt

27. Inter-integrated circuit (I^{2}C) interface

27.1. Introduction

The I^{2}C (inter-integrated circuit) bus interface handles communications between the microcontroller and the serial I^{2}C 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+).

For specific needs, DMA can be used to reduce the burden on the CPU.

27.2. Main features

- Slave and master modes
- Multimaster capability: can be master or slave
- Support different communication speeds
  - Standard mode (Sm): up to 100 kHz
  - Fast Mode (Fm): up to 400 kHz
  - Fast Boost Mode (Fm+): 1MHz
- As master
  - Clock generation
  - Generation of start and stop
- As Slave
  - Programmable I2C address detection
  - Discovery of the stop bit
- 7-bit addressing mode
- General call
- Status flag
  - Transmit/receive mode flags
  - Byte transfer complete flag
  - I2C busy flag bit
- Error flag
  - Master arbitration loss
  - ACK failure after address/data transfer
  - Start/Stop error
  - Overrun/Underrun (clock stretching function disable)
- Optional clock stretching
- Software reset
- Analog noise filter function
- Configurable PEC (packet error checking) generation and verification
  - The PEC value may be transmitted at the last byte in Tx mode
  - Receive the last byte and do PEC error check

27.3. I^{2}C functional description

27.3.1. I^{2}C block diagram

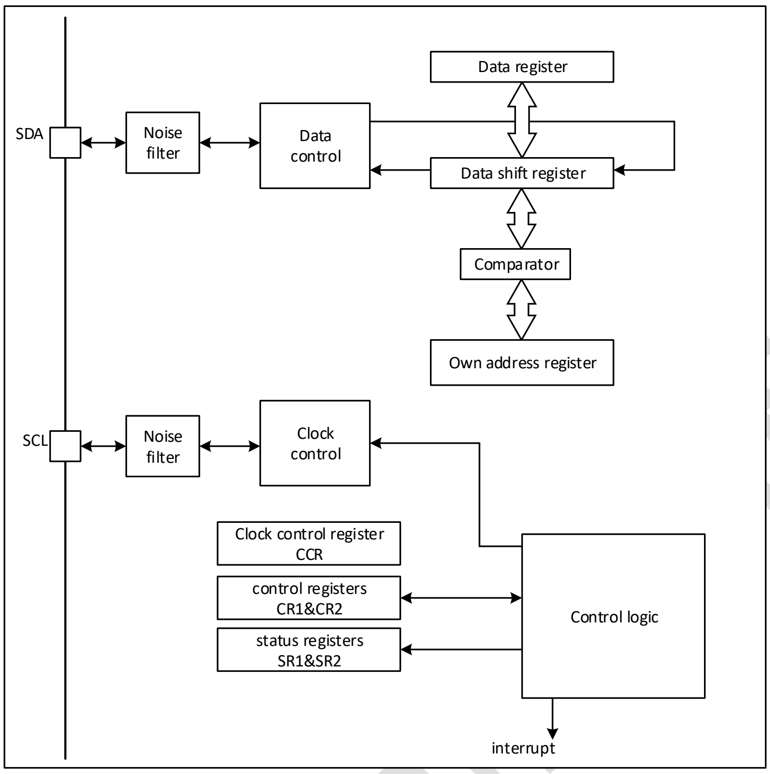


Figure 27-1 I^{2}C block diagram

27.3.2. Mode selection

The interface can operate in one of the four following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

By default, it operates in slave mode. The interface automatically switches from slave to master, after it generates a START condition and from master to slave, if an arbitration loss or a Stop generation occurs, allowing multimaster capability.

27.3.2.1. Communication flow

In Master mode, the I^{2}C 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 bits) and the general call address. The software can control enabling or disabling the recognition of the general call address.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the start condition contain the address. Addresses are only sent in master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledged bit to the transmitter. Refer to the following figure.Refer to the figure below.

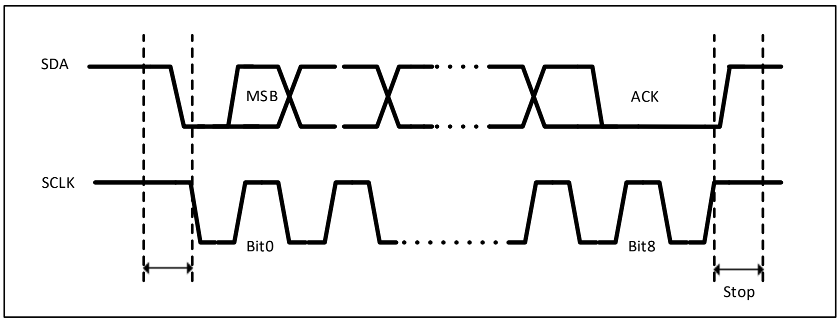


Figure 27-2 I²C bus protocol

The software may enable or disable the ACK bit. The software can also select the I2C interface address (dual addressing 7-bit/10-bit and/or general call address)

27.3.3. I²C initialization

27.3.3.1. Enable/turn off I2C module

The I²C peripheral clock must be configured and enabled by the I2C_EN bit in the RCC_APBENR1 register. Then the I²C can be enabled by setting the PE bit in the I2C_CR1 register.

27.3.3.2. I2C timings

In order to meet the accurate data hold and setup time in master and slave modes, I2C timing needs to be set. This is achieved by writing to the I2C _ CCR and I2C _ TRISE registers.

27.3.4. I²C slave mode

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

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).

Header segment or address mismatch:

The interface ignores it and waits for another Start condition.

Address match:

The interface generates in sequence:

● 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

The TRA bit indicates whether the slave is in Receiver or Transmitter mode.

27.3.4.1. Slave transmitter

Following the address reception and after clearing ADDR, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave stretches SCL low until ADDR is cleared and DR filled with the data to be sent.

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. The interface waits until BTF is cleared by a read to I2C_SR1 followed by a write to the I2C_DR register, stretching SCL low.

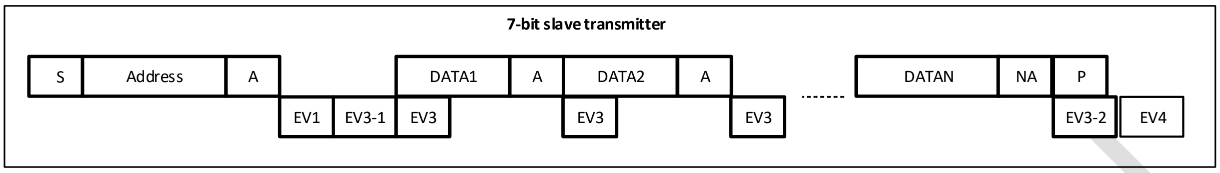


Figure 27-3 Transfer sequence diagram for slave transmitter

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, NA= Non-acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by reading SR2

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; AF is cleared by writing ‘0’ in AF bit of SR1 register.

27.3.4.2. Slave receiver

Following the address reception and after clearing ADDR, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- An acknowledge pulse if the ACK bit is set
- The RxNE bit is set by hardware. An interrupt is generated if the ITEVTEN and ITBUFEN bit is set.

If RxNE is set and the data in the DR register is not read before the end of the next data reception, the BTF bit is set, and the interface waits until BTF is cleared by a read fromI2C_SR1 followed by a read from the I2C_DR register, stretching SCL low. (See figure below).

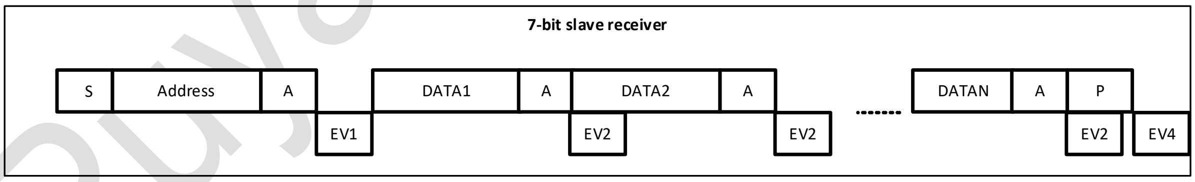


Figure 27-4 Transfer sequence diagram for slave receiver

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by 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.

Note:

1) The EV1 event stretches SCL low until the end of the corresponding software sequence.

2) The EV2 software sequence must be completed before the end of the current byte transfer.

3) After checking the SR1 register content, the user should perform the complete clearing sequence for each flag found set. Thus, for ADDR and STOPF flags, the following sequence is required inside the I^{2}C 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.

27.3.4.3. 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 STOPF bit is set and generates an interrupt if the ITEVFEN bit is set.

The STOPF bit is cleared by a read of the SR1 register followed by a write to the CR1 register. (See EV4 in figure above)

27.3.5. I²C master mode

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 register
- 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:

- In standard mode: 2 MHz
- In fast mode: 4 MHz

27.3.5.1. The master generates clock

The CCR bits are used to generate the high and low level of the SCL clock, starting from the generation of the rising edge. 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 (VIH data detection of the SCL), 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 used by the feedback loop is programmed in the TRISE bits, so that the SCL frequency remains stable whatever the SCL rising time.

27.3.5.2. 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: In master mode, setting the START bit causes the interface to generate a ReStart condition at the end of the current byte transfer.

Once the Start condition is issued:

- The SB bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.

The master reads the SR1 register, and then writes the slave address to the DR register. Transfer sequence EV5

27.3.5.3. Slave address transmission

Then the slave address is sent to the SDA line via the internal shift register.

- In 7-bit addressing mode, one address byte is sent.

As soon as the address byte is sent,

- The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.

The Master then reads the SR1 register and then reads the SR2 register.

The master can decide to enter Transmitter or Receiver mode depending on the LSB of the slave address sent.

- In 7-bit addressing mode,
  - To enter Transmitter mode, a master sends the slave address with LSB reset.
  - To enter Receiver mode, a master sends the slave address with LSB set.

The TRA bit indicates whether the master is in Receiver or Transmitter mode.

27.3.5.4. Master transmitter

Following the address transmission and after clearing ADDR, the master sends bytes from the DR register to the SDA line via the internal shift register.

The master waits until the first data byte is written into DR register (see EV8_1 in figure below).

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 to the DR register, the STOP bit is set by software to generate a Stop condition (see EV8_2 in figure below). The interface automatically goes back to slave mode (MSL bit cleared).

Note: When the TxE or BTF position is bit, the stop condition shall be scheduled for the occurrence of the EV8 _ 2 event.

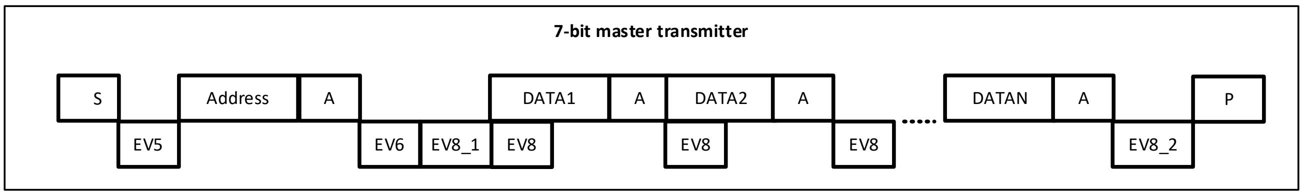


Figure 27-5 Transfer sequence diagram for master transmitter

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register with Address.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV8_1: TxE=1, shift register empty, data register empty, write Data1 in DR.

EV8: TxE=1, shift register not empty, data register empty, cleared by writing DR

EV8_2: TxE=1, BTF = 1, Program Stop request. TxE and BTF are cleared by hardware by the Stop condition

Note:

1) The EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

2) The EV8 software sequence must complete before the end of the current byte transfer. In case EV8 software sequence can not be managed before the current byte end of transfer, it is recommended to use BTF instead of TXE with the drawback of slowing the communication.

27.3.5.5. Master receiver

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 each byte the interface generates in sequence:

● An acknowledge pulse if the ACK bit is set

● 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: This method is for the case when the I^{2}C is used with interrupts that have the highest priority in the application.

Master sends a NACK after receiving the last byte from Slave. After receiving this NACK, the slave releases the control of the SCL and SDA lines. The Master can then 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) In case a single byte has to be received, the Acknowledge disable and the Stop condition generation are made just after EV6 (in EV6_1, just after ADDR is cleared).

4) After the Stop condition generation, the interface goes automatically back to slave mode (MSL bit cleared).

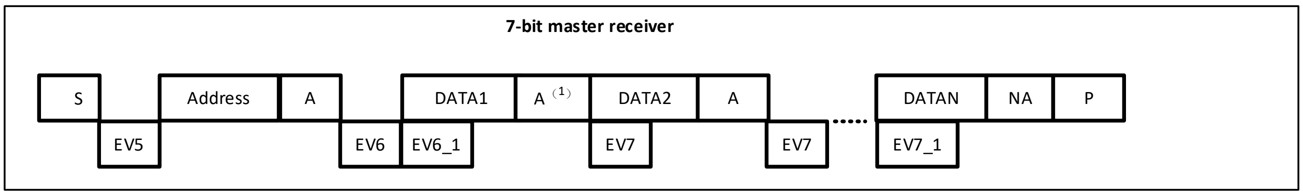


Figure 27-6 Method 1: transfer sequence diagram for master receiver

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if 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

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 case EV7 software sequence can not be managed before the current byte end of transfer. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication speed.
4) The EV6_1 or EV7_1 software sequence must complete before the ACK pulse of the current byte transfer.

Method 2: This method is for the case when the I^{2}C is used with interrupts that do not have the highest priority in the application or when the I^{2}C is used with polling.

With this method, DataN_2 is not read, so that after DataN_1, the communication is 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 that, just after reading DataN_2, set the STOP/ START bit and read DataN_1. After RxNE is set, read DataN.

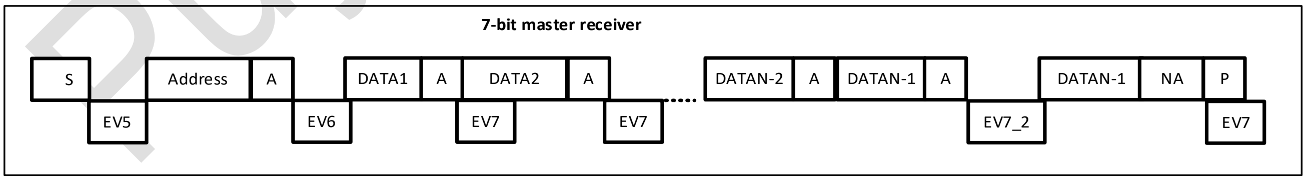


Figure 27-7 Method 2: transfer sequence diagram for master receiver when N>2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if 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

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.

Note:

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 case EV7 software sequence can not be managed before the current byte end of transfer. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication.

When 3 bytes remain to be read:

- RxNE = 1 = > Nothing (DataN-2 not read).
- DataN-1 received
- BTF = 1 because both shift and data registers are full: DataN-2 in DR and DataN-1 in the shift register => SCL tied low: no other data will be received on the bus.
- Clear ACK bit
- Read DataN-2 in DR => This will launch the DataN reception in the shift register
- DataN reception complete (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 two bytes to be 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

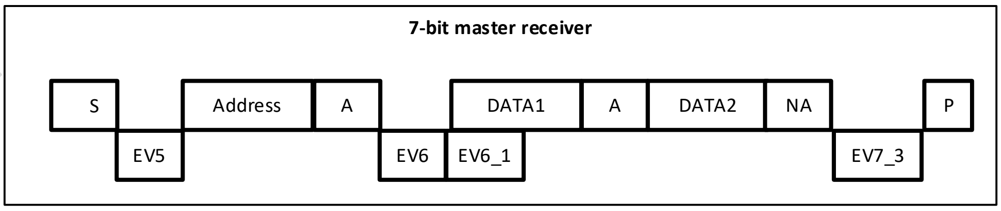


Figure 27-8 Method 2: transfer sequence diagram for master receiver when N=2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if 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.

Note:

1) The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
2) The EV6_1 software sequence must complete before the ACK pulse of the current byte transfer.

Case of a single byte to be received:

- In the ADDR event, clear the ACK bit
- Clear ADDR
- Program STOP/START bit
- Read the data after the RxNE flag is set.

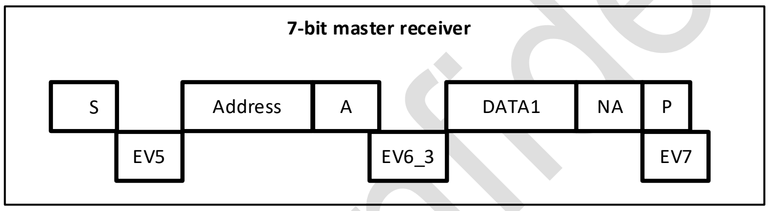


Figure 27-9 Method 2: transfer sequence diagram for master receiver when N=1

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if 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

Note:

The EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

27.3.6. Error conditions

27.3.6.1. Bus error (BERR)

This error occurs when the I^{2}C interface detects an external Stop or Start condition during an address or a data transfer. In this case:

● The BERR bit is set, and an interrupt is generated if the ITERREN bit is set;

● In Slave mode: data are discarded, and the lines are released by hardware:

─ If it is a wrong Start condition, slave thinks it is a Restart and waits for the address or stop condition

─ In case of a misplaced Stop, the slave behaves like for a Stop condition and the lines are released by hardware

● In Master mode: the lines are not released and the state of the current transmission is not affected. It is up to the software to abort or not the current transmission.

27.3.6.2. Acknowledge failure (AF)

This error occurs when the interface detects a non-acknowledge bit. In this case:

● The AF bit is set, and an interrupt is generated if the ITERREN bit is set

● A transmitter which receives a NACK must reset the communication:

─ If it is in slave mode, the hardware releases the bus.

─ If it is in master mode, the software must generate a stop condition or repeated start.

27.3.6.3. Arbitration Loss (ARLO)

This error occurs when the I^{2}C interface detects an arbitration lost condition. In this case:

● The ARLO bit is set by hardware (and an interrupt is generated if the ITERREN bit is set)

● The I^{2}C Interface goes automatically back to slave mode (the MSL bit is cleared). When the I^{2}C 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.

● Lines are released by hardware

27.3.6.4. Overrun/underrun error (OVR)

An overrun error can occur in slave mode when clock stretching is disabled and the I^{2}C 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 byte is lost.

● In case of Overrun error, software should clear the RxNE bit and the transmitter should re-transmit the last received byte.

● Underrun error can occur in slave mode when clock stretching is disabled and the I^{2}C 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 same byte in the DR register will be sent again

● The user should make sure that data received on the receiver side during an underrun error are discarded. The next bytes are written within the clock low time specified in the I^{2}C 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.

27.3.7. SDA/SCL line control

● If clock stretching is enabled:

- Transmitter mode: If TxE=1 and BTF=1: the interface holds the clock line low before transmission to wait for the microcontroller to read I2C_SR1 and then write the byte in the Data register (both DR and shift register are empty).

- Receiver mode: If RxNE=1 and BTF=1: the interface holds the clock line low after reception to wait for the microcontroller to read SR1 and then read the byte in the Data register (both DR and shift register are full).

● If clock extension is disabled in slave mode:

- Overrun Error in case of RxNE=1 and no read of DR has been done before the next byte is received. The last received byte is lost.

- Underrun Error in case TxE=1 and no write into DR register has been done before the next byte must be transmitted. The same byte will be sent again.

- Write collision not managed.

27.3.8. DMA Request

The DMA request (when enabled) is used only for data transmission. A DMA request is generated when the data register becomes empty at the time of sending, or when the data register becomes full at the time of receiving. The DMA must be initialized and enabled before the current byte transfer ends. The DMAEN bit (in the I2C _ CR2 register) must be enabled before the ADDR event occurs.

In master or slave mode, when the clock extension function is enabled, the DMAEN bit may be set during the ADDR event before clearing the ADDR. The DMA request must respond before the current byte transfer is complete. When the DMA transmission data length reaches the value set by DMA, the DMA controller sends an EOT (End of transfer) to I2C and generates a transfer complete interrupt (if the interrupt enable bit is valid):

● Master transmitter: In the EOT interrupt service routine, the DMA request needs to be prohibited, and then the stop condition needs to be set after waiting for the BTF event.

• Master receiver: When the number of data to be received is greater than or equal to 2, the DMA controller transmits a hardware signal EOT _ 1, which corresponds to the DMA transmission (number of bytes-1). If the LAST bit is set in the I2C _ CR2 register, the hardware will automatically send a NACK at the next byte after sending EOT _ 1. When the interrupt allows, the user can generate a stop condition in the interrupt service routine when the DMA transmission is completed.

27.3.8.1. DMA transmission

The DMA mode can be enabled by setting the DMAEN bit in the I2C _ CR2 register. As long as the TxE bit is set, data will be loaded by the DMA from the preset memory area into the I2C _ DR register. To assign a DMA channel to I2C, you must perform the following steps (x is the channel number):

1. Set the I2C_DR register address in the DMA_CPARx register. Data will be transferred from memory to this address after each TxE event.
2. Set the memory address in the DMA _ CMARx register. Data is transferred from this memory area to I2C _ DR after each TxE event.
3. Set the required number of transfer bytes in the DMA _ CNDTRx register. This value is decremented after each TxE event.
4. Use the PL [0: 1] bit in the DMA _ CCRx register to configure the channel priority.

5. Set the DIR bit in the DMA _ CCRx register, and according to the application requirements, it can be configured to issue an interrupt request when the entire transmission is half or complete.

6. Activate the channel by setting the EN bit on the DMA _ CCTx register.

When the number of data transmissions set in the DMA controller has been completed, the DMA controller sends an EOT/EOT _ 1 signal of end of transmission to the I2C interface. When interrupts allow, a DMA interrupt will be generated.

Note: Do not set the ITBUFEN bit of the I2C _ CR2 register when sending using DMA.

27.3.8.2. DMA reception

The DMA receive mode can be activated by setting the DMAEN bit in the I2C _ CR2 register. Each time a data byte is received, the data of the I2C _ DR register will be transferred by the DMA to the set memory area (refer to DMA description). To set up a DMA channel for I2C reception, you must perform the following steps (x is the channel number):

1. Set the address of the I2C _ DR register in the DMA _ CPARx register. Data will be transferred from this address to the storage area after each RxNE event.

2. Set the memory area address in the DMA _ CMARx register. Data will be transferred from the I2C _ DR register to this memory area after each RxNE event.

3. Set the required number of transfer bytes in the DMA _ CNDTRx register. This value is decremented after each RxNE event.

4. Configure the channel priority with PL [0: 1] in the DMA _ CCRx register.

5. Clear the DIR bit in the DMA _ CCRx register. According to application requirements, it can be set to issue an interrupt request when half or all data transmission is completed.

6. Set the EN bit in the DMA _ CCRx register to activate the channel.

When the number of data transmissions set in the DMA controller has been completed, the DMA controller sends an EOT/EOT _ 1 signal of end of transmission to the I2C interface. When interrupts allow, a DMA interrupt will be generated.

Note: Do not set the ITBUFEN bit of the I2C _ CR2 register when using DMA for reception.

27.3.9. Packet error check

A packet error check (PEC) calculator is used to improve the reliability of communication. This calculator uses the following CRC-8 polynomial to calculate each bit of serial data:

C (x) = x8 + x2 + x + 1

• PEC calculation is activated by the 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 transfer bit of the I2C _ CR1 register is set at the last TxE event, and the PEC will be transmitted after the last byte.

On reception: Set the PEC bit of the I2C _ CR1 register after the last RxNE event, and if the next received byte is not equal to the internally calculated PEC, the receiver sends a NACK. If it is the primary receiver, NACK will be sent after PEC regardless of the result of proofreading. 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:

– On transmission: When the I2C interface receives the EOT signal from the DMA controller, it automatically sends the PEC after the last byte.

-On reception: When the I2C interface receives an EOT _ 1 signal 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.

• In order to allow intermediate PEC transmissions, there is a control bit (LAST bit) in the I2C _ CR2 register for determining 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.

27.4. I^{2}C interrupts

Table 27-1 I^{2}C interrupt requests

Interrupt event	Event flags	Enable control bit
Start bit sent (Master)	SB	ITEVTEN
Address sent (Master) or Address matched (Slave)	ADDR	
ADD10	10-bit header sent (primary)	
Stop received (Slave)	STOPF	
Data byte transfer complete	BTF	
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Send buffer empty	TxE	
Bus error	BERR	ITERREN
Arbitration loss (Master)	ARLO	
Acknowledge failure	AF	
Overrun/Underrun	OVR	
PEC Error	PECERR	
Timeout/Tlow error	TIMEOUT	
SMBus alert	SMBALERT	

27.5. I2C registers

Registers can be accessed half-word or word.

27.5.1. I^{2}C control register 1 (I2C_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res.	Res.	Res.	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMB-TYPE	Res.	SMBUS	PE
RW				RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	SWRST	RW	0	Software reset When set, the I^{2}C is under reset state. Before resetting this bit, make sure the I^{2}C lines are released and the bus is free. 0: I^{2}C Peripheral not under reset 1: I^{2}C Peripheral under reset state Note: This bit can be used to reinitialize I2C in error or 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.

Bit	Name	R/W	Reset Value	Function
14:12	Reserved	-	-	Reserved
11	POS	RW	0	Ack/PEC position (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 the byte in the current shift register is 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 received in the shift register is the 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 stretch event after the POS bit has been configured.
10	ACK	RW	0	Acknowledge enable This bit is set and cleared by software and cleared by hardware when PE=0. 0: No acknowledge returned 1: Acknowledge returned after a byte is received. (matched address 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	Prohibit clock extension (Slave). 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: Prohibit PEC calculation 1: Start PEC calculation
4	ENARP	RW	0	ARP enable 0: ARP disabled; 1: Enable ARP; If SMBTYPE = 0, use the default address of the SMBus device; If SMBTYPE = 1, the primary address of SMBus is used. Note: This register is fixed to 0 if the SMBus function is not supported.
3	SMBTYPE	RW	0	SMBus type. 0: SMBus device 1: SMBus host; Note: This register is fixed to 0 if the SMBus function is not supported.
2	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
1	SMBUS	RW	0	SMBus mode. 0: I2C mode; 1: SMBus mode Note: This register is fixed to 0 if the SMBus function is not supported.
0	PE	RW	0	I2C module enabled. 0: Disabled 1: I2C enabled According to the configuration of SMBus bits, the corresponding I/O port needs to be configured as a multi-plexing function. 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 communica-tion. In master mode, this bit must not be reset before the end of the communication.

Note: When the STOP or START bit is set, the software must not perform any write access to I2C_CR1 before this bit is cleared by hardware. Otherwise, there is a risk of setting a second STOP or START request.

27.5.2. I^{2}C control register 2 (I2C_CR2)

Address offset: 0x04

Reset value: 0x0000

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
31:16	Reserved	-	-	Reserved
15:13	Reserved	-	-	Reserved
12	LAST	RW	0	DMA last transmission. 0: EOT of the next DMA is not the last transmis-sion 1: EOT of the next DMA is the last transmission Note: This bit is used in primary receive mode, so that a NACK can be generated when data is last received.
11	DMAEN	RW	0	DMA request enabled. 0: Prohibit DMA request; 1: When TxE = 1 or RxNE = 1, DMA request is allowed.
10	ITBUFEN	RW	0	Buffer interrupt enable 0: TxE = 1 or RxNE = 1 does not generate any interrupt. 1: Generate an event interrupt when TxE = 1 or RxNE = 1 (regardless of the value of DMAEN)
9	ITEVTEN	RW	0	Event interrupt enable 0: Disabled 1: Event interrupt enabled This interrupt is generated when: ● SB = 1 (Master) ● 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 enabled. 0: Disable error interrupt; 1: Allow error interrupt; This interrupt is generated when: ● BERR = 1

Bit	Name	R/W	Reset Value	Function
				● ARLO = 1 ● AF = 1 ● OVR = 1 ● PECERR = 1 ● TIMEOUT = 1; ● SMBAlert = 1.
7	Reserved	-	-	Reserved
6:0	FREQ	RW	0	I2C module 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 I^{2}C specifications. The minimum allowable settable frequencies are 24MHz (100k), 8MHz (400k), and the maximum frequency of 16MHz (1MHz) is the highest APB clock frequency of the chip. The correct input clock frequency must be set to produce the correct timing, and the allowable range is 2 ~ 36 MHz: 0000000: Disabled 0000001: Disabled 0000010: Disabled 0000011: Disabled 0000100: 4 MHz ….. 0100100: 36 MHz ….. 0110000: 48 MHz ….. 1001000: 72 MHz Greater than 1001000: Disabled.

27.5.3. I2C own address register 1 (I2C_OAR1)

Address offset: 0x08

Reset value: 0x0000

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	-	-	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); Note: This register is fixed to 0 if it does not support the 10bit address function.
14:10	Reserved	-	-	Reserved
9:8	ADD[9:8]	RW	0	Interface address. 7-bit address mode This register is invalid. 10-bit address mode 9 to 8 bits of the bit address. Note: This register is fixed to 0 if it does not support the 10bit address function.
7:1	ADD[7:1]	RW	0	Bits 7:1 of interface address
0	ADD0	RW	0	Interface address. 7-bit address mode This register is invalid. 10-bit address mode bit 0 bit of the address. Note: This register is fixed to 0 if it does not support the 10bit address function.

27.5.4. I2C own address register 2 (I2C_OAR2)

Note: This register does not support double-byte address function and is fixed to 0.

Address offset: 0x0C

Reset value: 0x0000

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	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:8	Reserved	-	-	Reserved

10:8	OA2MSK [2:0]	RW	0	ADD2 Shield Configuration.
				000: unshielded;
				001: ADD2 [1] mask, comparing only ADD2 [7: 2];
				010: ADD2 [2: 1] mask, comparing only ADD2 [7: 3];
				011: ADD2 [3: 1] mask, comparing only ADD2 [7: 4];
				100: ADD2 [4: 1] mask, comparing only ADD2 [7: 5];
				101: ADD2 [5: 1] mask, comparing only ADD2 [7: 6];
				110: ADD2 [6: 1] mask, comparing only ADD2 [7];
				111: ADD2 [7: 1] mask, ACK all received 7-bit addresses;
				Note: When OA2MSK is not 0, the reserved I2C addresses (0b0000xxx and 0b1111xxx) will not ACK, even if the addresses are aligned correctly.

7:1	ADD2 [7:1]	RW	0	Bits 7:1 of interface address 7 ~ 1 bits of the address in dual address mode.
0	ENDUAL	RW	0	Dual address mode enable bit. 0: In 7-bit address mode, only OAR1 is recognized; 1: In the 7-bit address mode, both OAR1 and OAR2 are recognized.

27.5.5. I^{2}C Data register (I2C_DR)

Address offset: 0x10

Reset value: 0x0000

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
31:16	Reserved	-	-	Reserved
15:8	Reserved	-	-	Reserved
7:0	DR[7:0]	RW	0	The 8-bit data register, inside the chip, actually two independent buffers 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). **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 DR

Bit	Name	R/W	Reset Value	Function
				2) Write collision is not managed (DR can be written if TxE=0). 3) If an ARLO event occurs on ACK pulse, the received byte is not copied into DR and so cannot be read.

27.5.6. I²C status register (I2C_SR1)

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMBALERT	TIMEOUT	Res.	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
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: In reception when a new byte is received (including ACK pulse) and the DR register has not been read yet. New received byte is lost. In transmission when a new byte should be sent and the DR register has not been written yet. The same byte is sent twice. 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).
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

Bit	Name	R/W	Reset Value	Function
				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 PEC is the next byte to be sent (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	-	-	Reserved
4	STOPF	R	0	Stop detection (slave mode) 0: No stop bit 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). Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0. Note: The STOPF bit is not set after a NACK reception.
3	Reserved	-	-	Reserved
2	BTF	R	0	Byte transfer complete flag 0: Data byte transfer not done 1: Byte transfer successful The hardware sets this register in the following cases (when slave mode, 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 _ SR2. TRA = 1, I2C _ CR1. PEC = 1), the BTF bit is not set.
1	ADDR	R	0	Address sent (master mode)/matched (slave mode) Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0. **Address matched (Slave):** 0: Address mismatched or not received. 1: Received address matched. The hardware will set this bit when the slave address is received. **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.

Bit	Name	R/W	Reset Value	Function
				Address sent (Master): 0: No end of address transmission 1: End of address transmission — When the 7-bit address is received, it is set after ACK byte. **Note: This register is not set after receiving NACK.**
0	SB	R	0	Start bit (Master mode) 0: No Start condition 1: Start condition generated. -Set the register when a start condition is sent. -After the software reads the I2C _ SR1 register, a write operation to the data register will clear this bit; Or cleared by hardware when PE = 0.

27.5.7. I²C status register 2 (I2C_SR2)

Address offset: 0x18

Reset value: 0x0000

Note: Even if the ADDR flag bit is set after reading the I2C _ SR1 register, reading the I2C _ SR2 register after reading I2C _ SR1 will clear the ADDR flag bit. Consequently, I2C_SR2 must be read only when ADDR is found set in I2C_SR1 or when the STOPF bit is cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.								Res.	Res.	Res.	GENCALL	Res.	TRA	BUSY	MSL
											R		R	R	R

Bit	Name	R/W	Reset Value	Function
15:5	Reserved	-	-	Reserved
4	GENCALL	R	0	General 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	-	-	Reserved
2	TRA	R	0	Transmit/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 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 0: slave 1: master - 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.

27.5.8. I2C Clock control register (I2C_CCR)

Address offset: 0x1C

Reset value: 0x0000

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	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
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 F+ master mode selection 0: Standard mode or fast mode, which one to choose is determined by bit15; 1: Fast mode plus; Note: When configuring this register to 1, the value of bit15 is not concerned;
12	Reserved	-	-	Reserved
11:0	CCR[11:0]	RW	0	Clock control register in Fm/Sm mode (Master mode) Controls the SCL clock in master mode. ■ Standard mode: — T_{high}=CCR x Tpclk — T_{low} =CCR x Tpclk ■ Fast mode: — DUTY=0: – T_{high}=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 ■ In fast boost mode: — DUTY=0: – Thigh = 3 * CCR x Tpclk – Tlow = 5 x CCR x Tpclk — DUTY = 1 (up to 1 MHz): – Thigh = 2 x CCR x Tpclk – Tlow = 3 x CCR x Tpclk Notes: ■ The minimum allowed value is 0x04, and the minimum allowed value in fast DUTY mode is 0x01 ■ Thigh=tr(SCL)+tw(SCLH) ■ Tlow=tr(SCL)+tw(SCLL) ■ These delays do not have filters ■ This register can only be configured if PE = 0;

Note: The combination of bit15 and bit13 extends the pattern as shown in the following table:

F +	F/S	Mode
0	0	Standard mode
0	1	Fast mode
1	0	Fast enhancement mode
1	1	Fast enhancement mode

27.5.9. I²C TRISE register (I2C_TRISE)

Address offset: 0x20

Reset value: 0x0082

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	THOLDDATA _ SEL	THOLDDATA					TRISE[6:0]						
									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:13	Reserved	-	-	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	THLDDATA	RW	1	Maximum data retention time in fast/standard/fast enhanced mode (transmission mode) These bits are the minimum holding time used to ensure the data holding time in the data transmission mode. For example, the maximum SDA drop time bit allowed in standard mode is 300ns. If the value in the I2C _ CR2 register type FREQ [6: 0] is equal to 0x08, Tpclk = 125ns, then configured to 0x03 in TRISE (300ns/125ns = 2.4 +1) If the result is not an integer, the integer part is written to THLDDATA to ensure configuration.
6:0	TRISE	RW	0x2	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 stable SCL frequency whatever the SCL rising edge duration. These bits must be programmed with the maximum SCL rise time given in the I^{2}C bus specification, incremented by 1. For instance: in Sm mode, the maximum allowed SCL rise time is 1000 ns. If, in the I2C_CR2 register, the value of FREQ[5:0] bits is equal to 0x08 and TPCLK1 = 125 ns therefore the TRISE[5:0] bits must be programmed with 09h.(1000 ns / 125 ns = 8 + 1 = 9). T he filter value can also be added to TRISE. If the result is not an integer, TRISE[5:0] must be programmed with the integer part, in order to respect the t_{HIGH} parameter. Note: TRISE[5:0] must be configured only when the I^{2}C is disabled (PE = 0).

28. Universal synchronous asynchronous receiver transmitter (USART)

This project designs and implements three USART modules, USART1 supports LIN, USART2 and USART3 have exactly the same functions, and do not support IrDA, LIN and SmartCard; The stop bits of 3 USART only support 1-bit and 2-bit.

28.1. Introduction

The USARTs provide a flexible method for full-duplex data exchange with external devices using the industry-standard NRZ asynchronous serial data format. The USART utilizes a fractional baud rate generator to provide a wide range of baud rate options.
The USART supports both synchronous one-way and Half-duplex Single-wire communications. It also supports multiprocessor communications.

28.2. USART main features

- Full-duplex asynchronous communication
- NRZ standard format
- Configurable 16 times or 8 times oversampling for increased flexibility in speed and clock tolerance
- Programmable baud rate shared by transmit and receive, up to 4.5 Mbit/s
- Auto baud rate detection
- Programmable data length of 8 or 9 bits
- Configurable stop bit (1 or 2 bits)
- Synchronous mode and clock output function for synchronous communication
- Single-wire Half-duplex communications
- Independent transmit and receive enable bits
- Hardware flow control
- Transfer detection flag
  - Receive buffer full
  - Send empty buffer
  - End of transmission flags
- Parity control
  - Transmit parity bit
  - Check the received data byte
- Flagged interrupt sources
  - CTS change
  - Transmit data register empty
  - Transmission complete
  - Receive full data register
  - Bus idle detected

- Overrun error
- Framing error
- Noise operation
- Detection error

- Multiprocessor communication
  - If addresses do not match, enter silent mode Wake up from silent mode: by idle detection and address flag detection
- Wake-up from silent mode: by idle detection and address flag detection

28.3. USART functional description

The interface is externally connected to another device by three pins. Any USART bidirectional communications require a minimum of two pins: Receive Data In (R_X) and Transmit Data Out (T_X).

RX: Receive data serial input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

TX: Transmit data output. When the transmitter is disabled, the output pin returns to its IO port configuration. When the transmitter is enabled and nothing is to be transmitted, the T_X pin is at high level. In single-wire mode, this IO is used to transmit and receive the data.

- An idle line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) the least significant bit first
- 1, 2 Stop bits indicating that the frame is complete
- Using a Fractional Baud Rate Generator: A Representation of 12-Bit Integers and 4 Decimals
- A status register (USART_SR)
- Data register (USART _ DR)
- A baud rate register (USART_BRR), 12-bit mantissa and 4-bit fraction.

The following pin is required to interface in synchronous mode:

CK: Transmitter clock output.

This pin outputs the transmitter data clock for synchronous transmission (no clock pulses on start bit and stop bit, and a software option to send a clock pulse on the last data bit). In parallel data can be received synchronously on R_X. This can be used to control peripherals that have shift registers (e.g. LCD drivers). The clock phase and polarity are software programmable.

The following pins are required in hardware flow control mode:

- **nCTS**: Clear to send blocks the data transmission at the end of the current transfer when high
- **nRTS: Request to send indicates that the USART is ready to receive a data (when low).**

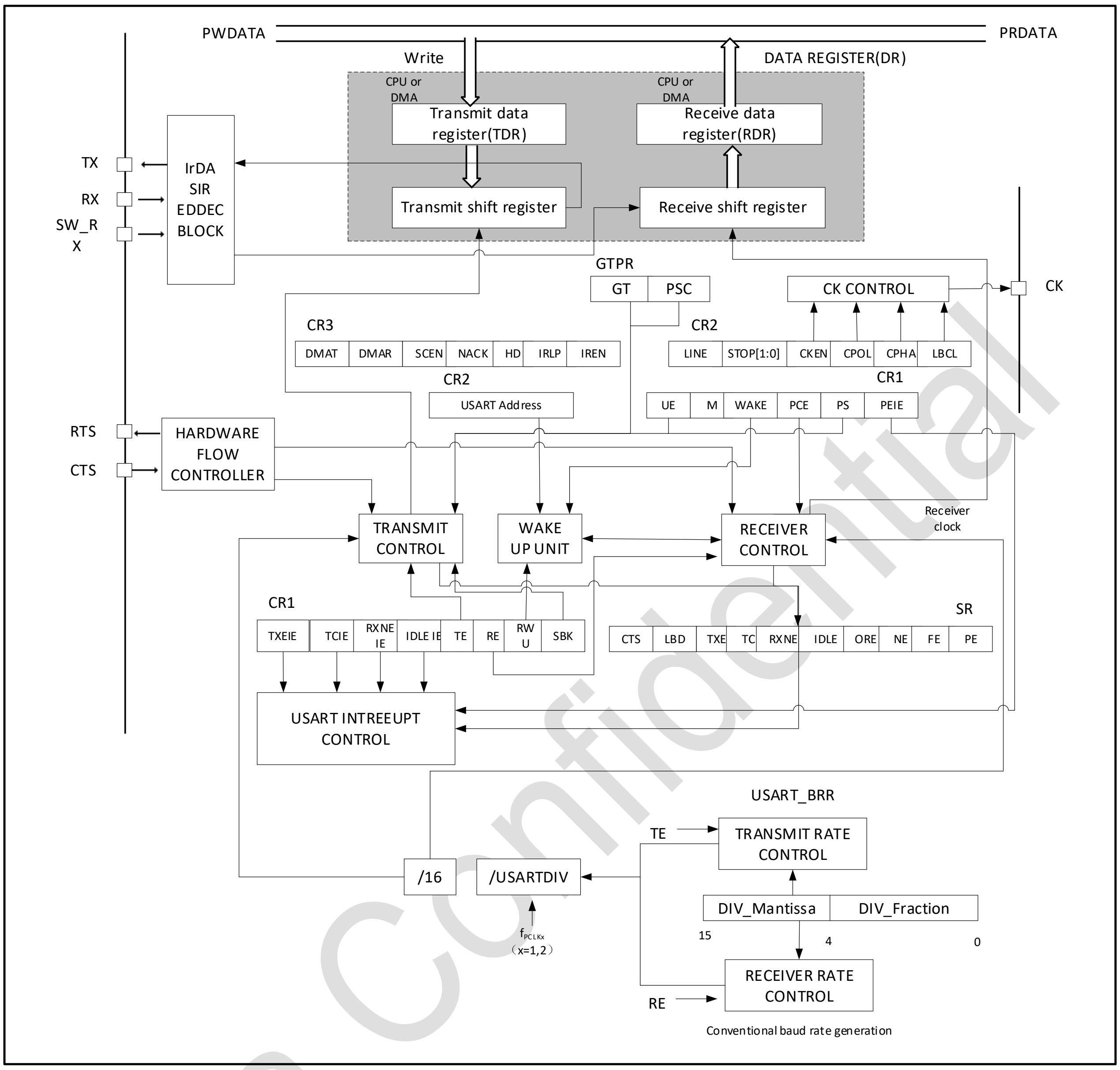


Figure 28-1 USART block diagram

28.3.1. USART character description

Word length may be selected as being either 8 or 9 bits by programming the M bit in the USART_CR1 register. The T_X pin is in low state during the start bit. It is in high state during the stop bit.

An Idle character is interpreted as an entire frame of 1 followed by the start bit of the next frame which contains data (The number of 1's will include the number of stop bits).

A Break character is considered to have received all '0' in a frame period (including during the stop bit, also '0'). At the end of the break character, the transmitter inserts 1 or 2 more stop bits ('1') in response to the start bit.

Transmission and reception are driven by a common baud rate generator, the clock for each is generated when the enable bit is set respectively for the transmitter and receiver.

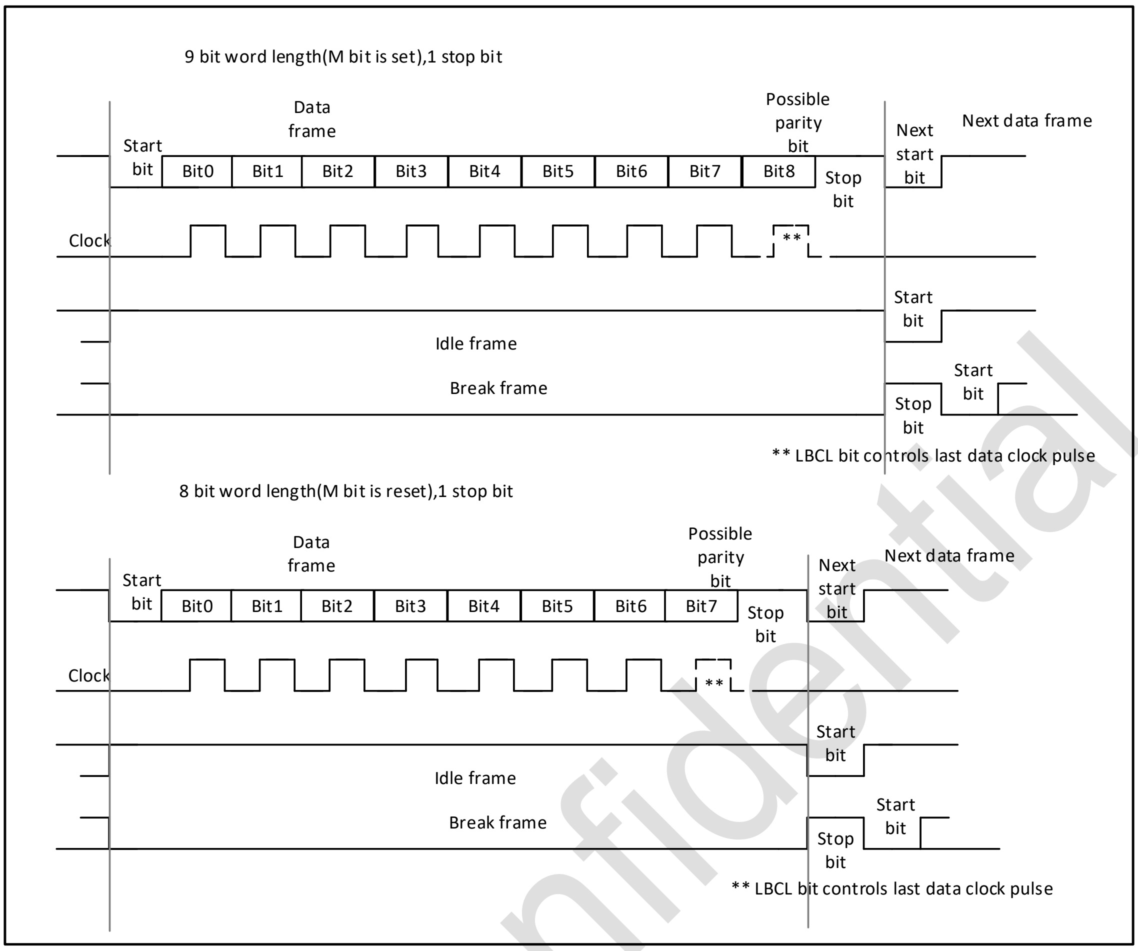


Figure 28-2 Word length setting

28.3.2. 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.

28.3.2.1. Character transmission

During a USART transmission, data shifts out the least significant bit first on the T_X pin. In this mode, the USART_DR register consists of a buffer (TDR) between the internal bus and the transmit shift register.

Each character is preceded by a low start bit; Following stop bits, the number of which can be configured. 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 T_X 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.

28.3.2.2. Configurable stop bits

The number of bits of the stop bits sent with each character can be programmed by bits 13, 12 of the control register 2.

1) 1 stop bit: the default value of the number of stop bits.

2) 2 stop bits: Can be used in regular USART mode, single-wire mode, and 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).

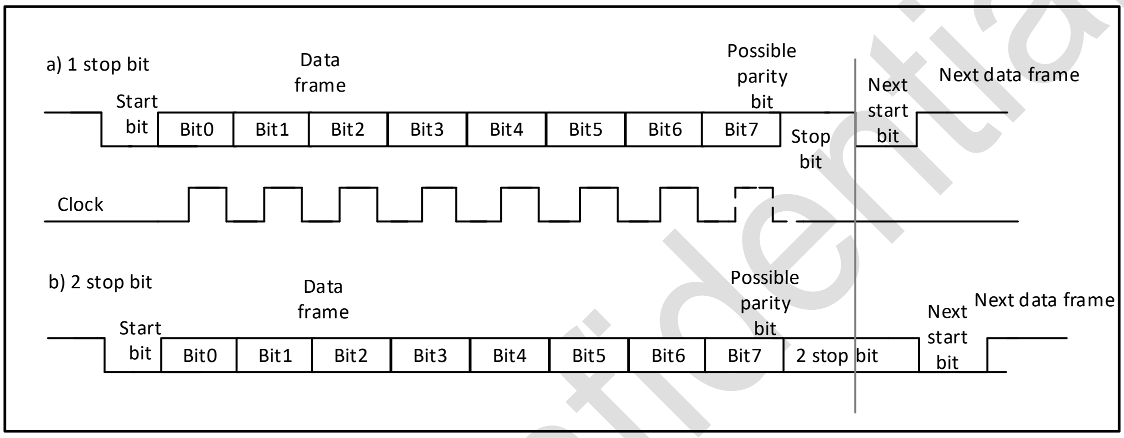


Figure 28-3 Configurable stop bits

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) Program the number of Stop bits in USART_CR2.

4) Select DMA enable (DMAT) in USART_CR3 if multibuffer communication must take place. Configure the DMA registers as described in Multi-Buffer 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 send in the USART_DR register (this clears the 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.

28.3.2.3. 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 emptied

- 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.

After writing the last data into the USART_DR register, it is mandatory to wait for TC=1 before disabling the USART or causing the microcontroller to enter the low-power mode.

Use the following software procedure to clear the TC bit:

1. Read the USART _ SR register once;

2. Write the USART _ DR register once.

Note: The TC bit is cleared by writing 0 to it. This clearing sequence is recommended only for Multibuffer communication.

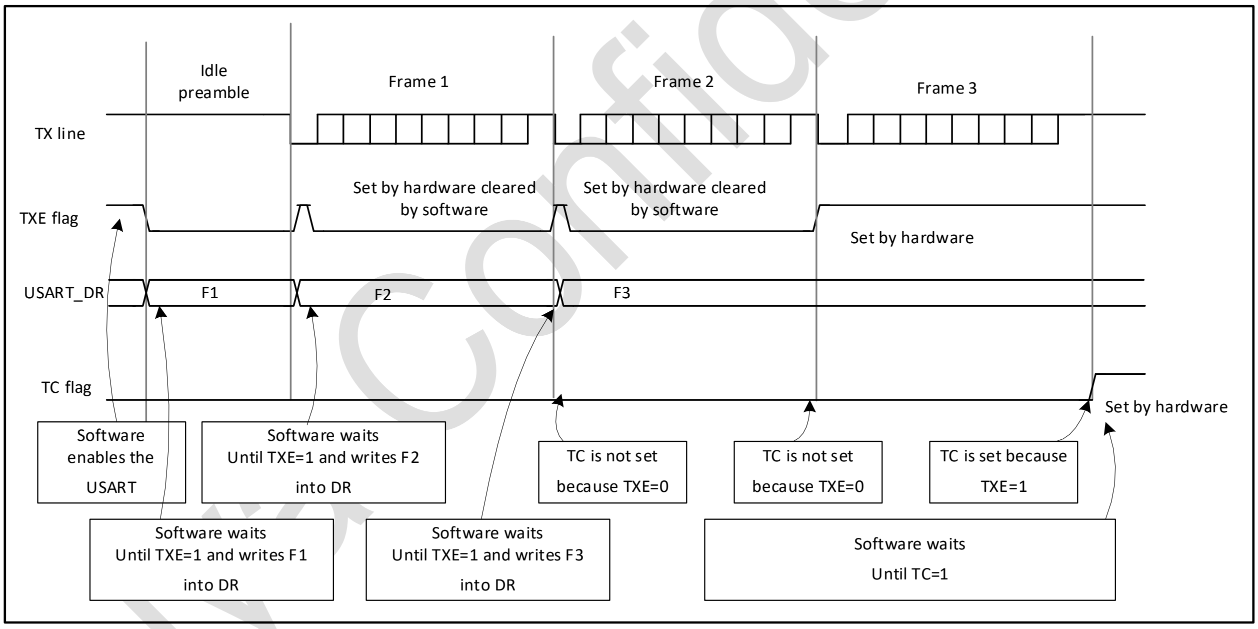

Idle preamble
Frame 1
Frame 2
Frame 3
TX line
Set by hardware cleared by software
Set by hardware cleared by software
TXE flag
Set by hardware
USART_DR
F1
F2
F3
TC flag
Set by hardware
Software enables the USART
Software waits Until TXE=1 and writes F2 into DR
TC is not set because TXE=0
TC is not set because TXE=0
TC is set because TXE=1
Software waits Until TXE=1 and writes F1 into DR
Software waits Until TXE=1 and writes F3 into DR
Software waits Until TC=1

Figure 28-4 TC/TXE behavior when transmitting

28.3.2.4. Disconnect symbol

Set the SBK to send a break symbol. The disconnect frame length depends on M bits. If SBK = 1 is set, a disconnect symbol will be sent on the TX line after the current data transmission is completed. The SBK is reset by hardware upon completion of the disconnect character transmission (at the disconnect of the stop bit of the symbol). USART inserts a logical '1' at the end of the last broken frame to ensure that the start bit of the next frame can be recognized.

Note: If the software resets the SBK bit again before starting to send the disconnect frame, the disconnect symbol will not be sent. If two consecutive broken frames are to be transmitted, the SBK bit should be set after the stop bit of the previous broken symbol.

28.3.2.5. Idle characters

Setting the TE bit drives the USART to send an idle frame before the first data frame.

28.3.3. Receiver

USART may receive an 8-bit or 9-bit data word according to M bits of USART _ CR1.

28.3.3.1. Start bit detection

In the USART, the start bit is detected when a specific sequence of samples is recognized. This sequence is: 1 1 1 0 X 0 X 0X 0X 0 X 0X 0.

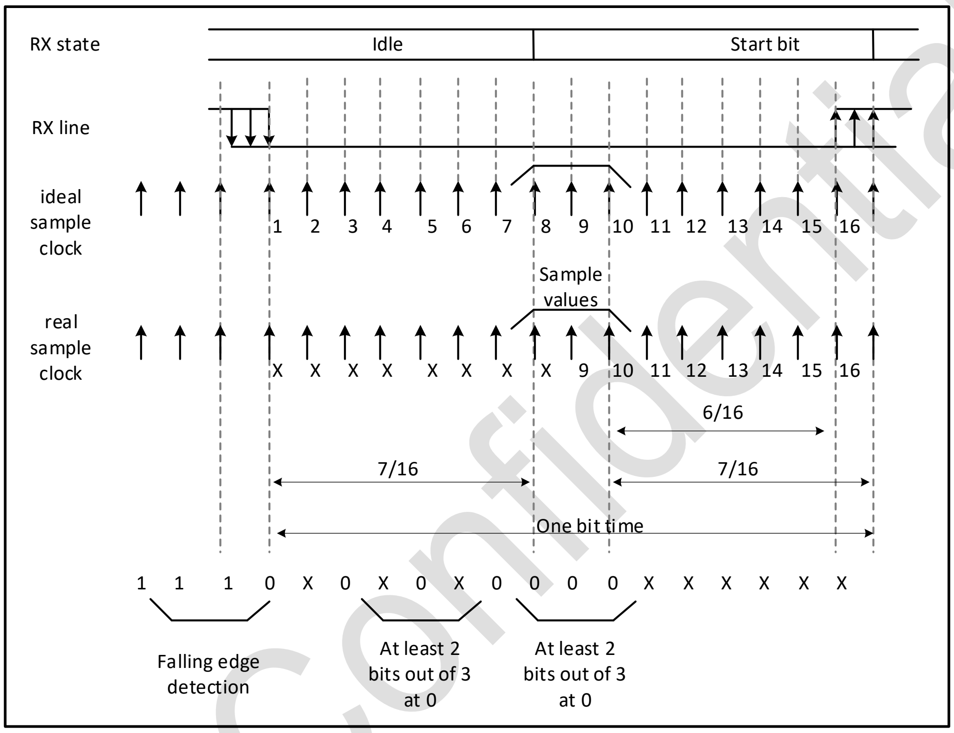


Figure 28-5 Start bit detection

If the sequence is not complete, the start bit detection aborts and the receiver returns to the idle state (no flag is set) where it waits for a 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 is not met, the start detection aborts and the receiver returns to the idle state (no flag 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.

28.3.3.2. Character reception

During USART reception, the least significant bits of the data are first moved in from 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. 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. Program the number of stop bits in USART_CR2.
4. Select DMA enable (DMAR) in USART_CR3 if multibuffer communication must take place. Configure the DMA registers as required for multi-buffer communication.
5. Select the desired baud rate using the baud rate register USART_BRR
6. Set the RE bit USART_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 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.

28.3.3.3. Disconnect symbol

When a broken frame is received, USART handles it as if it were a frame error.

28.3.3.4. 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.

28.3.3.5. 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.

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 EIE bit is 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 occur when new data is received during a read sequence (between a USART _ SR register read access and a USART _ DR read access).

28.3.3.6. Noise error

Over-sampling techniques are used (except in synchronous mode) for data recovery by discriminating between valid incoming data and noise.

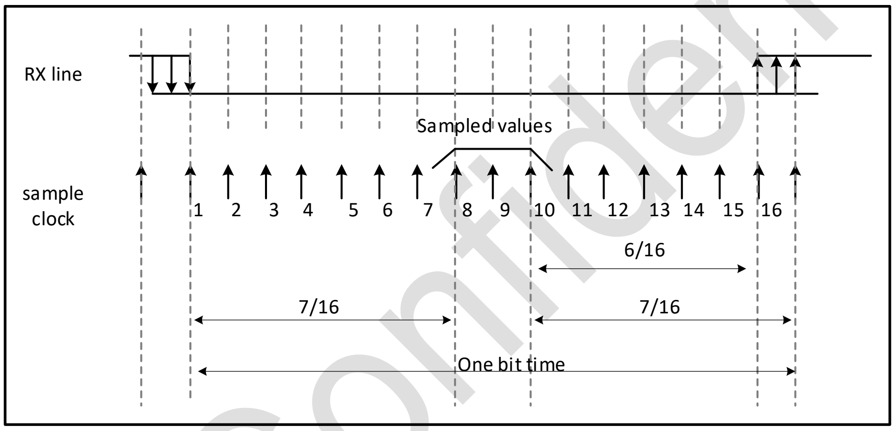


Figure 28-6 Data sampling for noise detection

Table 28-1 Data sampling for noise detection

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, because the NE flag bit and the RXNE flag bit are set at the same time, the RXNE will generate 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.

28.3.3.7. Framing error

Frame error is detected when:

Due to the lack of synchronization or a large amount of noise, the stop bit is not recognized at the expected time.

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.

28.3.3.8. Configurable stop bits during reception

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 no longer checks for frame errors. The RXNE flag is set at the end of the first stop bit.

28.3.4. USART baud rate generation

Generation of fractional baud rates The baud rates of the receiver and transmitter should be set to the same values in the integer and decimal registers of USARTDIV.

Tx / Rx baud $= f_{CK}$ /(16*USARTDIV)

Here fCK is the clock given to the peripheral USARTDIV is an unsigned fixed-point number. The 12-bit value is set in the USART _ BRR register.

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.

How to derive USARTDIV from USART_BRR register values

Example 1:

If DIV _ Mantissa = 27, DIV _ Fraction = 12 (USART _ BRR = 0x1BC),

Then:

Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = 12/16 = 0.75

Therefore, USARTDIV = 27.75

Example 2:

Therefore, USARTDIV = 25.62

There is:

DIV _ Fraction = 16 * 0.62 = 9.92

The closest integer is: 10 = 0x0A

DIV _ mantissa = mantissa (25.620) = 25 = 0x19

Thus, USART _ BRR = 0x19A

Example 3:

要求 USARTDIV = 50.99

There is:

DIV _ Fraction = 16 * 0.99 = 15.84

The closest integer is: 16 = 0x10 = > DIV _ frac [3: 0] Overflow = > The carry must be added to the fractional part

DIV _ mantissa = mantissa (50.990 + carry) = 51 = 0x33

Thus: USART _ BRR = 0x330, USARTDIV = 51

Baud rate		FPCLK=36 MHz			FPCLK=72 MHz		
No.	Kbps	Actual	Value placed in the baud rate register	Error (%)	Actual	Value placed in the baud rate register	Error (%)
1	2.4	2.400	937.5	0 %	2.4	1875	0 %
2	9.6	9.600	234.375	0 %	9.6	468.75	0 %
3	19.2	19.2	117.1875	0 %	19.2	234.375	0 %
4	57.6	57.6	39.0625	0 %	57.6	78.125	0 %
5	115.2	115.384	19.5	0.15 %	115.2	39.0625	0 %
6	230.4	230.769	9.75	0.16 %	230.769	19.5	0.16 %
7	460.8	461.538	4.875	0.16 %	461.538	9.75	0.16 %
8	921.6	923.076	2.4375	0.16 %	923.076	4.875	0.16 %
9	2250	2250	1	0 %	2250	2	0 %
10	4500	Impossible	Impossible	Impossible	4500	1	0 %

Note: The lower the clock frequency of the CPU, the lower the error of a specific baud rate. The upper limit of the achievable baud rate can be obtained from this set of data.

28.3.5. Tolerance of the USART receiver to clock deviation

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: 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).
- DTRA + DQUANT + DREC + DTCL < USART receiver tolerance
- The USART receiver tolerance to properly receive data is equal to the maximum tolerated deviation and depends on the following choices:
- 10- or 11-bit character length defined by the M bit in the USART_CR1 register
- Use of fractional baud rate or not

Table 28-2 Tolerance of the USART receiver when DIV_Fraction is 0

M bit	NF is an error	NF don't care
0	3.75 %	4.375%
1	3.41%	3.97%

Table 28-3 Tolerance of the USART receiver when DIV_Fraction is different from 0

M bit	NF is an error	NF don't care
0	3.33%	3.88%
1	3.03%	3.53 %

28.3.6. USART Auto baud rate detection

The USART can detect and automatically set the USARTx_BRR register value based on the reception of one character. Automatic baud rate detection is useful under following circumstances:

1) The communication speed of the system is not known in advance.
2) The system is using a relatively low accuracy clock source and this mechanism allows the correct baud rate to be obtained without measuring the clock deviation.

The clock source frequency must be compatible with the expected communication speed. (You must select 16x oversampling and choose between fCK/65535 and fCK/16)

The automatic baud rate detection mode must be selected (via the ABRMOD [1: 0] bit of the USARTx _ CR3 register) before the automatic baud rate detection can be activated. There are various patterns based on different character patterns.

In these auto baud rate modes, the baud rate is measured several times during the synchronization data reception and each measurement is compared to the previous one.

These modes are the following:

Mode 0: Any character starting with a bit at ‘1’. In this case the USART measures the duration of the start bit (falling edge to rising edge).

Mode 1: Any character starting with a 10xx bit pattern. In this case, the USART measures the duration of the Start and of the 1st data bit. The measurement is done falling edge to falling edge, to ensure a better accuracy in the case of slow signal slopes.

Another check of the transition of each RX will also be done in parallel. If the transition on the RX is not sufficiently synchronized with the receiver (the receiver's baud rate calculated based on bit 0), an error will be generated.

Prior to activating the auto baud rate detection, the USARTx_BRR register must be initialized by writing a non-zero baud rate value.

The automatic baud rate detection is activated by setting the ABREN bit in the USARTx_CR3 register. The USART will then wait for the first character on the RX line. The auto baud rate operation completion is indicated by the setting of the ABRF flag in the USARTx_ISR register. If the line is noisy, the correct baud rate detection cannot be guaranteed. In this case the BRR value may be corrupted and the ABRE error flag will be set. This also happens if the communication speed is not compatible with the automatic baud rate detection range (bit duration not between 16 and 65536 clock periods).

The RXNE interrupt will show the completion of the operation. The auto baud rate detection can be re-launched later by resetting the ABRF flag (by writing a ′0′).

Note: The BRR value might be corrupted if the USART is disabled (UE = 0) during an auto baud rate operation.

28.3.7. Multiprocessor communication

It is possible to perform USART multiprocessor communications (with several USARTs connected in a network). For instance one of the USARTs can be the master with its TX output connected to the RX inputs of the other USARTs, while the others are slaves with their respective TX outputs logically ANDed together and connected to the RX input of the master.

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 non-addressed devices can be placed in Mute mode. When the Mute mode is enabled:

- None of the reception status bits can be set;
- All the receive interrupts are inhibited.
- The RWU bit in the USARTx _ CR1 register is set to 1. RWU can be controlled automatically by hardware or by software under certain conditions.
- The USARTx can enter or exit from Mute mode using one of two methods, depending on the WAKE bit in the USARTx_CR1 register:
- Idle Line detection if the WAKE bit is reset,
- Address mark detection if the WAKE bit is set.

28.3.7.1. 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. Then the RWU bit is cleared by hardware, but the IDLE bit is not set in the USART_SR register. 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.

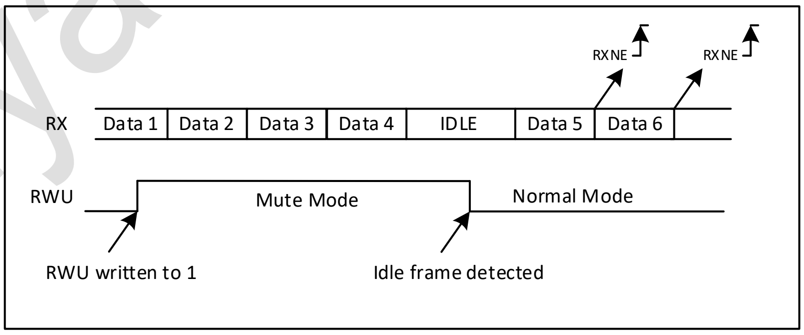


Figure 28-7 Mute mode using Idle line detection

28.3.7.2. Address mark detection (WAKE=1)

In this mode, bytes are recognized as addresses if their MSB is a ‘1’ else they are considered as data. In an address byte, the address of the targeted receiver is put in the 4 LSBs. This 4-bit word is compared by the receiver with its own address which is programmed in the ADD bits in the USART_CR2 register.

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 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.

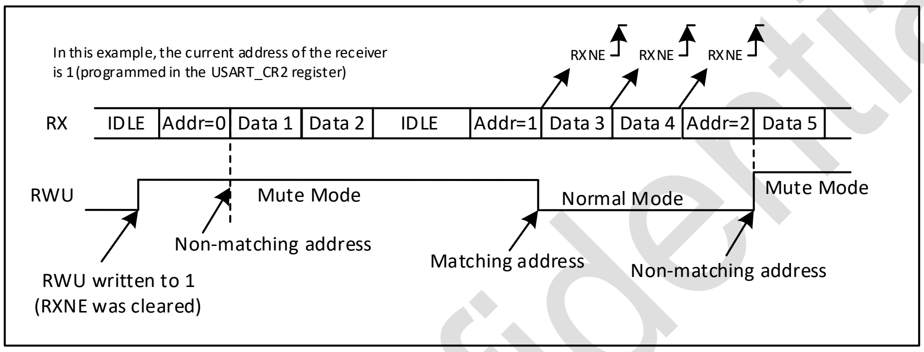


Figure 28-8 Mute mode using address mark detection

28.3.7.3. 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. Depending on the frame length defined by the M bit, the possible USART frame formats are as listed in the table below.

Table 28-4 Frame formats

M bit	PCE bit	USART fram
0	0	SB—8 bit data—STB
0	1	SB—7 bit data—PB—STB
1	0	SB—9 bit data—STB
1	1	SB—8 bit data—PB—STB

When waking up the device with an address tag, the matching of the address only takes into account the MSB bits of the data, regardless of the check bits. (MSB is the last one issued in the data bit, followed by the check bit or stop bit)

28.3.7.4. Even parity

The check bits are such that 7 or 8 pieces of LSB data in a frame and the number of '1' in the check bits are even.

As an example, if data = 00110101 and 4 bits are set, the parity bit is equal to 0 if even parity is selected (PS bit in USARTx_CR1 = 0).

28.3.7.5. Odd parity

This check bit makes 7 or 8 pieces of LSB data in a frame and the number of '1' in the check bit odd.

As an example, if data = 00110101 and 4 bits set, then the parity bit is equal to 1 if odd parity is selected (PS bit in USARTx_CR1 = 1).

28.3.7.6. Transmission mode:

If the PCE bit is set in USARTx_CR1, then the MSB bit of the data written in the data register is transmitted but is changed by the parity bit (even number of“1s” if even parity is selected or an odd number of “1s” if odd parity is selected). If the parity check fails, the PE flag is set in the USART_SR register and an interrupt is generated if PEIE is set in the USART_CR1 register.

28.3.8. USART synchronous mode

The synchronous master mode is selected by programming the CLKEN bit in the USART_CR2 register to ‘1’. In synchronous mode, the following bits must be kept cleared:

HDSEL bit 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.

During the Idle state, preamble and send break, the external CK clock is not activated.

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 asyn-chronous 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).

Note:

The CK pin operates in conjunction with the TX pin. Thus, the clock is provided only if the transmit-ter 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.

The correct configuration of the LBCL, CPOL and CPHA bits should be when both the transmitter and receiver are disabled; When the transmitter or receiver is enabled, these bits cannot be changed.

It is advised that TE and RE are set in the same instruction to minimize the setup and the hold time of the receiver.

The USART supports master mode only: it cannot receive or send data related to an input clock (CK is always an output).

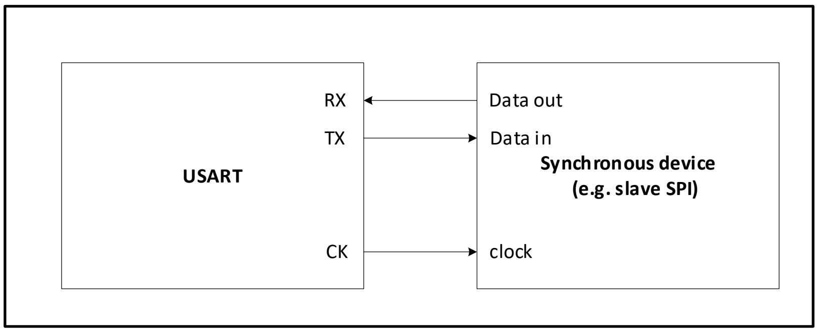


Figure 28-9 USART example of synchronous transmission

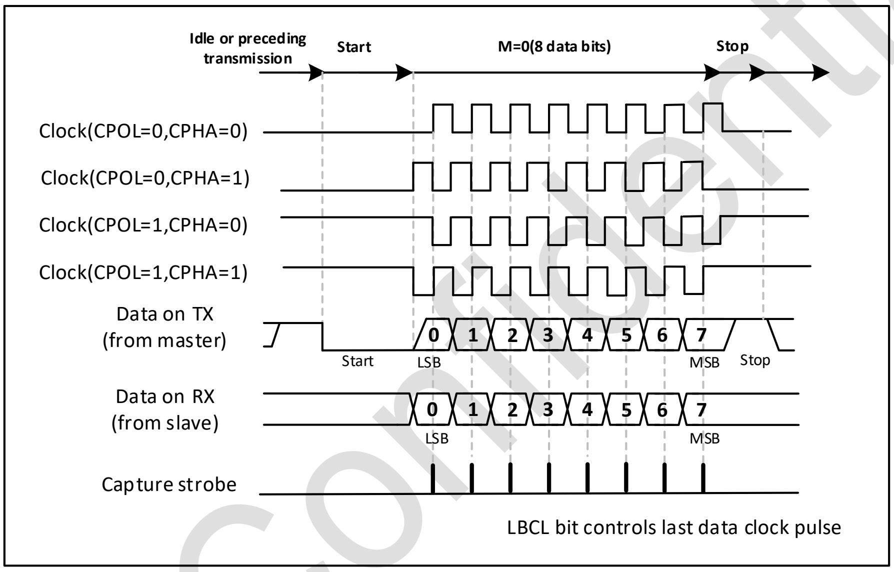


Figure 28-10 USART data clock timing diagram (M=0)

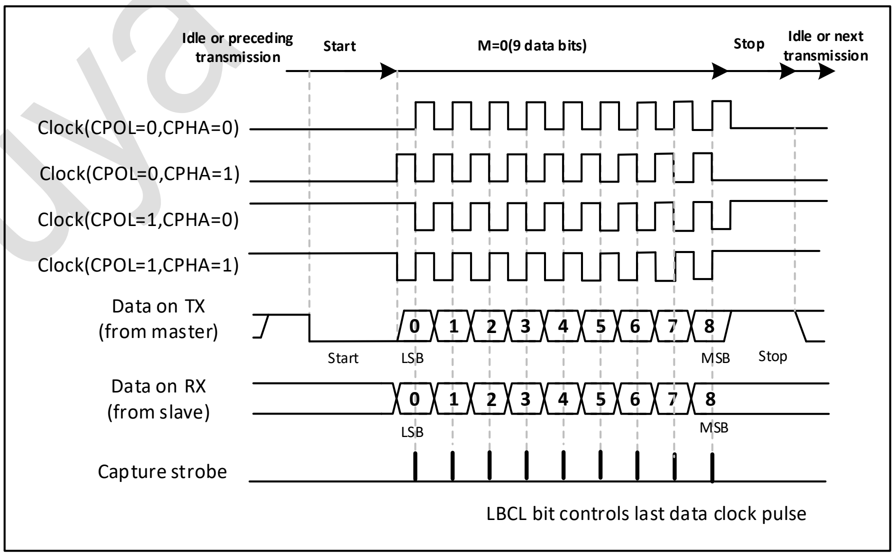


Figure 28-11 USART data clock timing diagram (M=1)

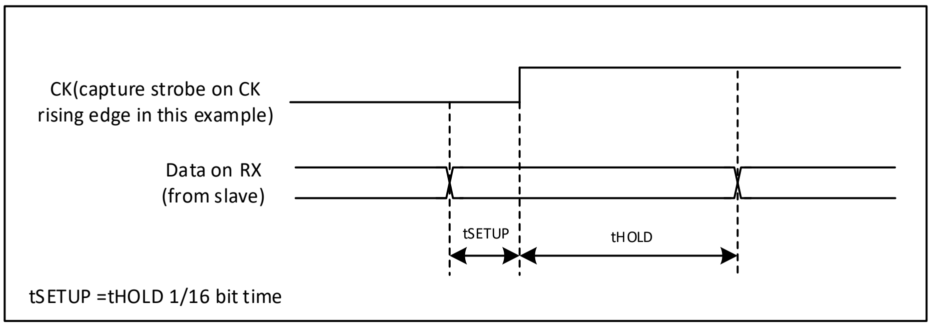

CK(capture strobe on CK rising edge in this example)
Data on RX (from slave)
tSETUP
tHOLD
tSETUP =tHOLD 1/16 bit time

Figure 28-12 RX data sample/hold time

28.3.9. Single-wire Half-duplex communications

The single-wire half-duplex mode is selected by setting the HDSEL bit in the USARTx_CR3 register. In this mode, the following bits must be kept cleared:

- CLKEN in the USARTx_CR2 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 communications are selected using the control bit "HALF DUPLEX SEL" (the HDSEL bit in USARTx _ CR3). When HDSEL is' 1 '

- RX is no longer in use
- The TX pin is always released when no data is transmitted. Thus, it acts as a standard I/O in idle or in reception. It means that the I/O must be configured so that TX is configured as alternate function open-drain with an external pull-up.

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, transmission is never hindered 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.

28.3.10. 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.

28.3.10.1. DMA transmission

Sending using DMA can be activated by setting the DMAT bit on 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. The steps to assign a DMA channel for USART transmission are as follows (x represents the channel number):

1) Configure the USART _ DR register address as the destination address of DMA transfer in the DMA control register. After each TXE event, data will be transferred to this address.

2) Configure the memory address as the source address of the DMA transfer in the DMA control register. After each TXE event, data will be read from this memory area and transferred to 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) According to the requirements of the application, configure a DMA interrupt when the transmission is half or all completed.

6) Clear the TC bit by writing 0.

7) Activate the channel in 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.

In the transmission mode, when the DMA has finished transmitting all the data to be transmitted, the DMA controller sets the TCIF flag of the DMA _ ISR register; Monitoring the TC flag of the USARTx _ SR register can confirm whether the USART communication has ended, so that it can avoid corrupting the last transmitted data before shutting down USART or entering shutdown mode; The software needs to wait for TXE = 1 before waiting for TC

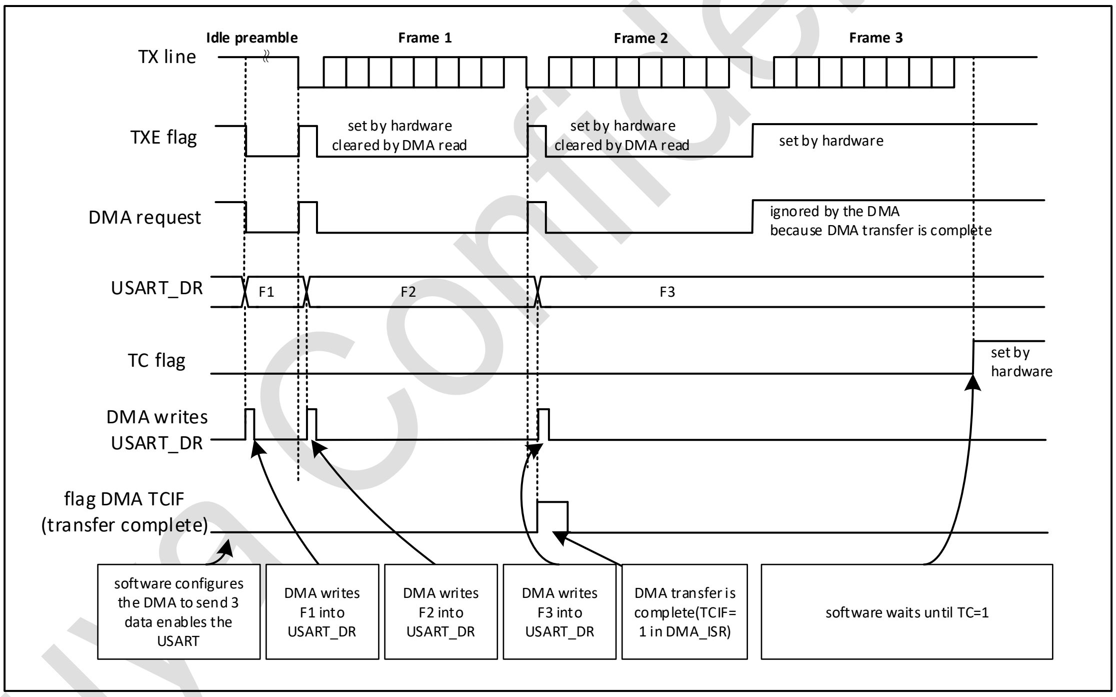

Idle preamble
Frame 1
Frame 2
Frame 3
TX line
TXE flag
set by hardware cleared by DMA read
set by hardware cleared by DMA read
set by hardware
DMA request
ignored by the DMA because DMA transfer is complete
USART_DR
F1
F2
F3
TC flag
set by hardware
DMA writes USART_DR
flag DMA TCIF (transfer complete)
software configures the DMA to send 3 data enables the USART
DMA writes F1 into USART_DR
DMA writes F2 into USART_DR
DMA writes F3 into USART_DR
DMA transfer is complete(TCIF= 1 in DMA_ISR)
software waits until TC=1

Figure 28-13 Using DMA Transmission

28.3.10.2. DMA reception

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 transmission through the DMA control register. After each RXNE event, data will be read out from this address and transferred to memory.

2) Configure the memory address as the destination address of the transfer through the DMA control register. After each RXNE event, data will be transferred from USART _ DR to this memory area.

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) According to the requirements of the application, configure whether the DMA interrupt is generated when the transmission is half or all completed.

6) Activate the channel in the DMA control register.

When the transmission amount specified by the DMA controller is received, the DMA controller generates an interrupt on the interrupt vector of the DMA channel.

28.3.10.3. 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. If the interrupt enable bit is set, an interrupt will be generated. In the case of single byte reception, the frame error, overflow error and noise flags set together with the RXNE have a separate error flag interrupt enable bit; If set, an interrupt will be generated after the current byte transfer is completed.

28.3.11. 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.

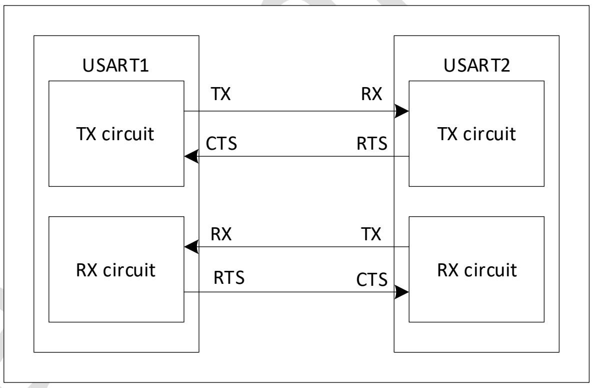


Figure 28-14 Hardware flow control between two USARTs

28.3.11.1. 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.

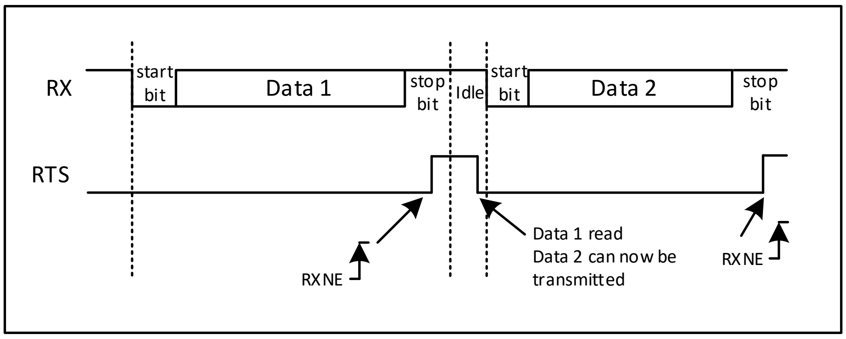


Figure 28-15 RTS flow control

28.3.11.2. 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.

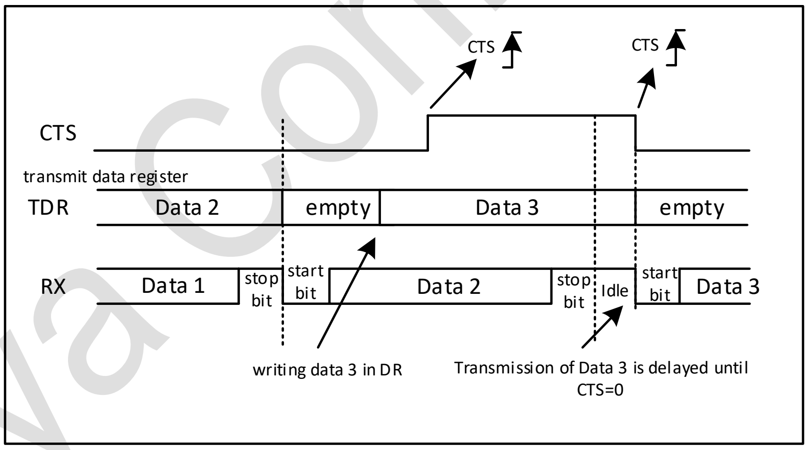


Figure28-16 CTS flow control

28.4. USART interrupts

No.	Interrupt event	Event flag	Enable Control bit	Transmission/Reception
1	Transmit data register empty	TXE	TXEIE	Transmission
2	CTS interrupt	CTSIF	CTSIE	Transmission
3	Transmission is complete	TC	TCIE	Transmission
4	Receive data register not empty (data ready to be read)	RXNE	RXNEIE	Reception
5	Overrun error	ORE		Reception

6	Idle line detected	IDLE	IDLEIE	Reception
7	Parity error	PE	PEIE	Reception
8	Noise error/Overrun error/Framing Error in multi-buffer communication	NR/ORE/FE	EIE	Reception

The USART interrupt events are connected to the same interrupt vector.

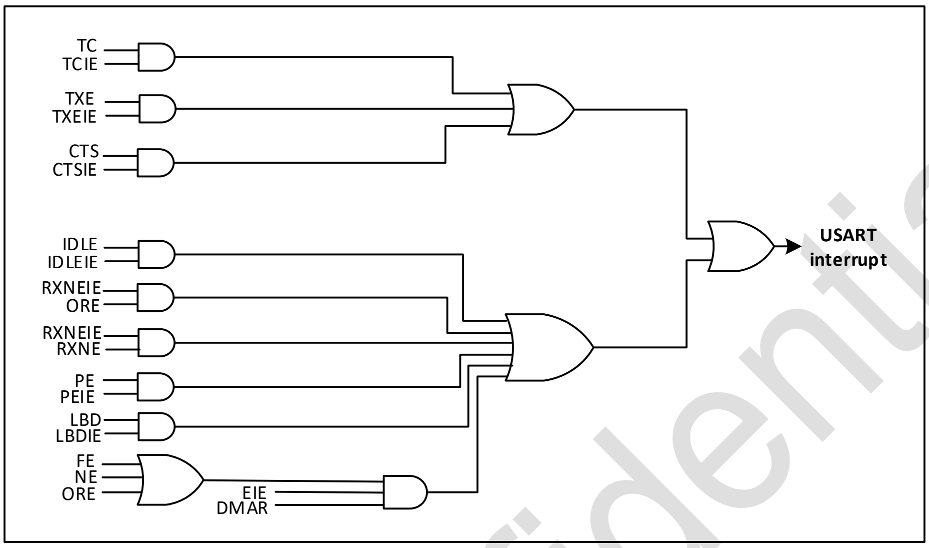


Figure 28-17 USART interrupt mapping diagram

28.5. USART register

28.5.1. USART status register (USART_SR)

Address offset: 0x00

Reset value: 0x0000 00C0

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.	ABRRQ	ABRE	ABRF	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
			W	R	R	RC_W0	RC_W0	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
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. This bit is set by hardware when the automatic baud rate has been set (RXNE is also set, generating an interrupt if RXNEIE = 1) or when the auto baud rate operation was completed without success (ABRE = 1) (ABRE, RXNE and FE are also set in this case).

Bit	Name	R/W	Reset Value	Function
				It is cleared by software, by writing 1 to the ABRRQ in the USART_RQR register.
9	CTS	RC_W0	0	CTS flag When CTS enters toggle, and CTSE = 1, the register is 1. The software writes 0 to clear it. An interrupt is generated on CTS flag if the CTSIE bit is set. 0: CTS line value unchanged 1: CTS line Value change
8	LBD	RC_W0	0	LIN break detection flag. The hardware configures this register when a LIN abort is detected. This bit is cleared by writing 0 An interrupt is generated if the LBDIE bit =1. 0: LIN break not detected; 1: LIN break detected; Note: This register is fixed to 0 if the LIN function is not supported.
7	TXE	R	1	Transfer register empty flag. This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TXEIE bit =1. It is cleared by a write to the USART_DR register. 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	RC_W0	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 first reads the USART _ SR register and then writes the USART _ DR register to clear this bit (for multiprocessor communication). The TC bit can also be cleared by writing a '0' to it. 0: Transmission not complete 1: Transmission is complete
5	RXNE	RC_W0	0	Read data register not empty This bit is set by hardware when the content of the shift register has been transferred to the USART_DR register. It is cleared by a read to the USART_DR register. The RXNE flag can also be cleared by writing a zero to it. An interrupt is generated if the RXNEIE bit =1. 0: Data is not received 1: Ready to receive data
4	IDLE	R	0	DLE line detected The IDLE line is detected and the hardware sets this register. An interrupt is generated if the IDLEIE bit =1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: IDLE line not detected 1: IDLE line detected
3	ORE	R	0	Overrun error 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. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). An interrupt is generated if the RXNEIE bit =1. 0: No Overrun error 1: Overrun error is detected 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	Noise error flag This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No noise is detected

Bit	Name	R/W	Reset Value	Function
				1: Noise is detected Note: This bit does not generate interrupt as it appears at the same time as the RXNE bit which itself generates an interrupting. Interrupt is generated on NE flag in case of MultiBuffer communication if the EIE bit is set.
1	FE	R	0	Framing error This bit is set by hardware when a de-synchronization, excessive noise or abort characters is detected. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No frame error detected 1: Frame error or break character detected Note: When RXNE and FE are generated at the same time, no interrupt is generated when FE = 1, but an interrupt is generated 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. Interrupt is generated on FE flag in case of MultiBuffer communication if the EIE bit is set.
0	PE	R	0	Parity error This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). The software must wait for the RXNE flag to be set before clearing the PE bit. An interrupt is generated if the PEIE bit =1. 0: No parity error 1: Parity error

28.5.2. USART data register (USART_DR)

Address offset: 0x04

Reset value: undefined

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	-	-	Reserved
8：0	DR[8:0]	RW	undefined	Data value Contains the Received or Transmitted data character, depending on whether it is read from or written to. 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). The TDR register provides the parallel interface between the internal bus and the output shift register. The RDR register provides the parallel interface between the input shift register and the internal bus. When transmitting with the parity enabled, the value written in the MSB (bit 7 or bit 8 depending on the data length) has no effect because it is replaced by the parity. When receiving with the parity enabled, the value read in the MSB bit is the received parity bit.

28.5.3. Baud rate register (USART_BRR)

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
DIV_Mantissa[11:0]												DIV_Faction[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

In automatic baud rate detection mode, the hardware updates this register.

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：4	DIV_Mantissa[15:4]	RW	0	12-bit integer
3：0	DIV_Faction[3:0]	RW	0	4bit decimal

28.5.4. USART control register 1 (USART_CR1)

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	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	Res
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31：14	Reserved	-	-	Reserved
13	UE	RW	0	USART enable When this bit is cleared the USART prescalers are stopped. This bit is set and cleared by software. 0: USART prescaler and output disabled, low-power mode 1: USART enabled The software needs to wait for USART _ ISR.TC to be set before it can clear the UE bit and enter the low power mode;
12	M	RW	0	0: 1 start bit, 8 data bits, n stop bit 1: 1 start bit, 9 data bit, n stop bit
11	WAKE	RW	0	Receive wakeup mode. Wake up mode from mute mode. Set and cleared by software. 0: Idle line rouse 1: Address Mark
10	PCE	RW	0	Parity control enable 0: Parity control disabled 1: Parity control enabled Parity bit: 9th bit of 9bit; The 8th bit of 8bit.
9	PS	RW	0	Parity selection Set and cleared by software. 0: Even check 1: odd check
8	PEIE	RW	0	PE interrupt enable Set and cleared by software. 0: Disabled 1：PE interrupt enabled
7	TXEIE	RW	0	TXE interrupt enable Set and cleared by soft-ware. 0: Disabled 1：TXE interrupt enabled
6	TCIE	RW	0	Transmission complete interrupt enable Set and cleared by software.

Bit	Name	R/W	Reset Value	Function
				0: Disabled 1： TC interrupt enabled
5	RXNEIE	RW	0	RXNE interrupt enable; Set and zeroed by software. 0: Disabled 1： ORE or RXNE interrupt enabled
4	IDLEIE	RW	0	IDLE interrupt enable Set and cleared by software. 0: Disabled 1： IDLE interrupt enabled
3	TE	RW	0	Transmitter enable 0: Transmitter is disabled 1: Transmitter is enabled
2	RE	RW	0	Receiver enable 0: Receiver is disabled 1: Receiver is enabled and begins searching for a start bit
1	RWU	RW	0	Receiver wakeup This bit indicates if the USART is in mute mode. When a mute mode sequence is received, the register is set; If a wake-up sequence is received, the register is cleared. The specific wake-up sequence (address or IDLE) is controlled by the register USART _ CR1. WAKEbit. 0: Receiver in active mode 1: Receiver in mute mode Note1: Before selecting Mute mode, the USART must first receive a data byte, otherwise it cannot function in Mute mode with wakeup by Idle line detection. Note: 2: In Address Mark Detection wakeup configuration (WAKE bit=1) the RWU bit cannot be modified by software while the RXNE bit is set.
0	SBK	RW	0	Send a break frame. The software sets this register and sends the break byte. After the stop bit of the Break frame is sent, the hardware clears this register. 0: No break byte is transmitted 1: Send break byte

28.5.5. USART control register 2 (USART_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.	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.	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	Res	ADD[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	LINEN	RW	0	LIN mode enabled. Software sets and clears. 0: LIN mode disabled; 1: LIN mode enabled; In LIN mode, LIN synchronization breaks (13 low bits) are sent by enabling the SBK bit.

Bit	Name	R/W	Reset Value	Function
				Note: This register is fixed to 0 if the LIN function is not supported.

13:12	STOP[1:0]	RW	2'b0	Stop bits
				00: 1 stop bit;
				01: 0.5 stop;
				10: 2 Stop bits
				11: 1.5 stop;
				Note: USART1, USART2 and USART3 only support 1stop bit and 2stop bit, i.e. only bit13 is valid and bit12 is fixed to 0.

11	CLKEN	RW	0	Clock enable 0: Interrupt is inhibited 1: CK pin enable; This bit is reserved when synchronous mode is not supported.
10	CPOL	RW	0	Clock polarity Synchronous mode, CK pin outputs clock polarity. 0: Outside the transmission window, CK pin is a stable low value; 1: Outside the transmission window, CK pin is a stable high value;
9	CPHA	RW	0	This bit is used in synchronous mode to select the phase of the CK pin output clock. It works in conjunction with the CPOL bit to produce the desired clock/data relationship. 0: the first clock transmission is the first data capture edge; 1: The second clock transmission is the first data capture edge;
8	LBCL	RW	0	Whether the clock pulse of the last bit of data is output at 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;
7	Reserved	-	-	Reserved
6	LBDIE	RW	0	LIN break interrupt enabled. 0: Interrupt is inhibited 1: Interrupt generation Note: This register is fixed to 0 if the LIN function is not supported.

5	LBDL	RW	0	LIN break detects length.
				0: 10 bits detection break;
				1: 11 bits detection break;
				Note: This register is fixed to 0 if the LIN function is not supported.

4	Reserved	-	-	Reserved
3:0	ADD[3:0]	RW	4'b0	Address of the USART node This is used in multiprocessor communication during mute mode, for a 4-bit wake up with address mark detection.

28.5.6. USART control register 3 (USART_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.	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.	ABR-MOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
	RW		RW	RW	RW	RW	RW	RW	RW			RW			RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14：13	ABRMOD[1:0]	RW	2'b0	Auto baud rate mode 00: Measure baud rate from start bit 01: Falling edge to falling edge measurement This bitfield can only be written when ABREN = 0 or the USART is disabled (UE=0).
12	ABREN	RW	0	Auto baud rate enable 0: Disabled 1: Auto baud rate detection is enabled
11	OVER8	RW	0	Oversampling mode. Oversampling by 16 Oversampling by 8 This bit is only writeable when UE = 0.
10	CTSIE	RW	0	CTS interrupt enable 0: Interrupt is inhibited 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.
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.
7	DMAT	RW	0	DMA is enabled when transmitting. 0: Interrupt is inhibited 1: DMA enabled during transmission
6	DMAR	RW	0	DMA is enabled on receive. 0: Interrupt is inhibited 1: DMA enabled on reception
5:4	Reserved	-	-	Reserved
3	HDSEL	RW	0	Half-duplex selection 0: Half duplex mode is not selected 1: Half duplex mode is selected
2:1	Reserved	-	-	Reserved
0	EIE	RW	0	Error interrupt enable 0: Interrupt is inhibited 1: Frame error FE, overrun error ORE, noise NF interrupt enable.

29. Serial peripheral interface/I2S (SPI/I2S)

Two SPI/I2S modules are designed and implemented in this project. The SPI1 module includes I2S and SPI CRC functions, while the SPI2 module does not include I2S and SPI CRC functions.

29.1. Introduction

The SPI/I2S 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.

The serial peripheral interface (SPI) protocol supports half-duplex, full-duplex and simplex synchronous, serial communication with external devices. The interface can be configured as master and in this case it provides the communication 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 using 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 supplies a clock signal to an external slave device through an interface.

29.2. Main features

29.2.1. SPI main features

- Master or Slave mode
- 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-bit or 16-bit transmission frame selection
- Support multi-master mode
- 8 Master mode baud rate prescaling factors (Max fPCLK/2)
- Slave mode frequency (Max f_{PCLK}/4)
- 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.

- Motorola mode
- Can trigger interrupt-causing Master mode faults, overrun errors and CRC errors
- Two embedded Rx and Tx FIFOs with DMA capability, depth of four, and width of 16 bits (8 bits when data frame is set to 8 bits)

29.2.2. I2S main features

- Half-duplex communication (only transmitter or receiver)
- Master or slave operations
- 8-bit programmable linear prescaler to reach accurate audio sample frequencies (from 8 KHz to 96 kHz)
- Data format may be 16-bit, 24-bit or 32-bit
- Packet frame is fixed to 16-bit (16-bit data frame) or 32-bit (16-bit, 24-bit, 32-bit data frame) by audio channel
- Programmable clock polarity (steady state)
- Underflow flag bits in slave transmit mode and overflow flag bits in master/slave receive mode
- 16-bit register for transmission and reception with one data register for both channel sides
- Support I2S protocols:
  - I2S Phillps standard
  - MSB-justified standard (left-justified)
  - LSB-justified standard (right-justified)
  - PCM standard (with short and long frame synchronization on 16-bit channel frame or 16-bit data frame extended 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)

29.3. SPI functional description

29.3.1. Overview

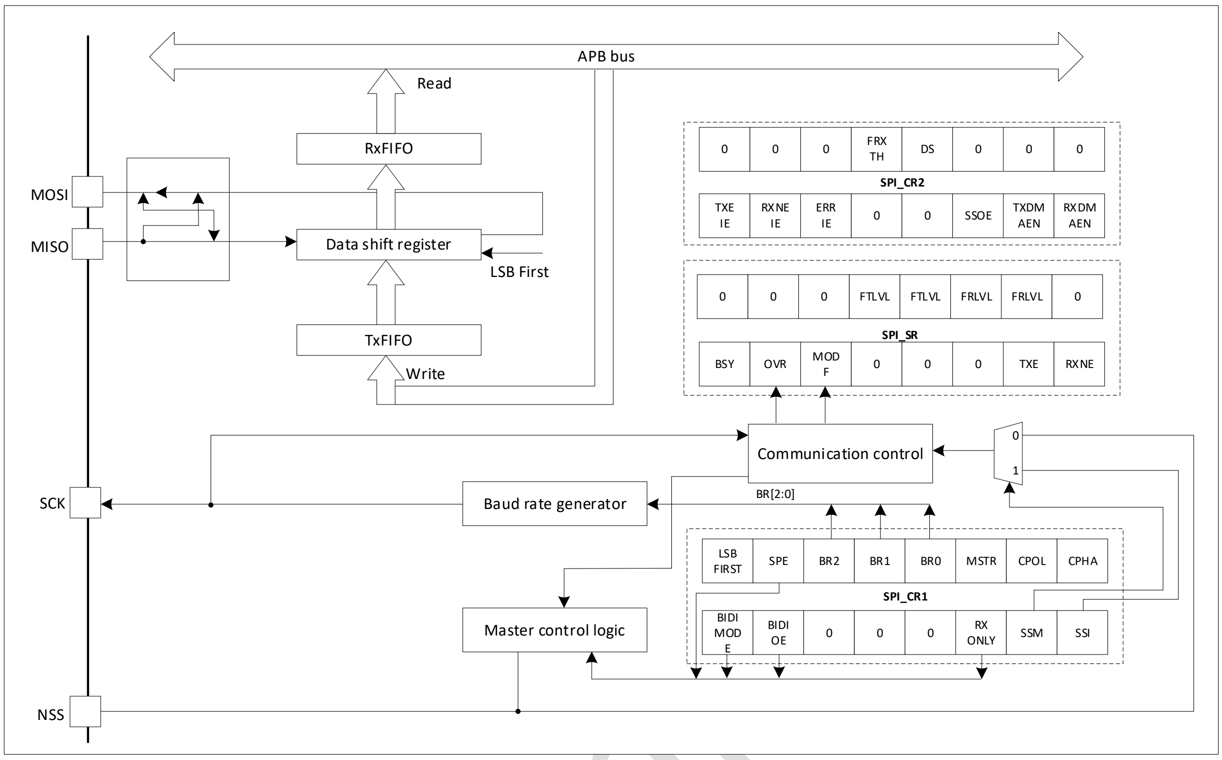


Figure 29-1 SPI block diagram

Four I/O pins are dedicated to SPI communication with external devices.

MISO: Master In / Slave Out data. In the general case, this pin is used to transmit data in slave mode and receive data in master mode.

MOSI: Master Out / Slave In data. In the general case, this pin is used to transmit data in master mode and receive data in slave mode.

SCK: Serial Clock output pin for SPI masters and input pin for SPI slaves.

NSS: Slave select pin. Depending on the SPI and NSS settings, this pin can be used to either:

─ Select an individual slave device for communication

─ Synchronize the data frame

─ Detect a conflict between multiple masters

The SPI bus allows the communication between one master device and one or more slave devices. The bus consists of at least two wires: one for the clock signal and the other for synchronous data transfer. Other signals can be added depending on the data exchange between SPI nodes and their slave select signal management.

29.3.2. 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-wire, 3-wire (software NSS management), or 4-wire (hardware NSS management). Communication is usually initiated by the host computer.

29.3.2.1. 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 connected together using two unidirectional lines between the MOSI and the MISO. 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 data frame transmission is completed (all bits are shifted), the information between the master and slave is exchanged.

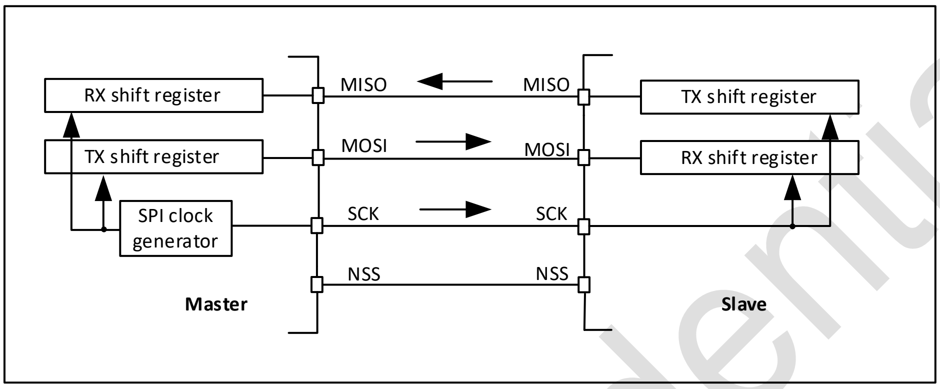


Figure 29-2 Full-duplex single master/ single slave application

29.3.2.2. Half-duplex communication

The SPI can communicate in half-duplex mode by setting the BIDIMODE bit in the SPI_CR1 register. In this configuration, the master and slave shift registers are connected with 1 data cable. 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 pin of master and the MOSI pin of slave are released as GPIO for other applications to use.

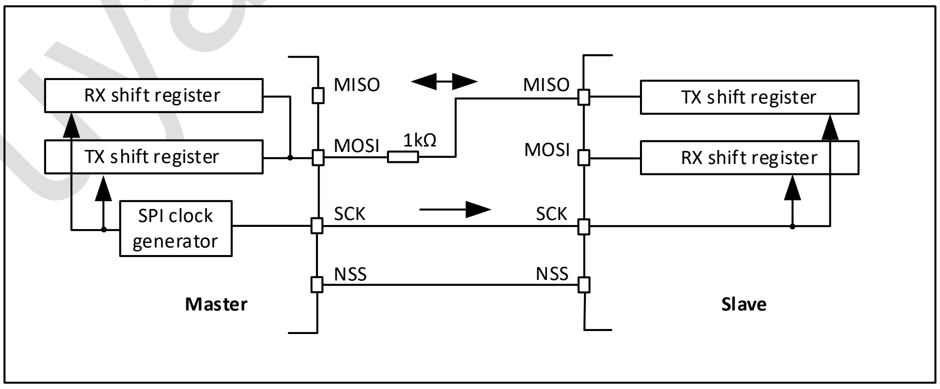


Figure 29-3 Half-duplex single master/ single slave application

The NSS pin may be used for hardware control flow between master and slave. Optionally, NSS may not be used. Then the flow has to be handled internally.

In this configuration, the MISO pin of master and the MOSI pin of slave can be used as the GPIO.

29.3.2.3. Simplex communications

By using RXONLY (SPI _ CR2 register), the SPI is set to transmit-only or receive-only, so that the SPI works in simplex mode. With this configuration, only 1 wire is used between the shift registers of master and slave. The remaining MISO and MOSI pins pair is not used for communication and can be used as standard GPIOs.

- Transmit-only mode (RXONLY=0): The configuration settings are the same as for full- duplex. The application has to ignore the information captured on the unused input pin. This pin can be used as a standard GPIO.
- Receive-only mode (RXONLY=1): The application can disable the SPI output functionby setting the RXONLY bit.
  - In the slave mode configuration, the MISO output is disabled, and this pin is used as the GPIO. When his Slave select signal is active, Slave continues to receive data from the MOSI pin. Received data events appear depending on the data buffer configuration.
  - In the master mode configuration, the MOSI output is disabled and the pin can be used as the 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, wait until the input pattern from the MISO pin is complete, and fill the data buffer.

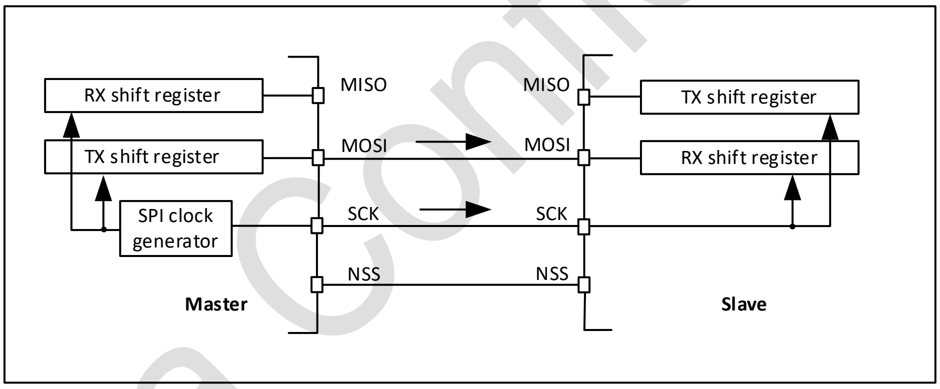


Figure 29-4 Single master/ single slave application (master in transmit-only/slave in receive-only mode)

(1) The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, NSS may not be used. Then the flow has to be handled internally.

(2) An accidental input information is captured at the input of transmitter Rx shift register. In the standard transmit-only mode, all events related to transmission reception must be ignored.

(3) In this configuration, both the MISO pins can be used as GPIOs.

Any simplex communication can be alternatively replaced by a variant of the half-duplex communication with a constant setting of the transaction direction (bidirectional mode is enabled while BIDIOE bit is not changed).

29.3.3. Multi-slave communication

In a configuration with two or more independent slaves, the master uses GPIO to manage NSS for each slave. The master must select one of the slaves individually by pulling low the GPIO connected to the slave NSS input. When this is done, a standard master and dedicated slave communication is established.

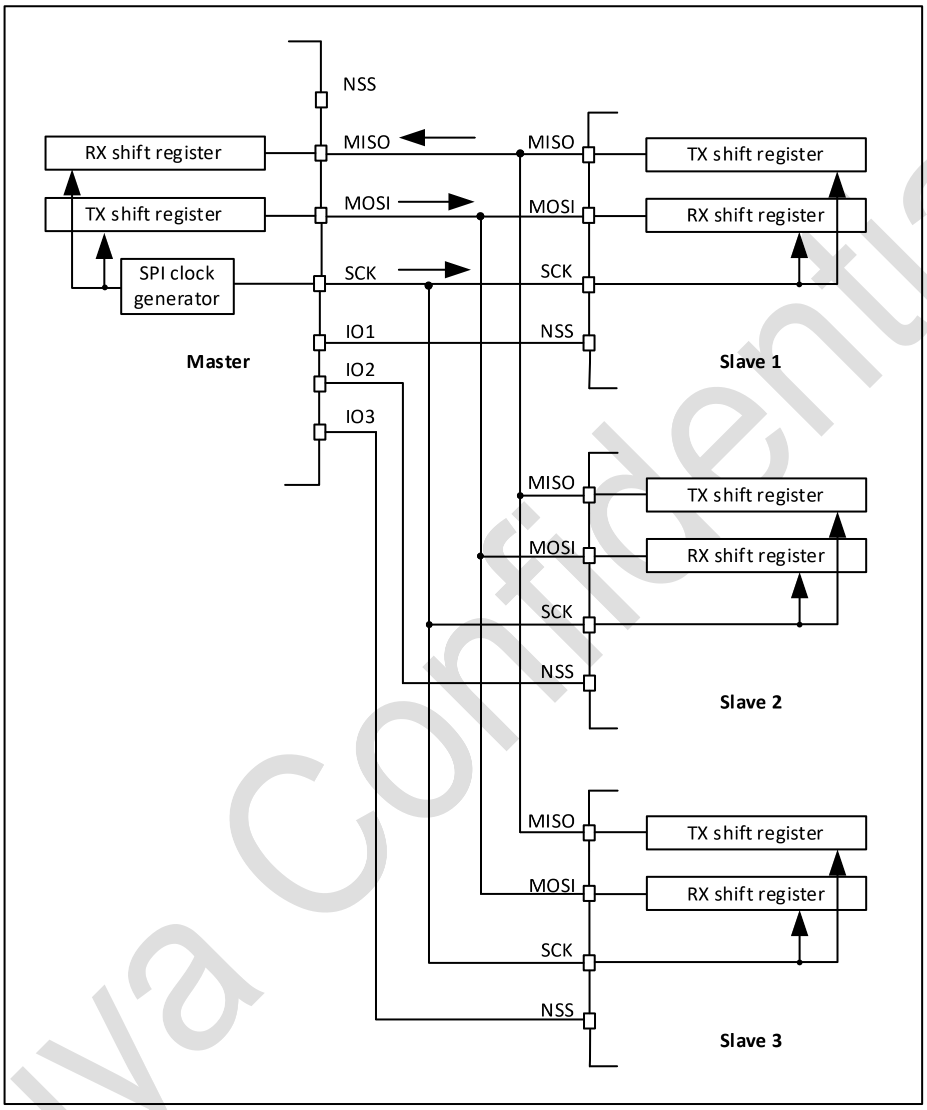


Figure 29-5 Master and three independent slaves

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.

29.3.4. Multi-master communication

Unless SPI bus is not designed for a multi-master capability primarily, the user can use build in feature which detects a potential conflict between two nodes trying to master the bus at the same time. 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 one node wants to overtake control on the bus, it switches itself into master mode and applies active level on the slave select input of the other node via dedicated GPIO pin. After the process is completed, the valid slave select signal is released, and the node controlling the bus temporarily returns to passive mode and waits for a new process to start.

If potentially both nodes raised their mastering request at the same time a bus conflict event appears (see mode fault MODF event). The user can then apply some simple arbitration procedure (e.g. postpone the next attempt by different timeouts defined in advance for the two nodes)

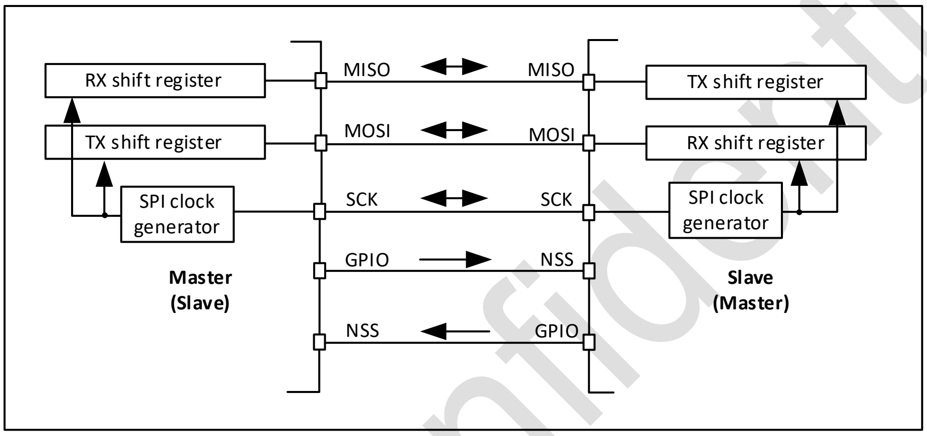


Figure 29-6 Multi-host application

The NSS is configured in a hardware input mode at both nodes. His active level enables MISO output control, while the passive node is configured as a slave.

29.3.5. 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. As an input it can prevent multimaster bus collision, and as an output it can drive a slave select signal of a single slave.

Hardware or software slave select management can be set using the SSM bit in the SPI_CR1 register:

- 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 disable (SSM=0, SSOE = 0): if the microcontroller is acting as the master on the bus, this configuration allows multimaster capability. If the NSS pin is pulled low in this mode, the SPI enters master mode fault state and the device is automatically reconfigured in 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.

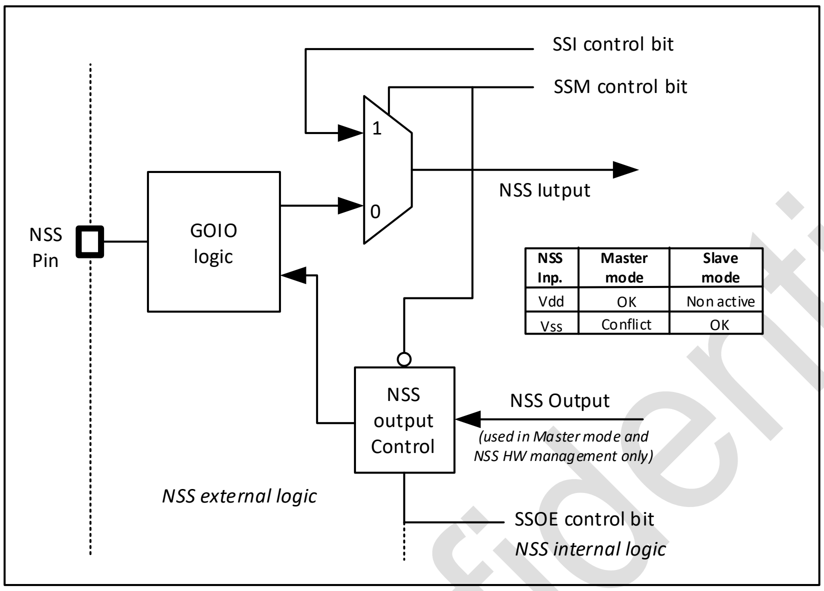


Figure 29-7 Hardware/software slave select management

29.3.6. Communication formats

During SPI communication, receive and transmit operations are performed simultaneously SCK (serial clock) synchronizes the information shift and sampling operation on the data line. 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.

29.3.6.1. Clock phase and polarity controls

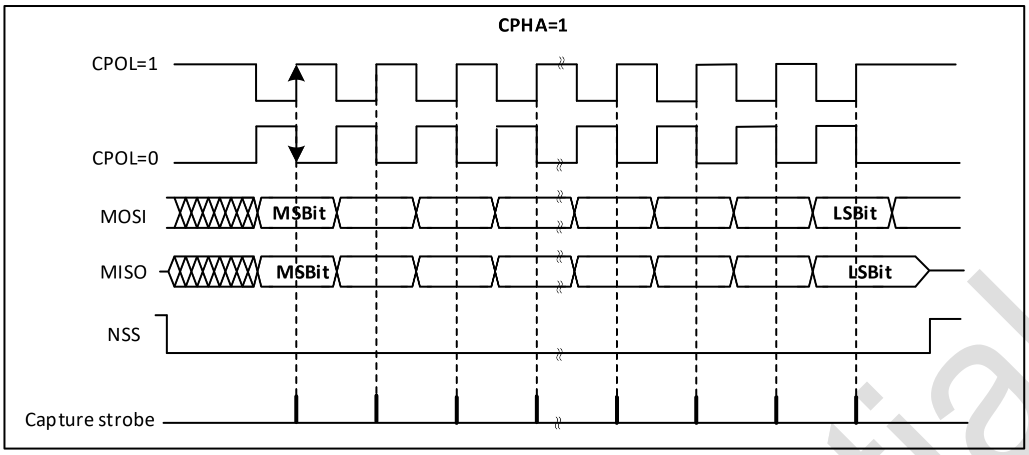
Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits in the SPI_CR1 register. CPOL (clock polarity) controls the IDLE state of the clock when there is no data transmission. This bit affects both master and slave modes. If CPOL is reset, the SCK pin has a low-level idle state. If CPOL is set, the SCK pin has a high-level idle state.

If the CPHA bit is set, the second edge on the SCK pin captures the first data bit transacted (falling edge if the CPOL bit is reset, rising edge if the CPOL bit is set). Data are latched on each occurrence of this clock transition type. If the CPHA bit is reset, the first edge on the SCK pin captures the first data bit transacted (falling edge if the CPOL bit is set, rising edge if the CPOL bit is reset). Data are latched on each occurrence of this clock transition type.

The combination of CPOL (clock polarity) and CPHA (clock phase) bits selects the data capture clock edge.

Prior to changing the CPOL/CPHA bits the SPI must be disabled by resetting the SPE bit.

The idle state of SCK must correspond to the polarity selected in the SPI_CR1 register.



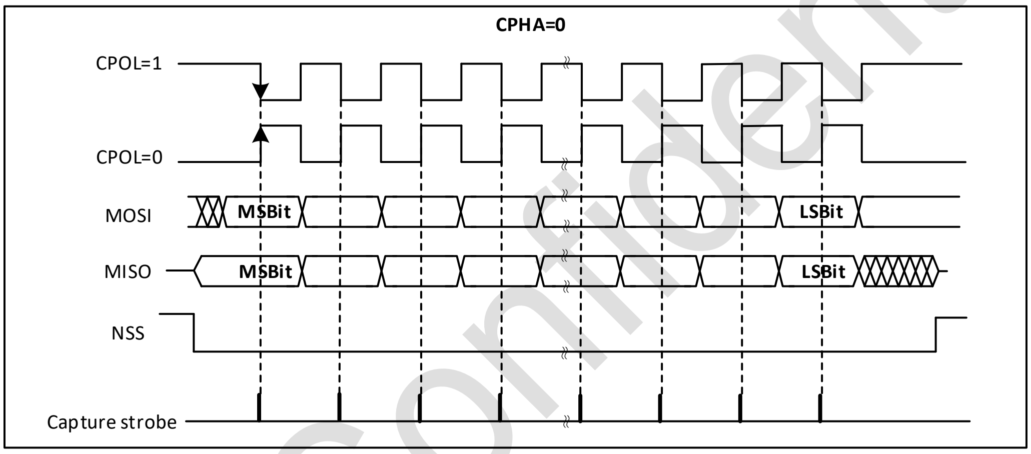


Figure 29-8 Data clock timing diagram

The order of data bits depends on LSBFIRST bit setting.

29.3.6.2. Data frame format

The SPI shift register can be set up to shift out MSB-first or LSB-first, depending on the value of the LSBFIRST bit in SPI_CR1 register. The data frame size is chosen by using the DS bit in SPI_CR2 register. It can be set 8-bit or 16-bit length and the setting applies for both transmission and reception.

29.3.7. Configuration of SPI

The configuration procedure is almost the same for master and slave. For specific pattern establishment, follow the special chapter introduction. When a standard communication is to be initialized, perform these steps:

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:

1) Configure the serial clock baud rate using the BR[2:0] bits (not required in slave mode)
2) Configure the CPOL and CPHA bits

3) Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE (RXONLY and BIDIMODE cannot be valid at the same time).
4) Configure the LSBFIRST bit
5) Configure SSM and SSI
6) Configure the MSTR bit (in multimaster NSS configuration, avoid conflict state on NSS if master is configured to prevent MODF error).

1. 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)
3) Configuration FRXTH bit. The RXFIFO threshold must be aligned to the read access size for the SPI_DR register.

2. Write the corresponding DMA register: Configure the SPI Tx and Rx channels of the DMA

29.3.8. Procedure for enabling SPI

It is recommended to enable the SPI slave before the master sends the clock. If not, undesired data transmission might occur. The slave's data register must already contain the data to be sent before starting communication with the master (either at the first edge of the communication clock or, if the clock signal is continuous, before the end of the ongoing communication). The SCK signal must be fixed at the IDLE state level (corresponding selected polarity) before the SPI slave is enabled.

Full-duplex mode (or transmit-only), when SPI is enabled and TXFIFO is not empty, or the next write is made to TXFIFO, the host starts communication.

In any master receive only mode (RXONLY=1 or BIDIMODE=1 & BIDIOE=0), master starts to communicate, and the clock starts running immediately after SPI is enabled.

For DMA processing, follow the specific sections.

29.3.9. Data transmission and reception procedures

29.3.9.1. RXFIFO and TXFIFO

SPI All data traffic passes through a FIFO with a depth of 4 and a width of 16 bits (8 bits when the data frame is set to 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.

FIFO processing depends on a variety of parameters, including: data exchange mode (full duplex, half duplex), 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 generally be aligned with the RXFIFO threshold, and the FTLVL [1: 0] and FRLVL [1: 0] bits show the current occupancy level of both 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. When RXNE is cleared, RXFIFO is considered to be empty.

In a similar way, write access of a data frame to be transmitted is managed by the TXE event. 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, both RXFIFO and TXFIFO can store 4 data frames

Both TXE and RXNE events can be handled by query, interrupt, and DMA.

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 BSY bit being set indicates ongoing transaction of a current data frame. When the clock signal runs continuously, the BSY flag stays set between data frames at master. But becomes low for a minimum duration of one SPI clock at slave between each data frame transfer.

In some application scenarios, when writing data to TXFIFO, you can clear the TXFIFO data by setting the CLRTXFIFO bit, so as to write new data to TXFIFO again and communicate again.

29.3.9.2. Sequence handling

Some data frames can be passed through single sequence to complete a piece of information. When sending is enabled and there is any data in the TXFIFO of the maser, the sequence starts and continues. The clock signal is provided continuously by the master until TXFIFO becomes empty, then it stops waiting for additional data.

In receive-only mode, i.e. half-duplex (BIDIMODE = 1, BIDIOE = 0) or simplex mode (BIDIMODE = 0, RXONLY = 1), the master starts transmitting immediately when SPI is enabled and receive-only mode is activated. The master will always provide the clock until the master stops SPI or receive-only mode. Master receives data continuously.

While the master can provide all the transactions in continuous mode (SCK signal is continuous) it has to respect slave capability to handle data flow and its content at anytime. When necessary, the master must slow down the communication and provide either a slower clock or separate frames or data sessions with sufficient delays. It should be noted that there is no underflow error signal for master or slave, and the data from slave is usually interacted with and processed by master (even if slave cannot prepare the data in time). For slave, a better method is to use DMA, especially when the data frame is small and the baud rate is high.

Each sequence must be enclosed by an NSS pulse, and one of the slaves to communicate with is selected in a multi-slave system. In a single slave system, there is no need to use NSS to control the slave.

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 just sampled, and the complete data frame is stored in the RXFIFO.

29.3.9.3. To disable SPI

When SPI is disabled, a specific disabling process must be followed. It is important to do this before the system enters a low-power mode when the peripheral clock is stopped. Ongoing transactions can be corrupted in this case. In some modes the disable procedure is the only way to stop continuous communication running.

In full duplex or transmit-only mode, the host can complete the interaction when it stops providing the data to be sent. In this case, the clock stops after the last data transaction. Pay extra attention to the packing mode (when interacting with an odd number of data frames to place some dummy bytes interacting). In these modes, the user must use the standard disable process before the SPI can be disabled. When the SPI is disabled at the master transmitter while a frame transaction is ongoing or next data frame is stored in TXFIFO, the SPI behavior is not guaranteed.

When the master is in any receive only mode, the only way to stop the continuous clock is to disable the peripheral by SPE=0. Specific procedure must be followed when disabling SPI in this mode.

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 unread data, make sure that the RXFIFO is empty when the SPI is disabled (using the correct disable process, or by resetting it with software to initialize all SPI registers).

Standard disable procedure is based on pulling BSY status together with FTLVL[1:0] to check if a transmission session is fully completed. This check can be done in specific cases, too, when it is necessary to identify the end of ongoing transactions, for example:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave. Or
- When the interactive data stream from the FIFO is completed, the last data frame or CRC frame is still in transmission.

The correct disable process is (except in receive-only mode):

1. Wait for FTLVL [1: 0] = 00 (no data to send)
2. Wait until BSY=0 (the last data frame is processed)
3. Disable SPI (SPE = 0)
4. Read data until FRLVL[1:0] = 00 (read all the received data)

The correct disable process for a specific receive-only mode 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[1:0] = 00 (read all the received data)

29.3.9.4. DMA transmission

In order to work at maximum speed and speed up the read/write process of data to avoid overrun, SPI has the DMA capability of simple request/response protocol.

When TXE or RXNE is set, a DMA request is generated. The Tx and Rx buffer have separate requests.

- When sending, each time TXE is set to 1, a DMA request is generated. The DMA then writes data to the SPI _ DR register.
- Upon reception, each time RXNE is set to 1, a DMA request is generated. The DMA then reads the data of the SPI _ DR register.

When the SPI is used only to transmit data, it is possible to enable only the SPI Tx DMA channel. In this case, the OVR flag bit is set because the received data is not read away. When the SPI is used only to receive data, it is possible to enable only the SPI Rx DMA channel.

At the time of transmission, when the DMA has written all the data to be transmitted (the TCIF flag bit of the DMA _ ISR register is set), you can monitor the BSY flag to ensure that the SPI communication is complete. This is used to prevent the final transmission from being corrupted when the SPI is disabled or enters stop mode. The software must first wait until FTLVL[1:0]=00 and then until BSY=0.

When you start using DMA communication, to prevent error events in DMA channel management, you must follow the following steps:

1. Enable DMA Rx buffer (RXDMAEN bit of SPI _ CR2) if Rx DMA is used
2. Enable Tx Rx DMA streams (in DMA register) (if steams are used)
3. Enable DMA Tx buffer (at TXDMAEN bit of SPI _ CR2 register) (if Tx DMA is used)
4. Enable SPI via SPE bit

Force closure of communication with the following steps:

1. Disable DMA Tx Rx streams (in DMA register) (if stream is used)
2. disable SPI via SPI disable process

disable DMA Tx and Rx buffer (if DMA Tx and/or Rx are used) by clearing TXDMAEN and RXDMAEN (SPI _ CR2 registers)

29.3.9.5. Communication diagrams

This section introduces some typical timings that are valid for queries, interrupts, or DMA. For simplicity, the LSBFIRST=0, CPOL=0 and CPHA=1 setting is used as a common assumption here. Full DMA operation configuration is also not available.

1. When NSS is valid, SPI is enabled, and Slave starts to control MISO, (when SPI is disabled or NSS is invalid, Slave no longer controls MSIO). For slave, sufficient time must be provided for the master to prepare data in advance before the transmission begins.
2. 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 SPI is disabled the SPI peripheral is disconnected from GPIO logic, so the levels at these lines depends on GPIO setting exclusively.
3. 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.
4. The TXE signal is cleared only if TXFIFO is full.
5. As soon as the TXDMAEN bit is set, the DMA arbitration process begins. The TXE interrupt is generated just after the TXEIE is set. When the TXE signal is active, data transmission to the TxFIFO is started until the TxFIFO becomes full, or the DMA transmission is complete.
6. If all the data to be sent is loaded into the TxFIFO, the DMA Tx TCIF flag goes high even before the SPI communication. This flag is always high until the SPI interaction is completed.

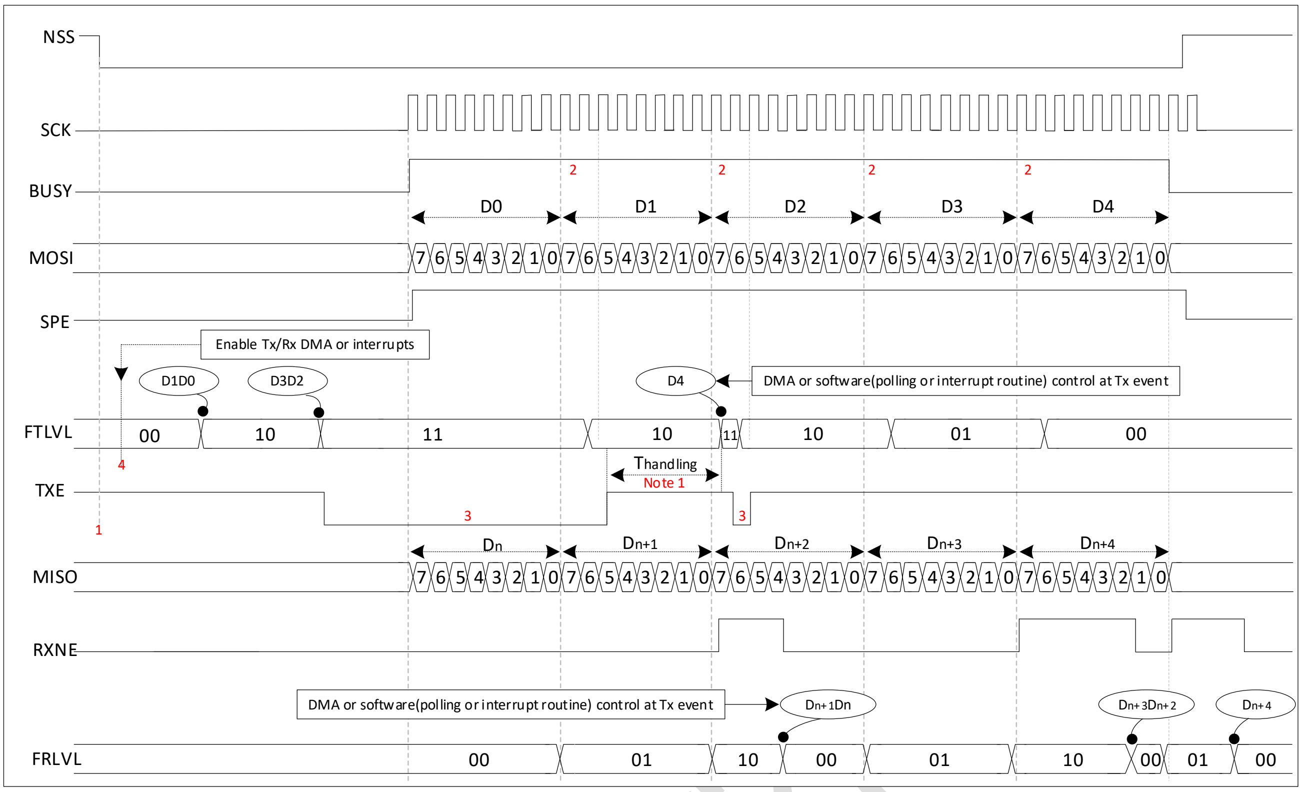

NSS
SCK
BUSY
MOSI
SPE
FTLVL
TXE
MISO
RXNE
FRLVL
D0
D1
D2
D3
D4
Enable Tx/Rx DMA or interrupts
D1D0
D3D2
D4
DMA or software(polling or interrupt routine) control at Tx event
Thandling
Note 1
Dn
Dn+1
Dn+2
Dn+3
Dn+4
DMA or software(polling or interrupt routine) control at Tx event
Dn+1Dn
Dn+3Dn+2
Dn+4

Figure 29-9 Master full-duplex communication (bit frame=8)

29.3.10. Status flags

The application can fully monitor the status of the SPI bus through 3 status flags.

29.3.10.1. Send buffer 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. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

29.3.10.2. Rx buffer not empty flag (RXNE)

The RXNE flag is set depending on the FRXTH bit value in the SPI_CR2 register.

The RXNE is cleared by hardware automatically when the above conditions are no longer true.

29.3.10.3. Busy flag (BSY)

The BSY flag is set and cleared by hardware (writing to this flag has no effect).

When it is set to '1', it indicates that the SPI is busy communicating with one exception: in the bidirectional reception mode of the master mode (MSTR = 1, BDM = 1, and BDOE = 0), the BSY flag remains 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 or its peripheral clock before entering a low-power mode which does not provide a clock for the peripheral. This avoids corrupting the last transfer.

The BSY flag is also useful for preventing write collisions in a multimaster system.

Except for the bidirectional reception mode of the master mode (MSTR = 1, BDM = 1, and BDOE = 0), the BSY flag is set to '1' when the transmission starts.

The BSY flag is cleared under any one of the following conditions:

● When SPI is correctly disabled

● When a fault is detected in Master mode (MODF bit set to 1)

● In Master mode, when it finishes a data transmission and no new data is ready to be sent

● 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 to handle each data send and receive. It is more appropriate to use TXE and RXNE.

29.3.11. Error flags

29.3.11.1. Mode fault (MODF)

Mode fault occurs when the master device has its internal NSS signal (NSS pin in NSS hardware mode) pulled low. 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.

Normally, the MODF bit of the slave device cannot be set to '1'. However, in a multi-master configuration, a device may be in slave mode with the MODF bit set; At this point, the MODF bit indicates that a multi-master conflict may have occurred. The interrupt program may perform a reset or return to the default state to recover from the error state.

29.3.11.2. Overload mode

An overrun condition occurs when data is received by a master or slave and the RXFIFO has not enough space to store this received data. This can happen if the software did not have enough time to read the previously received data (stored in the RXFIFO).

When over normal operation 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.

29.3.12. SPI interrupts

Table 29-1 SPI interrupt requests

Interrupt event	Event flag	Enable control bit
Transmit TXFIFO ready to be loaded	TXE	TXEIE
Data received in RXFIFO	RXNE	RXNEIE
Master Mode fault event	MODF	ERRIE
Overrun error	OVR	ERRIE

29.3.13. SPI CRC

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.

29.3.13.1. CRC Principle

Before SPI is enabled (SPE = 1), CRC calculation is enabled by setting the CRCEN bit in the SPIx _ CR1 register. CRC values are calculated using odd programmable polynomials on each bit. The calculation is processed on the sampling clock edge defined by the CPHA and CPOL bits in the SPIx_CR1 register. The calculated CRC value is automatically checked at the end of the data block and its transmission is managed by the CPU or DMA. When a mismatch is detected between the CRC calculated internally on the received data and the CRC sent by the transmitter, a CRCERR flag is set to indicate a data corruption error. The correct procedure to handle the CRC calculation depends on the SPI configuration and the transfer management mode selected.

29.3.13.2. CRC transfer controlled by CPU

Start and communicate continuously until the last data frame in the SPIx _ DR register is transmitted or received. The CRCNEXT bit must then be set in the SPIx _ CR1 register to instruct the start of transmission of the CRC frame after the currently processed data frame. The CRCNEXT bit must be set before the end of the last data frame transaction. The CRC calculation is stopped during CRC transmission.

The received CRC is stored in the RXFIFO like a data byte or word. This is why only 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. However, 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 SPIx _ RXCRC register. Software has to check the CRCERR flag in the SPIx_SR register to determine if the data transfers were corrupted or not. The software clears the CRCERR flag by writing a "0" to it. After the CRC is received, the CRC value is stored in the RXFIFO and the SPIx _ DR register must be read to clear the RXNE flag.

29.3.13.3. DMA-controlled CRC transmission

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 for the SPI transmission DMA channel has to be set to the number of data frames to transmit excluding the CRC frame. 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 reception channel counter should contain only the amount of data transferred, excluding the CRC calculation. Then a full transfer based on DMA, since it works as a single buffer in this mode, all CRC values must be read back from the FIFO by the software. At the end of data and CRC transmission, if an error occurs during the transmission, the CRCERR flag in the SPIx _ SR register is set.

29.4. I2S functional description

29.4.1. General Description

The block diagram of I2S module is as follows:

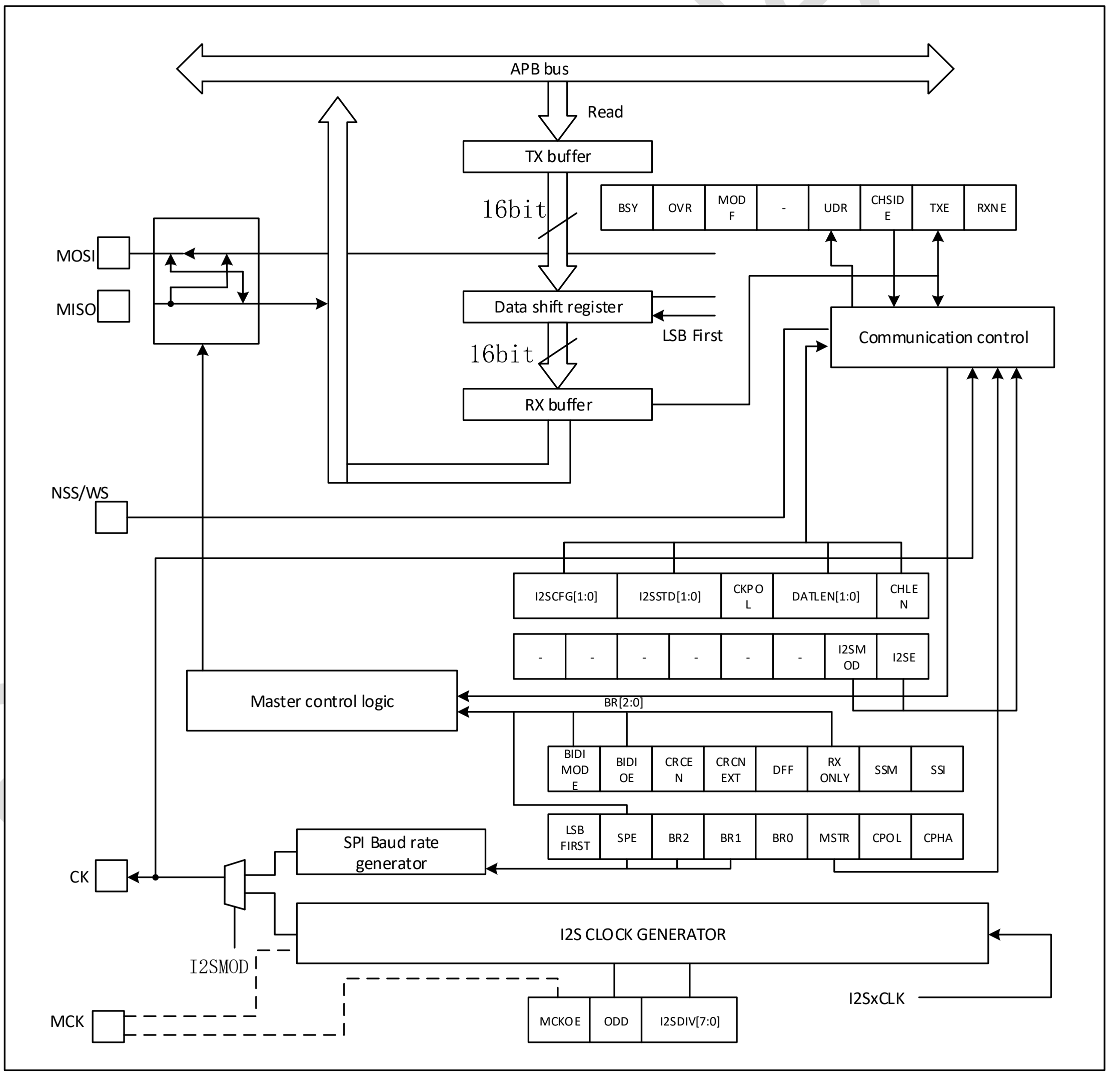


Figure 29-10 I2S block diagram

The I2S function can be enabled by setting the I2SMOD position of the register SPI _ I2SCFGR to '1'. At this time, the SPI module may be used as an I2S audio interface. 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.

When some external audio devices require a master clock, there can be an additional pin to output the clock:

MCK: Master clock (independent mapping), used as an extra clock signal pin for outputting when I2S is 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.

When the I2S is set to the master mode, the I2S generates a clock signal for communication using its own clock generator. 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 SPI_CR1 register and all CRC registers are not used in the I2S mode. Similarly, the SSOE bit of the register SPI _ CR2, and the MODF bit and the CRCERR bit of the register SPI _ SR are not used in the I2S mode.

I2S uses the same register SPI _ DR as SPI for 16-bit wide mode data transmission.

29.4.2. Supports 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 packet frames available. Data can be sent in the following four data formats:

- 16-bit data packed into 16-bit frames
- 16-bit data packed into 32-bit frames
- 24-bit data packed into 32-bit frames
- 32-bit data packed into 32-bit frames

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 require the CPU to perform two read or write operations to the register SPI _ DR, and when using DMA, two DMA transfers. For 24-bit data, after expansion to 32 bits, the lowest 8 bits are set to 0 by the hardware. For all data formats and communication standards, the highest bit (MSB) is always sent first.

The I2S interface supports four audio standards, which can be selected by setting the I2SSTD [1: 0] bit and the PCMSYNC bit of the register SPI _ I2SCFGR.

29.4.2.1. I2S Phillps standard

Under this standard, the pin WS is used to indicate which channel the data being transmitted belongs to. This pin is active 1 clock cycle before the first bit of data (MSB) is sent.

The timing diagram is as follows:

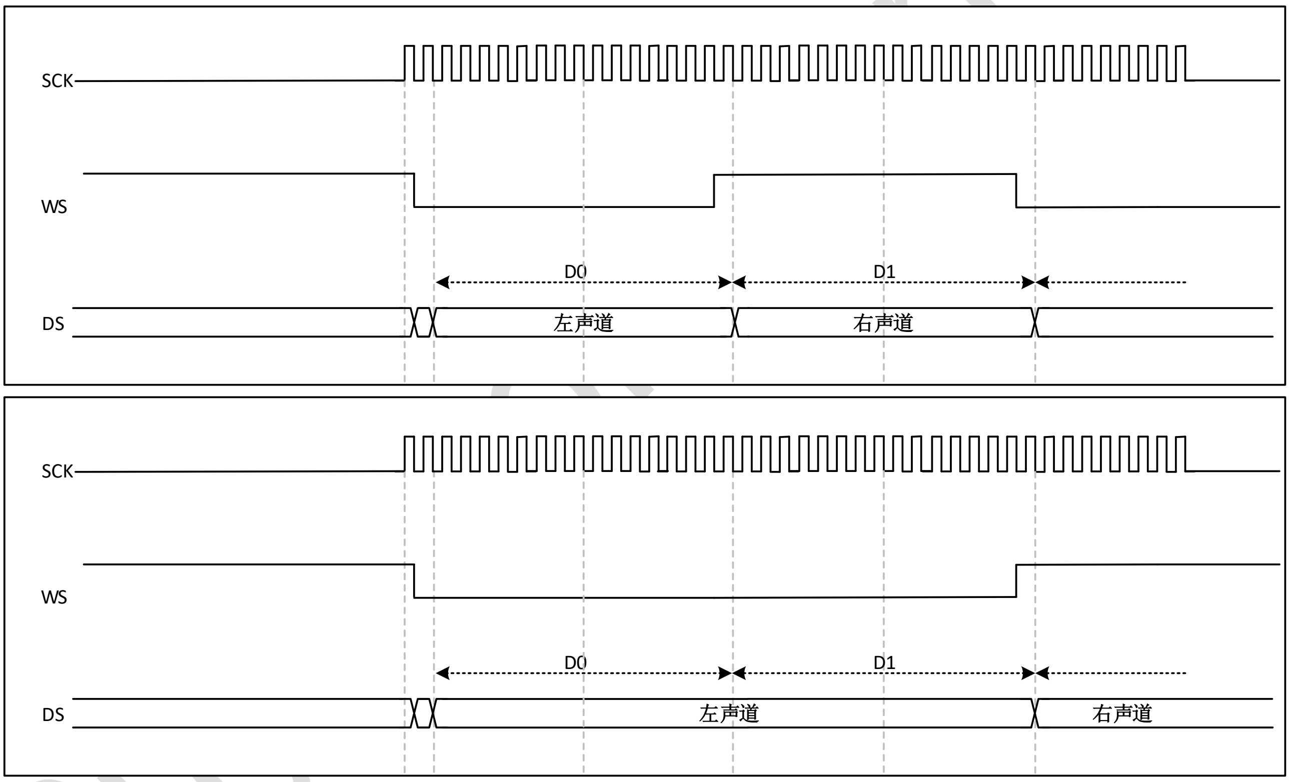
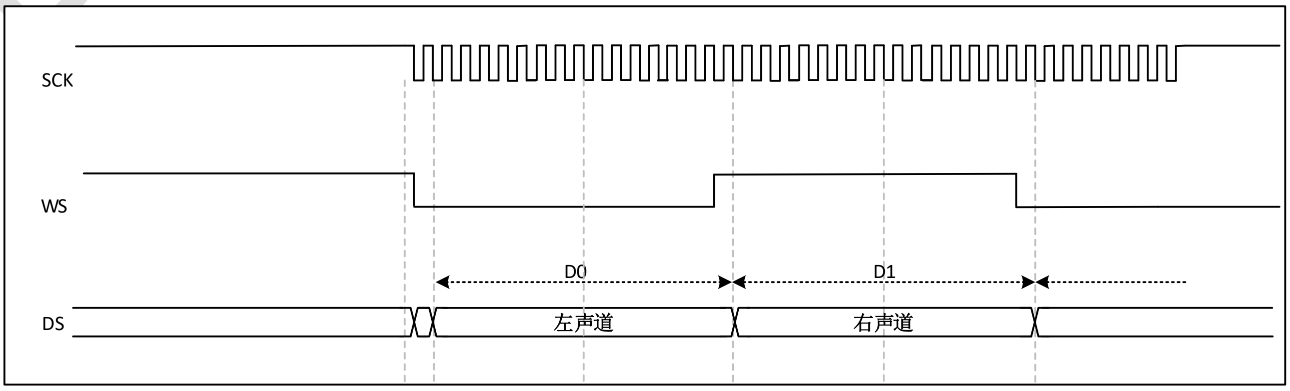


Figure 29-11 I2S Philips protocol (16/32 bit full accuracy, CPOL = 0)



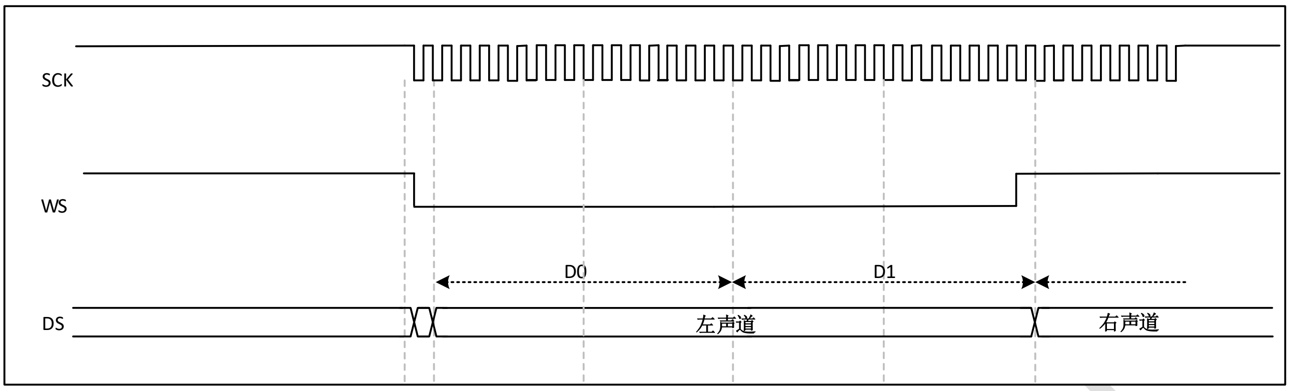


Figure 29-12 I2S Philips protocol (16/32 bit full accuracy, CPOL = 1)

Data are latched on the falling edge of CK (for the transmitter) and are read on the rising edge (for the receiver). The WS signal is also latched on the falling edge of clock signal.

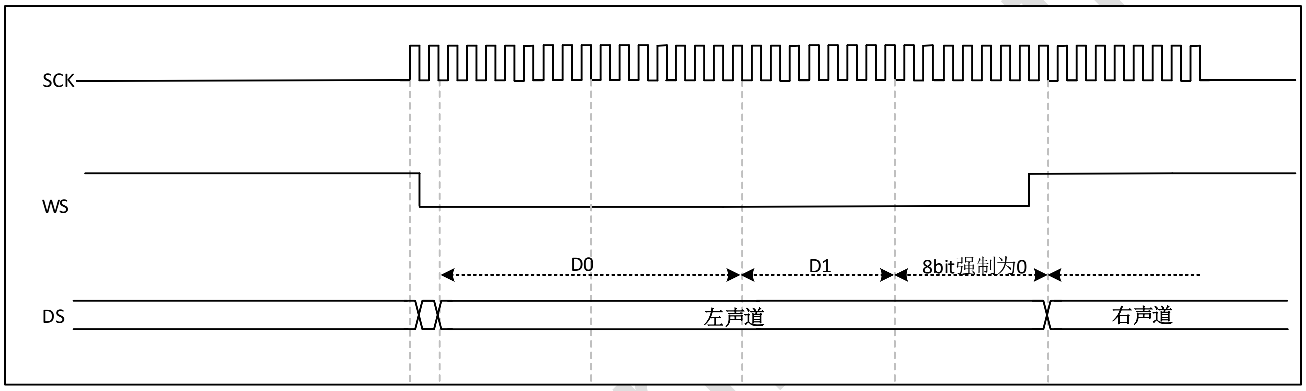


Figure 29-13 I2S Philips protocol (24-bit frame, CPOL = 0)

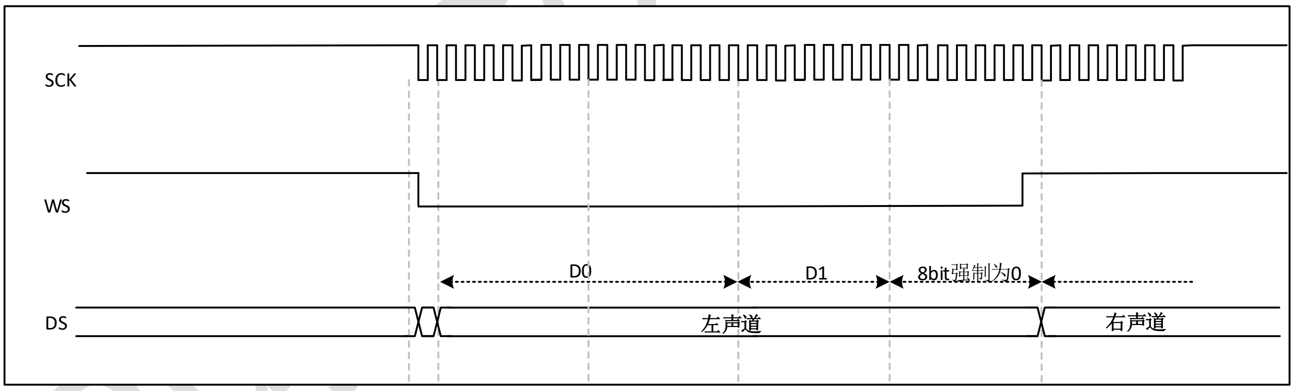


Figure 29-14 I2S Philips protocol (24-bit frame, CPOL = 1)

This mode requires two read or write operations to the register SPI _ DR.

- In send mode: If you need to send 0x8EAA33 (24 bits), first write 0x8EAA, second write 0x33xx, lower 8 bits meaningless
- In receive mode: if you receive 0x8EAA33 (24 bits), receive 0x8EAA for the first time, read 0x3300 for the second time, only the upper 8 bits are meaningful data, and the lower 8 bits are always 0

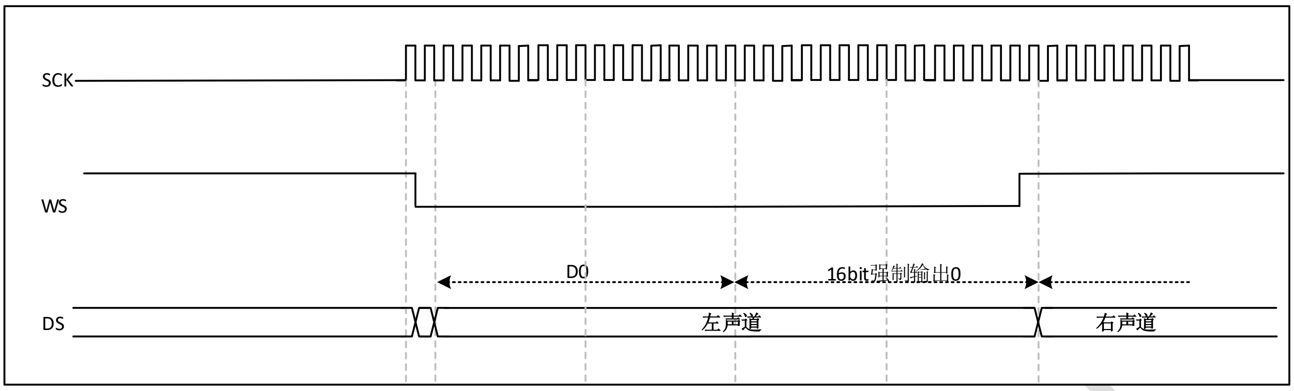


Figure 29-15 I2S Philips protocol (16-bit extended to 32-bit)

In the I2S configuration phase, if you choose to expand 16-bit data to 32-bit channel frames, you only need to access the register SPI _ DR once. The lower 16 bits used to scale to 32 bits are set to 0x0000 by hardware.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x76A30000), you only need to operate SPI _ DR once and write data 0x76A3.

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. The sending is done by hardware. Even if the last 16 bits of 0x0000 have not been sent, the TXE will be set and the corresponding interrupt will be generated. Upon reception, the flag bit RXNE is set to '1' after each upper 16-bit half-word (MSB) is received, and an interrupt can be generated if the corresponding interrupt is allowed.

In this way, there is more time between two reads and writes, and underflow or overflow can be prevented.

29.4.2.2. MSB justified standard

And the first data bit, the highest bit (MSB).

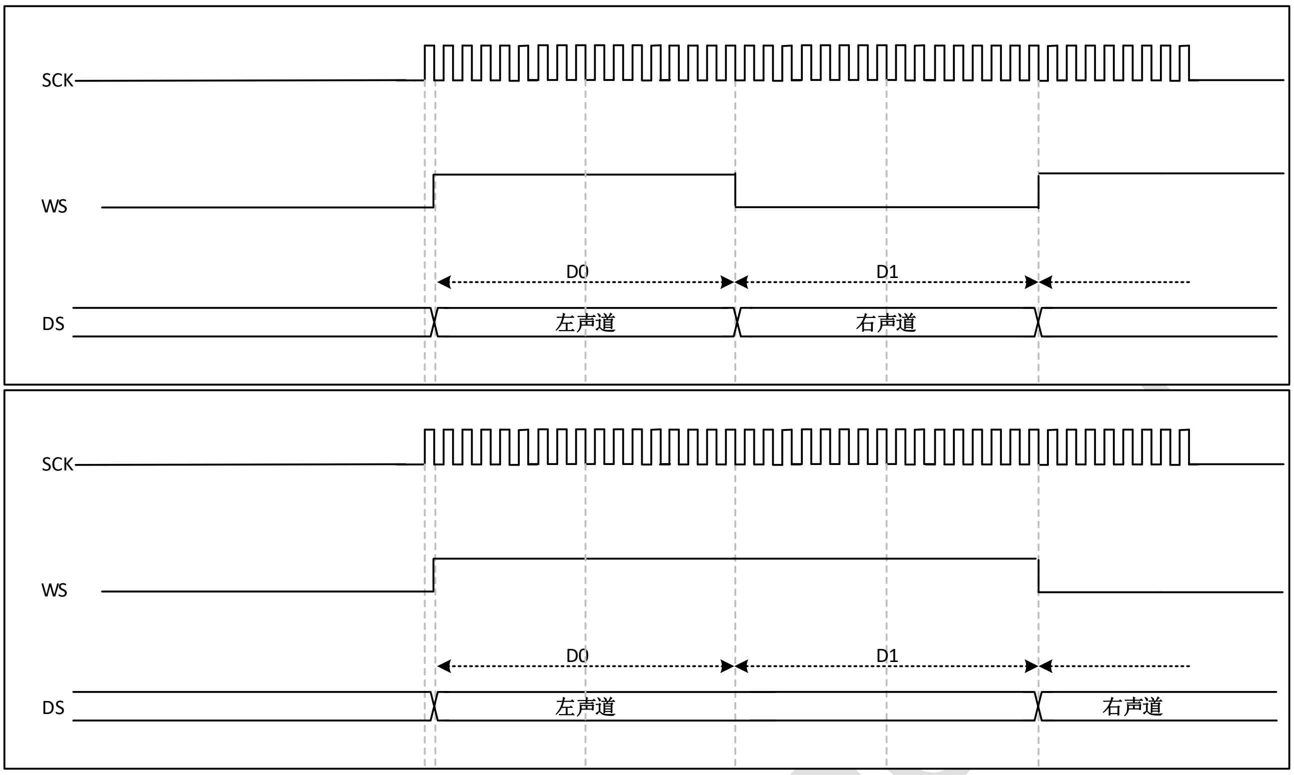


Figure 29-16 I2S MSB justified standard (16/32 bit full accuracy, CPOL = 0)

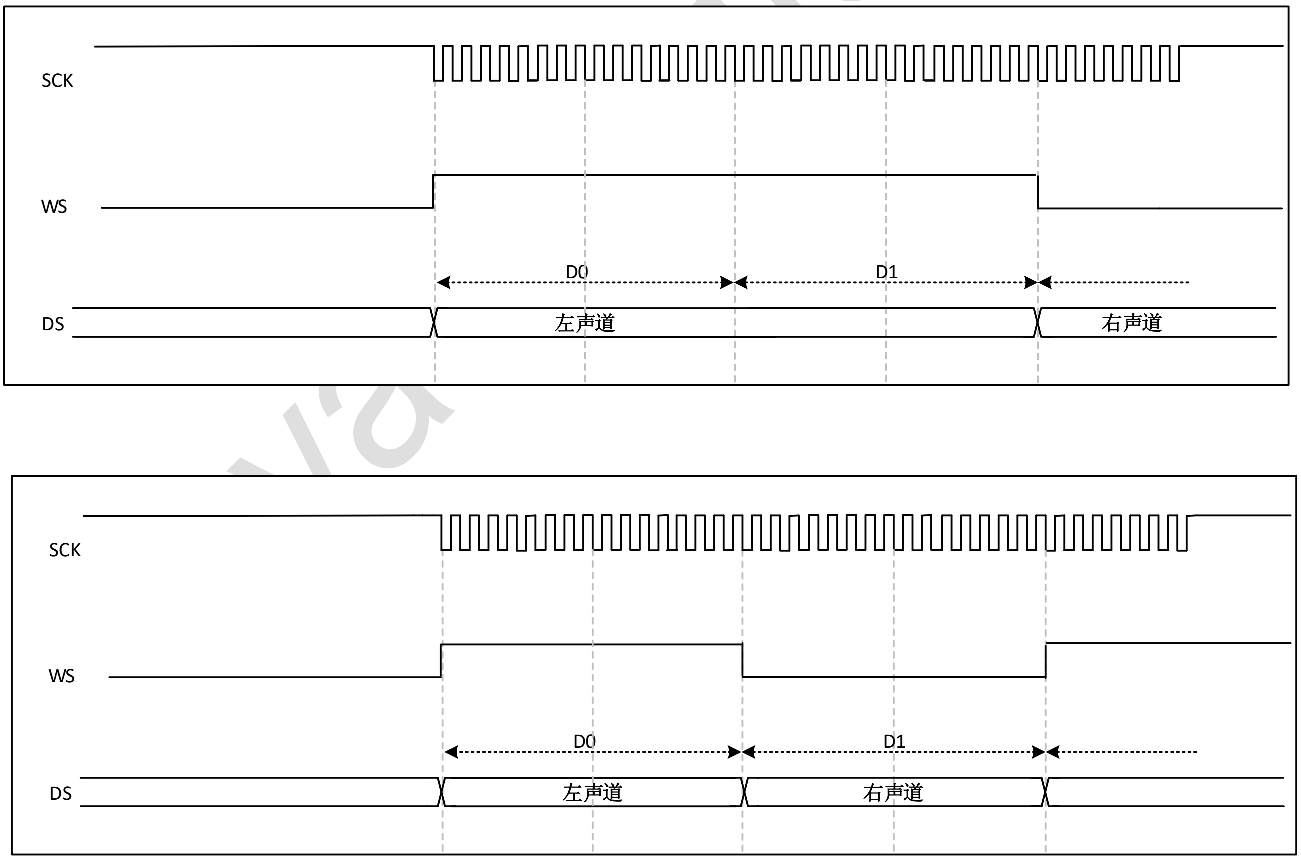


Figure 29-17 I2S MSB justified standard (16/32 bit full accuracy, CPOL = 1)

Data are latched on the falling edge of CK (for transmitter) and are read on the rising edge (for the receiver).

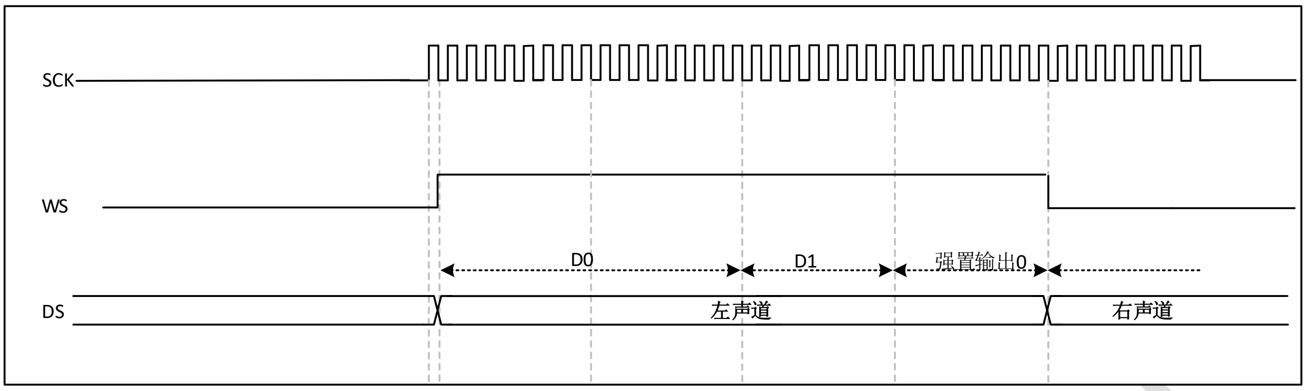


Figure 29-18 I2S MSB justified standard (24-bit frame, CPOL = 0)

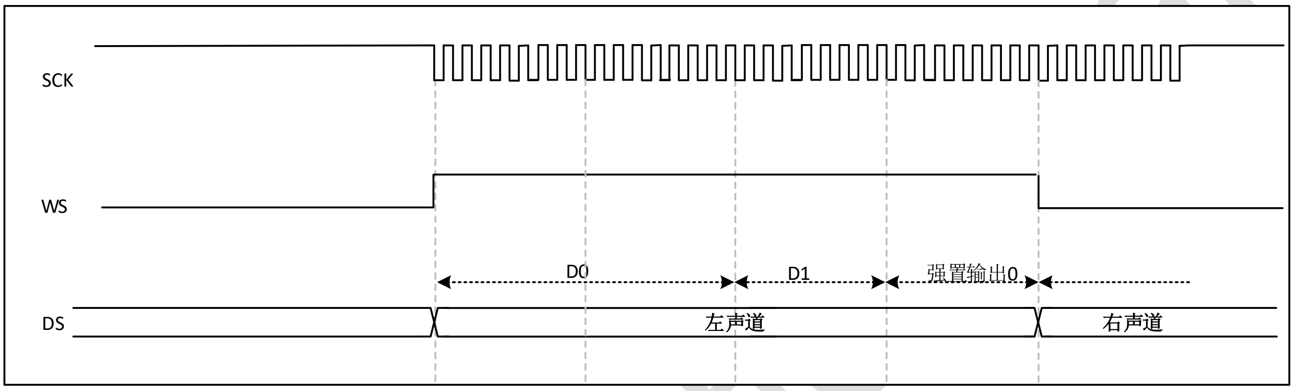


Figure 29-19 I2S MSB justified standard (24-bit frame, CPOL = 1)

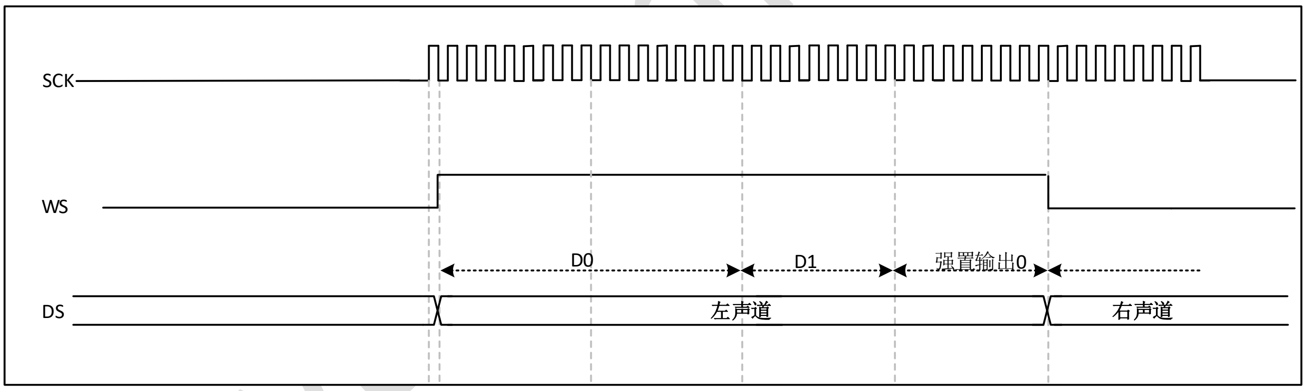


Figure 29-20I2S MSB justified standard (16-bit extended 32-bit, CPOL = 0)

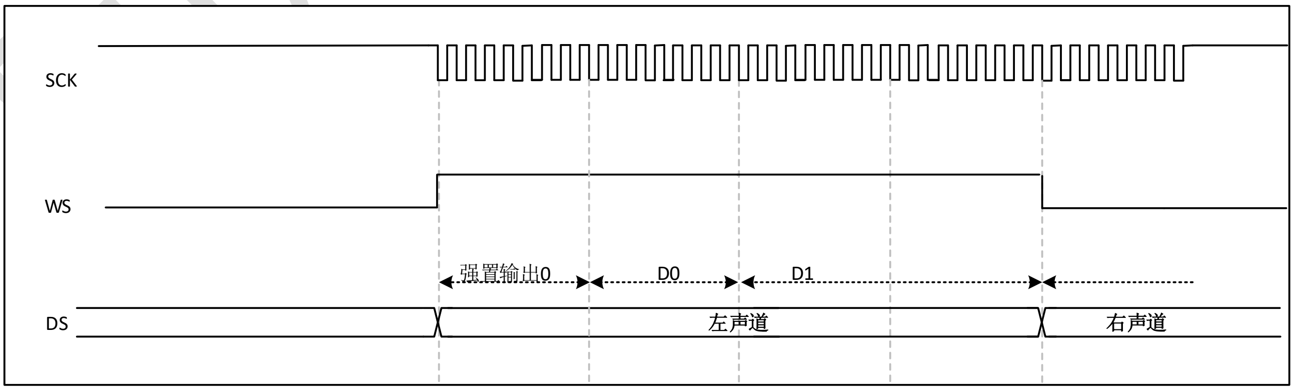


Figure 29-21 I2S MSB justified standard (16-bit extended 32-bit, CPOL = 1)

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).

29.4.2.3. LSB justified standard

This standard is similar to the MSB alignment standard (no difference in 16-bit or 32-bit full-precision frame formats).

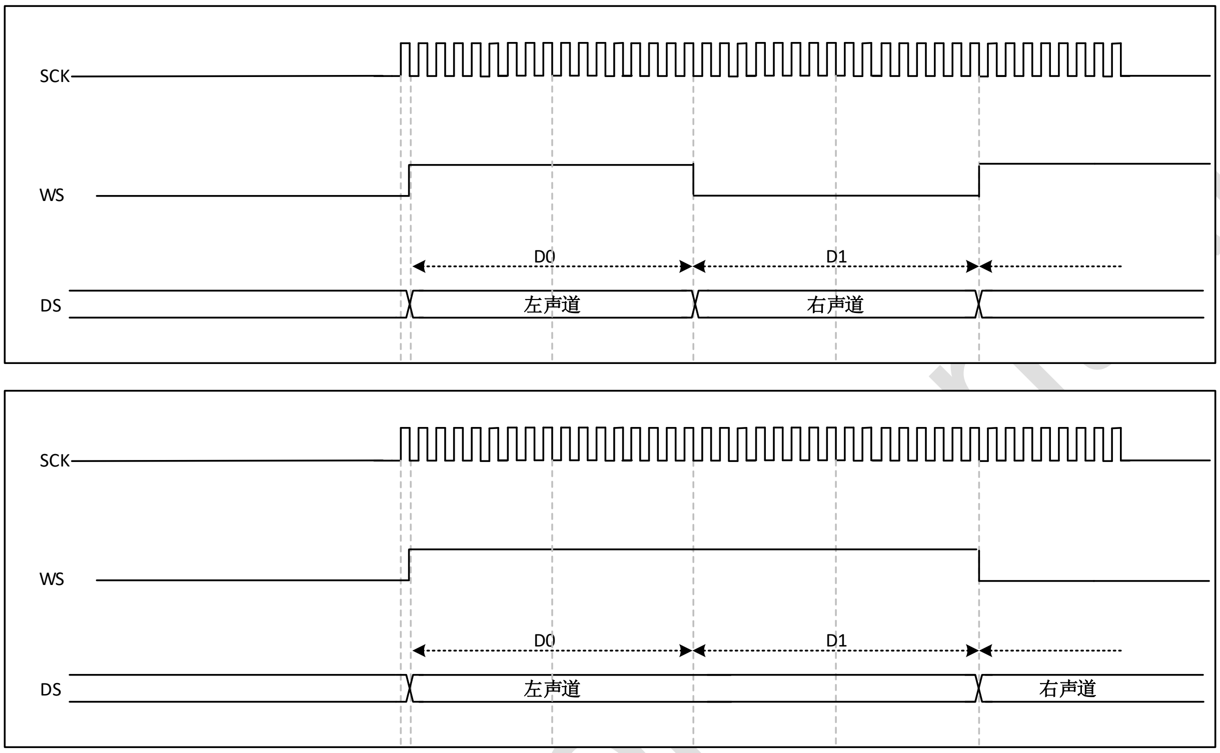


Figure 29-22 I2S LSB justified standard (16/32 bit full accuracy, CPOL = 0)

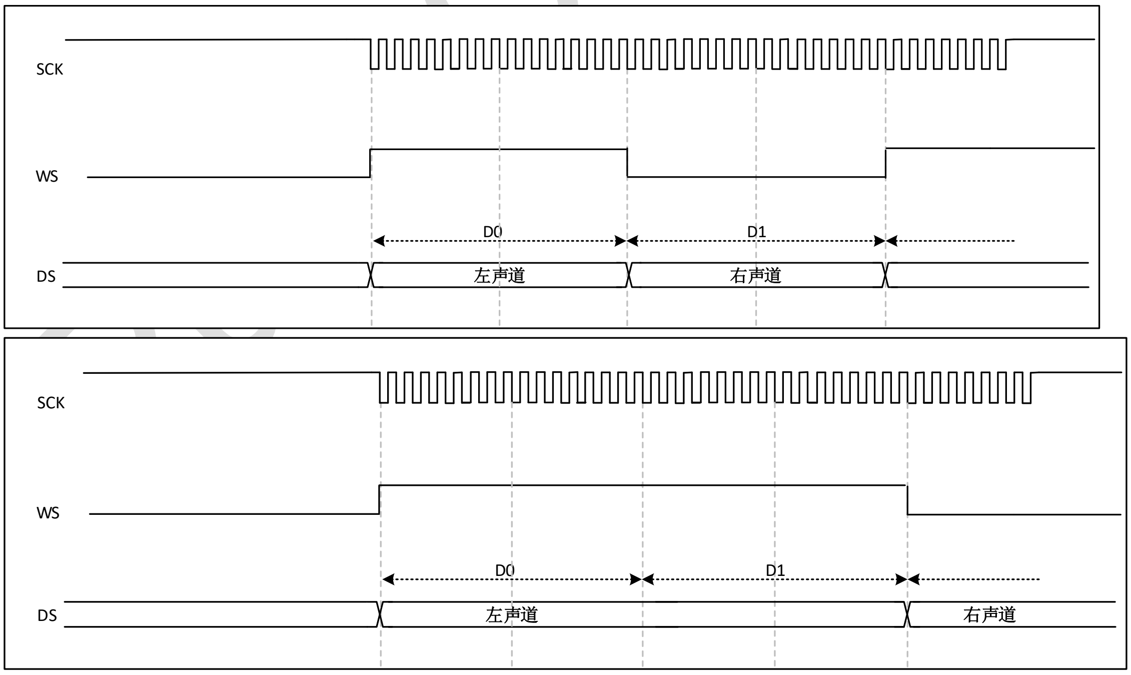


Figure 29-23 I2S LSB justified standard (16/32 bit full accuracy, CPOL = 1)

LSB aligns 24-bit data, CPOL = 0

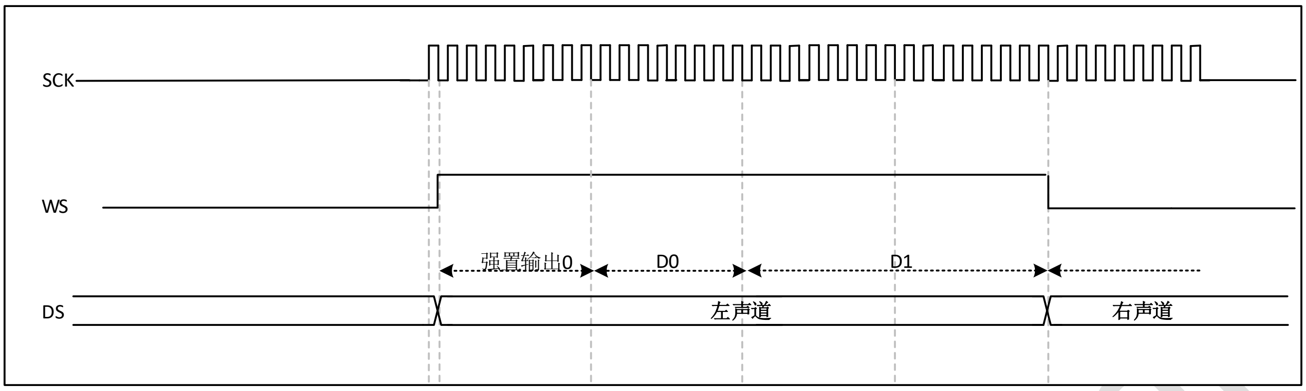


Figure 29-24 I2S LSB justified standard (24-bit frame, CPOL = 0)

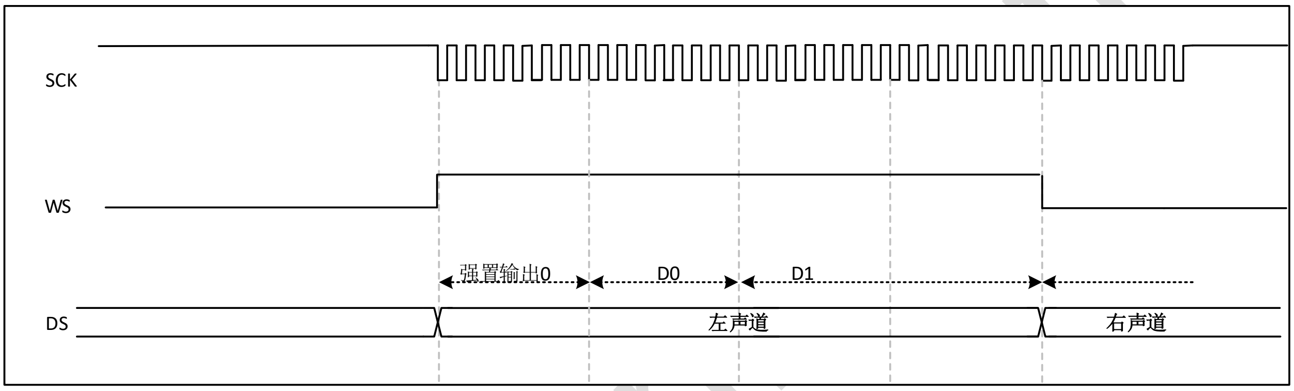


Figure 29-25 I2S LSB justified standard (24-bit frame, CPOL = 1)

- In transmission mode:

If data 0x3478AE have to be transmitted, two write operations to the SPI_DR register are required by software or by DMA. Data register 0xXX34 is written for the first time and data register 0x78AE is written for the second time.

- In reception mode:

If you want to receive data 0x3478AE, you need to read the register SPI _ DR once when two consecutive RXNE events occur. The first time I read 0x0034, only the lower 8 bits make sense; The second readout is 0x78AE.

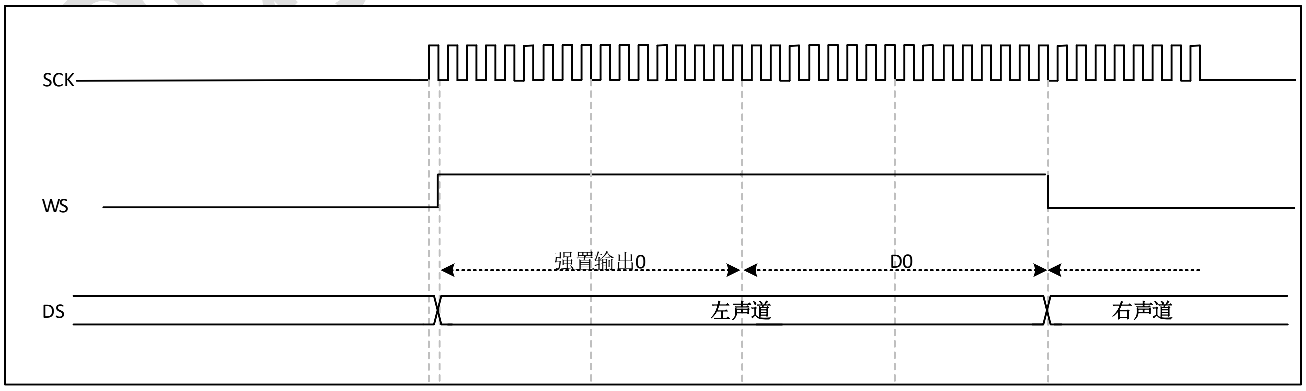


Figure 29-26 I2S LSB justified standard (16-bit extended 32-bit, CPOL = 0)

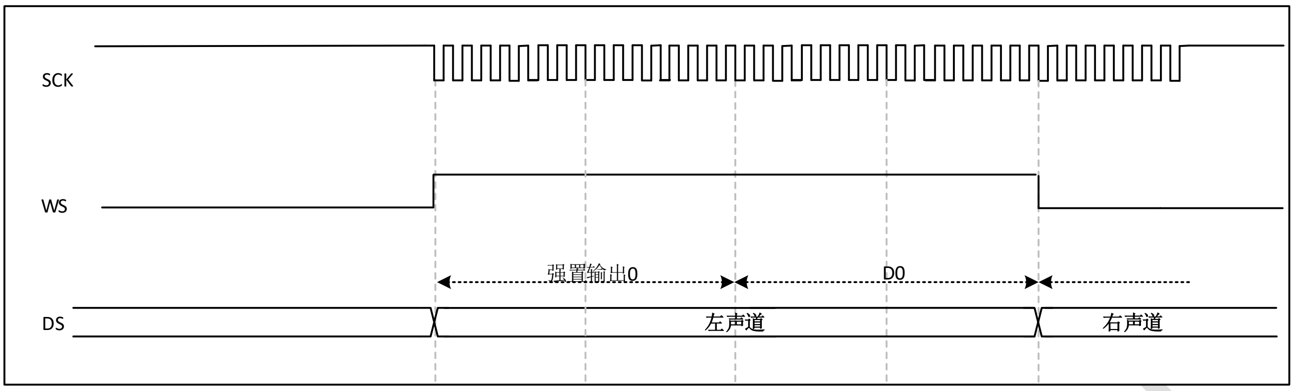


Figure 29-27 I2S LSB justified standard (16-bit extended 32-bit, CPOL = 1)

In the I2S configuration phase, if you choose to expand 16-bit data to 32-channel frames, you only need to access the register SPI _ DR once. 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 0x0000 76A3), you only need to operate the SPI _ DR register once and write 0x76A3.

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, there is more time between two reads and writes, and underflow or overflow can be prevented.

29.4.2.4. PCM standard

Under the PCM standard, there is no information for channel selection. The PCM standard has two available frame structures, short frame or long frame, which can be selected by setting the PCM-SYNC bit of the register SPI _ I2SCFGR.

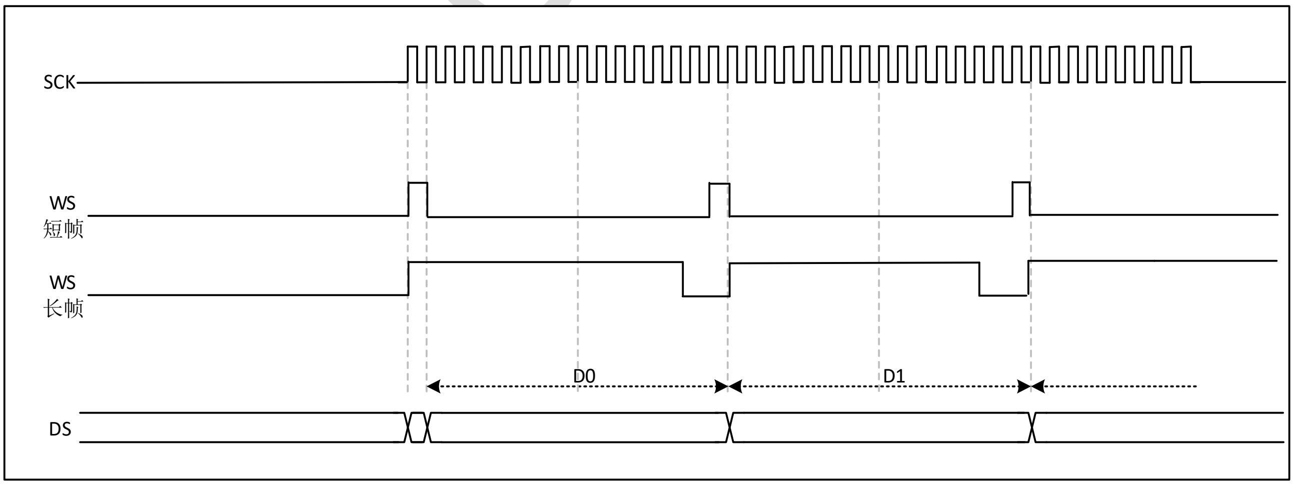


Figure 29-28 I2S PCM standard (16-bit frame, CPOL = 0)

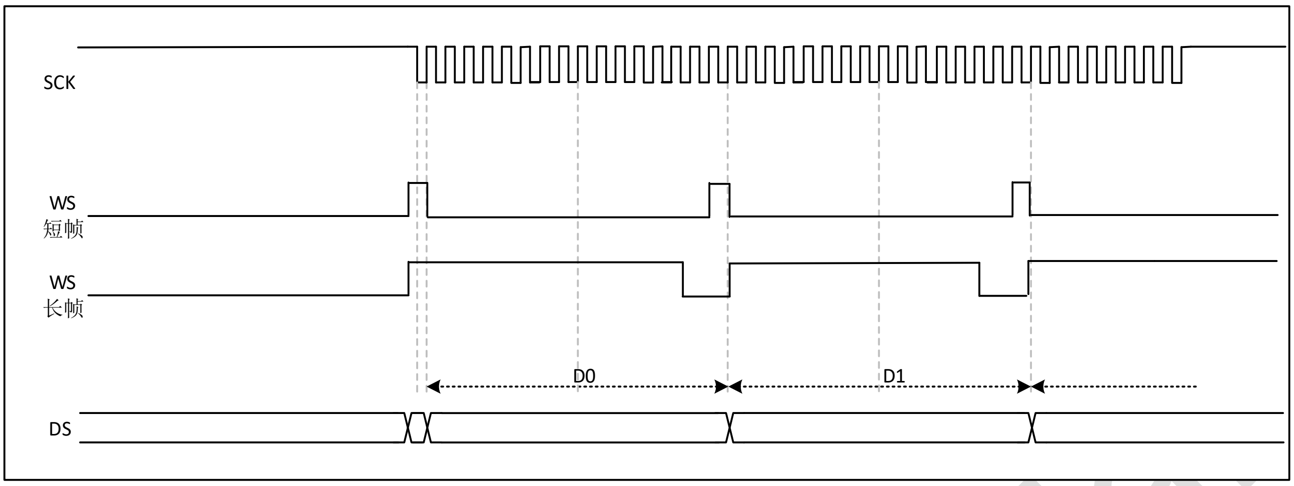


Figure 29-29 I2S PCM standard (16-bit frame, CPOL = 1)

For long frames, in master 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.

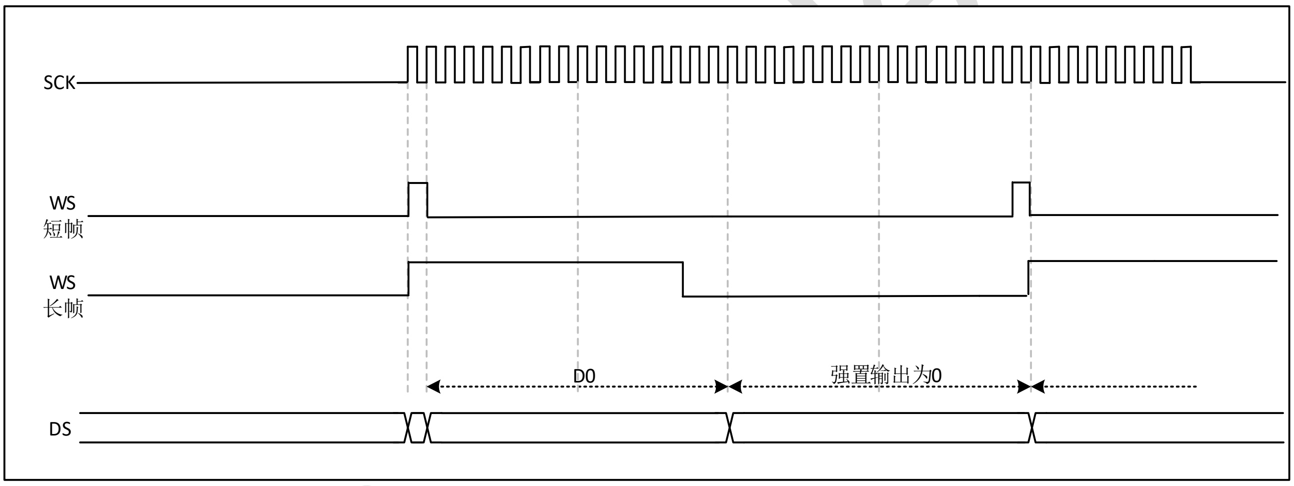


Figure 29-30 I2S PCM standard (16-bit extended to 32-bit, CPOL = 0)

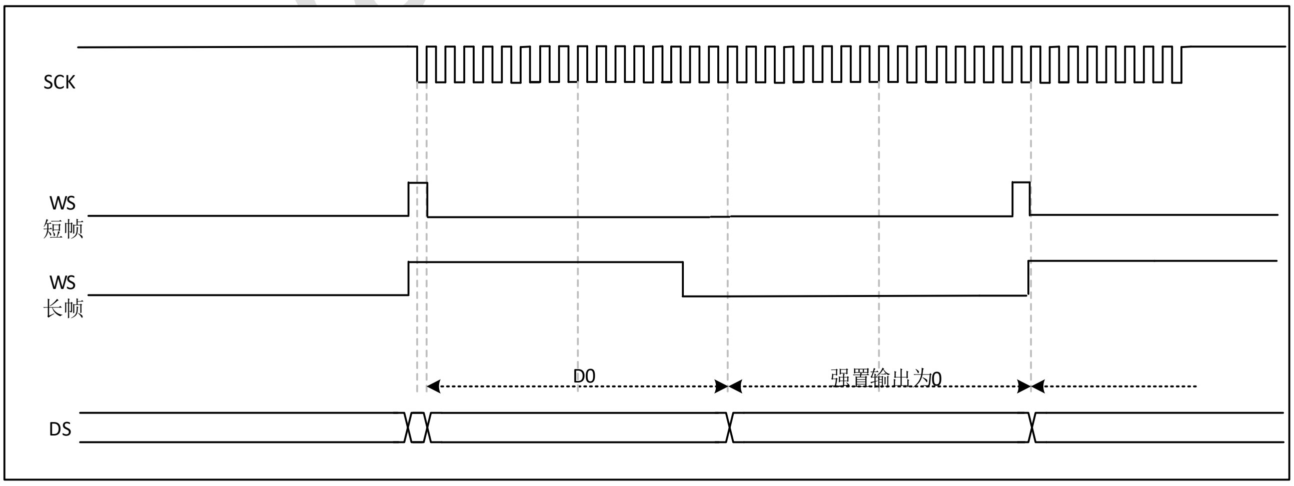


Figure 29-31 I2S PCM standard (16-bit extended to 32-bit, CPOL = 1)

Regardless of the mode (master or slave) and the 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 _ I2 SCFGR register.

29.4.3. Clock generator

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.

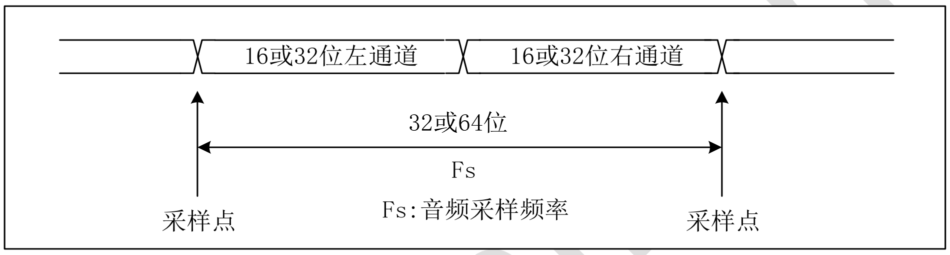
For an audio signal with left and right channels and 16 bits, the I2S bitrate is calculated as follows:

I2S bitrate = 16 × 2 × Fs;

If the packet length is 32 bits, there is:

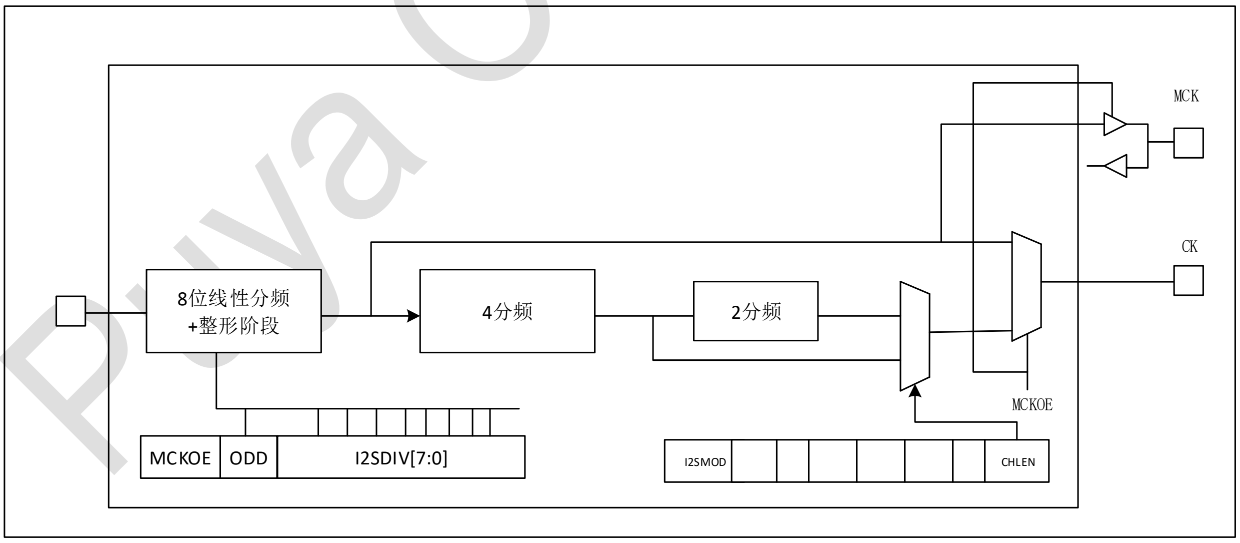
I2S bitrate = 32 × 2 × Fs

Audio sampling definition



In the master mode, in order to obtain the required audio frequency, the linear frequency divider needs to be set correctly.

I2S clock generator architecture



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.

29.4.4. Transmission

29.4.4.1. Master mode

Set I2S to work in master mode, the serial clock is output by pin CK, and the word selection signal is generated by pin WS. Outputting or not outputting the master clock (MCK) can be selected by setting the MCKOE bit of the register SPI _ I2SPR.

The configuration process is as follows:

1. Set the I2SDIV [7: 0] of the register SPI _ I2SPR to define the serial clock baud rate that matches the audio sampling frequency. The ODD bit of the register SPI _ I2SPR is also defined.
2. Select the CKPOL bit to define the steady level for the communication clock. If the master clock MCK needs to be supplied to an external DAC/ADC audio device, the MCKOE position of the register SPI _ I2SPR is set to '1', and the values of I2SDIV and ODD are calculated according to different MCK output states.
3. Set the I2SMOD bit of the register SPI _ I2SCFGR to '1' to activate the I2S function, set the I2S standard used for I2SSTD [1: 0] and PCMSYNC bit selection, and set CHLEN to select the number of data bits for each channel. Also set the I2SCFG [1: 0] of the register SPI _ I2SCFGR to select the I2S master mode and direction.
4. If necessary, the required interrupt function and DMA function can be turned on by setting the register SPI _ CR2.
5. The I2SE position of the register SPI _ I2SCFGR must be set to '1'.
6. Pins WS and CK need to be configured to output mode. If the MCKOE bit of the register SPI _ I2SPR is' 1 ', the pin MCK is also configured to output mode.

Sending process

When one and a half word (16 bits) of data is written to the transmission buffer, the transmission flow begins.

It is assumed that the first data written to the transmission buffer corresponds to left channel data. When the data is moved from the transmission buffer to the shift register, the flag bit TXE is set to '1', and at this time, the data corresponding to the right channel is written to the transmission buffer. The flag bit CHSIDE indicates which channel the data currently to be transmitted corresponds to. The value of the flag bit CHSIDE is updated when the TXE is' 1 ', so it makes sense when the TXE is' 1 '. Only after the data of the left channel and then the right channel are transmitted can it be considered a complete data frame. It is not possible to transmit only a portion of the data frame, such as data of only the left channel.

When the first bit of data is sent out, the half-word data is transferred in parallel to the 16-bit shift register, and then the subsequent bits are sent out from the pin MOSI/SD in the order of higher bits first. The flag bit TXE is set to '1' each time data is moved from the transmit buffer to the shift register, and if the TXEIE bit of register SPI _ CR2 is' 1 ', an interrupt is generated.

The operation of writing data depends on the chosen I2S standard, as detailed in Section 5.2. In order to ensure continuous audio data transmission, it is recommended to write the next data to be transmitted to the register SPI _ DR before the current transmission is completed. It is recommended that when the I2S function is to be turned off, wait for the flag bits TXE = 1 and BSY = 0, and then clear the I2SE bit to '0'.

Receiving Process

Regardless of the data and channel length, audio data is always received in 16-bit packets. That is, each time the reception buffer is filled, the flag bit RXNE is set to '1', and if the RXNEIE bit of the register SPI _ CR2 is' 1 ', an interrupt is generated. Depending on the configured data and channel length, it is necessary to transmit the data to the reception buffer once or twice to receive the data from the left channel or the right channel. The RXNE flag bit is cleared by reading the register SPI _ DR. The CHSIDE is updated after each reception. Its value depends on the WS signal generated by the I2S unit. The operation of reading the data depends on the selected I2S standard.

If the previously received data has not been read and new data is received, that is, an overflow occurs, and the flag bit OVR is set to '1', and if the ERRIE bit of register SPI _ CR2 is' 1 ', an interrupt is generated, indicating that an error has occurred.

Turn off I2S

To turn off the I2S function, special operations need to be performed to ensure that the I2S module can normally complete the transmission cycle without starting a new data transmission;

Specific operation process:

To turn off the I2S function, special operations need to be performed to ensure that the I2S module can complete the transmission cycle normally without starting a new data transmission. The operation procedure is related to the data configuration and channel length, and the mode of the audio protocol:

- 16-bit data extended to 32-bit channel length (DATLEN = 00 and CHLEN = 1) using LSB (low bit) alignment mode (I2SSTD = 10)
  - Waiting for the penultimate (n-1) RXNE = 1;
  - Wait for 17 I2S clock cycles (using software delay);
  - Turn off I2S (I2SE = 0).
- 16-bit data extension to 32-bit channel length (DATLEN = 00 and CHLEN = 1) using MSB (high) alignment, I2S or PCM mode (I2SSTD = 00, I2SSTD = 01 or I2SSTD = 11, respectively)
  - Wait for the last RXNE = 1;
  - Wait for 1 I2S clock cycle (using software delay);
  - Turn off I2S (I2SE = 0).
- All other combinations of DATLEN and CHLEN, any audio mode selected by I2SSTD, turns off I2S using the following manner:

— Waiting for the penultimate (n-1) RXNE = 1;

— Waiting for one I2S clock cycle (using software delay);

— Turn off I2S (I2SE = 0).

29.4.4.2. Slave mode

In the slave mode, the I2S may be set to transmit and receive modes. The configuration of the slave mode basically follows the same process as the configuration of the master mode. In slave mode, the I2S interface is not required to provide the clock. Both the clock signal and the WS signal are supplied by the external master I2S device, connected to the corresponding pins. So the user does not need to configure the clock.

The configuration steps are as follows:

1. Set the I2SMOD bit of the register SPI _ I2SCFGR to activate the I2S function; Setting I2SSTD [1: 0] to select the I2S standard used; Set DATLEN [1: 0] to select the number of bits of data; Set CHLEN to select the number of data bits per channel. The I2SCFG [1: 0] of the set register SPI _ I2SCFGR selects the data direction of the I2S slave mode.

2. As needed, set the register SPI _ CR2 to turn on the required interrupt function and DMA function.

3. The I2SE bit of the register SPI _ I2SCFGR must be set to '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 device must be enabled and written to the I2S data register before the external master device can begin 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 communication is started, data is transferred from the transmit buffer to the shift register, and then the flag bit TXE is set to '1'; At this time, the data item corresponding to the right channel is written into the I2S data register.

The flag bit CHSIDE indicates which channel the data currently to be transmitted corresponds to. In the slave mode, the CHSIDE depends on the WS signal from the external master I2S, as compared to the transmission flow of the master mode. This means that the slave I2S has to prepare the first data to be sent before receiving the clock signal generated by the host. A WS signal of '1' indicates that the left channel is transmitted first.

Note: The time when the I2SE bit is set to '1' should be at least 2 PCLK clock cycles earlier than the main I2S clock signal on the CK pin.

When the first bit of data is sent out, the half-word data is transmitted in parallel through the I2S internal bus to the 16-bit shift register, and then the other bits are sent out from the pin MOSI/SD in the order of higher bits first. Each time data is transferred from the transmit buffer to the shift register, the flag bit TXE is set to '1', and if the TXEIE bit of the register SPI _ CR2 is' 1 ', an interrupt is generated. Note that before writing data to the transmit buffer, it is confirmed that the flag bit TXE is' 1 '. The operation of writing data depends on the selected I2S standard, as detailed in Section 5.2.

In order to ensure continuous audio data transmission, it is recommended to write the next data to be transmitted to the register SPI _ DR before the current transmission is completed. If new data is still not written to register SPI _ DR before the first clock edge representing the next data transfer arrives, the underflow flag bit is set to '1' and an interrupt may be generated; It indicates that the software is sending data errors. If the ERRIE bit of the register SPI _ CR2 is' 1 ', an interrupt is generated when the flag bit UDR of the register SPI _ SR is high. It is recommended to turn off I2S at this time, and then start sending data from the left channel again.

It is recommended to wait for TXE = 1 and BSY = 0 before clearing the I2SE bit to close I2S.

Receiving Process

Regardless of the data and channel length, audio data is always received as a 16-bit packet, i.e. each time the receive buffer is filled, the flag bit RXNE is set to '1', and if the RXNEIE bit of register SPI _ CR2 is' 1 ', an interrupt is generated. According to different data and channel length settings, receiving left channel or right channel data will require one or two transmissions of data to the receiving 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 I2S unit. Reading the SPI _ DR register will clear the RXNE bit. The operation of reading data depends on the selected I2S standard.

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.

29.4.5. I2S flag bit

29.4.5.1. Status flag

There are 3 status flags for users to monitor the status 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. When transmission begins, the BSY flag is set to '1' unless the I2S module is in primary receive mode.

The flag bit is cleared when:

- When the transmission ends (except for the main transmit mode, where communication is continuous);
- When the I2S module is turned off.

When the 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.

— Send buffer null flag bit (TXE)

The flag bit of '1' indicates that the transmission buffer is empty, and new data to be transmitted can be written to the transmission buffer. When data already exists in the transmit buffer, the flag bit is cleared '0'. When I2S is off (I2SE bit is' 0 '), the flag bit is also' 0 '.

— Receive buffer non-empty flag bit (RXNE)

The flag position '1' indicates that valid data is received in the reception buffer. When register SPI _ DR is read, this bit is cleared '0'.

— Channel Flag (CHSIDE)

In send mode, this flag bit is refreshed when TXE is high, indicating the channel on which data is sent from the SD pin. If an underflow error occurs in the slave send mode, the value of this flag bit is invalid, and I2S needs to be turned off and turned on before restarting communication. In the receive mode, this flag bit is refreshed when the register SPI _ DR receives data, indicating the channel in which the received data is located. 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).

Under the PCM standard, this flag bit is meaningless regardless of short frame format or long frame format.

If the flag bit OVR or UDR of the register SPI _ SR is' 1 'and the ERRIE bit of the register SPI _ CR2 is' 1 ', an interrupt is generated and the interrupt flag can then be cleared by reading the register SPI _ SR.

29.4.5.2. Error flag

The I2S cell has 2 error flag bits.

1) Underrun flag (UDR)

In the slave transmit mode, if new data has not been written to the SPI _ DR register when the first clock edge of the data transmission arrives, the flag bit is set to '1'. This flag bit is not valid until the I2SMOD position '1' of the register SPI _ I2SCFGR. If the ERRIE bit of register SPI _ CR2 is' 1 ', an interrupt is generated. The flag bit is cleared by a read operation on register SPI _ SR.

2) Overflow flag bit (OVR)

If new data is received when the previously received data has not been read out, an overflow is generated, the flag position '1', and if the ERRIE bit of register SPI _ CR2 is' 1 ', an interrupt is generated to indicate that an error has occurred. At this time, the contents of the reception buffer 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 is lost. The flag bit is cleared by first reading register SPI _ SR and then reading register SPI _ DR.

29.4.6. I2S DMA

Except for CRC function, same as SPI. Because there is no data transmission protection system in I2S mode.

29.4.7. I2S Interrupt

The I2S interrupt request is as follows:

Interrupt event	Event flag bit	Enable flag bit
Send buffer null flag bit	TXE	TXEIE
Receive buffer non-empty flag bit	RXNE	RXNEIE
Underflow flag bit	OVR	ERRIE
Overflow flag bit	UDR	

29.5. SPI/I2S register

The peripheral registers have to be accessed by half-words (16 bits) or words (32 bits), while the DR register supports 32-bit, 16-bit and 8-bit access.

29.5.1. SPI control register 1 (SPI_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BI-DIMODE	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	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: 2-line unidirectional data mode selected 1: 1-line bidirectional data mode selected
14	BIDIOE	RW	0	Output enable in bidirectional mode This bit combined with the BIDIMODE bit selects the direction of transfer in bidirectional mode. 0: Output disabled (receive-only mode) 1: Output enabled (transmit-only mode) In master mode, the MOSI pin is used while the MISO pin is used in slave mode.
13	CRCEN	RW	0	Hardware CRC calculation enable 0: Disable CRC calculation; 1: Start CRC calculation. Note: This bit should be written only when SPI is disabled (SPE = '0') for correct operation. This bit can only be used in full duplex mode. Note: It is not used in I^{2}S mode. Note: This register is fixed to 0 if the I2C CRC function is not supported.
12	CRCNEXT	RW	0	Next transmit CRC (Transmit CRC next) 0: The value of the next transmission comes from the transmit buffer. 1: The next transmitted value comes from the transmit CRC register. Note: This bit should be set immediately after the last data is written to the SPI _ DR register. Note: It is not used in I^{2}S mode. Note: This register is fixed to 0 if the I2C CRC function is not supported.
11	DFF	RW	0	Data frame format 0: 8-bit data frame format is selected for trans-mission/reception 1: 16-bit data frame format is selected for trans-mission/reception

Bit	Name	R/W	Reset Value	Function
				Note: This bit should be written only when SPI is disabled (SPE = ‘0’) for correct operation. Note: It is not used in I^{2}S mode.
10	RXONLY	RW	0	Receive only This bit combined with the BIDIMODE bit selects the direction of transfer in 2-line unidirectional mode. This bit is also useful in a multislave system in which this particular slave is not accessed, the output from the accessed slave is not corrupted. 0: Full-duplex (Transmit and receive) 1: Output disabled (receive-only mode)
9	SSM	RW	0	Software slave management When the SSM bit is set, the NSS pin input is replaced with the value from the SSI bit. 0: Software slave management disabled 1: Software slave management enabled
8	SSI	RW	0	Internal slave select This bit has an effect only when the SSM bit is set. The value of this bit is forced onto the NSS pin and the IO value of the NSS pin is ignored.
7	LSBFIRST	RW	0	Frame format 0: MSB transmitted first 1: LSB transmitted first Note: This bit should not be changed when communication is ongoing. Note: It is not used in I^{2}S mode.
6	SPE	RW	0	SPI enable 0: Peripheral disabled 1: Peripheral enabled Note: It is not used in I^{2}S mode.
5:3	BR[2:0]	RW	0	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: This bit should not be changed when communication is ongoing. Note: It is not used in I^{2}S mode.
2	MSTR	RW	0	Master selection 0: Slave configuration 1: Master configuration Note: This bit should not be changed when communication is ongoing. Note: It is not used in I^{2}S mode.
1	CPOL	RW	0	Clock polarity 0: In idle state, SCK remains low 1: SCK maintains high level in idle state Note: This bit should not be changed when communication is ongoing. Note: It is not used in I^{2}S mode.
0	CPHA	RW	0	Clock phase

Bit	Name	R/W	Reset Value	Function
				0: Data sampling starts from the first clock edge 1: Data sampling starts from the second clock edge Note: This bit should not be changed when communication is ongoing. Note: It is not used in I^2S mode.

29.5.2. SPI control register 2 (SPI_CR2)

Address offset: 0x04

Reset value: 0x0700

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					Res	Res	Res	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	TXDMAEN	RXDMAEN
					RW	RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
10:8	Reserved	-	-	Reserved
7	TXEIE	RW	0	Tx buffer empty interrupt enable 0: TXE interrupt masked 1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set.
6	RXNEIE	RW	0	R_X buffer not empty interrupt enable 0: RXNE interrupt masked 1: RXNE interrupt not masked. Used to generate an interrupt request when the RXNE flag is set.
5	ERRIE	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled When CRCERR, OVR, or MODF is 1, an interrupt request is generated.
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	-	-	Reserved
2	SSOE	RW	0	SS output enable 0: SS output is disabled in master mode and the SPI interface can work in multimaster configuration. 1: SS output is enabled in master mode and when the SPI interface is enabled. The SPI interface cannot work in a multimaster environment. Note: It is not used in I^2S mode.
1	TXDMAEN	RW	0	Send buffer DMA enabled. 0: Prohibit transmission of buffer DMA 1: Enable transmit buffer DMA. When TXE = 1, a DMA request is issued.
0	RXDMAEN	RW	0	Receive buffer DMA enabled. 0: Prohibit reception buffer DMA 1: Enable receive buffer DMA. When TXE = 1, a DMA request is issued.

29.5.3. SPI status register (SPI_SR)

Address offset: 0x08

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	FTLVL [1:0]		FRLVL [1:0]		Res	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
			R	R	R	R		R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:11	FTLVL	R	0	FIFO transmission level Hardware set, hardware clear 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full (when the FIFO threshold is greater than 1/2, it is considered full) Note: It is not used in I^{2}S mode.
10:9	FRLVL	R	0	FIFO reception level Hardware set, hardware clear 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full Note: I2S mode and SPI with CRC check are not used in receive mode only.
8	Reserved	-	-	Reserved
7	BSY	R	0	Busy flag 0: SPI not busy 1: SPI is busy in communication or Tx buffer is not empty
6	OVR	R	0	Overrun flag 0: No overflow error 1: Overrun occurred This flag is set by hardware and reset by a software sequence.
5	MODF	R	0	Mode fault 0: No mode fault occurred 1: Mode fault occurred This flag is set by hardware and reset by a software sequence. Note: It is not used in I^{2}S mode.
4	CRCERR	R	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 bit is set by hardware and reset by software writing '0'. Note: It is not used in I^{2}S mode. Note: This register is fixed to 0 if the I2C CRC function is not supported.
3	UDR	R	0	Underflow flag bit. 0: No underflow occurs; 1: An underflow error occurred. Hardware set, software sequence cleared. Note: Not used in SPI mode.
2	CHSIDE	R	0	Channel control. 0: transmit or receive left channel 1: Transmit or receive the right channel. Note: Not used in SPI and I2S PCM modes.
1	TXE	R	1	Transmit buffer empty 0: Tx buffer not empty 1: Tx buffer empty
0	RXNE	R	0	Receive buffer not empty

Bit	Name	R/W	Reset Value	Function
				0: Rx buffer not empty 1: Rx buffer empty

29.5.4. SPI data register (SPI_DR)

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. Data received or to be transmitted. The data register serves as an interface between the Rx and Tx FIFOs. When the data register is read, RxFIFO is accessed while the write to data register accesses TxFIFO. Note: Depending on the DS bit (data frame width selection), data transmission or reception is 8-bit or 16-bit. For an 8-bit data frame, the buffers are 8-bit and only the LSB of the register is used for transmission/reception. When in reception mode, the MSB of the register (SPI_DR[15:8]) is forced to 0. For a 16-bit data frame, the buffers are 16-bit and the entire register, SPI_DR[15:0] is used for transmission/reception.

29.5.5. SPI CRC polynomial register (SPI _ CRCPR)

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 the polynomials used in CRC calculation. Its reset value is 0x0007, and other values can be set according to the application. Note: It is not used in I^{2}S mode. Note: Polynomial values can only be odd, even values are not supported. Note: This register is fixed to 0 if the I2C CRC function is not supported.

29.5.6. SPI Tx CRC register (SPI _ TXCRC)

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]															
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	RxCRC [15: 0]	RW	0	When CRC calculation is enabled in the Receive CRC register, RXCRC [15: 0] contains the CRC value calculated from the received bytes. When '1' is written at the CRCEN bit of SPI _ CR1, this register is reset. The CRC calculation uses the polynomial in SPI _ CRCPR. When the data frame format is set to 8 bits, only the lower 8 bits participate in the calculation and proceed according to the method of CRC8; When the data frame format is 16 bits, all 16 bits in the register participate in the calculation, and according to the standard of CRC16. Note: When reading this register when the BSY flag is' 1 ', an incorrect value may be read. Note: It is not used in I^2S mode. Note: This register is fixed to 0 if the I2C CRC function is not supported.

29.5.7. SPI Tx CRC register (SPI _ TXCRC)

Address offset: 0x18

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	TxCRC [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. When '1' is written at the CRCEN bit of SPI _ CR1, this register is reset. The CRC calculation uses the polynomial in SPI _ CRCPR. When the data frame format is set to 8 bits, only the lower 8 bits participate in the calculation and proceed according to the method of CRC8; When the data frame format is 16 bits, all 16 bits in the register participate in the calculation, and according to the standard of CRC16. Note: When reading this register when the BSY flag is' 1 ', an incorrect value may be read. Note: It is not used in I^2S mode. Note: This register is fixed to 0 if the I2C CRC function is not supported.

29.5.8. SPI _ I2S configuration register (SPI _ I2S _ CFGR)

Address offset: 0x1C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG [1:0]		PCM-SYNC	Res.	I2SSTD [1:0]		CKPOL	DTALEN[1:0]		CHLEN
				RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	I2SMOD	RW	0	I2S mode selection 0: SPI mode is selected; 1: Select I2S mode. Note: This bit can only be set when SPI or I2S is turned off.

Bit	Name	R/W	Reset Value	Function
				Note: This register is fixed to 0 if the I2S function is not supported.
10	I2SE	RW	0	I2S enable 0: Turn off I2S; 1: I2S enabled. Note: Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
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 set when I2S is turned off. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
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). Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
6	Reserved	-	-	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 standard. Note: For proper operation, this bit can only be set when I2S is turned off. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
3	CKPOL	RW	0	Steady state clock polarity 0: I2S clock quiescent state is low level; 1: The I2S clock stationary state is high. Note: For proper operation, this bit can only be set when I2S is turned off. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
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: Disabled. Note: For proper operation, this bit can only be set when I2S is turned off. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
0	CHLEN	RW	0	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 proper operation, this bit can only be set when I2S is turned off. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.

29.5.9. SPI _ I2S prescaler register (SPI _ I2SPR)

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV [7:0]							
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	MCKOE	RW	0	Master clock output enable 0: Turn off master device clock output; 1: Master device clock output enabled. Note: For proper operation, this bit can only be set when I2S is turned off. Use this bit only in I2S master mode. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
8	ODD	RW	0	Odd coefficient pre-division (Odd factor for the pre-scaler) 0: actual frequency division coefficient = I2SDIV * 2; 1: actual frequency division coefficient = (I2SDIV * 2) +1. Note: For proper operation, this bit can only be set when I2S is turned off. Use this bit only in I2S master mode. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.
7:0	I2SDIV	RW	0	I2S linear prescaler Disable setting I2SDIV [7: 0] = 0 or I2SDIV [7: 0] = 1 Note: For proper operation, this bit can only be set when I2S is turned off. Use this bit only in I2S master mode. Not used in SPI mode. Note: This register is fixed to 0 if the I2S function is not supported.

30. Hardware divider (HDIV)

30.1. Introduction

The Divider Divider module is mainly used to divide two 32-bit data of the input module. It needs to consume 16 (8 or 16 clock cycles can be configured by parameters) HCLK clock cycles to complete a division operation.

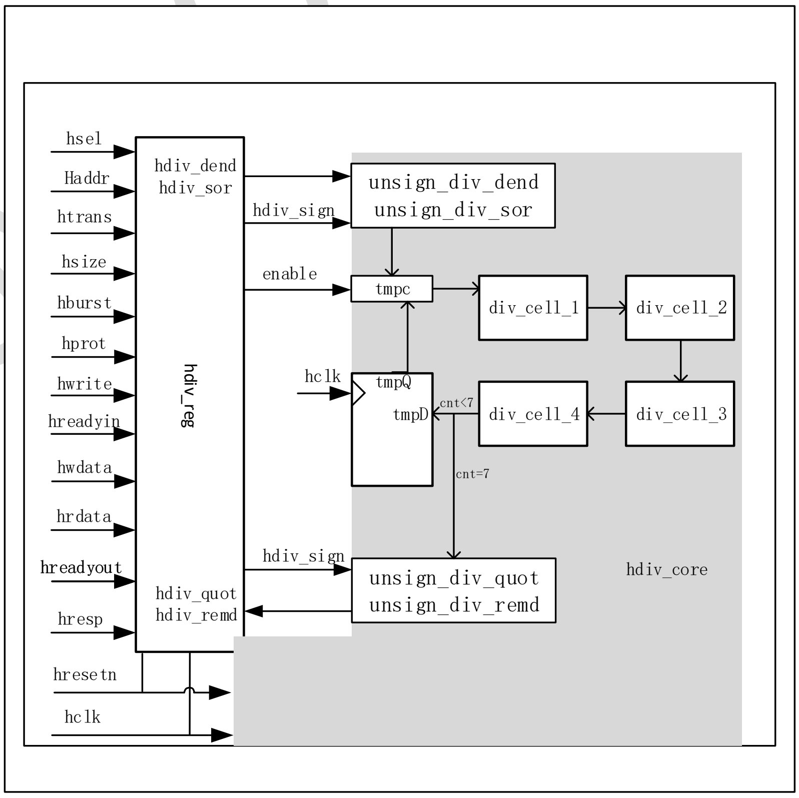
30.2. Main features

- Supports 32-bit division
- The data in the register cannot be changed while the current division is not finished
- Configurable signed/unsigned division calculation
- 32-bit dividend, 32-bit divisor
- Outputs 32-bit quotient and 32-bit remainder
- Divide-by-zero warning flag bit, end-of-division flag bit
- 8 clock cycles to complete a division operation
- Write the divisor register to trigger the start of the division circuit
- After writing the divisor, when reading the quotient and remainder registers, you need to wait for the completion flag DIV_END
- When the divisor is 0, the result of quotient and remainder is 0

30.3. Functional overview

30.3.1. Overview

The block diagram of HDIV module is as follows:



30.3.2. Operation Process

The HDIV operation process is as follows:

1. Turn on the clock enable register of the hardware divider in the system controller.

2. The configuration register HDIV _ SIGN sets signed/unsigned division operation.

3. Configure the register HDIV _ DEND to set the dividend.

4. Configure the register HDIV _ SOR to set the divisor.

5. When the division operation starts, the query register HDIV _ STAT operation ends flag bit HDIV _ END, and HDIV _ END is 1 to mark the end of the operation. Read register HDIV _ QUOT to get the quotient and read register HDIV _ REMD to get the remainder.

6. When the divisor is zero, the division operation ends immediately, the quotient and the remain-der are set to 0 at the same time, and the divisor is zero warning flag bit HDIV _ ZERO is set.

7. Before the end of the division operation, when reading the register HDIV _ QUOT/HDIV _ REMD, the CPU will read the last calculated value

30.4. HDIV register

30.4.1. DIV dividend register (HDIV _ DEND)

Address offset = 0x00,

Reset value = 32 'h0

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

Bit	Name	R/W	Reset Value	Function
31:0	HDIV _ DEND	RW	32' h0	Dividend register The bus feeds the dividend into the module

30.4.2. DIV divisor register (HDIV _ SOR)

Address offset = 0x04

Reset value = 32 'h00000001

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

Bit	Name	R/W	Reset Value	Function
31:0	HDIV _ SOR	RW	32'h00000001	Divisor register The bus inputs a divisor to the module

30.4.3. DIV quotient register (HDIV _ QUOT)

Address offset = 0x08

Reset value = 0x0000 _ 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HDIV QUOT [31: 16]															
RW															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HDIV QUOT [15: 0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:0	HDIV _ QUOT	R	32’h0	Store the quotient calculated by the divider

30.4.4. DIV remainder register (DIV _ REMD)

Address offset = 0x0C

Reset value = 0x0000 _ 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	HDIV _ REMD	R	32’h0	Store the remainder calculated by the divider

30.4.5. DIV sign register (HDIV _ SIGN)

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.	Res.	Res.	Res.	HDIV _ SIGN
															RW

Bit	Name	R/W	Reset Value	Function
31:1	Reserved		31’h0	
0	HDIV _ SIGN	RW	0	Symbol selection register. 0---unsigned division operation 1---signed division operation

30.4.6. DIV status register (HDIV _ STAT)

Address offset = 0x14,

Reset value = 0x0000 _ 0001

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.	HDIV _ ZERO	HDIV _ END
														R	R

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
1	HDIV _ ZERO	R	0	Divisor is zero warning flag bit. 0---The divisor is not zero 1---Divisor is zero
0	HDIV _ END	R	1	End of division flag bit. 0---operation in progress 1---Operation end

31. Digital co-processing

31.1. Overview

CORDIC (Coordinate Rotation Digital Computer) is the abbreviation of Coordinate Rotation Digital Computer algorithm, which is mainly used to solve the real-time calculation problems of trigonometric function and inverse trigonometric function in motor control. Compared to software, CORDIC speeds up functions and saves processor cycles to perform other tasks.

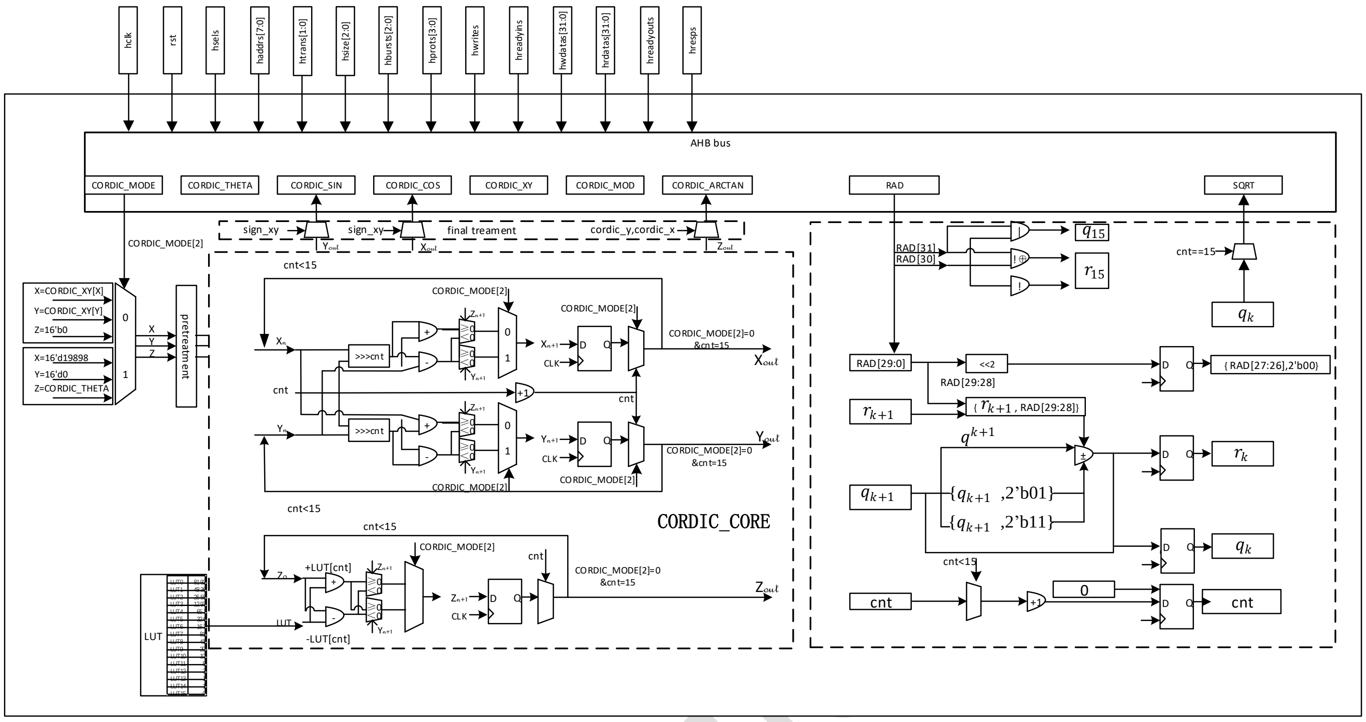
SQRT (Square Root) is short for Square Root. The square number of this module is a 32-bit unsigned integer, and the square root is a 16-bit unsigned integer.

31.2. Main features

- The number of CORDIC iterations can be configured from 1 to 24
- CORDIC rotary engine bit width is 24bit
- The number of clocks required for CORDIC calculation is the same as that number of iterations
- CORDIC implementation functions are sin/cos/arctan/modulus
- CORDIC arctan/modulus function overflow flag
- CORDIC writes input data register and writes START register bits to START calculation
- CORDIC input data error flag and calculation completion flag
- There are three CORDIC interrupt sources: overflow, input data error, and calculation completion.
- The CORDIC input data bit width can be adjusted to 32 bit and 16 bit. (24bit input can be set for arctan/modulus)
- SQRT input data 32 bits, output data 32 bits high
- SQRT calculations require 16 clocks
- SQRT interrupt source is one: calculation completion flag bit

31.3. Functional overview

31.3.1. Architectural block diagram



31.3.2. function

31.3.2.1. SIN and COS

The range of angle values θ in this article is [-π, π], the range of input and output registers is q1.31 or q1.15, and the value of the input register is radian, which has been divided by π, so the range of q1.31 or q1.15 [-1, 1] can represent the range [-π, π]. The output register is the same as above, and it needs to be multiplied by π to get its radian.

For example: Enter a signed radian of θ_i, and the method to express its value with q1.31 is below (decimals are discarded)

$$\frac{\theta_i}{\pi} \times 2^{31}$$

Its value is the radian represented by q1.31, which is replaced by q1.15.
Output signed radians, which are read out in q1.31 or q1.15 mode and restored to radian angle mode: θ_o

$$\frac{\theta_o}{2^{31}} \times \pi$$

If you read it in q1.15 mode, you need to replace the formula 2^{31} with 2^{15}.
When finding arctan and mod in this article, the input value and range are [-1, 1], and use q1.31 or 1.15 to write. y_0 x_0

Need to ensure that the scope satisfies the argument range of the arctan () function, i.e.: $\frac{y_0}{x_0}$ $\frac{y_0}{x_0}$

$$-\frac{\pi}{2} < \frac{y_0}{x_0} < \frac{\pi}{2}$$

The sin cos calculation adopts the cordic circumferential rotation mode. In the engine, any angle Z to [0, π/2] is first processed, and then Z is approximated several times. Each angle approximation can be completed by addition, subtraction and shift of (x, y).
The core algorithm of circumferential system is

$$\begin{cases} x_n = A_n(x_0\cos z_0 - y_0\sin z_0) \\ y_n = A_n(y_0\cos z_0 + x_0\sin z_0) \\ z_n = 0 \\ A_n = \prod_{i=0}^{n-1}\sqrt{1+2^{-2i}} \approx 1.646760258 \end{cases}$$

In order to implement the cos/sin function, let the above formula be $y_0 = 0, x_0 = 1/A_n$

$$\begin{cases} x_n = \cos z_0 \\ y_n = \sin z_0 \\ z_n = 0 \\ A_n = \prod_{i=0}^{n-1}\sqrt{1+2^{-2i}} \end{cases}$$

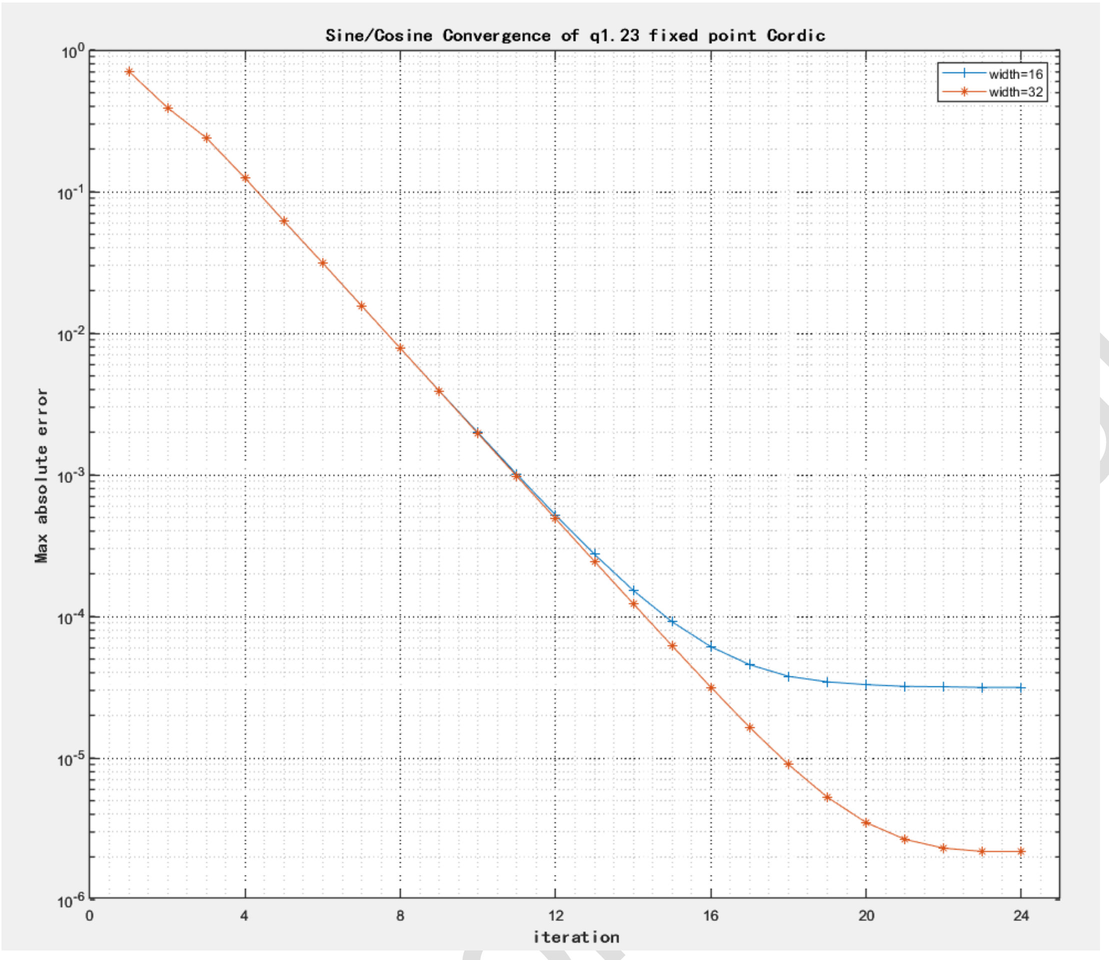
In this way, the cos/sin function can be realized. The above formula explains the function principle. If the lower left formula is not supported in the circuit, you have to convert it to another operation. $\tan\theta_i$
The key to the implementation process of the function is to convert it into a series of (right shift operations), then $\tan\theta\,2^{-2i}$

$$\begin{cases} x_{i+1} = \prod_{i=0}^{n}\cos\theta_i(x_i - y_i\tan\theta_i) \\ y_{i+1} = \prod_{i=0}^{n}\cos\theta_i(y_i + x_i\tan\theta_i) \\ z_{i+1} = \theta - \sum_{i=0}^{n}\theta_i \\ d_i = \begin{cases} +1 & z_i \geqslant 0 \\ -1 & z_i < 0 \end{cases} \end{cases} \xrightarrow{\theta_i = \tan^{-1}(d_i 2^{-i})} \begin{cases} x_{i+1} = \prod_{i=0}^{n}\cos\theta_i(x_i - y_i d_i 2^{-i}) \\ y_{i+1} = \prod_{i=0}^{n}\cos\theta_i(y_i + x_i d_i 2^{-i}) \\ z_{i+1} = z_i - d_i\tan^{-1}2^{-i} \\ d_i = \begin{cases} +1 & z_i \geqslant 0 \\ -1 & z_i < 0 \end{cases} \end{cases}$$

+

The above formula is the angle that needs to be rotated, which is the angle that differs from the target value after rotation. It is determined by the sign of. If it is greater than 0, it is = 1, otherwise it is-1 (when it is equal to 0, either plus or minus 1 can be taken). $\theta\ z_i\ \theta\ d_i\ z_i\ z_i\ d_i\ z_i\ d_i$
The above formula can be obtained: each rotation operation only needs to be resolved into one addition/subtraction and one shift operation to complete.

31.3.2.2. SIN and COS calculation accuracy



In the figure above, the X axis shows the number of iterations, the y axis shows the maximum error, and different curves show different input data bit widths. When the number of iterations exceeds 16 times, only the accuracy of the curve with an input data bit width of 32 bits is still increasing, and the accuracy growth of the curve with a bit width of 16 bits has bottomed out. Users can select the appropriate number of iterations and input data bit width according to the above figure.

31.3.2.3. Arctan/Modulus

In rotation mode, each iteration will approach 0 (which is the angle that has been rotated at present from the target angle, which can also be understood as getting closer and closer to the target value in each iteration). However, in vector mode, a (x, y) coordinate is given, and y approaches 0 by rotation angle. In this process, the accumulation of each rotation angle is the angle between the given initial coordinate and the vector composed of the origin to the positive half axis of x axis z_i z_i θ θ z_n

When the iteration is completed, we get:, and have, then we have the following inference

$$\begin{cases} x_n = A_n(x_0\cos\theta - y_0\sin\theta) \\ y_n = A_n(y_0\cos\theta + x_0\sin\theta) \end{cases}$$

$$y_n = 0$$

$$
\begin{aligned}
x_n &= A_n(x_0\cos\theta - y_0\sin\theta)\\
&= A_n(x_0\cos\theta - y_0\sin\theta + 0)\\
&= A_n(x_0\cos\theta - y_0\sin\theta + y_0\cos\theta + x_0\sin\theta)\\
&= A_n\sqrt{(x_0\cos\theta - y_0\sin\theta + y_0\cos\theta + x_0\sin\theta)^2}\\
&= A_n\sqrt{x_0^2\cos^2\theta + y_0^2\sin^2\theta - 2x_0y_0\sin\theta\cos\theta + y_0^2\cos^2\theta + x_0^2\sin^2\theta + 2x_0y_0\sin\theta\cos\theta}\\
&= A_n\sqrt{x_0^2(\cos^2\theta+\sin^2\theta) + y_0^2(\cos^2\theta+\sin^2\theta)}\\
&= A_n\sqrt{x_0^2+y_0^2}
\end{aligned}
$$

Then after multiple rotations, you get: $y_n = 0$

$$
\begin{cases}
x_n = A_n\sqrt{x_0^2+y_0^2}\\
y_n = 0\\
z_n = z_0 + \tan^{-1}(y_0/x_0)\\
A_n = \prod_{i=0}^{n}\sqrt{1+2^{-2i}}
\end{cases}
$$

The input signal of the vector mode is the x and y coordinates, and its point is set as P. The vector OP is rotated to the positive direction of the x-axis. The length of OP is, and the result is the angle formed by the initial coordinate and the positive direction of the x-axis plus (x_0, y_0) $\sqrt{x_0^2+y_0^2}$ z_n z_0

It is necessary to pay attention to the influence of module length expansion and contraction. Multiplying by 622/1024 can eliminate most of the errors. $x_n = A_n\sqrt{x_0^2+y_0^2}$ Because, 622 = 512 +64 +32 +8 +4 +2 A_n $\frac{1}{A_n} = 0.607252935$ x_n $622/1024 = 0.607422 \approx \frac{1}{A_n}$

Then more accurate MOD results can be obtained later.

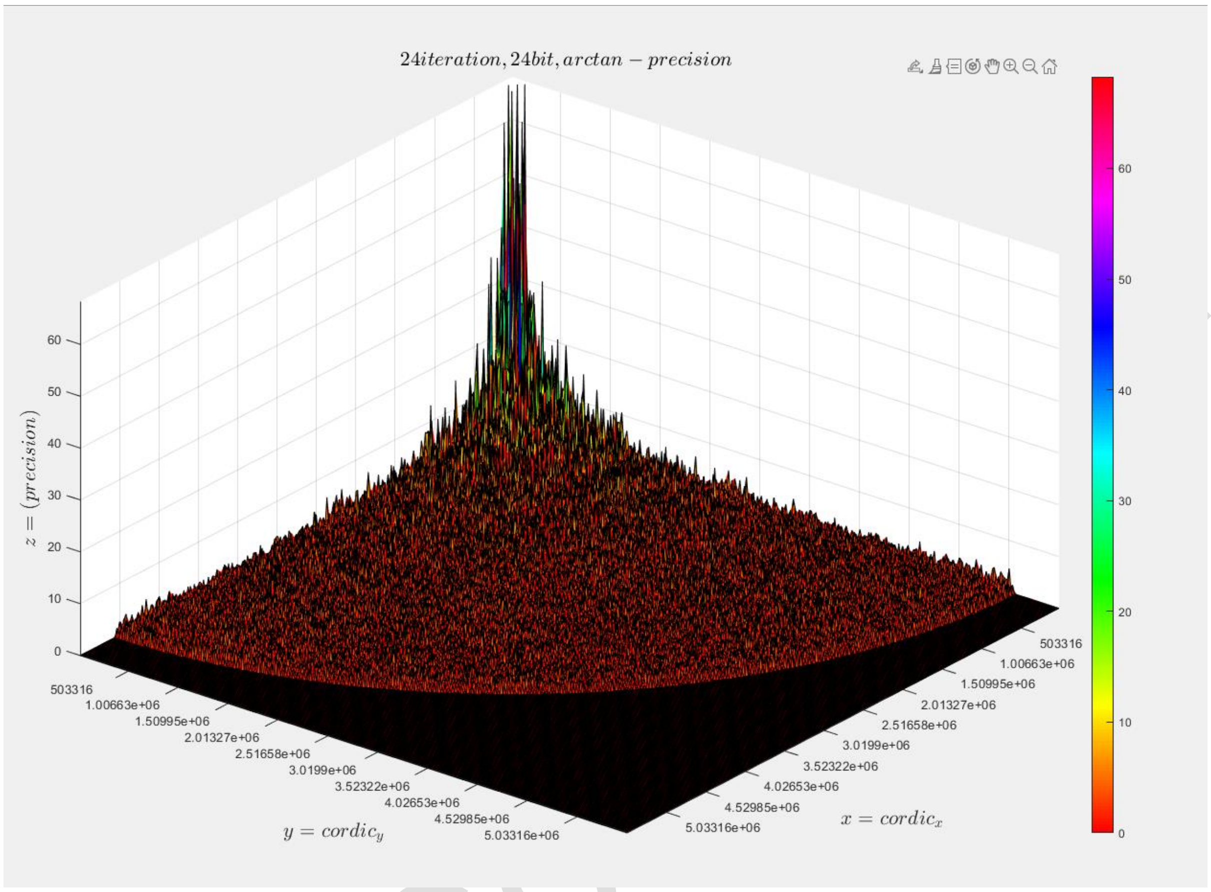
$((x_n \ll 9) + (x_n \ll 6) + (x_n \ll 5) + (x_n \ll 3) + (x_n \ll 2) + (x_n \ll 1)) \gg 10$

$$
\begin{cases}
x_{i+1} = \prod_{i=0}^{n}\cos\theta_i(x_i - y_i\tan\theta_i)\\
y_{i+1} = \prod_{i=0}^{n}\cos\theta_i(y_i + x_i\tan\theta_i)\\
z_{i+1} = \sum_{i=0}^{n} d_i\theta_i\\
d_i = \begin{cases} +1 & y_i \leqslant 0\\ -1 & y_i > 0\end{cases}
\end{cases}
\xrightarrow{\theta_i = \tan^{-1}(d_i 2^{-i})}
\begin{cases}
x_{i+1} = \prod_{i=0}^{n}\cos\theta_i(x_i - y_i d_i 2^{-i})\\
y_{i+1} = \prod_{i=0}^{n}\cos\theta_i(y_i + x_i d_i 2^{-i})\\
z_{i+1} = z_i - d_i\tan^{-1}2^{-i}\\
d_i = \begin{cases} +1 & y_i \leqslant 0\\ -1 & y_i > 0\end{cases}
\end{cases}
$$

In order to facilitate the iteration of the circuit, right shift is used instead, so that each iteration can be completed only by addition, subtraction and shift operations $\tan\theta_i$

31.3.2.4. Arctan/Modulus calculation accuracy

The figure below is a three-dimensional error statistical chart with CORIDC _ x and CORDIC _ y

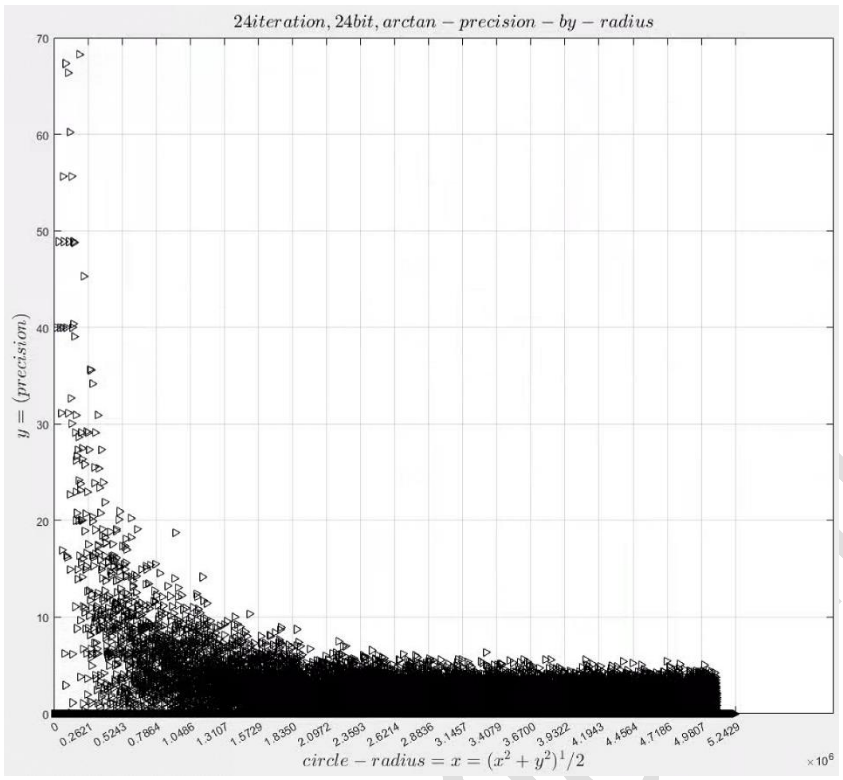


data as x and y axes, and arctan result error as z axis.

It can be seen from the figure that when the radius of the circumferential system is close to the center of the circle, the error is maximum. After the radius is greater than a certain value, an overflow occurs (the circuit processes it, and the arctan result after overflow is 0).

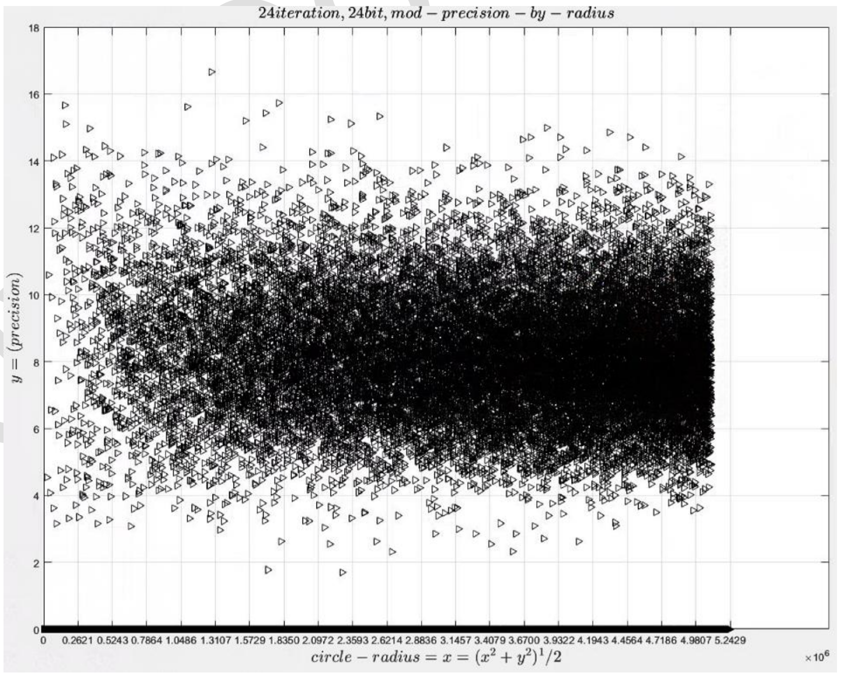
The horizontal axis of the figure below is the vector formed from the origin to the X and Y coordinates, that is, after the radius of the circle where X and Y are located is the X axis, the maximum result error corresponding to each radius is the Y axis. Draw the following radius-result error diagram. From the diagram, it can be calculated that overflow occurs when the radius exceeds 0.6072 times of the longest radius.

When the circumferential radius is too small, the error becomes large, so it is recommended to use it as a radius for calculation. $2^{22} = 4.1943 \times 10^6$



Although the above icon is 24 iterations, the input bit width is 24 bit. However, after changing the bit width of input data and the number of iterations, the errors in different situations have the same law, and users can follow the same law.

The error of MOD function result is shown in the figure below, X axis is the radius, and Y axis is



the error of MOD result

Note: In order to avoid the large error caused by the small radius of the circumferential system during calculation. Please use this number as the radius for calculations. $2^{22} = 4.1943 \times 10^6$

31.3.3. Square root operation

31.3.3.1. Do not recover remainder square root operation

The basic principle of non-recovery remainder method is:

For a 2n-bit square integer, its square root is an n-bit integer. And divide the open square into every two bits for calculation. Use D for the open square, Q for the root, and R for the remainder. The core idea of the non-recovery remainder method is to first assume the root from nth bit to 0bit to be 1, and then judge the remainder rn after this root is squared and followed by D to determine whether root 1 is correct

The number of iterations represented in the following formula of the non-recovery remainder method is the root square of the bit and the partial remainder. kRepresents the kth square root, which is the partial root of the i-th calculation, to the open square number of 2n bits, and the following operation is performed for the 2n bit integer:$q_k k r_k q_k q(\mathrm{i}) D_{2n-1} D_0$

k = n-1 $\quad r_k = D_{2k+1} D_{2k} - 01$

$r_k \geq 0$ = 1q_k

$r_k < 0$ = 0q_k

$0 \leq k$ <n-1

q_{k+1} = 1 $\quad r_k = r_{k+1} D_{2k+1} D_{2k} - q^{k+1} 01$

q_{k+1} = 0 $\quad r_k = r_{k+1} D_{2k+1} D_{2k} + q^{k+1} 01$

$r_k \geq 0$ = 1q_k

$r_k < 0$ = 0q_k

31.3.4. Operation process

31.3.4.1. CORDIC operation process

The CORDIC rotation mode calculation sin/cos operation flow is as follows:

1. Turn on the clock enable register of the coprocessor in the RCC module.
2. Configure registers. CORDIC _ CR, select input and output bit width, start mode, whether to mask interrupts, set interrupt enable, select CORDIC _ MODE = 1, and select iteration times
3. After configuring the register CORDIC _ THETA, poll the BSY flag bit or wait for interrupt or poll the ccff flag bit
4. After the calculation is completed. Read the CORDIC _ SIN, CORDIC _ COS registers and get the result

The operation process when calculating arctan/modulus in CORDIC vector mode is as follows:

1. Turn on the clock enable register of the coprocessor in the RCC module.
2. Configure registers. CORDIC _ CR, select input and output bit width, start mode, whether to mask interrupts, set interrupt enable, select CORDIC _ MODE = 0, and select iteration times
3. After configuring registers CORDIC _ X and CORDIC _ Y, poll the BSY flag bit or wait for an interrupt or poll the ccff flag bit
4. After the calculation is completed. Read the CORDIC _ ARCTAN and CORDIC _ MOD registers and get the result

31.3.4.2. SQRT operation process

The HDIV operation process is as follows:

1. Turn on the clock enable register of the coprocessor in the RCC module.

2. Configuration register CORDIC _ CR register set interrupt

3. Configure the register DSP _ RAD and write the square number (square only supports unsigned square, and there is only one start method: write to the DSP _ RAD register to start the calculation).

4. After waiting for the interrupt or polling the scff bit, read the DSP _ SQRT register to obtain the root square

31.4. Register

31.4.1. Cordic control register (CORIC _ CR)

Address: 0x00

Reset value: 0x0000 0018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARCTAN _ MOD _ OV _ mask.	CORDIC _ ER-ROR _ INT _ Mask	SQRT _ INT _ MASK	CORDIC _ INT _ MASK	Res	Res	Res	Res	Res	REGSIZE	RESSIZE	VECSIZE	Res	Res	Res	Res
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	START _ MODE	START	MODE	SQRT _ INT	CORDIC _ INT	ITERATION				
						RW	RC _W 1	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31	ARCTAN _ MOD _ OV _ MASK	RW	0	Arctan and mod functions calculate overflow interrupt masking 1: Mask the interrupt 0: The interrupt is not masked
30	CORDIC _ ERROR _ INT _ MASK	RW	0	Trigonometric function input data error interrupt masking 1: Mask the interrupt 0: The interrupt is not masked
29	SQRT _ INT _ MASK	RW	0	Square function calculation completes interrupt masking 1: Mask the interrupt 0: The interrupt is not masked
28	CORDIC _ INT _ MASK	RW	0	Trigonometric function calculation completes interrupt masking 1: Mask the interrupt 0: The interrupt is not masked
27:23	Reserved	-	-	Reserved
22	ARGSIZE	RW	0	CORDIC input data width 0: 32-bit 1: 16-bit ARGSIZE selects the number of bits used to represent the input data of the trigonometric function.

Bit	Name	R/W	Reset Value	Function
				If 32-bit data is selected, the trigonometric input register expects parameters in q1.31 format. If 16-bit data is selected, the trigonometric input register expects parameters in q1.15 format.
21	RESSIZE	RW	0	CORDIC output data width 0: 32-bit 1: 16-bit RESSIZE selects the number of bits used to represent the output data of the trigonometric function. If 32-bit data is selected, the register contains the results in q1.31 format. If 16-bit data is selected, the trigonometric output register expects parameters in q1.15 format.
20	VECSIZE	RW	0	arctan and mod mode data bit width 0: Use RESSIZE and ARGSIZE to decide 1: The bit width of arctan/mod input and output data is 24 bits. When using int32 input and output data, the upper 8 bits are the sign bit extension
19:10	Reserved	-	-	Reserved
9	START _ MODE	RW	0	CORDIC start mode selection 1: Calculate when the START register position bit is 1 0: Start calculation after performing write operations to CORDIC _ THETA, CORDIC _ X, and CORDIC _ Y registers
8	START	RC_W1	0	When STARTMODE is set to 1, this register bit is valid. At this time, when 1 is written and the BSY bit is 0, CORDIC starts a calculation. After the calculation is completed, the hardware automatically clears 0
7	CORDIC _ MODE	RW	0	CORDIC mode, 0: arctan, 1: sin/cos
6	SQRT _ INT	RW	0	Open square function computation completes interrupt enable When this bit is set to 1, an interrupt will be generated when the calculation is completed 1: An interrupt occurs when the calculation is completed 0: No interrupt is generated
5	CORDIC _ INT	RW	0	Trigonometric function calculation completes interrupt enable When this bit is set to 1, an interrupt will be generated when the calculation is completed 1: An interrupt occurs when the calculation is completed 0: No interrupt is generated
4:0	ITERATION	RW	5'd24	Iteration selection register, which can be selected from 1 to 24 iterations

31.4.2. SIN/COS input theta register (CORDIC _ theta)

Address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THETA[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**
THETA[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	THETA[31:16]	RW	0	Coprocessor sin/cos input angle register, when REGSIZE is 0 and the input register is in q1.31 format, then the THETA [31: 0] register needs to write 32 bits of data
15:0	THETA[15:0]	RW	0	Coprocessor sin/cos input angle register, when REGSIZE is 1 and the input register is in q1.15 format, then the THETA [15: 0] register needs to write 16 bits of data

31.4.3. CORDIC _ SIN Result register (CORDIC _ SIN)

Address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SIN[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIN[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	SIN[31:16]	R	0	Coprocessor SIN calculation result output register. When RESSIZE is 0 and the output register is in q1.31 format, the CODIC _ SIN [31: 0] register will read 32-bit SIN [31: 0] data
15:0	SIN[15:0]	R	0	The coprocessor SIN calculation result output register. When RESSIZE is 1 and the output register is in q1.15 format, the CODIC _ SIN [31: 0] register will read the lower 16 bits as SIN [15: 0] data and the upper 16 bits as SIN [15: 0] sign bit.

31.4.4. CORDIC _ COS Result register (CORDIC _ COS)

Address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COS[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COS[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	COS[31:16]	R	0	Coprocessor COS calculation result output register. When RESSIZE is 0 and the output register is in q1.31 format, the CODIC _ COS [31: 0] register will read 32-bit COS [31: 0] data
15:0	COS[15:0]	R	0	The coprocessor COS calculation result output register. When RESSIZE is 1 and the output register is in q1.15 format, the CODIC _ COS [31: 0] register will read the lower 16 bits as COS [15: 0] data and the upper 16 bits as COS [15: 0] data. The sign bit of COS [15: 0].

31.4.5. Arctan input coordinate register (CORDIC _ X)

Address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
X [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**
X [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	X [31: 16]	RW	0	The coprocessor arctan/module calculates the input coordinate X register. When REGSIZE is 0, the input register is in q1.31 format, and the CORDIC _ X [31: 0] register needs to write 32 bits of data
15:0	X [15: 0]	RW	0	The coprocessor arctan/module calculates the input coordinate X register. When REGSIZE is 1, the input register is in q1.15 format, and the CORDIC _ X [15: 0] register needs to write 16 bits of data

31.4.6. Arctan input coordinate register (CORDIC _ Y)

Address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Y [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**
Y [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	Y [31: 16]	RW	0	The coprocessor arctan/module calculates the input coordinate Y register. When REGSIZE is 0, the input register is in q1.31 format, and the CORDIC _ Y [31: 0] register needs to write 32 bits of data
15:0	Y [15: 0]	RW	0	The coprocessor arctan/module calculates the input coordinate Y register. When REGSIZE is 1, the input register is in q1.15 format, and the CORDIC _ Y [15: 0] register needs to write 16 bits of data

31.4.7. CORDIC Mod result register (CORDIC _ MOD)

Address: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOD[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOD[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	MOD[31:16]	R	0	Coprocessor MOD calculation result output register. When RESSIZE is 0 and the output register is in q1.31 format, CODIC _ MOD [31: 0] register will read 32-bit MOD [31: 0] data
15:0	MOD[15:0]	R	0	The output register of the coprocessor MOD calculation result. When the RESSIZE is 1 and the output register is q1.15 format, the CODIC _ MOD [31: 0] register will read out the lower 16 bits as MOD [15: 0] data and the higher 16 bits are MOD [15: 0] symbol bits.

31.4.8. ARCTAN result register (CORDIC _ ARCTAN)

Address: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARCTAN[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARCTAN[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	ARCTAN[31:16]	R	0	Coprocessor ARCTAN calculation result output register. When RESSIZE is 0 and the output register is q1.31 format, CODIC _ ARCTAN [31: 0] register will read 32-bit ARCTAN [31: 0] data
15:0	ARCTAN[15:0]	R	0	The coprocessor MOD calculates the result output register. When RESSIZE is 1 and the output register is in q1.15 format, the CODIC _ ARCTAN [31: 0] register will read the low 16 bits as ARCTAN [15: 0] data and the high 16 bits as ARCTAN [15: 0] sign bit.

31.4.9. Square root register (DSP _ RAD)

Address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RAD[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**
RAD[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	RAD	RW	32'h0	The open square number (SQRT calculation starts after writing to this register)

31.4.10. SQRT results or registers (DSP _ SQRT)

Address: 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**
CORDIC _ ARCTAN [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	-	-	Reserved, symbol extension when read out
15:0	SQRT	R	0	Square result register

31.4.11. Status register (CORIC _ SR)

Address: 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	BSY	acrf	ccef	scff	ccff
											R	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	BSY	R	0	When it is 1, it means that the CORDIC engine is performing calculations. After the calculation is completed, the hardware automatically clears 0.
3	acef	RC_W0	0	arctan and mod functions calculate overflow interrupt flags When CORDIC _ X and CORDIC _ Y are too large, causing arctan and mos calculations to overflow, the acef bit will be set to 1. The software must write 0 to clear it. Writing 1 to this bit is invalid 1: Calculate overflow interrupt generation 0: Compute overflow interrupt not generated
2	ccef	RC_W0	0	Trigonometric function input data error interrupt flag When BSY is 1, when writing data to CORDIC _ THETA, CORDIC _ X, and CORDIC _ Y, this bit is set to 1 by hardware and must be cleared by software writing 0. Writing 1 to this bit is invalid 1: Input data error interrupt generation 0: Input data error interrupt not generated
1	scff	RC_W0	0	Sqrt _ scff: Square function calculation completed interrupt flag When the calculation is complete, this bit is set to 1 by the hardware. Software must write 0 to clear, write 1 to this bit is not valid 1: Interrupt generation 0: Interrupt not generated
0	ccff	RC_W0	0	CORDIC _ ccff: CORDIC engine calculates interrupt flag When the calculation is complete, this bit is set to 1 by the hardware. Software must write 0 to clear, write 1 to this bit is not valid 1: Interrupt generation 0: Interrupt not generated

32. Debug support (DBG)

32.1. Overview

The device is built around a Cortex®-M0+ core which contains hardware extensions for advanced debugging features. The Debug extension allows the CPU to stop at a given instruction (break-point) or at data access (watchpoint). When stopped, the internal state of the CPU core and the external state of the system may be checked. Once the check is completed, the CPU Core and the system may be restored and the execution of the program continues.

The debug features are used by the debugger host when connecting to and debugging the MCUs. The debugging interface is SW-DP. The debug features embedded in the Cortex®-M0+ core are a subset of the Arm® CoreSight Design Kit.

M0 + provides integrated on-chip debug support. It is comprised of:

- SW-DP: serial wire
- BPU: Break point unit
- DWT: Data watchpoint trigger

It also includes debug features dedicated to the device:

- Flexible debug pinout assignment, SWIO @ PB6, SWCLK @ PA2
- MCU debug box (support for low-power modes, control over peripheral clocks, etc.)

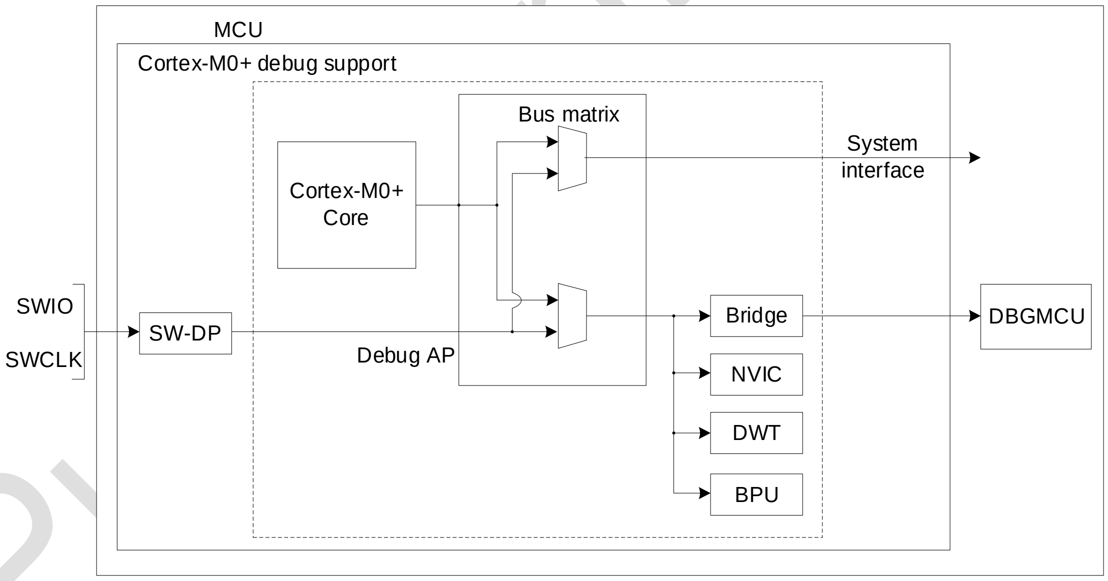


Figure 32-1 DBG block diagram

32.2. Pinout and debug port pins

32.2.1. SWD port

There are two pins related to debugging functions, which can be used as alternate functions of GPIO. These two pins are visible in all encapsulation forms.

Table 32-1 DBG block diagram

SW-DP pin name	SWD debug port pins		Pin assignment
	Type	Debug assignment	
SWDIO	I/O	Serial Wire Data Input/Output	PA13
SWDCLK	I	Serial Wire Clock	PA14

32.2.2. Flexible SWJ-DP pin assignment

After RESET (SYSRESETn or PORESETn), all pins used for the SWJ-DP are assigned as dedicated pins immediately usable by the debugger host.

In addition, the MCU offers the possibility to disable the SWD port and can then release the associated pins for general-purpose I/O (GPIO) usage

32.2.3. Internal pull-up & pull-down on SWD pins

Once the SW I/O is released by the user software, the GPIO controller takes control of these pins. The reset states of the GPIO control registers put the I/Os in the equivalent states:

- SWDIO: input pull-up
- SWCLK: input pull-down

On-chip pull-up and pull-down resistors save the need for additional resistance for the periphery.

32.3. ID codes and locking mechanism

There are several ID codes inside the MCU. It is strongly recommended the tool manufacturers (for example Keil and IAR) to lock their debugger using the MCU device ID located at address 0x40 015 800.

After the chip is powered on, the hardware reads the factory config of the flash. byte's 0x1FFF 0FF8 address, loaded into the DBG _ IDCODE register.

32.4. SWD port

32.4.1. SWD protocol introduction

This synchronous serial protocol uses two pins:

- SWCLK: clock from host to target
- SWDIO: bidirectional

The protocol allows two banks of registers (DPACC registers and APACC registers) to be read and written to. Bits are transferred LSB-first on the wire. For SWDIO bidirectional management, the line must be pulled-up on the board (100 kΩ recommended by Arm).

Each time the direction of SWDIO changes in the protocol, a turnaround time is inserted where the line is not driven by the host nor the target. By default, this steering time is 1 bit time, but the whole can be adjusted by configuring the SWCLK frequency.

32.4.2. SWD protocol introduction

Each sequence consist of three phases:

- Packet request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 32-2 Request packet (8-bits)

Bit	Acronym	Description
0	Start	Must be “1”
1	ApnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request 1: Read request
4:3	A[3:2]	Address field of the DP or AP registers
5	Parity	The check bit of the previous bit
6	Stop	0
7	Park	Not driven by the host. 1 is read out by the chip due to the pull-up attrib-ute.

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drive the line.

Table 32-3 ACK response (3 bits)

Bit	Acronym	Description
[2:0]	ACK	001: FAULT 010: WAIT 100: OK

The ACK Response must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 32-4 DATA Transfer (33bits)

Bit	Acronym	Description
[31:0]	WDATA or RDATA	Write or Read data
32	Parity	Single parity of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

32.4.3. SW-DP state machine (reset, idle states, ID code)

The State Machine of the SW-DP has an internal ID code which identifies the SW-DP. It follows the JEP-106 standard. This ID code is the default Arm one and is set to 0x0BB11477 (correspond-ing to Cortex®-M0+). Note: xxxxxx

32.4.4. DP and AP read/write accesses

- The operation of reading DP will not be posted: the chip response can be immediate (ACK = OK), or can be delayed (ACK = WAIT)
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. If the next access to be made is not an AP access, the DP-RDBUFF register must be called out to obtain this result.
- The READOK flag of the DP-CTRL/STAT register is updated on every AP read accessor RDBUFF read request to know if the AP read access was successful.
- SW-DP implements a write buffer (for DP and AP writes), which receives a write operation even when other operations are still incomplete. If the write buffer is full, the chip answer re-sponse is "WAIT". IDCODE read, CTRL/STAT read or ABORT write are exceptions (even if the write buffer is full)
- Since SWCLK and HCLK are asynchronous clocks, two additional SWCLK cycles are re-quired after the write operation (after the check bit) to ensure the internal validity of the write. These cycles should be applied when the drive signal line is low.

- The above is particularly important when writing CTRL/STAT for power-on requests. fail if the next action (which requires power-up) occurs immediately.

32.4.5. SW-DP registers

Access to these registers are initiated when APnDP=0

A[3:2]	R/W	CTRLSEL bit or SELECT registers	Register	Notes
00	Read		IDCODE	
00	Write		ABORT	
01	Read/Write	0	DP-CTRL/STAT	
01	Read/Write	1	WIRE CONTROL	
10	Read		READ RESEND	
10	Write		SELECT	
11	Read/Write		READ BUFFER	

32.4.6. SW-AP registers

Address	A [3: 2] value	Description
0x0	00	Reserved
0x4	01	DP CTRL/STAT register, used as ■ Request a system or debug power-up ■ Configure transport operations for AP access ■ Control pushed comparison and pushed verification operations ■ Read some status flags (overflow, power-up response)
0x8	10	DP SELECTRION register: Used as a register to select the current access port and active 4 words in the window. ■ Bit 31: 24: APSEL: Select current AP ■ Bit 23: 8: reserved ■ Bit 7: 4: APBANKSEL: In the current AP, select active 4 word register windows ■ Bit 3: 0: reserved
0xC	11	DP RDBUFF register: Used to provide the debugger with the final result after a sequence of operations (without requesting a new JTAG-DP operation)

32.5. Core debug

Core debugs can be accessed through the Core debug register. Debug access to these registers is by means of the debug access port. He consists of the following four registers

Table 32-5 Core debug registers

Register	Description
DHCSR	32bit Debug halting control and status register
DCRSR	17bit Debug Core register selector register
DHCDR	32bit debug Core register Data register
DEMCR	32bit debug exception and monitor control register

These registers are not reset by a system reset. They will be powered on and reset, reset off. In order to Hart on reset, you need:

- bit0 (VC _ CORRESET) of the debug and exception monitoring control register, enabled
- Debug stop control and status register, enabled

32.6. BPU (Break Point Unit)

The Cortex®-M0+ BPU implementation provides four breakpoint registers. BPU is a set of ARMv7-M flash patches and breakpoint (FPB) Blocks (Cortex-M3 & Cortex-M4).

32.6.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.

Refer to the ARMv6-M ARM and ARM Coresight Components Technical Reference Manual for more information on BPU Coresight's identity registers and their addresses and access kinds.

32.7. DWT (Data Watchpoint)

The Cortex-M0 DWT implementation provides 2 watchpoint registers.

32.7.1. DWT functionality

The processor watchpoints implement PC based watchpoint functionality.

32.7.2. DWT program counter sample register

A processor that implements the data watchpoint unit also implements the ARMv6-M optional DWT Program Counter Sample Register (DWT_PCSR). This register permits a debugger to periodically sample the PC without halting the processor. This mechanism provides coarse-grained analysis.

The Cortex®-M0+ DWT_PCSR records both instructions that pass their condition codes and those that fail.

32.8. MCU debug component (DBG)

The MCU debug component helps debuggers provide the following support:

- Low-power modes
- Clock control of timer and watchdog during breakpoint

32.8.1. Debug support for low-power modes

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU implements several low-power modes which can either deactivate the CPU clock or reduce the power of the CPU.

The CPU isdisabled to stop FCLK or HCLK during debug. Since these are the needs of the debugger connection, they must be kept on during a debugging period. The MCU integrates special means to allow the user to debug software in low-power modes.

Therefore, the debugger host must first set the contents of some debug configuration registers to change the low-power behavior:

- In sleep mode: FCLK and HCLK still work. Accordingly, this mode does not cause any restrictions on standard debugging functionality.
- In stop mode: The DBG _ stop bit must be set in advance by the debugger.

32.8.2. Supports debugging of timers, watchdogs, bxCAN and I2C

During a breakpoint, it is necessary to choose how the timer's counter and watchdog want to behave:

- They can continue to count in breakpoint. This is typically required, for example, when a PWM is controlling a motor.
- They can stop and count inside the breakpoint. This is determined by the characteristics of watchdog.

32.9. DBG register

32.9.1. DBG device ID code register (DBG _ IDCODE)

Address offset: 0x00

Only 32-bit access supported. Read-only

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD
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**
TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD	TBD
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	Reserved	-	-	Reserved

32.9.2. DBG configuration register (DBG_CR)

This register configures the low-power modes of the MCU under debug.

It is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support this feature, it is still possible for the user software to write to this register.

Address offset: 0x04

Reset value: 0x0000 0000 (will not be reset by system reset)

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	DBG_ STOP	DBG _ SLEEP
														RW	RW

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	Reserved
1	DBG_STOP	RW	0	Debug stop mode. 0: (FCLK = off, HCLK = off). In STOP mode, the clock controller disables all clocks (including HCLK and FCLK). When exiting from STOP mode, the clock configuration is identical to the one after RESET (CPU clocked by the internal RC oscillator (HSI)). Consequently, the software must reprogram the clock controller. 1: (FCLK = on, HCLK = on). In this case, when entering STOP mode, FCLK and HCLK are provided by the internal RC oscillator (HSI). 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=Off, 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.

32.9.3. DBG APB freeze register 1 (DBG _ APB _ FZ1)

This register configures the clocking of timers and IWDG of the MCU under debug: The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Address offset: 0x08

Power on Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_LPTIM_STOP	Res	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**
Res	Res	Res	DBG_IWDG_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res		DBG_TIM2_STOP
			RW												RW

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM_STOP	RW	0	Clocking of LPTIM counter when the core is halted 0: Enable 1: Disable
30：13	Reserved	-	-	Reserved
12	DBG_IWDG_STOP	RW	0	The clock control bit of the IWDG counter when the CPU stops 0: Enable 1: Disable
11	DBG _ WWDG _ STOP	RW	0	The clock control bit of the WWDG counter when the CPU stops 0: Enable 1: Disable
10	DBG _ RTC _ STOP	RW	0	The clock control bit of the RTC counter when the CPU stops 0: Enable 1: Disable
9:2	Reserved	-	-	Reserved
0	DBG_TIM2_STOP	RW		Clocking of TIM2 counter when the core is halted 0: Enable 1: Disable

32.9.4. DBG APB free register 2 (DBG _ APB _ FZ2)

This register configures the clocking of timer counters when the MCU is under debug. The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Address offset: 0x0C

Power on Reset value: 0x0000 0000

Only 32-bit access supported. Read-only

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	DBG_TIM17_STOP	DBG_TIM16_STOP	Res
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

DBG_ TIM14_ST OP	Re s	Re s	Re s	DBG_ TIM1_ST OP	Re s	Re s	Re s	Re s	Re s	Re s	Re s	Re s	Res	Res	Re s
RW				RW											

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	DBG_TIM17_STOP	RW		Clocking of TIM17 counter when the core is halted 0: Enable 1: Disable
17	DBG_TIM16_STOP	RW		Clocking of TIM16 counter when the core is halted 0: Enable 1: Disable
16	Reserved	-	-	Reserved
15	DBG_TIM14_STOP	RW		Clocking of TIM14 counter when the core is halted 0: Enable 1: Disable
14:12	Reserved	-	-	Reserved
11	DBG_TIM1_STOP	RW		Clocking of TIM1 counter when the core is halted 0: Enable 1: Disable
10:0	Reserved	-	-	Reserved

33. Version history

Version	Date	Descriptions
V0.5	2025.05.30	Initial version

Puya Semiconductor Co., Ltd.