PY32F07X Series

32-bit ARM® Cortex®-M0+ Microcontrollers

Reference Manual

Puya Semiconductor (Shanghai) Co., Ltd

Contents

1. List of abbreviations for register

Abbreviation	Describe
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. Reading this bit returns the reset value.
Read/clear write0 (rc_w0)	Software can read as well as clear this bit by writing 0. Writing ‘1’ has no effect on the bit value.
Read/clear write1 (rc_w1)	Software can read as well as clear this bit by writing 1. Writing ‘0’ has no effect on the bit value.
Read/clear write (rc_w)	Software can read as well as clear this bit by writing register. Writing to this bit has no effect.
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 clears it to ‘0’. Writing this bit has no effect on the bit value.
Read/set (rs)	Software can read as well as set this bit to ‘1’. Writing ‘0’ has no effect on the bit value.
Toggle (t)	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

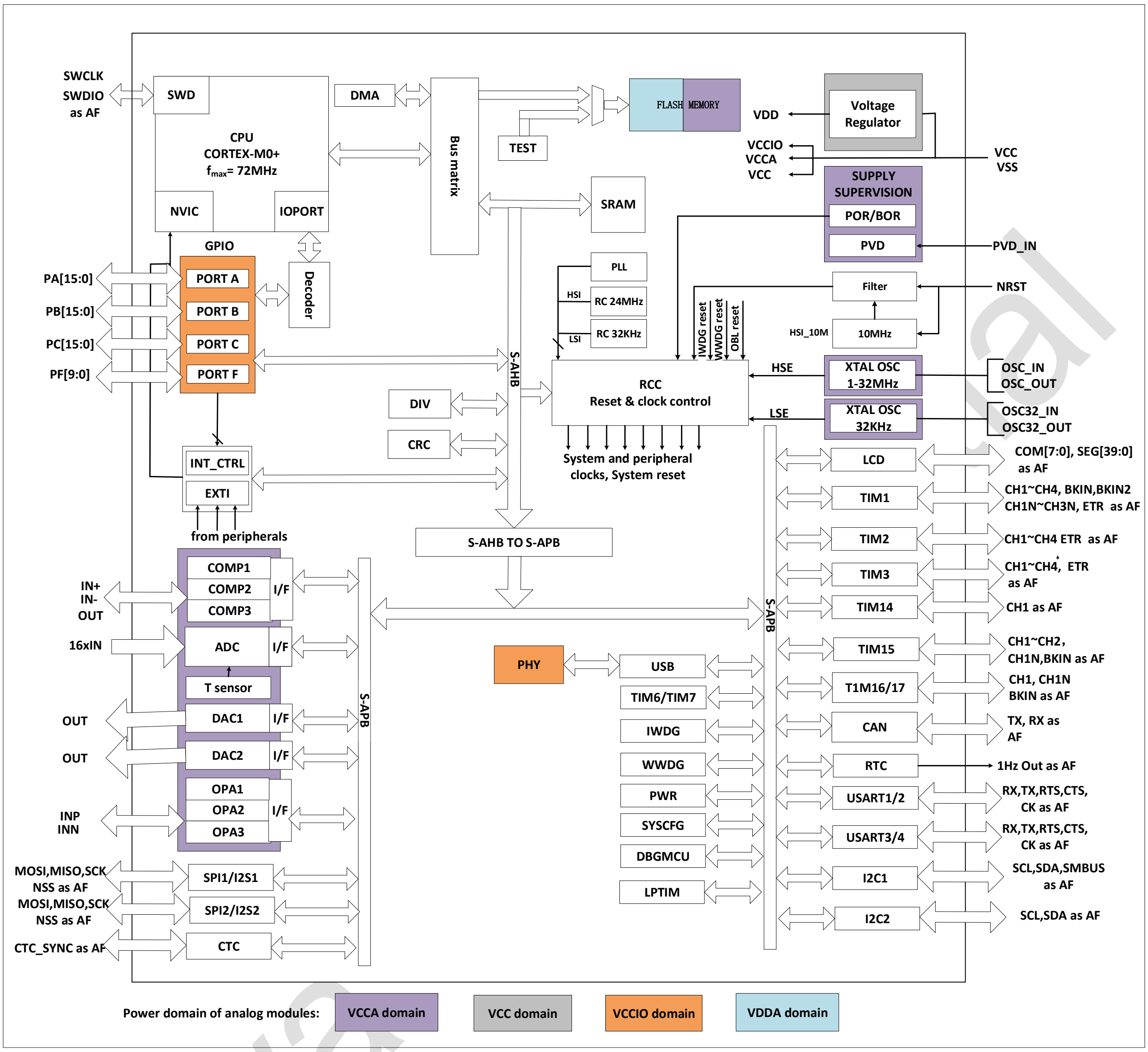


Figure 2-1 System architecture

3. Memory and bus architecture

3.1. System architecture

The system consists of the following parts:

- Two masters:
  - Cortex-M0+
  - General-purpose DMA
- Three Slaves
  - Internal SRAM
  - Internal Flash memory
  - AHB with AHB-APB Bridge

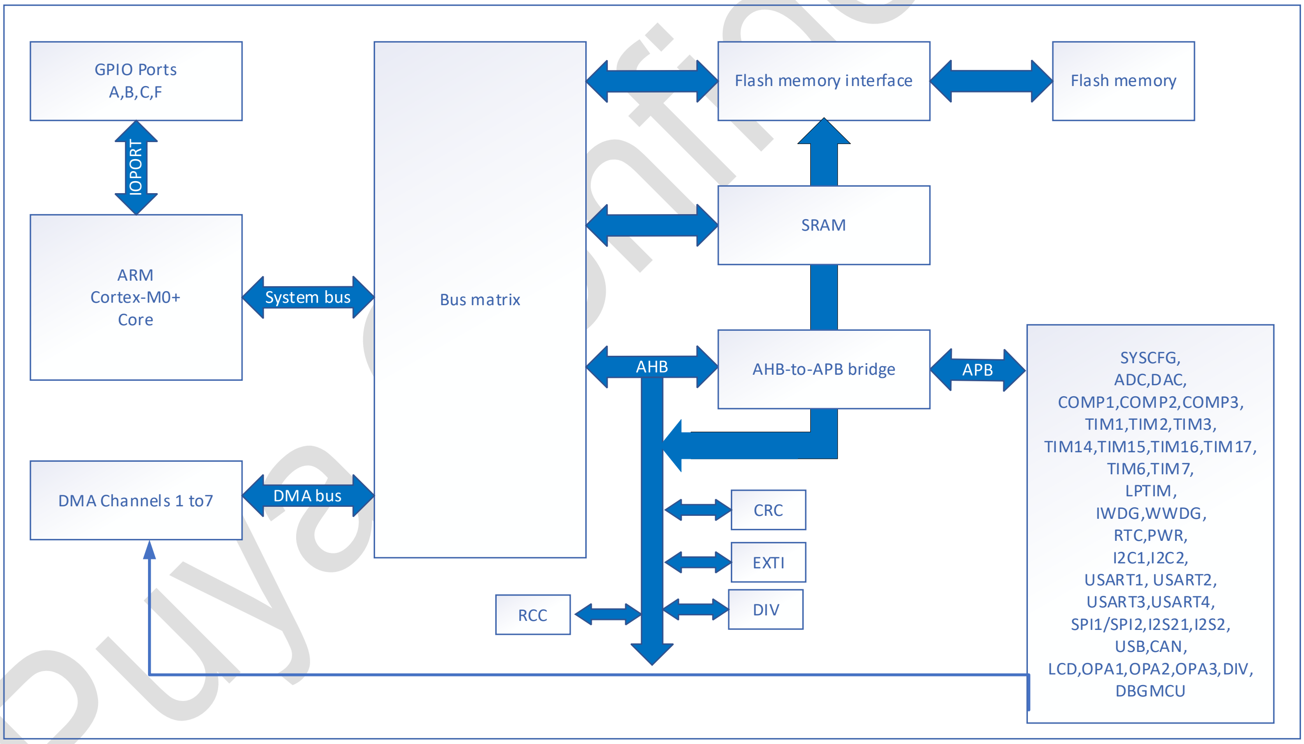


Figure 3-1 System architecture

- System bus

This bus connects the Cortex-M0+ system bus to the bus Matrix, which is used to manage the CPU and DMA arbitration.

- DMA bus

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

- Bus Matrix

The Bus Matrix is responsible for bus arbitration of the CPU bus and the DMA bus. This arbitration uses the Round Robin algorithm. The Bus Matrix consists of Masters (CPU, DMA) and slaves (Flash memory, SRAM and AHB-to-APB bridge).

- AHB-to-APB bridge (APB)

The AHB-to-APB bridge is responsible for the synchronization between the AHB and APB buses and the mapping of peripheral addresses.

3.2. Memory organization

Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbytes address space. The bytes are coded in memory in Little Endian format (in a word, the lowest numbered byte is considered the world’s least sinificant byte).

The addressable memory space is divided into 8 mian blocks, each of 512 Mbyte.

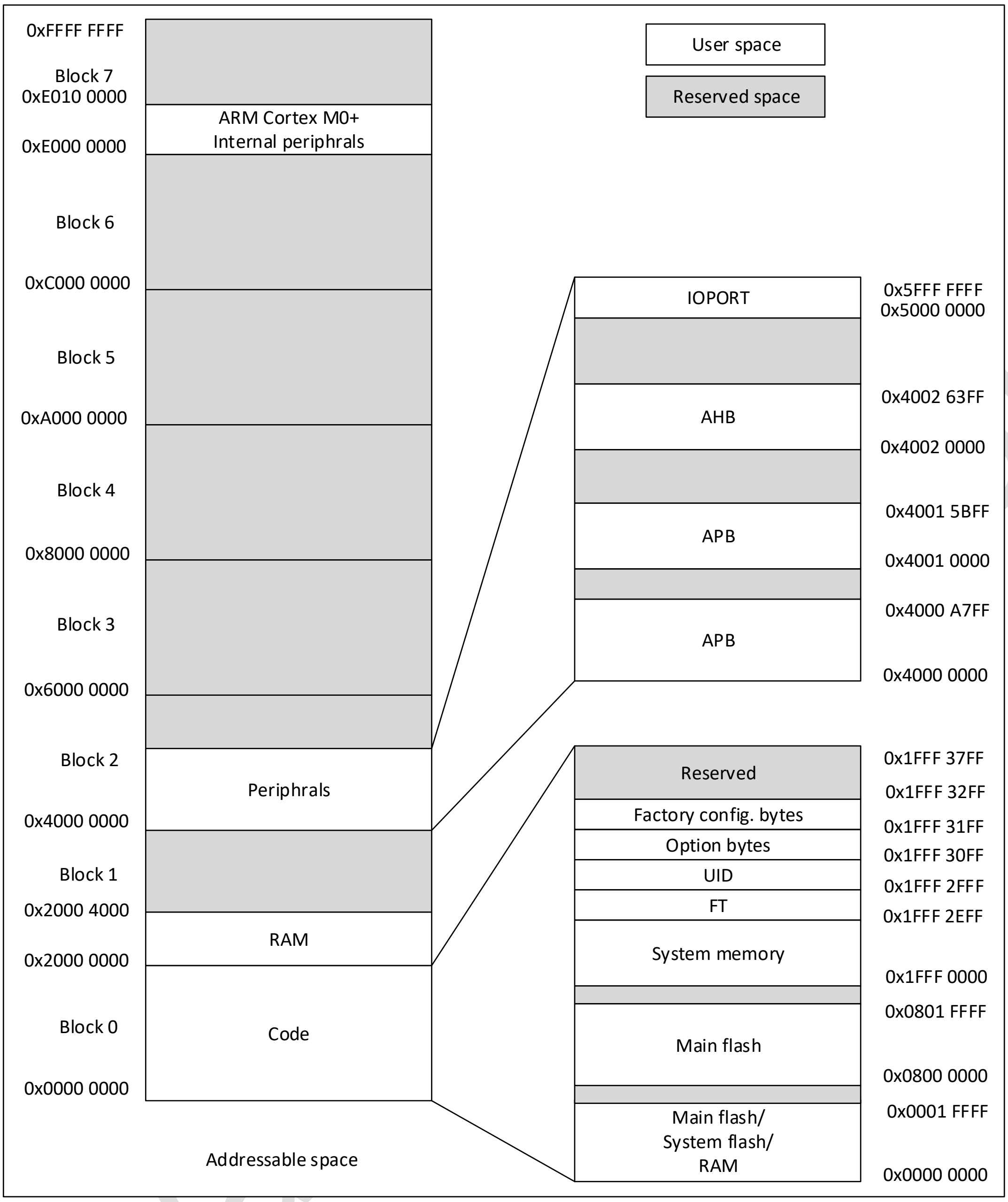


Figure 3-2 Memory map

Table 3-1 Memory boundary addresses

Type	Boundary Address	Size	Memory Area	Description
SRAM	0x2000 4000-0x3FFF FFFF	-	Reserved	-
	0x2000 0000-0x2000 3FFF	16 KBytes	SRAM	Maximum SRAM is 16 KBytes
Code	0x1FFF 3400-0x1FFF FFFF	-	Reserved	-
	0x1FFF 3300-0x1FFF 33FF	-	Reserved	Reserved
	0x1FFF 3200-0x1FFF 32FF	256 Bytes	FT infor0 bytes	Factory config
	0x1FFF 3100-0x1FFF 31FF	256 Bytes	Option bytes	Chip hardware and software Option bytes information

Type	Boundary Address	Size	Memory Area	Description
	0x1FFF 3000-0x1FFF 30FF	256 Bytes	UID bytes	Unique ID
	0x1FFF 2F00-0x1FFF 2FFF	256 Bytes	FT bytes	FT bytes
	0x1FFF 0000-0x1FFF 2EFF	11.75 KBytes	System memory	Store the boot loader
	0x0802 0000-0x1FFE FFFF	-	Reserved	-
	0x0800 0000-0x0801 FFFF	128 KBytes	Main flash memory	-
	0x0002 0000-0x07FF FFFF	-	Reserved	-
	0x0000 0000-0x0001 FFFF	128 KBytes	According to the Boot configuration: 1) Main Flash memory 2) System memory 3) SRAM	-

1. The above spaces are marked as reserved spaces, which cannot be written and read as 0 with a response error occurs.

Table 3-2 Peripheral register address

Bus	Boundary Address	Size	Peripheral
	0xE000 0000-0xE00F FFFF	1 MBytes	M0+
IOPORT	0x5000 1800-0x5FFF FFFF	256 MBytes	Reserved(1)
	0x5000 1400-0x5000 17FF	1 KBytes	GPIOF
	0x5000 1000-0x5000 13FF	1 KBytes	Reserved
	0x5000 0C00-0x5000 0FFF	1 KBytes	Reserved
	0x5000 0800-0x5000 0BFF	1 KBytes	GPIOC
	0x5000 0400-0x5000 07FF	1 KBytes	GPIOB
	0x5000 0000-0x5000 03FF	1 KBytes	GPIOA
AHB	0x4002 4000-0x4FFF FFFF	256 MBytes	Reserved
	0x4002 3C00-0x4002 3FFF	1 KBytes	Reserved
	0x4002 3800-0x4002 3BFF	1 KBytes	DIV
	0x4002 3400-0x4002 37FF	1 KBytes	Reserved
	0x4002 3000-0x4002 33FF	1 KBytes	CRC
	0x4002 2400-0x4002 2FFF	3 KBytes	Reserved
	0x4002 2000-0x4002 23FF	1 KBytes	FLASH
	0x4002 1C00-0x4002 1FFF	1 KBytes	Reserved
	0x4002 1800-0x4002 1BFF	1 KBytes	EXTI
	0x4002 1400-0x4002 17FF	1 KBytes	Reserved
	0x4002 1000-0x4002 13FF	1 KBytes	RCC(2)
	0x4002 0400-0x4002 0FFF	3 KBytes	Reserved
	0x4002 0000-0x4002 03FF	1 KBytes	DMA
APB	0x4001 5C00 - 0x4001 FFFF	41 KBytes	Reserved

Bus	Boundary Address	Size	Peripheral
	0x4001 5800 - 0x4001 5BFF	1 KBytes	DBG
	0x4001 4C00 - 0x4001 57FF	3 KBytes	Reserved
	0x4001 4800 - 0x4001 4BFF	1 KBytes	TIM17
	0x4001 4400 - 0x4001 47FF	1 KBytes	TIM16
	0x4001 4000 - 0x4001 43FF	1 KBytes	TIM15
	0x4001 3C00 - 0x4001 3FFF	1 KBytes	Reserved
	0x4001 3800 - 0x4001 3BFF	1 KBytes	USART1
	0x4001 3400 - 0x4001 37FF	1 KBytes	Reserved
	0x4001 3000 - 0x4001 33FF	1 KBytes	SPI1/I2S1
	0x4001 2C00 - 0x4001 2FFF	1 KBytes	TIM1
	0x4001 2800 - 0x4001 2BFF	1 KBytes	Reserved
	0x4001 2400 - 0x4001 27FF	1 KBytes	ADC
	0x4001 0400 - 0x4001 23FF	8 KBytes	Reserved
	0x4001 0300 - 0x4001 03FF	1 KBytes	OPA
	0x4001 0200 - 0x4001 02FF		COMP
	0x4001 0000 - 0x4001 01FF		SYSCFG
	0x4000 8000- 0x4000 FFFF	32 KBytes	Reserved
	0x4000 7C00 - 0x4000 7FFF	1 KBytes	LPTIM1
	0x4000 7800 - 0x4000 7BFF	1 KBytes	Reserved
	0x4000 7400 - 0x4000 77FF	1 KBytes	DAC
	0x4000 7000 - 0x4000 73FF	1 KBytes	PWR[3]
	0x4000 6C00 - 0x4000 6FFF	1 KBytes	CTC
	0x4000 6800 - 0x4000 6BFF	1 KBytes	Reserved
	0x4000 6400 - 0x4000 67FF	1 KBytes	CAN
	0x4000 6000 - 0x4000 63FF	1 KBytes	USB SRAM
	0x4000 5C00 - 0x4000 5FFF	1 KBytes	USB
	0x4000 5800 - 0x4000 5BFF	1 KBytes	I2C2
	0x4000 5400 - 0x4000 57FF	1 KBytes	I2C1
	0x4000 5000 - 0x4000 53FF	1 KBytes	Reserved
	0x4000 4C00 - 0x4000 4FFF	1 KBytes	USART4
	0x4000 4800 - 0x4000 4BFF	1 KBytes	USART3
	0x4000 4400 - 0x4000 47FF	1 KBytes	USART2
	0x4000 3C00 - 0x4000 43FF	2 KBytes	Reserved
	0x4000 3800 - 0x4000 3BFF	1 KBytes	SPI2/I2S2
	0x4000 3400 - 0x4000 37FF	1 KBytes	Reserved
	0x4000 3000 - 0x4000 33FF	1 KBytes	IWDG
	0x4000 2C00 - 0x4000 2FFF	1 KBytes	WWDG
	0x4000 2800 - 0x4000 2BFF	1 KBytes	RTC

Bus	Boundary Address	Size	Peripheral
	0x4000 2400 - 0x4000 27FF	1 KBytes	LCD
	0x4000 2000 - 0x4000 23FF	1 KBytes	TIM14
	0x4000 1800 - 0x4000 1FFF	2 KBytes	Reserved
	0x4000 1400 - 0x4000 17FF	1 KBytes	TIM7
	0x4000 1000 - 0x4000 13FF	1 KBytes	TIM6
	0x4000 0800 - 0x4000 0FFF	2 KBytes	Reserved
	0x4000 0400 - 0x4000 07FF	1 KBytes	TIM3
	0x4000 0000 - 0x4000 03FF	1 KBytes	TIM2

1. IOPORT, AHB, APB marked as Reserved address space, cannot be writen, read back to 0, will not generate hardfault.
2. Not only supports 32 bits word access, but also supports halfword and byte access.
3. Not only supports 32 bits word access, but also supports halfword access.

3.3. Embedded SRAM

The PY32F072 features up to 16 KBytes of SRAM. It can be accessed as bytes, half-word (16 bits) or full words (32 bits). A hard fault will be generated when the software reads and writes the space outside the setting range.

3.4. Flash memory

Flash memory consists of two physical areas:

- Main Flash area, 128 KBytes, it contains application and user data. Software access to spaces outside the set range generates hard fault.
- Information area, 14 KBytes, it includes the following parts:
  - FT infor0 bytes: 256 Bytes, used to store Normal and High TS DATA, HSI Re-trimming data
  - Option bytes: 256 Bytes, used to store chip software and hardware Option Bytes information
  - UID：256 Bytes, used to store the UID of the chip
  - System memory:11.75 KBytes, used to store Boot loader

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

3.5. Boot mode

Three different boot mode can be selected through the BOOT0 pin and boot selector option bit nBOOT1 (stored in the Option bytes), as shown in the following table:

Table 3-3 Boot mood

Boot mode configuration		Mode
nBOOT1 bit	BOOT0 pin	
X	0	Main Flash memory is selected as the boot area
1	1	System memory is seleced as the boot area
0	1	Embedded SRAM is selected as the boot area

The values on the Boot pins are latched on the 4th SYSCLK after a reset. It is up to the user to set the boot mode to choose according to the table above.

The CPU then takes the value at the top of the stack from address 0x0000 0000, then starts code executes from the boot memory starting from 0x0000 0004. Depending on the selected boot mode, main Flash memory, system memory or SRAM is accessible as follows:

- Boot from main Flash memory: 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 word, 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 still accessible from its original memory space (0x1FFF 0000).
- Boot from the embedded SRAM: the SRAM is aliased in the boot memory space (0x0000 0000), but still accessible at address 0x2000 0000.

3.5.1. Memory physical mapping

If boot mode is selected, the application software can modify the memory accessible in the program space. This modification is determined by the MEM_MODE bit selection in the SYSCFG_CFGR1 register (see the SYSCFG chapter for details).

3.5.2. Embedded boot loader

The embedded boot loader is located in the System memory, programmed during production. It is used to reprogram the Flash memory using the following serial interface:

- USART1， PA9/PA10； USART2,， PA4/PA15； USART3， PB10/PB11； USART4， PC10/PC11；

- USB，PA11/PA12

4. Embedded Flash memory

4.1. Key features

- Main flash block: maximum 128 KBytes
- Information block: 14 KBytes
- Page size: 256 Bytes
- Sector size: 8 KBytes

The Flash control interface circuit features:

- Flash write and erase
- Programming operations of option bytes
- Read protection
- Write protection
- SDK protection

4.2. Flash memory function introduction

4.2.1. Flash structure

Flash memory is composed of 64-bit wide storage units, which can be used for program and data storage. Page size is 256 Bytes, Sector size is 8 KBytes.

In terms of function, Flash memory is divided into Main flash and information flash, the former has a maximum capacity of 128 KBytes, and the latter has a capacity of 14 KBytes.

Table 4-1 Flash structure and boundary addresses

Block	Sector	Page	Base address	Size
Main flash	Sector 0	Page 0-31	0x0800 0000-0x0800 1FFF	8 KBytes
	Sector 1	Page 32-63	0x0800 2000-0x0800 3FFF	8 KBytes
	Sector 2	Page 64-95	0x0800 4000-0x0800 5FFF	8 KBytes
	…	…	…	…
	Sector 14	Page 448-479	0x0801 C000-0x0801 DFFF	8 KBytes
	Sector 15	Page 480-511	0x0801 E000-0x0801 FFFF	8 KBytes
System flash	INFO	Page 0-46	0x1FFF 0000-0x1FFF 2EFF	11.75 KBytes
FT		Page 47	0x1FFF 2F00-0x1FFF 2FFF	256 bytes
UID		Page 48	0x1FFF 3000-0x1FFF 30FF	256 bytes
Option bytes		Page 49	0x1FFF 3100-0x1FFF 31FF	256 bytes
Factory config		Page 50	0x1FFF 3200-0x1FFF 32FF	256 bytes

Block	Sector	Page	Base address	Size
Reserved		Page 51-55	0x1FFF 3300-0x1FFF 37FF	1280 bytes

4.2.2. Flash read operation and access latency

Flash can be used as a general memory space to accessed direct addressing. The contents of the Flash memory can be read through a special read control sequence. The instruction fetch and data access are both done through the AHB bus. Read can mange through the Latency of the FLASH_ACR register, which is the read operation increase the wait state or not.

FLASH_ACR.0(LATENCY) bit, When it is 0, the wait state of the Flash read operation is not added. When it is 1, the Flash read operation adds one wait state. When it is 2, the Flash read operation adds two wait states. This mechanism is specially designed to match high-speed system clock and relatively low-speed Flash read speed.

4.2.3. Flash program and erase operations

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

ICP: It is 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 MCU. ICP provides quick and efficient design iterations and eliminates unnecessary package handling or socketing of devices.

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

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

During a program and erase operations to the Flash memory, any attempt to read the Flash memory will stall the bus. The read operation will proceed correctly once the program and erase operations has completed. This means that code or data fetches cannot be made while programming and erasing operations are in progress.

For program and erase operations, the HSI must be turned on.

4.2.3.1. Unlocking the Flash memory

After reset, the Flash memory is protected against unwanted (like caused by electrical interference) write or erase operations. The FLASH_CR register is not accessible in write mode, except for the OB L_LAUNCH bits, used to reload option bit. Every time to write or erase the Flash, must write the FLASH_KEYR register, to generate an unlock sequence, and to open the access to the FLASH_CR register.

This sequence consists of two steps:

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

Step 2: Write KEY2 = 0xCDEF 89AB to the 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 Hard Fault 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.

In addition, the FLASH_CR register cannot be written when the BSY bit of the FLASH_SR register is set. In the meantime, 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

The Flash memory can be programmed the entire page in units of 32 bits each time (hardfault will be generated when the half word or byte operation is performed). 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 non 32-bit write will cause a hard fault interrupt.

If the address is write-protected by the FLASH_WRPR register, the program operation is skipped and a warning is issued by the WRPERR bit in the FLASH_SR register. At the end of the program operation, the EOP bit in the FLASH_SR register will be set.

The Flash memory programming sequence is as follows:

- Check that no Flash memory operation is ongoing by checking the BSY in the FLASH_SR register.
- If no Flash memory erase or program operation is ongoing, the software reads out the 64 words of the page (if the page already has data stored, perform this step, otherwise skip this step).
- To release the protection of the FLASH_CR register by programming KEY1 and KEY2 to the FLASH_KEYR register.
- Set the PG bit and the EOPIE bit in the FLASH_CR register.
- Programming to the target address from the 1st to 63rd word (only accept 32 bits program).
- Set the PGSTRT in FLASH_CR register.
- Write the 64th word.
- Wait until the BSY bit of the FLASH_SR register to be cleared.
- Check the EOP flag in the FLASH_SR register (It is set when the programming operation has succeeded), and then clear it by software.
- If there are no more program operations, software will clear the PG bit.

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

Flash Erase Operation

The Flash memory can be erased by page, or sector and mass erase.

4.2.3.3. Page erase

When a page is protected by WRP, it will not be erased and the WRPERR bit is set at this time. To excution the page erase operation, the following steps need to be performed:

1) Check that no Flash memory operation is ongoing by checking the BSY in the FLASH_SR register.
2) To release the protection of the FLASH_CR register by programming KEY1 and KEY2 to the FLASH_KEYR register.
3) Set the PER bit and the EOPIE bit in the FLASH_CR register.
4) Write arbitrary data (32-bit data) to the page.
5) Wait for the BSY bit to be cleared.
6) Check that the EOP flag is set.

7) Clear the EOP flag.

4.2.3.4. Mass erase

The Mass erase can used to completely erase the entire main Flash memory. Additionally, when WRP is enabled, the mass erase function is disabled and no mass erase operation occurs, the WRPERR bit is set.

The following sequence for mass erase:

- Check that no Flash memory operation is ongoing by checking the BSY.
- To release the protection of the FLASH_CR register by programming KEY1 and KEY2 to the FLASH_KEYR register.
- Set the MER bit and the EOPIE bit in the FLASH_CR register.
- Write arbitrary data (32-bit data) to the main Flash memory.
- Wait for the BSY bit to be cleared.
- Check that the EOP flag is set.
- Clear the EOP flag.

4.2.3.5. Sector erase

The sector erase can be used to erase the main Flash of 8 KBytes. In addition, when a sector is protected by WRP, it will not be erased, and the WRPERR bit is set.

The following sequence for sector erase:

- Check that no Flash memory operation is ongoing by checking the BSY.
- To release the protection of the FLASH_CR register by programming KEY1 and KEY2 to the FLASH_KEYR register.
- Set the SER bit and the EOPIE bit in the FLASH_CR register.
- Write arbitrary data to the sector.
- Wait for the BSY bit to be cleared.
- Check that the EOP flag is set.
- Clear the EOP flag.

4.2.3.6. Program and erase time configuration

Flash write and erase times need to be tightly controlled, otherwise the operation will fail. If you need to write and erase the Flash, you need to FLASH_PERTPE, FLASH_SMERTPE, FLASH_PRGTPE, FLASH_PRETPE to configure the Flash Write and Erase time control registers according to the HSI output frequency.

4.3. Flash option byte

4.3.1. Flash option word

Part of the information area is used as an option byte, which is used to store the hardware configuration that the chip or the user needs to perform for the application. For example, the watchdog can be selected in hardware or software mode.

For data security, the option bytes are stored separately in the code and one's complement code.

Table 4-2 Option byte format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Complemented Option byte 1								Complemented 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 option bytes can be read from the memory locations listed in the table option byte organization or from the relevant registers of the following option bytes :

- FLASH user option register (FLASH_OPTR)
- FLASH SDK area address register (FLASH_SDKR)
- FLASH WRP address register (FLASH_WRPR)

Table 4-3 Option byte organization

Word address	Describe
0x1FFF 3100	Option byte for Flash User option and its complemented
0x1FFF 3108	Option byte for Flash SDK area address and its complemented
0x1FFF 3110	Reserved
0x1FFF 3118	Option byte for Flash WRP address and its complemented
0x1FFF 3120	Reserved
0x1FFF 3128	Reserved
…	Reserved
…	Reserved
…	Reserved

0x1FFF 31F8	Reserved

1) Option byte for Flash User option

Flash memory address: 0x1FFF 3100

Production value:0x2755 D8AA

After the power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option bytes 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
~IWDG_STOP	~ nBOOT1	~ NRST_MODE	~ WWDG_SW	~ IWDG_SW	Res	Res	Res	~RDP[7:0]							
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**
IWDG_STOP	nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	Res	Res	Res	RDP[7:0]							
R	R	R	R	R				R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~IWDG_STOP	R	One's complement of IWDG_STOP
30	~ nBOOT1	R	One's complement of nBOOT1
29	~ NRST_MODE	R	One's complement of NRST_MODE
28	~ WWDG_SW	R	One's complement of WWDG_SW
27	~ IWDG_SW	R	One's complement of IWDG_SW
26:24	Resrved	-	Resrved
23:16	~RDP	R	One's complement of RDP
15	IWDG_STOP	R	with setting iwdg timer running state in stop mode 0: freze timer 1: normal operation
14	nBOOT1	R	Select boot mode with BOOT PIN
13	NRST_MODE	R	0: Reset input only 1: GPIO function
12	WWDG_SW	R	0： Hardware watchdog 1： Software watchdog
11	IWDG_SW	R	0: Hardware watchdog 1: Software watchdog
10：8	Resrved	-	

7：0	RDP	R	0xAA：level 0, read protection inactive Non 0xAA：level 1, read protection active

2) **Option byte for flash SDK area address**

Flash memory address: 0x1FFF 3108

Production value: 0xFFE0 001F

After the power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option bytes 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
~BOR_LEV[2:0]			~SDK_END[4:0]					Res	Res	~BOR_EN	~SDK_STRT[4:0]				
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**
BOR_LEV[2:0]			SDK_END[4:0]					Res	Res	BOR_EN	SDK_STRT[4:0]				
R	R	R	R	R	R	R	R			R	R	R	R	R	R

Bit	Name	R/W	Function
31：29	~BOR_LEV[2:0]	R	One's complement of BOR_LEV[2:0]
28：24	~SDK_END[4:0]	R	One's complement of SDK_END
23：22	Reserved	-	-
21	~BOR_EN	R	One's complement of BOR_EN
20：16	~SDK_STRT[4:0]	R	One's complement of SDK_STRT
15：13	BOR_LEV[2:0]	R	000: BOR rising threshold is 1.8 V, falling threshold is 1.7 V 001: BOR rising threshold is 2.0 V, falling threshold is 1.9 V 010: BOR rising threshold is 2.2 V, falling threshold is 2.1 V 011: BOR rising threshold is 2.4 V, falling threshold is 2.3 V 100: BOR rising threshold is 2.6 V, falling threshold is 2.5 V 101: BOR rising threshold is 2.8 V, falling threshold is 2.7 V 110: BOR rising threshold is 3.0 V, falling threshold is 2.9 V 111: BOR rising threshold is 3.2 V, falling threshold is 3.1 V
12：8	SDK_END[4:0]	R	SDK area end address，each corresponding STEP is 4 KBytes
7：6	Reserved	-	-
5	BOR_EN	R	BOR enable 0: BOR is disabled 1: BOR is enabled, BOR_LEV works
4：0	SDK_STRT[4:0]	R	SDK area start address，each corresponding STEP is 4 KBytes

3) **Option byte for Flash WRP address**

Flash memory address: 0x1FFF 3118

Production value: 0x0000 FFFF

After the power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option bytes 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	One’s complement of WRP
15：0	WRP	R	0: sector [y] is protected 1: sector [y] unprotected y = 0 to 15

4.3.2. Flash option byte write

After reset, the bits in the FLASH_CR register associated with the option byte are write-protected. The OPTLOCK bit in the FLASH_CR register must be cleared before the option byte can be manipulated. The following steps are used to unlock this register:

- Unlock sequence to unlock write protection of FLASH_CR register.
- Write OPTKEY1 = 0x0819 2A3B to the FLASH_OPTKEYR register.
- Write OPTKEY2 = 0x4C5D 6E7F to the FLASH_OPTKEYR 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 Hard Fault interrupt is generated.

User option (option bytes in information Flash memory) can be protected by software by writing the OPTLOCK bit of the FLASH_CR register to prevent unwanted erase/program operations.

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

Modifying user option bytes

Programming operation of the option byte is different from the operation to the main Flash memory.

To modify the option bytes, the following steps are required:

- Using the steps described previously to clear the OPTLOCK bit.
- Check that no Flash memory operation is ongoing by checking the BSY
- Write the desired value (1~3 words) to the option bytes register FLASH_OPTR/ FLASH_SDKAR/ FLASH_WRPR.
- Set OPTSTRT bit.
- Write any 32 bits data to the main Flash memory address 0x4002 2080 (trigger a formal program operation).
- Wait for the BSY bit to be cleared.
- Wait for EOP to be pulled high, software to be cleared.

Any change to the option bytes, the hardware will first erase the entire page to the option byte, and then program the value of the FLASH_OPTR, FLASH_SDKAR or FLASH_WRPR register to the option bytes. And, the hardware automatically calculates the corresponding complement, and programs the calculated value to the corresponding area of the option bytes.

Option byte loading

After the BSY bit is cleared, all new option bytes are written into the Flash information memory, but they are not applied to the system. The read operation of the option bytes register still returns the value in the last loaded option bytes. Once they are loaded with new values, it will work on the system.

The loading of option bytes is performed in the following two cases:

a) OBL_LAUNCH bit in the FLASH_CR register is set.

b) After power-on reset (POR, BOR)

Loading option bytes is: read the option bytes in the information memory area, and then store the read data in the internal option registers (FLASH_OPTR, FLASH_SDKAR and FLASH_WRPR). These internal registers configure the system and can be read by software. The OBL_LAUNCH bit is set to generate a reset, so that the loading of option bytes can be carried out under the reset of the system.

Each option bit has a corresponding complement at its same doubleword address (next half word). During the loading of the option bytes, the validation of the option bit and its complement ensures that the loading was performed correctly.

If the one's complement matches, the option bytes are copied into the option register.

If the one's complement does not match, the OPTVERR status bit in the FLASH_SR register is set. Unmatched values are written to the option register:

- For user option
  - BOR_LEV is written as 000 (the lowest threshold)
  - The BOR_EN bit is written as 0 (BOR is not enabled)
  - NRST_MODE bit written to 0 (reset input only)
  - RDP bit is written as 0xff (which is level 1)
  - The rest of the mismatched values are written as 1
- For SDK area option, SDKR_STRT [4:0] = 0x00, SDKR_END [4:0] = 0x1F, all Flash memory is set as SDK
- For the WRP option, the unmatched value is the default "no protection"

After system reset, the contents of option bytes are copied to the following option registers (readable and writable by software):

- 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 reflect the state of the system option.

4.4. Flash configuration bytes

Part of the interval (one page in total) of the information area of the Flash memory is used as factory config. byte.

Page 0 is stored for software to read information (only code, no one’s complement code is stored):

- HSI frequency selection control value, and corresponding trimming value.
- Erase and program time configuration parameter values corresponding to different frequencies of HSI.

Table 4-4 Factory config. byte organization

Page	Word	Address	Contents
0	0	0x1FFF 3200	Store HSI 4 MHz frequency selection control and corresponding trimming value
	1	0x1FFF 3208	Store HSI 8 MHz frequency selection control and corresponding trimming value
	2	0x1FFF 3210	Store HSI 16 MHz frequency selection control and corresponding trimming value
	3	0x1FFF 3218	Store HSI 22.12 MHz frequency selection control and corresponding trimming value
	4	0x1FFF 3220	Store HSI 24 MHz frequency selection control and corresponding trimming value
	5	0x1FFF 3228	TS_CAL1, 30℃ temperature sensor calibration value
	6	0x1FFF 3230	TS_CAL2, 85℃ temperature sensor calibration value
	7	0x1FFF 3238	Store the configuration values of the corresponding FLASH_TS0， FLASH_TS1 and FLASH_TS3 registers at the HSI 4 MHz frequency
	8	0x1FFF 3240	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at the HSI 4 MHz frequency
	9	0x1FFF 3248	Store the configuration value of the corresponding FLASH_PERTPE register at the HSI 4 MHz frequency
	10	0x1FFF 3250	Store the configuration value of the corresponding FLASH_ SMERTPE register at the HSI 4MHz frequency
	11	0x1FFF 3258	Store the configuration values of the corresponding FLASH_P RGTPE and FLASH_PRETPE registers at the HSI 4 MHz frequency
	12	0x1FFF 3260	Store the configuration values of the corresponding FLASH_TS0， FLASH_TS1 and FLASH_TS3 registers at the HSI 8 MHz frequency
	13	0x1FFF 3268	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at the HSI 8 MHz frequency
	14	0x1FFF 3270	Store the configuration value of the corresponding FLASH_PERTPE register at the HSI 8 MHz frequency
	15	0x1FFF 3278	Store the configuration value of the corresponding FLASH_ SMERTPE register at the HSI 8 MHz frequency
	16	0x1FFF 3280	Store the configuration values of the corresponding FLASH_PRGTPE and FLASH_PRETPE registers at the HSI 8 MHz frequency
	17	0x1FFF 3288	Store the configuration values of the corresponding FLASH_TS0， FLASH_TS1 and FLASH_TS3 registers at the HSI 16 MHz frequency
	18	0x1FFF 3290	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at the HSI 16 MHz frequency

	19	0x1FFF 3298	Store the configuration value of the corresponding FLASH_PERTPE register at the HSI 16 MHz frequency
	20	0x1FFF 32A0	Store the configuration value of the corresponding FLASH_ SMERTPE register at the HSI 16 MHz frequency
	21	0x1FFF 32A8	Store the configuration values of the corresponding FLASH_P RGTPE and FLASH_PRETPE registers at the HSI 16 MHz frequency
	22	0x1FFF 32B0	Store the configuration values of the corresponding FLASH_TS0，FLASH_TS1 and FLASH_TS3 registers at the HSI 22.12 MHz frequency
	23	0x1FFF 32B8	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at the HSI 22.12 MHz frequency
	24	0x1FFF 32C0	Store the configuration value of the corresponding FLASH_PERTPE register at the HSI 22.12 MHz frequency
	25	0x1FFF 32C8	Store the configuration value of the corresponding FLASH_ SMERTPE register at the HSI 22.12 MHz frequency
	26	0x1FFF 32D0	Store the configuration values of the corresponding FLASH_P RGTPE and FLASH_PRETPE registers at the HSI 22.12 MHz frequency
	27	0x1FFF 32D8	Store the configuration values of the corresponding FLASH_TS0，FLASH_TS1 and FLASH_TS3 registers at the HSI 24 MHz frequency
	28	0x1FFF 32E0	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at the HSI 2 4MHz frequency
	29	0x1FFF 32E8	Store the configuration value of the corresponding FLASH_PERTPE register at the HSI 24 MHz frequency
	30	0x1FFF 32F0	Store the configuration value of the corresponding FLASH_ SMERTPE register at the frequency of HSI 24 MHz frequency
	31	0x1FFF 32F8	Store the configuration values of the corresponding FLASH_PRGTPE and FLASH_PRETPE registers at the HSI 24 MHz frequency

4.4.1. HSI_TRIMMING_FOR_USER

Address: 0x1FFF 3200~0x1FFF 3220

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 data from this address, and then write to HSI_FS[2:0] and HSI_TRIM[12:0] corresponding to the RCC_ICSCR register to change the HSI frequency.

4.4.2. Calibration value of temperature sensor

Address: 0x1FFF 3228(30℃)、0x1FFF 3230(85℃)

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											
				TS_{CAL}[11:0]											

Software needs to read data from this address.

4.4.3. HSI_4 M/8 M/16 M/22.12 M/24 M_EPPARA0

Address: 0x1FFF 3238(4 MHz)、0x1FFF 3260(8 MHz)、0x1FFF 3288(16 MHz)、0x1FFF 32B0(22.12 MHz)、0x1FFF 32D8(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 set the HSI clock frequency according to the need, choose to read the data from the corresponding address, and then write the FLASH_TS0, FLASH_TS1, FLASH_TS3 registers to realize the configuration of the erasing and programming time required by the corresponding HSI frequency.

4.4.4. HSI_4 M/8 M/16 M/22.12 M/24 M_EPPARA1

Address: 0x1FFF 3240(4 MHz)、0x1FFF 3268(8 MHz)、0x1FFF 3290(16 MHz)、0x1FFF 32B8(22.12 MHz)、0x1FFF 32E0(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 set the HSI clock frequency according to the need, choose to read the data from the corresponding address, and then write the FLASH_TS2P and FLASH_TPS3 registers to realize the configuration of the erasing and programming time required for the corresponding HSI frequency.

4.4.5. HSI_4 M/8 M/16 M/22.12 M/24 M_EPPARA2

Address: 0x1FFF 3248(4 MHz)、0x1FFF 3270(8 MHz)、0x1FFF 3298(16 MHz)、0x1FFF 32C0(22.12 MHz)、0x1FFF 32E8(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 set the HSI clock frequency according to the need, choose to read the data from the corresponding address, and then write it into the FLASH_PERTPE register to realize the configuration of the erasing and programming time required for the corresponding HSI frequency.

4.4.6. HSI_4 M/8 M/16 M/22.12 M/24 M_EPPARA3

Address: 0x1FFF 3250(4 MHz)、0x1FFF 3278(8 MHz)、0x1FFF 32A0(16 MHz)、0x1FFF 32C8(22.12 MHz)、0x1FFF 32F0(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	SMER TPE[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 set the HSI clock frequency according to the need, choose to read the data from the corresponding address, and then write it into the FLASH_SMERTPE register to realize the configuration of the erasing and programming time required for the corresponding HSI frequency.

4.4.7. HSI_4 M/8 M/16 M/22.12 M/24 M_EPPARA4

Address: 0x1FFF 3258(4 MHz)、0x1FFF 3280(8 MHz)、0x1FFF 32A8(16 MHz)、0x1FFF 32D0(22.12 MHz)、0x1FFF 32F8(24 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	PRETPE[11:0]										
					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 set the HSI clock frequency according to the need, choose to read the data from the corresponding address, and then write it into the FLASH_PRGTPE and FLASH_PRETPE registers to realize the configuration of the erasing and programming time required for the corresponding HSI frequency.

4.5. Flash protection

The protection of Flash main memory includes the following mechanisms:

- Software design kit (SDK) is used to protect access to specific program areas, and the granularity is 4 KBytes.
- Read protection (RDP) is used to prevent access from outside.
- Wrtie protection (WRP) control is used to prevent unwanted writes (due to confusion of the program memory pointer PC). The granularity of write protection is designed to be 8 KBytes.
- Option byte write protection, special unlocking design.

4.5.1. Flash software development kit (SDK) area protection

The protection area is defined by SDKR _STRT[4:0], SDKR_END[4:0] of the FLASH_SDKR register, and each bit corresponds to 4 KBytes.

Start address

FLASH memory base address + SDK_STRT[4:0] x 0x1000(included)

End address

FLASH memory base address + (SDK_END[4:0]+1) x 0x1000(excluded)

When SDK _STRT[4:0] is greater than SDK_END[4:0], SDK protection is invalid. When SDK _STRT[4:0] is less than or equal to SDK _END[4:0], SDK protection is effective.

When the protection is in effect, when the FLASH_SDKR register is unprotected (writing SDK _STRT[4:0] is 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 the mass erase is used to protect the program in the SDK area), and then the value of the SDK option in the Flash option byte is updated (the updated value at this time is that the SDK protection is invalid).

At this time, the content of the FLASH_SDKR register will not be updated, until the power-on reset (POR/BOR/PDR) or OBL reset, the register content will be loaded from the SDK option in the Flash option byte into the register.

4.5.2. Flash read protection

By setting RDP option byte, and perform system reset (POR/BOR or OPL reset) to load a new RDP option byte to activate the read protection function. RDP protects main Flah memory, option byte, and SRAM.

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

When the RDP option byte and the two's complement code exist in the option byte, the Flash memory will be protected.

Table 4-5 Flash read protection status

RDP byte value	RDP complemented byte value	Read protection level
0xAA	0x55	Level 0
Any value except the combination of (0xAA and 0x55)		Level 1

Regardless of any protection level, system memory is access only read and program and erase operations cannot be performed.

Level 0: no protection

To read, program and erase the main Flash memory, as well as any operation to the option byte.

Level 1: Read protection

When the RDP and its two's complement in the option byte contain any combination rather than 0xAA, 0x55, the level 1 read protection takes effect, and the level 1 is the default protection level.

- User mode: The program executed in user mode (boot from main Flash memory) can perform all operations on main Flash and option byte.
- Debug, boot from SRAM, and boot from system memory mode (Boot loader): In debug mode, or when booting from SRAM or system boot from SRAM or system memory (Boot loader), main flash

is not accessible. In these modes, read or write access to main flash generates a bus error, and a hard fault interrupt.

When it is already at Level 1 (any number rather than 0xAA), changing to Level 0 by programming 0xAA, the hardware will perform a mass erase operation on the main Flash memory.

Table 4-6 The relationship between access status and 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	N/A	N/A	N/A	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
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	N/A	N/A	N/A	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. Mass erase command issued from any area will erase the SDK area.
2. Any modification of level 1 to level 0 will trigger the hardware mass erase of the main Flash memory.

3. The meaning of N/A is that when the SDK Area is disabled, since there is no SDK Area, no situation in which programs can be read out from the SDK Area in the above table, and no situation in which the programs read out from other areas can access the SDK Area.
4. There are two cases for executing programs from SRAM or system memory: one is Boot from, the other is boot from other memory, and the program jumps to SRAM or system memory.

4.5.3. Flash write protection

Flash can be set to be write-protected against unwanted writes. Define the control granularity of each bit of the WRP register as a write protection (WRP) area of 8 KBytes, that is, the size of 1 sector. See the description of the WRP register for details.

When the WRP area is activated, erase or program operations are not allowed. Accordingly, the mass erase function does not work even if only one area is set as write-protected.

In addition, if an attempt is made to erase or program a write-protected area, the write -protection error flag (WRPERR) of the FLASH_SR register will be set.

Note: Write protection only works on main Flash, and read doesn't work on system memory.

4.5.4. Option byte write protection

By default, Option bytes are readable and write-protected. To gain erase or program access to option bytes, the correct sequence needs to be written to the OPTKEYR register.

4.6. Flash interrupt

Table 4-7 Flash interrupt request

Interrupt event	Event flag	Time stamp/interrupt clear method	Control bit enable
End of operation	EOP	Write EOP = 1	EOPIE
Write protection	WRPERR	Write WRPERR = 1	ERRIE

The following events do not have a separate interrupt flag, but will generate a Hard fault:

- Sequence error of FLASH_CR register of unlock Flash memory.
- Unlock Flash option bytes write sequence error.
- Flash program operation is not aligned with 32-bit data.
- Flash erase (including page erase, sector erase and mass erase) operations do not perform 32-bit data alignment.

- To the option byte register is not aligned with 32-bit data.

4.7. Flash register description

4.7.1. 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	LATENCY	
														RW	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	-
1:0	LATENCY[1:0]	RW	0	The wait state corresponding to the read operation: 00:There is no wait state for Flash read operation（SYS-CLK<=24 MHz） 01:The Flash read operation has one wait state, which is two system clock cycles are required for each Flash read（24 MHz <SYSCLK<=48 MHz） 10:The Flash read operation has two wait state, which is four system clock cycles are required for each Flash read（48 MHz <SYSCLK<=72 MHz） 11:Reserved

4.7.2. FLASH key register (FLASH_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

All register bits are write-only and read as 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 0000	The following values must be written consecutively to unlock the FLASH_CR register and enable the program/erase operation of the Flash KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.7.3. FLASH option key register (FLASH_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

All register bits are write-only and read as 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 option register of the Flash and enable the program/erase operation of the option byte KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F

4.7.4. 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**
OPTVERR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	WRPERR	Res	Res	Res	EOP
RC_W1											RC_W1				RC_W1

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	BSY	R	0	Busy bit

Bit	Name	R/W	Reset Value	Function
				This bit indicates that the operation of the Flash is in progress. This bit is set by hardware at the beginning of a Flash operation, and is cleared by hardware when the operation is completed or an error occurs.
15	OPTVERR	RC_W1	0	Option and trimming bits loading validity error when the option and trimming bits and their one's complements do not match. Load unmatched option bytes, coerced to safe values. Software writes 1 to clear.
14:5	reserved	-	-	-
4	WRPERR	RC_W1	0	Write protection error This bit is set by hardware when the address to be programmed/erased is in a write-protected Flash region (WRP). Software write 1 to clear this bit.
3:1	Reserved	-	-	-
0	EOP	RC_W1	0	When the program/erase operation of the Flash completes successfully. This bit is only set if the EOPIE bit in the FLASH_CR register is enabled. Software write 1 to clear this bit.

4.7.5. 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	OPTLOCK	Res	Res	OBL_LAUNCH	Res	ERRIE	EOPIE	Res	Res	Res	Res	PGSTRT	Res	OPTSTRT	Res
RS	RS			RC_W1		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	Res	Res	MER	PER	PG
				RW									RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	FLASH_CR Lock bit. Software can only set this bit. When set, the FLASH_CR register is locked. When the unlock timing is successfully given, this bit is cleared by hardware, and the FLASH_CR register is unlocked. The software should set this bit after the program/erase operation is completed.

Bit	Name	R/W	Reset Value	Function
				When an unsuccessful unlock sequence is given, this bit remains set until the next system reset.
30	OPTLOCK	RS	1	Option bytes Lock bit. Software can only set this bit. When set, the bits related to option bytes in the FLASH_CR register are locked. When the unlock timing is successfully given, this bit is cleared by hardware, and the FLASH_CR register is unlocked. The software should set this bit after the program/erase operation is completed. When an unsuccessful unlock sequence is given, this bit remains set until the next system reset.
29:28	Reserved	-	-	-
27	OBL_LAUNCH	RC_W1	0	Force the option bytes loading. When set, this bit forces the system to perform a reload of option bytes. This bit is only cleared by hardware when the option byte load has been completed. This bit cannot be written if the OPTLOCK bit is set. 0: Option byte loading completed 1: Option byte loading request is generated, the system resets, and the option byte is reloaded.
25	ERRIE	RW	0	Error interrupt enable bit, when the WRPERR bit in the FLASH_SR register is set, if this bit is enabled, an interrupt request is generated. 0: No interrupt is generated 1: An interrupt is generated
24	EOPIE	RW	0	End of operation interrupt enable This bit enables interrupt generation when the EOP bit in the FLASH_SR register is set. 0: EOP interrupt disabled 1: EOP interrupt enable
23:18	Reserved	RW	-	-
19	PGSTRT	RW	0	The start bit of the program operation of the Flash main memory. Program operation of the main Flash memory, and is set by software. After the BSY bit of the FLASH_SR register is cleared, the hardware clears this bit.
18	Reserved	-	-	-
17	OPTSTRT	RW	0	Flash option bytes modified start bit This bit initiates modification of option bytes. Set by software and cleared by hardware after the BSY bit in the FLASH_SR register is cleared. Note: When modifying the Flash option bytes, the hardware will automatically perform the erase operation on the entire page of 128 bytes, and then perform the program operation,

Bit	Name	R/W	Reset Value	Function
				which also includes the automatic writing of the two's complement code.
16:12	Reserved	-	-	-
11	SER	RW	0	4 kbyte Sector erase operation 0: Sector erase operation of Flash is not selected 1: Select the sector erases operation of Flash Note: - Sector erase will not work on Flash information memory - Sector erase has no effect on areas set to WRP.
10:3	Reserved	-	-	-
2	MER	RW	0	Mass erase operation 0: Mass erase operation of Flash is not selected 1: Select the mass erases operation of Flash Note: Mass erase will not work on Flash information memory. Mass erase does not work when WRP is set
1	PER	RW	0	Page erase operation 0: Page erase operation of the Flash is not selected 1: Select the page erase operation of Flash
0	PG	RW	0	Program operation 0: Program operation of Flash is not selected 1: Select the program operation of Flash

4.7.6. 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 bytes area of the Flash in formation 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
IWDG_STOP	nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	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
31:16	Reserved	-	-	-
15	IWDG_STOP	RW	1	Set iwdg timer running state in stop mode 0:freze timer 1: normal operation
14	nBOOT1	RW	1	Select the boot mode with the BOOT PIN
13	NRST_MODE	RW	0	0: Reset input only 1: GPIO: GPIO function
12	WWDG_SW	RW	1	0: Hardware watchdog 1: Software watchdog
11	IWDG_SW	RW	1	0: Hardware watchdog 1: Software watchdog
10:8	Reserved	-	-	-
7:0	RDP	RW	0xAA	0xAA: level 0, read protection inactive Non 0xAA: level 1, read protection active

4.7.7. FLASH SDK address register (FLASH_SDKR)

Address offset: 0x24

Reset value: 0xxxx xxxx

After the power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the option bytes 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**
BOR_LEV[2:0]			SA_END[4:0]					Res	Res	BOR_EN	SA_STRT[4:0]				
RW			RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:13	BOR_LEV[2:0]	RW		000: BOR rising threshold is 1.8 V, falling threshold is 1.7 V 001: BOR rising threshold is 2.0 V, falling threshold is 1.9 V 010: BOR rising threshold is 2.2 V, falling threshold is 2.1 V 011: BOR rising threshold is 2.4 V, falling threshold is 2.3 V 100: BOR rising threshold is 2.6 V, falling threshold is 2.5 V 101: BOR rising threshold is 2.8 V, falling threshold is 2.7 V 110: BOR rising threshold is 3.0 V, falling threshold is 2.9 V 111: BOR rising threshold is 3.2 V, falling threshold is 3.1 V

Bit	Name	R/W	Reset Value	Function
12:8	SDK_END[4:0]	RW		SDK area end addres, each corresponding STEP is 4 KBytes
7:6	Reserved	-	-	-
5	BOR_EN	RW		BOR enable 0: BOR is not enabled 1: BOR is enabled, BOR_LEV works
4:0	SDK_STRT[4:0]	RW		SDK area start address, each corresponding STEP is 4 KBytes

4.7.8. 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 bytes area of the Flash in formation 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	-	-	-
15	WRP	RW	1	0: Sector 15, with write protection, program and erase are not allowed 1: Sector 15, no write protection
14	WRP	RW	1	0: Sector 14, with write protection, program and erase are not allowed 1: Sector 14, no write protection
13	WRP	RW	1	0: Sector 13, with write protection, program and erase are not allowed 1: Sector 13, no write protection
12	WRP	RW	1	0: Sector 12, with write protection, program and erase are not allowed 1: Sector 12, no write protection
11	WRP	RW	1	0: Sector 11, with write protection, program and erase are not allowed 1: Sector 11, no write protection

Bit	Name	R/W	Reset Value	Function
10	WRP	RW	1	0: Sector 10, with write protection, program and erase are not allowed 1: Sector 10, no write protection
9	WRP	RW	1	0: Sector 9, with write protection, program and erase are not allowed 1: Sector 9, no write protection
8	WRP	RW	1	0: Sector 8, with write protection, program and erase are not allowed 1: Sector 8, no write protection
7	WRP	RW	1	0: Sector 7, with write protection, program and erase are not allowed 1: Sector 7, no write protection
6	WRP	RW	1	0: Sector 6, with write protection, program and erase are not allowed 1: Sector 6, no write protection
5	WRP	RW	1	0: Sector 5, with write protection, program and erase are not allowed 1: Sector 5, no write protection
4	WRP	RW	1	0: Sector 4, with write protection, program and erase are not allowed 1: Sector 4, no write protection
3	WRP	RW	1	0: Sector 3, with write protection, program and erase are not allowed 1: Sector 3, no write protection
2	WRP	RW	1	0: Sector 2, with write protection, program and erase are not allowed 1: Sector 2, no write protection
1	WRP	RW	1	0: Sector 1, with write protection, program and erase are not allowed 1: Sector 1, no write protection
0	WRP	RW	1	0: Sector 0, with write protection, program and erase are not allowed 1: Sector 0, no write protection

4.7.9. FLASH sleep time configuration register (FLASH_STCR)

Address offset: 0x90

Reset value: 0x0000 6400

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	-	-	-
15:8	SLEEP_TIME	RW	0x64	FLASH sleep time count (counter based on HSI_10M clock) When the system clock selects LSI or LSE, in order to obtain more optimized power consumption in Run mode, which can use the function of this register (it is only recommended to use this function when LSI or LSE is the system clock). When this function is enabled, the time width of the Flash in the Sleep state in each half system clock low period is: t_{HSI_10M} *SLEEP_TIME Note : t_{HSI_10M} is the period of HSI_10M.
7:1	Reserved	-	-	-
0	SLEEP_EN	RW	0	FLASH Sleep enable 1:enable flash sleep 0:disable flash sleep

4.7.10. FLASH TS0 register (FLASH_TS0)

Address offset: 0x100

Reset value: 0x0000 00B4

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	-	-	-
7:0	TS0	RW	0xB4	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3240 8MHz : 0x1FFF 3268 16MHz : 0x1FFF 3290 22.12MHz : 0x1FFF 32B8 24MHz : 0x1FFF 32E0

4.7.11. FLASH TS1 register (FLASH_TS1)

Address offset: 0x104

Reset value: 0x0000 01B0

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	-	-	-
8:0	TS1	RW	0x1B0	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3240 8MHz : 0x1FFF 3268 16MHz : 0x1FFF 3290 22.12MHz : 0x1FFF 32B8 24MHz : 0x1FFF 32E0

4.7.12. FLASH TS2P register (FLASH_TS2P)

Address offset: 0x108

Reset value: 0x0000 00B4

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	-	-	-
7:0	TS2P	RW	0xB4	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency.

Bit	Name	R/W	Reset Value	Function
				The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3240 8MHz : 0x1FFF 3268 16MHz : 0x1FFF 3290 22.12MHz : 0x1FFF 32B8 24MHz : 0x1FFF 32E0

4.7.13. FLASH TPS3 register (FLASH_TPS3)

Address offset: 0x10C

Reset value: 0x0000 06C0

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	-	-	-
10:0	TPS3	RW	0x6C0	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3240 8MHz : 0x1FFF 3268 16MHz : 0x1FFF 3290 22.12MHz : 0x1FFF 32B8 24MHz : 0x1FFF 32E0

4.7.14. FLASH TS3 register (FLASH_TS3)

Address offset: 0x110

Reset value: 0x0000 00B4

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	-	-	-
7:0	TS3	RW	0xB4	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3238 8MHz : 0x1FFF 3260 16MHz : 0x1FFF 3288 22.12MHz : 0x1FFF 32B0 24MHz : 0x1FFF 32D8

4.7.15. FLASH page erase TPE register (FLASH_PERTPE)

Address offset: 0x114

Reset value: 0x0001 4820

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	-	-	-
16:0	PERTPE	RW	0x14820	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3248 8MHz : 0x1FFF 3270 16MHz : 0x1FFF 3298 22.12MHz : 0x1FFF 32C0 24MHz : 0x1FFF 32E8

4.7.16. FLASH sector/mass erase TPE register (FLASH_SMERTPE)

Address offset: 0x118

Reset value: 0x0001 4820

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	-	-	-
16:0	SMERTPE	RW	0x14820	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3250 8MHz : 0x1FFF 3278 16MHz : 0x1FFF 32A0 22.12MHz : 0x1FFF 32C8 24MHz : 0x1FFF 32F0

4.7.17. FLASH PROGRAM TPE register (FLASH_PRGTPE)

Address offset: 0x11C

Reset value: 0x0000 5DC0

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	-	-	-
15:0	PRGTPE	RW	0x5DC0	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3258 8MHz : 0x1FFF 3280 16MHz : 0x1FFF 32A8 22.12MHz : 0x1FFF 32D0 24MHz : 0x1FFF 32F8

4.7.18. FLASH pre-program TPE register (FLASH_PRETPE)

Address offset: 0x120

Reset value: 0x0000 12C0

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	0x12C0	The software reads out the data stored at the corresponding address in the information area and writes it to the corresponding register to configure the erase/write time required for the corresponding HSI frequency. The data is stored in the following address in the Flash: 4MHz : 0x1FFF 3258 8MHz : 0x1FFF 3280 16MHz : 0x1FFF 32A8 22.12MHz : 0x1FFF 32D0 24MHz : 0x1FFF 32F8

4.7.19. FLASH register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	FLASH_ACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LATENCY
	Reset value																																0
0x008	FLASH_KEYR	KEY[31:16]																KEY[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0C	FLASH_OPTKEYR	OPTKEY[31:16]																OPTKEY[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	FLASH_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY	OPTVERR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPERR	Res.	Res.	Res.	EOP
	Reset value																0	0											0				0
0x14	FLASH_CR	LOCK	OPTLOCK	Res.	Res.	OBL_LAUNCH	Res.	ERRIE	EOPIE	Res.	Res.	Res.	Res.	PGSTRT	Res.	OPTSTRT	Res.	Res.	Res.	Res.	Res.	SER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MER	PER	PG
	Reset value	0	0			0								0		0						0									0	0	0
0x20	FLASH_OPTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	BOR_LEV [2:0]			BOR_EN	RDP[7:0]							
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x24	FLASH_SDKR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SA_END[4:0]					Res.	Res.	Res.	SA_STRT[4:0]				
	Reset value																				X	X	X	X	X				X	X	X	X	X

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	FLASH_WRPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP[15:0]															
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x90	FLASH_STCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_TIME[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_EN
	Reset value																	0	1	1	0	0	1	0	0								0
0x100	FLASH_TS0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS0[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x104	FLASH_TS1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1[8:0]								
	Reset value																								1	1	0	1	1	0	0	0	0
0x108	FLASH_TS2P	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x1	FLASH_TPS3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLASH_TPS3[10:0]										

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0C	Reset value																						1	1	0	1	1	0	0	0	0	0	0
0x110	FLASH_TS3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS3[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x114	FLASH_PERTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERTPE	WRP[15:0]															
	Reset value																0	1	1	1	0	1	0	1	0	0	1	1	0	0	0	0	0
0x118	FLASH_SMERTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMERTPE[16:0]																
	Reset value																0	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	0
0x11C	FLASH_PRGTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRGTPE[15:0]															
	Reset value																	1	0	0	0	1	1	0	0	1	0	1	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x120	FLASH_PRETPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRETPE[13:0]													
	Reset value																			0	1	0	0	1	0	1	1	0	0	0	0	0	0

5. Power control

5.1. Power supply

5.1.1. Power block diagram

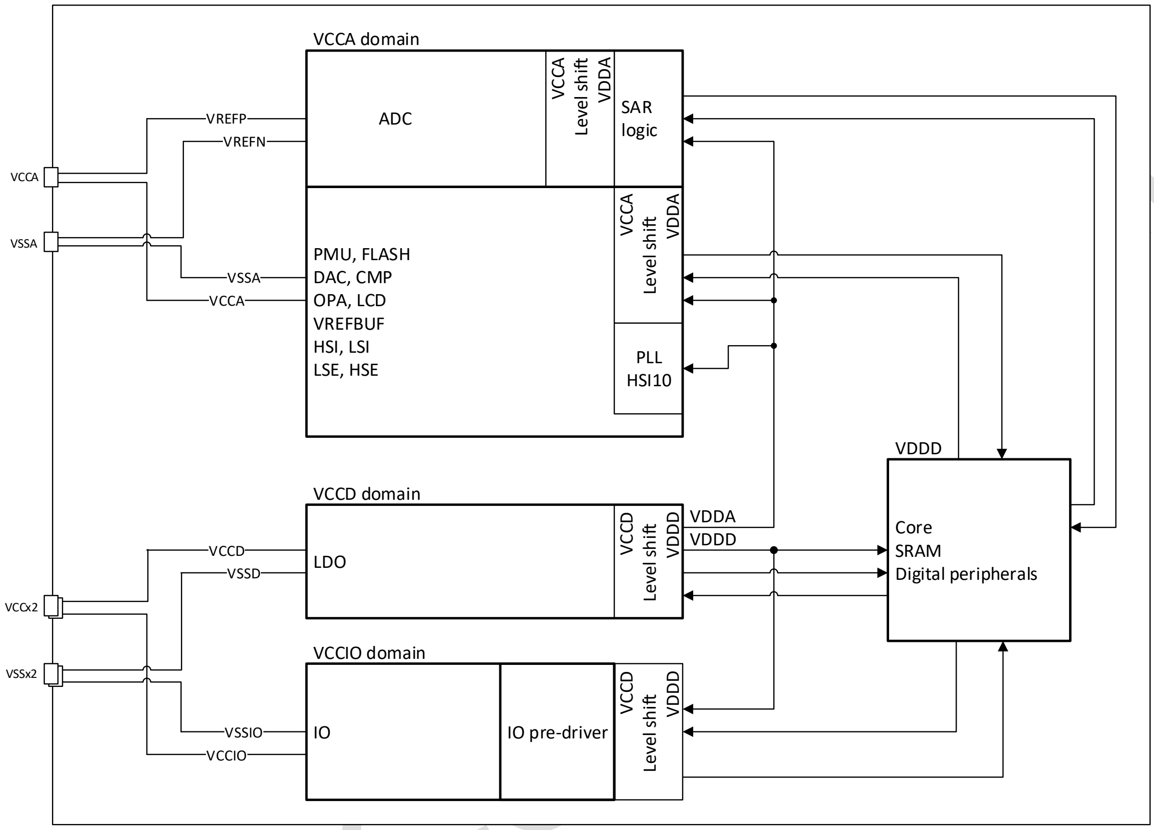


Figure 5-1 Power block diagram

Table 5-1 Power block

Numbering	Power supply	Power value	Describe
1	VCC	1.7 V~5.5 V	Power is provided to the chip through the power supply pins. The power supply modules are: part of the analog IP and IO circuits.
2	VCCA	1.7 V~5.5 V	Powering most of the analog modules comes from the VCCA PAD.
3	VCCIO	1.7 V~5.5 V	Power supply to IO, from VCC PAD
4	VDDx (VDDD/VDDA)	1.2 V/1.0 V/0.9 V/0.8 V±10%	The output from the VR supplies power to the main logic circuitry and SRAM inside the chip. When MR is powered, the output is 1.2 V. When in stop mode, it can be powered by MR or LPR depending on the software configuration, and the LPR output is 1.2 V /1.0 V/ 0.9 V/ 0.8 V depending on the software configuration.

5.2. Voltage regulator

The microcontroller designs two voltage regulators:

- Main regulator (MR) keeps working when the chip is in normal operating state.
- Low power regulator (LPR) provides a lower power consumption option in stop mode.

VDDx comes from MR or LPR depending on the working mode.

In run mode, MR keeps working, outputs is 1.2 V, and LPR is turned off.

In stop mode, it can be decided by software whether to supply from MR or LPR. Similarly, it is up to the software to decide whether VDDx is 1.2V or 1.0 V or 0.9 V or 0.8 V for the LPR supply case after entering stop.

5.3. Dynamic voltage value management

Dynamic voltage value management refers to adjusting the output VDDx voltage of VR, to obtain corresponding performance and power consumption with different voltages according to application requirements.

- **Range 1: High performance range**

The typically output of MR is 1.2 V (VDDx), and the system clock frequency can run as fast as 48 MHz.

- **Range 2: Low power range**

Only in stop mode, it is allowed to enter the low power range, and the range only works for LPR.

By default, the output of LPR is 1.2 V (VDDx) typical. When the VOS bit of the reset register is set, MR switches to LPR power when the chip enters stop mode (if the software selects stop mode to be powered by LPR) and LPR is switched to 1.0 V (VDDx) typical. At this time, some of the logic circuits in operation (LPTIMER) can run under LSI. In order to obtain lower power consumption in stop mode, LPR can be switched to a lower value of 0.9 V/0.8 V (VDDx). In this case, it is recommended to use IO for the wake-up method.

When the chip exits the stop mode, the chip resumes MR power supply and the VOS bit is cleared by hardware. The next time you enter stop mode, if you want to get lower power consumption, you still have to set the VOS bit by software so that the chip enters stop mode with LPR power supply of 1.0 V and lower 0.9 V/0.8 V.

5.4. Power monitoring

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

The POR/PDR module is designed in the chip and placed under the VDD power domain to provide power-on and power-off reset for the chip. 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 BOR is turned on, the BOR threshold can be selected by Option byte.

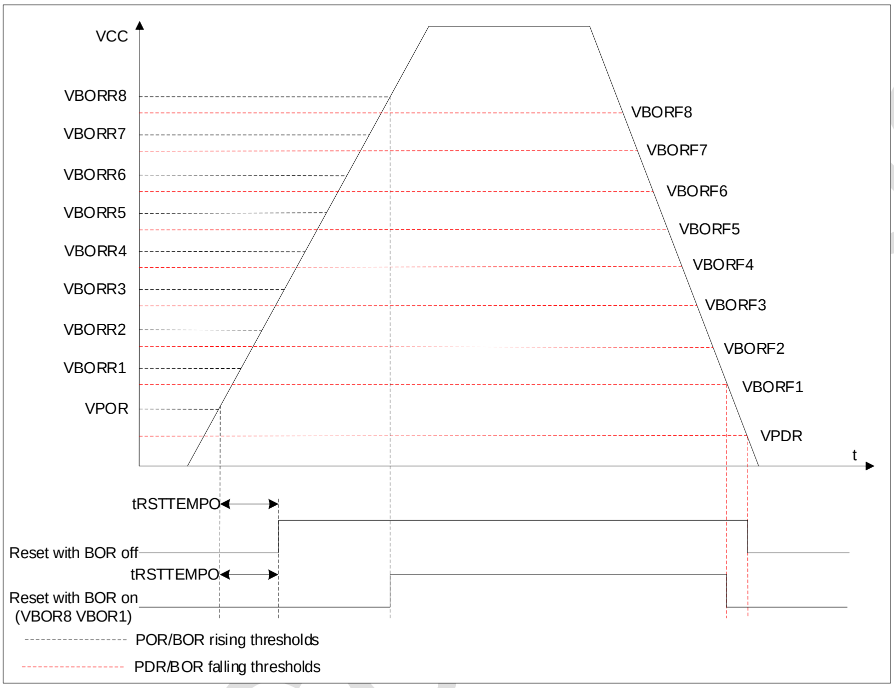


Figure 5-2 POR/PDR/BOR threshold

5.4.2. Programmable voltage detector (PVD)

This module can be used to detect the VCC power supply (also can detect the voltage of the PB7 pin), and the detection point can be configured through the register. When VCC is higher or lower than the detection point of PVD, a corresponding flag is generated.

This event is internally connected to line 16 of EXTI, depending on the rising/falling edge configuration of EXTI line 16, when VCC rises above the detection point of PVD, or VCC falls below the detection point of PVD, an interrupt is generated. In the service program, users can perform urgent shutdown tasks.

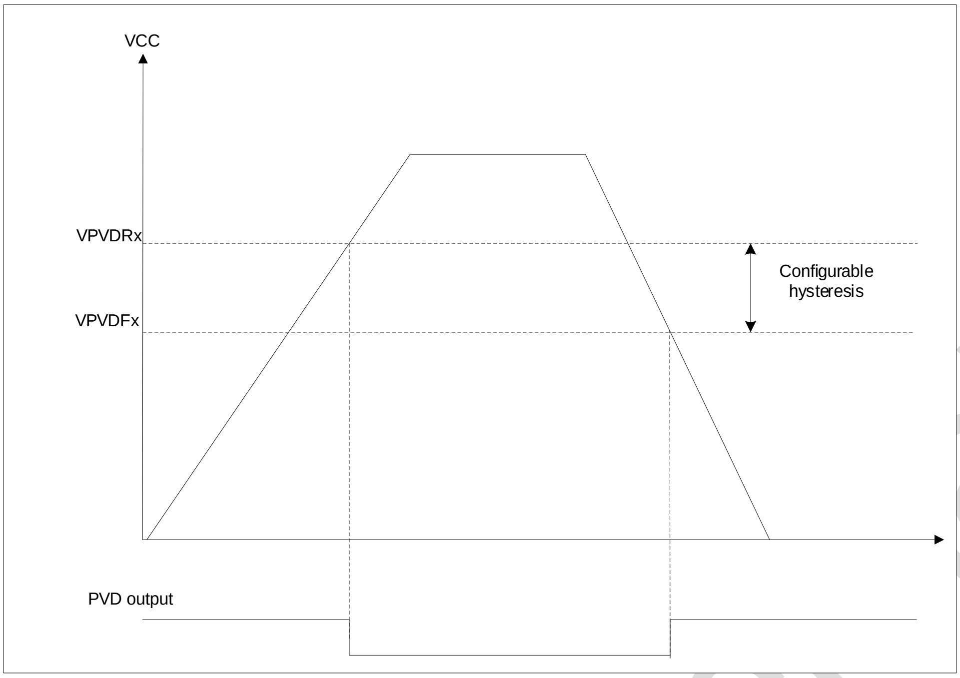


Figure 5-3 PVD threshold

6. Low-power control

By default, the microcontroller is in run mode after a system or a power reset. Several low-power modes are available to save power when the CPU does not need to be kept running, for example when waiting for an external event. Software can choose between power consumption, startup time, and wakeup sources.

6.1. Low-power mode

6.1.1. Introduction to low-power modes

There are two low-power modes:

- **Sleep mode:** CPU Core clock off (NVIC, SysTick, etc. work), peripherals can be configured to stay working. (It is recommended to enable only the modules that must work and turn off the module when it is finished working).
- **Stop mode:** In this mode, the contents of SRAM and registers are maintained, the high-speed clock PLL, HSI and HSE are turned off, and the clocks of most modules in the VDD domain are stopped.

In stop mode, LSI, LSE, RTC, LPTIMER, etc. can keep working. For details on the working conditions of each module in this mode, refer to Table 6-2.

In the stop mode, the corresponding VR state can be controlled by software and set to MR or LPR power supply. When the LPR power supply, the chip power consumption is greatly reduced, but the wake-up time is longer; when the MR power supply is maintained, the chip power consumption is higher, but it has a faster wake-up capability.

In addition, in run mode, the power consumption can be reduced by the following methods:

- Decrease system clock frequency
- For unused peripherals, turn off peripheral clocks (system clock and module clock)

In summary, the low-power mode transition diagram of this project is as follows.

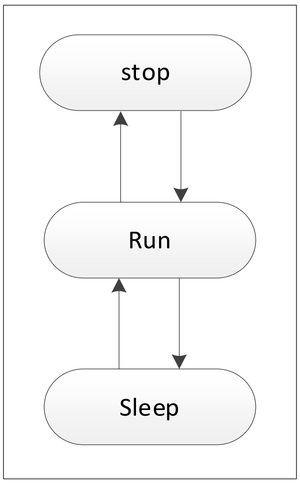


Figure 6-1 Low-power mode

6.1.2. Low-power mode switch

Table 6-1 Low power mode switch

Mode	Entry	Wakeup	Wake-up clock	Effects on the clock	Voltage regulator	
					MR	LPR
Sleep (sleep-now or sleep-on-exit)	WFI or Return from ISR	Any interruption	Same as before entering sleep mode	The CPU core clock is off and has no effect on other clocks and clock sources.	On(1)	Close
	WFE	Wakeup event				
Stop	SLEEPDEEP bit 1） WFI 2） Return from ISR 3） WFE Note: The system clock switches to HSI when entering stop	Any EXTI Line configured for wake-up (EXTI register configuration), IWDG reset, NRST	HSISYS (HSI maintains the frequency configuration before entering STOP, no crossover)	HSI, HSE, PLL off; LSI, LSE selectable on or off; LPTIMER, RTC, IWDG: configured by software to work or not; modules such as Low Power Wakeup and partial RCC remain operational; Clocking off for the remaining modules.	Software Configuration Switch	Software configuration switch, if on, output voltage 1.2 V/1.0 V/0.9 V/0.8 V configurable

1. The software must configure the VR state as MR mode to enter sleep mode.

6.1.3. Functions in each working mode

Table 6-2 Functions in each working mode[1]

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wakeup ability
CPU	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	-	-
PLL	O	O	-	-
LSI	O	O	O	-
LSE	O	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/USART4	O	O	-	-
I2C1/I2C2	O	O	-	-
SPI1/SPI2	O	O	-	-
ADC	O	O	-	-
COMP1/COMP2/COMP3	O	O	O	O
OPA1/OPA2/OPA3	O	O	-	-
Temperature sensor	O	O	-	-
DAC1/DAC2	O	O	-	-
USB	O	O	-	-
CAN	O	O	-	-
LCD	O	O	O	-
Timers(TIM1/TIM2/TIM3/TIM6/TIM7 /TIM14/TIM15/TIM16/TIM17)	O	O	-	-
LPTIM	O	O	O	O
IWDG	O	O	O	O
WWDG	O	O	-	-
SysTick timer	O	O	-	-
CRC	O	O	-	-
GPIOs	O	O	O	O

1. Y = Yes (enable), O = Optional (default disabled, can be enabled by software), - = Not available.

2. Flash is not powered off, but no clock is provided, and it enters the lowest power consumption state.

3. SRAM clock can be turned on or off.

4. The SRAM is not powered down, but no clock is provided and it enters the lowest power consumption state.

5. If LSE CSS is enabled before entering stop mode, the system will wake up and enter an NMI interrupt when there is a problem with LSE CSS.

6.2. Sleep mode

6.2.1. Entering sleep mode

The sleep mode is entered by executing the WFI (wait for interrupt) or WFE (wait for event) instructions. Two option 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 0, to enter sleep mode as soon as WFI or WFE instruction is excuted.
- Sleep-on-exit: If the SLEEPONEXIT bit is 1, to enter sleep mode as soon as it exits the low priority ISR.

  In the sleep mode, all IO pins keep the same state as in the run mode.

6.2.2. Exiting sleep mode

If the WFI instruction is used to enter sleep mode, any peripheral interrupt acknowledged by 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. The wakeup events can be generated in the following ways:

- Enable interrupts in the peripheral control register but not in the NVIC, and enabling the SEVON-PEND bit in the Cortex M0+. When the MCU resumes from WFE, the peripheral interrupt pending bit and the peripheral NVIC IRQ channel pending bit (in the NVIC interrupt clear pending register) must be cleared.
- Or configuring an external or internal EXTI line in event mode. When the CPU resumes from 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 offer the shortest wakeup time, and no time is wasted in interrupt entry and exit.

Table 6-3 Sleep-now

Sleep-now mode	Description
Mode entry	WFI or WFE while: - SLEEPDEEP = 0 and - SLEEPONEXIT = 0
Mode exit	Enter the sleep mode through WFI, the exit method is: interrupt. Enter the sleep mode through WFE, the exit method is: wakeup event.
Wakeup latency	None

Table 6-4 Sleep-on-exit

Sleep-on-exit	Description
Mode entry	WFI while: - SLEEPDEEP = 0 and - SLEEPONEXIT = 1
Mode exit	Interrupt
Wakeup latency	None

6.3. Stop mode

The stop mode is based on the Cortex-M0+ deep sleep mode combined with peripheral clock gating, and the VR can be configured as MR or LPR power supply. In stop mode, , HIS, HSE, PLL are turned off, SRAM and register contents are kept in a state, LSI, LSE, LPTIMER, RTC, IWDG can be configured by software whether to work, low-power wakeup and some RCC logic and so on, the clock inputs to the digital blocks of the remaining VCORE domains are turned off.

In the stop mode, all IO pins keep the same state as in the run mode.

6.3.1. Entering stop mode

To further reduce power consumption in stop mode, when PWR_CR.LPR = 1, VR can enter LPR to supply power.

If Flash memory programming is ongoing, the stop mode entry is delayed until the memory access is finished (the BSY bit of the FLASH_SR register is read by software to determine whether the current erase and program operations have been completed).

If an access to the APB domain is ongoing, the stop mode entry is delayed until the APB access is finished (controlled by software).

6.3.2. Exiting stop mode

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

In stop mode, if VR is in LPR state, there is an additional stabilization delay for wakup in stop mode.

In stop mode, if VR is in MR state, the current consumption will be large, but the wakeup time will be reduced.

Table 6-5 Stop mode

Stop mode	Description
Mode entry	WFI (wait for interrupt) or WFE (wait for event) while: - Configuration settings: ■ Configure the LPR bit of PWR_CR, to select VR to work under MR or LPR. ■ Configure the VOS bit of PWR_CR, to select LPR mode to provide 1.2 V, 1.0 V, 0.9 V, 0.8 V. ■ Configure the FLS_SLPTIME of PWR_CR, to set wake-up time of Flash. - Set the SLEEPDEEP bit of Cortex M0+ **Note:** To enter stop mode, all EXTI line pending bits (EXTI_PR register), all peripheral interrupt pending bits and RTC alarm flags must be reset. Otherwise , the stop mode entry proceduce is ignored and program execution continues. If the application needs to disable HSE before entering stop mode, the system clock source must be first switched to HSI and then clear the HSEON bit. To make the change of chip power consumption as balanced as possible, the software needs to follow the principle of gradual shutdown: gradually shut down the clock of each module, select HSI as the system clock, close PLL and HSE. To shorten the wakeup time, before entering the stop mode, the system clock should be configured to select the HSI high-frequency clock, and the HPRE of the RCC_CFGR register is set to 0, otherwise the hardware switching clock after wake-up will consume extra clocks.
Mode exit	If using WFI to enter stop mode: ■ Any EXTI Line configured in interrupt mode (the corresponding EXTI interrupt vector must be enabled in the NVIC). If using WFE to enter stop mode: ■ Any EXTI Line configured in event mode. ■ Interrupt pending bit when the CPU SEVONPEND bit is set.
Wakeup latency	LPR to MR wakeup time + HSI wakeup time + Flash wakeup time

6.4. Decreasing system clock frequency

In run mode, the frequency of the system clock (SYSCLK, HCLK, PCLK) can be reduced by frequency division through the prescaler register configuration. 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) for individual peripherals and memories can be stopped at any time to reduce power consumption.

To reduce the power consumption in sleep mode, peripheral clocks can be stopped before executing WFI or WFE instructions.

6.6. Power management register

The peripheral 's registers can be accessed through half-word or word.

6.6.1. Power control register 1 (PWR_CR1)

Address offset: 0x00

Reset value: 0x0000 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	HSION_CTRL	Res	Res	Res
												RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	LPR	FLS_SLP-TIME[1:0]		Res	VOS[1:0]		DBP	Res	Res	Res	Res	Res			
	RW	RW	RW		RW		RW								

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	HSION_CTRL	RW	0	HSI turns on time control when wakeup from stop mode. 0: After waiting for MR to stabilize, enable HIS. 1: Enable HSI when wakeup
18:15	Reserved			
14	LPR	RW	0	Low power regulator 0: Main regulator works in stop mode 1: Low power regulator works in stop mode
13:12	FLS_SLPTIME[1:0]	RW	2'b00	Wakeup sequence from stop mode, after the HSI is stable, 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, it means that the program is executed from SRAM instead of Flash after wakeup. And the program guarantees that Flash will not be accessed within 3 us after waking up the execution program.
11	Reserved	-	-	
10:9	VOS[1:0]	RW	0	Voltage scaling range selection

				00:After entering the stop mode, VCORE=1.2 V 01:After entering stop mode, VCORE=1.0 V 10: After entering stop mode, VCORE=0.9 V 11: After entering stop mode, VCORE=0.8 V
8	DBP	RW	0	RTC write protection disabled After reset, the RTC is write-protected to prevent accidental writes. To access the RTC this bit must be set to 1. 0: Disable access to RTC 1: RTC can be accessed
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 related to PVD function.

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	Res	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 filter time configuration The filter time is about 30.7 ms (1024 LSI or LSE clocks) 101: The filter time is about 3.8 ms (128 LSI or LSE clocks) 100: The filter time is about 1.92 ms (64 LSI or LSE clocks) 011: The filter time is about 480 us (16 LSI or LSE clocks) 010: The filter time is about 120 us (4 LSI or LSE clocks) 001: The filter time is about 60 us (2 LSI or LSE clocks) 000: The filter time is about 30 us (1 LSI or LSE clock)
8	FLTEN	RW	1	Digital filter function enable control 0: Disable 1: enable
7	Reserved	-	-	-
6:4	PVDT[2:0]	RW	000	Voltage rising edge detection threshold (falling edge detection threshold correspondingly reduced by 0.1 V) 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			
2	SRCSEL	RW	0	PVD detects power supply selection. 0:VCC 1:Detect PB7 pin If this bit is set to 1, the voltage on PB7 is internally compared to VREFINT (including rising and falling thresholds). In this case, the setting of the PVDT register is invalid.
1	Reserved	-	-	Reserved
0	PVDE	RW	0	Voltage detect enable bit 0: Voltage detection disable 1: Voltage detection enable If SYSCFG_CFG2.PVD_LOCK = 1, PVDE is write protected. Write protection is reset only after a system reset.

6.6.3. Power status register (PWR_SR)

Address offset: 0x14

Reset value: 0x0000 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	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 0: The detected VCC or PB7 exceeds the PVD selected compare threshold 1: Detected VCC or PB7 is below the PVD selected compare threshold
10:0	Reserved	-	-	Reserved

6.6.4. PWR register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	PWR_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSION_CTRL	Res.	Res.	Res.	Res.	LPR	FLS_SLPTIME[1:0]		Res	VOS[1:0]		DBP	Res.	Res.	Res.	Res.	Res.			
	Reset value													0					0	0	0	0	0	0	0				0	0	0	0	0
0x04	PWR_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLT_TIME[2:0]			FLTEN	Res.	PVDT[2:0]			Res.	SRCSEL	Res.	PVDE
	Reset value																					0	1	0	0	1	0	0	0		0		0
0x08	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x0C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x10	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	
0x14	PWR_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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

7. Reset

There are two types of resets defined as power reset and system reset.

7.1. Reset source

7.1.1. Power reset

A power reset sets all registers to their reset value, which occurs in the following situations:

- The POR/ BOR generated by the analog circuitry implements the detection of VCC. Release reset when the VCC voltage rises to a trigger value; generate reset when the VCC voltage falls to a certain trigger value;
- The PORI generated by the analog circuitry implements the detection of the VR output. Releasing reset when the Vcore voltage rises to a trigger value; generating reset when the Vcore voltage falls to a certain trigger value;

7.1.2. System reset

A system reset sets most registers to their reset values, except some special registers, such as the reset flag register.

A system reset generates 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)
- Power reset (POR/PDR, BOR)

7.1.3. NRST pin (external reset)

By loading the option byte (NRST_MODE bit), the NRST pin can be configured in the following modes (see option byte description for specific configuration):

- Reset input

In this mode, any valid reset signal on the NRST pin is passed to the internal logic, but the reset generated inside the chip is not output on the NRST pin.

In this configuration mode, the PF2 function of the GPIO is invalid.

After the NRST pin is input, an external reset of the chip is generated after the deburring circuit (deburring can be configured to be disabled).

- GPIO

In this mode, the PIN can be used as a standard GPIO, like PF2. The reset function on the pin does not work. Resets are only generated internally by the chip and cannot be passed to the pin.

Note: After power-on reset, NRST pin is configured to reset input mode by default.

7.1.4. Watchdog reset

See independent watchdog and system windows watchdog for details.

7.1.5. Software reset

A software reset can be achieved by setting the SYSRESETREQ bit in the ARM M0+ interrupt and reset control register.

7.1.6. Option byte loader reset

By configuring FLASH_CR.OBL_LAUNCH = 1, the software generates an option byte load reset, thereby starting the option byte load again.

8. Clock

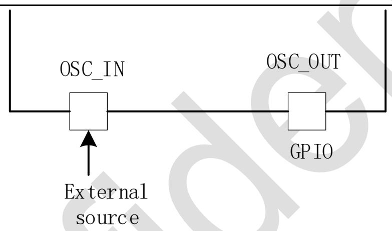
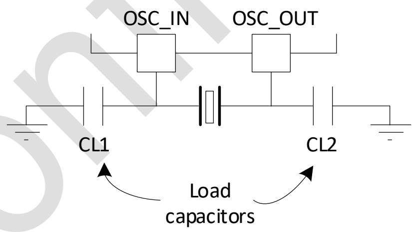
8.1. Clock source

8.1.1. High-speed external clock (HSE)

The high-speed external clock comes from two sources:

- External XTAL OSC + internal oscillation circuit
- External clock input via OSC_IN structure (HSEBYP=1)

Table 8-1 HSE/LSE clock sources

Clock source	Hardware configuration
External clock	
External crystal	

External high frequency OSC. Frequency range 4~32 MHz.

The stabilization time of HSE clock is configured by registers. When HSE goes from OFF to ON, it needs to wait for the stabilization time, and after stabilization, the hardware sets the RCC_CR.HSERDY register. When HSEBYP=1, the stabilization time is halved compared to non-bypass mode.

HSE clock related registers refer to RCC_ ECSCR.

8.1.2. High-speed internal clock (HSI)

Internal RC oscillator, reference frequency can be 4 MHz, 8 MHz, 16 MHz, 22.12 MHz and 24 MHz, compared to XTAL OSC, RC OSC has low power consumption and short stabilization time, but low accuracy. HSI analog module counts stabilization time, HSI output to system digital module at the same time.

After power-on reset, the HSI calibration value needs to be software loaded into the RCC_ ICSCR.HSITRIM register. This register will be reset when the system is reset.

After waking up from stop mode, only HSI can be used as the system clock source.

8.1.3. Low-speed internal clock (LSI)

Internal low frequency 32 KHz clock.

8.1.4. HSI10M Clock

This clock is used as a low-precision clock, as a filtered counter for the nRST pin, and for low-power processing when the FLASH is running at low speed.

8.1.5. PLL

PLL module reference clock is HSI or HSE, PLL input clock frequency range required for 12 MHz ~ 24 MHz, not in this range can not guarantee the output clock frequency and stability.

PLL supports 2x or 3x frequency, in the USB module work, you need to select the HSI reference frequency of 16 MHz, PLL multiplication factor of 3, resulting in 48 MHz frequency. At this time, if the system clock is selected PLL, the maximum frequency will also be limited to 48 MHz.

8.1.6. LSE Clock

The external 32.76 KHz OSC is used as a low-power clock.

A balance between stabilization time and power consumption can be done by configuring LSEDRV. the LSE stabilization time is configured by registers.

Similar to the HSE source, LSE has two sources:

- 32.768 K XTAL+ internal starting circuit
- External clock via OSC32_IN input (LSEBYP=1)

In the LSE bypass case, the stabilization time is halved compared to the non-bypass mode.

8.2. Clock tree

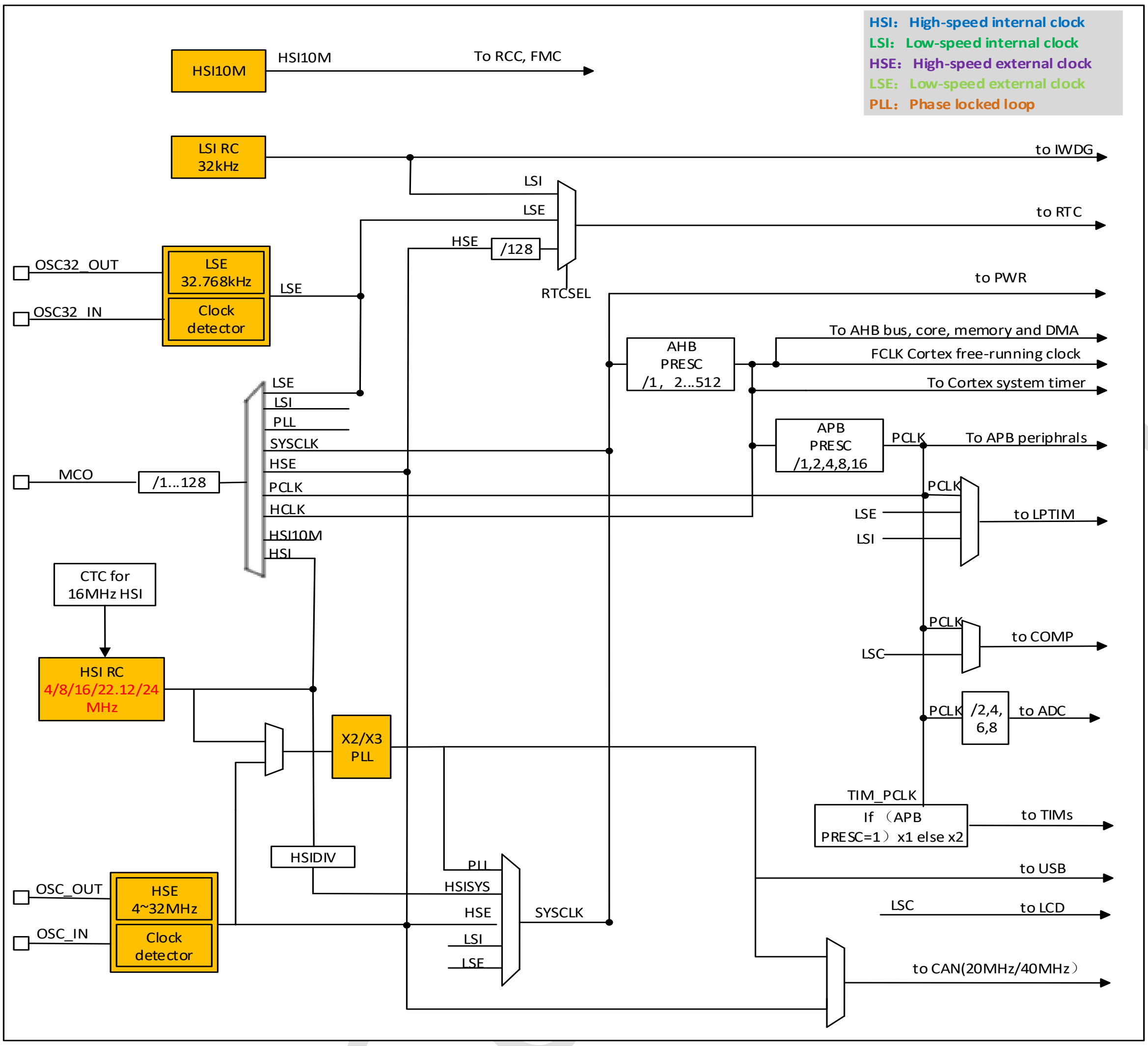


Figure 8-1 System clock structure

8.3. Clock Safety System (CSS)

Clock security includes the following main areas:

- Clock configuration and status security
- Clock source HSE security
- Clock source LSE security
- IWDG-based clock security
- Timer-based clock security
- Clock frequency calibration (TEST module implementation)

8.3.1. Clock configuration and state security

The software periodically reads back the clock configuration and status registers to obtain the system's current clock information and determine if it is consistent with expectations.

8.3.1.1. Clock source HSE monitoring

The HSE clock security system can be activated by software by configuring RCC_CR.CSSON. In this case, the clock detection function is turned on when HSE is activated. When the HSE is turned off, the clock detection function is turned off.

If a clock failure is detected on the HSE, the HSE is automatically turned off and a clock failure event is sent to the TIM1 (advanced timer) and TIM15/TIM16/TIM17 (general purpose timer) brake inputs and an interrupt is generated to notify the software of this failure (Clock Security System CSSI is linked to the NMI (Non-maskable inter-rupt) exception vector of the Cortex-M0+.

Note: Once CSS is enabled, and if the HSE clock fails, a CSS interrupt is generated and an NMI is automatically generated, which is executed continuously until the CSS interrupt pending bit is cleared. Therefore, the CSS interrupt must be cleared in the NMI handler by setting the CSSC bit in the Clock Interrupt Register (RCC_CICR).

If the HSE is used directly or indirectly as the system clock (indirect in the sense that it is used as an input to the PLL and the PLL is used as the system clock), clock failure will cause the system clock to automatically switch to the HSI and turn off the HSE. if clock failure occurs when the HSE is the input clock to the PLL, the PLL will also be turned off.

8.3.1.2. Clock source LSE monitoring

The LSE clock security system can be activated by software by configuring RCC_BDCR.LSECSSON. In this case, the clock detection function is turned on when LSE is activated. When the LSE is turned off, the clock detect function is turned off.

If a clock FAILURE is detected on the LSE, the LSE is automatically turned off and a clock FAILURE event is sent to the brake inputs of TIM1 (advanced timer) and TIM15/TIM16/TIM17 (general purpose timer) and an interrupt is generated to notify the software of this FAILURE (Clock Security System The CSSI is linked to the NMI (Non-maskable inter-rupt) exception vector of the Cortex-M0+.

Note: Once LSECSS is enabled, and if the LSE clock fails, a CSS interrupt is generated and an NMI is automatically generated, which is executed continuously until the CSS interrupt pending bit is cleared. Therefore, the CSS interrupt must be cleared in the NMI handler by setting the CSSC bit in the Clock Interrupt Register (RCC_CICR).

If LSE is used as the system clock, Clock Failure will cause the system clock to automatically switch to LSI while turning off LSE. also, if LPTIM and RTC count clocks select LSE, they will automatically switch to LSI as well.

8.4. Clock-out capability

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

Table 8-2 Output clock selection

Clock source	MCO output clock source
HSI	√
SYSCLK	√
HSE	√
LSI	√
PLL	√
LSE	√

Note: When switching the MCO clock source and selecting the GPIO AF function as the initial stage of the MCO, the MCO may generate glitches, and this period of time needs to be avoided.

8.5. Reset/Clock register

The registers of the module can be accessed in word (32 bits), half-word (16 bits) and byte (8 bits).

8.5.1. 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	PLLRDY	PLLON	Res	ADC_DIV		Res	CSS ON	HSE BYP	HSE RDY	HSE ON
						R	RW		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. Set by hardware to indicate that the PLL clock is locked 0: PLL unlocked 1: PLL locked
24	PLLON	RW	0	PLL enabled. This bit is cleared by hardware when entering stop mode. This bit cannot be reset if the PLL clock is used as the system clock. 0: PLL OFF 1: PLL ON
23	Reserved	-	-	Reserved
22:21	ADC_DIV	RW	0	ADC crossover factor 00: 2-division frequency 01: 4 crossover frequencies 10: 6 crossover frequencies 11: 8 crossover frequencies
20	Reserved	-	-	Reserved
19	CSSON	RS	0x0	HSE clock security system enable. When this bit is 1, the hardware enables the clock detection module if the HSE OSC is ready; if the HSE detection fails, the clock detection module is turned off. 0: clock security system off (clock detection off); 1: clock security system on (clock detection on if HSE clock is stable, otherwise clock detection off)
18	HSEBYP	RW	0	HSE shields the crystal and selects the pin input clock. This bit can only be written when HSEON=0. 0: HSE crystal not shielded, external high speed clock selects external crystal; 1: HSE crystal shielded, external high speed clock selects external pin input clock source;
17	HSERDY	R	0	HSE crystal clock ready flag. This bit is set by hardware to indicate that the HSE crystal is stable. 0: HSE crystal is not ready; 1: HSE crystal ready; Note: When HSEON is cleared, HSERDY is cleared after 6 HSE clock cycles.
16	HSEON	RW	0	HSE crystal enable.

Bit	Name	R/W	Reset Value	Function
				When the system enters stop mode, the hardware will clear this bit and turn off the HSE crystal. When HSE is used as the system clock source, this bit cannot be set to 0. 0: HSE crystal OFF; 1: HSE crystal ON;
15:14	Reserved	-	-	Reserved
13:11	HSIDIV	RW	0x0	Frequency division factor when the HSI generates 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. A hardware bit indicates that the HSI OSC is stable. This bit is valid only when HSION=1. 0: HSI OSC not ready; 1: HSI OSC ready;
9	Reserved	-		Reserved
8	HSION	RW	1	HSI clock enable. The hardware will clear this register to stop HSI as needed when it enters stop mode. 0: HSI OSC OFF; 1: HSI OSC ON;
7:0	Reserved	-		Reserved

8.5.2. Internal clock source calibration register (RCC_ICSCR)

Address offset:0x04

Reset value:0x00FF_1080

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

Bit	Name	R/W	Reset Value	Function
24:16	LSI_TRIM	RW	9'h0FF	The internal low-speed clock frequency is adjusted so that the internal low-speed clock can output 32.768 KHz by calibration. The calibration value is saved in Flash at the following address: 32.768 KHz calibration value address: 0x1FFF 3348
15:13	HSI_FS	RW	3'b000	HSI frequency selection: 000: 4 MHz 001: 8 MHz 010: 16 MHz 011: 22.12 MHz 100: 24 MHz ＞ = 101: 4 MHz
12:0	HSI_TRIM	RW	13'h1080	Clock frequency adjustment, changing the value of this register can adjust the output frequency of HSI. Each increase of the register value by 1 will increase the output frequency of HSI by about 0.2%, and the total adjustment range is 4~24 MHz. The calibration values corresponding to 24 MHz/22.12 MHz/16 MHz/8 MHz/4 MHz are stored in the Flash at the following addresses: 24 MHz calibration value address: 0x1FFF 3200 22.12 MHz calibration value address: 0x1FFF 3208 16 MHz calibration value address: 0x1FFF 3210 8 MHz calibration value address: 0x1FFF 3218 4 MHz calibration value address: 0x1FFF 3220

8.5.3. Clock configuration register (RCC_CFGR)

Address offset:0x08

Reset value:0x0000 0000

When the clock source is switched, there is 1 or 2 clock wait cycles to access this register.

When the APH or AHB division value is updated, there may be 0 to 15 clock cycles to access this register.

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	RW	RW	RW	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	RW	RW	RW	RW	RW			R	R	R	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30:28	MCOPRE[2:0]	RW	0	Microcontroller clock output (MCO) frequency division factor. Software controls these bits to set the division factor of the MCO output: 000: 1 001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128 Set these bits before enabling the MCO output.
27:24	MCOSEL[3:0]	RW	0	MCO selection 000: No clock, MCO output disabled 001: SYSCLK 010: HSI_10 M 011: HIS 100: HSE 101: PLL CLK 110: LSI 111: LSE Note: Incomplete output clock conditions may occur during the clock startup or switchover 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 factor 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. Software controls this bit. In order to generate the HCLK clock, it sets the frequency division factor of SYSCLK as follows: 0xxx: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 64 1101: 128 1110: 256 1111: 512

Bit	Name	R/W	Reset Value	Function
				In order to ensure the normal operation of the system, it is necessary to configure an appropriate frequency according to the VR power supply. Note: It is recommended to switch the frequency division factor step by step.
7:6	Reserved	-	-	Reserved
5:3	SWS[2:0]	R	0	System clock switch status bits 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 100: LSE Others: Reserved
2:0	SW[2:0]	RW	0	System clock source selection bits. Controlled by software and hardware, these bits select the system clock: 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE Others: Reserved 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, and PLL is the system clock source)

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:2	PLLMUL[1:0]	RW	2'b0	PLL frequency multiplication factor

Bit	Name	R/W	Reset Value	Function
				00: x2 01: x3 11: eserved
1:0	PLLSRC[1:0]	RW	0	PLL clock source selection. 00: No clock 01: Reserved 10: HSI 11: HSE

8.5.5. 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_DRV	
										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_DRV	
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:20	LSE_STARTUP	RW	0x0	LSE crystal 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: 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	-	-	Reserved
17:16	LSE_DRV	RW	0x3	LSE drive capability setting, default 11 00: reserved 01: Idd 315 nA, gm 3.5 uA/V 10: Idd 500 nA, gm 7.5 uA/V 11: Idd 630 nA, gm 10 uA/V
15:5	Reserved	RES	-	Reserved
4:3	HSE_STARTUP	RW	0x0	HSE stabilization time selection. HSEBYP=0. 00: 4096 HSE clocks; 01: 2048 HSE clocks;

Bit	Name	R/W	Reset Value	Function
				10: 8192 HSE clocks; 11: direct output, regardless of stabilization time; HSEBYP=1. 00: 2048 HSE clocks; 01: 1024 HSE clocks; 10: 4096 HSE clocks; 11: direct output, regardless of stabilization time;
2	Reserved	-	-	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

8.5.6. 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	PLL RDYIE	HSE RDYIE	HSI RDYIE	Res	LSE RDYIE	LSI RDYIE
										RW	RW	RW		RW	RW

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

8.5.7. 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 CSSF	CSSF	Res	Res	PLL RDYF	HSE RDYF	HSI RDYF	Res	LSE RDYF	LSI RDYF
						R	R			R	R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSF	R		LSE clock security system (CSS) interrupt flag. When hardware detects LSE, this register is set when the OSC clock fails. 0: LSE clock detection failure interrupt is not generated, 1: LSE clock detection failure interrupt generation, LSECSSC register 1 clears this bit.
8	CSSF	R	0	HSE clock security system interrupt flag. When hardware detects LSE, this register is set when the OSC clock fails. 0: HSE clock detection failure interrupt is not generated, 1: HSE clock detection failure interrupt generation, Write CSSC register 1 clears this bit.
7:6	Reserved	-	-	Reserved
5	PLLRDYF	R	0	PLL ready interrupt flag. This register is set by hardware when PLL lock and PLLRDYDIE=1. 0: PLL lock interrupt is not generated; 1: PLL lock interrupt is generated; Write PLLRDYC register 1 to clear this bit.
4	HSERDYF	R	0	HSE ready interrupt flag This bit is set by hardware when HSE is stable and HSER-DYIE is enabled. Software clears this bit by setting the HSERDYC bit. 0: No clock ready interrupt caused by HSE 1: Clock ready interrupt caused by HSE Write HSERDYC register 1 to clear the bit.
3	HSIRDYF	R	0	HSI clock ready interrupt flag. Hardware sets this register when the HSI clock is stable and HSIRDYIE=1. 0: HSI clock ready interrupt is not generated;

Bit	Name	R/W	Reset Value	Function
				1: HSI clock ready interrupt is generated; Write HSIRDYC register 1 to clear this bit.
2	Reserved	-	-	Reserved
1	LSERDYF	R	0	LSERDY clock ready interrupt flag. Hardware sets this register when the LSE clock is stable and LSERDYDIE=1. 0: LSERDY clock ready interrupt is not generated; 1: LSERDY clock ready interrupt is generated; Write LSERDYC register 1 to clear this bit.
0	LSIRDYF	R	0	LSI ready interrupt identification bit This bit is set by hardware when the LSI is stable and LSIRDYIE is enabled. Software clears this bit by setting the LSIRDYC bit. 0: No LSI-induced clock ready interrupt 1: There is a clock ready interrupt caused by LSI Write LSIRDYC register 1 to clear this bit.

8.5.8. 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	LSECSSC	CSSC	Res	Res	PLLRDYC	HSERDYC	HSIRDYC	Res	LSERDYC	LSIRDYC
						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 (CSS) interrupt flag is cleared. 0: No effect, 1: Clear the LSECSSF flag
8	CSSC	W	0	Clock safe interrupt clear bit. 0: No effect. 1: Clear the CSSF flag.
7:6	Reserved	-	-	Reserved
5	PLLRDYC	W	0	PLL ready flag is cleared. 0: No effect. 1: Clear the PLLRDYF bit.

Bit	Name	R/W	Reset Value	Function
4	HSERDYC	W	0	HSE ready flag is cleared. 0: No effect. 1: Clear the HSERDYF bit.
3	HSIRDYC	W	0	HSI ready flag is cleared. 0: No effect. 1: Clear the HSIRDYF bit.
2	Reserved	-	-	Reserved
1	LSERDYC	W	0	LSE ready flag is cleared. 0: No effect. 1: Clear the LSERDYF bit.
0	LSIRDYC	W	0	LSI ready flag is cleared. 0: No effect. 1: Clear the LSIRDYF bit.

8.5.9. 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	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	GPIOF RST	Res	Res	GPIO CRST	GPIOB RST	GPIOA RST
										RW			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 reset
4:3	Reserved	-	-	Reserved
2	GPIOCRST	RW	0	I/O PortC reset. 0: no effect, 1: PortF I/O reset
1	GPIOBRST	RW	0	I/O PortB resets. 0: no effect, 1: Port B I/O reset
0	GPIOARST	RW	0	I/O PortA resets. 0: no effect, 1: PortA I/O reset

8.5.10. AHB peripheral reset register (RCC_AHBRSTR)

Address offset:0x28

Reset value:0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	DIV RST	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	CRC RST	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMA RST
			RW												RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	DIVRST	RW	0	The divider module is reset. 0: no effect; 1: reset of the divider module;
23:13	Reserved			Reserved
12	CRCRST	RW	0	CRC module reset. 0: no effect, 1: CRC module reset,
11:9	Reserved	-	-	Reserved
8:1	Reserved	-	-	Reserved
0	DMARST	RW	0	DMA reset. 0: no effect, 1: DMA module reset

8.5.11. APB peripheral reset register 1 (RCC_APBRSTR1)

Address offset:0x2C

Reset value:0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM RST	OPA RST	DAC RST	PWR RST	CTC RST	Res	CAN RST	Res	USB RST	I2C2 RST	I2C1 RST	Res	USART4 RST	USART3 RST	USART2 RST	Res
RW	RW	RW	RW	RW		RW		RW	RW	RW		RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Res	SPI2RST	Res	Res	WWDGRST	RTCAPBRST	Res	Res	Res	Res	TIM7RST	TIM6RST	Res	Res	TIM3RST	TIM2RST
	RW			RW	RW					RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMRST	RW	0	LP Timer module reset. 0: no effect, 1: The module is reset,
30	OPARST	RW	0	OPA module reset. 0: no effect, 1: The module is reset,
29	DACRST	RW	0	DAC module reset. 0: no effect, 1: The module is reset,
28	PWRRST	RW	0	Power interface module reset. 0: no effect, 1: The module is reset,
27	CTCRST	RW	0	CTC module reset. 0: no effect, 1: The module is reset,
26	Reserved	-	-	Reserved
25	CANRST	RW	0	CAN module reset. 0: no effect, 1: The module is reset,
24	Reserved	-	-	Reserved
23	USBRST	RW		USB module reset. 0: no effect, 1: The module is reset,
22	I2C2RST	RW	0	I2C2 module reset. 0: no effect, 1: The module is reset,
21	I2C1RST	RW	0	I2C1 module reset. 0: no effect, 1: The module is reset,
20	Reserved	-	-	Reserved
19	USART4RST	RW	0	USART4 module reset. 0: no effect, 1: The module is reset,
18	USART3RST	RW	0	USART3 module reset. 0: no effect, 1: The module is reset,

Bit	Name	R/W	Reset Value	Function
17	USART2RST	RW	0	USART2 module reset. 0: no effect, 1: The module is reset,
16:15	Reserved	-	-	Reserved
14	SPI2RST	RW	0	SPI2 module reset. 0: no effect, 1: The module is reset,
13:12	Reserved	-	-	Reserved
11	WWDGRST	RW	0	WWDG module reset. 0: no effect, 1: The module is reset,
10	RTCAPBRST	RW	0	RTC module reset. 0: no effect, 1: The module is reset,
9:6	Reserved	-	-	Reserved
5	TIM7RST	RW	0	TIM7 module reset. 0: no effect, 1: The module is reset,
4	TIM6RST	RW	0	TIM6 module reset. 0: no effect, 1: The module is reset,
3:2	Reserved	-	-	Reserved
1	TIM3RST	RW	0	TIM3 module reset. 0: no effect, 1: The module is reset,
0	TIM2RST	RW	0	TIM2 module reset. 0: no effect, 1: The module is reset,

8.5.12. APB peripheral reset register 2 (RCC_APBRSTR2)

Address offset:0x30

Reset value:0x0000 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Re s	Re s	Res	Res	Re s	Re s	LCD RST	CO MP3 RST	COM P2 RST	COM P1 RST	Re s	TIM 17 RST	TIM 16 RST	TIM 15 RST
								RW	RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM 14	USAR T1	Re s	SPI 1	TIM 1	MCUD BG	AD C	Re s	Res	Res	Res	Res	Re s	Res	Res	SYS CFG

RST	RST		RS T	RS T	RST	RS T									RST
RW	RW		RW	RW	RW	RW									RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	LCDRST	RW	0	LCD module reset. 0: no effect, 1: The module is reset,
22	COMP3RST	RW	0	COMP3 module reset. 0: no effect, 1: The module is reset,
21	COMP2RST	RW	0	COMP2 module reset. 0: no effect, 1: The module is reset,
20	COMP1RST	RW	0	COMP1 module reset. 0: no effect, 1: The module is reset,
19	Reserved	-	-	Reserved
18	TIM17RST	RW	0	TIM17 module reset. 0: no effect, 1: The module is reset,
17	TIM16RST	RW	0	TIM16 module reset. 0: no effect, 1: The module is reset,
16	TIM15RST	RW	0	TIM15 module reset. 0: no effect, 1: The module is reset,
15	TIM14RST	RW	0	TIM14 module reset. 0: no effect, 1: The module is reset,
14	USART1RST	RW	0	USART1 module reset. 0: no effect, 1: The module is reset,
13	Reserved	-	-	Reserved
12	SPI1RST	RW	0	SPI1 module reset. 0: no effect, 1: The module is reset,
11	TIM1RST	RW	0	TIM1 module reset. 0: no effect, 1: The module is reset,
10	MCUDBGRST	RW	0	MCU Debug module reset. 0: no effect, 1: The module is reset,
9	ADCRST	RW	0	ADC module reset.

Bit	Name	R/W	Reset Value	Function
				0: no effect, 1: The module is reset,
8:1	Reserved	-	-	Reserved
0	SYSCFGRST	RWs	0	SYSCFG、COMP module reset. 0: no effect, 1: The module is reset,

8.5.13. I/O interface clock enable register (RCC_IOPENR)

Address offset:0x34

Reset value:0x0000 0000

This register is set and cleared by software.

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	GPIOF EN	Res	Res	GPIOC EN	GPIOB EN	GPIOA EN
										RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	GPIOFEN	RW	0	I/O PortF clock enable. 0: Clock disabled, 1: Clock enable
4:3	Reserved	-	-	Reserved
2	GPIOCEN	RW	0	I/O PortC clock enable. 0: Clock disabled, 1: Clock enable
1	GPIOBEN	RW	0	I/O PortB clock enable. 0: Clock disabled, 1: Clock enable
0	GPIOAEN	RW	0	I/O PortA clock enable. 0: Clock disabled, 1: Clock enable

8.5.14. AHB peripheral clock enable register (RCC_AHBENR)

Address offset:0x38

Reset value:0x0000 0300

This register is set and cleared by software.

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

Res	Res	Res	Res	Res	Res	Res	DIVEN	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	CRCEN	Res	Res	Res	FLASHEN	Res	Res	Res	Res	Res	Res	Res	DMAEN
			RW				RW								RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	DIVEN	RW	0	Divider module clock enable. 0: Disable 1: Enable
23:13	Reserved	-	-	Reserved
12	CRCEN	RW	0	CRC module clock enable. 0: Disable 1: Enable
11:10	Reserved	-	-	Reserved
9	SRAMEN	RW	1	In sleep mode, the clock enable control of SRAM 0: The module clock is disabled in sleep mode 1: The module clock is enabled in sleep mode Note: This bit only affects the clock enable of this module in sleep mode, in run mode, the clock of this module will not be disabled
8	FLASHEN	RW	1	In sleep mode, the clock enable control of FLASH 0: The module clock is disabled in sleep mode 1: The module clock is enabled in sleep mode Note: This bit only affects the clock enable of this module in sleep mode, in run mode, the clock of this module will not be disabled
7:1	Reserved	-	-	Reserved
0	DMAEN	RW	0	DMA module clock enable. 0: Disable 1: Enable

8.5.15. APB peripheral clock enable register 1 (RCC_APBENR1)

Address offset:0x3C

Reset value:0x0000 0000

This register is set and cleared by software.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIMEN	OPAEN	DACEN	PWREN	CTCEN	Res	CANEN	Res	USBEN	I2C2EN	I2C1EN	Res	USART4EN	USART3EN	USART2EN	Res
RW	RW	RW	RW	RW		RW		RW	RW	RW		RW	RW	RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	SPI2 EN	Re s	Res	WW DG EN	RTC APB EN	Res	Res	Res	Res	TIM7 EN	TIM 6EN	Res	Res	TIM3 EN	TIM2 EN
	RW			RW	RW		RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMEN	RW	0	LP Timer1 module clock enable. 0: Disable 1: Enable
30	OPAEN	RW	0	OPA module clock enable. 0: Disable 1: Enable
29	DACEN	RW	0	DAC module clock enable. 0: Disable 1: Enable
28	PWREN	RW	0	Low power control block clock enable. 0: Disable 1: Enable
27	CTCEN	RW	0	CTC module clock enable. 0: Disable 1: Enable
26	Reserved	-	-	Reserved
25	CANEN	RW		CAN module clock enable. 0: Disable 1: Enable
24	Reserved	-	-	Reserved
23	USBEN	RW	0	USB module clock enable. 0: Disable 1: Enable
22	I2C2EN	RW	0	I2C2 module clock enable. 0: Disable 1: Enable
21	I2C1EN	RW	0	I2C1 module clock enable. 0: Disable 1: Enable
20	Reserved	-	-	Reserved
19	USART4EN	RW	0	USART4 module clock enable. 0: Disable 1: Enable
18	USART3EN	RW	0	USART3 module clock enable. 0: Disable 1: Enable
17	USART2EN	RW	0	USART2 module clock enable. 0: Disable

Bit	Name	R/W	Reset Value	Function
				1: Enable
16:15	Reserved	-	-	Reserved
14	SPI2EN	RW		SPI2 module clock enable. 0: Disable 1: Enable
13:12	Reserved	-	-	Reserved
11	WWDGEN	RW	0	Window WDG module clock enable. 0: Disable 1: Enable This register is cleared by hardware system reset.
10	RTCAPBEN	RW	0	RTC Module APB clock enable. 0: Disable 1: Enable
9:6	Reserved	-	-	Reserved
5	TIM7EN	RW	0	TIM7 module clock enable. 0: Disable 1: Enable
4	TIM6EN	RW	0	TIM6 module clock enable. 0: Disable 1: Enable
3:2	Reserved	-	-	Reserved
1	TIM3EN	RW	0	TIM3 module clock enable. 0: Disable 1: Enable
0	TIM2EN	RW	0	TIM2 module clock enable. 0: Disable 1: Enable

8.5.16. APB peripheral clock enable register 2 (RCC_APBENR2)

Address offset:0x40

Reset value:0x0000 0000

This register is set and cleared by software.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	LCDEN	COMP3EN	COMP2EN	COMP1EN	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	MCUDBGEN	ADCEN	Res	Res	Res	Res	Res	Res	Res	Res	SYSCFG

															EN
RW	RW		R W	RW	RW	RW									R W

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	LCDEN	RW	0	LCD module clock enable. 0: Disable 1: Enable
22	COMP3EN	RW	0	COMP3 module clock enable. 0: Disable 1: Enable
21	COMP2EN	RW	0	COMP2 module clock enable. 0: Disable 1: Enable
20	COMP1EN	RW	0	COMP1 module clock enable. 0: Disable 1: Enable
19	Reserved	-	-	Reserved
18	TIM17EN	RW	0	TIM17 module clock enable. 0: Disable 1: Enable
17	TIM16EN	RW	0	TIM16 module clock enable. 0: Disable 1: Enable
16	TIM15EN	RW	0	TIM15 module clock enable. 0: Disable 1: Enable
15	TIM14EN	RW	0	TIM14 module clock enable. 0: Disable 1: Enable
14	USART1EN	RW	0	USART1 module clock enable. 0: Disable 1: Enable
13	Reserved	-	-	Reserved
12	SPIEN	RW	0	SPI1 module clock enable. 0: Disable 1: Enable
11	TIM1EN	RW	0	TIM1 module clock enable. 0: Disable 1: Enable
10	MCUDBGEN	RW-	0	MCUDBG module clock enable. 0: Disable 1: Enable
9	ADCEN	RW	0	ADC module clock enable.

Bit	Name	R/W	Reset Value	Function
				0: Disable 1: Enable
8:1	Reserved	-	-	Reserved
0	SYSCFGEN	RW	1	SYSCFG, COMP and VREFBUF module clock enable. 0: Disable 1: Enable

8.5.17. Peripheral independent clock configuration register (RCC_CCIPR)

Address offset:0x54

Reset value:0x0000 0000

This register is set and cleared by software.

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	LPTIM1SEL [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	COMP3 SEL	COMP2 SEL	COMP1 SEL	PVD SEL	CAN SEL	Res	Res	Res	Res	Res	Res
					RW	RW	RW	RW	RW						

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19:18	LPTIMSEL[1:0]	RW	2'b00	LPTIM1 internal clock source selection. 00: PCLK 01: LSI 10: No clock 11: LSE
17:11	Reserved	-	-	Reserved
10	COMP3SEL	RW	0	COMP3 module clock source selection. 0: PCLK 1: LSC (clock after RCC_BDCR.LSCOSEL selection) Note: Configure to select the LSC clock before enabling COMP3_FR.FLTEN.
9	COMP2SEL	RW	0	COMP2 module clock source selection. 0: PCLK 1: LSC (clock after RCC_BDCR.LSCOSEL selection) Note: Configure to select the LSC clock before enabling COMP2_FR2.FLTEN.
8	COMP1SEL	RW	0	COMP1 module clock source selection. 0: PCLK 1: LSC (clock after RCC_BDCR.LSCOSEL selection)

Bit	Name	R/W	Reset Value	Function
				Note: Configure this register to select the clock before enabling COMP1_FR1.FLTEN.
7	PVDSEL	RW	0	PVD detect clock source selection. 0: PCLK 1: LSC (clock after RCC_BDCR.LSCOSEL selection) Note: When PCLK is selected as clock source, PWR_CR1.FLTEN needs to be configured to 0. If FLTEN is enabled, LSC clock must be selected and LSC clock is configured to be selected before enabling FLTEN.
6	CANSEL	RW	0	CAN module clock source selection. 0: PLL 1: HSE
5:0	Reserved	-	-	Reserved

8.5.18. RTC domain control register (RCC_BDCR)

Address offset:0x5C

Reset value:0x0000 0000，reset by POR/BOR

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

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

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	LSCOSEL	RW	0	Low-speed clock selection. 0: LSI 1: LSE
24	LSCOEN	RW	0	Low-speed clock enable. 0: Disable 1: Enable
23:17	Reserved	-	-	Reserved
16	BDRST	RW	0	RTC domain soft reset. 0: No effect 1: Reset

Bit	Name	R/W	Reset Value	Function
15	RTCEN	RW		RTC clock enable. Software set or reset. 0: Disable 1: Enable
14:10	Reserved	-	-	Reserved
9:8	RTCSEL[1：0]	RW	0	RTC clock source selection. 00: No clock 01: LSE 10: LSI 11: HSE divided by 128 Once the RTC clock source is selected, it cannot be changed, except in the following cases: ■ RTC is reset to 00 ■ Selected as LSE (LSECSSD = 1) but no LSE
7	Reserved	-	-	Reserved
6	LSECSSD	R	0	CSS detect LSE failure. This bit is set by hardware to indicate that the CSS failed to detect the 32 KHz OSC (LSE). 0: Failure to detect LSE 1: Failure to detect LSE
5	LSECSSON	RW		CSS enables the LSE clock. 0: disable; 1: enable; Note: LSEON=1 and LSERDY=1 must be enabled before LSECSSON can be enabled. Once this bit is enabled, it cannot be disabled again un-less LSECSSD=1.
4:3	Reserved	-	-	Reserved
2	LSEBYP	RW		LSE OSC bypass 0: Not bypassed, low-speed external clock selects the crystal; 1: Bypassed, low-speed external clock selects the external interface input clock; Note: This bit can only be written when the external 32 KHz OSC is disabled (LSEON=0 and LSERDY=0).
1	LSERDY	R		LSE OSC ready. Hardware configuration of this bit to 1 indicates that the LSE clock is ready.
0	LSEON	RW		LSE OSC enable. 0: disable; 1: enable;

8.5.19. Control/status register (RCC_CSR)

Address offset:0x60

Reset value:0x0000 0000

The reset flag bit in this register can only be reset by power reset, the others are reset by system re-set.

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	WWDGRSTF	IWDGRSTF	SFTRSTF	PWRRSTF	PINRSTF	OBLRSTF	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	PINRST_FLTDIS	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. Setting RMVF to 1 clears this bit.
29	IWDGRSTF	R	0	IWDG reset flag. Setting RMVF to 1 clears this bit.
28	SFTRSTF	R	0	Soft reset flag. Setting RMVF to 1 clears this bit.
27	PWRRSTF	R	0	BOR/POR/PDR reset flag. Setting RMVF to 1 clears this bit.
26	PINRSTF	R	0	External NRST pin reset flag. Setting RMVF to 1 clears this bit.
25	OBLRSTF	R	0	Option byte loader reset flag. Setting RMVF to 1 clears this bit.
24	Reserved	-	-	Reserved
23	RMVF	RW	0	Reset flags [30:25] need to be cleared by software.
22:9	Reserved	-	-	Reserved
8	PINRST_FLTDIS	RW	0	NRST filter width 20 us disable 0: HSI_10 M enabled, and the filtering 20 us width function is enabled 1: The filtering function is disabled and synchronized to the system clock to generate a system reset
7:2	Reserved	-	-	Reserved
1	LSIRDY	R	0	LSI OSC stable flag. 0: LSI is not stable 1: LSI has stabilized
0	LSION	RW	0	LSI OSC enable. 0: Disable

Bit	Name	R/W	Reset Value	Function
				1: Enabled Hardware enable for analog LSI: 1. hardware IWDG enable; 2. LSECSS enable;

8.5.20. RCC register address map

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	32	RCC_CR	Reserved						PLL_RDY	PLLON	Reserved	ADC_DIV		Reserved	CSSON	HSEBYP	HSERDY	HSEON	Reserved		HSIDIV[2:0]			HSIRDY	Reserved	HSION	Reserved							
		Read/Write							R	RW		RW			RW	RW	R	RW			RW	RW	RW	R		RW								
		Reset Value	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0x04	32	RCC_ICSCR	Reserved							LSI_TRIM[8:0]									HSI_FS[2:0]			HSI_TRIM[12:0]												
		Read/Write								RW									RW															
		Reset Value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	1
0x08	32	RCC_CFGR	Reserved	MCOPRE[2:0]			MCOSEL[3:0]				Reserved									PPRE[2:0]			HPRE[3:0]				Reserved		SW[2:0]			SW[2:0]		
		Read/Write		RW			RW													RW			RW						R			RW		
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0C	32	RCC_PLLCFGR	Reserved																												PLLMUL[1:0]		PLLSRC[1:0]	
		Read/Write																													RW		RW	
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x010	32	RCC_ECSCR	Reserved										LSE_STARTUP[1:0]		Reserved		LSE_DRV[1:0]		Reserved											HSE_STARTUP[1:0]		Reserved	HSE_DRV[1:0]	
		Read/Write											RW				RW													RW			RW	
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	32	RCC_CIER	Reserved																										PLLRDYIE	HSERDYIE	HSIRDYIE	Reserved	LSERDYIE	LSIRDYIE
		Read/Write																											RW	RW	RW		RW	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	32	RCC_CIFR	Reserved																						LSECSSF	CSSF	Reserved		PLLRDYF	HSERDYF	HSIRDYF	Reserved	LSERDYF	LSIRDYF
		Read/																							R	R			R	R	R		R	R

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Write																																
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	32	RCC_CICR	Reserved																						LSECSSC	CSSC	Reserved		PLLRDYC	HSERDYC	HSIRDYC	Reserved	LSERDYC	LSIRDYC
		Read/Write																							W	W			W	W	W		W	W
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	32	RCC_IOPRSTR	Reserved																										GPIOFRST	Reserved		GPIOCRST	GPIOBRST	GPIOARST
		Read/Write																											RW			RW	RW	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	32	RCC_AHBRSTR	Reserved							DIVRST	Reserved											CRCRST	Reserved											DMARST
		Read/Write								RW												RW												RW
		Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Value																																
0x2C	32	RCC_APBRSTR1	LPTIMRST	OPARST	DACRST	PWRRST	CTCRST	Reserved	CANRST	Reserved	USBRST	I2C2RST	I2C1RST	Reserved	USART4RST	USART3RST	USART2RST	Reserved		SPI2RST	Reserved		WWDGRST	RTCAPBRST	Reserved				TIM7RST	TIM6RST	Reserved		TIM3RST	TIM2RST
		Read/Write	RW	RW	RW	RW	RW		RW		RW	RW	RW		RW	RW	RW			RW			RW	RW					RW	RW			RW	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	32	RCC_APBRSTR2	Reserved								LCDRST	COMP3RST	COMP2RST	COMP1RST	Reserved	TIM17RST	TIM16RST	TIM15RST	TIM14RST	USART1RST	Reserved	SPI1RST	TIM1RST	MCUDBGRST	ADCRST	Reserved								SYSCFGRST
		Read/Write									RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW									RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	32	RCC_IOPENR	Reserved																										GPIOFEN	Reserved		GPIOCEN	GPIOBEN	GPIOAEN
		Read/Write																											RW			RW	RW	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x38	32	RCC_AHBENR	Reserved							DIVEN	Reserved											CRCEN	Reserved		SRAMEN	FLASHEN	Reserved							DMAEN
		Read/Write								RW												RW			RW	RW								RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	32	RCC_APBENR1	LPTIMEN	OPAEN	DACEN	PWREN	CTCEN	Reserved	CANEN	Reserved	USBEN	I2C2EN	I2C1EN	Reserved	USART4EN	USART3EN	USART2EN	Reserved		SPI2EN	Reserved		WWDGEN	RTCAPBEN	Reserved				TIM7EN	TIM6EN	Reserved		TIM3EN	TIM2EN
		Read/Write	RW	RW	RW	RW	RW		RW		RW	RW	RW		RW	RW	RW			RW			RW	RW					RW	RW			RW	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	32	RCC_APBENR2	Reserved								LCDEN	COMP3EN	COMP2EN	COMP1EN	Reserved	TIM17EN	TIM16EN	TIM15EN	TIM14EN	USART1EN	Reserved	SPI1EN	TIM1EN	MCUDBGEN	ADCEN	Reserved								SYSCFGEN
		Read/Write									RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW									RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x54	32	RCC_CCIPR	Reserved												LPTIMSEL[1:0]		Reserved							COMP3SEL	COMP2SEL	COMP1SEL	PVDSEL	CANSEL	Reserved					

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Read/Write													RW									RW	RW	RW	RW	RW						
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	32	RCC_BDCR	Reserved						LSCOSEL	LSCOEN	Reserved							BDRST	RTCEN	Reserved					RTCSEL[1:0]		Reserved	LSECSSD	LSECSSON	Reserved		LSEBYP	LSERDY	LSEON
		Read/Write							RW	RW								RW	RW						RW			R	RW			RW	R	RW
		Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	32	RCC_CSR	Reserved	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLSTF	Reserved	RMVF	Reserved														PINRST_FLTDIS	Reserved						LSIRDY	LSION
		Read/Write		R	R	R	R	R	R		RW															RW							R	RW
		Reset Value	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9. Clock Calibration Controller (CTC)

9.1. CTC introduction

The clock calibration controller (CTC) uses hardware to automatically calibrate the RC crystal (HSI) when the internal configuration is 16 MHz and use the 3x PLL (48 M) as the clock source of the USBD module, hereinafter the PLL generated by the HSI configuration of 16 M and 3x frequency is referred to as PLL48M. The CTC module is based on an external high-precision reference source to calibrate the HSI clock frequency by automatically or manually adjusting the calibration value to obtain an accurate PLL48M clock.

9.2. CTC main features

- Three external reference signal sources: GPIO, LSE clock, USBD_SOF
- Provision of software-referenced synchronization pulses;
- Hardware auto-calibration without software operation;
- 16 bits calibration counter with reference signal source capture and reload function;
- 8 bits of clock calibration base for frequency evaluation and auto-calibration;
- Flag bits and interrupts to indicate the status of the clock calibration: calibration success status (CKOKIF), warning status (CKWARNIF) and error status (ERRIF).

9.3. CTC Function Description

9.3.1. CTC Block Diagram

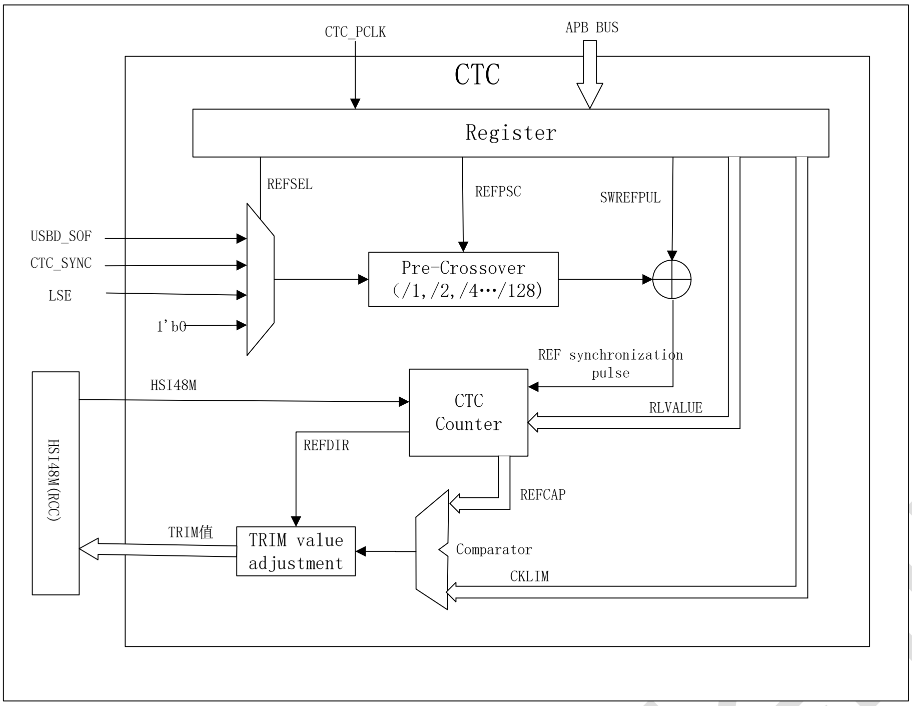


Figure 9-1 CTC Architecture Block Diagram

9.3.2. REF synchronous pulse generator

First, the reference signal source: GPIO, LSE clock output or USBD_SOF is selected by setting the REFSEL bit in the CTC_CTL1 register (CTC control register 1).

Then, the signal polarity when synchronizing the reference source can be configured by setting the REFPOL bit in the CTC_CTL1 register, and a suitable synchronous clock frequency signal (no greater than 48KHz) can be generated by setting the REFPSC bit in the CTC_CTL1 register.

If a software reference pulse signal needs to be used, the SWREFPUL bit in the CTC_CTL0 register (CTC control register 0) needs to be set to 1. The software reference pulse signal and the external reference pulse signal perform a logical 'or' operation at the end.

9.3.3. CTC calibration counter

The CTC clock calibration counter is clocked by the PLL48M. After setting the CNTEN bit in the CTC_CTL0 register, the counter starts counting down from the RLVALUE value (RLVALUE is defined in the CTC_CTL1 register) when the first REF sync pulse signal is detected. Each time a REF synchronization pulse signal is detected, the counter reloads the RLVALUE value and starts counting down again. If the REF sync pulse signal is never detected, the counter counts down to zero, then counts up to 128 x CKLIM (CKLIM is defined in CTC_CTL1), and finally stops until the next REF sync pulse signal is detected. Once the REF sync pulse signal is detected, the count value of the current CTC calibration counter is captured and stored in the REFCAP bit in CTC_STAT CTC Status Register), and the count direction of the current counter is stored in the REFDIR bit in CTC_STAT. The details are shown in the figure below.

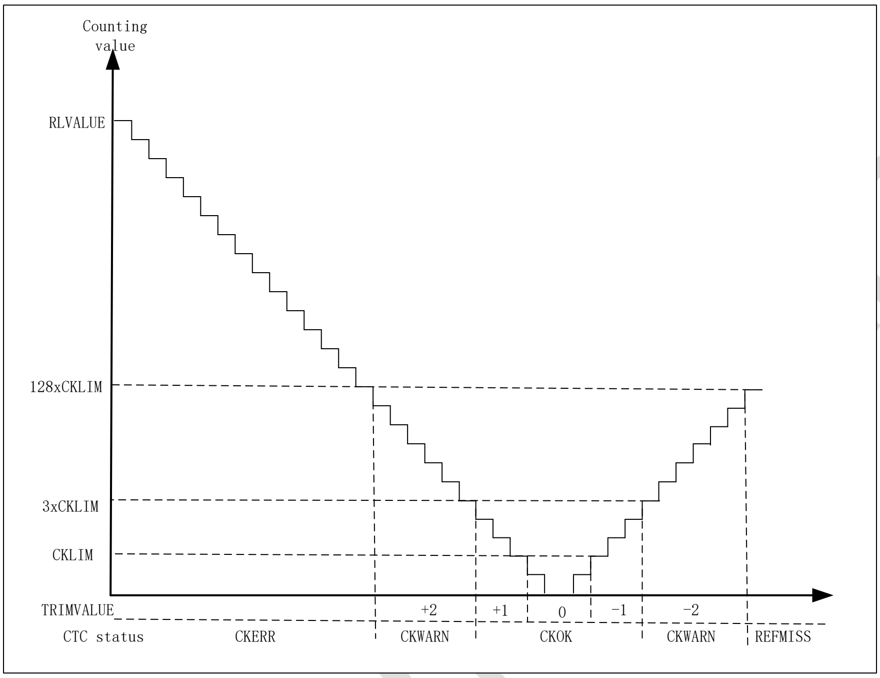


Figure 9-2 CTC calibration counter

9.3.4. Frequency evaluation and automatic calibration process

When the REF synchronization pulse signal appears, the clock frequency evaluation function starts to execute. If the REF synchronization pulse signal appears during the counter counting down, the current clock frequency is slower than the desired clock frequency (frequency of 48 M), and the TRIMVALUE value (clock calibration value) in CTC_CTL0 needs to be increased. If the REF synchronization pulse signal appears during the counter up counting, it means the current clock frequency is faster than the desired clock frequency, and the TRIMVALUE value needs to be decreased. The CKOKIF bit, CKWARNIF bit, CKERR bit and REFMISS bit in CTC_STAT reflect the status of the frequency evaluation.

If the AUTOTRIM (hardware auto-calibration mode) position 1 in CTC_CTL0, the hardware auto-calibration mode is enabled. In this mode, if the REF sync pulse signal appears during the counter counting down, it means the current clock frequency is slower than the desired clock frequency, and the TRIMVALUE value in CTC_CTL0 will be increased automatically to increase the current clock frequency. Conversely, if the REF synchronization pulse signal appears in the process of counter up counting, it means the current clock frequency is faster than the desired clock frequency, and the TRIMVALUE value will be automatically decreased to reduce the current clock frequency.

- When Counter < CKLIM, the REF synchronization pulse signal is detected:
  The CKOKIF bit (clock calibration success flag bit) in CTC_STAT is set, and an interrupt will be generated if the CKOKIE bit (clock calibration completion interrupt enable bit) in CTC_CTL0 is set to 1. If AUTOTRIM in CTC_CTL0 is set to 1, the TRIMVALUE value in CTC_CTL0 remains unchanged.
- When CKLIM ≤ Counter < 3 x CKLIM, the REF sync pulse signal is detected:
  The CKOKIF bit in CTC_STAT is set and an interrupt will be generated if CKOKIE position 1 in CTC_CTL0. If the AUTOTRIM position in CTC_CTL0 is 1, the TRIMVALUE value in CTC_CTL0 will be added by 1 during the count down and subtracted by 1 during the count up.
- When 3 x CKLIM ≤ Counter < 128 x CKLIM, the REF synchronization pulse signal is detected:
  The CKWARNIF bit (Clock Calibration Warning Interrupt bit) in CTC_STAT is set, and an interrupt will be generated if the CKWARNIE bit (Clock Calibration Warning Interrupt Enable bit) in CTC_CTL0 is set to 1. If the AUTOTRIM bit in CTC_CTL0 is set to 1, the TRIMVALUE value in CTC_CTL0 will be added by 2 during the count down and subtracted by 2 during the count up.
- Counter ≥ 128 x CKLIM, the counter detects the REF synchronization pulse signal during the count down:
  The CKERR bit (clock calibration error bit) in CTC_STAT is set, and an interrupt will be generated if the ERRIE bit (error interrupt enable bit) in CTC_CTL0 is set to 1. The TRIMVALUE value in CTC_CTL0 remains unchanged.
- Counter = 128 x CKLIM, the counter is counting upwards during:
  REFMISS bit (REF synchronous pulse loss bit) in CTC_STAT is set and an interrupt will be generated if ERRIE position 1 in CTC_CTL0. The TRIMVALUE value in CTC_CTL0 remains unchanged.

If the calibration value of TRIMVALUE in CTC_CTL0 is greater than 127, an overflow event will occur, and if the calibration value of TRIMVALUE is less than 0, a underflow event will occur. The value of TRIMVALUE is from 0 to 127 (when an overflow event occurs, TRIMVALUE value is 127; when a underflow event occurs, TRIMVALUE value is 0). Then, the TRIMERR bit (calibration value error bit) in CTC_STAT will be set and an interrupt will be generated if the ERRIE position 1 in CTC_CTL0 is set.

9.3.5. Software Programming Guidelines

The RLVALUE bit and the CKLIM bit in CTC_CTL1 are key for clock frequency evaluation and hardware auto-calibration. Their values are calculated from the frequency of the desired clock (PLL: 48 MHz) and the frequency of the REF synchronization pulse signal. Ideally, the REF sync pulse signal appears when the CTC counter counts to zero, so the value of RLVALUE is:

RLVALUE = (Fclock ÷ FREF) – 1

The value of CKLIM is set by the user according to the accuracy of the clock, and it is generally recommended to set it to half of the step size, so the value of CKLIM is:

CKLIM = (Fclock ÷ FREF) × 0.12% ÷ 2

The typical step value is 0.12%, Fclock is the frequency of the desired clock (48 MHz), FREF is the frequency of the REF synchronization pulse signal.

TRIMVALUE in CTC_CTL0 can be written by software when AUTOTRIM bit is 0. However, modifying TRIMVALUE will directly affect the frequency of the HSI clock, therefore, TRIMVALUE should not be modified by software at will. it is recommended that users modify it in the middle of two reference signals, judging from the flag bit (see Frequency evaluation and auto-calibration process for details); Or if a reliable value already exists, the user can directly modify the value of HSI_TRIM in RCC_ICSCR.

9.4. CTC Register

9.4.1. CTC control register 0 (CTC_CTL0)

Address offset:0x00

Reset value:0x0000 4000

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	TRIMVALUE[6:0]							SW REF PUL	AU- TO- TRI M	CNT EN	Res	ERE FIE	ERR IE	CK WA RNI E	CKO KIE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
14:8	TRIMVALUE[6:0]	RW	7'b1000000	PLL48M Calibration Value. When the AUTOTRIM value in CTC_CTL0 is 0, this bit is set and cleared by software and this mode is used for the software calibration process. When the AUTOTRIM value in CTC_CTL0 is 1, this bit is read-only and is automatically modified by hardware; this mode is used for the hardware calibration process. The middle value of TRIMVALUE is 64, when TRIMVALUE value is added by 1, the PLL48M clock frequency is increased by approximately 48KHz. When TRIMVALUE value is subtracted by 1, the PLL48M clock frequency is decreased by approximately 48 KHz.
7	SWREFPUL	RW	0	Software generated synchronous reference signal pulse. This bit is set by software and provides a synchronous reference pulse signal to the CTC counter. This bit is automatically cleared by hardware and returns 0 on read operation. 0: no effect; 1: software generates a synchronous reference pulse signal;
6	AUTOTRIM	RW	0	Hardware auto-calibration mode. This bit is set or cleared by software. When this position is 1, the hardware auto-calibration mode is enabled and the TRIMVALUE value in CTC_CTL0 is continuously and automatically modified by hardware until the clock frequency of PLL48M reaches 48 MHz. 0: Disable hardware auto-calibration mode 1: Enable hardware auto-calibration mode
5	CNTEN	RW	0	CTC Counter Enable. This bit is set or cleared by software and is used to enable or disable the CTC counter. When this position is 1, the value of CTC_CTL1 cannot be modified. 0: disables the CTC counter 1: Enables the CTC counter

Bit	Name	R/W	Reset Value	Function
4	Reserved	-	-	Reserved
3	EREFIE	RW	0	Expected reference signal interrupt enable. 0: Disable interrupt generation by the desired reference signal 1: Enable the desired reference signal to generate interrupts
2	ERRIE	RW	0	Error interrupt enable. 0: Disable error interrupt 1: Enable error interrupt
1	CKWARNIE	RW	0	Clock calibration warning interrupt enable. 0: Disable clock calibration warning interrupt 1: Enable clock calibration warning interrupt
0	CKOKIE	RW	0	Clock calibration completion interrupt enable. 0: Disable clock calibration completion interrupt 1: Enable clock calibration completion interrupt

9.4.2. CTC control register 1 (CTC_CTL1)

Address offset:0x04

Reset value:0x2022 BB7F

This register can only be accessed by word (32 bits). When CNTEN is 1, the value of this register cannot be modified.

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

Bit	Name	R/W	Reset Value	Function
31	REFPOL	RW	0	Reference Signal Source Polarity. This bit is set or cleared by software and is used to select the synchronous polarity of the reference signal source. 0: Selects the rising edge 1: Selects the falling edge
30	Reserved	-	-	Reserved
29:28	REFSEL[1:0]	RW	2'b10	Reference Signal Source Select. This bit is set or cleared by software and is used to select the reference signal source. 00: Selects the GPIO input signal 01: Selects the LSE clock

Bit	Name	R/W	Reset Value	Function
				10: Select USBD_SOF signal 11: Reserved, select 0
27	Reserved	-	-	Reserved
26:24	REFPSC[2:0]	RW	3'b000	Reference signal source prescaler. This bit is set or cleared by software. 000: Reference signal is not crossed over 001: Reference signal 2-division 010: Reference signal 4-division 011: Reference signal 8 divisions 100: Reference signal 16 divisions 101: Reference signal 32 divisions 110: Reference signal 64 divisions 111: Reference signal 128 divisions
23:16	CKLIM[7:0]	RW	0x22	Clock Calibration Timebase Limit. This bit is set or cleared by software and is used to define the clock calibration time base limits. This bit is used for frequency evaluation and automatic calibration processes.
15:0	RLVALUE[15:0]	RW	0xBB7F	CTC Counter Reload Value. This bit is set or cleared by software and is used to define the CTC counter reload value that will be reloaded into the CTC calibration counter when a synchronous reference pulse is detected.

9.4.3. CTC Status Register 1 (CTC_SR)

Address offset:0x08

Reset value:0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REFCAP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REFDIR	Res	Res	Res	Res	TRIMERR	REFMISS	CKERR	Res	Res	Res	Res	EREFIF	ERRIF	CKWARNIF	CKOKIF
R					R	R	R					R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	REFCAP[15:0]	R	0	The CTC counter captures the value. When a synchronous reference pulse signal is detected, the count value in the CTC calibration counter is stored in the REFCAP bit.
15	REFDIR	R	1	CTC calibrates the clock count direction.

Bit	Name	R/W	Reset Value	Function
				When a synchronous reference pulse signal is detected, the CTC calibration counter count direction is stored in the REFDIR bit. 0: Count up 1: Count down
14:11	Reserved	-	-	Reserved
10	TRIMERR	R	0	Calibration value error bit. This bit is set by hardware when the TRIMVALUE value in CTC_CTL0 overflows or underflows. If ERRIE position 1 in CTC_CTL0, an interrupt will be generated. The TRIMERR bit can be cleared to zero by writing 1 to the ERRIC bit in CTC_INTC. 0: No calibration value error occurred 1: Calibration value error occurred
9	REFMISS	R	0	Synchronous reference pulse signal is lost. This bit is set by hardware when the synchronous reference pulse signal is lost. The REFMISS position bit when the CTC calibration counter counts up to 128 x CKLIM without detecting the synchronous reference pulse signal during incremental counting. This indicates that the current clock is too fast to calibrate to the desired frequency value, or that some other error has been generated. The REFMISS bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No synchronous reference pulse signal is missing 1: Loss of synchronous reference pulse signal Note: To prevent the REFMISS from setting again after clearing the REFMISS and the ERRIF generated at the same time, you can write the ERRIC bit repeatedly until the flag bit is no longer set by clearing the CNTEN bit first.
8	CKERR	R	0	Clock calibration error bit. This bit is set by hardware when a clock calibration error is generated. When the CTC calibration counter count value is greater than or equal to 128 x CKLIM during decimal counting and a synchronous reference pulse signal is detected, CKERR is set, indicating that the current clock is too slow to calibrate to the desired frequency value. When ERRIE in CTC_CTL0 is set to 1, an interrupt is generated. The CKERR bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No clock calibration error occurs 1: A clock calibration error occurred
7:4	Reserved	-	-	Reserved
3	EREFIF	R	0	Desired reference interrupt flag bit.

Bit	Name	R/W	Reset Value	Function
				When the CTC calibration clock counter counts to 0, a reference signal is detected and this bit is set by hardware. When EREFIE in CTC_CTL0 is set to 1, an interrupt is generated. The EREFIF bit can be cleared to zero by writing 1 to the EREFIC bit in CTC_INTC. 0: No expected reference signal generation 1: Expected reference signal generation
2	ERRIF	R	0	Error interrupt flag bit. This bit is set by hardware when an error occurs. This bit is set whenever a TRIMERR, REFMISS or CKERR error occurs. When ERRIE in CTC_CTL0 is set, an interrupt is generated. The ERRIF bit can be cleared by writing 1 to the ERRIC bit in CTC_INTC. 0: No error occurred 1: Error occurred
1	CKWARNIF	R	0	Clock calibration warning interrupt flag bit. This bit is set by hardware when a clock calibration warning is generated. CKWARNIF is set when the CTC calibration counter count is greater than or equal to 3xCKLIM and less than 128xCKLIM, and a synchronous reference pulse signal is detected. This indicates that the current clock frequency is too slow or too fast, but that the desired frequency value can be achieved by calibration. When the clock calibration warning is generated, the TRIMVALUE value is added by 2 or subtracted by 2. When CKWARNIE in CTC_CTL0 is set to 1, an interrupt is generated. The CKWARNIF bit can be cleared to zero by writing 1 to the CKWARNIC bit in CTC_INTC. 0: No clock calibration warning occurs 1: A clock calibration warning has occurred
0	CKOKIF	R	0	Clock calibration success interrupt flag bit. This bit is set by hardware when the clock calibration is successful. If a synchronous reference pulse signal is detected when the CTC calibration counter count value is less than 3 x CKLIM, CKOKIF is set. This means that the current clock frequency is normal and can be used without clock calibration by TRIMVALUE value. When CKOK-IE in CTC_CTL0 is set to 1, an interrupt is generated. The CKOKIF bit can be cleared to zero by writing 1 to the CKOKIC bit in CTC_INTC. 0: clock calibration unsuccessful 1: Clock calibration successful

9.4.4. CTC interrupt clear register (CTC_INTC)

Address offset:0x0C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	ERE-FIC	ERRIC	CKWARNIC	CKOKIC
												W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	EREFIC	W	0	EREFIF interrupt clear bit. This bit can only be written by software, read operation returns 0. Writing 1 clears the EREFIF bit in CTC_STAT, writing 0 has no effect.
2	ERRIC	W	0	ERRIF interrupt clear bit. This bit can only be written by software, and a read operation returns 0. A write of 1 clears the ERRIF, TRIMERR, REFMISS, and CKERR bits in CTC_STAT; a write of 0 has no effect.
1	CKWARNIC	W	0	CKWARNIF interrupt clear bit. This bit can only be written by software, a read operation returns 0. A write of 1 clears the CKWARNIF bit in CTC_STAT, a write of 0 has no effect.
0	CKOKIC	W	0	CKOKIF interrupt clear bit. This bit can only be written by software, read operation returns 0. Writing 1 clears the CKOKIF bit in CTC_STAT, writing 0 has no effect

9.4.5. CTC register Image

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CTC_CTL0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRIMVALUE[6:0]							SWREFPUL.	AUTOTRIM.	CNTEN.	Res.	EREFIE.	ERRIE.	CKWARNIE.	CKOKIE.
	Reset va																		7'b1000000							0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	lue																																
0x04	CTC_CTL1	Res.	Res.	Res.	Res.	Res.	REFPSC[2:0]			CKLIM[7:0]								RLVALUE[15:0]															
	Reset value						0			0x22								0xBB7F															
0x08	CTC_SR	REFCAP[15:0]																REFDIR.	Res.	Res.	Res.	Res.	TRIMERR.	REFMISS.	CKERR.	Res.	Res.	Res.	Res.	EREFIF.	ERRIF.	CKWARNIF.	CKOKIF.
	Reset value	0																1					0	0	0					0	0	0	0
0x0C	CTC_INTC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EREFIC.	ERRIC.	CKWARNIC.	CKOKIC.
	Reset value																													0	0	0	0

10. General-purpose I/Os (GPIO)

10.1. GPIO introduction

The GPIOs contain PA[15:0], PB[15:0], PC[15:0] and PF[9:0], and each GPIO port has:

- Four 32-bit configuration registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR)
- Two 32-bit data registers (GPIOx_IDR and GPIOx_ODR)
- One 32-bit set/reset register (GPIOx_BSRR)
- One 32-bit lock register (GPIOx_LCKR)
- Two alternate function selection registers (GPIOx_AFRH and GPIOx_AFRL).

10.2. GPIO main features

- Register support IO Port/AHB bus read/write
- Output status: push-pull or open drain + pull-up/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)
- Set/reset register (GPIOx_BSRR) for bitwise write access to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configuration function
- Analog function
- Alternate functions selection registers (at most 16 AFs per I/O port)
- Fast toggle capable of changing every single cycle
- Highly flexible pin multiplexing allows the use of I/O pins as GPIOs or as one of several peripheral function

10.3. GPIO functional description

Each port bit of the GPIO ports can be individually configured by software in several modes:

- Input floating

- Input pull-up
- Input pull-down
- Analog input
- Output open-drain with pull-up or pull-down capability
- Output puss-pull with pull-up or pull -down capability
- Alternate function push-pull with pull-up or pull–down capability
- Alternate function open-drain with pull-up or pull-down capability

Each I/O port bit is freely programmable, however the I/O port registers have to be accessed as 32-bit words, half-words or bytes. The purpose of the GPIOx_BSRR and GPIOx_BRR registers is to allow atomic read/modify accesses to any of the GPIOx_ODR registers. In this way, there is no risk of an IRQ occurring between read and modify access.

The following diagram gives the basic structure of an I/O port (1bit).

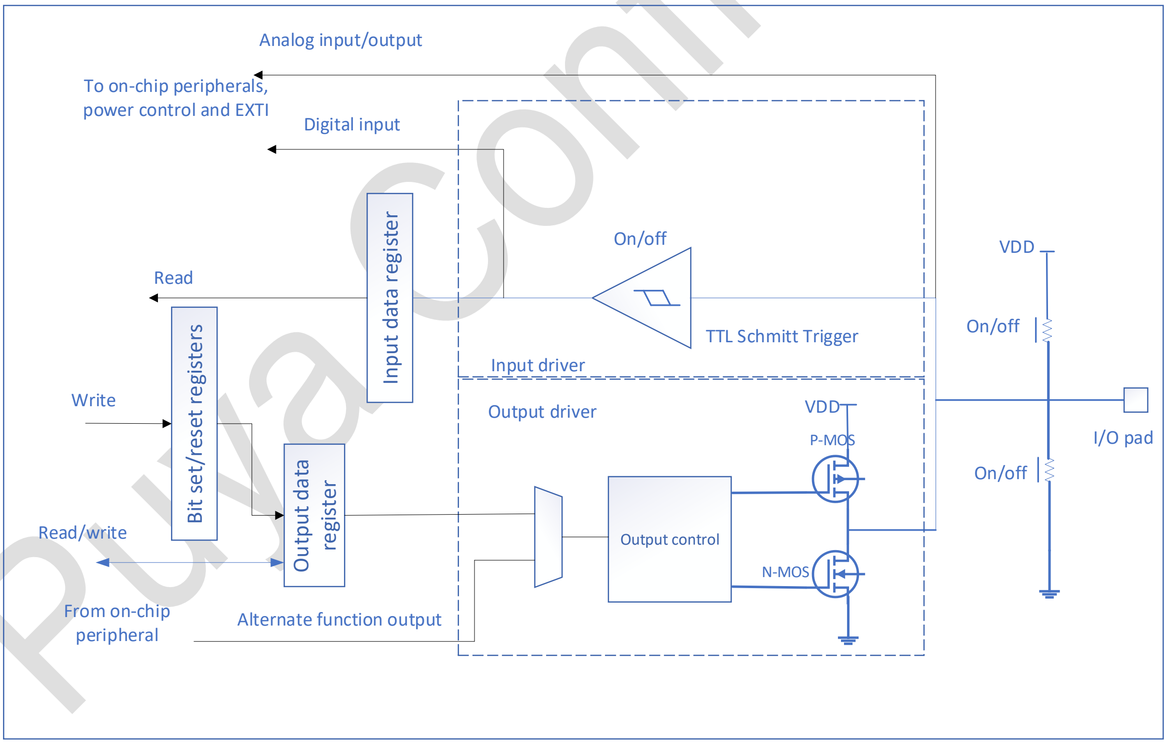


Figure 10-1 Basic structure of an I/O port bit

10.3.1. General-purpose I/O (GPIO)

During and after reset, the alternate functions are not active and most of the IOs are configured in analog mode.

The debug pins are in alternate function pull-up or pull-down after reset:

- PA14-SWCLK: in pull-down mode
- PA13-SWDIO: in pull-up mode

Boot pin is set to input pull-down mode after reset:

- PF8-Boot: in pull-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 the output drive in push-pull mode or open-drain mode (only the low level is driven, 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 or not depending on the value in the GPIOx_PUPDR register.

10.3.2. I/O pin alternate function multiplexer and mapping

The device I/O pins are connected to on-board peripherals/modules through multiplexers 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.

Each I/O port has a multiplexer with up to 16 alternate function inputs (AF0 to AF15), which can be configured through the registers GPIOx_AFRL (for pin 0 to 7) and GPIOx_AFRH (for pin 8 to 15).

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

- The alternate function assignments for each pin are details in section 3.1-3.3.

In addition to this flexible multiplexer architecture, each peripheral has alternate functions mapped onto different I/O pins to optimize the number of peripherals available in smaller packages.

The user configures IO as follows:

- Debug function: After each reset, these pins are assigned as alternate function pins immediately usable by the debugger host.
- GPIO: Configure the corresponding I/O port as output, input or analog mode in GPIOx_MODER register.

- Peripheral multiplexing function:
  - The I/O corresponding to the register GPIOx_AFRL or GPIOx_AFRH configuration is the alternate function x (x = 0… 15).
  - Registers GPIOx_OTYPER, GPIOx_PUPDR and GPIOX_OSPEEDER configure the type, pull-up/pull-down and output speed respectively.
  - Configure the corresponding I/O as an alternate function in the GPIOx_MODER register.
- Bus read/write function: The GPIO module not only supports reading and writing GPIOx registers through IO port, but also supports reading and writing registers through AHB bus. The register access mode is selected by the GPIO_AHB_SEL bit of SYSCFG module. When GPIO_AHB_SEL bit is 0, only GPIOx registers can be accessed through IO port; when GPIO_AHB_SEL is 1, only GPIOx registers can be accessed through AHB bus.
- The AHB bus accesses GPIO registers by supporting DMA in addition to CPU, i.e. DMA can access GPIO registers directly through the AHB bus.
- Additional functions:
  - ADC and COMP functions are enabled in the registers of the ADC and COMP modules, in every I/O configuration. When the I/O is used as ADC or COMP, it is recommended to configure the port as analog mode through the register GPIOx_MODER
  - For additional functions of the crystal oscillator, configure the respective functions in the corresponding PWR and RCC module registers. These configurations have higher priority than standard GPIO configurations.

10.3.3. I/O port control registers

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

10.3.4. 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/written accessible. The data input through the I/O are stored into the input data register (GPIOx_IDR), a read-only register.

10.3.5. I/O data bitwise handling

The bit set reset register (GPIOx_BSRR) is a 32-bit register that allows the application to set and reset each individual bits in 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 of GPIOx_BSRR: BS(i) and BR(i). When written bit BS(i) to 1 can set the corresponding bit of GPIOx_ODR to 1, and setting bit BR(i) to 1 can clear the corresponding bit of GPIOx_ODR to 0.

Write any bit to 0 in GPIOx_BSRR does not have any effect on the corresponding bit in GPIOx_ODR. If there is an attempt to both set and reset a bit in GPIOx_BSRR, the set operation has priority.

Using the GPIOx_BSRR register to change the values of individual bit 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.

Software does not need to disable interrupts when performing bit operations on GPIOx_ODR: one or more bits can be modified in a single atomic AHB1 write access.

Register GPIOx_LCKR can freeze the IO control registers through a series of special write timings, including GPIOx_MODER,GPIOx_OTYPER,GPIOx_OSPEEDR,GPIOx_PUPDR,GPIOx_AFRL and GPIOx_AFRH.

A special write/read sequence can manipulate the register GPIOx_LCKR. When the right lock sequence is applied to bit 16 in this register, the value of LCKR[15:0] can LOCK the I/O (during the write sequence, the value of LCKR[15:0] remains unchanged). When the LOCK sequence has been applied to a port bit, the value of the port bit cannot be modify until the next MCU reset or peripheral reset.

Each GPIOx_LCKR bit freezes the corresponding bit in the control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR, GPIOx_AFRL and GPIOx_AFRH).

The LOCK timing can only access the GPIOx_LCKR register with a word (32 bits long) because the GPIOx_LCKR bit 16 setting also sets the [15:0] bits.

10.3.6. I/O alternate function input/output

Two registers are provided to select one of the alternate function input/outputs available for each I/O. The user can connect an alternate function to the IO port according as required by the application. This means that a number of possible peripheral functions are multiplexed on each GPIO using the GPIOx_AFRL and GPIOx_AFRH alternate function registers. The application can thus select any one of the possible functions for each I/O. 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.

10.3.7. External interrupt/wakeup lines

All ports have external interrupt capability. To use the external interrupt lines, the given pin must be disabled in analog mode or as oscillator pin, so the input trigger is kepy enabled.

10.3.8. I/O input configuration

When the I/O port is configured as input:

- The output buffer is disabled
- The Schmitt trigger input is enable
- The pull-up and pull-down resistors can be enabled/disabled according to the configuration of the GPIOx_PUPDR register
- The data present on the I/O pins are sampled into the input data register on every AHB clock cycle
- A read access to the input data register provides the I/O status

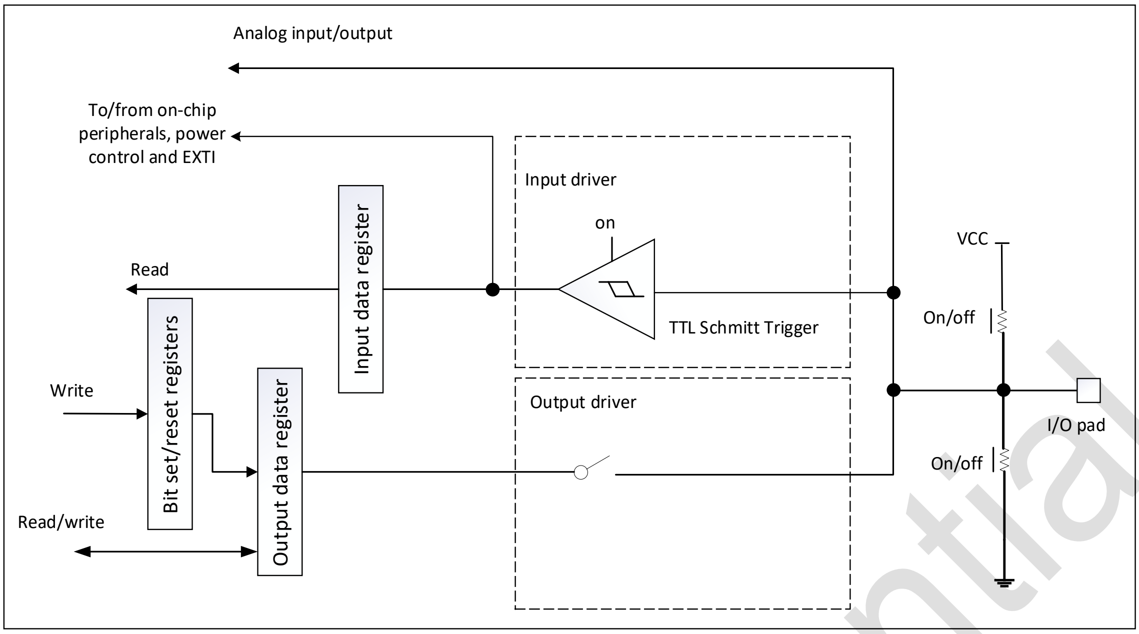


Figure 10-2 Input floating/pull up/pull down configurations

10.3.9. I/O output configuration

When the I/O port is configured as output:

- The output buffer is enabled:
  - Open-drain mode: A '0' in the output register activates the N-MOS whereas a '1' in the output register leaves the port in a high-impedance state (the 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 can be enabled/disabled according to the configuration of the GPIOx_PUPDR register
- The data present on the I/O pins are sampled into the input data register every AHB clock cycle
- A read access to the input data register gets the I/O state
- A read access to the output data register gets the value of the last write

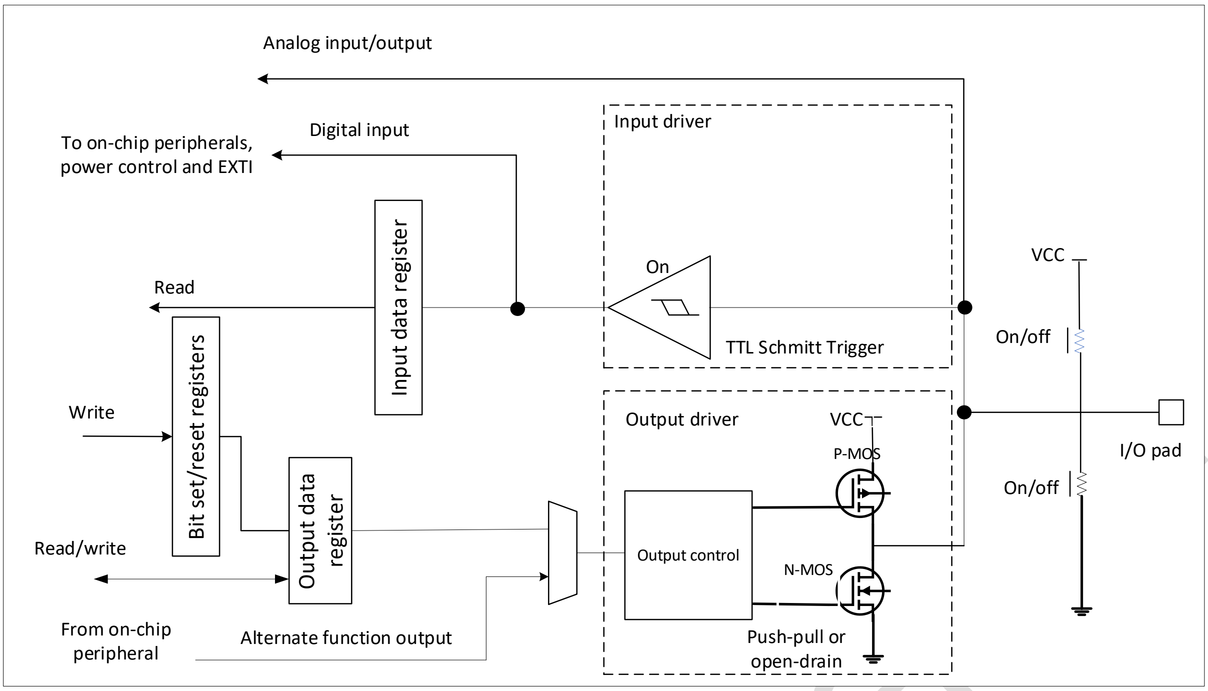


Figure 10-3 Output configuration

10.3.10. Alternate function configuration

When an I/O port is configured as alternate function:

- In an open-drain or push-pull configuration, the output buffer is turned on
- Built-in peripheral signal-driven output buffer (multiplexed function output)
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors can be enabled/disabled according to the configuration of the GPIOx_PUPDR register
- The data present on the I/O pins are sampled into the input data register every AHB clock cycle
- A read access to the input data register gets the I/O state

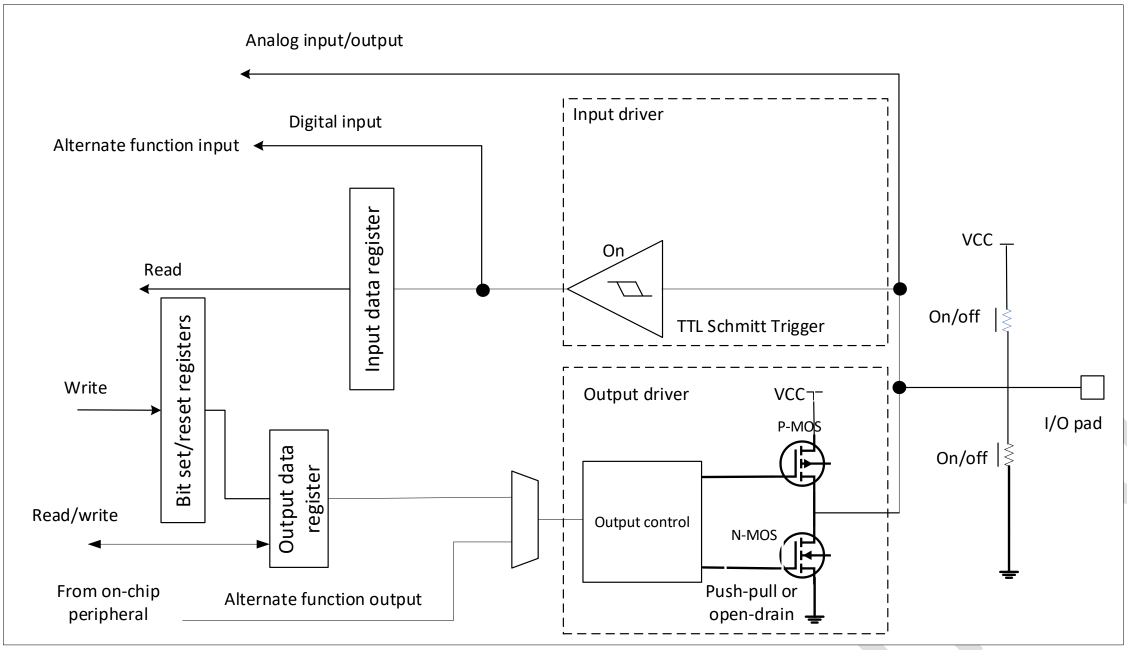


Figure 10-4 Alternate function configuration

10.3.11. Analog configuration

When an I/O port is configured 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 output of Schmitt trigger is forced to '0'
- The weak pull-up and pull-down resistors are disabled (software setting required)
- Read access to the input data register gets the value is '0'

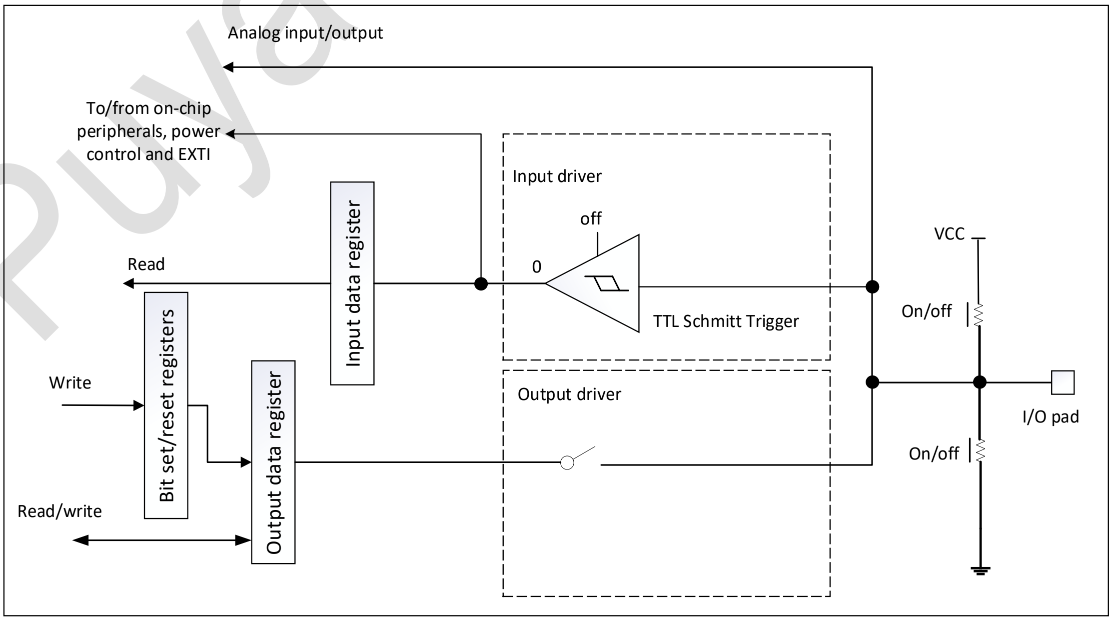


Figure 10-5 High impedance-analog configuration

10.3.12. Use the HSE/LSE oscillator pins as GPIOs

When the HSE or LSE oscillator is switch off (default state after reset), the related oscillator pins can be used as normal GPIOs.

When the HSE or LSE oscillator is switch on (by setting the HSEON or LSEON bit in the RCC_CSR register) the corresponding port needs to be configured as an analog port by software.

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

10.4. GPIO registers

The GPIO related registers can be written in word, half word and byte mode.

10.4.1. GPIO port mode register (GPIOx_MODER) (x = A, B, F)

Address offset: 0x00

Reset value:

0xEBFF FFFF for GPIOA

0xFFFF FFFF for GPIOB

0xFFFF FFFF for GPIOC

0x000C FFFF 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: Analog mode (reset state)

10.4.2. 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	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	MODE[15:0]	RW		These bits are written by software to configure the I/O output type 0: Output push-pull (reset state) 1: Output open-drain

10.4.3. 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 speed 01:Low speed 10:High speed 11:Very high speed

10.4.4. GPIO port pull-up and pull-down register (GPIOx_PUPDR) (x = A, B, F)

Address offset: 0x0C

Reset value:

- 0x2400 0000(for port A)
- 0x0000 0000(for port B,C)
- 0x0002 0000(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 or pull-down 01: Pull-up 10: Pull-down 11: Reserved

10.4.5. 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	-	-	-
15:0	Idy	R		y = 15..0 This is read-only, it contain the input value of the corre-sponds I/O port

10.4.6. 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 are readable and writable by software. Note: For GPIOx_BSRR or GPIOx_BRR registers. (x = A,B,F), each ODR bit can be independently set/cleared.

10.4.7. 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	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
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
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	BRy	W	-	y = 15..0 These bits are write-only. A read to these bits returns the value of 0. 0: No action on the corresponding ODRy bit 1: Clear the corresponding ODRy bit Note: If the corresponding bits of Bsy and Bry are set at the same time, the Bsy bit has priority.
15: 0	BSy	W	-	y = 15..0 These bits are write-only. A read to these bits returns the value of 0. 0: No action on the corresponding ODRy bit 1: Set the corresponding ODRy bit

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

This register is used to lock the configuration of the port bits when the correct write sequence is applied to bit 16 (LCKK) set. The value of bits [15:0] is used to lock the configuration of the GPIO, the value of

LCKR [15:0] must not change. When the LOCK sequence has been applied on the a port bit, the configuration of the port bits cannot be changed until the next system reset.

Note: A special write sequence is used to write the GPIOx_LCKR register. Only word accesses can be performed during the lock 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 by the lock key write sequence 0: The port configuration lock key not active 1: The port configuration lock key activated, and the GPIOx_LCKR register is locked until the next system reset LOCK key write sequence: The write sequence of the lock key: WR LCKR[16] = '1' + LCKR[15:0] WR LCKR[16] = '0' + LCKR[15:0] WR LCKR[16] = '1' + LCKR[15:0] RD LCKR[16] = '0' RD LCKR[16] = '1' (This read operation is optional, but it confirms that the lock is active) Note: During the LOCK key write sequence, the value of LCK[15:0] must not change. Any error in the lock sequence will stop the lock key from being activated. After the first lock sequence on any bit of the port, any read access on the LCKK will return 1 until the next MCU reset or peripheral reset.
15: 0	LCKy	RW		y = 15..0

Bit	Name	R/W	Reset Value	Function
				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

10.4.9. GPIO alternate function register (low) (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 alternate function I/O. AFSELy selection: 0000: AF0 1000: AF8 0001: AF1 1001: AF9 0010: AF2 1010: AF10 0011: AF3 1011: AF11 0100: AF4 1100: AF12 0101: AF5 1101: AF13 0110: AF6 1110: AF14 0111: AF7 1111: AF15

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

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	AFSELy[3:0] ((y= 8 to 15))	RW		These bits are written by software to configure alternate function I/O. AFSELy selection: 0000: AF0 1000: AF8 0001: AF1 1001: AF9 0010: AF2 1010: AF10 0011: AF3 1011: AF11 0100: AF4 1100: AF12 0101: AF5 1101: AF13 0110: AF6 1110: AF14 0111: AF7 1111: AF15

10.4.11. 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	-	-	-
15:0	Bry	W		y = 15..0 These bits are write-only. A read to these bots returns the value of 0. 0: No action on the corresponding Ody bit 1: Clear the corresponding Ody bit

10.4.12. GPIO register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	GPIOA_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[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]	
	Reset	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
	GPIOB_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[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]	
	Re-set value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	GPIOF_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[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]	
	Re-set value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x04	GPIOF_OTYPER (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
	Re-set value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	GPIOA_OSPEEDR	OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]		OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
	Re-set value	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x08	GPIOx_OSPEEDR (x=B, F)	OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]		OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOA_PUPDR	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[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]	
	Reset value	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOB_PUPDR	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[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]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOF_PUPDR	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[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]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x10	GPIOx_IDR (x=A,	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	B, F)																																
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x14	GPIOx_ODR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	GPIOx_BSRR (x=A, B, F)	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	GPIOx_LCKR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x20	GPIOx_AFRL (x=A, B, F)	AFSEL7 [3:0]				AFSEL6 [3:0]				AFSEL5 [3:0]				AFSEL4 [3:0]				AFSEL3 [3:0]				AFSEL2 [3:0]				AFSEL1 [3:0]				AFSEL0 [3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	GPIOx_AFRH (x=A, B, F)	AFSEL15 [3:0]				AFSEL14 [3:0]				AFSEL13 [3:0]				AFSEL12 [3:0]				AFSEL11[3:0]				AFSEL10 [3:0]				AFSEL9 [3:0]				AFSEL8 [3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	GPIOx_BRR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11. System configuration controller (SYSCFG)

The devices feature a set of configuration registers. The main purpose of the system configuration controller are:

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

11.1. System configuration register

11.1.1. SYSCFG configuration register 1 (SYSCFG_CFGR1)

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 by software, and bypass the hardware BOOT selection. After reset, these bits take the value configured by the actual boot mode.

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					ETR_SRC_TIM3[2:0]		
							RW						RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RES	ETR_SRC_TIM2[2:0]			RES	ETR_SRC_TIM1[2:0]			TIM3_IC1_SRC		TIM2_IC4_SRC		TIM1_IC1_SRC		MEM_MODE	
	RW				RW			RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31:25	RES	RES	-	RES
24	GPIO_AHB_SEL	RW	0	CPU FASTIO or AHB bus access to GPIO register control. 0: FASTIO bus access; 1: AHB bus access;
23:19	RES	RES	-	RES
18:16	ETR_SRC_TIM3[2:0]	RW	0	TIMER3 ETR input source selection. 3'b000: ETR sourced from GPIO; 3'b001: ETR originated from COMP1; 3'b010: ETR derived from COMP2; 3'b011: ETR derived from ADC; 3'b100: ETR from COMP3. others: reserved

Bit	Name	R/W	Reset Value	Function
15	RES	RES	-	RES
14:12	ETR_SRC_TIM2[2:0]	RW	0	TIMER2 ETR input source selection. 3'b000: ETR sourced from GPIO; 3'b001: ETR originated from COMP1; 3'b010: ETR derived from COMP2; 3'b011: ETR derived from ADC; 3'b100: ETR from COMP3. others: reserved
11	RES	RES	-	RES
10:8	ETR_SRC_TIM1[2:0]	RW	0	TIMER1 ETR input source selection. 3'b000: ETR sourced from GPIO; 3'b001: ETR originated from COMP1; 3'b010: ETR originated from COMP2; 3'b011: ETR derived from ADC; 3'b100: ETR from COMP3. others: reserved
7:6	TIM3_IC1_SRC	RW	0	TIM13 CH1 input source 00: from TIM3_CH1 IO; 01: from comp 1; 10: from comp 2; 11: from comp 3;
5:4	TIM2_IC4_SRC	RW	0	TIM2 CH4 input source 00: from TIM2_CH4 IO; 01: from comp 1; 10: from comp 2; 11: from comp 3;
3:2	TIM1_IC1_SRC	RW	0	TIM1 CH1 input source 00: from TIM1_CH1 IO; 01: from comp 1; 10: from comp 2; 11: from comp 3;
1:0	MEM_MODE	RW	x	Memory area mapping selection. x0: Main Flash memory mapped at 0x0000 0000 01: System Flash memory mapped at 0x0000 0000 11: Embedded SRAM mapped at 0x0000 0000

11.1.2. SYSCFG configuration register 2 (SYSCFG_CFGR2)

Address offset:0x04

Reset value： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES	RES	RES	RES	RES	RES	RES	RES	COMP3_Ocref_	COMP3_Ocref_	COMP3_Ocref_	COMP2_Ocref_	COMP2_Ocref_	COMP2_Ocref_	COMP1_Ocref_	COMP1_Ocref_

								CLR_TIM3	CLR_TIM2	CLR_TIM1	CLR_TIM3	CLR_TIM2	CLR_TIM1	CLR_TIM3	CLR_TIM2
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP1_Ocref_CLR_TIM1	COMP3BRK_TIM17	COMP2_BRK_TIM17	COMP1_BRK_TIM17	COMP3BRK_TIM16	COMP2_BRK_TIM16	COMP1_BRK_TIM16	COMP3BRK_TIM15	COMP2_BRK_TIM15	COMP1_BRK_TIM15	COMP3BRK_TIM1	COMP2_BRK_TIM1	COMP1_BRK_TIM1	PVD_LOCK	RES	LOCKUP_LOCK
RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:24	RES	RES	-	RES
23	COMP3_Ocref_CLR_TIM3	RW	0	1: CMP3 output is used as the TIM3 ocref_clr input; 0: CMP3 output is not used as TIM3 ocref_clr input
22	COMP3_Ocref_CLR_TIM2	RW	0	1: CMP3 output is used as the TIM2 ocref_clr input; 0: CMP3 output is not used as TIM2 ocref_clr input
21	COMP3_Ocref_CLR_TIM1	RW	0	1: CMP3 output is used as the TIM1 ocref_clr input; 0: CMP3 output is not used as TIM1 ocref_clr input
20	COMP2_Ocref_CLR_TIM3	RW	0	1: COMP2 output is used as TIM3 ocref_clr input; 0: COMP2 output is not used as TIM3 ocref_clr input
19	COMP2_Ocref_CLR_TIM2	RW	0	1: COMP2 output is used as TIM2 ocref_clr input; 0: COMP2 output is not used as TIM2 ocref_clr input
18	COMP2_Ocref_CLR_TIM1	RW	0	1: COMP2 output is used as the TIM1 ocref_clr input; 0: COMP2 output is not used as TIM1 ocref_clr input
17	COMP1_Ocref_CLR_TIM3	RW	0	1: COMP1 output is used as the TIM3 ocref_clr input; 0: COMP1 output is not used as TIM3 ocref_clr input
16	COMP1_Ocref_CLR_TIM2	RW	0	1: COMP1 output is used as the TIM2 ocref_clr input; 0: COMP1 output is not used as TIM2 ocref_clr input
15	COMP1_Ocref_CLR_TIM1	RW	0	1: COMP1 output is used as the TIM1 ocref_clr input; 0: COMP1 output is not used as TIM1 ocref_clr input
14	COMP3 BRK_TIM17	RW	0	COMP3 is enabled as TIM17 break input. 0: COMP3 output is not connected to the TIM17 break input; 1: COMP3 output is used as the TIM17 break input;
13	COMP2_ BRK_TIM17	RW	0	COMP2 is enabled as TIM17 break input. 0: COMP2 output is not connected to the TIM17 break input;

Bit	Name	R/W	Reset Value	Function
				1: COMP2 output is used as TIM17 break input;
12	COMP1_BRK_TIM17	RW	0	COMP1 is enabled as TIM1 break input. 0: COMP1 output is not connected to the TIM1 break input; 1: COMP1 output is used as TIM1 break input;
11	COMP3 BRK_TIM16	RW	0	COMP3 is enabled as TIM16 break input. 0: COMP3 output is not connected to the TIM16 break input; 1: COMP3 output is used as the TIM16 break input;
10	COMP2_ BRK_TIM16	RW	0	COMP2 is enabled as TIM16 break input. 0: COMP2 output is not connected to the TIM16 break input; 1: COMP2 output is used as TIM16 break input;
9	COMP1_BRK_TIM16	RW	0	COMP1 is enabled as TIM16 break input. 0: COMP1 output is not connected to the TIM16 break input; 1: COMP1 output is used as TIM16 break input;
8	COMP3 BRK_TIM15	RW	0	COMP3 is enabled as TIM15 break input. 0: COMP3 output is not connected to the TIM15 break input; 1: COMP3 output is used as the TIM15 break input;
7	COMP2_ BRK_TIM15	RW	0	COMP2 is enabled as TIM15 break input. 0: COMP2 output is not connected to the TIM15 break input; 1: COMP2 output is used as TIM15 break input;
6	COMP1_BRK_TIM15	RW	0	COMP1 is enabled as TIM15 break input. 0: COMP1 output is not connected to the TIM15 break input; 1: COMP1 output is used as TIM15 break input;
5	COMP3 BRK_TIM1	RW	0	COMP3 is enabled as TIM1 break input. 0: COMP3 output is not connected to the TIM1 break input; 1: COMP3 output is used as the TIM1 break input;
4	COMP2_ BRK_TIM1	RW	0	COMP2 is enabled as TIM1 break input. 0: COMP2 output is not connected to the TIM1 break input; 1: COMP2 output is used as TIM1 break input;
3	COMP1_BRK_TIM1	RW	0	COMP1 is enabled as TIM1 break input. 0: COMP1 output is not connected to the TIM1 break input; 1: COMP1 output is used as TIM1 break input;
2	PVD_LOCK	RW	0	PVD lock enable bit. 0: PVD interrupt is not connected to TIM1/15/16/17 break input, PVD function is configured by PVDE and PLS; 1: PVD interrupt is connected to TIM1/15/16/17 break input, PVDE and PLS are read only;
1	RES	RES	-	RES
0	LOCKUP_LOCK	RW	0	Cortex-M0+ LOCK"U"P bit lock enable bit. 0: Cortex-M0+ LOCKUP bit is not connected to TIM1/15/16/17 break input; 1: Cortex-M0+ LOCKUP bit is connected to TIM1/15/16/17 break input;

11.1.3. SYSCFG configuration register 3 (SYSCFG_CFGR3)

Address offset:0x08

Reset value:0x3F3F 3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES		DMA4_MAP						RES		DMA3_MAP					
		RW								RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RES		DMA2_MAP						RES		DMA1_MAP					
		RW								RW					

Bit	Name	R/W	Reset Value	Function
31:30	RES	RES	-	RES
29:24	DMA4_MAP	RW	0x3F	DMA1 channel 4 mapping. See DMA1_MAP description.
23:22	RES	RES	-	RES
21:16	DMA3_MAP	RW	0x3F	DMA1 channel 3 mapping. See DMA1_MAP description.
15:14	RES	RES	-	RES
13:8	DMA2_MAP	RW	0x3F	DMA1 channel 2 mapping. See DMA1_MAP description.
7:6	RES	RES	-	RES
5:0	DMA1_MAP	RW	0x3F	DMA1 channel 1 mapping. 000000: ADC1; 000001: DAC1; 000010: DAC2; 000011: SPI1_RD; 000100: SPI1_WR; 000101: SPI2_RD; 000110: SPI2_WR; 000111: USART1_RD; 001000: USART1_WR; 001001: USART2_RD; 001010: USART2_WR; 001011: USART3_RD; 001100: USART3_WR; 001101: USART4_RD; 001110: USART4_WR; 001111: I2C1_RD; 010000: I2C1_WR; 010001: I2C2_RD; 010010: I2C2_WR; 010011: TIM1_CH1; 010100: TIM1_CH2; 010101: TIM1_CH3; 010110: TIM1_CH4;

Bit	Name	R/W	Reset Value	Function
				010111: TIM1_COM; 011000: TIM1_TRG; 011001: TIM1_UP; 011010: TIM2_CH1; 011011: TIM2_CH2; 011100: TIM2_CH3; 011101: TIM2_CH4; 011110: TIM2_UP; 011111: TIM2_TRG; 100000: TIM3_CH1; 100001: TIM3_CH2; 100010: TIM3_CH3; 100011: TIM3_CH4; 100100: TIM3_UP; 100101: TIM3_TRG; 100110: TIM6_UP; 100111: TIM7_UP; 101000: TIM15_CH1; 101001: TIM15_CH2; 101010: TIM15_UP; 101011: TIM15_TRG; 101100: TIM15_COM, 101101: TIM16_CH1; 101110: TIM16_UP; 101111: TIM17_CH1; 110000: TIM17_UP; 110001: USB; 110010: LCD; 1Others: Reserved

11.1.4. SYSCFG configuration register 4 (SYSCFG_CFGR4)

Address offset:0x0C

Reset value:0x003F 3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES										DMA7_MAP					
										RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RES		DMA6_MAP						RES		DMA5_MAP					
		RW								RW					

Bit	Name	R/W	Reset Value	Function
31:22	RES	RES	-	RES
21:16	DMA7_MAP	RW	0x3F	DMA1 channel 7 mapping.

Bit	Name	R/W	Reset Value	Function
				See DMA1_MAP description.
15:14	RES	RES	-	RES
13:8	DMA6_MAP	RW	0x3F	DMA1 channel 6 mapping. See DMA1_MAP description.
7:6	RES	RES	-	RES
5:0	DMA5_MAP	RW	0x3F	DMA1 channel 5 mapping. See DMA1_MAP description.

11.1.5. GPIOA filter enable（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															
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	RES	-	RES
15:0	PA_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

11.1.6. GPIOB filter enable（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															
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	RES	-	RES
15:0	PB_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

11.1.7. GPIOC filter enable（PC_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															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:0	PC_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

11.1.8. GPIOF filter enable（PF_ENS）

Address offset:0x1C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	RES	-	RES
9:0	PF_ENS[x]	RW	0x000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

11.1.9. I2C type IO configuration register (SYSCFG_EIIC)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						PF_EIIC[1:0]		RES					PB_EIIC[10:0]		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								RES						PA_EIIC[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
31:26	Reserved	-	-	-
25:24	PF_EIIC[x]	-	-	PF I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled
23:19	Reserved	-	-	-
18:8	PB_EIIC[x]	-	-	PB I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled
7:2	Reserved	-	-	-
1:0	PA_EIIC[x]	RW	0	PA I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

11.1.10. SYSCFG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SYSCFG_CFGR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIO AHB SEL	I2C PA10 ANF	Res.	Res.	Res.	GPIO AHB SEL	ETR_SRC_TIM3[2:0]			Res.	ETR_SRC_TIM2[2:0]			Res.	ETR_SRC_TIM1[2:0]			TIM3_IC1_SRC		TIM2_IC4_SRC		TIM1_IC1_SRC		MEM_MODE[1:0]	
	Reset value								0	0				0	0	0	0		0	0	0		0	0	0	0	0	0	0	0	0	X	X
0x04	SYSCFG_CFGR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	COMP3 Ocref CLR TIM3	COMP3 Ocref CLR TIM2	COMP3 Ocref CLR TIM1	COMP2 Ocref CLR TIM3	COMP2 Ocref CLR TIM2	COMP2 Ocref CLR TIM1	COMP1 Ocref CLR TIM3	COMP1 Ocref CLR TIM2	COMP1 Ocref CLR TIM1	COMP3 BRK TIM17	COMP2 BRK TIM17	COMP1 BRK TIM17	COMP3 BRK TIM16	COMP2 BRK TIM16	COMP1 BRK TIM16	COMP3 BRK TIM15	COMP2 BRK TIM15	COMP1 BRK TIM15	COMP1 BRK TIM16	COMP2 BRK TIM1	COMP1 BRK TIM1	PVD LOCK	Res.	LOCKUP LOCK
	Reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0
0x08	SYSCFG_CFGR3	Res.	Res.	DMA4_MAP						Res.	Res.	DMA3_MAP						Res.	Res.	DMA2_MAP						Res.	Res.	DMA1_MAP					
	Reset value											0x3F								0x3F								0x3F					
0x0C	SYSCFG_CFGR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA7_MAP						Res.	Res.	DMA6_MAP						Res.	Res.	DMA5_MAP					
	Reset value											0x3F								0x3F								0x3F					

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x10	PA_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PA_ENS[x]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	PB_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PB_ENS[x]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	PC_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PC_ENS[x]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	PF_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF_ENS[x]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	SYSCFG_EIIC	Res.	Res.	Res.	Res.	Res.	Res.	PF_IECC[x]							PB_IECC[x]																	PA_IECC[x]	
	Reset value														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

12. DMA

12.1. DMA introduction

Direct Memory Access (DMA) is used to provide high-speed data transfer between peripherals and memory or between memory and memory. The DMA controller has seven channels, each dedicated to managing requests for memory access from one or more peripherals. There is also an arbiter to coordinate the priority of individual DMA requests.

12.2. DMA main features

- Single AHB master
- Supports peripheral-to-memory, memory-to-peripheral, memory-to-memory and peripheral-to-peripheral data transfers
- On-chip memory devices, such as FLASH, SRAM, AHB and APB peripherals, as source and destination
- All DMA channels are independently configurable:
  - Each channel is either associated with a DMA request signal from a peripheral or with a software trigger in a memory-to-memory transfer. This configuration is done by software.
  - The priority between requests is programmable by software (4 levels per channel: very high, high, medium, low) and in case of equality by hardware (e.g. requests to channel 1 have priority over requests to channel 2).
  - The source and destination transfer sizes are independent (byte, halfword, word), simulating packetization and unpacketization. Source and destination addresses must be aligned by data size.
  - Programmable number of data to be transmitted: 0 ~ 65535
- One interrupt request is generated per channel. Each interrupt request is caused by any one of three DMA events: transfer completion, half-transfer, or transfer error.

12.3. DMA functional description

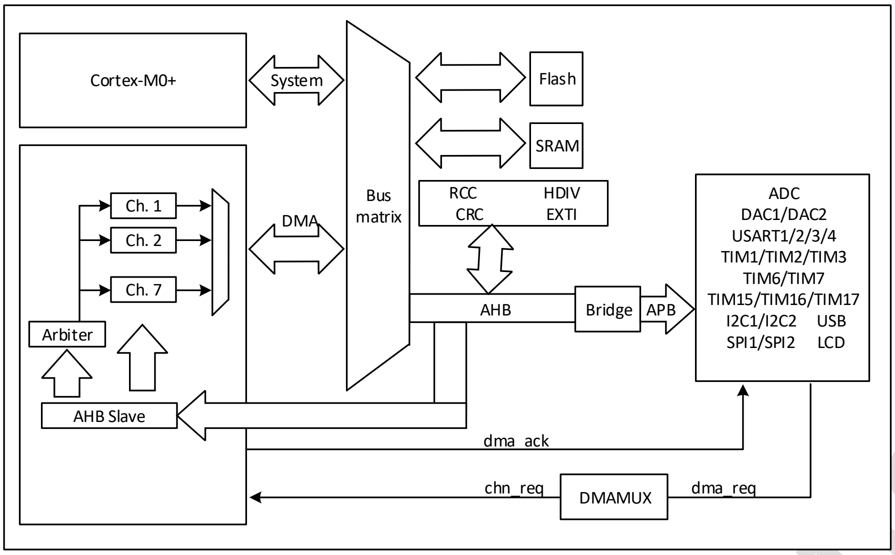


Figure 12-1 DMA block diagram

12.3.1. DMA transactions

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 accessing the peripheral that sent the request, the DMA controller sends it an answer signal at the end of the transfer. When the answer signal is received from the DMA controller, the peripheral immediately releases its request. Once the peripheral has released the request, the DMA controller revokes the answer signal. If there are more requests, the peripheral can initiate the next transfer.

In summary, each DMA transfer consists of three operations:

- The loading of data from the peripheral data register or a location in memory addressed through an internal current peripheral/memory address register. The start address used for the first transfer is the base peripheral/memory address programmed in the DMA_CPARx or DMA_CMARx register
- The storage of the data loaded to the peripheral data register or a location in memory addressed through an internal current peripheral/memory address register. The start address used for the first transfer is the base peripheral/memory address programmed in the DMA_CPARx or DMA_CMARx register.
- Performs a decrement operation of the DMA_CNDTRx register, which indicates the number of outstanding operations.

12.3.2. Arbiter

The arbiter manages the channel requests based on their priority and launches the peripheral/memory access sequences.

The priorities are managed in two stages:

- Software: each channel priority can be configured in the DMA_CCRx register. There are four levels:
  - Very high priority
  - High priority
  - Medium priority
  - Low priority
- Hardware: if 2 requests have the same software priority level, the channel with the lowest number will get priority versus the channel with the highest number. For example, channel 2 gets priority over channel 4.

12.3.3. DMA channels

Each channel can perform a DMA transfer between a peripheral register with a fixed address and a memory address. the amount of data transferred by DMA is programmable up to 65535 bytes, and this register value is decremented after each data transfer.

Transfer data amount programmable

Transfer data sizes of the peripheral and memory are fully programmable through the PSIZE and MSIZE bits in the DMA_CCRx register.

Address Pointer Increment

Peripheral and memory pointers can optionally be automatically post-incremented after each transaction depending on the PINC and MINC bits in the DMA_CCRx register.

If incremented mode is enabled, the address of the next transfer will be the address of the previous one incremented by 1, 2 or 4 depending on the chosen data size.The first transfer address is the one programmed in the DMA_CPARx/DMA_CMARx registers. During transfer operations, these registers keep the initially programmed value. The current transfer addresses (in the current internal peripheral/memory address register) are not accessible by software.

If the channel is configured in noncircular mode, no DMA request is served after the last transfer (that is once the number of data items to be transferred has reached zero). In order to reload a new number of data items to be transferred into the DMA_CNDTRx register, the DMA channel must be disabled.

In circular mode, after the last transfer, the DMA_CNDTRx register is automatically reloaded with the initially programmed value. The current internal address registers are reloaded with the base address values from the DMA_CPARx/DMA_CMARx registers.

Channel configuration procedure

The following sequence should be followed to configure a DMA channelx (where x is the channel number).

- Set the peripheral register address in the DMA_CPARx register. The data will be moved from/ to this address to/ from the memory after the peripheral event.
- Set the memory address in the DMA_CMARx register. The data will be written to or read from this memory after the peripheral event.
- Configure the total number of data to be transferred in the DMA_CNDTRx register. After each peripheral event, this value will be decremented.
- Configure the DMA_CCRx register with the following parameters:
  - The priority of the channel.
  - Data transfer direction
  - Cyclic mode
  - Peripheral and memory incremental mode
  - Peripheral and memory data size
  - Interrupt enable
- Activate the channel by setting the ENABLE bit in the DMA_CCRx register.

As soon as the channel is enabled, it can serve any DMA request from the peripheral connected on the channel.

Once half of the bytes are transferred, the half-transfer flag (HTIF) is set and an interrupt is generated if the Half-Transfer Interrupt Enable bit (HTIE) is set. At the end of the transfer, the Transfer Complete Flag (TCIF) is set and an interrupt is generated if the Transfer Complete Interrupt Enable bit (TCIE) is set.

Channel Status and Disabled Channels

An active channel x is a channel that is enabled (read DMA_CCRx.EN = 1). An active channel x is a channel that must have been enabled by software (DMA_CCRx. EN = 1) and then no transmission error has occurred (DMA_ISR.TEIFx = 0). If a transmission error occurs, the channel is automatically disabled by hardware (DMA_CCRx.EN = 0).

The following 3 scenarios may occur:

- Pause and resume the channel

  This corresponds to the following two actions .

  - The active channel is disabled by the software (write DMA_CCRx.EN = 0).
  - The software re-enables the channel (DMA_CCRx.EN = 1), but does not reconfigure the other channel registers (e.g. DMA_CNDTRx, DMA_CPARx, and DMA_CMARx); or an in-complete transfer hangs the bus while the software disables it.
  - The DMA hardware does not support this case and therefore cannot guarantee correct exe-cution of the remaining data transfers.

- Stopping and suspending a channel

  The active channel can be disabled by software if the application no longer needs the channel. The channel is stopped and aborted, but the DMA_CNDTRx register contents may not correctly reflect the remaining data transfers.

  - Disable and restart the channel

This corresponds to the software sequence:Disable the active channel, then reconfigure the channel and enable it again.

Hardware support when the following conditions are met.

- The application guarantees that there are no transfers in progress (not yet completed) on the DMA when the channel is disabled by software. For example, an application can first disable a peripheral in DMA mode to ensure that there are no pending hardware DMA requests for that peripheral.

- Software must have independent write access to the same DMA_CCRx register: first disable the channel, second reconfigure the channel for the next block transfer, including DMA_CCRx, if a configuration change is needed. finally enable the channel again.

The hardware clears the EN bit of the DMA_CCRx register when a channel transfer error occurs. This EN bit cannot be set again by software to reactivate channel x until the TEIFx bit of the DMA_ISR register is set.

DMA circular mode

Circular mode is available to handle circular buffers and continuous data flows (e.g. ADC scan mode). This feature can be enabled using the CIRC bit in the DMA_CCRx register.When circular mode is activated, the number of data to be transferred is automatically reloaded with the initial value programmed during the channel configuration phase, and the DMA requests continue to be served.

Note: Cyclic mode cannot be used in memory-to-memory mode. The MEM2MEM bit of the DMA_CCRx register must be cleared by software before cyclic mode (CIRC = 1) can enable the channel. When cyclic mode is enabled, the amount of data to be transferred will be automatically re-loaded during the channel configuration phase using the programmed initial value and DMA requests will continue to be responded to.

In order to stop cyclic transfers, software needs to stop the peripheral from generating DMA requests before disabling the DMA channel (e.g. exiting ADC scan mode).

Before starting/enabling a transfer, and after stopping a cyclic transfer, software must explicitly program the DMA_CNDTRx value.

Memory-to-memory mode

The DMA channels can also work without being triggered by a request from a peripheral.This mode is called Memory to Memory mode.

If the MEM2MEM bit in the DMA_CCRx register is set, then the channel initiates transfers as soon as it is enabled by software by setting the Enable bit (EN) in the DMA_CCRx register. The transfer stops once the DMA_CNDTRx register reaches zero.

Memory to Memory mode may not be used at the same time as Circular mode.

Peripheral to Peripheral Mode

Any DMA channel can be operated in Peripheral to Peripheral mode:

- When a hardware request from a peripheral is selected to trigger a DMA channel

  This peripheral is the DMA initiator, and data is transferred between this peripheral and registers belonging to another memory-mapped peripheral (which is not configured for DMA mode).

- When no peripheral request is selected and connected to the DMA channel

The software configures the register-to-register transfer by setting the MEM2MEM bit of the DMA_CCRx register.

Configuring the transfer direction, specifying the source/destination

The value of the DIR bit of the DMA_CCRx register sets the direction of the transfer, thus it identifies the source and the target, regardless of the source/target type (peripheral or memory):

- DIR = 1 usually defines a memory to peripheral transfer. More generally, if DIR = 1:
  - The source attribute is defined by the DMA_MARx register, the MSIZE[1:0] field, and the MINC bits of the DMA_CCRx register.
  - Regardless of their usual naming, these "peripheral" registers, fields, and bits are used to define the target memory in memory-to-memory mode.
  - The target attributes are defined by the DMA_PARx register, the PSIZE[1:0] field of the DMA_CCRx register, and the PINC bits.
  - Regardless of their usual naming, these "peripheral" registers, fields, and bits are used to define the target memory in memory-to-memory mode.
- DIR = 0 usually defines a peripheral-to-memory transfer. More generally, if DIR = 0.
  - The source attributes are defined by the DMA_PARx register, the PSIZE[1:0] field of the DMA_CCRx register, and the PINC bits.

    Regardless of their usual naming, these "peripheral" registers, fields, and bits are used to define the source memory in memory-to-memory mode.
  - The target attribute is defined by the DMA_MARx register, the MSIZE[1:0] field of the DMA_CCRx register, and the MINC bits.

Regardless of their usual naming, these "memory" registers, fields and bits are used to define the target peripheral in peripheral-to-peripheral mode.

12.3.4. Data transfer width/alignment/size end

When PSIZE and MSIZE are not equal, the DMA performs some data alignments as described in Table 12-1.

Table 12-1 Programmable data width and endian behavior (when bits PINC = MINC = 1)

Source port width	Destination port width	Number of data items to transfer (NDT)	Source content: address / data	Transfer operations	Destination content: address / data
8	8	4	0x0/B0	1: READ B0[7:0] @0x0 then WRITE B0[7:0] @0x0	0x0/B0
			0x1/B1	2: READ B1[7:0] @0x1 then WRITE B0[7:0] @0x1	0x1/B1
			0x2/B2	3: READ B2[7:0] @0x2 then WRITE B2[7:0] @0x2	0x2/B2
			0x3/B3	4: READ B3[7:0] @0x3 then WRITE B3[7:0] @0x3	0x3/B3
8	16	4	0x0/B0	1: READ B0[7:0] @0x0 then WRITE 00B0[15:0] @0x0	0x0/00B0
			0x1/B1	2: READ B1[7:0] @0x1 then WRITE 00B1[15:0] @0x2	0x2/00B1
			0x2/B2	3: READ B3[7:0] @0x2 then WRITE 00B2[15:0] @0x4	0x4/00B2
			0x3/B3	4: READ B4[7:0] @0x3 then WRITE 00B3[15:0] @0x6	0x6/00B3
8	32	4	0x0/B0	1: READ B0[7:0] @0x0 then WRITE 000000B0[31:0] @0x0	0x0/000000B0

Source port width	Destination port width	Number of data items to transfer (NDT)	Source content: address / data	Transfer operations	Destination content: address / data
			0x1/B1	2: READ B1[7:0] @0x1 then WRITE 000000B1[31:0] @0x4	0x4/000000B1
			0x2/B2	3: READ B3[7:0] @0x2 then WRITE 000000B2[31:0] @0x8	0x8/000000B2
			0x3/B3	4: READ B4[7:0] @0x3 then WRITE 000000B3[31:0] @0xC	0xC/000000B3
16	8	4	0x0/B1B0	1: READ B1B0[15:0] @0x0 then WRITE B0[7:0] @0x0	0x0/B0
			0x2/B3B2	2: READ B3B2[15:0] @0x2 then WRITE B2[7:0] @0x1	0x1/B2
			0x4/B5B4	3: READ B5B4[15:0] @0x4 then WRITE B4[7:0] @0x2	0x2/B4
			0x6/B7B6	4: READ B7B6[15:0] @0x6 then WRITE B6[7:0] @0x3	0x3/B6
16	16	4	0x0/B1B0	1: READ B1B0[15:0] @0x0 then WRITE B1B0[15:0] @0x0	0x0/B1B0
			0x2/B3B2	2: READ B3B2[15:0] @0x2 then WRITE B3B2[15:0] @0x2	0x2/B3B2
			0x4/B5B4	3: READ B5B4[15:0] @0x4 then WRITE B5B4[15:0] @0x4	0x4/B5B4
			0x6/B7B6	4: READ B7B6[15:0] @0x6 then WRITE B7B6[15:0] @0x6	0x6/B7B6

Source port width	Destination port width	Number of data items to transfer (NDT)	Source content: address / data	Transfer operations	Destination content: address / data
16	32	4	0x0/B1B0	1: READ B1B0[15:0] @0x0 then WRITE 0000B1B0[31:0] @0x0	0x0/0000B1B0
			0x2/B3B2	2: READ B3B2[15:0] @0x2 then WRITE 0000B3B2[31:0] @0x4	0x4/000B3B2
			0x4/B5B4	3: READ B5B4[15:0] @0x4 then WRITE 0000B5B4[31:0] @0x8	0x8/0000B5B4
			0x6/B7B6	4: READ B7B6[15:0] @0x6 then WRITE 0000B7B6[31:0] @0xC	0xC/0000B7B6
32	8	4	0x0/B3B2B1B0	1: READ B3B2B1B0[31:0] @0x0 then WRITE B0[7:0] @0x0	0x0/B0
			0x4/B7B6B5B4	2: READ B7B6B5B4[31:0] @0x4 then WRITE B4[7:0] @0x1	0x1/B4
			0x8/BBBAB9B8	3: READ BBBAB9B8[31:0] @0x8 then WRITE B8[7:0] @0x2	0x2/B8
			0xC/BFBEBDBC	4: READ BFBEBDBC[31:0] @0xC then WRITE BC[7:0] @0x3	0x3/BC
32	16	4	0x0/B3B2B1B0	1: READ B3B2B1B0[31:0] @0x0 then WRITE B1B0[7:0] @0x0	0x0/B1B0
			0x4/B7B6B5B4	2: READ B7B6B5B4[31:0] @0x4 then WRITE B5B4[7:0] @0x1	0x2/B5B4
			0x8/BBBAB9B8	3: READ BBBAB9B8[31:0] @0x8 then WRITE B9B8[7:0] @0x2	0x4/B9B8

Source port width	Destination port width	Number of data items to transfer (NDT)	Source content: address / data	Transfer operations	Destination content: address / data
			0xC/BFBEBDBC	4: READ BFBEBDBC[31:0] @0xC then WRITE BDBC[7:0] @0x3	0x6/BDBC
32	32	4	0x0/B3B2B1B0	1: READ B3B2B1B0[31:0] @0x0 then WRITE B3B2B1B0[31:0] @0x0	0x0/B3B2B1B0
			0x4/B7B6B5B4	2: READ B7B6B5B4[31:0] @0x4 then WRITE B7B6B5B4[31:0] @0x4	0x4/B7B6B5B4
			0x8/BBBAB9B8	3: READ BBBAB9B8[31:0] @0x8 then WRITE BBBAB9B8[31:0] @0x8	0x8/BBBAB9B8
			0xC/BFBEBDBC	4: READ BFBEBDBC[31:0] @0xC then WRITE BFBEBDBC[31:0] @0xC	0xC/BFBEBDBC

Addressing an AHB peripheral that does not support byte or halfword write operations

When the DMA initiates an AHB byte or halfword write operation, the data are duplicated on the unused lanes of the HWDATA[31:0] bus. So when the used AHB slave peripheral does not support byte or halfword write operations (when HSIZE is not used by the peripheral) and does not generate any error, the DMA writes the 32 HWDATA bits as shown in the two examples below:

- To write the halfword “0xABCD”, the DMA sets the HWDATA bus to “0xABCDABCD” with HSIZE = HalfWord
- To write the byte “0xAB”, the DMA sets the HWDATA bus to “0xABABABAB” with HSIZE = Byte

Assuming that the AHB/APB bridge is an AHB 32-bit slave peripheral that does not take the HSIZE data into account, it will transform any AHB byte or halfword operation into a 32-bit APB operation in the following manner:

- An AHB byte write operation of the data “0xB0” to 0x0 (or to 0x1, 0x2 or 0x3) will be converted to an APB word write operation of the data “0xB0B0B0B0” to 0x0

- An AHB halfword write operation of the data “0xB1B0” to 0x0 (or to 0x2) will be converted to an APB word write operation of the data “0xB1B0B1B0” to 0x0.

12.3.5. Error management

A DMA transfer error can be generated by reading from or writing to a reserved address space.

When a DMA transfer error occurs during a DMA read or a write access, the faulty channel is automatically disabled through a hardware clear of its EN bit in the corresponding Channel configuration register (DMA_CCRx). The channel's transfer error interrupt flag (TEIF) in the DMA_IFR register is set and an interrupt is generated if the transfer error interrupt enable bit (TEIE) in the DMA_CCRx register is set.

The EN bit of the DMA_CCRx register cannot be set again by software (channel x reactivated) until the TEIFx bit of the DMA_ISR register is cleared (by setting the CTEIFx bit of the DMA_IFCR register).

When software receives a notification of a transfer error via a channel involving a peripheral, software first stops this peripheral in DMA mode in order to disable any pending or future DMA requests. Software then typically reconfigures the DMA and peripheral in DMA mode for a new transfer.

12.3.6. DMA Interrupts

An interrupt can be produced on a Half-transfer, Transfer complete or Transfer error for each DMA channel. Separate interrupt enable bits are available for flexibility.

Table 12-2 DMA interrupt requests

Interrput event	Event flag	Enable Control bit
Half-transfer	HTIFx	HTIEx
Transfer complete	TCIFx	TCIEx
Transfer error	TEIFx	TEIEx
Transfer halfway/transfer complete/transfer error	GIFx	-

1. 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 complete.

2. Both the HTIF and TCIF flags are generated when the transmission length NDT is odd (greater than 1). The internal signal TCIF will be generated when NDT = 1; HTIF will be generated when (NDT - (NDT/2 (rounded) - 1)). If NDT=5, TCIF will be generated when NDT decreases to 1; HTIF will be generated when NDT decreases to 4. 3.

3. When the transmission length NDT is even (greater than 1), both HTIF and TCIF flags are generated. The internal signal TCIF will be generated when NDT = 1; HTIF will be generated when (NDT - (NDT/2 (rounded) - 1)). If NDT=10, TCIF is generated when NDT is reduced to 1; HTIF is generated when NDT is reduced to 6.

12.3.7. DMA peripheral request mapping

The mapping of DMA peripheral requests to individual channels of the DMA is controlled by the DMAx_MAP register bits in two SYSCFG registers (SYSCFG_CFGR3 and SYSCFGR_CFGR4), and each peripheral request can be mapped to any of the seven channels through configuration.

Table 12-3 DMA requests per channel

Request MUX input serial number	Source	Request MUX input serial number	Source	Request MUX input serial number	Source
0	ADC1	17	I2C2_RD	34	TIM3_CH3
1	DAC1	18	I2C2_WR	35	TIM3_CH4
2	DAC2	19	TIM1_CH1	36	TIM3_UP
3	SPI1_RD	20	TIM1_CH2	37	TIM3_TRIG
4	SPI1_WR	21	TIM1_CH3	38	TIM6_UP
5	SPI2_RD	22	TIM1_CH4	39	TIM7_UP
6	SPI2_WR	23	TIM1_COM	40	TIM15_CH1
7	USART1_RD	24	TIM1_TRIG	41	TIM15_CH2
8	USART1_WR	25	TIM1_UP	42	TIM15_UP
9	USART2_RD	26	TIM2_CH1	43	TIM15_TRIG
10	USART2_WR	27	TIM2_CH2	44	TIM15_COM
11	USART3_RD	28	TIM2_CH3	45	TIM16_CH1
12	USART3_WR	29	TIM2_CH4	46	TIM16_UP
13	USART4_RD	30	TIM2_UP	47	TIM17_CH1
14	USART4_WR	31	TIM2_TRG	48	TIM17_UP
15	I2C1_RD	32	TIM3_CH1	49	USB
16	I2C1_WR	33	TIM3_CH2	50	LCD

12.4. DMA registers

12.4.1. 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.	TEIF 7	HTIF 7	TCIF 7	GIF 7	TEIF 6	HTIF 6	TCIF 6	GIF 6	TEIF 5	HTIF 5	TCIF 5	GIF 5
				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**
TEIF 4	HTIF 4	TCIF 4	GIF 4	TEIF 3	HTIF 3	TCIF 3	GIF 3	TEIF 2	HTIF 2	TCIF 2	GIF 2	TEIF 1	HTIF 1	TCIF 1	GIF 1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	TEIF7	R	0	Channel 7 transmit error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 7 transmission error (TE);
26	HTIF7	R	0	Channel 7 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 7;
25	TCIF7	R	0	Channel 7 transmission completion flag. 0: no transmission completion (TC); 1: Channel 7 transmission complete (TC);
24	GIF7	R	0	Channel 7 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 7;
23	TEIF6	R	0	Channel 6 transmit error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 6 transmission error (TE);
22	HTIF6	R	0	Channel 6 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 6;
21	TCIF6	R	0	Channel 6 transmission completion flag. 0: no transmission completion (TC); 1: Channel 6 transmission complete (TC);

Bit	Name	R/W	Reset Value	Function
20	GIF6	R	0	Channel 6 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 6;
19	TEIF5	R	0	Channel 5 transmit error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 5 transmission error (TE);
18	HTIF5	R	0	Channel 5 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 5;
17	TCIF5	R	0	Channel 5 transmission completion flag. 0: no transmission completion (TC); 1: Channel 5 transmission complete (TC);
16	GIF5	R	0	Channel 5 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 5;
15	TEIF4	R	0	Channel 4 transmission error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 4 transmission error (TE);
14	HTIF4	R	0	Channel 4 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 4;
13	TCIF4	R	0	Channel 4 transmission completion flag. 0: no transmission completion (TC); 1: Channel 4 transmission complete (TC);
12	GIF4	R	0	Channel 4 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 4;
11	TEIF3	R	0	Channel 3 transmit error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 3 transmission error (TE);
10	HTIF3	R	0	Channel 3 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 3;
9	TCIF3	R	0	Channel 3 transmission completion flag. 0: no transmission completion (TC); 1: Channel 3 transmission complete (TC);

Bit	Name	R/W	Reset Value	Function
8	GIF3	R	0	Channel 3 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 3;
7	TEIF2	R	0	Channel 2 transmit error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 2 transmission error (TE);
6	HTIF2	R	0	Channel 2 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transmission event; 1: half-transmission event on channel 2;
5	TCIF2	R	0	Channel 2 transfer complete flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission completion (TC); 1: Channel 2 transmission complete (TC);
4	GIF2	R	0	Channel 2 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC events; 1: TE/HT/TC event on channel 2;
3	TEIF1	R	0	Channel 1 transmission error flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission error (TE); 1: Channel 1 transmission error (TE);
2	HTIF1	R	0	Channel 1 half-transfer flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no half-transfer event; 1: half-transmission event on channel 1;
1	TCIF1	R	0	Channel 1 transfer complete flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no transmission completion (TC); 1: Channel 1 transmission complete (TC);
0	GIF1	R	0	Channel 1 global interrupt flag. Hardware set, software write DMA_IFCR=1 to clear. 0: no TE/HT/TC event; 1: TE/HT/TC event on channel 1;

12.4.2. 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	CTEIF7	CHTIF7	CTCIF7	CGIF7	CTEIF6	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5	CTCIF5	CGIF5

				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**
CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	CTEIF7	W	0	Channel 7 transmission error flag cleared. 0: No effect; 1: Clear TEIF7;
26	CHTIF7	W	0	The channel 7 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF7;
25	CTCIF7	W	0	The channel 7 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF7;
24	CGIF7	W	0	Channel 7 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 7;
23	CTEIF6	W	0	Channel 6 transmission error flag cleared. 0: No effect; 1: Clear TEIF6;
22	CHTIF6	W	0	Channel 6 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF6;
21	CTCIF6	W	0	Channel 6 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF6;
20	CGIF6	W	0	Channel 6 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 6;
19	CTEIF5	W	0	Channel 5 transmission error flag cleared. 0: No effect; 1: Clear TEIF5;
18	CHTIF5	W	0	The channel 5 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF5;
17	CTCIF5	W	0	Channel 5 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF5;
16	CGIF5	W	0	Channel 5 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 5;
15	CTEIF4	W	0	Channel 4 transmission error flag cleared.

Bit	Name	R/W	Reset Value	Function
				0: No effect; 1: Clear TEIF4;
14	CHTIF4	W	0	The channel 4 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF4;
13	CTCIF4	W	0	The channel 4 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF4;
12	CGIF4	W	0	Channel 4 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 4;
11	CTEIF3	W	0	Channel 3 transmission error flag is cleared. 0: No effect; 1: Clear TEIF3;
10	CHTIF3	W	0	The channel 3 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF3;
9	CTCIF3	W	0	Channel 3 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF3;
8	CGIF3	W	0	Channel 3 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 3;
7	CTEIF2	W	0	Channel 2 transmission error flag cleared. 0: No effect; 1: Clear TEIF2;
6	CHTIF2	W	0	The channel 2 half-transmission flag is cleared. 0: No effect; 1: Clear HTIF2;
5	CTCIF2	W	0	Channel 2 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF2;
4	CGIF2	W	0	Channel 2 global interrupt flag is cleared to zero. 0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 2;
3	CTEIF1	W	0	Channel 1 transmission error flag is cleared. 0: No effect; 1: Clear TEIF1;
2	CHTIF1	W	0	Channel 1 half-transmission flag is cleared to zero. 0: No effect; 1: Clear HTIF1;
1	CTCIF1	W	0	Channel 1 transmission completion flag is cleared. 0: No effect; 1: Clear TCIF1;
0	CGIF1	W	0	Channel 1 global interrupt flag is cleared to zero.

Bit	Name	R/W	Reset Value	Function
				0: No effect; 1: Clear GIF/TEIF/HTIF/TCIF for channel 1;

12.4.3. DMA channel 1 configuration register (DMA_CCR1)

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	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: disable; 1: memory-to-memory mode enable;
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: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel 1 peripheral address incremental mode. 0: disabled; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel 1 cyclic mode.

Bit	Name	R/W	Reset Value	Function
				0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel 1 data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel 1 transmission error interrupt (TE) enable. 0: disable; 1: TE interrupt enable;
2	HTIE	RW	0	Channel 1 half-transmission interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel 1 Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel 1 enable . 0: disable; 1: Channel 1 enabled;

12.4.4. DMA channel 1 number of data register (DMA_CNDTR1)

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**
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 Number of data to transfer Number of data to be transferred (0 up to 65535). This register can only be written when the channel is disabled. Once the channel is enabled, this register is read-only, indicating the remaining bytes to be transmitted. This register decrements after each DMA transfer. Once the transfer is completed, this register can either stay at zero or be reloaded automatically by the value previously programmed if the channel is configured in autoreload mode.

Bit	Name	R/W	Reset Value	Function
				If this register is zero, no transaction can be served whether the channel is enabled or not.

12.4.5. DMA channel 1 peripheral address register (DMA_CPAR1)

Address offset:0x10

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, which is used as the source or destination for data transfer. 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] bits are not used. The operation is automatically aligned with the word address.

12.4.6. DMA channel 1 memory address register (DMA_CMAR1)

Address offset:0x14

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.

Bit	Name	R/W	Reset Value	Function
				When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.7. DMA channel 2 configuration register (DMA_CCR2)

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	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 2 memory-to-memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel 2 priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel 2 memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
9：8	PSIZE[1:0]	RW	0	Channel 2 peripheral data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
7	MINC	RW	0	Channel 2 memory address increment mode. 0: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel 2 peripheral address incremental mode.

Bit	Name	R/W	Reset Value	Function
				0: disable; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel 2 loop mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel 2 data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel 2 Transmission Error Interrupt (TE) Enable. 0: disable; 1: TE interrupt enable;
2	HTIE	RW	0	Channel 2 half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel 2 Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel 2 enable . 0: disable; 1: Channel 1 enabled;

12.4.8. DMA channel 2 number of data register (DMA_CNDTR2)

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**
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 2 Number of data to transfer Number of data to be transferred (0 up to 65535). This register can only be written when the channel is disabled. Once the channel is enabled, this register is read-only, indicating the remaining bytes to be transmitted. This register decrements after each DMA transfer.

Bit	Name	R/W	Reset Value	Function
				Once the transfer is completed, this register can either stay at zero or be reloaded automatically by the value previously programmed if the channel is configured in autoreload mode. If this register is zero, no transaction can be served whether the channel is enabled or not.

12.4.9. DMA channel 2 peripheral address register (DMA_CPAR2)

Address offset:0x24

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 2 peripheral address. The base address of the Channel 2 peripheral data register, which is used as the source or destination for data transfer. 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] bits are not used. The operation is automatically aligned with the word address.

12.4.10. DMA channel 2 memory address register (DMA_CMAR2)

Address offset:0x28

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 2 memory address.

				Channel 2 memory address, as the source or destination of data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.11. DMA channel 3 configuration register (DMA_CCR3)

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	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 3 memory-to-memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel 3 priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel 3 memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
9：8	PSIZE[1:0]	RW	0	Channel 3 peripheral data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
7	MINC	RW	0	Channel 3 memory address increment mode. 0: disable; 1: memory address increment mode enable;

Bit	Name	R/W	Reset Value	Function
6	PINC	RW	0	Channel 3 Peripheral Address Increment Mode. 0: disable; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel 3 loop mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel 3 data transfer direction. 0: Read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel 3 Transmission Error Interrupt (TE) Enable. 0: disable; 1: TE interrupt enable;
2	HTIE	RW	0	Channel 3 half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel 3 Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel 3 enable . 0: disable; 1: Channel 1 enabled;

12.4.12. DMA channel 3 number of data register (DMA_CNDTR3)

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**
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 3 Number of data to transfer Number of data to be transferred (0 up to 65535). This register can only be written when the channel is disabled. Once the channel is enabled, this register is read-only, indicating the remaining bytes to be transmitted. This register decrements after each DMA transfer.

Bit	Name	R/W	Reset Value	Function
				Once the transfer is completed, this register can either stay at zero or be reloaded automatically by the value previously programmed if the channel is configured in autoreload mode. If this register is zero, no transaction can be served whether the channel is enabled or not.

12.4.13. DMA channel 3 peripheral address register (DMA_CPAR3)

Address offset:0x38

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 3 peripheral address. The base address of the Channel 3 peripheral data register, which is used as the source or destination for data transfer. 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] bits are not used. The operation is automatically aligned with the word address.

12.4.14. DMA channel 3 memory address register (DMA_CMAR3)

Address offset:0x3C

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 3 memory address. Channel 3 memory address, as the source or destination of data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.15. DMA channel 4 configuration register (DMA_CCR4)

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**
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 memory to memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
9：8	PSIZE[1:0]	RW	0	Channel peripheral data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.

Bit	Name	R/W	Reset Value	Function
7	MINC	RW	0	Channel memory address increment mode. 0: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel Peripheral Address Increment Mode. 0: disabled; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel cycling mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel transmission error interrupt (TE) enable. 0: disabled; 1: TE interrupt enable;
2	HTIE	RW	0	Channel half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel enable . 0: disable; 1: channel enable;

12.4.16. DMA channel 4 number of data register (DMA_CNDTR4)

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**
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	Number of channel data transfers. The number of data transfers is from 0 to 65535. This register is only written when the channel is not operating (DMA_CCR3.EN=0). This register

Bit	Name	R/W	Reset Value	Function
				is read-only after the channel is enabled, indicating the number of bytes remaining to be transferred. This register value decrements after each DMA transfer. After the data transfer is completed, the contents of the register either change to 0, or when the channel is configured in cyclic mode, the contents of the register will be automatically reloaded to the value it had when it was previously configured. When this register value is 0, no data will be transferred even if the DMA channel starts.

12.4.17. DMA channel 4 peripheral address register (DMA_CPAR4)

Address offset:0x4C

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 Peripheral Address. The base address of the channel peripheral data register, used as the source or destination for data transfer. When PSIZE=2'b01, the PA[0] bit is not used. Operation is automatically aligned with the half-word address. When PSIZE=2'b10, the PA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.18. DMA channel 4 memory address register (DMA_CMAR4)

Address offset:0x50

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 memory address. The channel memory address, as the source or destination for data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.19. DMA channel 5 configuration register (DMA_CCR5)

Address offset:0x58

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 memory to memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
9：8	PSIZE[1:0]	RW	0	Channel peripheral data width. 00: 8 bits; 01: 16 bits;

Bit	Name	R/W	Reset Value	Function
				10: 32 bits; 11: Reserved.
7	MINC	RW	0	Channel memory address increment mode. 0: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel Peripheral Address Increment Mode. 0: disabled; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel cycling mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel transmission error interrupt (TE) enable. 0: disabled; 1: TE interrupt enable;
2	HTIE	RW	0	Channel half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel enable . 0: disable; 1: channel enable;

12.4.20. DMA channel 5 number of data register (DMA_CNDTR5)

Address offset:0x5C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
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	Number of channel data transfers. The number of data transfers is from 0 to 65535. This register is only written when the channel is not operating (DMA_CCR3.EN=0). This register is read-only after the channel is enabled, indicating the number of bytes remaining to be transferred. This register value decrements after each DMA transfer. After the data transfer is completed, the contents of the register either change to 0, or when the channel is configured in cyclic mode, the contents of the register will be automatically reloaded to the value it had when it was previously configured. When this register value is 0, no data will be transferred even if the DMA channel starts.

12.4.21. DMA channel 5 peripheral address register (DMA_CPAR5)

Address offset:0x60

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 Peripheral Address. The base address of the channel peripheral data register, used as the source or destination for data transfer. When PSIZE=2'b01, the PA[0] bit is not used. Operation is automatically aligned with the half-word address. When PSIZE=2'b10, the PA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.22. DMA channel 5 memory address register (DMA_CMAR5)

Address offset:0x64

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 memory address. The channel memory address, as the source or destination for data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.23. DMA channel 6 configuration register (DMA_CCR6)

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	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 memory to memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.

Bit	Name	R/W	Reset Value	Function
9：8	PSIZE[1:0]	RW	0	Channel peripheral data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
7	MINC	RW	0	Channel memory address increment mode. 0: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel Peripheral Address Increment Mode. 0: disabled; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel cycling mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel transmission error interrupt (TE) enable. 0: disabled; 1: TE interrupt enable;
2	HTIE	RW	0	Channel half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel enable . 0: disable; 1: channel enable;

12.4.24. DMA channel 6 number of data register (DMA_CNDTR6)

Address offset:0x70

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	Number of channel data transfers. The number of data transfers is from 0 to 65535. This register is only written when the channel is not operating (DMA_CCR3.EN=0). This register is read-only after the channel is enabled, indicating the number of bytes remaining to be transferred. This register value decrements after each DMA transfer. After the data transfer is completed, the contents of the register either change to 0, or when the channel is configured in cyclic mode, the contents of the register will be automatically reloaded to the value it had when it was previously configured. When this register value is 0, no data will be transferred even if the DMA channel starts.

12.4.25. DMA channel 6 peripheral address register (DMA_CPAR6)

Address offset:0x74

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 Peripheral Address. The base address of the channel peripheral data register, used as the source or destination for data transfer. When PSIZE=2'b01, the PA[0] bit is not used. Operation is automatically aligned with the half-word address. When PSIZE=2'b10, the PA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.26. DMA channel 6 memory address register (DMA_CMAR6)

Address offset:0x78

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 memory address. The channel memory address, as the source or destination for data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.27. DMA channel 7 configuration register (DMA_CCR7)

Address offset:0x80

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 memory to memory mode. 0: disable; 1: memory-to-memory mode enable;
13：12	PL[1:0]	RW	0	Channel priority configuration. 00: low; 01: medium; 10: high; 11: very high;
11：10	MSIZE[1:0]	RW	0	Channel memory data width. 00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
9：8	PSIZE[1:0]	RW	0	Channel peripheral data width.

Bit	Name	R/W	Reset Value	Function
				00: 8 bits; 01: 16 bits; 10: 32 bits; 11: Reserved.
7	MINC	RW	0	Channel memory address increment mode. 0: disable; 1: memory address increment mode enable;
6	PINC	RW	0	Channel Peripheral Address Increment Mode. 0: disabled; 1: Peripheral address increment mode enable;
5	CIRC	RW	0	Channel cycling mode. 0: disable; 1: cyclic mode enable;
4	DIR	RW	0	Channel data transfer direction. 0: read from peripheral; 1: Read from memory;
3	TEIE	RW	0	Channel transmission error interrupt (TE) enable. 0: disabled; 1: TE interrupt enable;
2	HTIE	RW	0	Channel half-transfer interrupt (HT) enable. 0: disable; 1: HT interrupt enable;
1	TCIE	RW	0	Channel Transmission Completion Interrupt (TC) enable. 0: disable; 1: TC interrupt enable;
0	EN	RW	0	Channel enable . 0: disable; 1: channel enable;

12.4.28. DMA channel 7 number of data register (DMA_CNDTR7)

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.	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	Number of channel data transfers. The number of data transfers is from 0 to 65535. This register is only written when the channel is not operating (DMA_CCR3.EN=0). This register is read-only after the channel is enabled, indicating the number of bytes remaining to be transferred. This register value decrements after each DMA transfer. After the data transfer is completed, the contents of the register either change to 0, or when the channel is configured in cyclic mode, the contents of the register will be automatically reloaded to the value it had when it was previously configured. When this register value is 0, no data will be transferred even if the DMA channel starts.

12.4.29. DMA channel 7 peripheral address register (DMA_CPAR7)

Address offset:0x88

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 Peripheral Address. The base address of the channel peripheral data register, used as the source or destination for data transfer. When PSIZE=2'b01, the PA[0] bit is not used. Operation is automatically aligned with the half-word address. When PSIZE=2'b10, the PA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.30. DMA channel 7 memory address register (DMA_CMAR7)

Address offset:0x8C

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 memory address. The channel memory address, as the source or destination for data transfer. When MSIZE=2'b01, the MA[0] bit is not used. Operation is automatically aligned with the half-word address. When MSIZE=2'b10, the MA[1:0] bits are not used. The operation is automatically aligned with the word address.

12.4.31. DMA register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DMA_ISR	Res.	Res.	Res.	Res.	TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	DMA_IFCR	Res.	Res.	Res.	Res.	CTEIF7	CHTIF7	CTCIF7.	CGIF7	CTEIF6.	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5.	CTCIF5	CGIF5	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	DMA_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DMA_CNDTR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	DMA_CPAR1	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	DMA_CMAR1	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	DMA_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	DMA_CNDTR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	DMA_CPAR2	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x28	DMA_CMAR2	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x30	DMA_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	DMA_CNDTR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	DMA_CPAR3	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	DMA_CMAR3	MA[31:0]																															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	DMA_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	DMA_CNDTR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	DMA_CPAR4	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	DMA_CMAR4	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x58	DMA_CCR5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x5C	DMA_CNDTR5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	DMA_CPAR5	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	DMA_CMAR5	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x6C	DMA_CCR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x70	DMA_CNDTR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x74	DMA_CPAR6	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x78	DMA_CMAR6	MA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x80	DMA_CCR7	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x84	DMA_CNDTR7	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x88	DMA_CPAR7	PA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	DMA_	MA[31:0]																															

Offset	**Register**	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
8C	CMAR7																																
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13. Interrupts and events

13.1. Nested vectored interrupt controller (NVIC)

13.1.1. NVIC main features

- 32 maskable interrupt channels (not including the 16 ARM® Cortex®-M0 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. All interrupts including the core exceptions are managed by the NVIC.

13.1.2. SysTick calibration value register

The SysTick calibration value is set to 6000, which gives a reference time base of 1 ms with the SysTick clock set to 6 MHz (max $f_{HCLK}/8$).

13.1.3. Interrupt and exception vectors

Position	Priority	Type of priority	Acronym	Description	Address
-	-	-	-	Reserved	0x0000_0000
-	-3	fixed	Reset	Reset	0x0000_0004
-	-2	fixed	NMI_Handler	Non maskable interrupt. The RCC Clock Security System (CSS) is linked to the NMI vector.	0x0000_0008
-	-1	fixed	HardFualt_Handler	All class of fault	0x0000_000C
-	3	settable	SVCall	System service call 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 watchdog interrupt	0x0000_0040
1	8	settable	PVD	Supply voltage detection interrupt (EXTI line 16)	0x0000_0044
2	9	settable	RTC	RTC interrupt (combined EXTI lines 19)	0x0000_0048
3	10	settable	Flash	Flash global interrupt	0x0000_004C
4	11	settable	RCC_CTC	RCC and CTC global interrupt	0x0000_0050

Position	Priority	Type of priority	Acronym	Description	Address
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
9	16	settable	DMA_Channel1	DMA channel 1 interrupt	0x0000_0064
10	17	settable	DMA_Channel2_3	DMA channel 2& 3 interrupt	0x0000_0068
11	18	settable	DMA_Channel4_5_6_7	DMA channel 4 & 5 & 6 & 7 interrupts	0x0000_006C
12	19	settable	ADC_COMP	ADC and COMP interrupts (COMP combined with EXTI 17 & 18 & 20)	0x0000_0070
13	20	settable	TIM1_BRK_UP_TRG_COM	TIM1 disconnect, update, trigger and communication interrupt	0x0000_0074
14	21	settable	TIM1_CC	TIM1 Capture/Compare Interrupt	0x0000_0078
15	22	settable	TIM2	TIM2 Global Interrupt	0x0000_007C
16	23	settable	TIM3	TIM3 Global Interrupt	0x0000_0080
17	24	settable	TIM6/LPTIM/DAC	TIM6/LPTIM/DAC Global Interrupt	0x0000_0084
18	25	settable	TIM7	TIM7 Global Interrupt	0x0000_0088
19	26	settable	TIM14	TIM14 Global Interrupt	0x0000_008C
20	27	settable	TIM15	TIM15 Global Interrupt	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_4	USART3_4 global interrupt	0x0000_00B4
30	37	settable	CAN	CAN Global Interrupt	0x0000_00B8
31	38	settable	USB	USB Global Interrupt	0x0000_00BC

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

13.2. Extended interrupts and events controller (EXTI)

The extended interrupt and event controller, through configurable (configurable) and direct (direct event) input (Lines), manages the CPU and system wake-up functions, and outputs the following request signals:

- Interrupt request, sent to the int_ctrl module to generate the IRQ of the CPU

- Event request, event input to CPU (RXEV)
- Wake-up request, sent to power management control module

EXTI wakeup request allows the system to wake up from stop mode, interrupt request and event request can also be used in run mode.

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

13.2.1. EXTI main features

- The system can be woken up by GPIO and specified module (PVD/COMP/RTC/LPTIM) input events
  - Configurable type events (from I/O, or peripherals without stateful pending bits, peripherals that generate pulses)
  - Optional valid trigger edge (rising/falling edge)
  - Interrupt pending flag bit
  - Independent interrupt and event generation mask bits
  - Software triggerable
- Direct-type events (peripherals with associated flags and interrupt pending status bits)
- Fixed rising edge trigger
- No interrupt pending bit in the EXTI module
- Independent interrupt and event generation mask bits
- No software triggering
- IO port selection

13.2.2. EXTI diagram

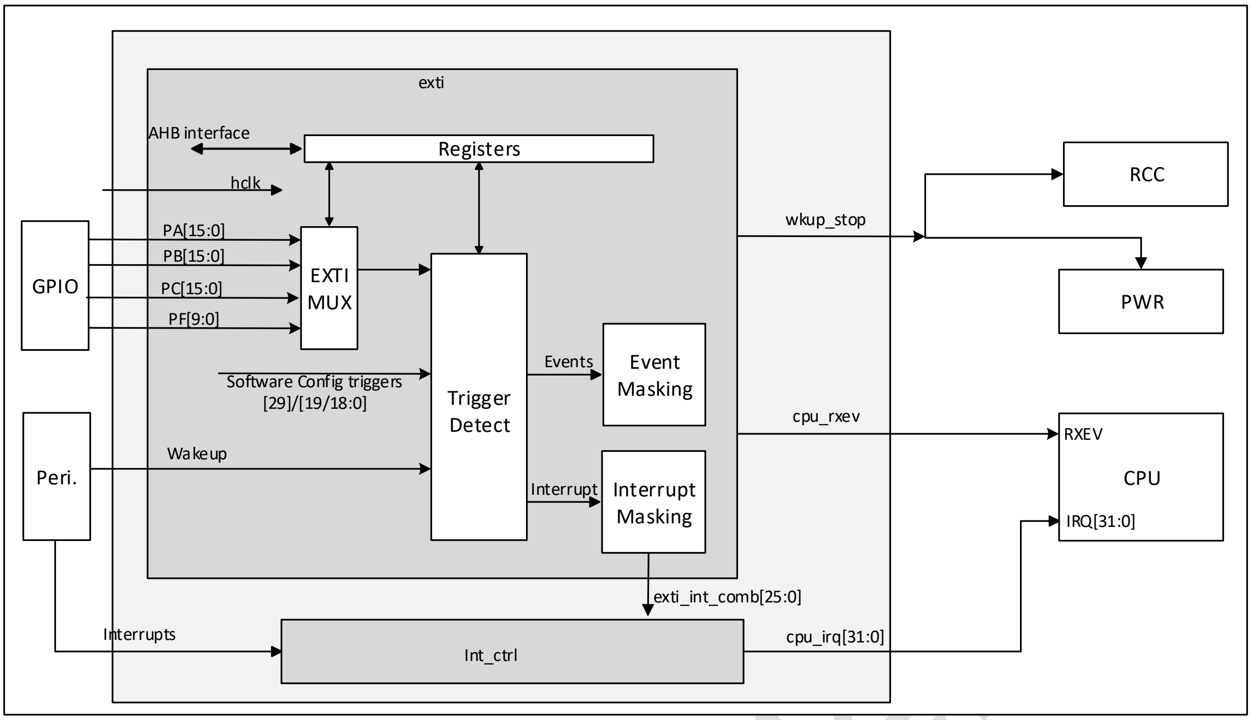


Figure 13-1 EXTI diagram

13.2.3. EXTI connection between peripherals and CPU

A peripheral that can generate a wake-up or interrupt event signal in stop mode is connected to the EXTI module.

- A wake-up signal that generates a pulse, or has no interrupt status bits inside the peripheral, is connected to the configurable line of the EXTI module. At this time, the EXTI module generates an interrupt pending bit (this bit needs to be cleared), and the EXTI interrupt will be used as the interrupt signal of the CPU.
- The interrupt and wake-up signal of the peripheral with the associated status bit (the bit is cleared in the peripheral) is connected to the wake-up trigger signal line of the EXTI module.
- All GPIO ports are input to the EXTI MUX module, and can be selected as a system wake-up signal through configurable configuration.

13.2.4. EXTI configurable event trigger wake-up

By configuring the EXTI_SWIER1 register, software can trigger the wake-up function.

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

The CPU has dedicated interrupt mask registers and event mask registers. The event generated to the CPU after the event is enabled. The only event input signal rxev that is output to the CPU after all events to the CPU are OR'ed.

Configurable type events have a unique interrupt pending request register, which is shared with the CPU. The pending register is only set when the CPU Interrupt Mask Register (EXTI_IMR) is configured as unmasked. Each configurable type event corresponds to a CPU external interrupt signal (some will be multiplexed to the same CPU external interrupt signal). Configurable type event interrupt requires the CPU to confirm through the EXTI_PR register (write 1 to clear).

Note: When a bit of the interrupt pending register (EXTI_PR) remains valid (not cleared), the system cannot enter the low power consumption mode.

13.2.5. EXTI direct type event input wakeup

The direct type event will generate an interrupt in the EXTI module, and will generate an event signal to wake up the system and the CPU subsystem. When the CPU processes the interrupt generated by this type of trigger event, it needs to clear the interrupt status bit of the peripheral module.

13.2.6. External and internal interrupt/event line mapping

The GPIOs are connected to the 16 external interrupt/event lines in the following manner:

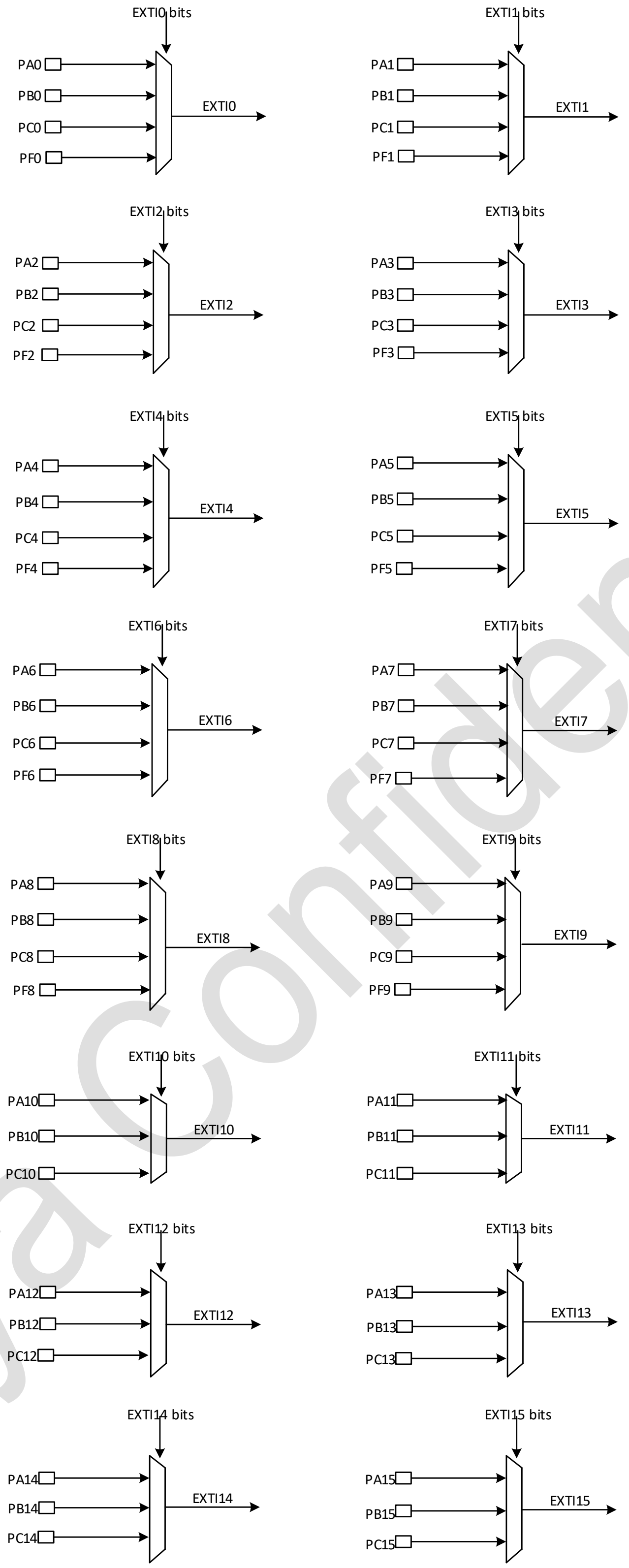


Figure 13-2 External interrupt/event GPIO mapping

The remaining lines are connected as follow:

EXTI line	Line source	Line type
Line 0-15	GPIO	configurable
Line 16	PVD output	Configurable
Line 17	COMP 1 output	Configurable

EXTI line	Line source	Line type
Line 18	COMP 2 output	Configurable
Line 19	RTC	Direct
Line 20	COMP3 output	Configurable
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
Line 30~33	Reserved	

13.3. EXTI registers

The registers of this peripheral can be accessed with word (32bit), half-word (16bit) and byte (8bit).

13.3.1. Rising trigger selection register (EXTI_RTSR)

Address offset: 0x00

Reset value: 0x0000 0000

Contains only register control 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	RT20	Res	RT18	RT17	RT16
											RW		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: 21	Reserved	-	-	-
20	RT20	RW	0	Configurable type EXTI line20 rising edge trigger configuration. 0: Disable 1: enable
19	Reserved	-	-	-
18	RT18	RW	0	Configurable type EXTI line18 rising edge trigger configuration. 0: Disable 1: enable
17	RT17	RW	0	Configurable type EXTI line17 rising edge trigger configuration. 0: Disable 1: enable
16	RT16	RW	0	Configurable type EXTI line16 rising edge trigger configuration. 0: Disable

Bit	Name	R/W	Reset Value	Function
				1: enable
15	RT15	RW	0	Configurable type EXTI line15 rising edge trigger configuration. 0: Disable 1: enable
14	RT14	RW	0	Configurable type EXTI line14 rising edge trigger configuration. 0: Disable 1: enable
13	RT13	RW	0	Configurable type EXTI line13 rising edge trigger configuration. 0: Disable 1: enable
12	RT12	RW	0	Configurable type EXTI line12 rising edge trigger configuration. 0: Disable 1: enable
11	RT11	RW	0	Configurable type EXTI line11 rising edge trigger configuration. 0: Disable 1: enable
10	RT10	RW	0	Configurable type EXTI line10 rising edge trigger configuration. 0: Disable 1: enable
9	RT9	RW	0	Configurable type EXTI line9 rising edge trigger configuration. 0: Disable 1: enable
8	RT8	RW	0	Configurable type EXTI line8 rising edge trigger configuration. 0: Disable 1: enable
7	RT7	RW	0	Configurable type EXTI line7 rising edge trigger configuration. 0: Disable 1: enable
6	RT6	RW	0	Configurable type EXTI line6 rising edge trigger configuration. 0: Disable 1: enable
5	RT5	RW	0	Configurable type EXTI line5 rising edge trigger configuration. 0: Disable 1: enable
4	RT4	RW	0	Configurable type EXTI line4 rising edge trigger configuration. 0: Disable 1: enable
3	RT3	RW	0	Configurable type EXTI line3 rising edge trigger configuration. 0: Disable 1: enable
2	RT2	RW	0	Configurable type EXTI line2 rising edge trigger configuration. 0: Disable 1: enable
1	RT1	RW	0	Configurable type EXTI line1 rising edge trigger configuration. 0: Disable

Bit	Name	R/W	Reset Value	Function
				1: enable
0	RT0	RW	0	Configurable type EXTI line0 rising edge trigger configuration. 0: Disable 1: enable

Configurable lines are edge-triggered, and glitches cannot be generated on these lines. If a rising edge occurs on the configurable interrupt line during a write to the EXTI_RTSR register, the associated Pending bit is not set.

Both rising and falling edges can be set on the same line, in which case both edges will generate a trigger condition.

13.3.2. Falling trigger selection register (EXTI_FTSR)

Address offset: 0x04

Reset value: 0x0000 0000

Contains only register control 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	FT20	Res	FT18	FT17	FT16
											RW		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: 21	Reserved	-	-	-
20	FT20	RW	0	Configurable type EXTI line20 falling edge trigger configuration. 0: Disable 1: enable
19	Reserved	-	-	-
18	FT18	RW	0	Configurable type EXTI line18 falling edge trigger configuration. 0: Disable 1: enable
17	FT17	RW	0	Configurable type EXTI line17 falling edge trigger configuration. 0: Disable 1: enable
16	FT16	RW	0	Configurable type EXTI line16 falling edge trigger configuration. 0: Disable 1: enable
15	FT15	RW	0	Configurable type EXTI line15 falling edge trigger configuration. 0: Disable 1: enable

Bit	Name	R/W	Reset Value	Function
14	FT14	RW	0	Configurable type EXTI line14 falling edge trigger configuration. 0: Disable 1: enable
13	FT13	RW	0	Configurable type EXTI line13 falling edge trigger configuration. 0: Disable 1: enable
12	FT12	RW	0	Configurable type EXTI line12 falling edge trigger configuration. 0: Disable 1: enable
11	FT11	RW	0	Configurable type EXTI line11 falling edge trigger configuration. 0: Disable 1: enable
10	FT10	RW	0	Configurable type EXTI line10 falling edge trigger configuration. 0: Disable 1: enable
9	FT9	RW	0	Configurable type EXTI line9 falling edge trigger configuration. 0: Disable 1: enable
8	FT8	RW	0	Configurable type EXTI line8 falling edge trigger configuration. 0: Disable 1: enable
7	FT7	RW	0	Configurable type EXTI line7 falling edge trigger configuration. 0: Disable 1: enable
6	FT6	RW	0	Configurable type EXTI line6 falling edge trigger configuration. 0: Disable 1: enable
5	FT5	RW	0	Configurable type EXTI line5 falling edge trigger configuration. 0: Disable 1: enable
4	FT4	RW	0	Configurable type EXTI line4 falling edge trigger configuration. 0: Disable 1: enable
3	FT3	RW	0	Configurable type EXTI line3 falling edge trigger configuration. 0: Disable 1: enable
2	FT2	RW	0	Configurable type EXTI line2 falling edge trigger configuration. 0: Disable 1: enable
1	FT1	RW	0	Configurable type EXTI line1 falling edge trigger configuration. 0: Disable 1: enable
0	FT0	RW	0	Configurable type EXTI line0 falling edge trigger configuration. 0: Disable 1: enable

Configurable lines are edge-triggered, and no burrs can be generated on these lines. If a falling edge occurs on a Configurable line during a write to the EXTI_FTSR register, the associated Pending bit is not set.

Both rising and falling edges can be set on the same line, in which case both edges will generate a triggering condition.

13.3.3. Software interrupt event register (EXTI_SWIER)

Address offset: 0x08

Reset value: 0x0000 0000

Contains only register control 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	SW18	Res	SW18	SW17	SW16
											RW		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
31：21	Reserved	-	-	-
20	SWI20	RW	0	Configurable type EXTI line20 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
19	Reseverd	-	-	-
18	SWI18	RW	0	Configurable type EXTI line18 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
17	SWI17	RW	0	Configurable type EXTI line17 software rising edge trigger configuration. 0: No effect

Bit	Name	R/W	Reset Value	Function
				1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
16	SWI16	RW	0	Configurable type EXTI line16 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
15	SWI15	RW	0	Configurable type EXTI line15 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
14	SWI14	RW	0	Configurable type EXTI line14 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
13	SWI13	RW	0	Configurable type EXTI line13 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
12	SWI12	RW	0	Configurable type EXTI line12 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
11	SWI11	RW	0	Configurable type EXTI line11 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt

Bit	Name	R/W	Reset Value	Function
				This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
10	SWI10	RW	0	Configurable type EXTI line10 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
9	SWI9	RW	0	Configurable type EXTI line9 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
8	SWI8	RW	0	Configurable type EXTI line8 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
7	SWI7	RW	0	Configurable type EXTI line7 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
6	SWI6	RW	0	Configurable type EXTI line6 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
5	SWI5	RW	0	Configurable type EXTI line5 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)

Bit	Name	R/W	Reset Value	Function
4	SWI4	RW	0	Configurable type EXTI line4 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
3	SWI3	RW	0	Configurable type EXTI line3 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
2	SWI2	RW	0	Configurable type EXTI line2 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
1	SWI1	RW	0	Configurable type EXTI line1 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
0	SWI0	RW	0	Configurable type EXTI line0 software rising edge trigger configuration. 0: No effect 1: Generate a rising edge trigger event, which in turn generates an interrupt This bit is cleared by hardware, and a read returns 0 (after hardware clearing) or configuration value (before hardware clearing)

13.3.4. Pending register (EXTI_PR)

Address offset: 0x0C

Reset value: undefined

Contains only register control 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	PR20	Res	PR18	PR17	PR16

											rc_w1		rc_w1	rc_w1	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	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit	Name	R/W	Reset Value	Function
31：21	Reserved	-	-	-
20	PR20	RC_W1	0	Configurable type EXTI line20 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
19	Reserved	-	-	-
18	PR18	RC_W1	0	Configurable type EXTI line18 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
17	PR17	RC_W1	0	Configurable type EXTI line17 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
16	PR16	RC_W1	0	Configurable type EXTI line16 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
15	PR15	RC_W1	0	Configurable type EXTI line15 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated,

Bit	Name	R/W	Reset Value	Function
				1: Generate rising edge/falling edge/software trigger event request,
14	PR14	RC_W1	0	Configurable type EXTI line14 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
13	PR13	RC_W1	0	Configurable type EXTI line13 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
12	PR12	RC_W1	0	Configurable type EXTI line12 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
11	PR11	RC_W1	0	Configurable type EXTI line11 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
10	PR10	RC_W1	0	Configurable type EXTI line10 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
9	PR9	RC_W1	0	Configurable type EXTI line9 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,

Bit	Name	R/W	Reset Value	Function
8	PR8	RC_W1	0	Configurable type EXTI line8 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
7	PR7	RC_W1	0	Configurable type EXTI line7 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
6	PR6	RC_W1	0	Configurable type EXTI line6 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
5	PR5	RC_W1	0	Configurable type EXTI line5 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
4	PR4	RC_W1	0	Configurable type EXTI line4 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
3	PR3	RC_W1	0	Configurable type EXTI line3 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,

Bit	Name	R/W	Reset Value	Function
2	RPIF2	RC_W1	0	Configurable type EXTI line2 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
1	PR1	RC_W1	0	Configurable type EXTI line1 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,
0	PR0	RC_W1	0	Configurable type EXTI line0 event pending flag. This bit is set when software or hardware generates a rising/falling edge trigger event. Software writes 1 to clear. 0: no event request is generated, 1: Generate rising edge/falling edge/software trigger event request,

13.3.5. External interrupt select 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	EXTI3 corresponds to GPIO port selection. 2'b00: PA[3] pin 2'b01: PB[3] pin 2'b10: PC[3] pin 2'b11: PF[3] pin
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

Bit	Name	R/W	Reset Value	Function
				2'b10: PC[2] pin 2'b11: PF[2] pin
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: PC[1] pin 2'b11: PF[1] pin
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: PC[0] pin 2'b11: PF[0] pin

13.3.6. External interrupt select 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[1:0]		Res	Res	Res	Res	Res	Res	EXTI6[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	EXTI5[1:0]		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: PC[7] pin 2'b11: PF[7] pin
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: PC[6] pin 2'b11: PF[6] pin
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: PC[5] pin

Bit	Name	R/W	Reset Value	Function
				2'b11: PF[5] pin
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: PC[4] pin 2'b11: PF[4] pin

13.3.7. External interrupt select register 3 (EXTI_EXTICR3)

Address offset: 0x68

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	EXTI11[1:0]		Res	Res	Res	Res	Res	Res	EXTI10[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	EXTI9[1:0]		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. 2'b00: PA[11] pin 2'b01: PB[11] pin 2'b10: PC[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: PC[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: PC[9] pin 2'b11: PF[9] pin
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: PC[8] pin 2'b11: PF[8] pin

13.3.8. External interrupt select register 4 (EXTI_EXTICR4)

Address offset:0x6C

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	EXTI15[1:0]		Res	Res	Res	Res	Res	Res	EXTI14[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	EXTI13[1:0]		Res	Res	Res	Res	Res	Res	EXTI12[1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI15[1:0]	RW	0	EXTI15 corresponds to GPIO port selection. 2'b00: PA[15] pin 2'b01: PB[15] pin 2'b10: PC[15] pin 2'b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI14[1:0]	RW	0	EXTI14 corresponds to GPIO port selection. 2'b00: PA[14] pin 2'b01: PB[14] pin 2'b10: PC[14] pin 2'b11: reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI13[1:0]	RW	0	EXTI13 corresponds to GPIO port selection. 2'b00: PA[13] pin 2'b01: PB[13] pin 2'b10: PC[13] pin 2'b11: reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI12[1:0]	RW	0	EXTI12 corresponds to GPIO port selection. 2'b00: PA[12] pin 2'b01: PB[12] pin 2'b10: PC[12] pin 2'b11: reserved

13.3.9. Interrupt mask register (EXTI_IMR)

Address offset:0x80

Reset value:0x2008 0000

The interrupt mask bit of the Direct type line is 1 by default, that is, the line is not masked, the mask bit of the configurable line, the default is 0, that is, the line is masked.

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	IM20	IM19	IM18	IM17	IM16
		RW									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	EXTI line29 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
28:21	Reserved	-	-	-
20	IM20	RW	1	EXTI line20 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
19	IM19	RW	1	EXTI line19 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
18	IM18	RW	0	EXTI line18 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
17	IM17	RW	0	EXTI line17 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
16	IM16	RW	0	EXTI line16 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
15	IM15	RW	0	EXTI line15 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
14	IM14	RW	0	EXTI line14 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
13	IM13	RW	0	EXTI line13 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask

Bit	Name	R/W	Reset Value	Function
				1: Interrupt wake-up is not masked
12	IM12	RW	0	EXTI line12 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
11	IM11	RW	0	EXTI line11 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
10	IM10	RW	0	EXTI line10 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
9	IM9	RW	0	EXTI line9 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
8	IM8	RW	0	EXTI line8 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
7	IM7	RW	0	EXTI line7 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
6	IM6	RW	0	EXTI line6 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
5	IM5	RW	0	EXTI line5 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
4	IM4	RW	0	EXTI line4 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
3	IM3	RW	0	EXTI line3 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
2	IM2	RW	0	EXTI line2 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked

Bit	Name	R/W	Reset Value	Function
1	IM1	RW	0	EXTI line1 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked
0	IM0	RW	0	EXTI line0 is used as an interrupt to wake up the CPU mask control. 0: interrupt wake-up mask 1: Interrupt wake-up is not masked

13.3.10. 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	EM20	EM19	EM18	EM17	EM16
		RW									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	EXTI line29 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
28:21	Reserved	-	-	-
20	EM20	RW	0	EXTI line20 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
19	EM19	RW	0	EXTI line19 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
18	EM18	RW	0	EXTI line18 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
17	EM17	RW	0	EXTI line17 wakes up the CPU mask control as an event. 0: Event wake-up mask

Bit	Name	R/W	Reset Value	Function
				1: Event wakeup is not masked
16	EM16	RW	0	EXTI line16 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
15	EM15	RW	0	EXTI line15 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
14	EM14	RW	0	EXTI line14 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
13	EM13	RW	0	EXTI line13 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
12	EM12	RW	0	EXTI line12 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
11	EM11	RW	0	EXTI line11 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
10	EM10	RW	0	EXTI line10 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
9	EM9	RW	0	EXTI line9 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
8	EM8	RW	0	EXTI line8 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
7	EM7	RW	0	EXTI line7 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
6	EM6	RW	0	EXTI line6 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked

Bit	Name	R/W	Reset Value	Function
5	EM5	RW	0	EXTI line5 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
4	EM4	RW	0	EXTI line4 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
3	EM3	RW	0	EXTI line3 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
2	EM2	RW	0	EXTI line2 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
1	EM1	RW	0	EXTI line1 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked
0	EM0	RW	0	EXTI line0 wakes up the CPU mask control as an event. 0: Event wake-up mask 1: Event wakeup is not masked

13.3.11. EXTI register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	EXTI_RTSR	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT20		RT18	RT17	RT16	RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	EXTI_F	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT20		FT18	FT17	FT16	FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0

04	TSR																																
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	EXTI_SWIER	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI20		SWI18	SWI17	SWI16	SWI15	SWI14	SWI13	SWI12	SWI11	SWI10	SWI9	SWI8	SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	EXTI_PR	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR20		PR18	PR17	PR16	PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	EXTI_EXTICR1	Res.	Res.	Res.	Res.	Res.	Res.	EXTI3[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI2[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI1[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI0[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x64	EXTI_EXTICR2	Res.	Res.	Res.	Res.	Res.	Res.	EXTI7[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI6[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI5[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI4[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x	EXTI_E	Res.	Res.	Res.	Res.	Res.	Res.	EXTI11[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI10[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI10[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI8[1:0]	

68	XTICR3																																
	Reset value							0	0							0	0							0	0							0	0
0x6C	EXTI_EXTICR4	Res.	Res.	Res.	Res.	Res.	Res.	EXTI15[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI14[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI12[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI11[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x84	EXTI_EMR	Res.	Res.	EM29	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EM20	EM19	EM18	EM17	EM16	EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
	Reset value			0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14. Cyclic redundancy check calculation unit (CRC)

14.1. Introduction

According to the generator polynomial, the CRC calculation unit will operate the input 32-bit data to generate a CRC result.

14.2. CRC main features

- Uses CRC-32 (Ethernet) polynomial: 0x4C11DB7
  - $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$
- Support 32-bit data input
- A single input/output 32 data and result output share one register
- 8-bit register for general purpose (can be used as temporary storage)
- Computation time: 4 AHB clocks for 32 bits data

14.3. CRC functional description

14.3.1. CRC block diagram

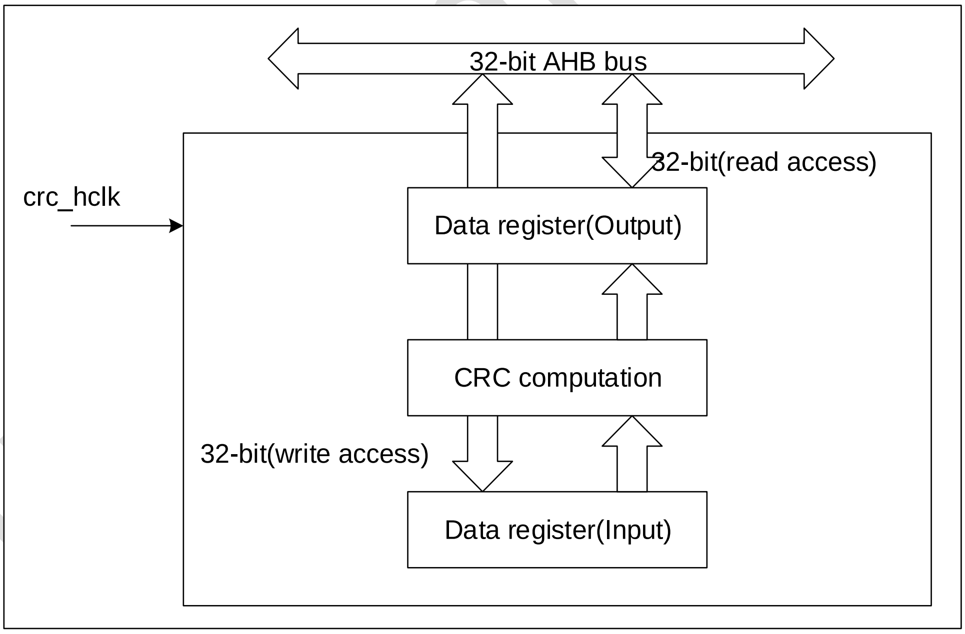


Figure 14-1 CRC calculation unit block diagram

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

- When writing to this register, as an input register, new data to be calculated by CRC can be input.
- When the register is read, the result of the last CRC calculation is returned.

Each time a data register is written, the result of the calculation is the combination of the previous CRC calculation and the new calculation (CRC is calculated on the entire 32-bit word, not byte by byte).

Supports configuration of CRC initial values.

The register CRC_DR can be reset to 0xFFFF FFFF by setting the RESET bit of the register CRC_CR. This operation does not affect the data in register CRC_IDR.

14.4. CRC registers

14.4.1. 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	0xFFFFFFFF	data register. When writing new data, it is used as an input register. When read, the previous CRC calculation result is retained.

14.4.2. Independent data register (CRC_IDR)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
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	RES	-	Reserved
7:0	IDR[7:0]	RW	0	General-purpose 8-bit data register bits.

Bit	Name	R/W	Reset Value	Function
				Can be used to temporarily store 1 byte of data. The CRC reset generated by the RESET bit of register CRC_CR has no effect on this register. Note: This register is not involved in CRC calculation and can store any data.

14.4.3. 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	RES	-	Reserved
0	RESET	W	0	A software reset will reset the CRC module and the data register will be loaded with the value of the CRC_INIT register. The software can only write 1 and is cleared by hardware.

14.4.4. CRC register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRC_DR	DR[31:0]																															
	Reset val	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ue																																
0x04	CRC_IDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x08	CRC_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESET
	Reset value																																0

15. Digital-to-analog converter (DAC)

15.1. DAC introduction

The digital/analog converter module (DAC) is a 12-bit digital input, voltage output digital/analog converter. The DAC can be configured in 8-bit or 12-bit mode, or it can be used with a DMA controller. When the DAC is operating in 12-bit mode, the data can be set to either left-aligned or right-aligned. The DAC module has 2 output channels, each with a separate converter. In dual DAC mode, the 2 channels can be converted independently, or they can be converted simultaneously and the outputs of the 2 channels can be updated synchronously. The DAC can input the reference voltage VREF+ through the pins to obtain more accurate conversion results.

15.2. DAC main features

- 2 DAC converters: each converter corresponds to 1 output channel;
- Left- or right-aligned data in 12-bit mode;
- Synchronous update function;
- Noise waveform generation;
- Triangular waveform generation;
- simultaneous or separate conversion of the dual DAC channels;
- DMA function for each channel;
- support for DMA underflow error detection;
- External trigger conversion;
- Input reference voltage V_{REF+};

The block diagram of a single DAC channel is shown below

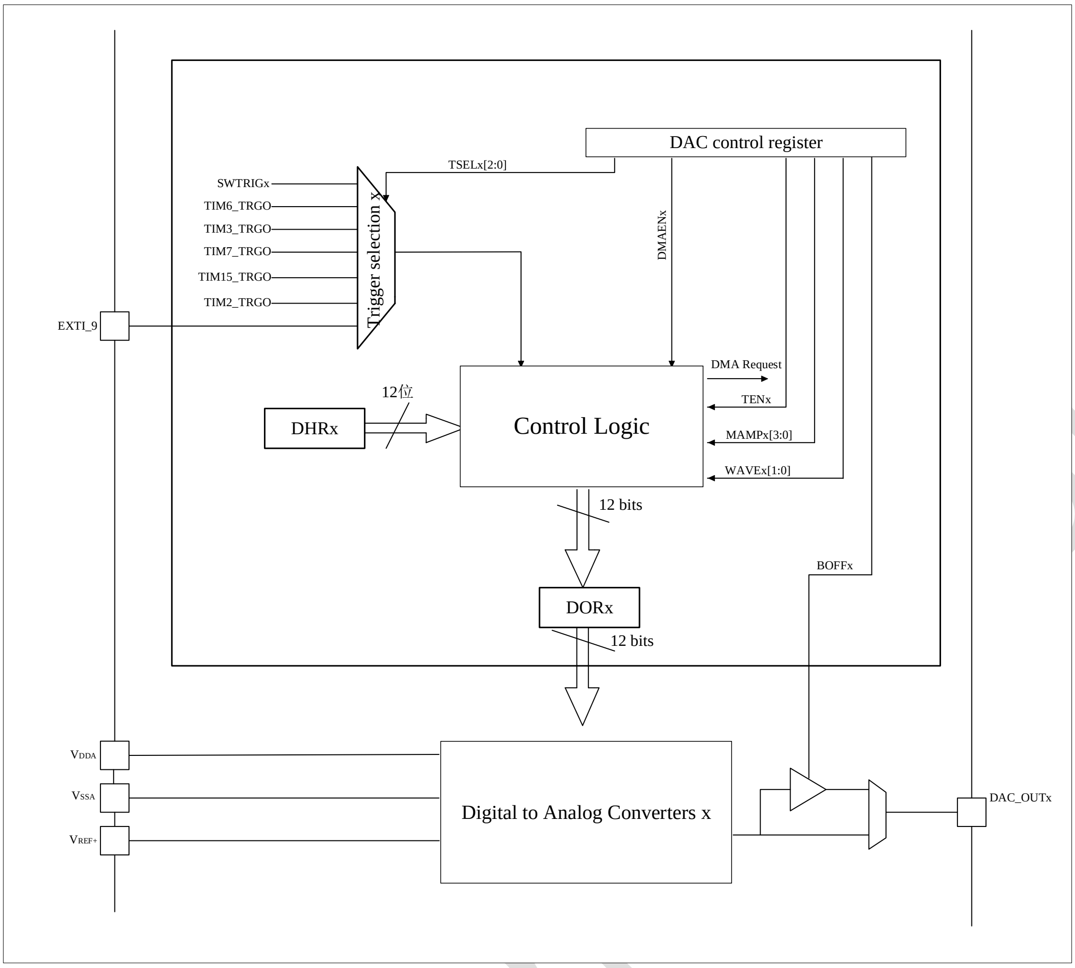


Figure 15-1 Block diagram of DAC channels

Table 15-1 DAC Pins

Name	Model Type	Description
V_{DDA}	Input, Analog Power	Analog Power
V_{SSA}	Input, Analog Power Ground	Analog Power Ground
V_{REF+}	Input, analog positive reference voltage	DAC positive reference voltage. 2.6 V ≤ V_{REF+} ≤ V_{DDA}
DAC_OUTx	Analog output signal	DACx analog output

Note: Before enabling the DAC module, the GPIO ports (PA4 corresponds to DAC channel 1 and PA5 corresponds to DAC channel 2) should be configured to analog mode.

15.3. DAC Function Description

15.3.1. Using DAC channels

The power supply to DAC channel x is turned on by setting the ENx position '1' of the DAC_CR register. After a start-up time of tWAKEUP, DAC channel x is enabled.

Note: The ENx bit will only enable the analog portion of DAC channel x. Even if this bit is set to '0', the digital portion of DAC channel x will still work.

15.3.2. Using DAC output cache

The DAC has two integrated output caches that can be used to reduce the output impedance and drive external loads directly without external op-amps. Each DAC channel output buffer can be enabled or disabled by setting the BOFFx bit of the DAC_CR register.

15.3.3. DAC data format

Depending on the selected configuration mode, data is written to the specified registers as described below:

- Single DAC channel x, with 3 cases:
- 8-bit data right-aligned: The user must write the data to the register DAC_DHR8Rx[7:0] bits (actually it is stored in the register DHRx[11:4] bits)
- 12-bit data left-aligned: the user must write the data to the register DAC_DHR12Lx[15:4] (actually it is stored in the register DHRx[11:0])
- 12-bit data right-aligned: the user must write the data to the register DAC_DHR12Rx[11:0] bits (actually it is stored in the register DHRx[11:0] bits)

According to the operation of the DAC_DHRyyyx register, the written data is dumped into the DHRx register after the corresponding shift (DHRx is the internal data retention register x). Subsequently, the contents of the DHRx register are transferred to the DORx register either automatically or by software trigger or external event trigger.

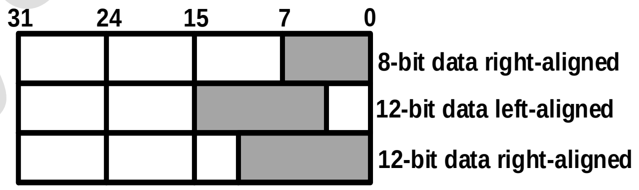


Figure 15-2 Data register for single DAC channel mode

- Dual DAC channels, with 3 cases:
  1) 8-bit data right-aligned: Users must write DAC channel 1 data to register DAC_DHR8RD[7:0] bits (actually stored in register DHR1[11:4] bits) and write DAC channel 2 data to register DAC_DHR8RD[15:8] bits (actually stored in register DHR2[11:4] bits).

2) 12-bit data left-aligned: users must write DAC channel 1 data to register DAC_DHR12LD[15:4] bits (actually stored in register DHR1[11:0] bits) and write DAC channel 2 data to register DAC_DHR12LD[31:20] bits (actually stored in register DHR2[11:0] bits)

3) 12-bit data right-aligned: Users must write DAC channel 1 data to register DAC_DHR12RD[11:0] bits (actually stored in register DHR1[11:0] bits), and write DAC channel 2 data to register DAC_DHR12RD[27:16] bits (actually stored in register DHR2[11:0] bits).

According to the operation of the DAC_DHRyyyD register, the written data is transferred to the DHR1 and DHR2 registers after the corresponding shift (DHR1 and DHR2 are the internal data storage registers x). Subsequently, the contents of DHR1 and DHR2 are either automatically transferred to the DORx registers, or triggered by software or external events.

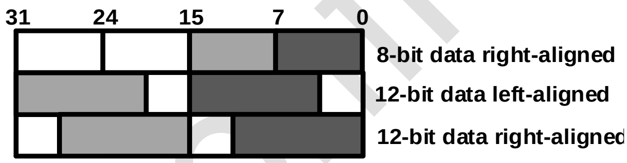


Figure 15-3 Data register for dual DAC channel mode

15.3.4. DAC conversion

Data cannot be written directly to register DAC_DORx. Any data output to DAC channel x must be written to the DAC_DHRx register (the data is actually written to the DAC_DHR8Rx, DAC_DHR12Lx, DAC_DHR12Rx, DAC_DHR8RD, DAC_DHR12LD, or DAC_DHR12RD registers).

If hardware trigger is not selected (TENx position '0' of register DAC_CRx), the data stored in register DAC_DHRx will be automatically passed to register DAC_DORx after 1 APB clock cycle. '1'), the data transfer is completed after 3 APB clock cycles after the trigger occurs.

Once the data is loaded from the DAC_DHRx register into the DAC_DORx register, the output is valid after the time tSETTLING, the length of which varies depending on the supply voltage and the analog output load.

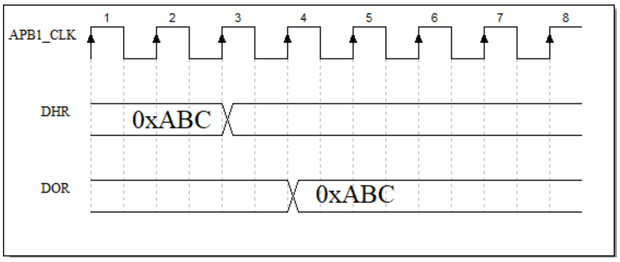


Figure 15-4 TEN=0 is the time block diagram of the conversion

15.3.5. DAC output voltage

The digital input is linearly converted by the DAC to an analog voltage output in the range of 0 to VREF+.

The output voltage on any DAC channel pin satisfies the following relationship:

DAC output = V_{REF} x (DOR / 4096)。

15.3.6. Selecting DAC Trigger

If the TENx bit is set to 1, the DAC conversion can be triggered by an external event (timer counter, external interrupt line). The configuration control bits TSELx[2:0] can select one of seven trigger events to trigger the DAC conversion.

Table 15-2 External Trigger

Trigger source	Type	TSELx[2:0]
Timer 6 TRGO Events	Internal signal from on-chip timer	000
Timer 3 TRGO Events		001
Timer 7 TRGO Events		010
Timer 15 TRGO Events		011
Timer 2 TRGO Events		100
Reserved		101
EXTI Line 9	External Pins	110
SWTRIG (Software Triggered)	Software control bits	111

Each time the DAC interface detects a rising edge from the selected timer TRGO or external interrupt line 9, the data recently stored in register DAC_DHRx is transferred to register DAC_DORx. After 3 APB clock cycles, register DAC_DORx is updated to the new value.

If software trigger is selected, the conversion starts as soon as SWTRIG position '1' is set. After the data is transferred from the DAC_DHRx register to the DAC_DORx register, the SWTRIG bit is automatically cleared '0' by hardware.

Note: 1. The TSELx[2:0] bits cannot be changed again when ENx is '1'.

2. If software trigger is selected, data transfer from register DAC_DHRx to register DAC_DORx takes only 1 APB clock cycle.

3. The frequency of Trigger trigger cannot exceed 1M.

15.3.7. DMA function

15.3.7.1. DMA Request

Either DAC channel has DMA function. The 2 DMA channels can be used for DMA requests for each of the 2 DAC channels.

If DMAENx position '1', once an external trigger (not a software trigger) occurs, a DMA request is generated after 3 clock cycles, and then the data in the DAC_DHRx register is transferred to the DAC_DORx register.

In dual DAC mode (i.e. using DAC_DHR12RD or DAC_DHR12LD or DAC_DHR8RD registers), only one bit in DMAENx should be set.

15.3.7.2. DMA underflow detection

The DAC's DMA request has no buffer queue, so if the second external trigger arrives before the answer to the first external trigger (first request) is received, no new DMA request will be issued and the DMA underflow flag DMAUDRx in the DAC_SR register will be set to "1", reporting an error condition . The DAC channel continues to convert old data.

The software should clear the DMAUDRx flag by writing a "1", clear the DMAEN bit of the DMA channel used, and reinitialize the DMA and DAC channels to properly restart the DMA transfer. The software should also modify the DAC trigger transition frequency or reduce the DMA workload to avoid another DMA underflow condition. Finally, the DAC conversion can be continued by enabling the DMA channels and configuring external triggering.

For each DAC channel, an interrupt will be generated if the corresponding DMAUDRIEx bit in the DAC_CR register is enabled.

15.3.8. Noise Generation

The Linear Feedback Shift Register LFSR can be used to generate pseudo-noise with varying amplitude. The DAC noise generation function is selected by setting the WAVE[1:0] bits to '01'. The register LFSR is preloaded with a value of 0xAAA. The value of this register is updated after 3 APB clock cycles after each trigger event according to a specific algorithm.

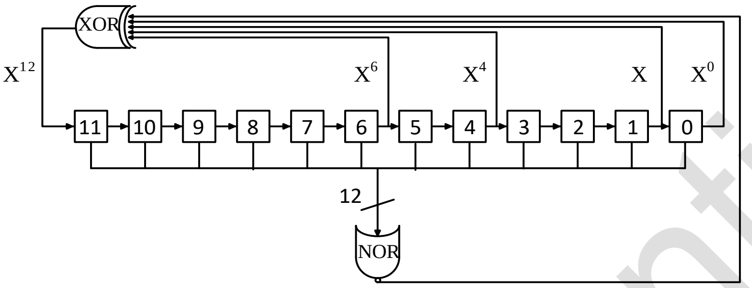


Figure 15-5 DAC LFSR register algorithm

Setting the MAMPx[3:0] bits of the DAC_CR register can mask some or all of the LFSR data, so that the value of LFSR obtained is added to the value of DAC_DHRx. If the value obtained has overflow, the maximum value that the register can keep is written to DAC_DORx, and if the value obtained does not have overflow, the value is written to DAC_DORx .

If the register LFSR value is 0x000, a '1' will be injected (anti-lock mechanism).

Setting WAVEx[1:0] location '0' can reset the generation algorithm of LFSR waveform.

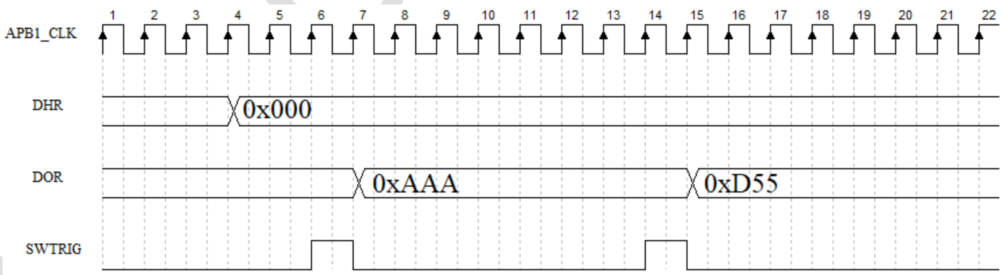


Figure 15-6 DAC conversion with LFSR waveform generation (enable software triggering)

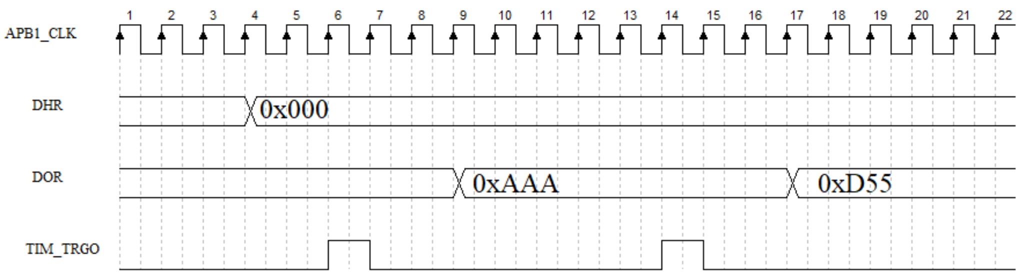


Figure 15-7 DAC conversion with LFSR waveform generation (enable hardware trigger)

1. In order to generate noise, the DAC trigger must be enabled, i.e., set the TENx bit of the DAC_CR register to '1';
2. If the value of DHR is changed, the value of LFSR register will be reset to 0xAAA;
3. If TENx drops from "1" to "0", the value of LFSR register will also be reset to 0xAAA.

15.3.9. Triangular wave generation

A small amplitude triangle wave can be added to the DC or slowly changing signal. Set the WAVEx[1:0] bits to "10" to select the triangle generation function of the DAC. Set the MAMPx[3:0] bits of the DAC_CR register to select the amplitude of the triangle waveform. The internal triangular wave counter accumulates 1 after 3 APB clock cycles after each trigger event, and the counter value is added to the DAC_DHRx register and the overflow bit is discarded and the DAC_DORx register is written. The delta counter is gradually accumulated when the value passed into the DAC_DORx register is less than the maximum amplitude defined by the MAMP[3:0] bits. Once the set maximum amplitude is reached, the counter starts to decrement, reaches 0 and then starts to accumulate again, and so on and so forth.

Setting WAVEx[1:0] position '0' can reset the generation of triangle waveform.

Note: When the base value is the maximum value of register DAC_DORx, the output of DAC_DORx will be its maximum value regardless of the value of MAMPx.

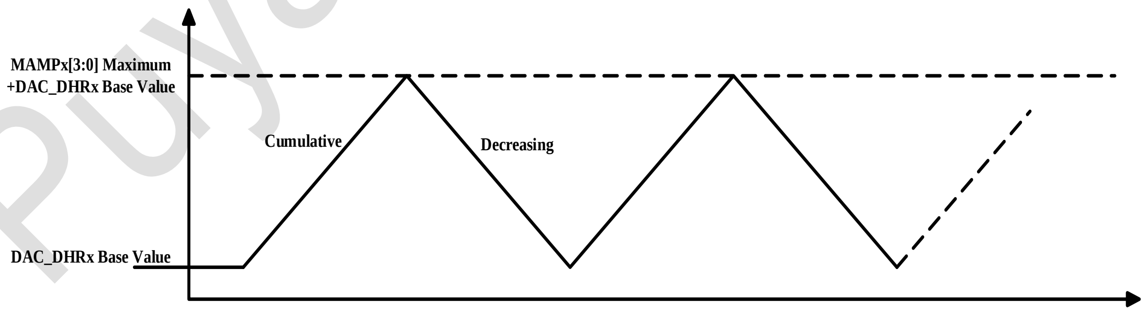


Figure 15-8 DAC delta wave generation

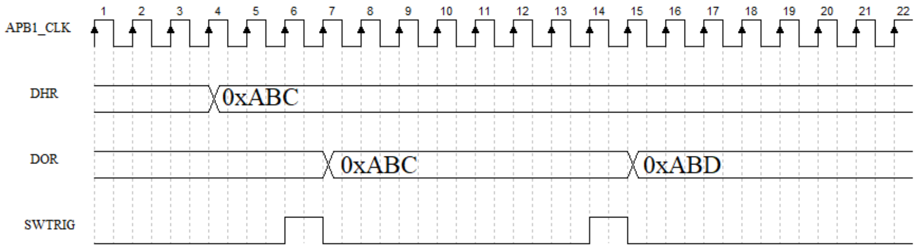


Figure 15-9 DAC conversion with delta generation (enabling software triggering)

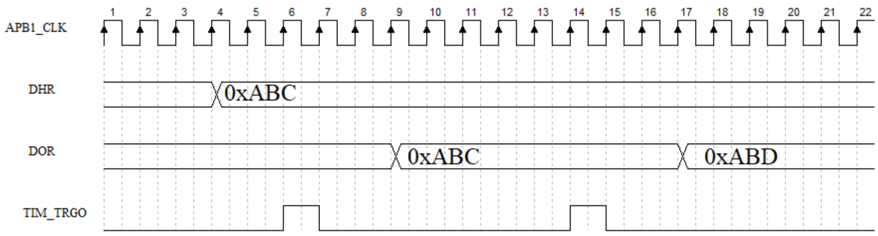


Figure 15-10 DAC conversion with delta generation (enable hardware triggering)

1. The DAC trigger must be enabled in order to generate a delta wave, i.e. set the TENx bit of the DAC_CR register to '1';
2. The MAMP[3:0] bits must be set prior to enabling the DAC, otherwise their value cannot be modified;
3. If the value of the DHR is changed, the delta counter is reset to 0
4. If TENx drops from "1" to "0", the triangle counter is also reset to 0.

15.3.10. Dual DAC channel conversion

To make more efficient use of the bus bandwidth when two DACs are required to operate simultaneously, the DAC incorporates three registers for dual DAC mode: DHR8RD, DHR12RD and DHR12LD, so that only one register can be accessed to drive two DAC channels simultaneously.

For dual DAC channel conversion and these dedicated registers, a total of 11 conversion modes are available. These conversion modes can still be operated via a separate DHRx register when only one DAC channel is used.

All modes are detailed in the following sections.

15.3.10.1. Not using a waveform generator

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure different trigger sources for each of the 2 DAC channels by setting the TSEL1[2:0] and TSEL2[2:0] bits to different values;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a DAC channel 1 trigger event occurs, (after a delay of 3 APB clock cycles) the value of register DHR1 is passed to register DAC_DOR1.

When a DAC channel 2 trigger event occurs, (after a delay of 3 APB clock cycles) the value of register DHR2 is passed to register DAC_DOR2.

15.3.10.2. Independent triggering using the same LFSR

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure different trigger sources for each of the 2 DAC channels by setting the TSEL1[2:0] and TSEL2[2:0] bits to different values;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "01" and setting MAMPx[3:0] to the same LFSR mask value;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a DAC channel 1 trigger event occurs, the LFSR1 counter value with the same mask is added to the DHR1 register value (after a delay of 3 APB clock cycles) and the result is passed to register DAC_DOR1, which then updates the LFSR1 counter. When a DAC channel 2 trigger event occurs, the LFSR2 counter value with the same mask is added to the DHR2 register value and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR2 register and the LFSR2 counter is updated.

15.3.10.3. Independent triggering using different LFSR's

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure different trigger sources for each of the 2 DAC channels by setting the TSEL1[2:0] and TSEL2[2:0] bits to different values;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "01" and setting MAMPx[3:0] to different LFSR mask values;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a DAC channel 1 trigger event occurs, the LFSR1 counter value masked according to MAMP1[3:0] is added to the DHR1 register value, (after a delay of 3 APB clock cycles) and the result is passed to register DAC_DOR1, then the LFSR1 counter is updated. When a DAC channel 2 trigger event occurs, the LFSR2 counter value, masked according to MAMP2[3:0], is added to the DHR2 register value and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR2 register and then the LFSR2 counter is updated.

15.3.10.4. Independent triggering to generate identical triangular waves

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure different trigger sources for each of the 2 DAC channels by setting the TSEL1[2:0] and TSEL2[2:0] bits to different values;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "1x" and setting MAMPx[3:0] to the same delta amplitude;

- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a DAC channel 1 trigger event occurs, the same delta amplitude value is added to the DHR1 register and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR1 register and the DAC channel 1 delta counter is updated. When a DAC channel 2 trigger event occurs, the same delta amplitude value is added to the DHR2 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR2 register, which then updates the DAC channel 2 delta counter.

15.3.10.5. Independent triggering for generating different triangular waves

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure different trigger sources for each of the 2 DAC channels by setting the TSEL1[2:0] and TSEL2[2:0] bits to different values;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "1x" and setting MAMPx[3:0] to the same delta amplitude;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a DAC channel 1 trigger event occurs, the same delta amplitude value is added to the DHR1 register and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR1 register and the DAC channel 1 delta counter is updated. When a DAC channel 2 trigger event occurs, the same delta amplitude value is added to the DHR2 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR2 register, which then updates the DAC channel 2 delta counter.

15.3.10.6. Simultaneous software start

Follow the procedure below to set the DAC to operate in this conversion mode:

- Load the dual DAC channel conversion data into the required DHR register (DHR12RD, DHR12LD or DHR8RD).

In this configuration, the values of the DHR1 and DHR2 registers are passed into the DAC_DOR1 and DAC_DOR2 registers respectively after one APB clock cycle.

15.3.10.7. Simultaneous triggering without waveform generator

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure the 2 DAC channels to use the same trigger source by setting the TSEL1[2:0] and TSEL2[2:0] bits to the same value, respectively;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a trigger event occurs, (after a delay of 3 APB clock cycles) the values of the DHR1 and DHR2 registers are passed to the DAC_DOR1 and DAC_DOR2 registers respectively.

15.3.10.8. Simultaneous triggering using the same LFSR

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure the 2 DAC channels to use the same trigger source by setting the TSEL1[2:0] and TSEL2[2:0] bits to the same value, respectively;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "01" and setting MAMPx[3:0] to the same LFSR mask value;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD);

When a trigger event occurs, the LFSR1 counter value set by MAMP1[3:0] is added to the value in the DHR1 register and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR1 register, which then updates the LFSR1 counter.

Similarly, the LFSR2 counter value set by MAMP1[3:0] is added to the DHR2 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR2 register, which then updates the LFSR2 counter.

15.3.10.9. Simultaneous triggering using different LFSRs

Set the DAC to operate in this conversion mode in the following order:

- Setting the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels respectively;
- Configure the 2 DAC channels to use the same trigger source by setting the TSEL1[2:0] and TSEL2[2:0] bits to the same value, respectively;
- setting the WAVEx[1:0] bits of the 2 DAC channels to "01" and setting MAMPx[3:0] to different LFSR mask values;
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a trigger event occurs, the LFSR1 counter value with the same mask is added to the DHR1 register value and (after a delay of 3 APB clock cycles) the result is passed to register DAC_DOR1, which then updates the LFSR1 counter.

At the same time, the LFSR2 counter value with the same mask is added to the DHR2 register value and (after a delay of 3 APB clock cycles) the result is passed to the DAC_DOR2 register and the LFSR2 counter is updated.

15.3.10.10. Simultaneous triggering using the same triangle wave generator

Set the DAC to operate in this conversion mode in the following order:

- Set the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels:

- Configure each of the 2 DAC channels to use the same trigger source by setting the TSEL1[2:0] and TSEL2[2:0] bits to the same value.
- Set the WAVEx[1:0] bits of both DAC channels to "1x" and set MAMPx[3:0] to the same delta amplitude.
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a trigger event occurs, the same delta amplitude is added to the DHR1 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR1 register, which then updates the delta counter.

At the same time, the same delta wave amplitude is added to the DHR2 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR2 register, which then updates the delta wave counter.

15.3.10.11. Simultaneous triggering using different triangle wave generators

Set the DAC to operate in this conversion mode in the following order:

- Set the trigger enable bits TEN1 and TEN2 to '1' for each of the 2 DAC channels:
- Configure each of the 2 DAC channels to use the same trigger source by setting the TSEL1[2:0] and TSEL2[2:0] bits to the same value.
- Set the WAVEx[1:0] bits of the 2 DAC channels to "1x" and set MAMPx[3:0] to different delta amplitude values.
- Load the dual DAC channel conversion data into the required DHR registers (DHR12RD, DHR12LD or DHR8RD).

When a trigger event occurs, the triangle amplitude value set by MAMP1[3:0] is added to the DHR1 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR1 register, which then updates the triangle counter.

At the same time, the triangle amplitude value set by MAMP2[3:0] is added to the DHR2 register (after a delay of 3 APB clock cycles) and the result is passed to the DAC_DOR2 register, which then updates the triangle counter.

15.3.11. DAC output clock

DAC output clock (DAC_CLK) is the analog side of the sampling DAC_DOR data clock, the clock is only valid when DAC_ENx is "1". When hardware trigger is selected, set "1" for 5 APB cycles after the rising edge of the hardware trigger is detected, and clear 0 after holding 3 APB cycles; when software trigger is selected, set "1" for 3 APB cycles after the rising edge of the software trigger is detected When software trigger is selected, set "1" for the 3 APB cycles after the rising edge of the software trigger is detected, and clear 0 after holding for 3 APB cycles; when no trigger is selected, set "1" for the 2 APB cycles after the data change is detected, and clear 0 after holding for 3 APB cycles.

Note: When TEN is lowered from "1" to "0" for 1 APB cycle, the data in DAC_DOR is updated to the data in the DAC_DHR register. If the data in the DAC_DHR register has been updated, a DAC_CLK will be generated when DAC_DOR is updated to the value of DAC_DHR; conversely, no DAC_CLK will be generated.

15.4. DAC register

These peripheral registers must be operated in a word (32-bit) manner.

15.4.1. DAC Control Register (DAC_CR)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	DACCEN2	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2[2:0]			TEN2	BOFF2	EN2
	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	DACCEN1	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[2:0]			TEN1	BOFF1	EN1
	RW	RW	RW	RW				RW		RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	0	-
30	DACEN2	RW	0	DAC channel 2 internal interconnect enable.

				0: output via IO; 1: internal interconnect to COMP/ADC
29	DMAUDRIE2	RW	0	DAC channel 2 DMA underflow interrupt enable This bit is set and cleared by software. 0: does not enable the DAC channel 2 DMA underflow interrupt; 1: Enables the DAC channel 2 DMA underflow interrupt.
28	DMAEN2	RW	0	DAC channel2 DMA enable (DAC channel2 DMA enable) This bit is set and cleared by software. 0: disables the DAC channel2 DMA mode; 1: Enables DAC channel2 DMA mode.
27:24	MAMP2[3:0]	RW	0	DAC channel2 mask/amplitude selector These bits are set by software to select the mask bit in noise generation mode and to select the amplitude of the waveform in triangle generation mode. 0000: no mask LSFR bit 0/amplitude of delta waveform equal to 1; 0001: non-masked LSFR bits [1:0]/triangle wave amplitude equal to 3; 0010: unmasked LSFR bits [2:0]/triangle wave amplitude equal to 7; 0011: unmasked LSFR bits [3:0]/triangle wave amplitude equal to 15; 0100: unmasked LSFR bits [4:0]/triangle wave amplitude equal to 31; 0101: unmasked LSFR bits [5:0]/triangle wave amplitude equal to 63; 0110: unmasked LSFR bits [6:0]/triangle wave amplitude equal to 127; 0111: unmasked LSFR bits [7:0]/triangle wave amplitude equal to 255; 1000: unmasked LSFR bits [8:0]/triangle amplitude equal to 511; 1001: unmasked LSFR bits [9:0]/triangle wave amplitude equal to 1023; 1010: unmasked LSFR bits [10:0]/triangle amplitude equal to 2047; ≥1011: unmasked LSFR bits [11:0]/triangle wave amplitude equal to 4095;
23:22	WAVE2[1:0]	RW	0	DAC channel2 noise/triangle wave generation enable The 2 bits are set and cleared by software. 00: disables the waveform generator; 01: enables the noise waveform generator; 1x: enables the triangle wave generation.
21:19	TSEL2[2:0]	RW	0	DAC channel2 trigger selection

				These 3 bits are used to select the external trigger event for DAC channel 2. 000: TIM6 TRGO event; 001: TIM3 TRGO event; 010: TIM7 TRGO event; 011: TIM15 TRGO event; 100: TIM2 TRGO event; 101: Reserved; 110: External interrupt line 9; 111: software trigger. Note: These 3 bits can only be used when TEN2 = 1 (DAC channel 2 trigger enable).
18	TEN2	RW	0	DAC channel2 trigger enable This bit is set and cleared by software to enable/disable the DAC channel2 trigger. 0: disable DAC channel2 trigger, the data written to DAC_DHRx register will be transferred to DAC_DOR2 register after 1 APB clock cycle; 1: Enable DAC channel 2 triggering, the data written to DAC_DHRx register is transferred to DAC_DOR2 regis-ter after 3 APB clock cycles. Note:If software triggering is selected, data written to register DAC_DHRx is passed into register DAC_DOR2 after only one APB always cycle.
17	BOFF2	RW	0	DAC channel2 output buffer disable This bit is set and cleared by software and is used to en-able/disable the DAC channel 2 output buffer. 0:enable the DAC channel2 output buffer; 1:Disable DAC channel2 output buffer.
16	EN2	RW	0	DAC channel2 enable This bit is set and cleared by software and is used to en-able/disable DAC channel 2. 0:disables DAC channel 2; 1:enable DAC channel2;
15	Reserved	-	-	-
14	DACEN1	RW	0	DAC channel 1 internal interconnect enable. 0: output via IO; 1: internal interconnect to COMP/ADC
13	DMAUDRIE1	RW	0	DAC channel 1 DMA underflow interrupt enable This bit is set and cleared by software. 0: does not enable the DAC channel 1 DMA underflow interrupt; 1: Enable DAC channel 1 DMA underflow interrupt
12	DMAEN1	RW	0	DAC channel1 DMA enable (DAC channel1 DMA ena-ble) This bit is set and cleared by software.

				0: disables the DAC channel1 DMA mode; 1: Enables DAC channel1 DMA mode.
11:8	MAMP1[3:0]	RW	0	DAC channel1 mask/amplitude selector (DAC channel1 mask/amplitude selector) These bits are set by software to select the mask bit in noise generation mode and the amplitude value of the waveform in triangle generation mode. 0000: no mask LSFR bit 0/amplitude of delta waveform equal to 1; 0001: non-masked LSFR bits [1:0]/triangle wave amplitude equal to 3; 0010: unmasked LSFR bits [2:0]/triangle wave amplitude equal to 7; 0011: unmasked LSFR bits [3:0]/triangle wave amplitude equal to 15; 0100: unmasked LSFR bits [4:0]/triangle wave amplitude equal to 31; 0101: unmasked LSFR bits [5:0]/triangle wave amplitude equal to 63; 0110: unmasked LSFR bits [6:0]/triangle wave amplitude equal to 127; 0111: unmasked LSFR bits [7:0]/triangle wave amplitude equal to 255; 1000: unmasked LSFR bits [8:0]/triangle amplitude equal to 511; 1001: unmasked LSFR bits [9:0]/triangle wave amplitude equal to 1023; 1010: unmasked LSFR bits [10:0]/triangle amplitude equal to 2047; ≥1011: unmasked LSFR bits [11:0]/triangle wave amplitude equal to 4095;
7:6	WAVE1[1:0]	RW	0	DAC channel1 noise/triangle wave generation enable These 2 bits are set and cleared by software. 00:disables the waveform generator; 01:enable the noise waveform generator; 1x:Enable triangle wave generation.
5:3	TSEL1[2:0]	RW	0	DAC channel1 trigger selection These 3 bits are used to select the external trigger event for DAC channel 1. 000: TIM6 TRGO event; 001: TIM3 TRGO event; 010: TIM7 TRGO event; 011: TIM15 TRGO event; 100: TIM2 TRGO event; 101: Reserved; 110: External interrupt line 9;

				111: software trigger. Note: These 3 bits can only be used when TEN1 = 1 (DAC channel 1 trigger enable).
2	TEN1	RW	0	DAC channel1 trigger enable This bit is set and cleared by software to enable/disable DAC channel1 trigger. 0: disable DAC channel1 trigger, the data written to DAC_DHRx register will be transferred to DAC_DOR1 register after 1 APB clock cycle; 1: Enable DAC channel 1 triggering, the data written to the DAC_DHRx register is transferred to the DAC_DOR1 register after 3 APB clock cycles. Note:If software triggering is selected, data written to register DAC_DHRx is passed into register DAC_DOR1 after only one APB always cycle.
1	BOFF1	RW	0	DAC channel1 output buffer disable This bit is set and cleared by software and is used to en-able/disable the DAC channel 1 output buffer. 0: Enables the DAC channel 1 output buffer; 1: Disables the DAC channel 1 output buffer.
0	EN1	RW	0	DAC channel1 enable This bit is set and cleared by software and is used to en-able/disable DAC channel 1. 0: disables DAC channel 1; 1: enables DAC channel 1;

15.4.2. DAC Software Trigger Register (DAC_SWTRIGR)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	SWTRIG2	SWTRIG1
														W	W

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	0	-
1	SWTRIG2	W	0	DAC channel2 software trigger This bit is set and cleared by software and is used to en-able/disable software trigger. 0: disables DAC channel2 software trigger; 1: Enables DAC channel2 software trigger.

				Note: Once the data from register DAC_DHR2 is passed into register DAC_DOR2, (after 1 APB clock cycle) this bit is set to '0' by hardware.
0	SWTRIG1	W	0	DAC channel1 software trigger This bit is set and cleared by software and is used to enable/disable software trigger. 0: disables DAC channel1 software trigger; 1: Enables DAC channel1 software trigger. Note: Once the data from register DAC_DHR1 is passed into register DAC_DOR1, (after 1 APB clock cycle) this bit is set to '0' by hardware

15.4.3. 12-bit right-aligned data holding register for DAC channel 1 (DAC_DHR12R1)

Address offset:0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	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	DACC1DHR[11:0]											
				RW											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved		0	
11:0	DACC1DHR[11:0]	RW	0	DAC channel1 12-bit right-aligned data (DAC channel1 12-bit right-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 1

15.4.4. 12-bit left-aligned data holding register for DAC channel 1 (DAC_DHR12L1)

Address offset:0x0C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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**
DACC1DHR[11:0]												Res	Res	Res	Res
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	-

15:4	DACC1DHR[11:0]	RW	0	DAC channel1 12-bit left-aligned data (DAC channel1 12-bit left-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 1.
3:0	Reserved	-	0	-

15.4.5. 8-bit right-aligned data holding register for DAC channel 1 (DAC_DHR8R1)

Address offset:0x10

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	DACC1DHR[7:0]							
								RW							

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	-
7:0	DACC1DHR[7:0]	RW	0	DAC channel1 8-bit right-aligned data (DAC channel1 8-bit right-aligned data) This bit is written by software and represents the 8-bit data of DAC channel 1.

15.4.6. 12-bit right-aligned data holding register for DAC channel 2 (DAC_DHR12R2)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	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	DACC2DHR[11:0]											
				RW											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	0	-
11:0	DACC2DHR[11:0]	RW	0	DAC channel2 12-bit right-aligned data (DAC channel2 12-bit right-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 2.

15.4.7. 12-bit left-aligned data holding register for DAC channel 2 (DAC_DHR12L2)

Address offset:0x18

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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**
DACC2DHR[11:0]												Res	Res	Res	Res
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	-
15:4	DACC2DHR[11:0]	RW	0	DAC channel2 12-bit left-aligned data (DAC channel2 12-bit left-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 2.
3:0	Reserved		0	

15.4.8. 8-bit right-aligned data holding register for DAC channel 2 (DAC_DHR8R2)

Address offset:0x1C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	DACC2DHR[7:0]							
								RW							

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	-
7:0	DACC2DHR[7:0]	RW	0	DAC channel2 8-bit right-aligned data (DAC channel2 8-bit right-aligned data) This bit is written by software and represents the 8-bit data of DAC channel 2.

15.4.9. 12-bit right-aligned data holding register for dual DACs (DAC_DHR12RD)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	DACC2DHR[11:0]											
				RW											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	DACC1DHR[11:0]											

				RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	0	-
27:16	DACC2DHR[11:0]	RW	0	DAC channel2 12-bit right-aligned data (DAC channel2 12-bit right-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 2.
15:12	Reserved	-	0	-
11:0	DACC1DHR[11:0]	RW	0	DAC channel1 12-bit right-aligned data (DAC channel1 12-bit right-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 1.

15.4.10. 12-bit left-aligned data holding register for dual DACs (DAC_DHR12LD)

Address offset:0x24

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DACC2DHR[11:0]												Res	Res	Res	Res
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DACC1DHR[11:0]												Res	Res	Res	Res
RW															

Bit	Name	R/W	Reset Value	Function
31:20	DACC2DHR[11:0]	RW	0	DAC channel2 12-bit left-aligned data (DAC channel2 12-bit left-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 2.
19:16	Reserved		0	
15:4	DACC1DHR[11:0]	RW	0	DAC channel1 12-bit left-aligned data (DAC channel1 12-bit left-aligned data) This bit is written by software and represents the 12-bit data of DAC channel 1.
3:0	Reserved		0	

15.4.11. 8-bit right-aligned data holding register for dual DACs (DAC_DHR8RD)

Address offset:0x28

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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
DACC2DHR[7:0]								DACC1DHR[7:0]							
RW								RW							

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	-
15:8	DACC2DHR[7:0]	RW	0	DAC channel2 8-bit right-aligned data (DAC channel2 8-bit right-aligned data) This bit is written by software and represents the 8-bit data of DAC channel 2.
7:0	DACC1DHR[7:0]	RW	0	DAC channel1 8-bit right-aligned data (DAC channel1 8-bit right-aligned data) This bit is written by software and represents the 8-bit data of DAC channel 1.

15.4.12. DAC channel 1 data output register (DAC_DOR1)

Address offset:0x2C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	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	DACC1DOR[11:0]											
				R											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	0	-
11:0	DACC1DOR[11:0]	R	0	DAC channel1 data output) This bit is read only and indicates the output data of DAC channel 1.

15.4.13. DAC channel 2 data output register (DAC_DOR2)

Address offset:0x30

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	DACC2DOR[11:0]											
				R											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	0	-
11:0	DACC2DOR[11:0]	R	0	DAC channel2 data output) This bit is read only and indicates the DAC channel2 output data.

15.4.14. DAC Status Register (DAC_SR)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	DMAUDR2	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
		rc_w1													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	DMAUDR1	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
		rc_w1													

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	0	-
29	DMAUDR2	rc_w1	0	DAC channel 2 DMA underrun flag This bit is set to 1 by hardware and cleared to 0 by soft-ware write 1 0: no DMA underrun has occurred on DAC channel 2; 1: DAC channel 2 has a DMA underrun condition. Note: Software can only write "1" to clear "0" after DMAUDR2 is set to "1"
28:14	Reserved	-	0	-
13	DMAUDR1	rc_w1	0	DAC channel 1 DMA underrun flag This bit is set to 1 by hardware and cleared to 0 by soft-ware write 1 0: no DMA underrun has occurred on DAC channel 1; 1: DAC channel 1 has a DMA underrun condition. Note: Software can only write "1" to clear "0" after DMAUDR1 is set to "1"
12:0	Reserved	-	0	-

15.4.15. DAC register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DAC_CR	Res.	DACCEN2	DMAUDRIE2	DMAEN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2 [2:0]			TEN2	BOFF2	EN2	Res.	DACCEN1	DMAUDRIE1	DMAEN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1 [2:0]			TEN1	BOFF1	EN1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reset value		0	0	0	0	0	0	0	0	0	0	0		0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	DAC_SWTRIGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWTRIG2	SWTRIG1
	Reset value																															0	0
0x08	DAC_DHR12R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DAC_DHR12R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[11:0]												Res.	Res.	Res.	Res.
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0				
0x10	DAC_DHR8R1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DHR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x14	DAC_DHR12R2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x18	DAC_DHR12L2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[11:0]												Res.	Res.	Res.	Res.
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0				
0x1C	DAC_DHR8R2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x20	DAC_DHR12RD	Res.	Res.	Res.	Res.	DACC2DHR[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x24	DAC_DHR1	DACC2DHR[11:0]												Res.	Res.	Res.	Res.	DACC1DHR[11:0]												Res.	Res.	Res.	Res.

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	2LD																																
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
0x28	DAC_DHR8RD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DHR[7:0]								DACC1DHR[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	DAC_DOR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC1DOR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x30	DAC_DOR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACC2DOR[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x34	DAC_SR	Res.	Res.	DMAUDR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAUDR1		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value			0																0													

16. Analog-to-digital converter (ADC)

16.1. Introduction

The 12-bit ADC is a successive approximation type analogue-to-digital converter. It has up to 24 channels and can measure up to 16 external and 8 internal signal sources. The A/D conversion of each channel can be performed in single, continuous, sweep or intermittent mode. The results of the ADC can be stored in a 16-bit data register in either left- or right-aligned mode.

The analogue watchdog feature allows the application to detect if the input voltage exceeds a user defined high/low threshold.

16.2. ADC main features

- High performance
  - 12-bit, 10-bit, 8-bit or 6-bit configurable resolution
  - ADC conversion time: 1.0 µs for 12-bit resolution (1 MHz)
  - Self-calibration
  - Programmable sampling time
  - Programmable data alignment mode
  - DMA support for rule groups
- Analogue input channels
  - 16 external analogue input channels
  - 1 internal temperature sensor channel (TSENSOR)
  - 1 internal reference voltage channel (VREFINT)
  - 1 internal reference voltage input channel (VERF Buffer)
  - 3 internal OPA input voltage channels (OPA)
  - 2 internal DAC input voltage channels
- The conversion operation can be initiated via
  - Software start-up
  - Hardware start (TIM1, TIM2, TIM3, TIM15 or GPIO)
- Conversion mode

  - Single mode: 1 single channel can be converted
  - Scan mode: a series of channels can be scanned
  - Continuous mode: continuous conversion of the selected channel
  - Intermittent mode: each trigger converts a sub-sequence of channels, with multiple triggers until the complete sequence is converted
- Interruptions are generated
  - At the end of the conversion
  - Simulating a watchdog event
- Analogue watchdog
- ADC power requirements: 1.7 V to 5.5 V
- ADC input range: VREF- ≤ VIN ≤ VREF+

16.3. ADC functional description

16.3.1. ADC diagram

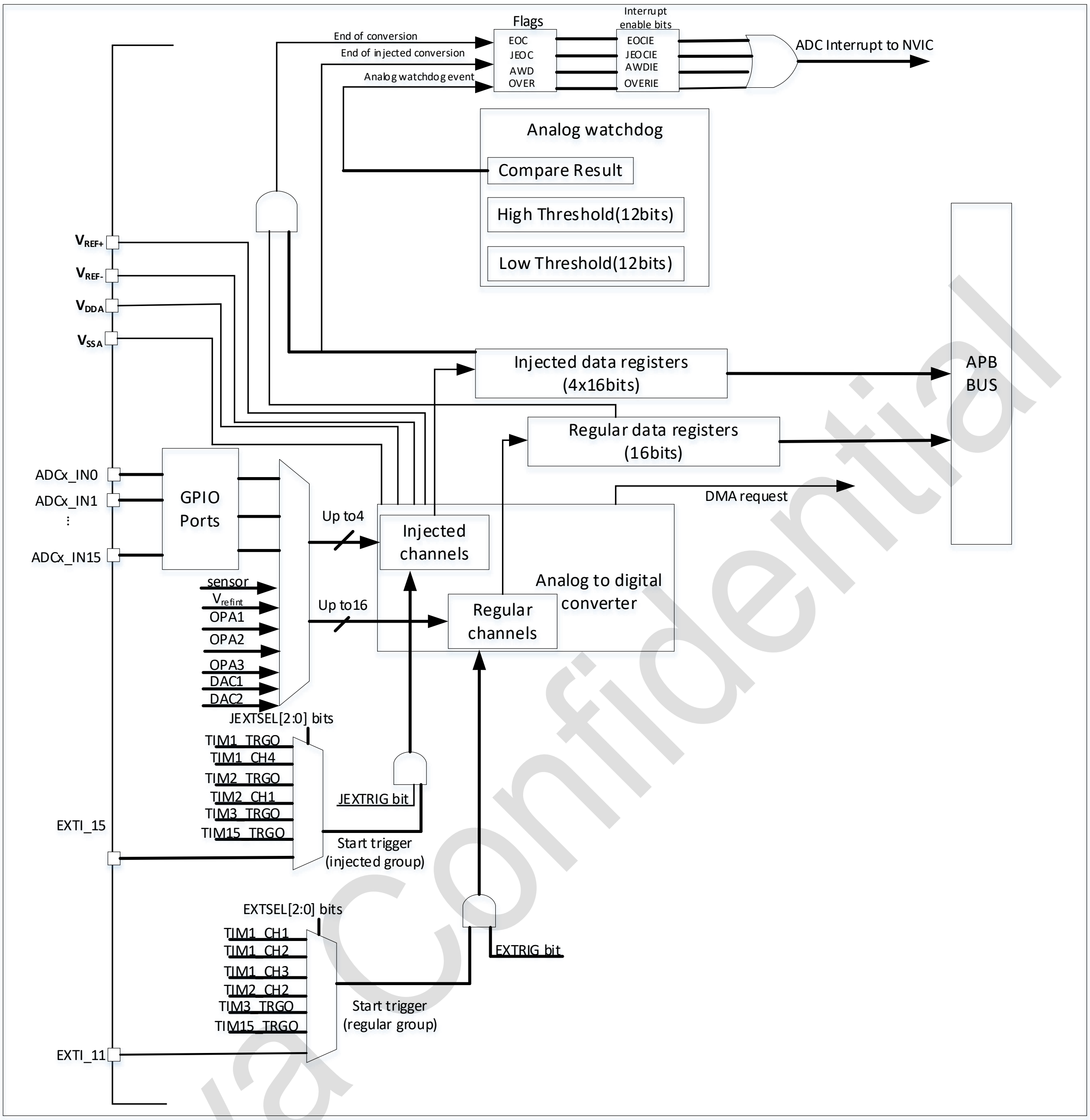


Figure 16-1 ADC block diagram

16.3.2. Calibration

The ADC has a software calibration function. During calibration, the ADC calculates a calibration factor for use within the ADC (lost when the ADC is powered down). The ADC module cannot be used by the application during ADC calibration and until the calibration has been completed.

The calibration operation is performed before the ADC conversion is used. Calibration is used to eliminate chip-to-chip, offset errors due to process variations.

Calibration can only be started when the ADC is not enabled (ADON=0) and only supports the selection of the system clock as the ADC's clock. When calibration is complete, CAL is cleared by hardware to 0.

When the operating conditions of the ADC change (VCC change is the main factor for ADC offset offset, temperature change is second), a recalibration operation is recommended.

The software procedure for calibration:

- Confirm ADON=0
- Set CAL=1
- Wait until CAL=0

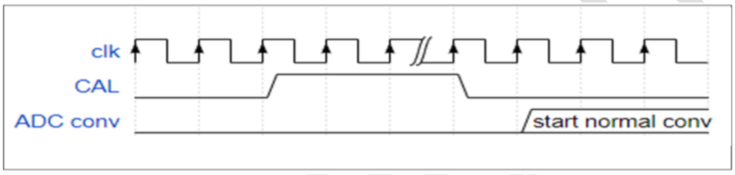


Figure 16-1 ADC Calibration Timing Diagram

16.3.3. ADC on-off control

The ADC can be powered up by setting the ADON bit of the ADC_CR2 register. When the ADON bit is set for the first time, it wakes up the ADC from a power-down state.

from a power-down state.

The ADC starts to convert after a power-up delay (t_{STAB}, not less than 20 us).

To save power, the ADC analogue sub-module will go into power-down mode when ADON is 0. By clearing the ADON bit, conversion can be stopped and the ADC placed in power-down mode.

16.3.4. ADC clock

The ADCCLK clock provided by the RCC control is synchronised with the PCLK (APB clock). the RCC controller (CLK controller) provides a dedicated programmable prescaler for the ADC clock and the ADCCLK clock source has an RCC matchable divider from the PCLK, see RCC_ADC_DIV for the phase.

16.3.5. Channel selection

There are 16 external and 8 internal channels, of which the internal channels are

Temperature sensor /V_{REFINT} internal channel

The temperature sensor is connected to channel ADC1_IN16 and the internal reference voltage V_{REFINT} is connected to ADC1_IN17.

VCC/3

VCC/3 and channel ADC1_IN18 are connected.

DAC

DAC1_VIN is connected to channel ADC1_IN19 and DAC2_VIN is connected to channel ADC1_IN20.

OPA

OPA1_VIN is connected to channel ADC1_IN21, OPA2_VIN is connected to channel ADC1_IN22 and OPA3_VIN is connected to channel ADC1_IN23.

Conversions can be organised into two groups: rule groups and injection groups. A series of transitions performed in any order on any number of channels constitutes a group of transitions. For example, conversions can be done in the following order: channel 3, channel 8, channel 2, channel 2, channel 0, channel 2, channel 2, channel 15.

Rule groups consist of up to 16 conversions. The rule channels and the order of their conversions are selected in the ADC_SQRx register. The total number of conversions in the rule group should be written to the L[3:0] bits of the ADC_SQR1 register.

The injection group consists of up to 4 conversions. The injection channels and the order of their conversions are selected in the ADC_JSQR register. The total number of conversions in the injection group should be written into the JL[1:0] bits of the ADC_JSQR register.

If the ADC_SQRx or ADC_JSQR registers are changed during conversion, the current conversion is cleared and a new start pulse is sent to the ADC to convert the newly selected group.

16.3.6. Programmable sampling time

The ADC uses a number of ADC_CLK periods to sample the input voltage, the number of sampling periods can be changed using the SMP[2:0] bits in the ADC_SMPR1, ADC_SMPR2 and ADC_SMPR3 registers. Each channel can be sampled separately with a different time.

The total conversion time is calculated as follows:

T_{CONV} = Sampling time + 12.5 cycles

Example:

When ADCCLK = 14MHz, the sampling time is 3.5 cycles

T_{CONV} = 3.5 + 12.5 = 16 cycles = 1.14μs

16.3.7. Configurable resolutions

Fast conversions can be performed by reducing the ADC resolution. the RESSEL bit in the ADC_CR1 register is used to select the number of bits available in the data register. The minimum conversion time for each resolution is as follows, sample time + conversion time:

1) 12 bits: 3.5 + 12.5 = 16 ADCCLK cycles
2) 10 bits: 3.5 + 10.5 = 14 ADCCLK cycle
3) 8 bits: 3.5 + 8.5 = 12 ADCCLK cycle
4) 6 bits: 3.5 + 6.5 = 10 ADCCLK cycles

16.3.8. Single conversion mode

In single conversion mode, the ADC performs only one conversion. This mode can be started either by setting the ADON bit in the ADC_CR2 register (for regular channels only) or by external triggering (for regular or injected channels), when the CONT bit is 0.

Once the conversion of the selected channel is complete:

- If a rule channel is converted:
  - Conversion data is stored in the 16-bit ADC_DR register
  - EOC (end of conversion) flag is set
  - If EOCIE is set, an interrupt is generated
- If an injection channel is converted:
  - Conversion data is stored in the 16-bit ADC_JDRx register
  - JEOC (end of injection conversion) flag is set
  - -If the JEOCIE bit is set, an interrupt is generated.

Then the ADC stops.

16.3.9. Continuous conversion mode

In continuous conversion mode, another conversion is started as soon as the previous ADC conversion is completed. This mode can be started by an external trigger or by setting the ADON bit on the ADC_CR2 register, when the CONT bit is 1.

- After each conversion:
  - If a rule channel is converted:
  - Conversion data is stored in the 16-bit ADC_DR register
  - The EOC (end of conversion) flag is set
  - If EOCIE is set, an interrupt is generated.
- If an injection channel is converted:
  - EOC (end of conversion flag) flag set
- Conversion data is stored in the 16-bit ADC_JDRx register
- The JEOC (end of injection conversion) flag is set
- If the JEOCIE bit is set, an interrupt is generated.

16.3.10. Scan mode

This mode is used to scan a group of analogue channels.

The scan mode can be selected by setting the SCAN bit of the ADC_CR1 register. Once this bit is set, the ADC scans all channels selected by the ADC_SQRx register (for regular channels) or ADC_JSQR (for injection channels). A single conversion is performed on each channel in each group. At the end of each conversion, the next channel in the same group is automatically converted. If the CONT bit is set, the conversion does not stop at the last channel of the selected group, but continues again from the first channel of the selected group.

If the DMA bit is set, the DMA controller transfers the conversion data of the channels of the rule group to the SRAM. And the data for the injection channel conversion is always stored in the ADC_JDRx register.

16.3.11. Intermittent conversion mode

Rules group

This mode is activated by setting the DISCEN bit on the ADC_CR1 register. It can be used to perform a short sequence of n conversions (n<=8) which is part of the conversion sequence selected by the ADC_SQRx register. The value n is given by the DISCNUM[2:0] bits of the ADC_CR1 register.

An external trigger signal initiates the next round of n conversions described in the ADC_SQRx register until all conversions in this sequence have been completed. The total sequence length is defined by L[3:0] of the ADC_SQR1 register.

Example:

n = 3, channels being converted = 0, 1, 2, 3, 6, 7, 9, 10

First trigger: the converted sequence is 0, 1, 2

Second trigger: the sequence to be converted is 3, 6, 7

Third trigger: sequence of conversions is 9, 10 and EOC event is generated

Fourth trigger: sequence of conversions 0, 1, 2

Note: When converting a rule group in intermittent mode, the conversion sequence does not automatically start from the beginning at the end.

When all subgroups have been converted, the next trigger starts the conversion of the first subgroup. In the example above, the fourth trigger reconverts channels 0, 1 and 2 of the first subgroup.

Injection group

This mode is activated by setting the JDISCEN bit of the ADC_CR1 register. After an external trigger event, this mode converts the sequence selected in the ADC_JSQR register one by one in channel order.

An external trigger signal initiates the conversion of the next channel sequence selected by the ADC_JSQR register until all

of the sequence are completed. The total sequence length is defined by the JL[1:0] bits of the ADC_JSQR register.

Example:

n = 1, channel being converted = 1, 2, 3

First trigger: channel 1 is converted

Second trigger: channel 2 is converted

Third trigger: channel 3 is converted and generates EOC and JEOC events

Fourth trigger: channel 1 is converted

Note: 1 When all injection channel conversions are completed, the next trigger initiates the conversion of the 1st injection channel. In the above example, the fourth trigger reconverts the 1st injection channel 1.

2 Automatic injection and intermittent mode cannot be used at the same time.

16.3.12. Injecting channel management

The external trigger of the injection channel has a higher priority than the external trigger of the rule channel, i.e. the external trigger of the injection channel can interrupt a rule channel transition in progress. There are two types of interrupt for the injection channel: triggered injection and automatic injection.

16.3.12.1. Trigger Injection

If the JAUTO bit of the ADC_CR1 register is cleared, and the SCAN bit is set.

- The conversion of a set of regular channels is initiated using an external trigger or by setting the ADON bit of the ADC_CR2 register.
- If an external injection trigger is generated during a rule channel conversion, the current conversion is reset and the sequence of injected channels is converted in a single scan.
- The last interrupted rule group channel conversion is then resumed. If a rule event is generated during an injection transition, the injection transition is not interrupted, but the rule sequence will be executed after the injection sequence has finished.

Note: When using triggered injection conversions, it must be ensured that the interval between trigger events is longer than the injection sequence. For example, if the sequence length is 28 ADC clock cycles (i.e. 2 conversions with a 1.5 clock interval sample time), the minimum interval between triggers must be 29 ADC clock cycles.

16.3.12.2. Automatic injection

If the JAUTO bit is set, the injection group channels are automatically converted after the rule group channels. This can be used to convert up to 20 conversion sequences set in the ADC_SQRx and ADC_JSQR registers.

In this mode, external triggering of the injection channels must be disabled.

If the CONT bit is set in addition to the JAUTO bit, the conversion sequence from the rule channel to the injection channel is executed continuously.

Note: It is not possible to use both auto-inject and intermittent modes

16.3.13. 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.

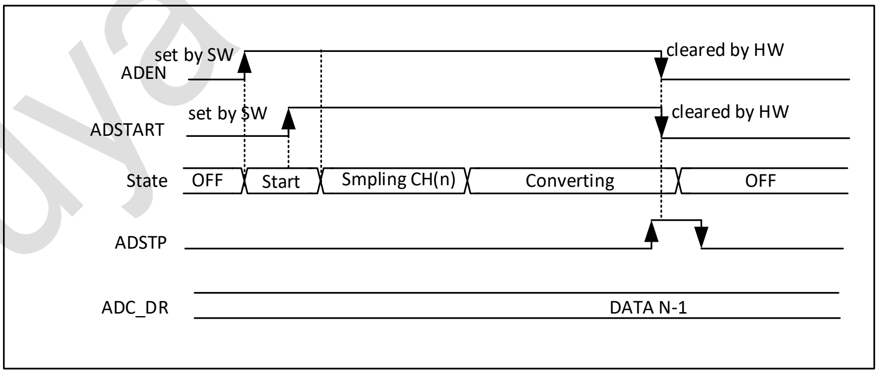


Figure 16-2 Stop timing

16.4. Watchdog simulation

The AWD analogue watchdog status bit is set to 1 if the analogue voltage converted by the ADC is below the lower threshold or above the upper threshold. these thresholds are set in the 12 least significant bits of the ADC_HTR and ADC_LTR registers. Interrupts are generated by setting the AWDIE bit in the ADC_CR1 register.

The thresholds are independent of the alignment selected for the ALIGN bit in the ADC_CR2 register. The threshold comparison is done before the alignment (before the injection channel is subtracted from the offset value).

By configuring the ADC_CR1 register, the analogue watchdog can act on 1 or more channels, as shown in the following table:

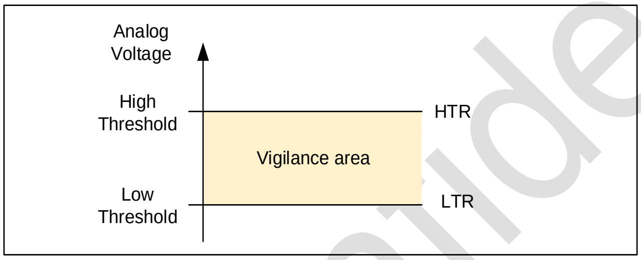


Figure 16-3 Analogue Watchdog Reserve

Table 16-2 Analogue watchdog channel selection

Analogue watchdog protection channel	ADC_CR1 register control bit		
	AWDSGL	AWDEN	JAWDEN
None	X	0	0
All injection channels	0	0	1
All rule channels	0	1	0
All injection and rule channels	0	1	1
Single injection channel	1	0	1
Single rule access	1	1	0
Single injection or rule channel	1	1	1

16.5. External trigger conversion

Conversions can be triggered by external events (e.g. timer capture, EXTI interrupt). If the EXTTRIG or JEXTTRIG control bits are set, an external event can trigger the conversion. the EXTSEL[2:0] and JEXTSEL2:0] control bits allow the application to select one of the 8 possible events that can trigger the sampling of the rule and injection groups.

Note: When an external trigger signal is selected for an ADC rule or injection conversion, only the rising edge can initiate the conversion.

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

Table 16-3 ADC for external triggering of rule channels

Trigger source	Type	EXTSEL[2:0]
CH1 output of timer 1	Internal signals for on-chip timers	000
CH2 output of timer 1		001
CH3 output of timer 1		010
TRGO output of timer 2		011
TRGO output of timer 3		TRGO output of timer 2
TRGO output of timer 15		101
EXTI Line 11	External pins	110
SWSTART	Software control bits	111

Table 16-4 ADC for external triggering of injection channels

Trigger source	Type	JEXTSEL[2:0]
TRGO output of timer 1	Internal signals for on-chip timers	000
CH4 output of timer 1		001
TRGO output of timer 2		010
CH1 output of timer 2		011
CH4 output of timer 3		100
TRGO output of timer 15		101
EXTI line 15 output	External pins	110
JSWSTART	Software control bits	111

16.6. Data alignment

At the end of each conversion (when the EOC event is generated) the resultant data of the conversion is stored in the 16-bit wide ADC_DR data register. the ALIGN bit in the ADC_CR2 register selects the alignment of the converted data storage. The data can be left or right aligned, e.g. the converted data value for the injection group channel has been subtracted from the offset defined in the

ADC_JOFRx register, so the result can be a negative value. The SEXT bit is the extended sign value.

For regular group channels, the offset value is not subtracted and therefore only 12 bits are valid.

An example of right alignment of data is as follows:

Injection group

SEXT	SEXT	SEXT	SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Rules group

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

An example of left alignment of data is as follows:

Injection group

SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0

Rules group

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

16.7. Data overload

The ADC Overshoot flag (OVER) is a buffer overshoot event. The rule group ADC overshoot occurs when the converted data is not read by the CPU or DMA in time and another conversion is already valid.

16.8. DMA requests

Because the value of a rule channel conversion is stored in a data-only register, DMA is required when converting multiple rule channels, which prevents the loss of data already stored in the ADC_DR register.

A DMA request is only generated at the end of a rule channel conversion and the converted data is transferred from the ADC_DR register to the user specified destination address.

16.9. Temperature sensor and internal reference voltage

The temperature sensor can be used to measure the temperature (TA) around the device. The temperature sensor is internally connected to the ADC1_IN16 channel, which converts the voltage output from the sensor to a digital value. The recommended sampling time for the analogue input of the temperature sensor is 17.1 us and the sensor can be placed in power-down mode when not in use. The output voltage of the temperature sensor varies linearly with temperature. Due to variations in the production process, the offset of the temperature profile can vary from chip to chip (up to 45°C). Internal temperature sensors are better suited to detecting temperature variations than to measuring absolute temperatures. If an accurate temperature measurement is required, an external temperature sensor should be used.

Note: The TSVREFE bit must be set to activate the conversion of the internal channels: ADC1_IN16 (temperature sensor) and ADC1_IN17 (VREFINT).

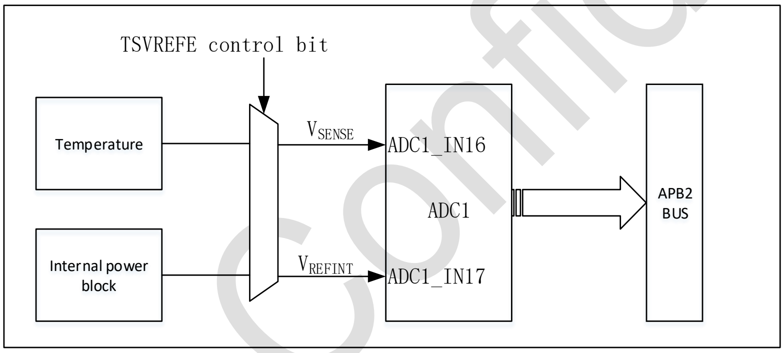


Figure 16-4 Block diagram of temperature sensor channels

Read temperature key features

- Temperature range supported: -40 °C to 125 °C
- Accuracy: ±1.5 °C

Temperature reading

To use the sensor, do the following:

1. select ADC1_IN16

2. Select a sample time that is greater than the minimum sample time specified in the datasheet.

3. wake up the temperature sensor from power-down mode by setting TSVREFE to position 1 in the ADC_CR2 register

4. start the ADC conversion by setting ADON to position 1 and enabling the external trigger

5. read the VSENSE data generated in the ADC data register

6. Calculate the temperature using the following formula:

Temperature（°C） = {(VSENSE - V25) / Avg_Slope} + 25

Where

V25 = VSENSE value at 25 °C

Avg_Slope = average slope of the temperature vs VSENSE curve (in mV/°C or μV/°C)

(For information on the actual values of V25 and Avg_Slope, see the Electrical Characteristics section of the data sheet.)

Note: The sensor requires a set-up time to wake up from power-down mode, after which the correct level of VSENSE is output. The ADC also requires a set-up time after power-up, so to reduce the delay, both ADON and TSVREFE should be set.

16.10. ADC interrputs

Interrupts can be generated at the end of rule and injection group conversions, when the analogue watchdog status bit is set, and when rule group conversion data is not read in time. They all have separate interrupt enable bits.

There are 2 other flags in the ADC_SR register, but they do not have interrupts associated with them:

- JSTRT (Initiation of Injection Group Channel Transformation)
- STRT (initiation of Rule Group Channel Transformation)

Table 16-5 ADC interrupts

Interrupt event	Event flag	Enable control bit
End of rule group conversion	EOC	EOCIE
End of injection group conversion	JEOC	JEOCIE
Analog watchdog status bits are set	AWD	AWDIE
Overflow sign	OVER	OVERIE

16.11. ADC registers

16.11.1. ADC Status Register (ADC_SR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVER	STRT	JSTRT	JEOC	EOC	AWD
										RC	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
5	OVER	RC	0	ADC Overload This bit is set by hardware when an overload occurs. When the EOC flag is set it indicates that a new conversion has been completed. 0: no overload has occurred (DMA or CPU has read the result of the last conversion) 1: Overload has occurred
4	STRT	RC_W0	0	Rule channel start status bit This bit is set by hardware at the start of a rule channel transition and is cleared by software. 0: Rule channel conversion has not started 1: Rule channel conversion has started
3	JSTRT	RC_W0	0	Injection channel start status bit This bit is set by hardware at the start of an injection channel group conversion and is cleared by software. 0: Injection channel conversion has not started 1: Injection channel conversion has started
2	JEOC	RC_W0	0	End of injection channel conversion status bit This bit is set by hardware at the end of conversion for all injected channel groups and is cleared by software. 0: Conversion not completed 1: Conversion complete
1	EOC	RC_W0	0	End of conversion status bit This bit is set by hardware at the end of (regular or injected) channel group conversion, cleared by software or cleared by reading ADC_DR. 0: conversion not completed 1: Conversion complete
0	AWD	RC_W0	0	Analogue watchdog flag bit This bit is set by hardware to 1 when the converted voltage value is outside the range defined by the ADC_LTR or below the ADC_HTR register and is cleared by software.

Bit	Name	R/W	Reset Value	Function
				0: No analogue watchdog event occurred 1: An analogue watchdog event has occurred

16.11.2. ADC Control Register 1 (ADC_CR1)

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	OVRIE	Res	AD-STP	Res	RES-SEL[1:0]		AWDEN	JAWDEN	Res	Res	Res	Res	Res	Res
		RW		R_W1		RW		RW	RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DISCNUM[2:0]			JDISCEN	DIS-CEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EO-CIE	AWDCH[4:0]				
RW			RW	RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	OVRIE	RW	0	OVER FLAG interrupt enable 0: Overload interrupt disabled 1: Allow overload interrupt
28	Reserved	-	-	Reserved
27	ADSTP	R_W1	0	ADC conversion stop enable. Software write 1 set to 1. Hardware set to 0 when ADC stop signal is received. 1: Stops ADC conversion. 0: Does not stop ADC conversion.
26	Reserved	-	-	Reserved
25:24	RESSEL[1:0]	RW	0	Resolution control bits. The resolution of the conversion can be selected by writing these bits in software. 00: 12 bits (15 ADCCLK cycle) 01: 10 bits (13 ADCCLK cycle) 10: 8 bits (11 ADCCLK cycle) 11: 6 bits (9 ADCCLK cycle)
23	AWDEN	RW	0	Rule channel analog watchdog enable. Enables the analog watchdog on the rule channel. This bit is set to 1 by a software write 1 and to 0 by a write 0. 0: disables the analogue watchdog on the rule channel

Bit	Name	R/W	Reset Value	Function
				1: Use analog watchdog on rule channel
22	JAWDEN	RW	0	Inject channel analog watchdog enable. Enables the analogue watchdog on the injection channel. This bit is set to 1 by a software write 1 and to 0 by a write 0. 0: disables the analogue watchdog on the injection channel 1: Use analog watchdog on the injection channel
21:16	Reserved	-	-	Reserved
15:13	DISCNUM[2:0]	RW	0	Intermittent Mode Channel Count. In intermittent mode, the number of channels converted in the rule channel group after an external trigger is received. Software write operations set these bits. 000: 1 channel 001: 2 channels ...... 111: 8 channels
12	JDISCEN	RW	0	Inject channel intermittent mode enable. This bit is set to 1 by a software write 1 and 0 by a write 0 to enable or disable intermittent mode on the injection channel group. 0: Disable intermittent mode on the injection channel group 1: Intermittent mode is used on the injection channel group
11	DISCEN	RW	0	Intermittent mode enable for rule channels This bit is set to 1 by a software write 1 and 0 by a write 0 to enable or disable intermittent mode on the rule channel group 0: Disable intermittent mode on the rule channel group 1: Intermittent mode is used on the rule channel group
10	JAUTO	RW	0	Auto-Inject Enable This bit is used by software write 1 to set 1 and write 0 to set 0 to enable or disable automatic injection channel group conversion after the end of a rule channel group conversion 0: turns off automatic injection channel group conversion 1: Enables automatic injection channel group conversion
9	AWDSGL	RW	0	Single channel watchdog enable.

Bit	Name	R/W	Reset Value	Function
				This bit is used by software write 1 to set 1 and write 0 to set 0 to enable or disable the analog watchdog on the channel defined by AWDCH[4:0]. 0: use analogue watchdog on all channels 1: use analogue watchdog on a single channel
8	SCAN	RW	0	Scan mode enable This bit is set to 1 by a software write 1 and 0 by a software write 0 to turn scan mode on or off. In scan mode, the channels selected by the ADC_SQRx or ADC_JSQRx registers are converted. 0: turn off scan mode 1: Use scan mode
7	JEOCIE	RW	0	Injection channel conversion end interrupt enable This bit is set to 1 by a software write 1 and 0 by a write 0. When enabled, this bit generates an interrupt request when JEOC is active. 0: JEOC interrupt disabled 1: JEOC interrupt is allowed.
6	AWDIE	RW	0	Analogue watchdog interrupt enable This bit is enabled by software write 1 to set 1 and write 0 to set 0. When enabled, this bit generates an interrupt request when AWD is active. 0: Disable the analogue watchdog interrupt 1: Allow the analogue watchdog interrupt
5	EOCIE	RW	0	End of rule channel conversion interrupt enable This bit is enabled by software write 1 to set 1 and write 0 to set 0. When enabled, this bit generates an interrupt request when the EOC is active. 0: EOC interrupt disabled 1: EOC interrupt is allowed.
4:0	AWDCH[4:0]	RW	0	Analogue watchdog channel select bit This bit is set by a software write and is used to select the input channel for the analogue watchdog. 00000: ADC analogue input channel 0 00001: ADC analogue input channel 1 ...... 01111: ADC Analogue Input Channel 15 10000: TS_VIN input

Bit	Name	R/W	Reset Value	Function
				10001: VREFINT input 10010: VCCA/3 10011: DAC1_VIN 10100 DAC2_VIN 10101: OPA1_VIN 10110: OPA2_VIN 10111: OPA3_VIN All other values are reserved.

16.11.3. ADC Control Register (ADC_CR2)

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	VREF-BUFF_SEL		VREFBUFFERE	Res	TSVREFE	SWSTART	JSWSTART	EX-TTRIG	EXTSEL[2:0]			Res
				RW	RW	RW		RW	R_W1	R_W1	RW	RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JEXTTRIG	JEXTSEL[2:0]			ALIGN	Res	Res	DMA	Res	Res	Res	Res	RSTCAL	CAL	CONT	ADON
RW	RW	RW	RW	RW			RW					R_W1	R_W1	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:26	VREFBUFF_SEL	RW	0	VREFBUF output voltage select 00: 1.5 V 01: 2.048 V 10: 2.5 V 11: reserved
25	VREF BUFFERE	RW	0	Enable VerfBuffer Software write 0 set to 0, write 1 set to 1 0: disable VerfBuffer 1: enable VerfBuffer
24	Reserved	-	-	-
23	TSVREFE	RW	0	Temperature sensor and VREFINT enable Used to enable or disable the temperature sensor and VREFINT channels by software write 1 to set 1 and write 0 to set 0. Enabling is possible in ADC1. 0: disables the temperature sensor and VREFINT

Bit	Name	R/W	Reset Value	Function
				1: Enables the temperature sensor and VREFINT
22	SWSTART	R_W1	0	Start conversion rule channel enable Start conversion by software write 1 to set 1, immediately after start conversion is cleared by hardware to set 0. 0: reset state 1: Start of conversion rule channel
21	JSWSTART	R_W1	0	Start conversion injection channel enable Start conversion by software write 1 to set 1, cleared by hardware to set 0 immediately after starting conversion. 0: reset state 1: Start conversion of the injection channel
20	EXTTRIG	RW	0	Rule channel external trigger enable This bit is cleared by software write 1 to set 1 and write 0 to set 0. It is used to enable or disable external trigger signals that can initiate rule channel group transitions. 0: disables external triggering of the rule channel 1: Enables external triggering of the rule channel
19:17	EXTSEL[2:0]	RW	0	Rule channel external trigger event selection bit. Selects the external event that initiates the conversion of the rule channel group ADC1,ADC2 external trigger events are as follows: 000: CH1 event for timer 1 001: CH2 event for timer 1 010: CH3 event for timer 1 011: CH2 event of timer 2 100: TRGO event for timer 3 101: Trigger timer15_TRGO event 110: EXTI11 111: SWSTART
16	Reserved	-	-	-
15	JEXTTRIG	RW	0	External trigger enable for the injection channel 0: Inject channel external trigger disable 1: Enable injection channel external trigger
14:12	JEXTSEL[2:0]	RW	0	Injection of channel group external trigger event selection bits The external trigger events for ADC1 and ADC2 are as follows 000: TRGO event for timer 1

Bit	Name	R/W	Reset Value	Function
				001: CH4 event for timer 1 010: TRGO event for timer 2 011: CH1 event for timer 2 100: TRGO event for timer 3 101: TRGO event for timer 15 110: EXTI15 111: JSWSTART
11	ALIGN	RW	0	Data alignment control bit This bit is cleared by a software write 1 to set 1 and a write 0 to set 0. 0: right-aligned 1: Left aligned
10:9	Reserved	-	-	-
8	DMA	RW	0	DMA enable bit This bit is cleared by software write 1 to set 1 and write 0 to set 0. 0: DMA mode disabled 1: Enables DMA mode
7:4	Reserved	-	-	-
3	RSTCAL	RES	0	Calibration reset enable bit This bit is set to 1 by a software write 1 and cleared to 0 by hardware and is cleared after the calibration register has been initialized (i.e. after RSTCAL has been set to 1). 0: Calibration register is initialised 1: Calibration register initialised Note: When a conversion is in progress, clear-ing the calibration register requires an additional cycle if RSTCAL is set.
2	CAL	R_W1	0	Calibration enable This bit is set to 1 by a software write 1 to start calibration and is cleared by hardware on cali-bration failure or calibration success. When ADON is invalid and SWSTART, JSWSTART is invalid; software initiates calibration. 0: Calibration complete 1: Enables calibration
1	CONT	RW	0	Continuous conversion enable This bit is cleared by software write 1 to set 1 and write 0 to set 0. If this bit is set, conversion will be continuous until this bit is cleared. 0: Single conversion mode 1: Continuous conversion mode
0	ADON	RW	0	On/Off A/D converter

Bit	Name	R/W	Reset Value	Function
				ADC converter operating enable, when set to 1 the ADC converter wakes up, there is a delay tSTAB between the converter powering up and the start of conversion. when set to 0 the ADC converter is in power down mode. 0: ADC conversion is disabled and the ADC converter enters power-down mode 1: Enables the ADC converter. Note: If SWSTART,JSWSTART is changed in this register together with ADON, the conversion is not triggered. This is to prevent wrong conversions from being triggered.

16.11.4. ADC Sample Time Register 1 (ADC_SMPR1)

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				SMP23[2:0]			SMP22[2:0]			SMP21[2:0]			SMP20[2:0]		
				RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	SMPx[2:0]	RW	0	Selecting the sampling time for channel x These bits are set by software to select the sample time for each channel independently. The channel selection bits must remain unchanged during the sampling cycle. 000: 3.5 cycles 100: 28.5 cycles 001: 5.5 cycles 101: 41.5 cycles 010: 7.5 cycles 110: 71.5 cycles 011: 13.5 cycles 111: 239.5 cycles

16.11.5. ADC Sample Time Register 2 (ADC_SMPR2)

Address offset:0x10

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
		RW			RW			RW			RW			RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15[0]	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
RW	RW			RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	0	Selecting the sampling time for channel x These bits are set by software to select the sample time for each channel independently. The channel selection bits must remain un-changed during the sampling cycle. 000: 3.5 cycles 100: 28.5 cycles 001: 5.5 cycles 101: 41.5 cycles 010: 7.5 cycles 110: 71.5 cycles 011: 13.5 cycles 111: 239.5 cycles

16.11.6. ADC Sample Time Register 3 (ADC_SMPR3)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		RW			RW			RW			RW			RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP5[0]	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
RW	RW			RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	0	Selecting the sampling time for channel x These bits are set by software to select the sample time for each channel independently. The channel selection bits must remain un-changed during the sampling cycle. 000: 3.5 cycles 100: 28.5 cycles 001: 5.5 cycles 101: 41.5 cycles 010: 7.5 cycles 110: 71.5 cycles 011: 13.5 cycles 111: 239.5 cycles

16.11.7. ADC Injection Channel Data Offset Register x (ADC_JOFRx) (x=1..4)

Address offset:0x18-0X24

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				JOFFSETx[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	JOFFSETx[11:0]	RW	0	Injection of the xth conversion data offset of the channel The software configures the value of these bits. These bits define the value used to subtract from the original conversion data when the conversion is injected into the channel. The final conversion result can be read out in the ADC_JDRx register.

16.11.8. ADC Watchdog High Threshold Register (ADC_HTR)

Address offset:0x28

Reset value:0x0000 0FFF

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				HT[11:0]											
				RW											

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
11:0	HT[11:0]	RW	0xFFF	Analogue Watchdog High Threshold The software configures the values of these bits. These bits define the high threshold limit for the analogue watchdog.

16.11.9. ADC Watchdog Low Threshold Register (ADC_LRT)

Address offset:0x2C

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**
Reserved				HT[11:0]											
				RW											

Bit	Name	R/W	Reset Value	Function
31：11	Reserved	-	-	Reserved
11:0	LT[11:0]	RW	0x000	Analogue Watchdog Low Threshold The software configures the values of these bits. These bits define the threshold low limit for the analogue watchdog.

16.11.10. ADC Rule Sequence Register 1 (ADC_SQR1)

Address offset:0x30

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								L[3:0]				SQ16[4:1]			
								RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ16[0]	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23:20	L[3:0]	RW	0	Rule channel sequence length The software configures the value of these bits. These bits define the number of channels in the regular channel conversion sequence. 0000: 1 conversion 0001: 2 conversions ...... 1111: 16 conversions
19:15	SQ16[4:0]	RW	0	The software configures the value of these bits. The 16th conversion in the rule sequence, these bits define the number (0 to 23) of the 16th conversion channel in the conversion sequence.
14:10	SQ15[4:0]	RW	0	The software configures the value of these bits. The 15th conversion in the rule sequence, these bits define the number (0 to 23) of the 15th conversion channel in the conversion sequence.
9:5	SQ14[4:0]	RW	0	The software configures the value of these bits. The 14th conversion in the rule sequence, these bits define the number (0 to 23) of the 14th conversion channel in the conversion sequence.
4:0	SQ13[4:0]	RW	0	The software configures the value of these bits. The 13th conversion in the rule sequence, these bits define the number (0 to 23) of the 13th conversion channel in the conversion sequence.

16.11.11. ADC Rule Sequence Register 2 (ADC_SQR2)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SQ12[4:0]					SQ11[4:0]					SQ10[4:1]			
		RW					RW					RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ10[0]	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:25	SQ12[4:0]	RW	0	The software configures the value of these bits. The 12th conversion in the rule sequence, these bits define the number (0 to 23) of the 12th conversion channel in the conversion sequence.
24:20	SQ11[4:0]	RW	0	The software configures the value of these bits. The 11th conversion in the rule sequence, these bits define the number (0 to 23) of the 11th conversion channel in the conversion sequence.
19:15	SQ10[4:0]	RW	0	The software configures the value of these bits. The 10th conversion in the rule sequence, these bits define the number (0 to 23) of the 10th conversion channel in the conversion sequence.
14:10	SQ9[4:0]	RW	0	The software configures the value of these bits. The 9th conversion in the rule sequence, these bits define the number (0 to 23) of the 9th conversion channel in the conversion sequence.
9:5	SQ8[4:0]	RW	0	The software configures the value of these bits. The 8th conversion in the rule sequence, these bits define the number of the 8th conversion channel in the conversion sequence (0 to 23).
4:0	SQ7[4:0]	RW	0	The software configures the value of these bits. The 7th conversion in the rule sequence, these bits define the number of the 7th conversion channel in the conversion sequence (0 to 23).

16.11.12. ADC Rule Sequence Register 3 (ADC_SQR3)

Address offset:0x38

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			

		RW					RW					RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ4[0]	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:25	SQ6[4:0]	RW	0	The software configures the value of these bits. The 6th conversion in the rule sequence, these bits define the number of the 6th conversion channel in the conversion sequence (0 to 23).
24:20	SQ5[4:0]	RW	0	The software configures the value of these bits. The 5th conversion in the rule sequence, these bits define the number of the 5th conversion channel in the conversion sequence (0 to 23).
19:15	SQ4[4:0]	RW	0	The software configures the value of these bits. The 4th conversion in the rule sequence, these bits define the number of the 4th conversion channel in the conversion sequence (0 to 23).
14:10	SQ3[4:0]	RW	0	The software configures the value of these bits. The 3rd conversion in the rule sequence, these bits define the number of the 3rd conversion channel in the conversion sequence (0 to 23).
9:5	SQ2[4:0]	RW	0	The software configures the value of these bits. The 2nd conversion in the rule sequence, these bits define the number of the 2nd conversion channel in the conversion sequence (0 to 23).
4:0	SQ1[4:0]	RW	0	The software configures the value of these bits. The 1st conversion in the rule sequence, these bits define the number of the 1st conversion channel in the conversion sequence (0 to 23).

16.11.13. ADC Injection Sequence Register (ADC_JSQR)

Address offset:0x3C

Reset value:0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res								JL[3:0]				JSQ4[4:1]			
								RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JSQ4[0]	JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23:20	JL[3:0]	RW	0	Inject the channel sequence length The software configures the value of these bits. These bits define the number of channels in the injected channel conversion sequence. 00: 1 conversion 01: 2 transitions 10: 3 conversions 11: 4 transitions
19:15	JSQ4[4:0]	RW	0	Injection of the 4th transition in the sequence The software configures the value of these bits. These bits define the number of the 4th conversion channel in the conversion sequence (0 to 23). Note: Unlike a regular conversion sequence, if the length of JL[1:0] is less than 4, the sequence order of conversions starts with (4-JL). For example: ADC_JSQR[21:0] = 10 00011 00011 00111 00010, means that the scan conversion will be done in the following channel Sequential conversion: 7, 3, 3, instead of 2, 7, 3
14:10	JSQ3[4:0]	RW	0	The software configures the value of these bits. Injection of the 3rd conversion in the sequence
9:5	JSQ2[4:0]	RW	0	The software configures the value of these bits. Injection of the 2nd conversion in the sequence
4:0	JSQ1[4:0]	RW	0	The software configures the value of these bits. Injection of the 1st transition in the sequence

16.11.14. ADC injection data register x (ADC_JDRx) (x= 1..4)

Address offset:0x40-4C

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**
JDATA[15:0]															
R															

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
15:0	JDATA[15:0]	R	0	Injection of channel conversion results The software can read the value of these bits. These bits are read-only and contain the conversion result of the injected channel. Data is left or right aligned

16.11.15. ADC Rule Data Register (ADC_DR)

Address offset:0x40-4C

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**
DATA[15:0]															
R															

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
15:0	DATA[15:0]	R	0	Rule channel conversion results The software can read the value of these bits. These bits are read-only and contain the result of the conversion of the rule channel. Data is left or right aligned

16.11.16. 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
CAL ON.	CAP SUC	OFF SUC	Res												
RO	RC_W1	RC_W1													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAL SET	CAL BYP	CALSMP[1:0]		CAL SEL	Reserved										
R_W1	R_W1	RW		RW											

Bit	Name	R/W	Reset Value	Function
31	CALON	RO	0	Calibration flag to mark that ADC calibration is in progress. 1: ADC calibration is in progress 0: ADC calibration has been completed or ADC calibration has not been initiated
30	CAPSUC	RC_W1	0	Capacitance calibration status bit.

Bit	Name	R/W	Reset Value	Function
				Indicates whether the ADC capacitance calibration was successful. Hardware set to 1; software write 1 set to 0; CALON=0, CALSEL=0,CALSUC=1: Invalid state CALON=0, CALSEL=0, CALSUC=0: CAPs not calibrated CALON=0, CALSEL=1, CALSUC =1: ADC CAPs calibrated successfully CALON=0, CALSEL=1, CALSUC =0: Calibration of ADC CAPs failed
29	OFFSUC	RC_W1	0	Offset calibration status bit. Indicates whether the ADC offset calibration was successful. Hardware set to 1; software write 1 set to 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	RC_W1	0	Calibration factor selection. Software write 1 set to 1 when CAL is 0. Hardware set to 0 when CAL is valid or injection/rule channel SWSTART, JWSTART is valid. 1: Set CAL_CXIN data as the 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	R_W1	0	Calibration factor bypass. Software write 1 set to 1 when CAL is 0. Hardware set to 0 when CAL is valid or injection/rule channel SWSTART, JWSTART is valid. 1: Calibration result is a reset value 0: Calibration result is self-calibration result or calibration factor input value
13:12	CALSMP[1:0]	RW	0	Calibration sample time seletion Configure the number of clock cycles for the calibration sample phase according to the following information: 00: 1 ADC clock period

Bit	Name	R/W	Reset Value	Function
				01: 2 ADC clock periods 10: 4 ADC clock periods 11: 8 ADC clock periods The longer the period of the SMP configured during calibration, the more accurate the calibration results, but this configuration can lead to longer calibration periods
11	CALSEL	RW	0	Calibration content selection bit for selecting the content to be calibrated 1: Calibration of OFFSET and linearity 0: Calibration of OFFSET only
10:0	Reserved	-	-	Reserved

16.11.17. ADC register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	ADC_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OVER	STRT	JSTRT	JEOC	EOC	AWD
	Reset value																											0	0	0	0	0	0
0x04	ADC_CR1	Res.	Res.	OVERIE	Res.	ADSTP	Res.	RESSEL[1:0]		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.	DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
	Reset value			0		0		0		0	0							0			0	0	0	0	0	0	0	0	0				
0x08	ADC_CR2	Res.	Res.	Res.	Res.	Verfbuff_sel.		Verfbuffen.	Res.	TSVREFE	SWSTART	JSWSTART	EXTTRIG	EXTSEL[2:0]			Res.	JEXTTRIG	JEXTSEL[2:0]			ALIGN	Res.	Res.	DMA	Res.	Res.	Res.	Res.	RSTCAL	CAL	CONT	ADON
	Reset value					0		0		0	0	0	0	0				0	0			0			0					0	0	0	0
0x0C	ADC_S	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMP23[2:0]			SMP22[2:0]			SMP21[2:0]			SMP20[2:0]		

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR1																																
	Reset value																					0			0			0			0		
0x10	ADC_SMPR2	Res.	Res.	SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:0]			SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
	Reset value			0			0			0			0			0			0			0			0			0			0		
0x14	ADC_SMPR3	Res.	Res.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:0]			SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
	Reset value			0			0			0			0			0			0			0			0			0			0		
0x18-0x24	ADC_JOFRx	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSETx[11:0]											
	Reset value																					0											
0x28	ADC_HTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT[11:0]											
	Reset value																					0xFFF											

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	ADC_LRT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT[11:0]											
	Reset value																					0											
0x30	ADC_SQR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	L[3:0]				SQ16[4:0]					SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
	Reset value									0				0					0					0					0				
0x34	ADC_SQR2	Res.	Res.	SQ12[4:0]					SQ11[4:0]					SQ10[4:0]					SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
	Reset value			0					0					0					0					0					0				
0x38	ADC_SQR3	Res.	Res.	SQ6[4:0]					SQ5[4:0]					SQ4[4:0]					SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
	Reset value			0					0					0					0					0					0				
0x3C	ADC_JSQR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JL[3:0]				JSQ4[4:0]					JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
	Reset									0				0					0					0					0				

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x40-0x4C	ADC_JDRx	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset value																	0															
0x50	ADC_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATA[15:0]															
	Reset value																	0															
0x54	ADC_CCSR	CALON.	CAPSUC.	OFFSUC.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALSET.	CALBYP.	CALSMP		CALSEL.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0														0	0	0		0											

17. Liquid Crystal Controller (LCD)

17.1. Introduction

The LCD controller is a digital controller/driver for monochrome passive liquid crystal displays (LCDs) with up to 8 common terminals (COM) and 40 zone terminals (SEG) to drive 160 (4x40) or 288 (8x36) LCD image elements. The exact number of terminals depends on the device pins described in the datasheet.The LCD consists of a number of zones (pixels or full symbols) which can be lit or switched off. Each segment contains a layer of liquid crystal molecules aligned between two electrodes. When a voltage above the threshold voltage is applied to the liquid crystal, the corresponding zone is visible. The zone voltage must be AC to avoid electrophoretic effects in the liquid crystal (which would affect the display). Afterwards, waveforms must be generated at the ends of the zones to avoid DC.

Glossary

Liquid Crystal (LCD): Passive display panel with terminals leading directly to the zones.

Common (COM): Electrical connection terminals to multiple zones.

Bias (BIAS): The voltage level used to drive the LCD, defined as 1/(the number of voltage levels to drive the LCD display - 1).

ZONE (SEG): The smallest visible unit (the smallest constituent element, line or dot, on the LCD display).

Duty cycle (DUTY): a number defined as 1/(number of common terminals on the LCD display).

Frame: One cycle of the waveform written to the zone.

Frame rate: The number of frames per second, i.e. the number of times the LCD zone is excited per second.

17.2. LCD main features

- Highly flexible frame rate control.
- Static, 1/2, 1/3, 1/4, 1/6 and 1/8 duty cycle support.
- Support for 1/2 and 1/3 bias voltage.

- LCD data RAM with up to 16 registers.
- Contrast ratio of LCD can be configured via software.
- 2 types of drive waveform generation
  - internal resistor divider, external resistor divider.
  - The power consumption of the internal resistor divider method can be configured via software to match the capacitive charge required by the LCD panel.
- Low power mode support: LCD controller can be displayed in run, sleep and stop modes.
- Configurable frame interrupt.
- Support for LCD blink function and configurable blink frequency
- Unused LCD zones and common pins can be configured as digital or analogue functions.

17.3. LCD block diagram

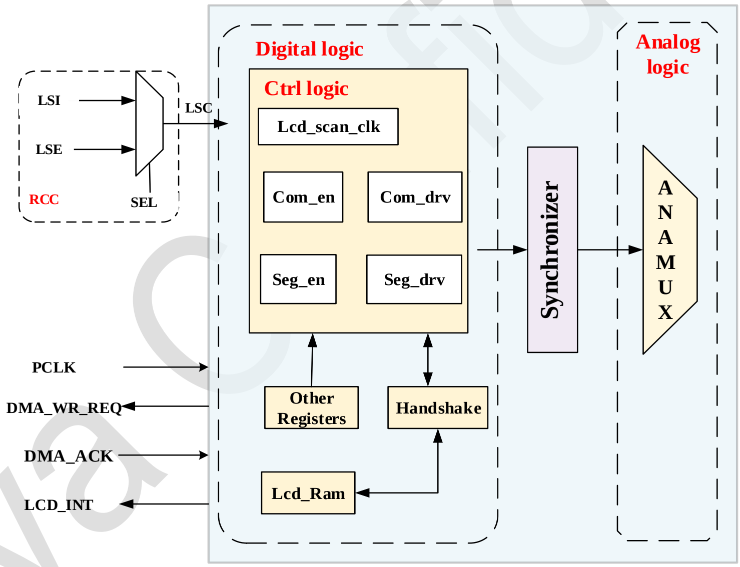


Figure 17-1 LCD block diagram

17.4. LCD clock

The LCD clock can be selected as LSI or LSE, and the input clock can be determined by LSCOSEL and LSCOEN in the RCC_BDCR register.

17.5. LCD drive waveform

The LCD supports 5 duty cycle drive waveforms: static, 1/2, 1/3, 1/4, 1/6 and 1/8, which are set by LCD_CR0. The LCD supports 2 types of Bias drive waveforms: 1/2, 1/3, set by LCD_CR0. The recommended combinations are shown in the table below:

Table 17-1 LCD supported drive waveforms

	1/2 Duty	1/3 Duty	1/4 Duty	1/6 Duty	1/8 Duty
1/2 Bias	✓	✓	Not recommended	Not recommended	Not recommended
1/3 Bias	Not recommended	Not recommended	✓	✓	✓

The drive waveforms in each mode are shown below:

17.5.1. Static drive waveforms

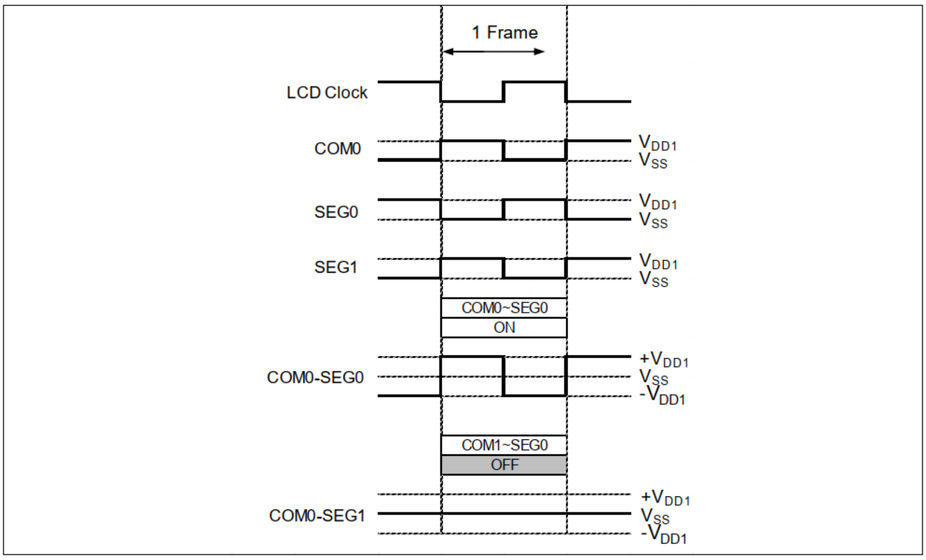


Figure 17-2 Static drive waveforms

17.5.2. 1/2Duty 1/2Bias Drive waveforms

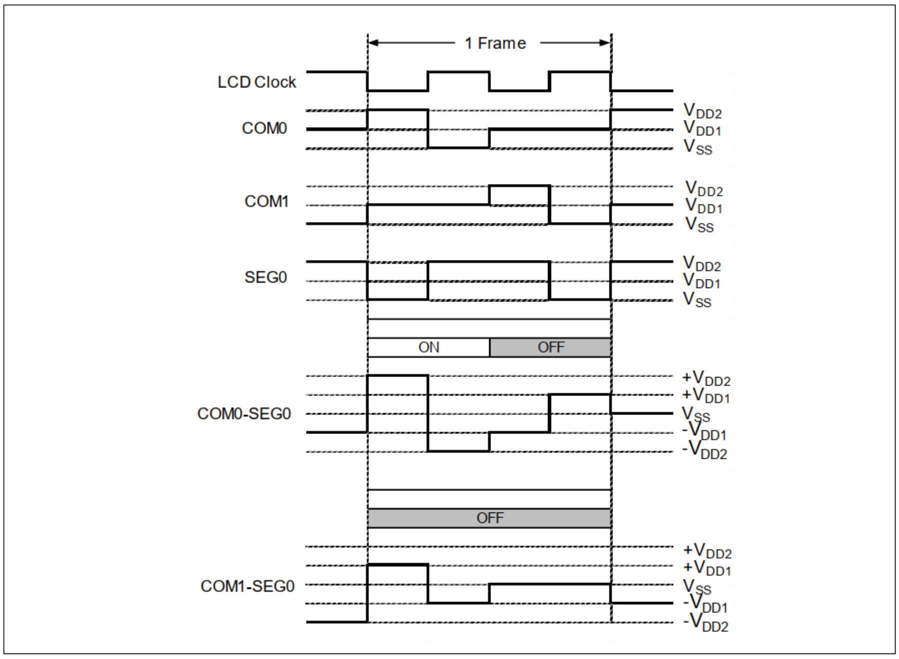


Figure 17-3 1/2Duty 1/2Bias Drive waveforms

17.5.3. 1/2Duty 1/3Bias Drive waveforms

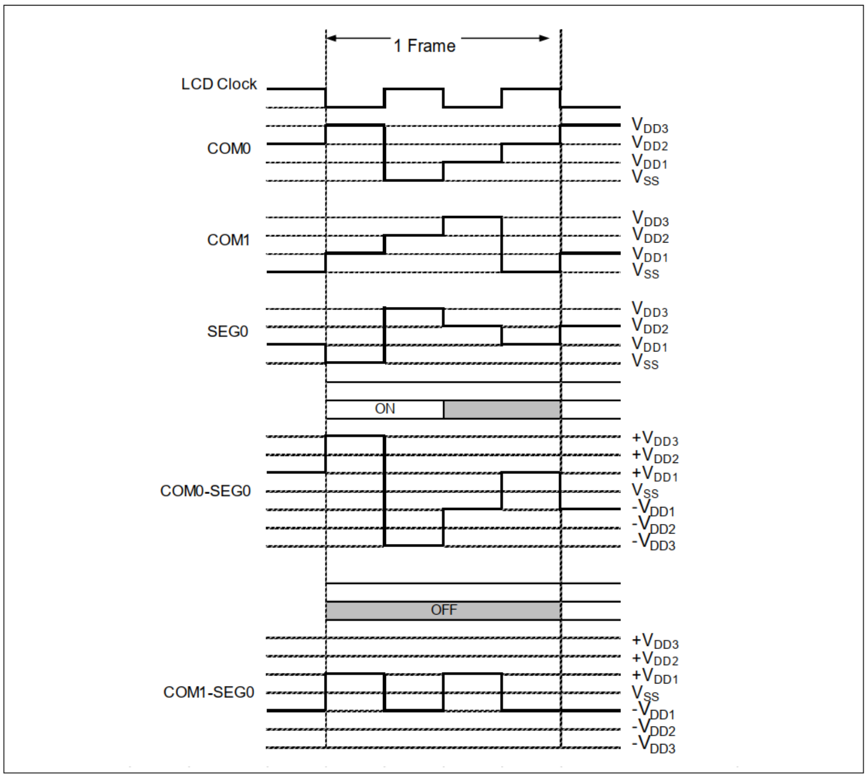


Figure 17-4 1/2Duty 1/3Bias Drive waveforms

17.5.4. 1/3Duty 1/2Bias Drive waveforms

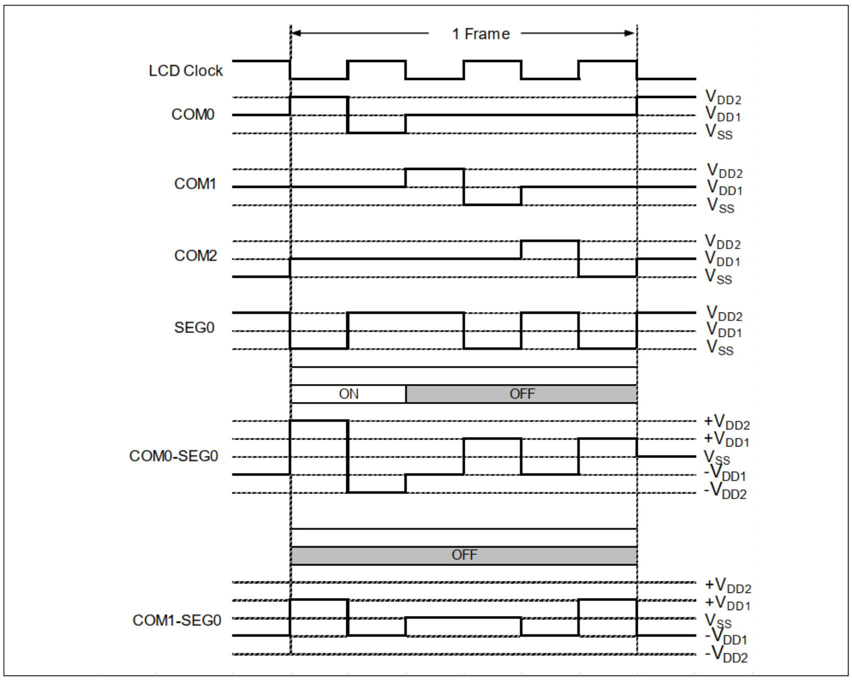


Figure 17-5 1/3Duty 1/2Bias Drive waveforms

17.5.5. 1/3Buty 1/3Bias Drive waveforms

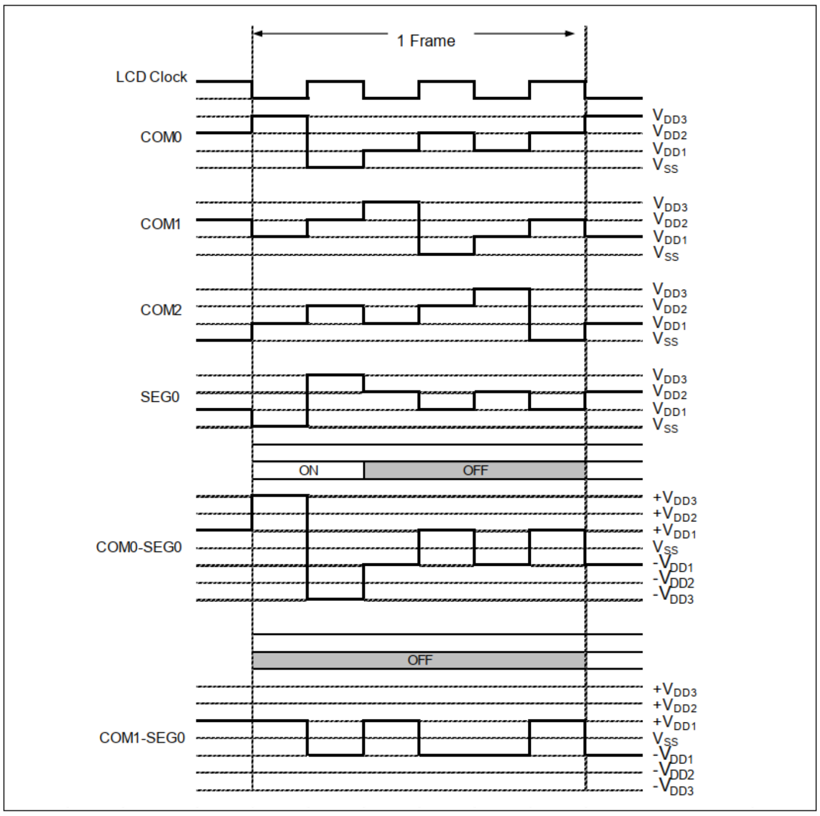


Figure 17-6 1/3Duty 1/3Bias Drive waveforms

17.5.6. 1/4Duty 1/2Bias Drive waveforms

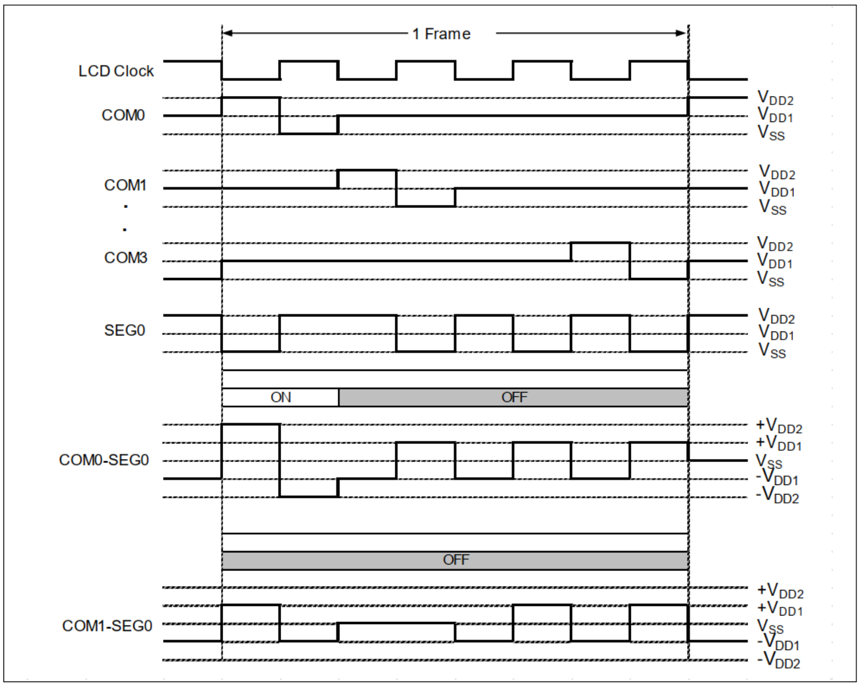


Figure 17-7 1/4Duty 1/2Bias Drive waveforms

17.5.7. 1/4Duty 1/3Bias Drive waveforms

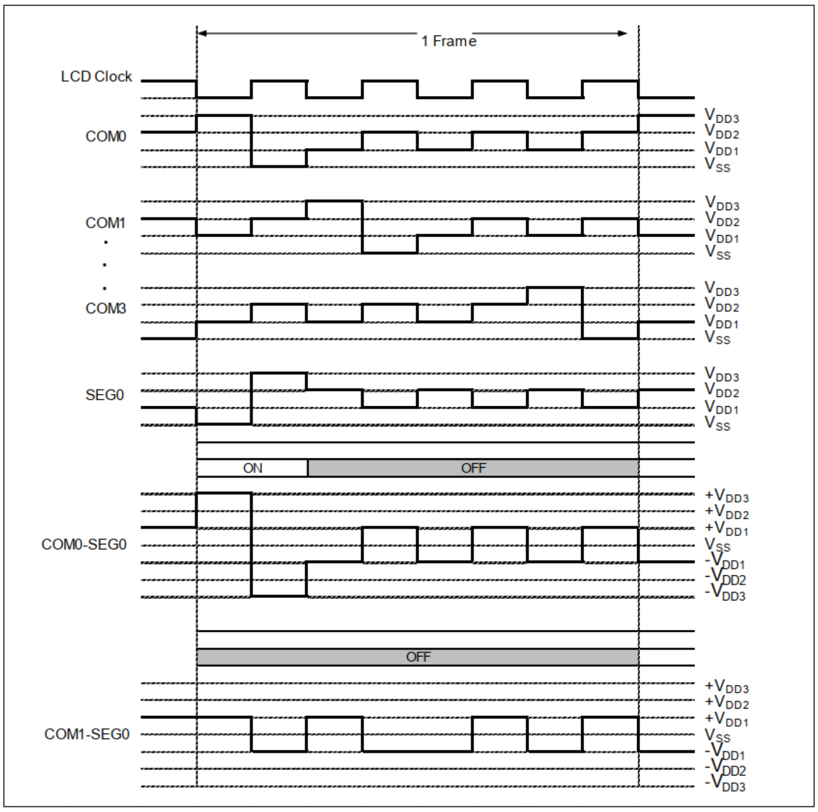


Figure 17-8 1/4Duty 1/3Bias Drive waveforms

17.5.8. 1/6Duty 1/3Bias Drive waveforms

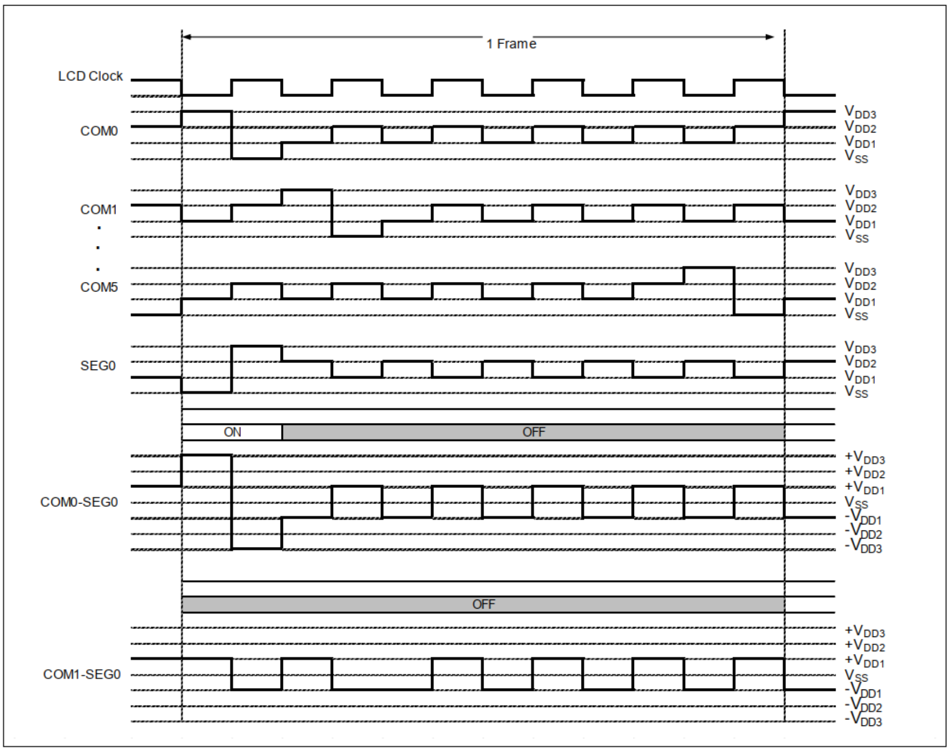


Figure 17-9 1/6Duty 1/3Bias Drive waveforms

17.5.9. 1/8Duty 1/3Bias Drive waveforms

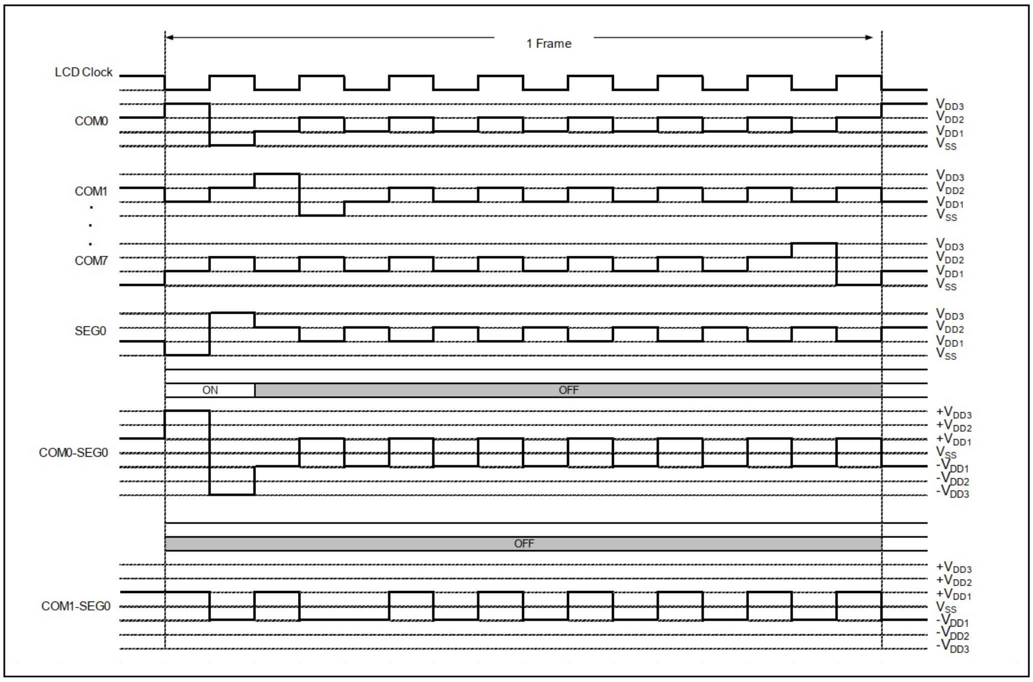


Figure 17-10 1/8Duty 1/3Bias Drive waveforms

17.6. LCD Bias Generation Circuit

The LCD's Bias voltage has 2 sources: internal resistive divider and external resistive divider. When internal resistor divider is selected, the chip will automatically switch the internal circuit to generate the voltage in accordance with Bias and Duty. When external resistor divider is selected, it requires the user to build the relevant circuitry on the peripheral pins of the chip.

17.6.1. Internal resistance mode

Internal resistor mode VLCDH, VLCD1~VLCD3 can be used as LCD SEG output or IO port.

In internal resistor mode, the drive voltage of the LCD is controlled by CR0.Contrast, as shown in the table below:

Table 17-2 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

CR0.Contrast	VLCD(1/3 bias)	VLCD(1/2 bias)
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

17.6.2. External resistance mode

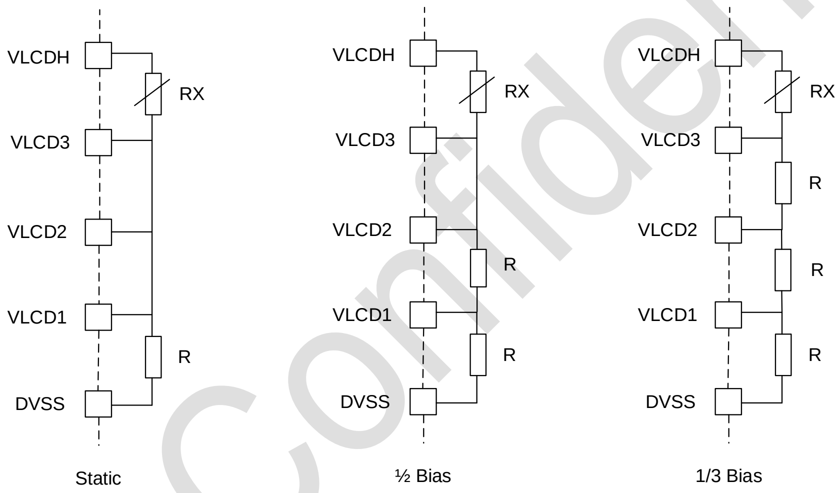


Figure 17-11 External resistance mode

Note:

- Rx is an adjustable resistor for adjusting the contrast of the LCD display.
- Select the appropriate resistor R according to the LCD screen being used.

17.7. DMA

The LCD supports both software and hardware triggered DMA data transfer, which automatically moves the content to be displayed from RAM or ROM to the LCD display RAM. The hardware trigger uses the frame interrupt signal. The DMA channel number used by the LCD is [8].

DMA data transfer configuration flow:

- enable DMA
- select LCD DMA

- Set the transfer type, transfer length and transfer method
- set the source start address, target start address
- Set the incremental method of source address and target address
- Enable DMA interrupt as required
- Enable LCD DMA trigger

17.8. Interruptions

When the LCD setting is active, the LCD interrupt can be configured to generate an interrupt for the number of frames.

17.9. LCD display mode

The LCD supports two display modes. One uses COM as the display unit, with all COM segments of the same SEG in the same byte (mode 0). The other is where different SEGs of the same COM are in the same byte (mode 1).

Selecting the appropriate display mode according to the LCD panel simplifies the operation of the programme.

17.9.1. LCD display mode 1 (MODE = 1)

1/8 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7
																													SEG35	SEG34	SEG33	SEG32	
																								COM0									LCDRAM8
																								COM1									LCDRAM9
																								COM2									LCDRAMA
																								COM3									LCDRAMB
																								COM4									LCDRAMC
																								COM5									LCDRAMD
																								COM6									LCDRAME
																								COM7									LCDRAMF

Figure 17-12 1/8 Duty LCD display mode 1

1/6 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
																											SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																								COM0									LCDRAM8
																								COM1									LCDRAM9
																								COM2									LCDRAMA
																								COM3									LCDRAMB
																								COM4									LCDRAMC
																								COM5									LCDRAMD
																																	LCDRAME
																																	LCDRAMF

Figure 17-13 1/6 Duty LCD display mode 1

1/4 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
																																	LCDRAM4
																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
																									SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																								COM0									LCDRAM8
																								COM1									LCDRAM9
																								COM2									LCDRAMA
																								COM3									LCDRAMB
																																	LCDRAMC
																																	LCDRAMD
																																	LCDRAME
																																	LCDRAMF

Figure 17-14 1/4 Duty LCD display mode 1

1/3 Duty 1/2 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7
																									SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																								COM0									LCDRAM8
																								COM1									LCDRAM9
																								COM2									LCDRAMA
																								COM3									LCDRAMB
																								COM4									LCDRAMC
																								COM5									LCDRAMD
																								COM6									LCDRAME
																								COM7									LCDRAMF

Figure 17-15 1/3 1/2 Duty LCD display mode 1

17.9.2. LCD Display mode 0 (MODE = 0)

1/8 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG29								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																																LCDRAMC
																																LCDRAMD
																																LCDRAME
																																LCDRAMF

Figure 17-16 1/8 Duty LCD display mode 0

1/6 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
		COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG29								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																															SEG36	LCDRAMC
																															SEG37	LCDRAMD
																																LCDRAME
																																LCDRAMF

Figure 17-17 1/6 Duty LCD display mode 0

1/4 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
				COM3	COM2	COM1	COM0					COM3	COM2	COM1	COM0					COM3	COM2	COM1	COM0					COM3	COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG29								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																															SEG36	LCDRAMC
																															SEG37	LCDRAMD
																															SEG38	LCDRAME
																															SEG39	LCDRAMF

Figure 17-18 1/4 Duty LCD display mode 0

1/3 Duty 1/2 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
					COM2	COM1	COM0						COM2	COM1	COM0						COM2	COM1	COM0						COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG27								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																															SEG36	LCDRAMC
																															SEG37	LCDRAMD
																															SEG38	LCDRAME
																															SEG39	LCDRAMF

Figure 17-19 1/3 1/2 Duty LCD display mode 0

17.10. LCD register

17.10.1. 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				BSEL			DUTY			BIAS	Res	Res	LCDCLK		EN
RW				RW			RW			RW			RW		RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	Contrast	RW	3’b000	LCD contrast adjustment Note: Valid only when internal resistor divider is selected as the source of Bias voltage. The higher the Constrast value, the lower the amplitude of the LCD waveform. At 0x0, the LCD waveform amplitude is maximum and the contrast is maximum; ...... 0xF, the LCD waveform amplitude is the smallest and the contrast is the smallest;
11:9	BSEL	RW	3’b000	Bias voltage source selection 111: Reserved 110: Internal resistive voltage divider, high power mode 101: Reserved 100: Internal resistive voltage divider, small power consumption mode

Bit	Name	R/W	Reset Value	Function
				011: Reserved 010: Internal resistive divider, medium power consumption mode 001: Reserved 000: External resistor mode, external circuitry required
8:6	DUTY	RW	3'b011	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	LCD Bias Configuration 0: 1/3 bias (initial value) 1: 1/2 bias
4:3	Reserved	-	-	Reserved
2:1	LCDCLK	RW	2'b01	LCD scan frequency selection 00: 64 Hz 01: 128 Hz 10: 256 Hz 11: 512 Hz Note: LCD frame rate = LCD scan frequency x Duty
0	EN	RW	0	LCD enable control 1: Enable 0: disable

17.10.2. 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	Res	Res	Res	INTF	DMAEN	IE	MODE	Res	BLINKEN	BLINKCNT					
				RO	RW	RW	RW		RW	RW					

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	INTF	RO	0	LCD interrupt flags

Bit	Name	R/W	Reset Value	Function
				1: Interrupt 0: No interrupt
10	DMAEN	RW	0	DMA hardware trigger enable 1: Enables LCD interrupt triggering DMA 0: Disable LCD interrupt triggered DMA
9	IE	RW	0	Interrupt enable 1: Enabled 0: disable
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: Disable
5:0	BLINKCNT	RW	0	Blink frequency and LCD interrupt interval set-ting Note: LCD blink frequency is = LCD frame rate / (BlinkCnt+1) LCD interrupt interval = (BlinkCnt+1)*(1/LCD frame frequency)

17.10.3. 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	Res	Res	Res	Res	INTF_CLR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
					R1W0										

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

17.10.4. Output configuration register (LCD_POEN0)

Address offset:0x0C

Reset value:0xFFFFFFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16

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
S15	S14	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
31:0	Sx	RW	0xFFFFFFFF	Segx output control bit 0: SEG output enable 1: SEG output off, other functions such as IO, analog input and output can be used

17.10.5. 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	MUX	Cx				SxCy				S			
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	MUX	RW	1	SEG32~SEG35 port function selection, refer to the selection table for details
11:8	Cx	RW	0xF	COMx output control bits (com0_com3) 0: COM output enable 1: COM output off, other functions such as IO, analogue inputs and outputs can be used
7:4	SxCy	RW	0xF	Segx/COMy output control bit 0: SEG/COM output enable 1: SEG/COM output off, other functions can be used, e.g. IO, analogue input/output SEG COM pin function selection is determined by CR0.DUTY
3:0	S	RW	0xF	Segx output control bit 0: SEG output enable 1: SEG output off, other functions such as IO, analog input and output can be used

Table 17-3 Register configuration

VLCDxSFGxPAD			How to configure the registers				
			MUX	S<35:32>	BSEL		
VLCDxSFGxPAD choose GPIO, when LCD disable (LCD_ON = 0)		VLCDHSEG35 = IO	1	1111	X	X	X
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
VLCDxSFGxPAD choose IO,when LCD enable (LCD_ON = 1), Only internal resistance operation mode can be selected	Small resistance (high current) mode	VLCDHSEG35 = IO	1	1111	1	1	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
	Medium resistance (medium current) mode	VLCDHSEG35 = IO	1	1111	0	1	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
	High resistance (low current) mode	VLCDHSEG35 = IO	1	1111	1	0	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
Selecting the internal resistance operating mode	Small resistance (high current) mode	VLCDHSEG35 = IO	1	1111	1	1	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
	Medium resistance (medium current) mode	VLCDHSEG35 = IO	1	1111	0	1	0
		VLCDHSEG34 = IO					

VLCDxSFGxPAD			How to configure the registers				
			MUX	S<35:32>	BSEL		
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
	High resistance (low current) mode	VLCDHSEG35 = IO	1	1111	1	0	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					
Select external resistor operation mode, internal resistor short circuit		VLCDHSEG35 = IO	1	1111	0	0	0
		VLCDHSEG34 = IO					
		VLCDHSEG33 = IO					
		VLCDHSEG32 = IO					

17.10.6. LCD_RAM0~7

Address offset:0x14~0x30

Reset value:0x00000000

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	0	LCD dot output, display reference LCD display mode 0 the corresponding SEG COM crosspoint is not illuminated; 1 the corresponding SEG COM crosspoint is illuminated;

17.10.7. LCD_RAM8~F

Address offset:0x34~0x50

Reset value:0x00

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	D7	D6	D5	D4	D3	D2	D1	D0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	Dx	RW	0	LCD dot output, display reference LCD display mode 0 the corresponding SEG COM crosspoint is not illuminated; 1 the corresponding SEG COM crosspoint is illuminated;

17.10.8. LCD register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	LCD_CR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CONTRAST				BSEL			DUTY			BIAS	Res.	Res.	LCD_CLK		EN
	Reset value																	0	0	0	0	0	0	0	0	1	1	0			0	1	0
0x04	LCD_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTF	DMAEN	IE	MODE	Res.	BLINKEN	BLINKCNT					
	Reset value																					0	0	0	0		0	0	0	0	0	0	0
0x08	LCD_INTCLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res	INTF_CLR	Res.									
	Reset value																						1										
	LCD_	S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16	S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0C	POEN0																																
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	LCD_POEN1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MUX	C3	C2	C1	C0	S39/C4	S38/C5	S37/C6	S36/C7	S35	S34	S33	S32
	Reset value																				1	1	1	1	1	1	1	1	1	1	1	1	1
0x14-0x30	LCD_RAM0~7	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34-0x50	LCD_RAM8~F	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	D7	D6	D5	D4	D3	D2	D1	D0
	Reset value																									0	0	0	0	0	0	0	0

18. Comparator (COMP)

18.1. Introduction

Three general purpose comparators (COMP1, COMP2 and COMP3) are integrated into the chip and can be used as individual modules or in combination with a timer.

The comparators can be used as follows:

- Triggered by analogue signals to generate low-power mode wake-up
- Analogue signal regulation
- Cycle by cycle current control loop when connected to the PWM output from the timer

18.2. COMP main features

- Each comparator has configurable positive or negative input for flexible voltage selection
  - Multiple I/O pins
  - VCC
  - Output of temperature sensor
  - Internal reference voltage and 3 fractional values (1/4, 1/2, 3/4) provided by voltage divider
- Configurable hysteresis function
- Programmable speed and power consumption
- Output can be connected to I/O or timer input as trigger
  - OCREF_CLR event (cycle by cycle current control)
  - Brake for fast PWM shutdown
- COMP1 and COMP2 can be combined into window COMP
- Each COMP has interrupt generation capability, which is used as wake-up (via EXTI) from low-power modes (sleep and stop modes)

18.3. COMP function description

18.3.1. COMP diagram

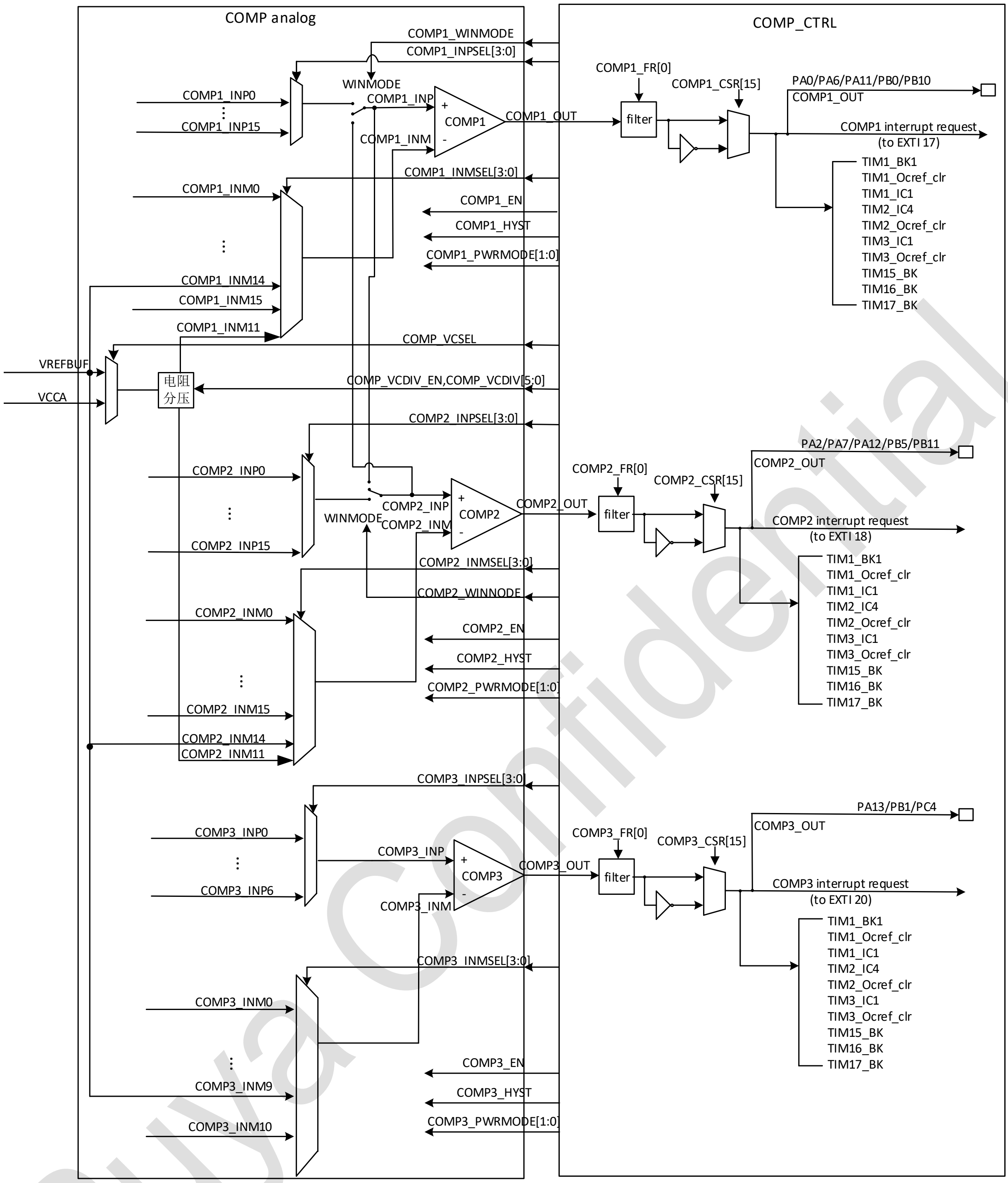


Figure 18-1 Comparator architecture block diagram

18.3.2. COMP pins and internal signals

The I/O used as comparator input must be configured in analog mode in the GPIO register.

The comparator output can be connected to the I/O pin through the alternate function channel (alternate function) on the GPIO.

The outputs can also be internally connected to the inputs of various timers for the following purposes:

- When the brake input is connected, the emergency shutdown of the PWM signal

- Cycle-by-cycle current control using OCREF_CLR input
- Input capture for timing measurements

18.3.3. COMP reset and clock

The COMP module has two clock sources:

- PCLK (APB clock), used to clock the configuration registers
- COMP clock, used to clock the circuitry after the analog comparator output (latching circuitry for analog outputs, burr filtering circuitry, etc.), can be selected as PCLK or LSI. select LSI when operation in stop mode is required.

Notes:PCLK and COMP CLK are enabled simultaneously by the RCC_APBENR2 control bit. If this enable is off, PCLK and COMP CLK are off, if it is enabled, PCLK and COMP CLK are on at the same time.

Before entering Stop mode, it is recommended to configure COMP CLK as LSI or LSE and then enter Stop mode.

The COMP module reset signal consists of the APB reset source and the COMP module software reset source:

1) APB reset for COMP register reset
2) COMP software reset for resetting the circuits after the analogue comparator output (latching circuits, burr filtering circuits, etc. for the analogue output)

Notes:When the reset signal in RCC_APBRSTR2 is enabled, both the COMP module PRESETn and COMP_RSETn signals will be reset.

18.3.4. Window Comparator

The function of the window comparator is to monitor whether the analogue voltage is within the low and high thresholds.

A window comparator can be created using two comparators. The analogue voltage being monitored is connected to the non-inverting (+ terminal) input of both comparators simultaneously, the high and low thresholds are connected to the inverting inputs (- terminal) of both comparators respectively.

By enabling the WINMODE bit, the non-inverting (+ input) of the two comparators can be connected together, serving to save one I/O pin.

Note: The WINMODE mode of both COMPs cannot be enabled at the same time.

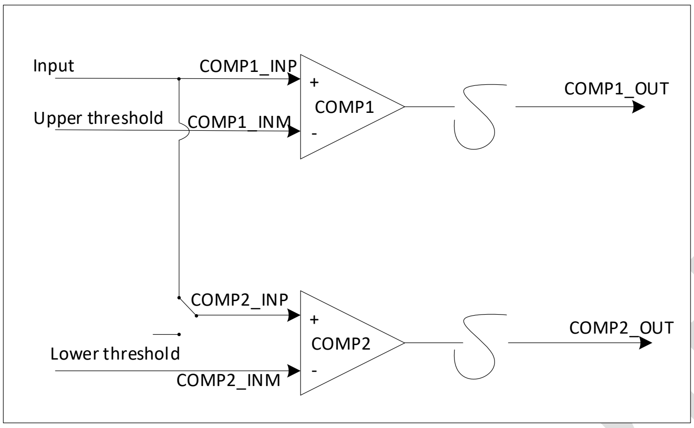


Figure 18-2 window comparator

18.3.5. Hysteresis

To avoid false output transitions in the case of noisy signals, the comparators can be enabled with hysteresis (COMP1, COMP2 and COMMP3 have separate hysteresis enable signals COMP1_HYST, COMP2_HYST and COMMP3_HYST). To avoid false output transitions in the case of noisy signals, the comparators can be enabled with hysteresis (COMP1, COMP2 and COMMP3 have separate hysteresis enable signals COMP1_HYST, COMP2_HYST and COMMP3_HYST).

18.3.6. Power consumption mode

The power consumption and transmission delay of the comparator can be selected using the PWR-MODE[1:0] bits of the COMPx_CSR register to achieve the most suitable trade-off for a particular application. The high speed mode consumes more power and has less delay. Note that if the PWR_CR2 register LPR=1 is selected (i.e. low power regulator is selected) before entering stop, COMP needs to be set at medium speed first (PWRMODE=01).

In addition, to reduce power consumption, the APB clock and COMP clock are controlled by RCC_APBENR2.COMP1EN, RCC_APBENR2.COMP2EN and RCC_APBENR2.COMP3EN, and software can enable this register only when the COMP module is in use.

18.3.7. 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 done before COMP_EN is enabled.

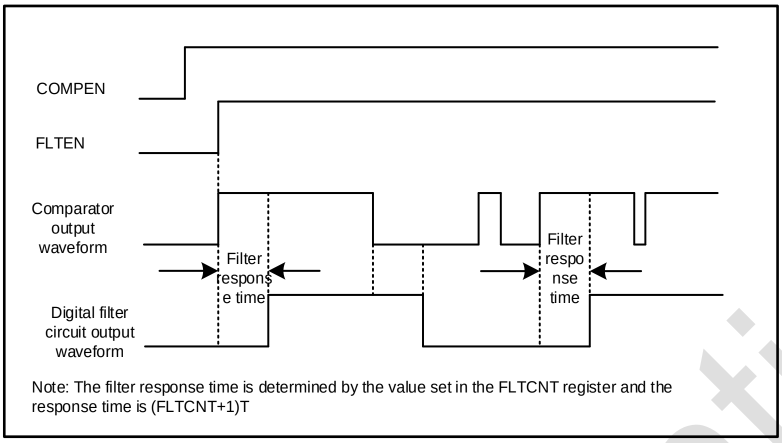


Figure 18-3 COMP filter

18.3.8. COMP interruption

The comparator outputs are internally connected to EXTI controllers (extended interrupts and events) on the chip. Each comparator has a separate EXTI line (17 and 18) and can generate interrupts or events. The same mechanism is used for wake-up from low power consumption.

18.4. COMP register

18.4.1. COMP1 control and status register (COMP1_CSR)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	COMP_OUT	Res	Res	COMP_VCSEL	COMP_VCDIV_EN	COMP_VCDIV[5:0]						PWR-MODE[1:0]		Res	COMP1_HYST
	R			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PO-LAR-ITY	Res	Res	Res	WINMODE	Res	INPSEL[3:0]				INNSEL[3:0]				Res	COMP1_EN
RW				RW		RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	0	-

Bit	Name	R/W	Reset Value	Function
30	COMP_OUT	R	0	COMP1 output status This bit is read only and reflects the output level of COMP1 after polarity selection.
29：28	Reserved	-	0	-
27	COMP_VCSEL	RW	0	COMP1,COMP2 reference voltage Vref selection 0: VCC 1: ADC reference voltage
26	COMP_VCDIV_EN	RW	0	Voltage division enable 1: Enabled 0: not enabled
25:20	COMP_VCDIV[5:0]	RW	0	Pressure divider selection 00_0000: 1/64 Vref 00_0001: 2/64 Vref 00_0010: 3/64 Vref ... 11_1110: 63/64 Vref 11_1111: Vref
19:18	PWRMODE[1:0]	RW	0	COMP1 power consumption mode selection The power consumption and consequently the speed of COMP1 are selected 00: High speed (250uA) 01: Medium speed (5uA) 10: Reserved 11: reserved
17	Reserved	-	-	-
16	COMP1_HYST	RW	0	COMP1 hysteresis function enable control 0: No hysteresis 1: Hysteresis voltage approx. 20mV
15	POLARITY	RW	0	COMP1 output polarity selection Software readable and writable 0: not inverted 1: Inverted
14:12	Reserved	-	0	-
11	WINMODE	RW	0	COMP1 non-inverted output selection (window mode) Software readable and writable 0: signal selected by INPSEL[3:0] 1: Signal COMP2_INP for COMP2 Note that the WINMODE mode of both COMPs cannot be enabled at the same time.
10	Reserved	-	0	-
9:6	INPSEL[3:0]	RW	0	Signal selection for COMP1 non-inverted input 0000: COMP1_INP0 from PC0 0001: COMP1_INP1 from PC1 0010: COMP1_INP2 from PC2 0011: COMP1_INP3 from PC3

Bit	Name	R/W	Reset Value	Function
				0100: COMP1_INP4 from PA0 0101: COMP1_INP5 from PA1 0110: COMP1_INP6 from PA2 0111: COMP1_INP7 from PA3 1000: COMP1_INP8 from PA4 1001: COMP1_INP9 from PA5 1010: COMP1_INP10 from PA6 1011: COMP1_INP11 from PA7 1100: COMP1_INP12 from PB4 1101: COMP1_INP13 from PB5 1110: COMP1_INP14 from PB6 1111: COMP1_INP15 from DAC1_VIN
5:2	INNSEL[3:0]	RW	0	Signal selection for COMP1 non-inverted input 0000: COMP1_INM0 from PA0 (Note: COMP1_INM0 in the IO mapping table corresponds to COMP1_INN0) 0001: COMP1_INM1 from PA1 (Note: COMP1_INM1 in the IO mapping table corresponds to COMP1_INN1) 0010: COMP1_INM2 from PA2 (Note: COMP1_INM2 in the IO mapping table corresponds to COMP1_INN2) 0011: COMP1_INM3 from PA3 (Note: COMP1_INM3 in the IO mapping table corresponds to COMP1_INN3) 0100: COMP1_INM4 from PA4 (Note: COMP1_INM4 in the IO mapping table corresponds to COMP1_INN4) 0101: COMP1_INM5 from PA5 (Note: COMP1_INM5 in the IO mapping table corresponds to COMP1_INN5) 0110: COMP1_INM6 from PA6 (Note: COMP1_INM6 in the IO mapping table corresponds to COMP1_INN6) 0111: COMP1_INM7 from PA7 (Note: COMP1_INM7 in the IO mapping table corresponds to COMP1_INN7) 1000: COMP1_INM8 from PC4 (Note: COMP1_INM8 in the IO mapping table corresponds to COMP1_INN8) 1001: COMP1_INM9 from PC5 (Note: COMP1_INM9 in the IO mapping table corresponds to COMP1_INN9) 1010: COMP1_INM10 from DAC1_VIN 1011: COMP1_INM11 from resistor voltage divider

Bit	Name	R/W	Reset Value	Function
				1100: COMP1_INM12 from TS_VIN (temperature sensor voltage) 1101: COMP1_INM13 from VREF1P2 (internal reference 1.2V output voltage) 1110: COMP1_INM14 from VREFBUF (VREFBUFFERE bit and Verfbuff_sel[1:0] of ADC module need to be enabled) 1111: COMP1_INM15 from OPA1_VIN
1	Reserved	-	0	-
0	COMP1_EN	RW	0	COMP1 enable bit Software readable and writable (if not locked) 0: Disable 1: Enable

18.4.2. COMP1 filter register (COMP1_FR)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1[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[15:0]	RW	0	Comparator 1 sampling filter counter The sample clock is APB or LSI or LSE, and the filter count value is configurable. When the number of samples reaches the filter count value, the result is output consistently. Sample count period = FLTCNT[15:0]
15:1	Reserved	-	0	-
0	FLTEN1	RW	0	Comparator 1 digital filtering function configuration 0: Disable digital filtering function 1: Enable the digital filter function Note: This bit must be set when COMP1_EN is 0

18.4.3. COMP2 control and status registers (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

Res	COMP_OUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PWR-MODE[1:0]		Res	Res
	R											RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PO-LAR-ITY	Res	Res	Res	WINMODE	Res	INPSEL[3:0]				INNSEL[3:0]				COMP2_HYST	COMP2_EN
RW				RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	0	-
30	COMP_OUT	R	0	COMP2 Output Status This bit is read-only and it reflects the output level of COMP2 after polarity selection.
29：20	Reserved	-	0	-
19:18	PWRMODE[1:0]	RW	0	COMP2 power consumption mode selection The power consumption and consequently the speed of COMP2 are selected 00: High speed 01: Medium speed 10: reserved 11: reserved
17:16	Reserved	-	-	-
15	POLARITY	RW	0	COMP2 output polarity selection Software readable and writable 0: No reverse 1: Reverse
14:12	Reserved	-	0	-
11	WINMODE	RW	0	COMP2 non-inverted output selection Software readable and writable 0: Signal is selected by INPSEL[1:0] 1: COMP2_INP signal of COMP2 Note that the WINMODE mode of both COMPs cannot be enabled at the same time.
10	Reserved	-	0	-
9:6	INPSEL[3:0]	RW	0	Signal selection for COMP2 non-inverted input, software readable and writable 0000: COMP2_INP0 from PA0 0001: COMP2_INP1 from PA1 0010: COMP2_INP2 from PA2 0011: COMP2_INP3 from PA3 0100: COMP2_INP4 from PA4 0101: COMP2_INP5 from PA5 0110: COMP2_INP6 from PB1

Bit	Name	R/W	Reset Value	Function
				0111: COMP2_INP7 from PB2 1000: COMP2_INP8 from PB10 1001: COMP2_INP9 from PB12 1010: COMP2_INP10 from PB13 1011: COMP2_INP11 from PB14 1100: COMP2_INP12 from PB4 1101: COMP2_INP13 from PB6 (Note: COMP2_INP13 in the IO mapping table corresponds to COMP2_INP14) 1110: COMP2_INP14 from PB7 (Note: COMP2_INP14 in the IO mapping table corresponds to COMP1_INP15) 1111: COMP2_INP15 from DAC2_VIN
5:2	INNSEL[3:0]	RW	0	Signal selection for COMP2 non-inverted input 0000: COMP2_INM0 from PC0 (Note: COMP2_INM0 in the IO mapping table corresponds to COMP2_INN0) 0001: COMP2_INM1 from PC1 (Note: COMP2_INM1 in the IO mapping table corresponds to COMP2_INN1) 0010: COMP2_INM2 from PC2 (Note: COMP2_INM2 in the IO mapping table corresponds to COMP2_INN2) 0011: COMP2_INM3 from PC3 (Note: COMP2_INM3 in the IO mapping table corresponds to COMP2_INN3) 0100: COMP2_INM4 from PA0 (Note: COMP2_INM4 in the IO mapping table corresponds to COMP2_INN4) 0101: COMP2_INM5 from PA1 (Note: COMP2_INM5 in the IO mapping table corresponds to COMP2_INN5) 0110: COMP1_INM6 from PB0 (Note: COMP2_INM6 in the IO mapping table corresponds to COMP2_INN6) 0111: COMP1_INM7 from PB1 (Note: COMP2_INM7 in the IO mapping table corresponds to COMP2_INN7) 1000: COMP2_INM8 from PB2 (Note: COMP2_INM8 in the IO mapping table corresponds to COMP2_INN8) 1001: COMP2_INM9 from PB3 (Note: COMP2_INM9 in the IO mapping table corresponds to COMP2_INN9) 1010: COMP2_INM10 from DAC2_VIN

Bit	Name	R/W	Reset Value	Function
				1011: COMP2_INM11 from resistor divider voltage 1100: COMP2_INM12 from TS_VIN (temperature sensor voltage) 1101: COMP2_INM13 from VREF1P2 (internal reference 1.2V output voltage) 1110: COMP2_INM14 from VREFBUF (VREFBUFFERE bit and Verfbuff_sel[1:0] of ADC module need to be enabled) 1111: COMP2_INM15 from OPA2_VIN (OPA2 output voltage)
1	COMP2_HYST	RW	0	COMP2 hysteresis function enable control 0: No hysteresis 1: Hysteresis voltage about 20mV
0	COMP2_EN	RW	0	COMP2 enable bit Software readable and writable 0：Disable 1：Enable

18.4.4. COMP2 filter 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[15:0]	RW	0	Comparator 2 Sample Filter Counter The sample clock is APB or LSI or LSE, and the filter count value is configurable. When the number of samples reaches the filter count value, the result is output consistently. Sample count period = FLTCNT[15:0]
15:1	Reserved	-	0	-
0	FLTEN2	RW	0	Comparator 2 digital filtering function configuration 0: Disable the digital filter function 1: Enables the digital filter function Note: This bit must be set when COMP2_EN is 0

18.4.5. COMP3 control and status registers (COMP3_CSR)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	COMP_OUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PWR-MODE[1:0]		Res	Res
	R											RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PO-LARITY	Res	Res	Res	Res	Res	INPSEL[3:0]				INNSEL[3:0]				COMP3_HYST	COMP3_EN
RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	0	-
30	COMP_OUT	R	0	COMP3 Output Status This bit is read-only and it reflects the output level of the COMP3 after polarity selection.
29：20	Reserved	-	0	-
19:18	PWRMODE[1:0]	RW	0	COMP3 power consumption mode selection Selects the power consumption and consequently the speed of the COMP3 00: High speed 01: Medium speed 10:reserved 11:reserved
17:16	Reserved	-	-	-
15	POLARITY	RW	0	COMP3 output polarity selection Software readable and writable 0: No reverse 1: Reverse
14:11	Reserved	-	0	-
10	Reserved	-	0	-
9:6	INPSEL[3:0]	RW	0	Signal selection for COMP3 non-inverted input, software readable and writable 0000: COMP3_INP0 from PA5 0001: COMP3_INP1 from PB1 0010: COMP3_INP2 from PB11 (Note: COMP3_INP2 in IO mapping table corresponds to COMP3_INP8) 0011: COMP3_INP3 from PB14 (Note: COMP3_INP3 in the IO mapping table corre-sponds to COMP3_INP9) 0100: COMP3_INP4 from PC7

Bit	Name	R/W	Reset Value	Function
				(Note: COMP3_INP4 in the IO mapping table corresponds to COMP3_INP13) 0101: COMP3_INP5 from DAC1_VIN 0110:COMP3_INP6 from DAC2_VIN other bits:reserved
5:2	INNSEL[3:0]	RW	0	Signal selection for COMP3 non-inverted input 0000: COMP3_INM0 from PA5 (Note: COMP3_INM0 in the IO mapping table corresponds to COMP3_INN0) 0001:COMP3_INM1 from PB1 (Note: COMP3_INM1 in IO mapping table corresponds to COMP3_INP1) 0010:COMP3_INM2 from PB11 (Note: COMP3_INM2 in the IO mapping table corresponds to COMP3_INN4) 0011:COMP3_INM3 from PB14 (Note: COMP3_INM3 in the IO mapping table corresponds to COMP3_INN5) 0100: COMP3_INM4 from PC7 (Note: COMP3_INM4 in IO mapping table corresponds to COMP3_INN8) 0101:COMP3_INM5 from DAC1_VIN 0110:COMP3_INM6 from DAC2_VIN 0111: CMP3_INM7 from TS_VIN (temperature sensor voltage) 1000: CMP3_INM8 from VREF1P2 (internal reference 1.2V output voltage) 1001: CMP3_INM9 from VREFBUF (need to enable VREFBUFFERE bit and Verfbuff_sel[1:0] of ADC module) 1010: CMP3_INM10 from OPA3_VIN (OPA3 output voltage) Other bits:Reserved
1	COMP3_HYST	RW	0	COMP3 hysteresis function enable control 0: No hysteresis 1: Hysteresis voltage about 20mV
0	COMP3_EN	RW	0	COMP3 enable bit Software readable and writable 0:Disable 1:Enable

18.4.6. COMP3 filter register (COMP3_FR)

Address offset:0x24

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT3[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FLTEN3
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT3[15:0]	RW	0	Comparator 3 Sample Filter Counter The sample clock is APB or LSI or LSE, and the filter count value is configurable. When the number of samples reaches the filter count value, the result is output consistently. Sample count period = FLTCNT[15:0]
15:1	Reserved	-	0	-
0	FLTEN3	RW	0	Comparator 2 digital filtering function configuration 0: Disable the digital filter function 1: Enables the digital filter function Note: This bit must be set when COMP2_EN is 0

18.4.7. COMP register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	COMP1_CSR	Res.	COMP_OUT	Res.	Res.	COMP_VCSEL	COMP_VCDIV_EN	COPM_VCDIV[5:0]						PWRMODE[1:0]		Res.	COMP1_HYST	POLARITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL[3:0]				INNSEL[3:0]				Res.	COMP1_EN
	Reset value		0			0	0							0	0		0	0				0		0	0	0	0	0	0	0	0		0
0x04	COMP1_FR	FLTCNT1[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x10	COMP2_CSR	Res.	COMP_OUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRMODE[1:0]		Res.	Res.	POLARITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL[3:0]				INNSEL[3:0]				COMP2_HYST	COMP2_EN
	Reset value		0											0	0			0				0		0	0	0	0	0	0	0	0	0	0
0x14	COMP2_FR	FLTCNT2[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																0
0x20	COMP3_CSR	Res.	COMP_OUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRMODE[1:0]		Res.	Res.	POLARITY	Res.	Res.	Res.	Res.	Res.	INPSEL[3:0]				INNSEL[3:0]				COMP3_HYST	COMP3_EN
	Reset value		0											0	0			0						0	0	0	0	0	0	0	0	0	0
0x24	COMP3_FR	FLTCNT3[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN3
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																0

19. Operational Amplifier (OPA)

19.1. OPA Introduction

The OPA module is suitable for simple amplifier applications. The three internal op-amps can be cascaded using external resistors. The OPA has an input range of 0V to VCCA and an output range of 0.1 V to VCCA-0.2 V.

19.2. OPA Main Features

- 3 independently configurable op-amps
- OPA input range is 0 to VCCA and output range is 0.1 V to VCCA-0.2 V programmable gain
- The following modes can be configured
  - General purpose op-amp mode

19.3. OPA Function Description

The 3 OPAs can be amplified by using external components to form an amplifier to amplify the small and large signal analog input signal, and the output is the amplified signal.

19.3.1. OPA Block Diagram

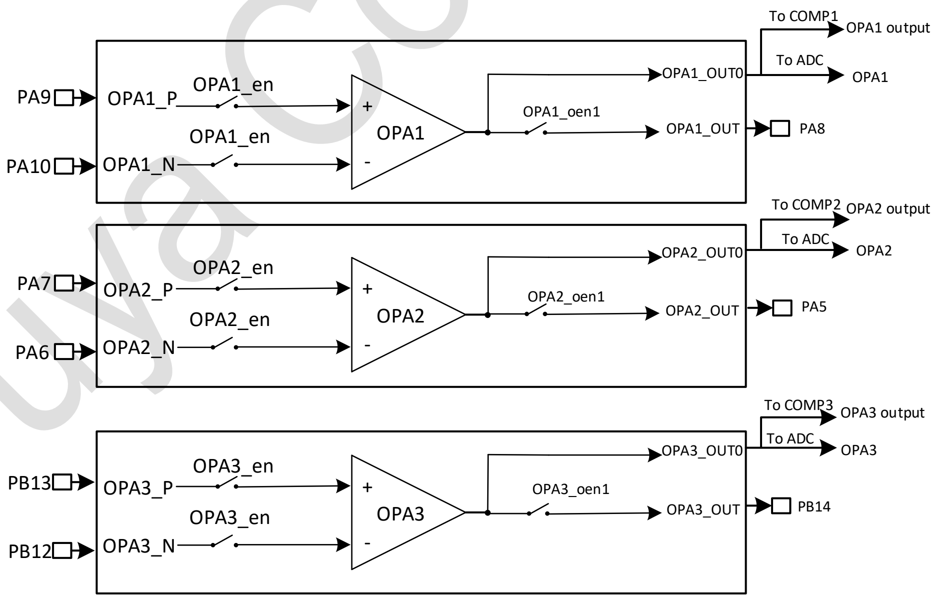


Figure 19-1 OPA Architecture Block Diagram

19.4. OPA Register

19.4.1. OPA Output Enable Register (OPA_CR0)

Address offset:0x30

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res				OP3OEN1	Res				OP2OEN1	Res				OP1OEN1	Res
				RW					RW					RW	

Bit	Name	R/W	Reset Value	Function
31:12	Res	-	-	-
11	OP3OEN1	RW	0	OP3 output 1 enable
10:7	Res	-	-	-
6	OP2OEN1	RW	0	OP2 output 1 enable
5:2	Res	-	-	-
1	OP1OEN1	RW	0	OP1 output 1 enable
0	Res	-	-	-

19.4.2. OPA Control Register (OPA_CR1)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								EN3	EN2	EN1	Res				
								RW	RW	RW					

Bit	Name	R/W	Reset Value	Function
31:8	Res	-	-	-
7	EN3	RW	0	OPA3 Enable
6	EN2	RW	0	OPA2 Enable
5	EN1	RW	0	OPA1 Enable
4:0	Res	-	-	-

19.4.3. OPA register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30	OPA_CR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OP3OEN1.	Res.	Res.	Res.	Res.	OP2OEN1.	Res.	Res.	Res.	Res.	OP1OEN1.	Res.
	Reset value																					0					0					0	
0x34	OPA_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EN3.	EN2.	EN1.	Res.	Res.	Res.	Res.	Res.
	Reset value																									0	0	0					

20. Hardware Divider (DIV)

20.1. DIV Introduction

DIV (Divider) is a 32-bit signed/unsigned integer hardware divider.

20.2. DIV main features

- Support 32-bit division
- The data in the register cannot be changed while the current division is not finished
- Configurable signed/unsigned integer division calculation
- 32-bit divisor, 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 divide operation
- 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

20.3. DIV Function Description

20.3.1. DIV operation flow

- Turn on the module clock enable bit of the hardware divider in the RCC system clock controller.
- Configure register DIV_SIGN to set the signed/unsigned division operation.
- Configure register DIV_DEND to set the number to be divided.
- Configuration register DIV_SOR sets the divisor and the division operation starts.
- Query register DIV_STAT for the operation end flag bit DIV_END, DIV_END is 1 to mark the operation end. Read register DIV_QUOT to get the quotient, and read register HDIV_REMD to get the remainder.
- When the divisor is zero, the division operation ends immediately and the quotient and remainder are set to zero at the same time, and the divisor zero warning flag bit DIV_ZERO is set.

- When reading register DIV_QUOT/DIV_REMD before the end of division operation, the CPU will read the last calculated value

Example: Calculate an unsigned division, the divisor is 1917887483 (0x7250A3FB) and the divisor is 9597 (0x257D)

Step 1, configure register DIV_SIGN to 0, i.e. unsigned division operation

Step 2, configure register DIV_DEND to 0x7250A3FB, i.e. set the divisor

Step 3, configure register DIV_SOR to 0x257D, i.e. set the divisor and start calculation

Step 4, query the DIV_STAT end of operation flag bit DIV_END, DIV_END is 1 flag, the operation ends.

The operation is finished.

Read register DIV_QUOT to get the quotient 199842 (0x30CA2)

Read register DIV_REMD to get the remainder 3809(0xEE1)

20.4. DIV Register

20.4.1. DIV divisor register (DIV_DEND)

Address offset:0x00

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_DEND	RW	0	Divisor register

20.4.2. DIV divisor register (DIV_SOR)

Address offset:0x04

Reset value:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIV_SOR[31:16]															

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_SOR	RW	1	Divisor register (writing this register automatically triggers the division operation)

20.4.3. DIV Merchant Register (DIV_QUOT)

Address offset:0x08

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_QUOT	RW	0	Store the quotient calculated by the divider

20.4.4. 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
DIV_REMD [31:16]															
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV_REMD [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:0	DIV_REMD	RW	0	Store the remainder calculated by the divider

20.4.5. DIV symbol register (DIV_SIGN)

Address offset:0x10

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**
Reserved															SIGN
															RW

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	-
0	SIGN	RW	0	Symbol selection register 0: Unsigned division operation 1: Signed division operation

20.4.6. DIV Status Register (DIV_STAT)

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**
Reserved														DIV_ZERO	DIV_END
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	-
1	DIV_ZERO	R	0	Divisor is zero warning flag bit 0: Divisor is not zero 1: Divisor is zero
0	DIV_END	R	0	Division end flag bit 0: Operation in progress 1: End of operation

20.4.7. DIV register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DIV_DEND	DIV_DEND[31:0]																															
	Reset	0																															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x04	DIV_SOR	DIV_SOR[31:0]																															
	Reset value	1																															
0x08	DIV_QUOT	DIV_QUOT[31:0]																															
	Reset value	0																															
0x0C	DIV_REMD	DIV_REMD [31:0]																															
	Reset value	0																															
0x10	DIV_SIGN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIGN
	Reset va																																0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	lue																																
0x14	DIV_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_ZERO.	DIV_END.
	Reset value																															0	0

21. Advanced-control timer (TIM1)

21.1. TIM1 introduction

The advanced-control timers (TIM1) 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, complementary PWM with dead-time insertion).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The advanced-control (TIM1) and general-purpose (TIMx) timers are completely independent, and do not share any resources. They can be synchronized together.

21.2. TIM1 main features

- 16-bit up, down, up/down auto-reload counter.
- 16-bit programmable prescaler allowing dividing (also "on the fly") the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time
- Synchronization circuit to control the timer with external signals and to interconnect several timers together.
- Repetition counter to update the timer registers only after a given number of cycles of the counter.
- Break input to put the timer's output signals in reset state or in a known state.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - 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

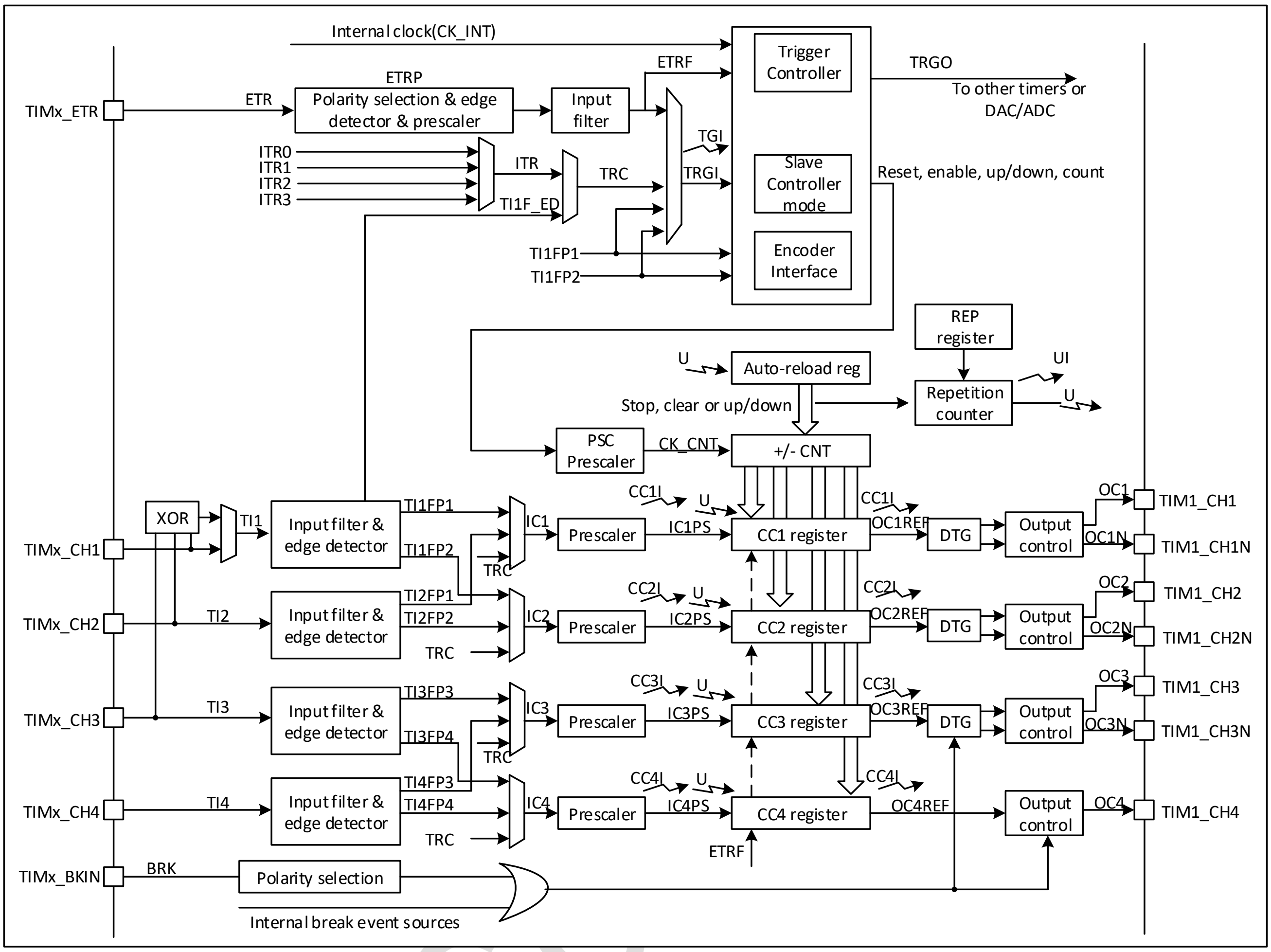


Figure 21-1 Advanced-control timer block diagram

21.3. TIM1 functional description

21.3.1. 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 (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

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 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 TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. 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.

Figure 21-2 and Figure 21-3 give some examples of the counter behavior when the prescaler ratio is changed on the fly:

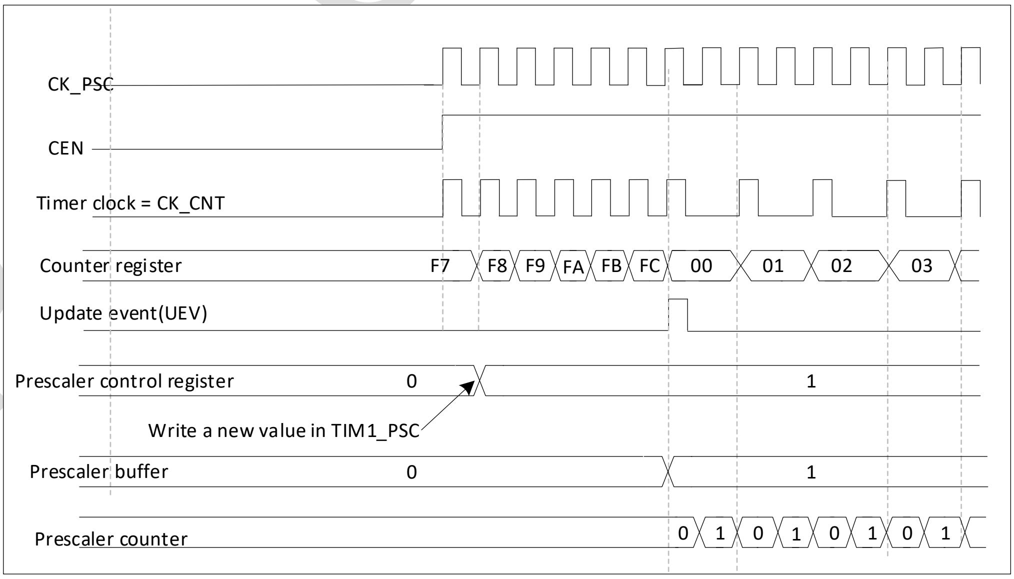


Figure 21-2 Counter timing diagram with prescaler division change from 1 to 2

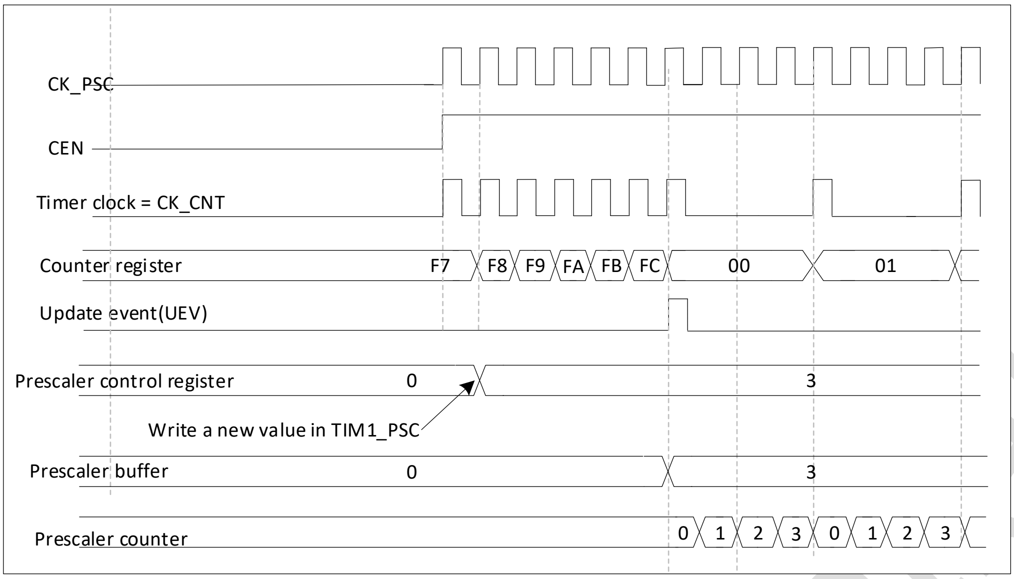


Figure 21-3 Counter timing diagram with prescaler division change from 1 to 4

21.3.2. Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), 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 plus one (TIMx_RCR+1). 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_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 auto-reload 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_PSCregister)

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

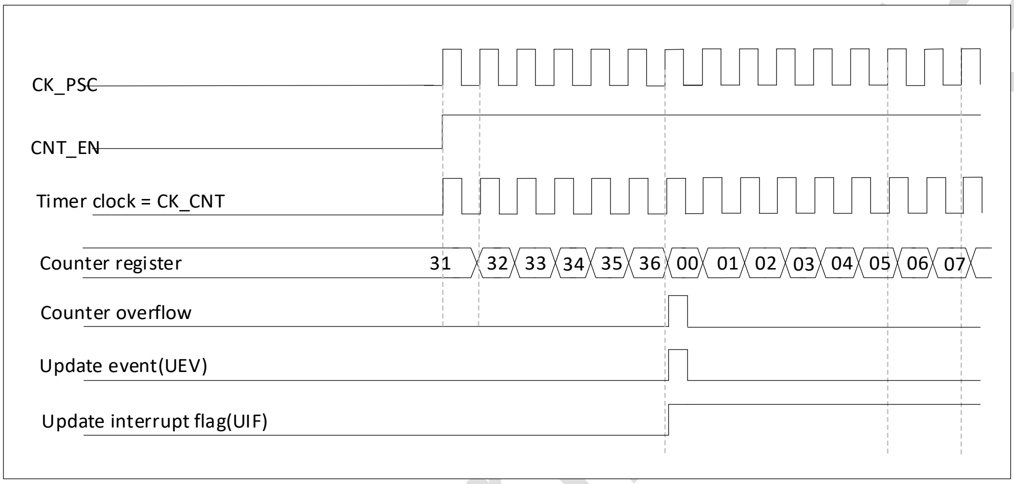


Figure 21-4 Counter timing diagram, internal clock divided by 1

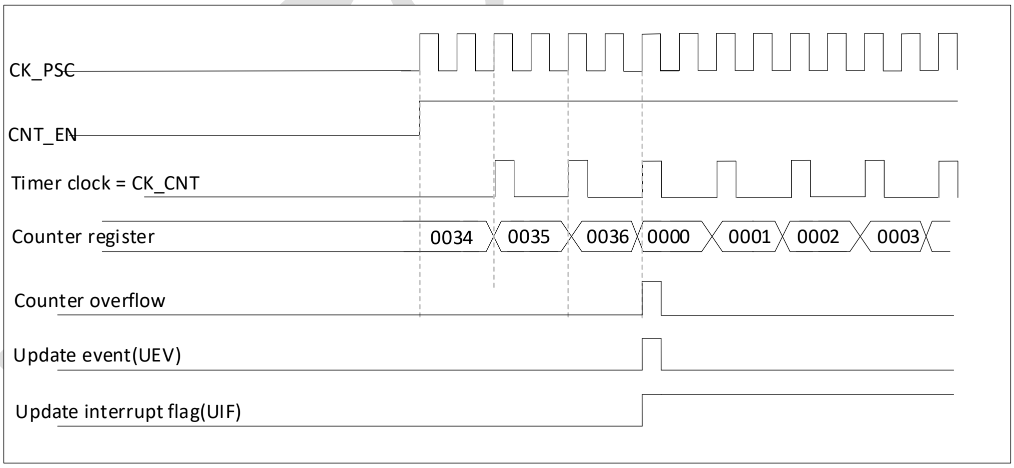


Figure 21-5 Counter timing diagram, internal clock divided by 2

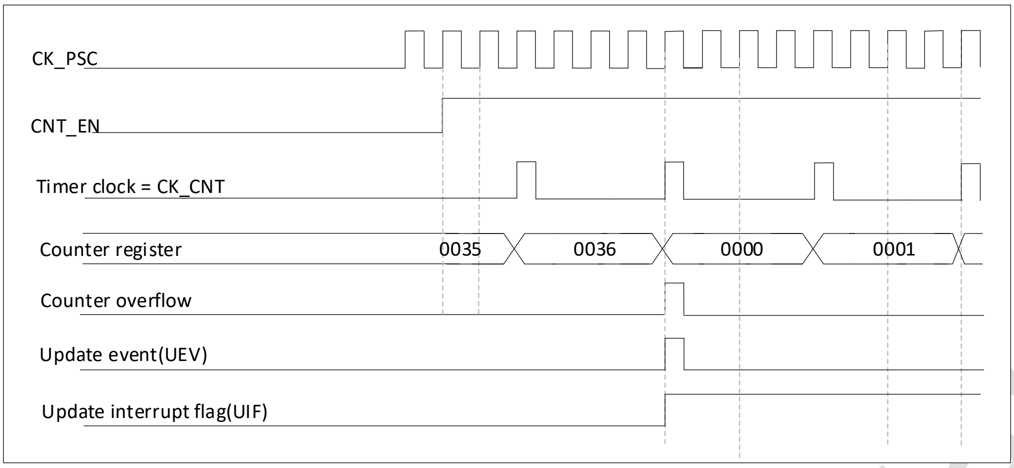


Figure 21-6 Counter timing diagram, internal clock divided by 4

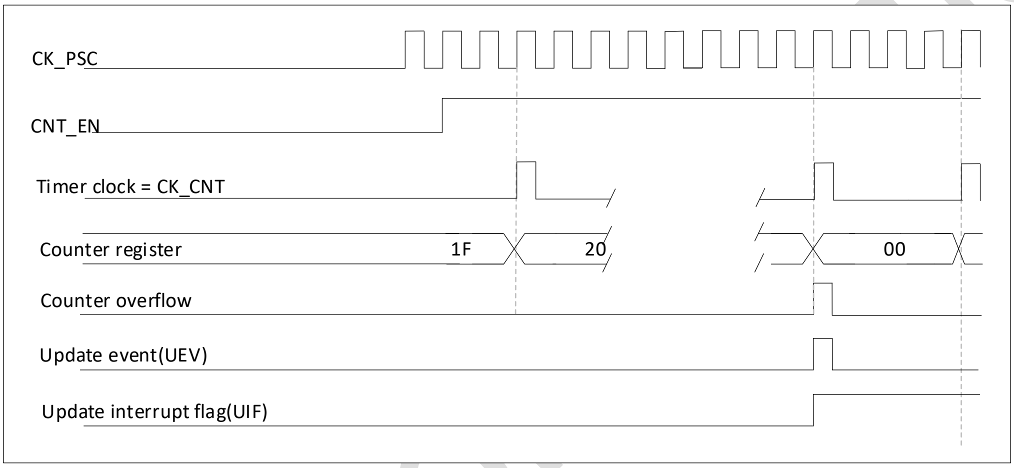


Figure 21-7 Counter timing diagram, internal clock divided by N

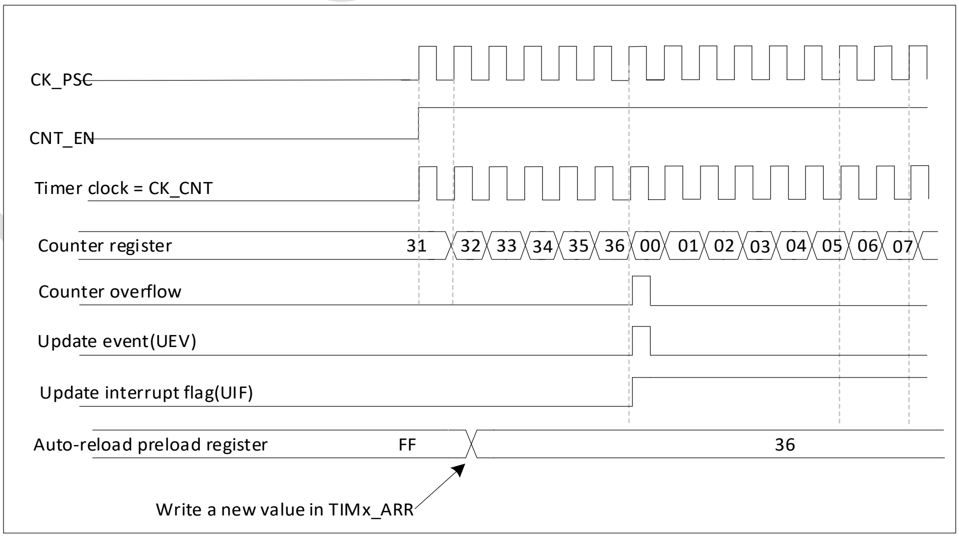


Figure 21-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR no preloaded)

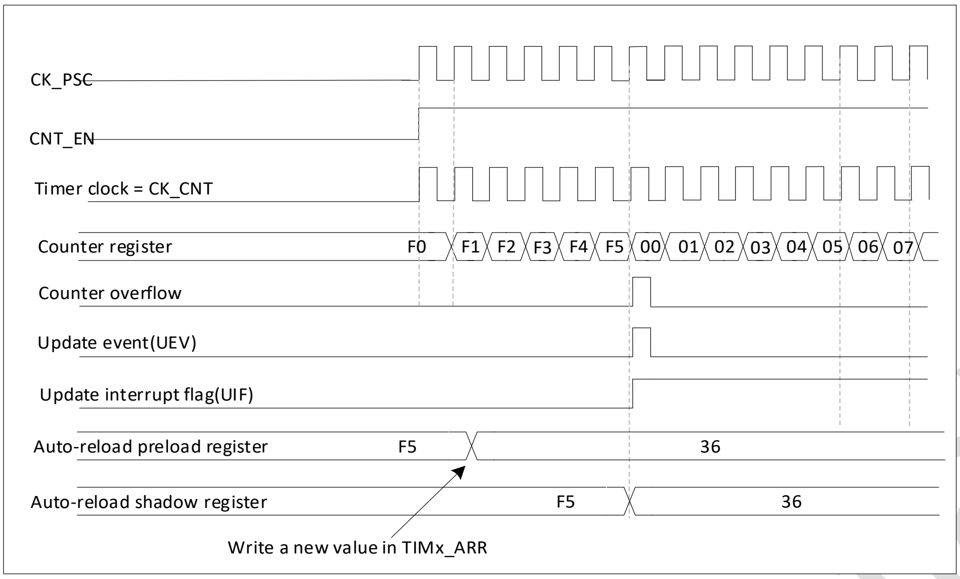


Figure 21-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMx_ARR register) 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 update event can be disabled by software by setting the UDIS bit in 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 prescale 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 update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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: the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

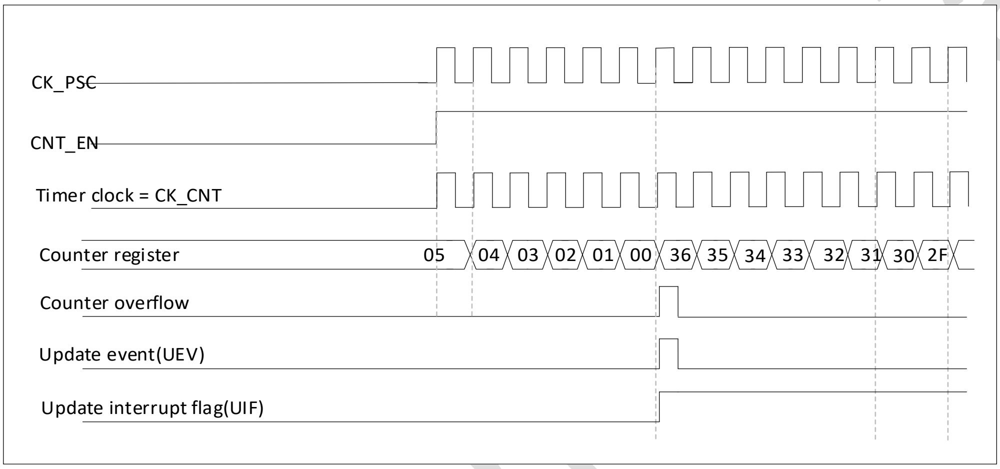


Figure 21-10 Counter timing diagram, internal clock divided by 1

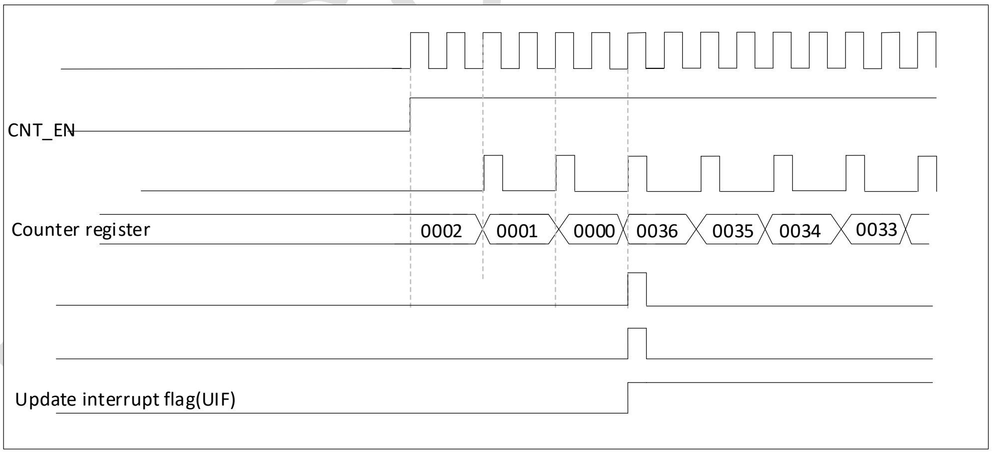


Figure 21-11 Counter timing diagram, internal clock divided by 2

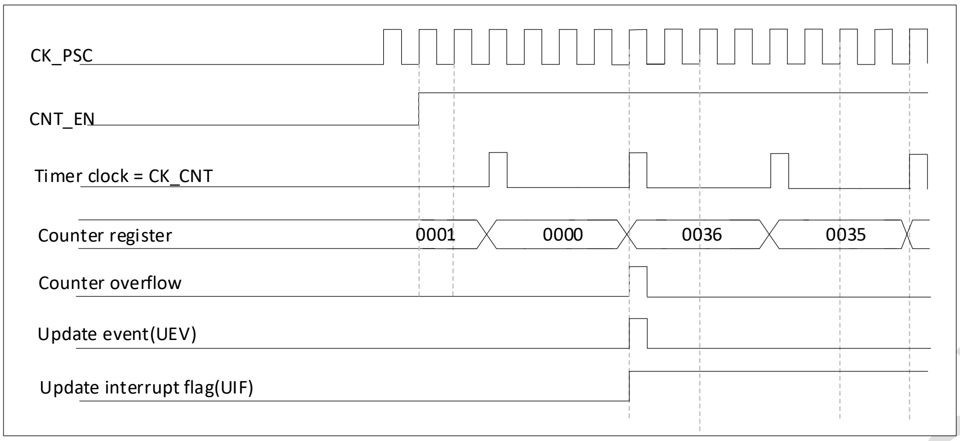


Figure 21-12 Counter timing diagram, internal clock divided by 4

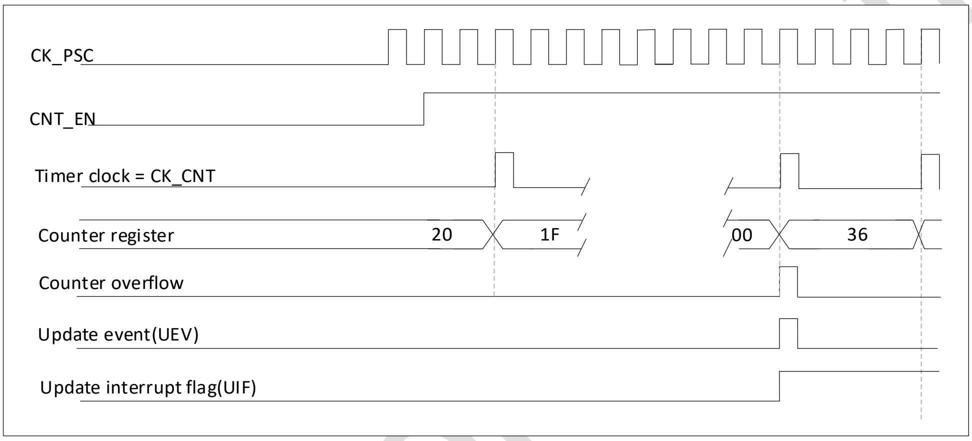


Figure 21-13 Counter timing diagram, internal clock divided by N

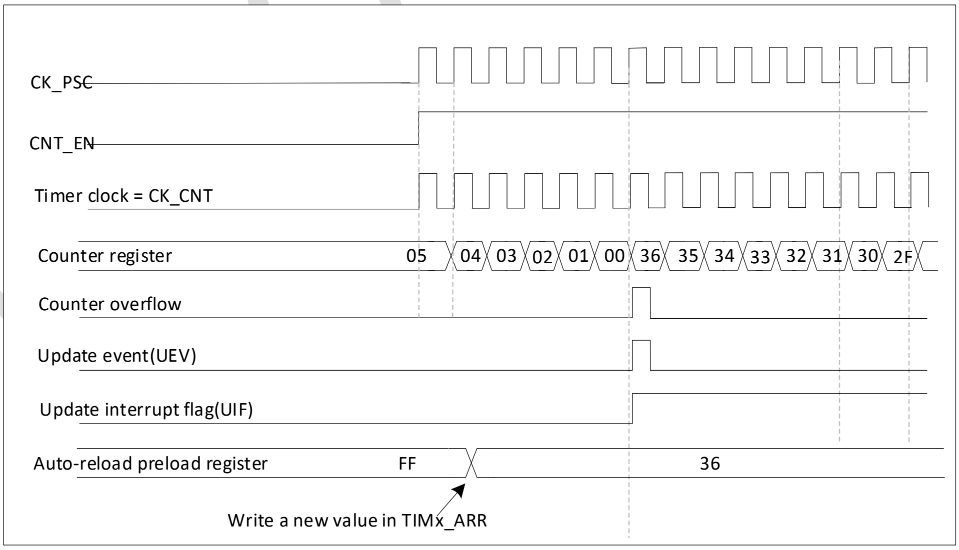


Figure 21-14 Counter timing diagram, update event when repetition counter is not used

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 autoreload 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 '00'. 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 from 0, as well as the counter of the prescaler.

The UEV update 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 or DMA request is 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: if the update source is a counter overflow, the autoreload is updated before the counter is re-loaded, so that the next period is the expected one (the counter is loaded with the new value).

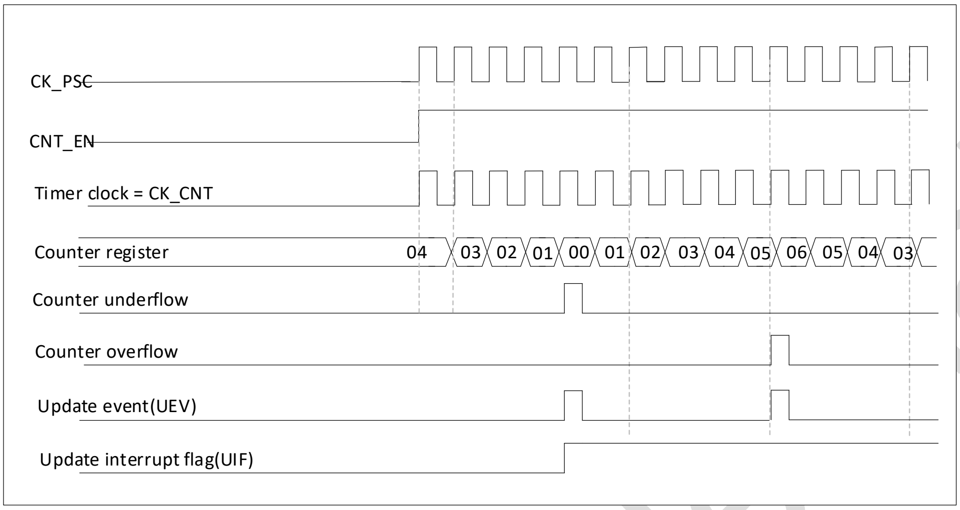


Figure 21-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

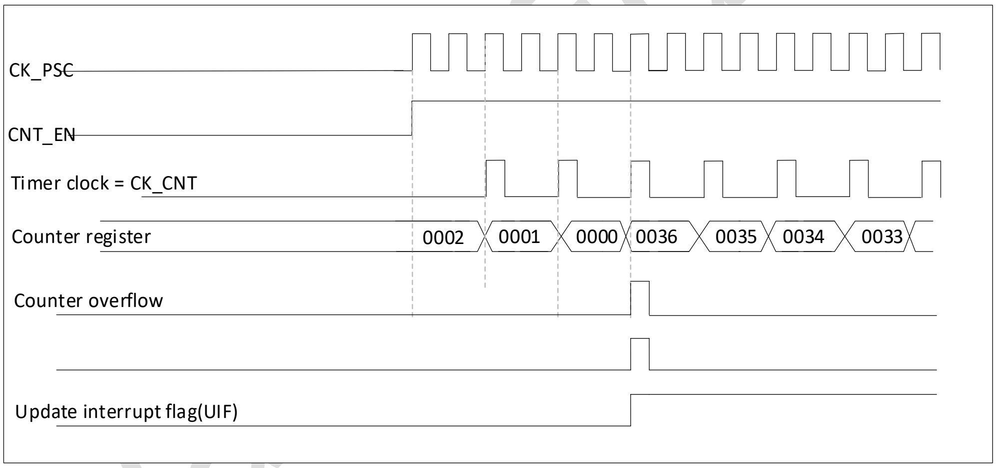


Figure 21-16 Counter timing diagram, internal clock divided by 2, TIMx_ARR = 0x36

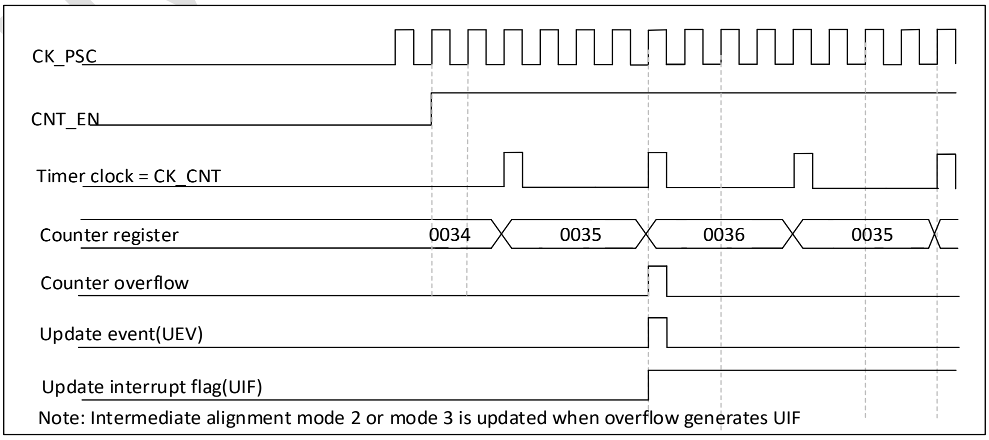


Figure 21-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR = 0x36

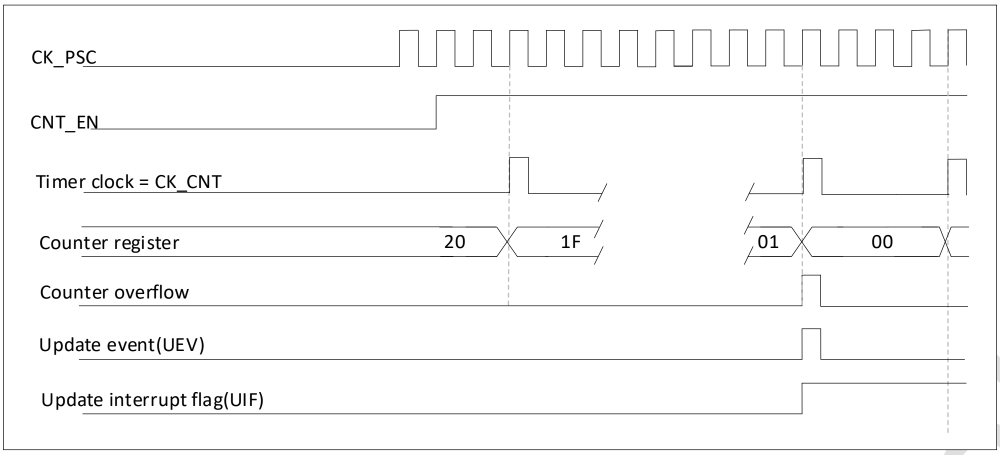


Figure 21-18 Counter timing diagram, internal clock divided by N

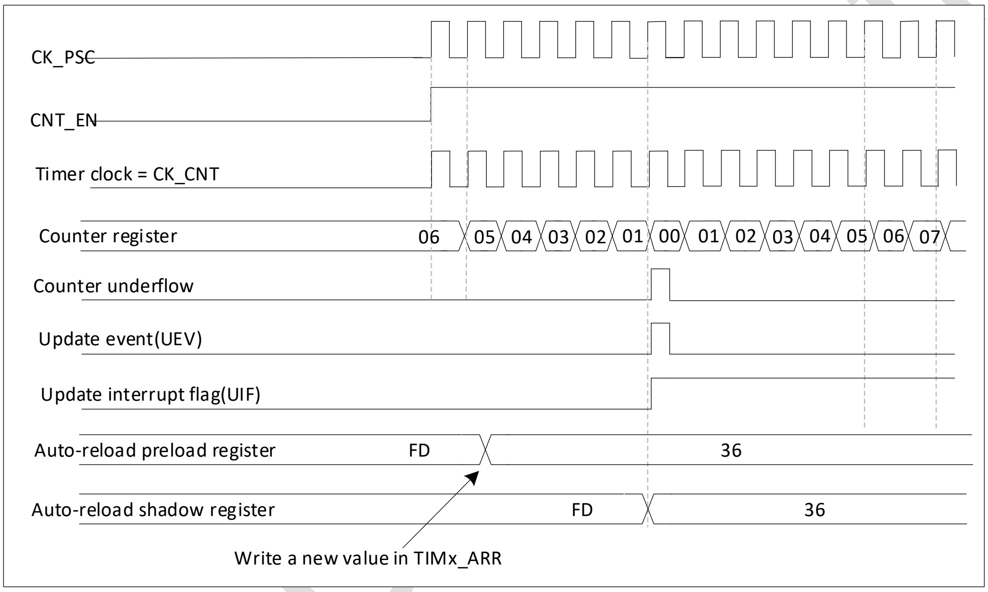


Figure 21-19 Counter timing diagram, update event with ARPE = 1 (counter underflow)

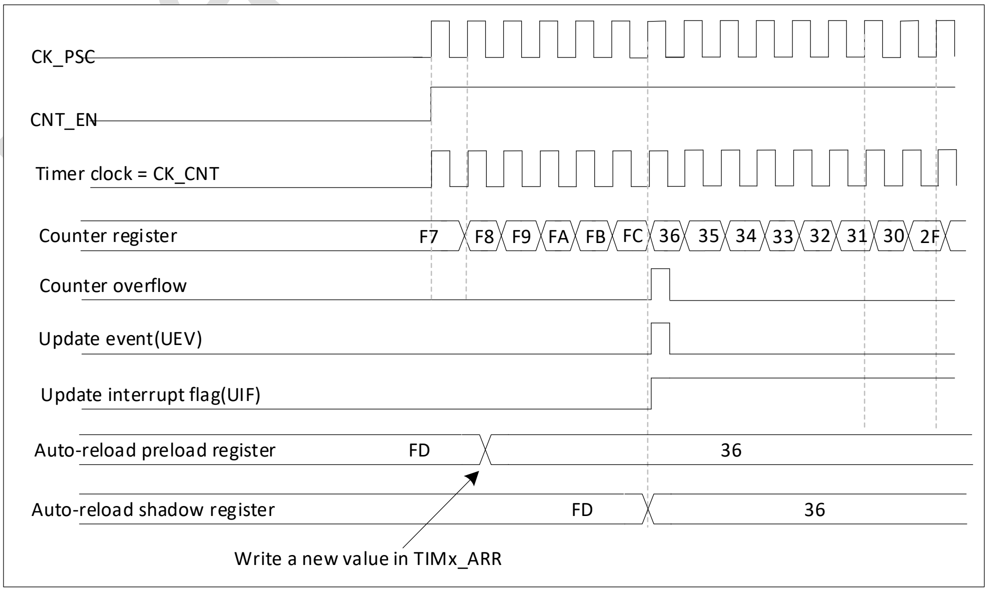


Figure 21-20 Counter timing diagram, Update event with ARPE = 1 (counter overflow)

21.3.3. Repeat 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. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is 2xTck, due to the symmetry of the pattern.

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 center-aligned mode, for odd values of RCR, the update event occurs either on the overflow or on the underflow depending on when the RCR register was written and when the counter was started. If the RCR was written before starting the counter, the UEV occurs on the overflow. If the RCR was written after starting the counter, the UEV occurs on the underflow. For example for RCR = 3, the UEV is generated on each 4th overflow or underflow event depending on when RCR was written.

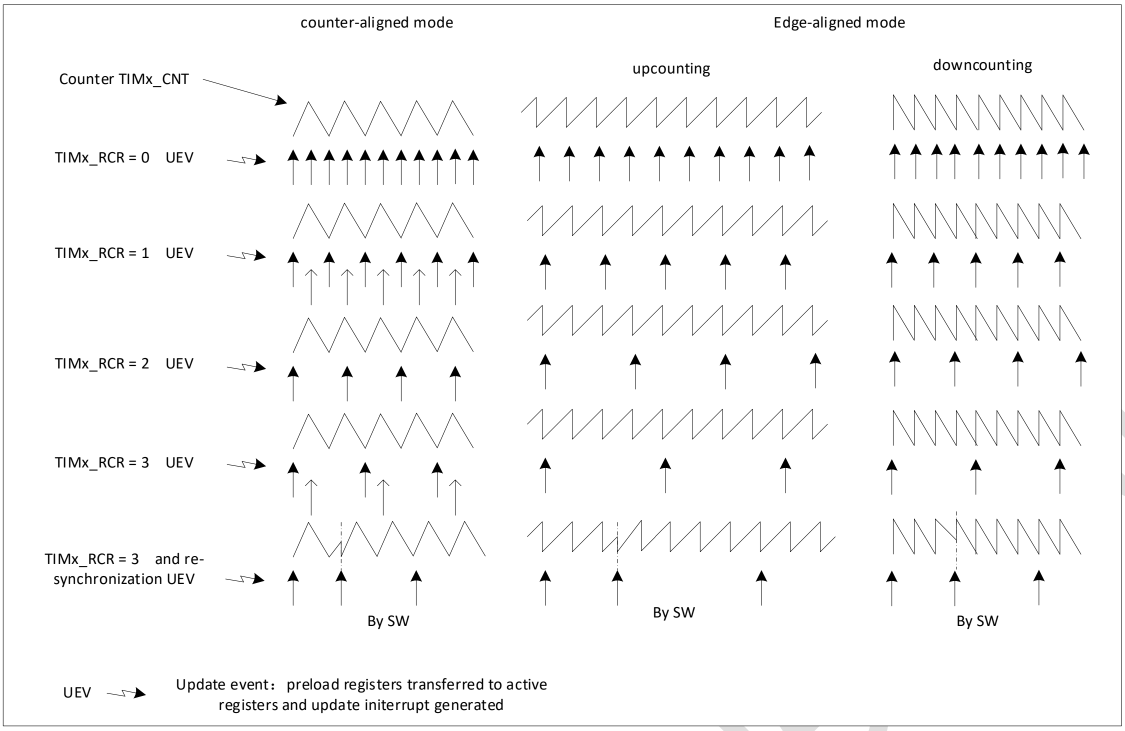


Figure 21-21 Update rate examples depending on mode and TIMx_RCR register settings

21.3.4. Clock selection

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, the user can configure Timer 1 to act as a prescaler for Timer 3.

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS = 000), 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 (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the pre-scaler is clocked by the internal clock CK_INT.

The following diagram shows the operation of the control circuit and the up counter in general mode without prescaler

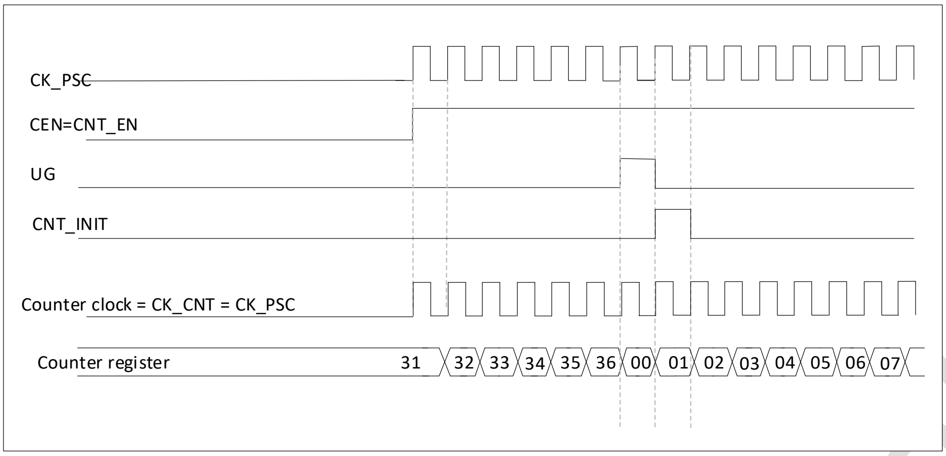


Figure 21-22 Control circuit in normal mode, internal clock divided by 1

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.

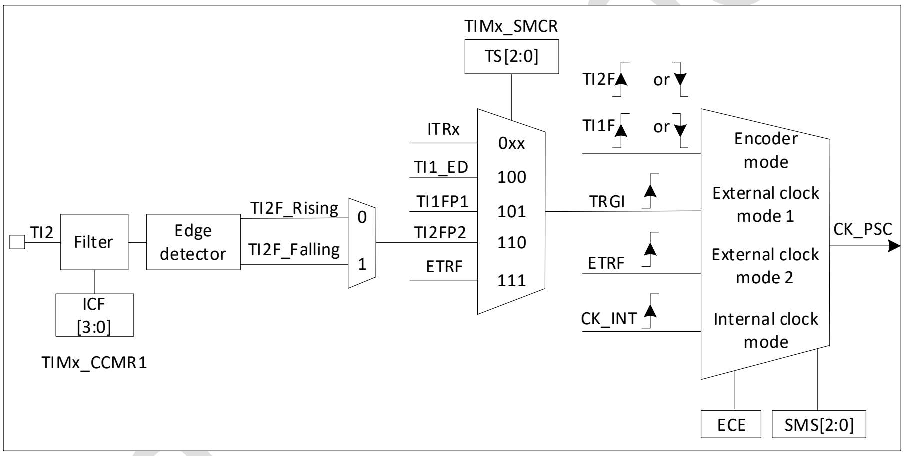


Figure 21-23 TI2 external clock connection example

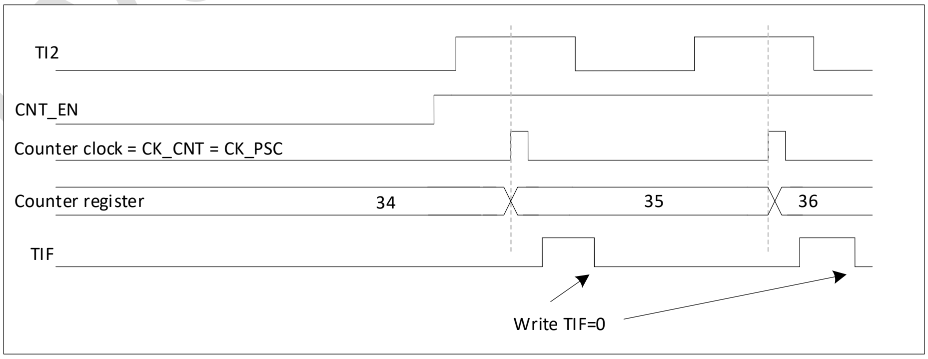


Figure 21-24 Control circuit in external clock mode 1

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.

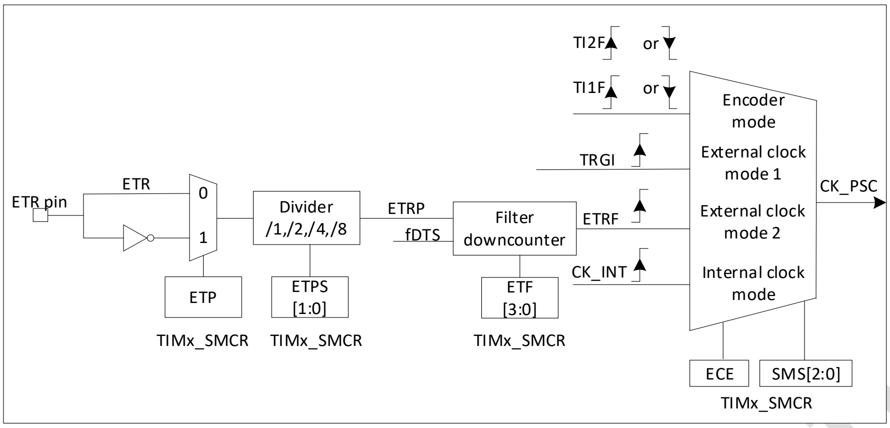


Figure 21-25 External trigger input block

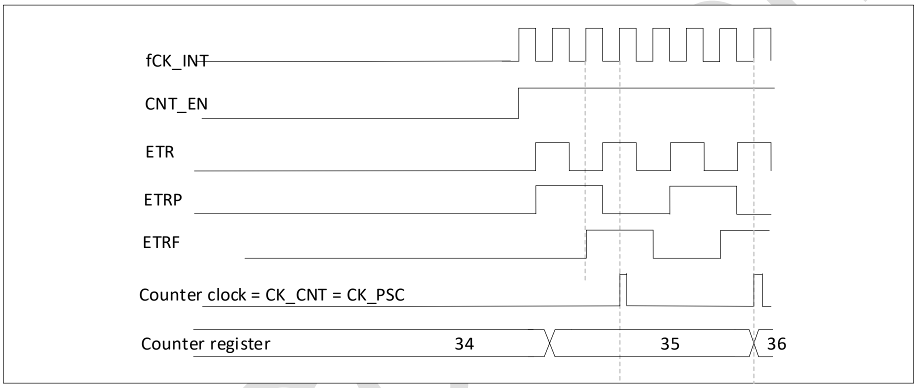


Figure 21-26 Control circuit in external clock mode 2

21.3.5. 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).

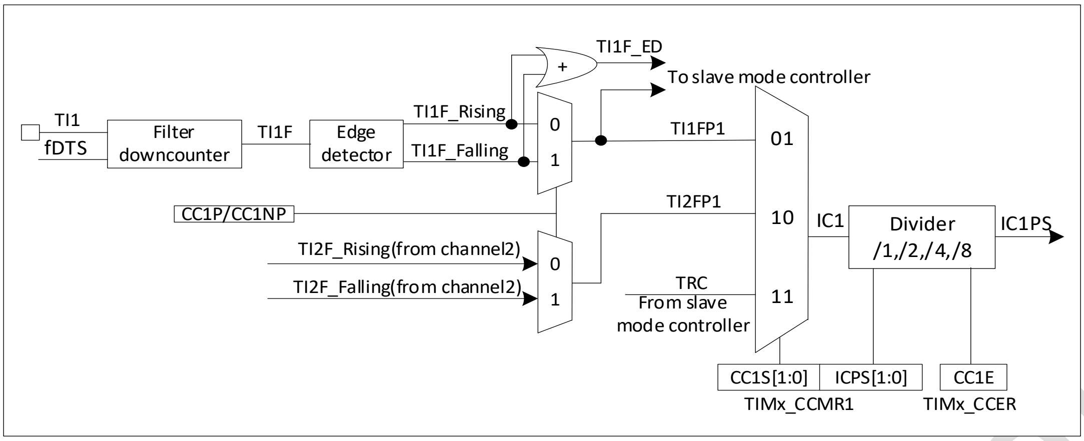


Figure 21-27 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform that is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

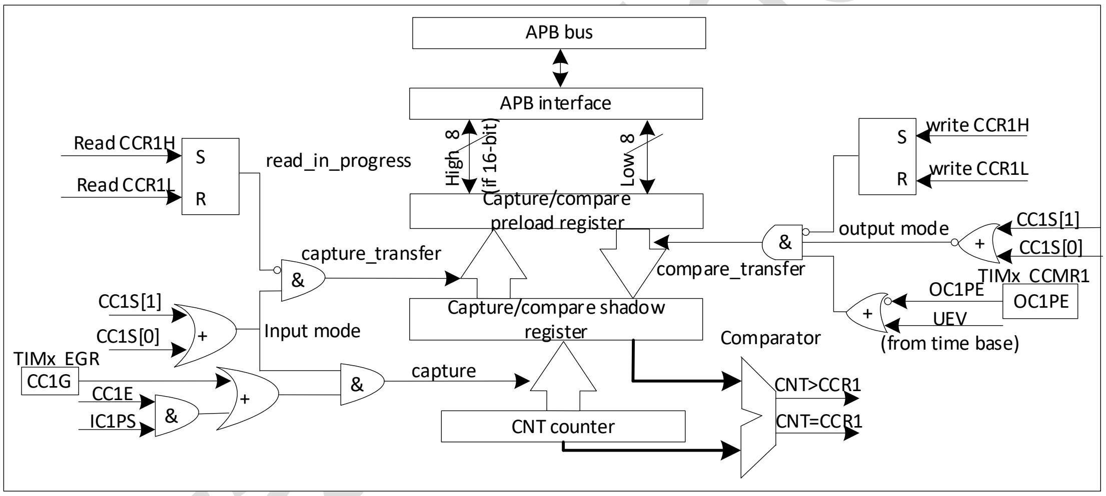


Figure 21-28 Capture/compare channel 1 main circuit

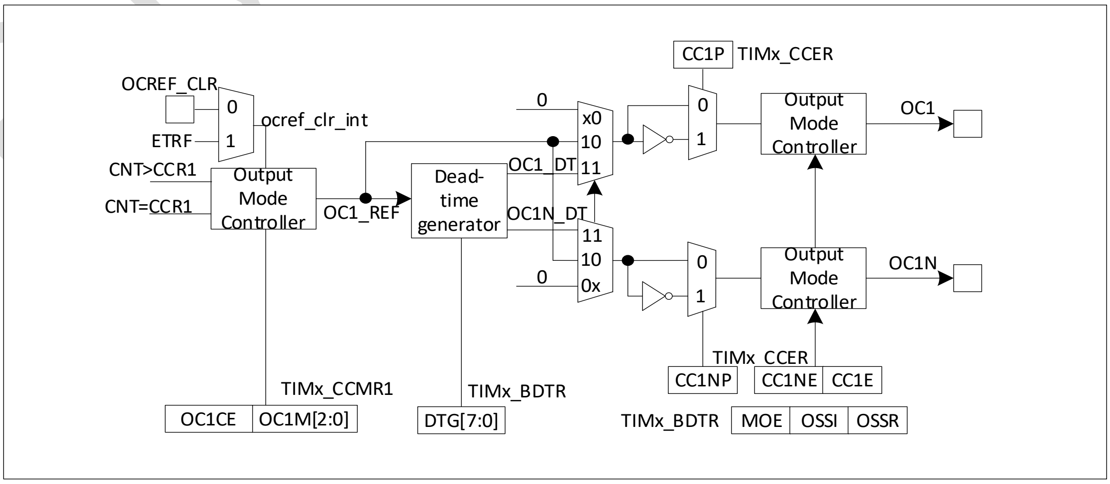


Figure 21-29 Output stage of capture/compare channel (channel 1 to 3)

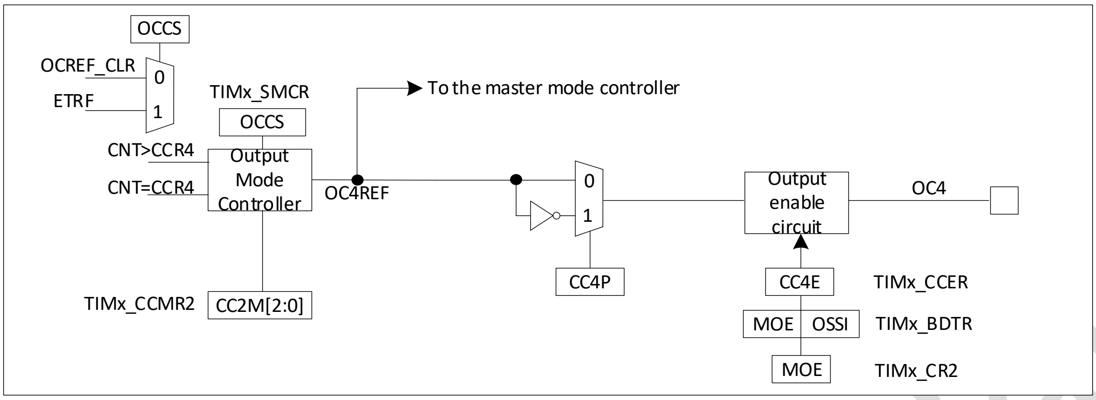


Figure 21-30 Output stage of capture/compare channel (channel 4)

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.

21.3.6. Input capture mode

In Input capture mode, the Capture/Compare registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. 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 written to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input 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 needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at must five internal clock cycles. We must

program a filter duration longer than these five clockcycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.

- Select the 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 needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request 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.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

21.3.7. 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.
- The 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, user 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 CC2P bit to ‘1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

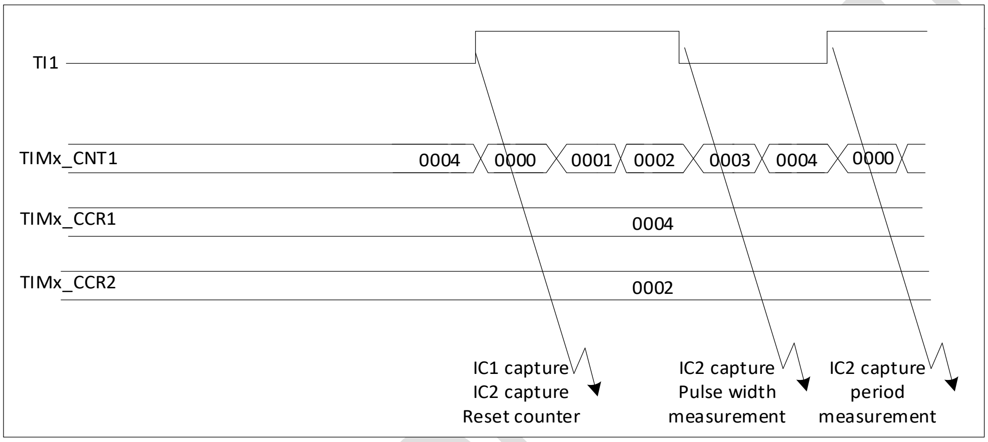


Figure 21-31 PWM input mode timing

21.3.8. Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare 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, the user just needs to write 101 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.

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.

21.3.9. 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 comparison function does the following:

- 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).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

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).

Configuration steps for output comparison mode:

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_CCRxshadow register is updated only at the next update event UEV). An example is given in the figure below.

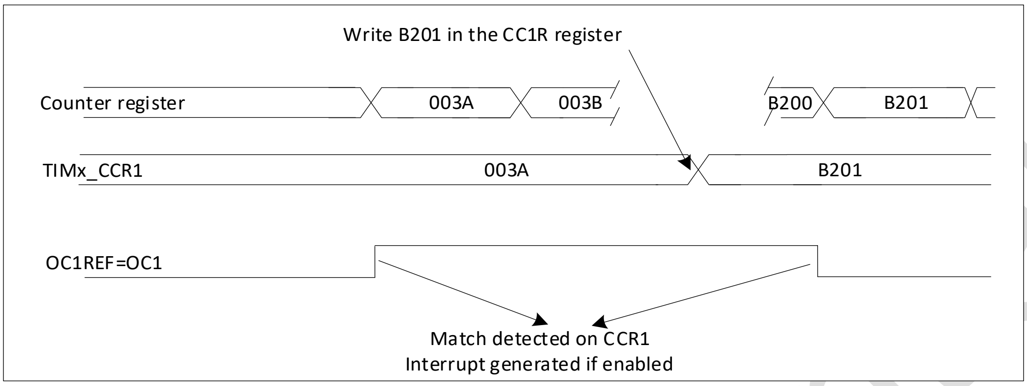


Figure 21-32 Output compare mode, toggle on OC1.

21.3.10. PWM mode

Pulse Width Modulation mode allows generating 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 corresponding preload register must be enabled 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, the user must initialize all the registers 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. Refer to Upcounting mode.

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to Upcounting 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’. The following figure shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

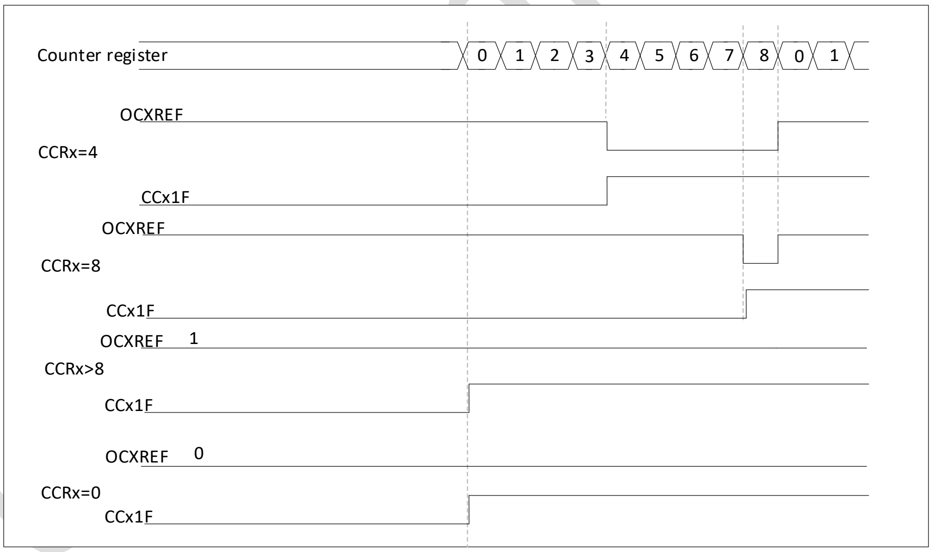


Figure 21-33 Edge-aligned PWM waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_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. Refer to Center-aligned mode (up/down counting).

- 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.

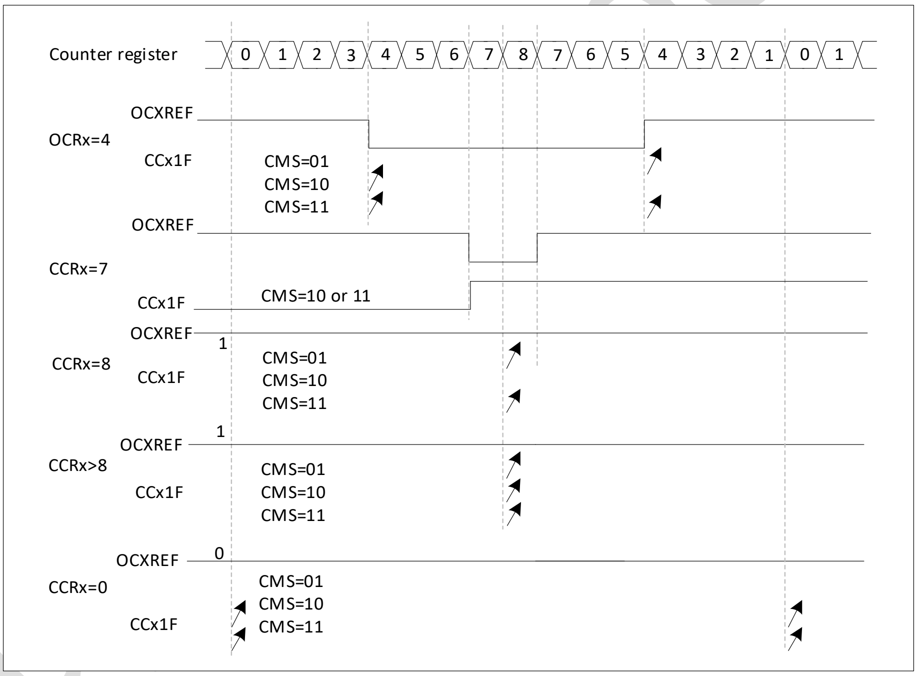


Figure 21-34 Center-aligned PWM waveforms (ARR = 8)

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 the user writes a

value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR). For example, if the counter was counting up, it will continue to count up. – The direction is updated if the user writes 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.

21.3.11. 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 it has to be adjust it depending on the devices connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches).

User 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. Refer to Table 17-3 for more details. 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. DTG[7:0] bits of the TIMx_BDTR register are used to control the dead-time generation for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1 in these examples).

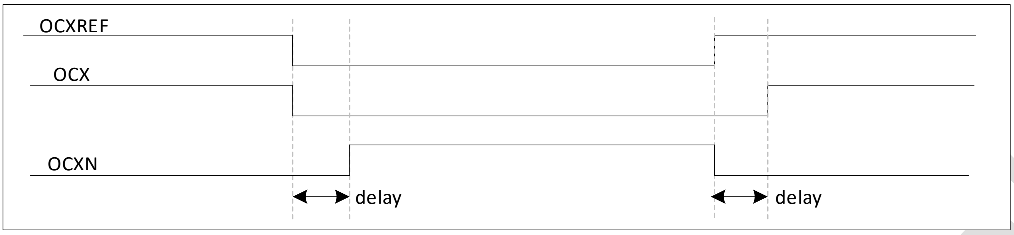


Figure 21-35 Complementary output with dead-time insertion

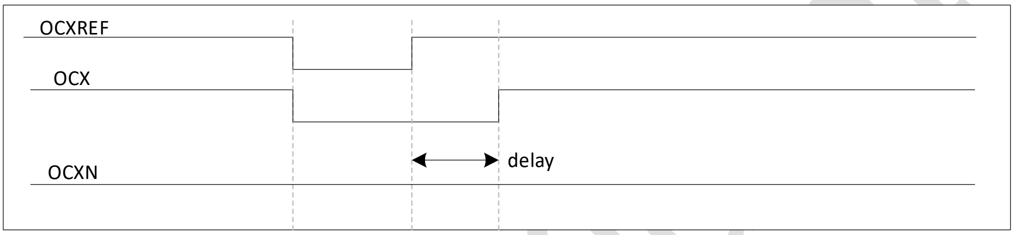


Figure 21-36 Dead-time waveforms with delay greater than the negative pulse

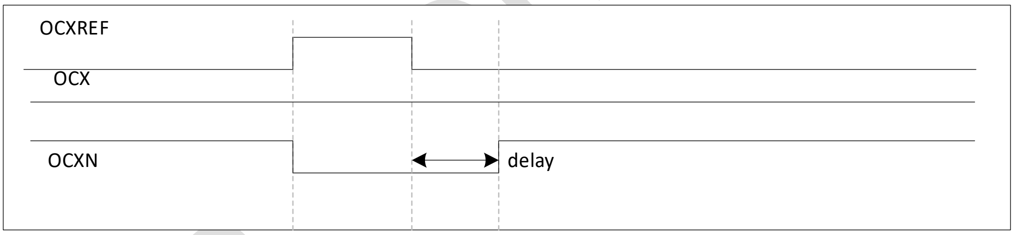


Figure 21-37 Dead-time waveforms with delay greater than the positive pulse

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 the user to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other 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.

21.3.12. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 17-3 for more details.

The brake source can be either the brake input pin, or the following internal sources:

- Output from CPU LOCKUP
- Ouput from PVD
- Clock failure events generated by the Clock Security System (CSS)
- Output from comparator

After reset, the break circuit is disabled and the MOE bit is low. User 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 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, 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 MOE is written to 1 whereas it was low, a delay (dummy instruction) must be inserted before reading it correctly. This is because the user writes an asynchronous signal, but reads a synchronous signal.

When a break occurs (selected level on the break input):

- he MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in 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 because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
  - 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 again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written to ‘1’ again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs is acting 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 cannot 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.

By using 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 freezing the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The user can choose from three 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 following figure shows the example of the output in response to a break.

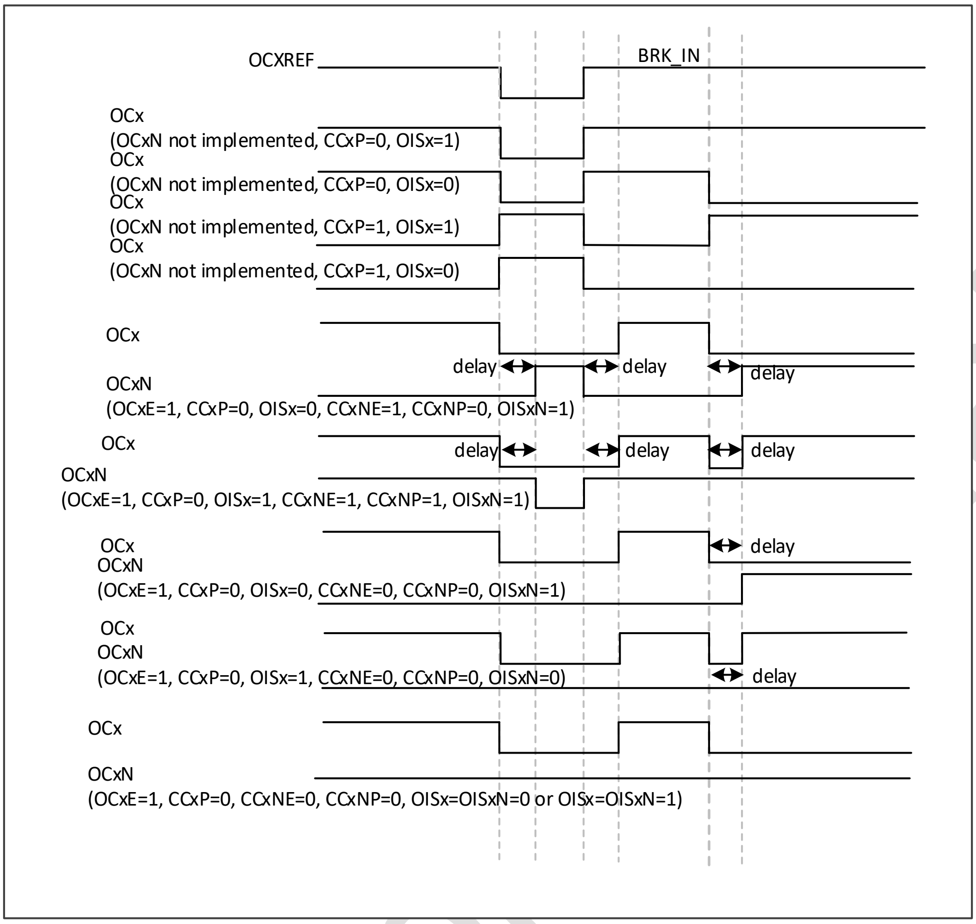


Figure 21-38 Output behavior in response to a break

21.3.13. Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMRx register set to ‘1’). The OCxREF signal remains Low until the next update event of UEV occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the ETR signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to ‘00’.
- The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.

- 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.

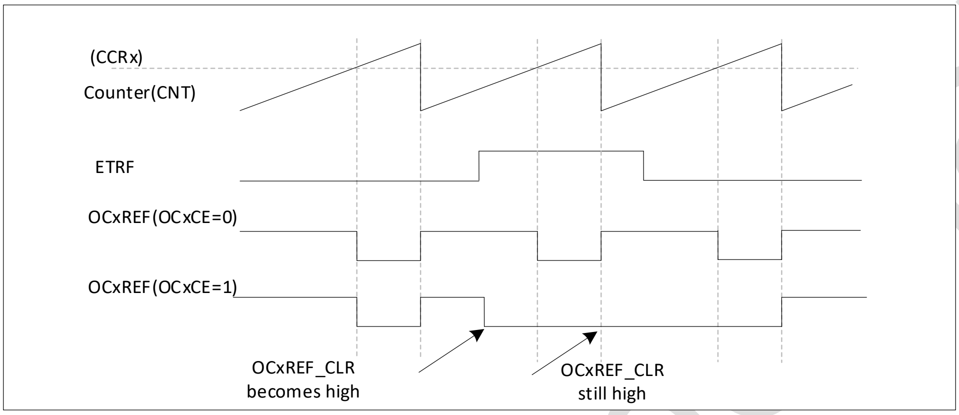


Figure 21-39 Clearing TIM1 OCxREF

21.3.14. 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. The user can thus 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).

A flag is set when the COM event occurs (COMIF bit in the TIMx_SR register), which can generate an interrupt (if the COMIE bit is set in the TIMx_DIER register) or a DMA request (if the COMDE bit is set in the TIMx_DIER register).

The figure blow describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

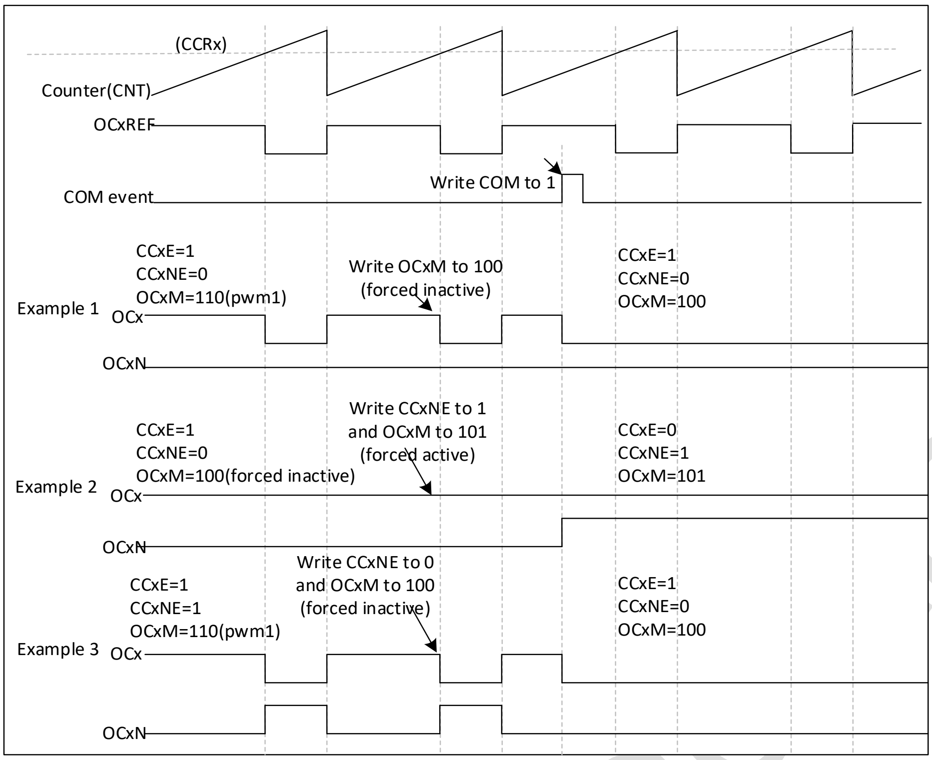


Figure 21-40 6-step generation, COM example (OSSR = 1)

21.3.15. 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 wave-form can be done in output compare mode or PWM mode. Select One-pulse mode 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

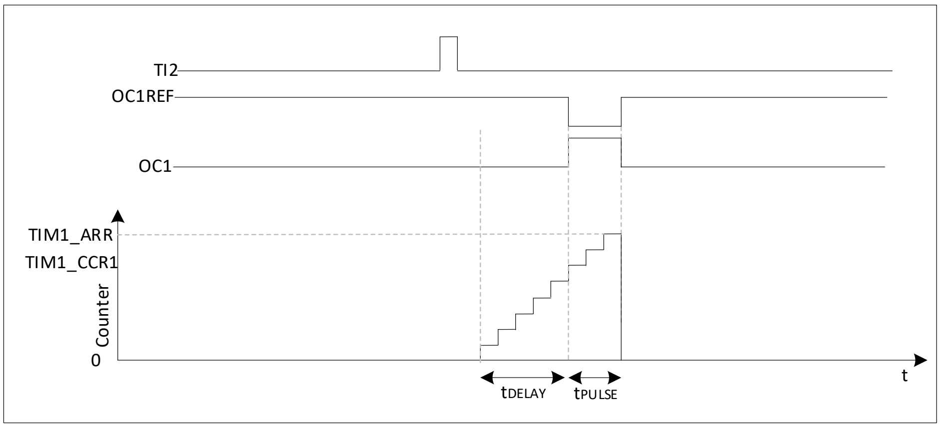


Figure 21-41 Example of one pulse mode

For example the user 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 to 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 (trig-ger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- The tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR-TIMx_CCR1).
- When the user to build a waveform with a transition from '0' to '1' when a compare match oc-curs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this, enable PWM mode 2 by writing OC1M = 111 in the TIMx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE = '1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case the compare value

must be written 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 this example, the DIR and CMS bits in the TIMx_CR1 register should be low.

The user only wants one pulse (Single mode), so '1’ must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0). When OPM bit in the TIMx_CR1 register is set to '0', so the Repetitive Mode is selected.

Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations and it limits the minimum delay tDELAY.

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIMx_CCMRx register must be set. Then OCxRef (and OCx) are 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 PWM2 mode.

21.3.16. 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, the user can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 17-1. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1 = TI1 if not filtered and not inverted,TI2FP2 = TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to ‘1’). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR 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 incremental 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 below summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 21-1 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 be 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 = ‘0’, and IC1F = ‘0000’ (TIMx_CCER register, TI1FP1 non inverted, TI1FP1 = TI1).
- CC2P = ‘0’, and IC2F = ‘0000’ (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).

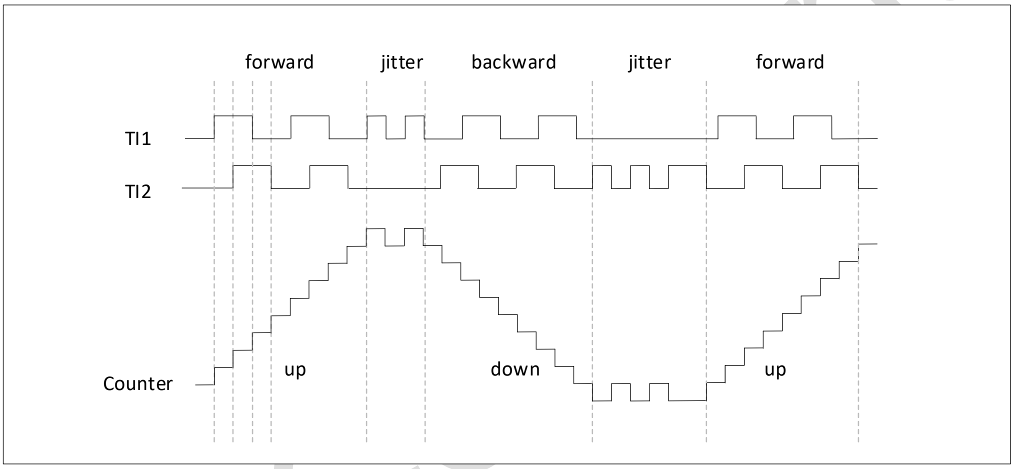


Figure 21-42 Example of counter operation in encoder interface mode.

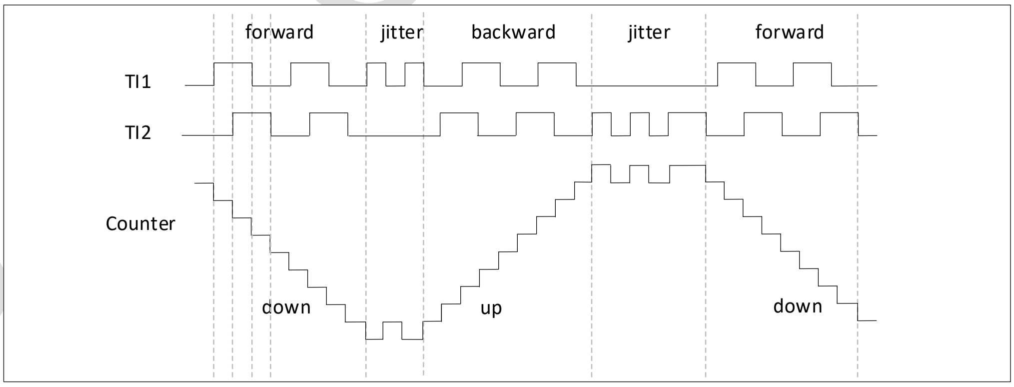


Figure 21-43 Example of encoder interface mode with IC1FP1 inversion

When the timer is configured in encoder interface mode, information about the current position of the sensor is provided. Using a second timer configured in capture mode, the interval between two encoder events can be measured to obtain dynamic information (velocity, acceleration, deceleration). The encoder output indicating the mechanical zero point can be used for this purpose. Depending on the interval between two events, the counter can be read out at a fixed time. If possible, the value of the counter can be latched to a third input capture register (the capture signal must be periodic and can be generated by another timer); it can also be read by a DMA request generated by a real-time clock.

21.3.17. Timer input XOR function

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of 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.

21.3.18. Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIMx (TIM3) referred to as “interfacing timer” in Figure 17-44. The “interfacing timer” captures the 3 timer input pins (TIMx_CH1, TIMx_CH2, and TIMx_CH3) connected through a 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 (see Figure 17-27). 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: the user wants to change the PWM configuration of the advanced-control timer TIM1 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 channel 1 in capture mode (TRC selected): write the CC1S bits in the TIMx_CCMR1 register to ‘11’. The user can also program the digital filter 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 pre-loaded (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 (this can be done in an interrupt subroutine generated by the rising edge of OC2REF).

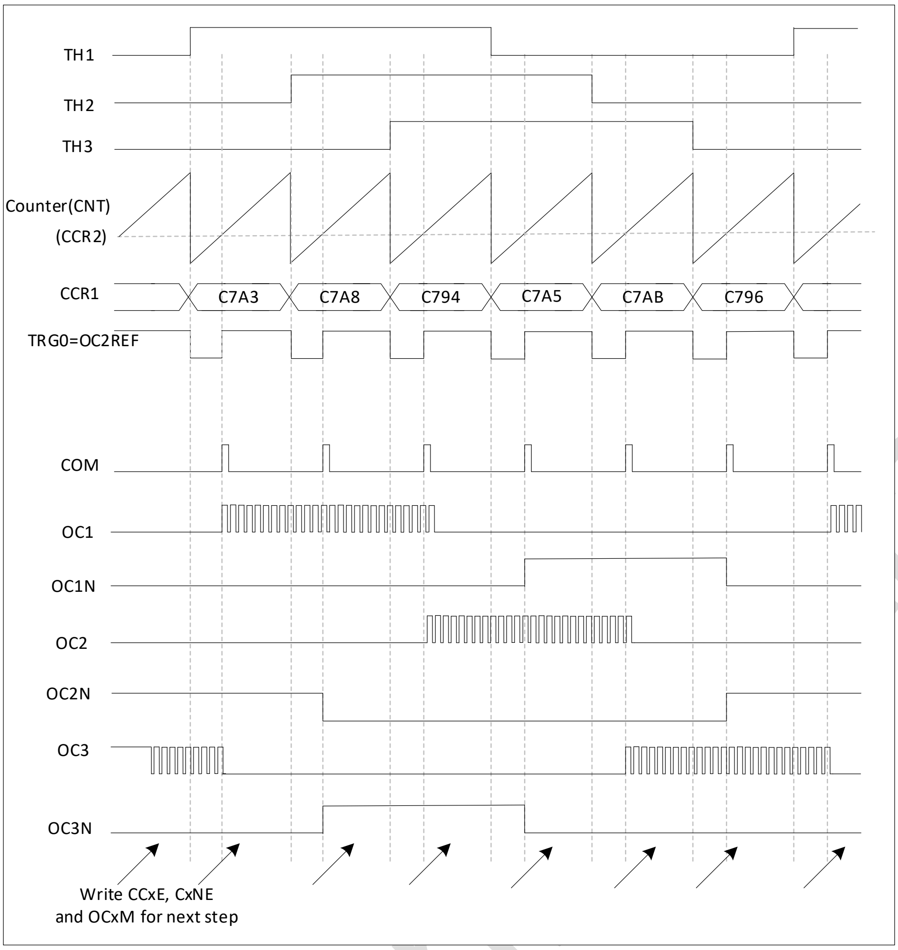


Figure 21-44 Example of Hall sensor interface

21.3.19. TIMx 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 UDIS 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 don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the 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.
- Start 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. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request, or a DMA request can be sent if enabled (depending on the TIE and TDE bits in 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.

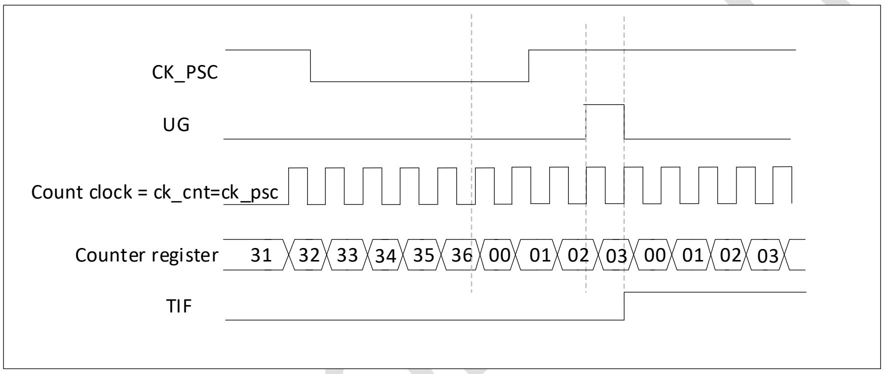


Figure 21-45 Control circuit in reset mode

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 don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so the user does not need to configure it. 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. 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 (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 start or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

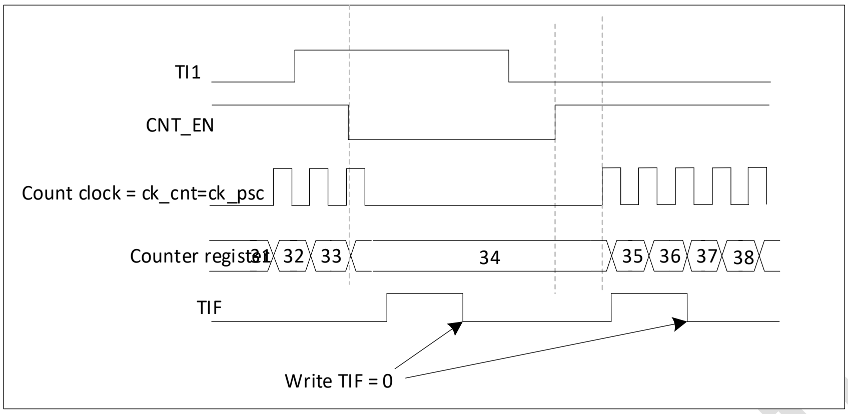


Figure 21-46 Control circuit in gated mode

The counter can start in response to an event on a selected input.

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 don't need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are configured to select the input capture source only, CC2S = 01 in TIMx_CCMR1 register. Write CC2P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIMx_SMCR register. Select TI2 as the input source by writing TS = 110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

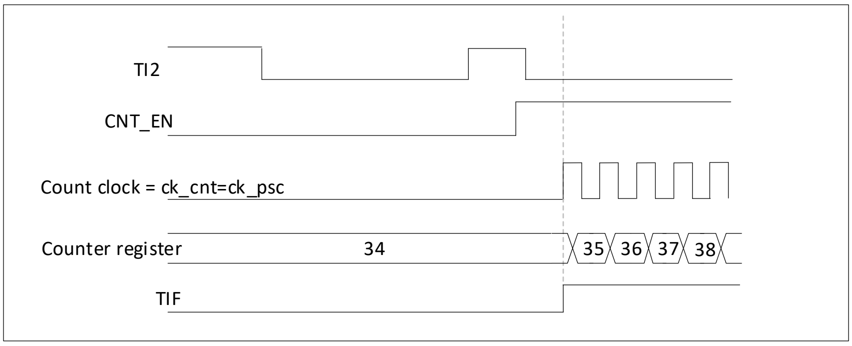


Figure 21-47 Control circuit in trigger mode

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 (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:

- 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
- Configure the channel 1 as follows, to detect rising edges on TI:
  - IC1F = 0000: no filter.
  - The capture prescaler is not used for triggering and does not need to be configured.
  - CC1S = 01 in TIMx_CCMR1 register to select only the input capture source
  - CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edge only).
- 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.

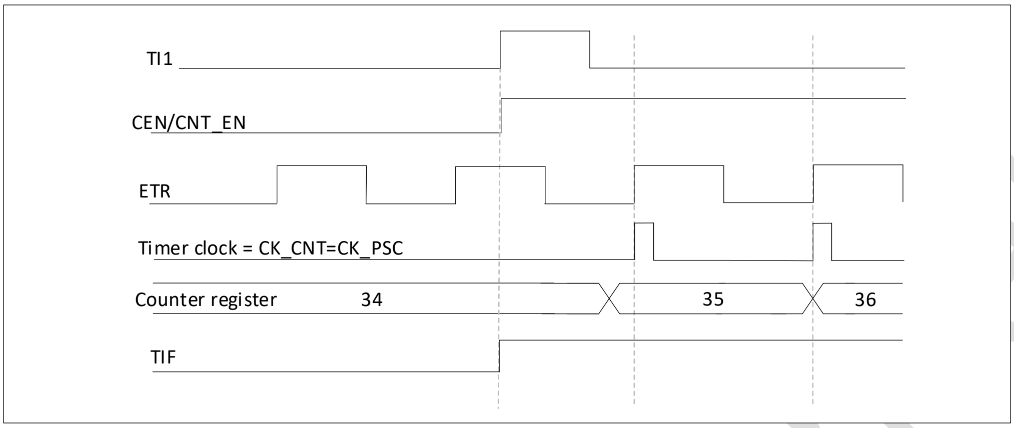


Figure 21-48 Control circuit in external clock mode 2 + trigger mode

21.3.20. Timer synchronization

The TIM timers are linked together internally for timer synchronization or chaining. When a timer is in master mode, it can reset, start, stop or clock the counter of another timer in slave mode.

21.3.21. Debug mode

When the chip enters the debug mode, according to the setting of DBG_TIMx_STOP in the DBG module, the TIMx counter can continue to work normally or stop working.

21.4. TIM1 registers

21.4.1. TIM1 control register1 (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	-	-	-
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (tDTS)used by the dead-time generators and the digital filters (ETR, TIx), 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved, do not program this value
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIMx_CCMRx register) are set both when the counter is counting up or 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	Counting direction 0: Counter counts up 1: Counter counts down Note: When the counter is configured in center alignment mode or encoder mode, this bit is read-only
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: Any of the following events generate an update interrupt or DMA request if enabled.

Bit	Name	R/W	Reset Value	Function
				These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
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.

21.4.2. TIM1 control register2 (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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	0	Reserved, must be kept at reset value.
14	OIS4	RW	0	Output Idle state 4 (OC4 output) refer to OIS1 bit
13	OIS3N	RW	0	Output Idle state 3 (OC3N output) refer to OIS1N bit
12	OIS3	RW	0	Output Idle state 3 (OC3 output) refer to OIS1 bit
11	OIS2N	RW	0	Output Idle state 2 (OC2N output) refer to OIS1N bit
10	OIS2	RW	0	Output Idle state 2 (OC2 output) refer to OIS1 bit
9	OIS1N	RW	0	Output Idle state 1 (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 as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR 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 cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register)
7	TI1S	RW	0	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6:4	MMS[2:0]	RW	000	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 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 enable. The Counter Enable signal is generated by a logic OR 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 TIMx_SMCR register).

Bit	Name	R/W	Reset Value	Function
				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	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
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 COMG bit only 1: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COMG bit or when an rising edge occurs on TRGI *Note: This bit acts only on channels that have a complementary output.*
1	Res	-	0	Reserved, must be kept at reset value.
0	CCPC	RW	0	CCPC: Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a communication event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

21.4.3. 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	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	-	-	-
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or $\overline{ETR}$ 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. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS = 111 and TS = 111). 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13：12	ETPS[1:0]	RW	00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. 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 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, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6

Bit	Name	R/W	Reset Value	Function
				0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
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: Internal Trigger 0 (ITR0) 001: Reserved 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: External Trigger input (ETRF) Note: These bits must be changed only when they are not used (e.g. when SMS = 000) 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 the 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 TI2FP1 edge depending on TI1FP2 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP2 edge depending on TI2FP1 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input.

Bit	Name	R/W	Reset Value	Function
				100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - 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_ED 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. Note: The clock of the slave timer must be enabled prior to receive events from the master timer, and must not be changed on-the-fly while triggers are received from the master timer.

Table 21-2 TIM1 Internal trigger connection

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	TIM15_TRGO	TIM2_TRGO	TIM3_TRGO	TIM17_TRGO

21.4.4. 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, must be kept at reset value.
14	TDE	RW	0	TDE: Trigger DMA request enable 0: Trigger DMA request disabled

Bit	Name	R/W	Reset Value	Function
				1: Trigger DMA request enabled
13	COMDE	RW	0	COMDE: COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled
12	CC4DE	RW	0	CC4DE: Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	RW	0	CC3DE: Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	RW	0	CC2DE: Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	RW	0	CC1DE: Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled 1: Update DMA request 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
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	RW	0	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

21.4.5. 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	CC1IF	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
31:13	Reserved	-	0	Reserved, always 0
23	IC4IF	RC_W0	0	Falling edge capture 4 flag See IC1IF description.
22	IC3IF	RC_W0	0	Falling edge capture 3 flag See IC1IF description.
21	IC2IF	RC_W0	0	Falling edge capture 2 flag See IC1IF description.
20	IC1IF	RC_W0	0	Falling Edge Capture 1 Flag This flag can be set to 1 by hardware only when the corresponding channel is configured for input capture and the capture event is triggered by a falling edge. it is cleared '0' by software or '0' by reading TIMx_CCR1. 0: no repeat captures are generated; 1: falling edge capture event occurs.
19	IC4IR	RC_W0	0	Rising edge capture 4 flag See IC1IR description.
18	IC3IR	RC_W0	0	Rising edge capture 3 flag See IC1IR description.
17	IC2IR	RC_W0	0	Rising edge capture 2 flag See IC1IR description.
16	IC1IR	RC_W0	0	Rising Edge Capture 1 Flag This flag can be set to 1 by hardware only when the corresponding channel is configured for input capture and the capture event is triggered by a rising edge. it is cleared '0' by software or '0' by reading TIMx_CCR1. 0: no repeat captures are generated; 1: Rising edge capture event occurs.
15:13	Reserved	-	-	Reserved
12	CC4OF	RC_W0	0	Capture/Compare4 Over Capture Marker

Bit	Name	R/W	Reset Value	Function
				See CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 Over Capture Marker See CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare2 Over Capture Marker See CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 Over Capture Flag This flag can be set to 1 by hardware only when the corresponding channel is configured for input capture. writing 0 clears this bit. 0: no overcapture is generated; 1: When CC1OF is set to 1, the value of the counter has been captured to the TIM1_CCR1 register.
8	Res	RC_W0	0	Reserved, always read as 0.
7	BIF	RC_W0	0	Brake interrupt flag Once the brake input is valid, this bit is cleared by hardware to position 1. If the brake input is invalid, this bit can be cleared by software to 0. 0: no brake event is generated; 1: Valid level detected on the brake input.
6	TIF	RC_W0	0	Trigger interrupt flag The trigger event (when a valid edge is detected at the TRGI input when the slave controller is in a mode other than gated mode, or any edge in gated mode) is set to 1 by hardware. It is cleared to 0 by software. 0: no trigger event is generated; 1: Trigger interrupt waiting for response
5	COMIF	RC_W0	0	COM interrupt flag Once a COM event is generated (when CcxE, CcxNE, OCxM has been updated) this bit is set to 1 by hardware. it is cleared to 0 by software. 0: no COM event is generated; 1: COM interrupt waiting for response
4	CC4IF	RC_W0	0	Capture/Compare4 Interrupt Marker Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 Interrupt Flag Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare2 Interrupt Flag Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare1 Interrupt Flag If channel CC1 is configured in output mode: This bit is set to 1 by hardware when the counter value matches the comparison value, except in centrosymmetric mode (refer to the TIM1_CR1 register

Bit	Name	R/W	Reset Value	Function
				CMS bit of the TIM1_CR1 register). It is cleared by software to 0. 0: no match occurs; 1: the value of TIM1_CNT matches the value of TIM1_CCR1. If channel CC1 is configured in input mode: This bit is set to 1 by hardware when a capture event occurs, it is cleared by software to 0 or by reading TIM1_CCR1 to 0. 0: no input capture is generated; 1: input capture is generated and the counter value is loaded into TIM1_CCR1 (edge of the same polarity as selected is detected on IC1). Note: When CEN is turned on, this bit is also set.
0	UIF	RC_W0	0	Update Interrupt Flag This bit is set to 1 by hardware when an update event is generated. it is cleared to 0 by software. 0: no update event is generated; 1: Update event waiting for response. This bit is set to 1 by hardware when the register is updated. - If UDIS=0 in the TIM1_CR1 register, an update event is generated when REP_CNT=0 (when the repeat down counter overflows or underflows); - If UDIS=0 and URS=0 of TIM1_CR1 register, the update event is generated when UG=1 of TIM1_EGR register. event (software reinitialization of CNT); - If UDIS=0 and URS=0 of TIM1_CR1 register, the update event is generated when the CNT is reinitialized by the trigger event. event. (Refer to the Slave Mode Control Register (TIM1_SMCR))

21.4.6. 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	-	0	Reserved, must be kept at reset value.
7	BG	W	0	Generate Brake Event This bit is set to 1 by software to generate a brake event and is automatically cleared to 0 by hardware. 0: no action; 1: Generate a brake event. At this time, MOE=0, BIF=1, if the corresponding interrupt is turned on, the corresponding interrupt 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: The TIF flag is set in TIMx_SR register. If the corresponding interrupt is turned on, the corresponding interrupt 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 having a complementary output.
4	CC4G	W	0	CC4G: Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	CC3G: Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	CC2G: 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 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware.

Bit	Name	R/W	Reset Value	Function
				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). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIMx_ARR) if DIR = 1 (downcounting).

21.4.7. TIM1 capture/compare mode register1 (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	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**
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	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, must be kept at reset value.
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 is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7	OC1CE	RW	0	Output Compare 1 clear enable

Bit	Name	R/W	Reset Value	Function
				OC1CE: Output Compare 1 Clear Enable 0: OC1Ref is not affected by the ETRF Input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input
6:4	OC1M[2:0]	RW	00	Output Compare 1 mode These three bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs (this mode is used to generate a timing base). 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 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = ‘0’) as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = ’1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: 1: 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). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode. 3: On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the OC1M active bits take the new value from the preloaded bits only when a COM event is generated.

Bit	Name	R/W	Reset Value	Function
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note: 1: 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). 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). 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 CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. 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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_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, must be kept at reset value.
15:12	IC2F	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 (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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER)
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: $f_{SAMPLING} = f_{CK\_INT}$, N = 2 0010: $f_{SAMPLING} = f_{CK\_INT}$, N = 4 0011: $f_{SAMPLING} = f_{CK\_INT}$, N = 8 0100: $f_{SAMPLING} = f_{DTS/2}$, N = 6 0101: $f_{SAMPLING} = f_{DTS/2}$, N = 8 0110: $f_{SAMPLING} = f_{DTS/4}$, N = 6 0111: $f_{SAMPLING} = f_{DTS} / 4$, N = 8 1000: $f_{SAMPLING} = f_{DTS} / 8$, N = 6 1001: $f_{SAMPLING} = f_{DTS} / 8$, N = 8 1010: $f_{SAMPLING} = f_{DTS} / 16$, N = 5 1011: $f_{SAMPLING} = f_{DTS} / 16$, N = 6 1100: $f_{SAMPLING} = f_{DTS} / 16$, N = 8 1101: $f_{SAMPLING} = f_{DTS} / 32$, N = 5 1110: $f_{SAMPLING} = f_{DTS} / 32$, N = 6 1111: $f_{SAMPLING} = f_{DTS} / 32$, N = 8

Bit	Name	R/W	Reset Value	Function
				Note: Care must be taken that fDTS is replaced in the formula by CK_INT when ICxF[3:0] = 1, 2 or 3.
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’ (TIMx_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	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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

21.4.8. 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	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
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: 16	Reserved	-		Reserved, must be kept at reset value.
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 TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIMx_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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIMx_CCER)

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved, must be kept at reset value.
15:12	IC4F[3:0]	RW	0000	Input capture 4 filter
11:10	IC4PSC[1:0]	RW	00	Input capture 4 prescaler
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

Bit	Name	R/W	Reset Value	Function
				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 TS bit (TIMx_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 3 filter
3:2	IC3PSC[1:0]	RW	00	Input capture 3 prescaler
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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER).

21.4.9. 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	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	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	-	0	Reserved, must be kept at reset value.
13	CC4P	RW	0	Capture/Compare 4 output polarity refer to CC1P description
12	CC4E	RW	0	Capture/Compare 4 output enable refer to CC1E description

Bit	Name	R/W	Reset Value	Function
11	CC3NP	RW	0	Capture/Compare 3 complementary output polarity refer to CC1NP description
10	CC3NE	RW	0	Capture/Compare 3 complementary output enable refer to CC1NE description
9	CC3P	RW	0	Capture/Compare 3 output polarity refer to CC1P description
8	CC3E	RW	0	Capture/Compare 3 output enable refer to CC1E description
7	CC2NP	RW	0	Capture/Compare 2 complementary output polarity refer to CC1NP description
6	CC2NE	RW	0	Capture/Compare 2 complementary output enable refer to CC1NE description
5	CC2P	RW	0	Capture/Compare 2 output polarity refer to CC1P description
4	CC2E	RW	0	Capture/Compare 2 output enable refer to CC1E description
3	CC1NP	RW	0	Input/capture 1 complementary output polarity 0: OC1N high active 1: OC1N active low Note: This bit cannot be modified once the LOCK level (LCCK bit in TIM1_BDTR register) is set to 3 or 2 and CC1S=00 (channel is configured as output).
2	CC1NE	RW	0	Input / Capture 1 Complementary Output Enable 0: Off - OC1N output is disabled, so the output level of OC1N depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, CC1E bits. 1: ON- OC1N signal is output to the corresponding output pin, its output level depends on the value of 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 CC1NP/CC1P bits select the polarity of TI1FP1 and TI2FP1 as trigger or capture signals. 00: Non-inverting/rising edge: TIxFP1 rising edge active (capture, trigger in reset mode, external clock or in trigger mode); TIxFP1 not inverted (gated mode, encoder mode). 01: Inverted/falling edge: TIxFP1 falling edge valid (capture, trigger in reset mode, external clock or trigger mode); TIxFP1 inverted (gated mode, encoder mode). 10: Reserved, do not use this configuration.

Bit	Name	R/W	Reset Value	Function
				11: Not inverted/dual edge Both rising and falling edges of TIxFP1 are active (capture, trigger in reset mode, external clock or trigger mode); TIxFP1 is not inverted (gated mode). This configuration cannot be applied in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit in the TIMx_CR2 register is set, then the actual valid bits of CC1P are loaded with preloaded values only when a com event occurs. 2. 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 The CC1 channel is configured to output: 0: Off- OC1 output is disabled, so the output level of OC1 depends on the values of MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits. 1: On- OC1 signal is output to the corresponding output pin, its output level depends on the value of MOE, OSSI, OSSR, OIS1, OIS1N, CC1NE bits. The CC1 channel is configured as an input: This bit determines whether the counter value can be captured into the TIM1_CCR1 register. 0: Capture Disable 0: Capture enable Notes: For complementary output channels, this bit is preloaded. If the CCPC bit in the TIMx_CR2 register is set, then the actual valid bits of CC1E are loaded with preloaded values only when a com event occurs.

Table21-3 Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states(1)	
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 = OCxREF xor CCxP,	Output Disabled (not driven by the timer)

Control bits					Output states(1)	
MOE	**OSSI**	**OSSR**	**CcxE**	**CcxNE**	**OCx output state**	**OCxN output state**
					OCx_EN = 1	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	Off-State (output enabled with inactive state) 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 OCxN_EN = 1
0	0	x	0	0	Output Disabled (not driven by the timer) Asynchronously: OCx = CCxP, OCx_EN = 0, OCxN = CCxNP, OCxN_EN = 0 Then if the clock is present: 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		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1 Then if the clock is present: 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		

21.4.10. TIM1 counter (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, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

21.4.11. 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, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

21.4.12. TIM1 auto-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, must be kept at reset value.
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.

21.4.13. 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, must be kept at reset value.
7:0	REP[7:0]	RW	0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. 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.

21.4.14. 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, must be kept at reset value.
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).

21.4.15. 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, must be kept at reset value.
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 TIMx_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 TIMx_CNT and signalled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

21.4.16. 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, must be kept at reset value.
15:0	CCR3[15:0]	RW	0	Capture/Compare 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 TIMx_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 TIMx_CNT and signalled on OC3 output. **If channel CC3 is configured as input:** CCR3 is the counter value transferred by the last input capture 3 event (IC3).

21.4.17. 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, must be kept at reset value.
15:0	CCR4[15:0]	RW	0	Capture/Compare value

Bit	Name	R/W	Reset Value	Function
				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 TIMx_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 TIMx_CNT and signalled on OC4 output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

21.4.18. 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	RW	0	Reserved, must be kept at reset value.
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically 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 TIMx_CCER register).
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 the break input is not be active) Note: This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	RW	0	Brake input polarity

Bit	Name	R/W	Reset Value	Function
				0: Brake input active low; 1: Brake input active high. Notes: 1、Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified. 2, Any write operation to this bit requires a delay of one APB clock before it can take effect.
12	BKE	RW	0	Brake function enable 0: brake input (BRK and BRK_ACTH) is disabled; 1: Brake input (BRK and BRK_ACTH) is enabled. Notes: 1、Once the LOCK level (LOCK bit in TIM1_BDTR register) is set to 1, this bit cannot be modified. 2、Any write operation to this bit requires a delay of one APB clock before it can take effect.
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 is not implemented if no complementary output is implemented in the timer. 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. Then, OC/OCN enable output signal = 1 Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are forced first with 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 soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_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 bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register and

Bit	Name	R/W	Reset Value	Function
				BKE/BKP/AOE bits in TIMx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_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 TIMx_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 TIMx_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]x tdtg with tdtg = tDTS. DTG[7:5] = 10x = > DT = (64+DTG[5:0])xtdtg with Tdtg = 2xtDTS. DTG[7:5] = 110 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 8xtDTS. DTG[7:5] = 111 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 16xtDTS. Example if TDTS = 125 ns (8 MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 us to 31750 ns by 250 ns steps, 32 us to 63 us by 1 us steps, 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 TIMx_BDTR register).

21.4.19. 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, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the number of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIMx_DMAR address) 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers ... 10001: 18 transfers
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address This 5-bit vector defines the base-address for DMA transfers (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. Example: 00000: TIMx_CR1, 00001: TIMx_CR2, 00010: TIMx_SMCR, ... **Example:** Let us consider the following transfer: DBL = 7 transfers and DBA = TIMx_CR1. In this case the transfer is done to/from 7 registers starting from the TIMx_CR1 address.

21.4.20. TIM1 DMA address for full transfer (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	-	-	-
15:0	DMAB[15:0]	RW	0	DMA Continuous Transfer Register A read or write to the TIMx_DMAR register results in an access operation to the register at the following address: TIMx_CR1 address + DBA + DMA pointer, where: "TIMx_CR1 address" is the address of control register 1;

Bit	Name	R/W	Reset Value	Function
				"DBA" is the base address defined in the TIMx_DCR register; The "DMA pointer" is an offset automatically controlled by the DMA, which depends on the DBL defined in the TIMx_DCR register.

21.4.21. TIM1 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM1_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x04	TIM1_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS [2:0]			CCDS	CCUS	Res.	CCPC
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0		0
0x08	TIM1_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS [2:0]		
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIM1_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	TIM1_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4IR	IC3IR	IC2IR	IC1IR	IC4IF	IC3IF	IC2IF	IC1IF	Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	0	0		0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x14	TIM1_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																									0	0	0	0	0	0	0	0
0x18	TIM1_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M [2:0]			OC2PE	CO2FE	CC2S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIM1_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIM1_CCMR2(output capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M [2:0]			OC4PE	CO4FE	CC4S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3S [1:0]	
	Reset																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x1C	TIM1_CCMR2(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIM1_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	Reset value																									0	0	0	0	0	0	0	0
0x24	TIM1_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM1_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM1_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30	TIM1_RCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x34	TIM1_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIM1_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIM1_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIM1_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TIM1_BDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	TIM1_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.			DBA[4:0]				

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
48	Re-set value																				0	0	0	0	0				0	0	0	0	0
0x4C	TIM1_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Re-set value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22. General-purpose timer (TIM 2/3)

22.1. TIM2/TIM3 introduction

The general-purpose timer TIM3 consist of a 16-bit auto-reload counter driven by a programmable prescaler. The general-purpose timer TIM2 consist of a 32-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 and 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 timers are completely independent, and do not share any resources. They can be synchronized together

22.2. TIM 2/3 main features

General-purpose TIM 2/3 timer features include:

- 16-bit (TIM3), 32-bit (TIM2) up, down, up/down auto-reload counter.
- 16-bit programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge- and Center-aligned modes)
  - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect several timers.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)

- ➢ Input capture
- ➢ Output compare

- ■ Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes.
- ■ Trigger input for external clock or cycle-by-cycle current management.

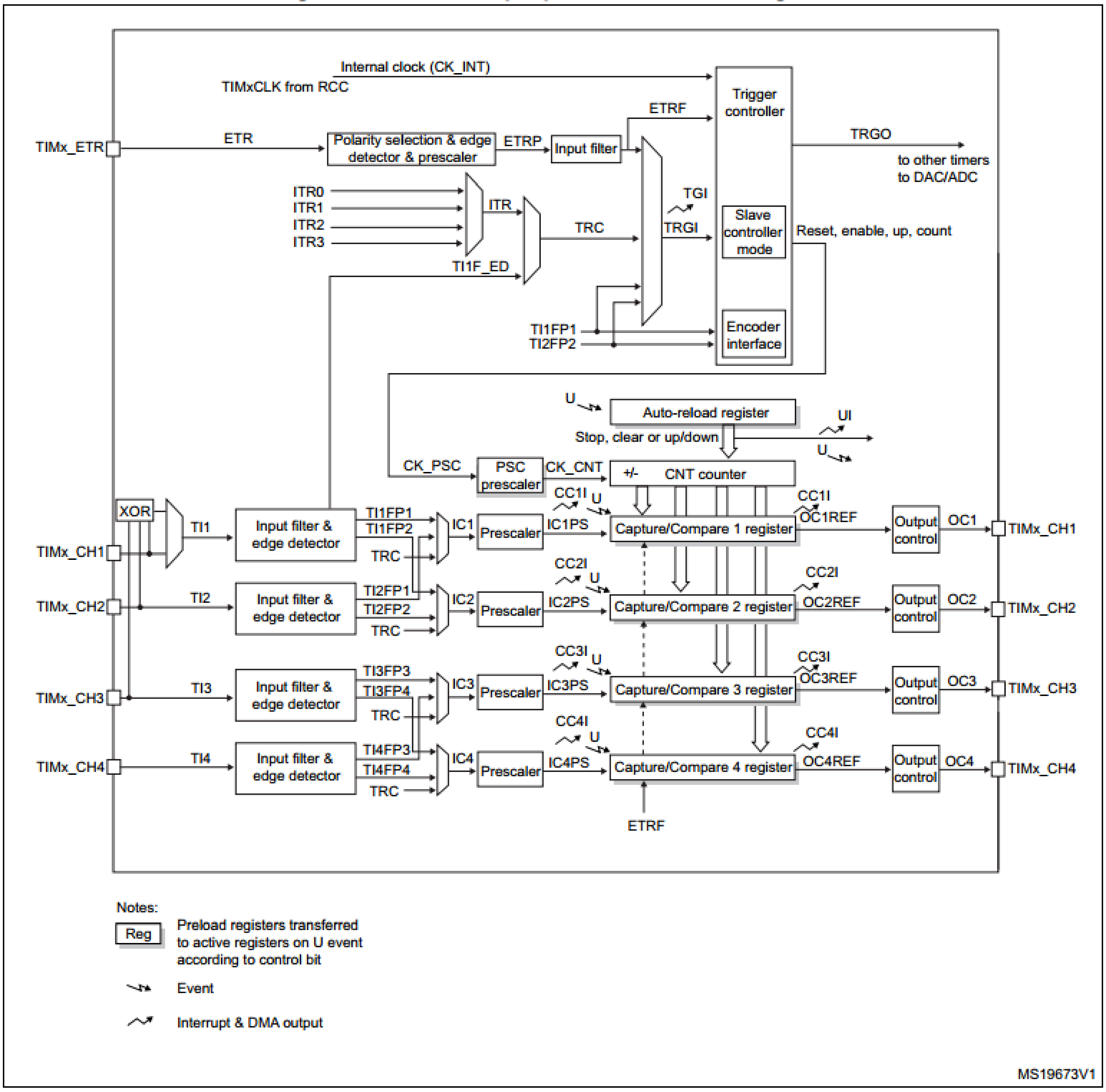


Figure 22-1General-purpose timer Architecture Diagram

22.3. TIM 2/3 functional description

22.3.1. Time-base unit

The main block of the programmable timer is a 16-bit (TIM3) or 32-bit (TIM2) counter with its related autoreload register. The counter can count up but also 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 (TIM3_CNT)
- Prescaler Register (TIM3_PSC)
- Auto-Reload Register (TIM3_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 TIM3_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 TIM3_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 TIM3_CR1 register is set.

Note that the actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

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 TIM3_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.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly:

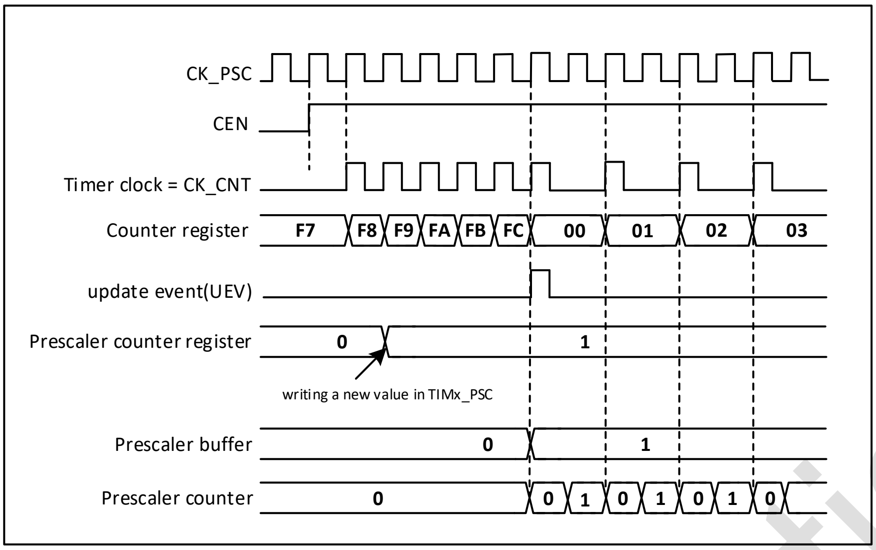


Figure 22-2 Counter timing diagram with prescaler division change from 1 to 2

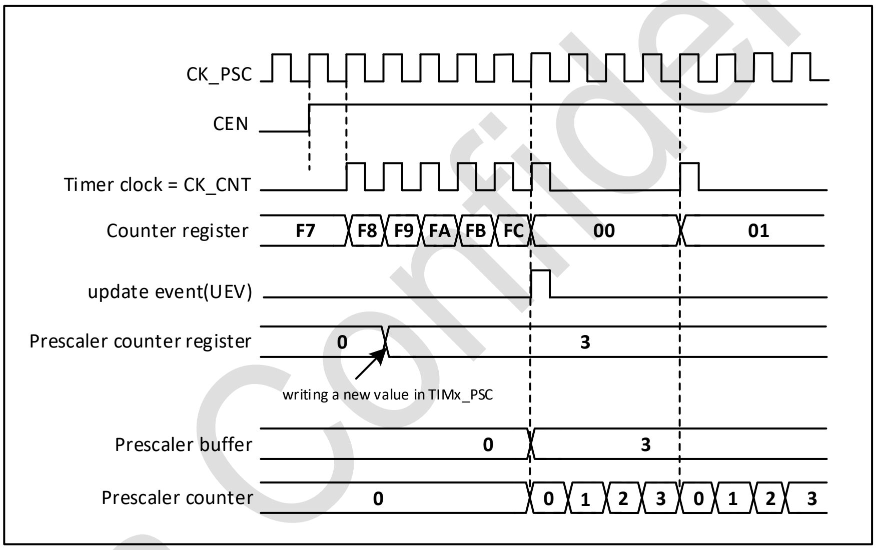


Figure 22-3 Counter timing diagram with prescaler division change from 1 to 4

22.3.2. Counter modes

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.

An Update event can be generated by setting the UG bit in the TIM3_EGR register (by software or by using the slave mode controller).

The UEV event can be disabled by software by setting the UDIS bit in TIM3_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM3_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 TIM3_SR register) is set (depending on the URS bit):

- The auto-reload shadow register is updated with the preload value (TIM3_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).
- The following figures show some examples of the counter behavior for different clock frequencies when TIM3_ARR = 0x36.

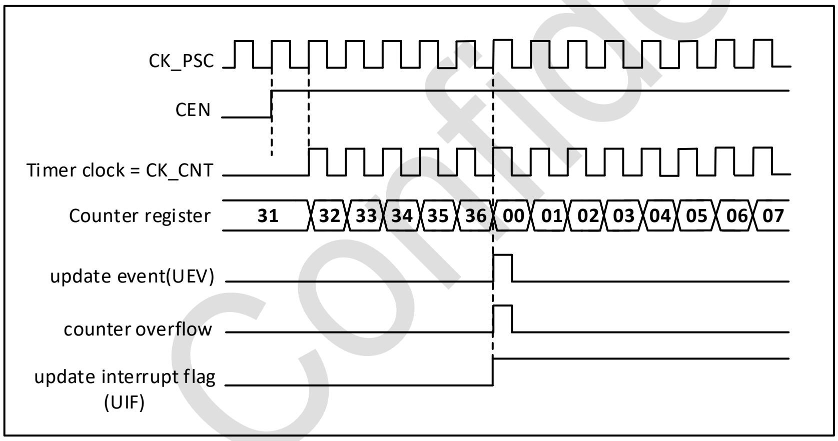


Figure 22-4 Counter timing diagram, internal clock divided by 1

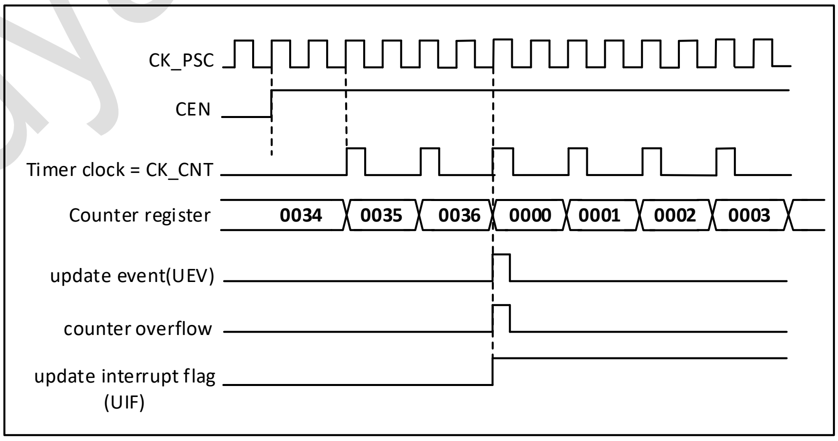


Figure 22-5 Counter timing diagram, internal clock divided by 2

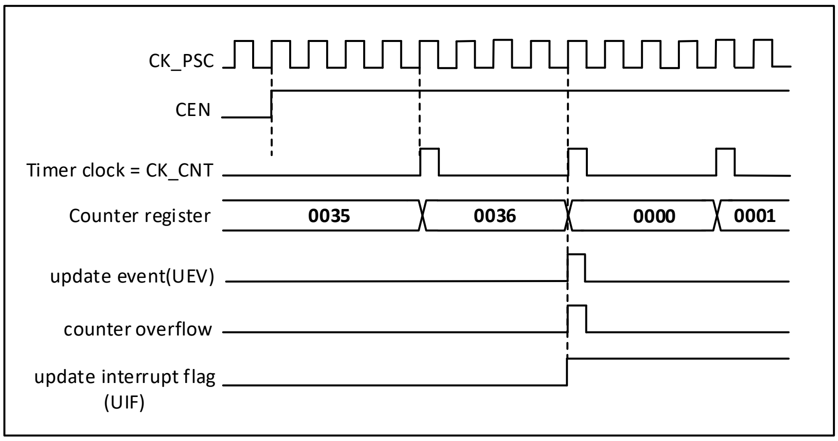


Figure 22-6 Counter timing diagram, internal clock divided by 4

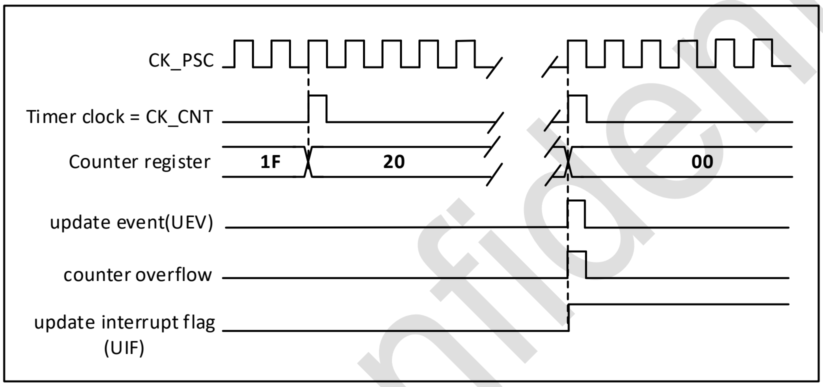


Figure 22-7 Counter timing diagram, internal clock divided by N

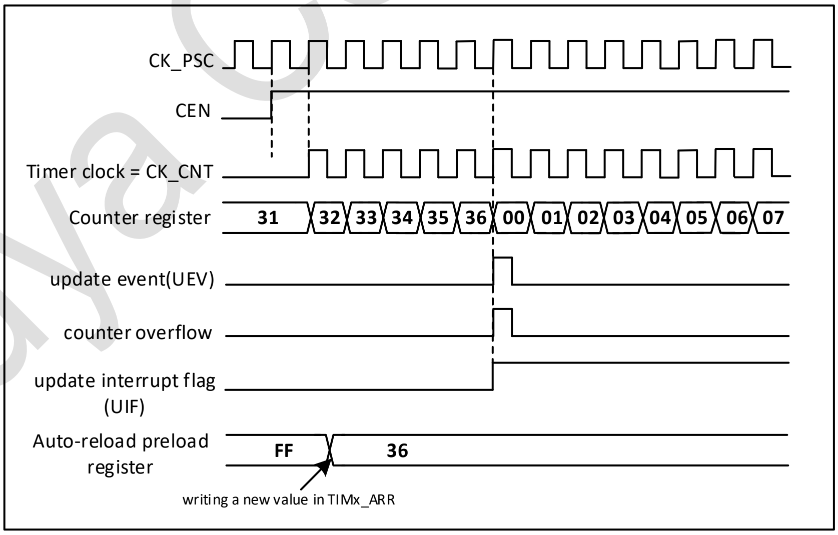


Figure 22-8 Counter timing diagram, Update event when ARPE = 0

(TIM3_ARR not preloaded)

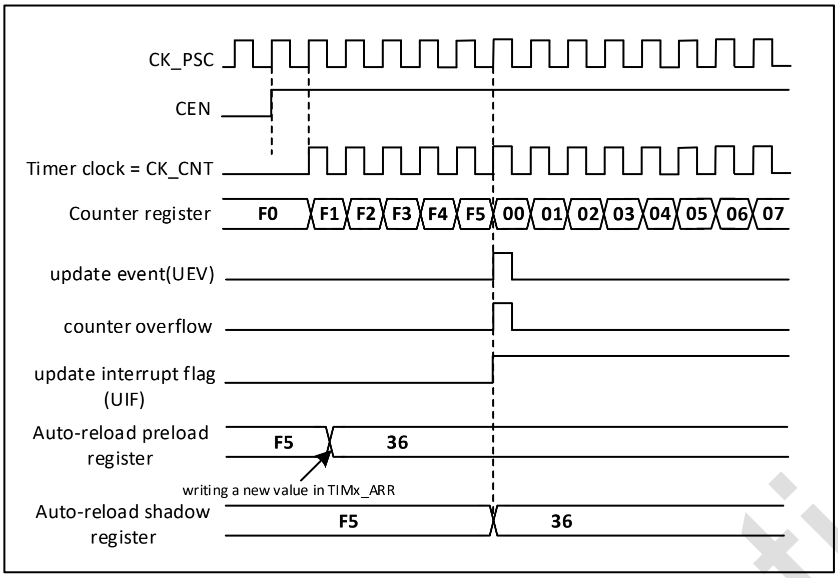


Figure 22-9 Counter timing diagram, Update event when ARPE = 1

(TIM3_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.

An Update event can be generated by setting the UG bit in the TIM3_EGR register (by software or by using the slave mode controller).

The UEV update event can be disabled by software by setting the UDIS bit in TIM3_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 prescale rate doesn’t change).

In addition, if the URS bit (update request selection) in TIM3_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 TIM3_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).

- he auto-reload active register is updated with the preload value (content of the TIM3_ARR register).
- Note that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

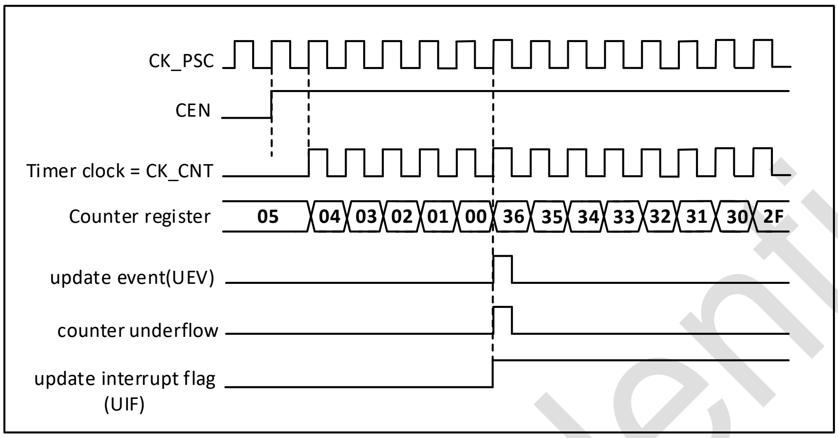


Figure 22-10 Counter timing diagram, internal clock divided by 1

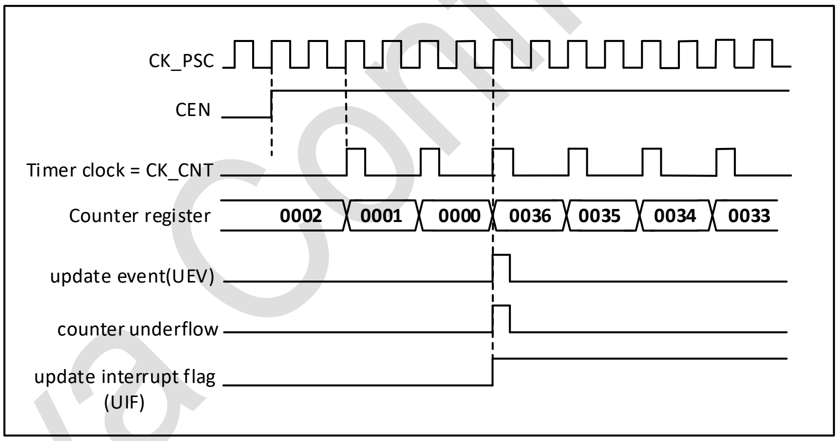


Figure 22-11 Counter timing diagram, internal clock divided by 2

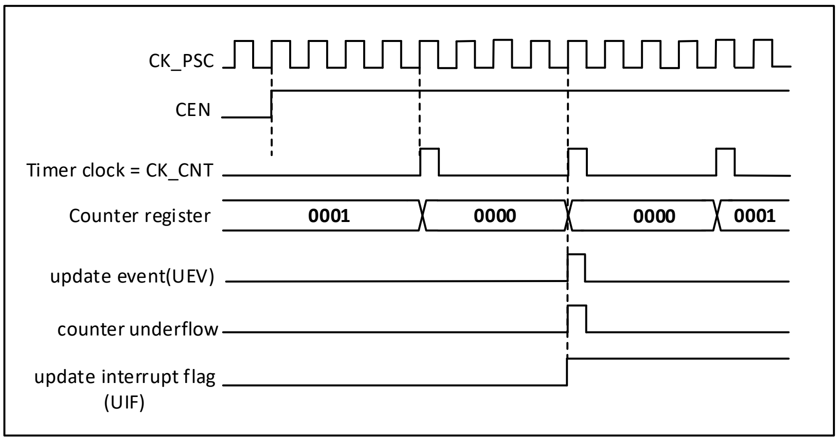


Figure 22-12 Counter timing diagram, internal clock divided by 4

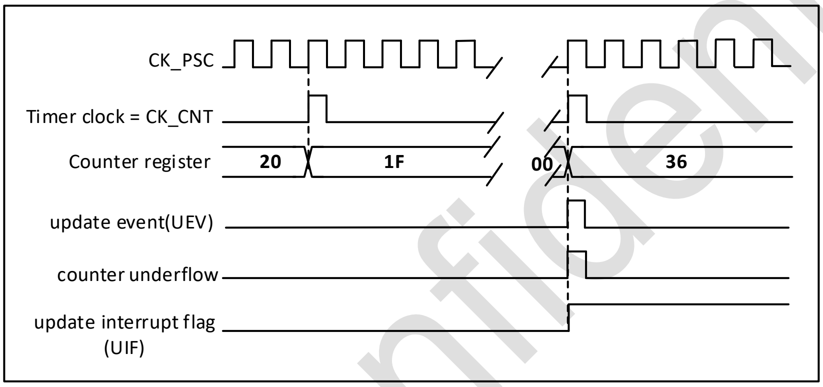


Figure 22-13 Counter timing diagram, internal clock divided by N

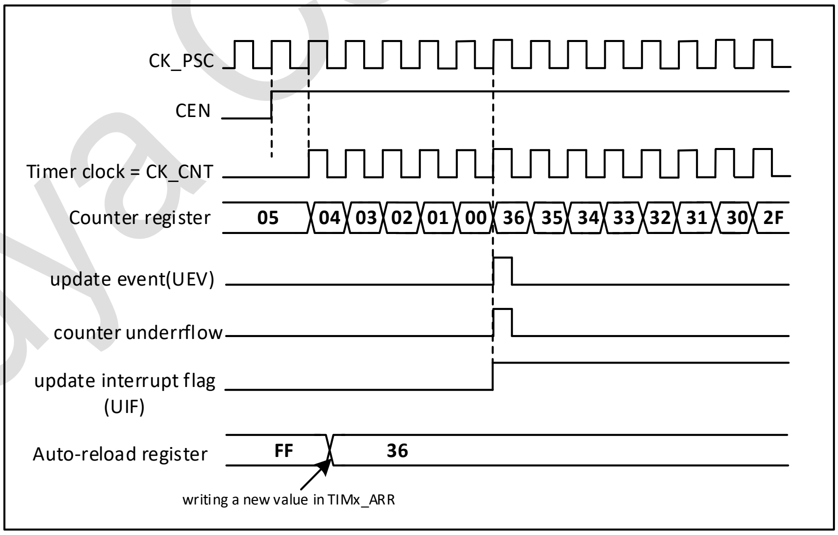


Figure 22-14 Counter timing diagram, Update event when repetition counter is not used

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIM3_ARR register) – 1, generates a counter overflow event, then counts from the autoreload 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 TIM3_CR1 register are not equal to '00'. 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 direction bit (DIR from TIM3_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 TIM3_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by software by setting the UDIS bit in TIM3_CR1 register. 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 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 update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupt 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 TIM3_SR register) is set (depending on the URS bit):

- The buffer of the prescaler is reloaded with the preload value (content of the TIM3_PSC register).
- The auto-reload active register is updated with the preload value (content of the TIM3_ARR register).

Note that if the update source is a counter overflow, the autoreload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

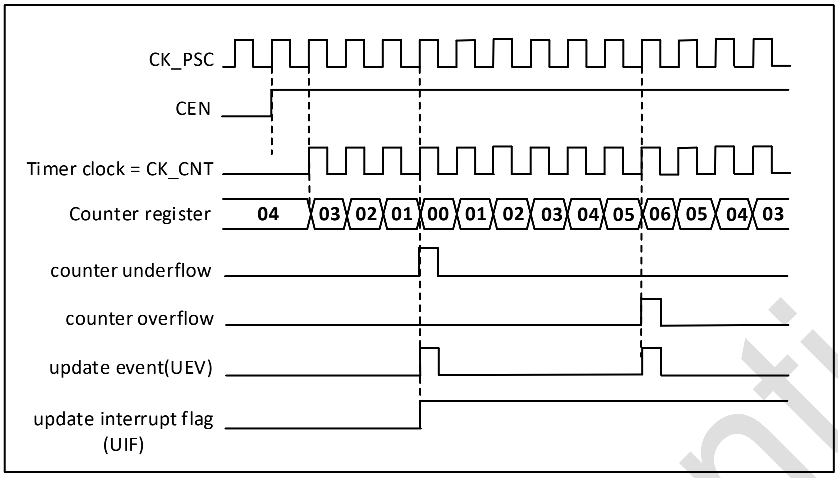


Figure 22-15 Counter timing diagram, internal clock divided by 1, TIM3_ARR = 0x6

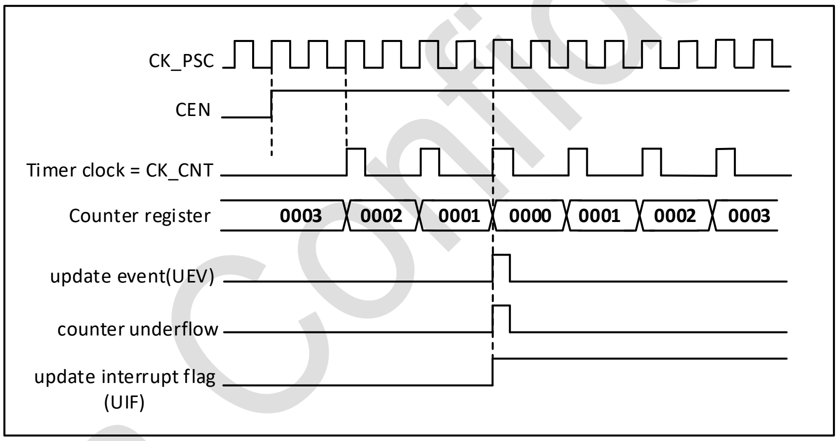


Figure 22-16 Counter timing diagram, internal clock divided by 2

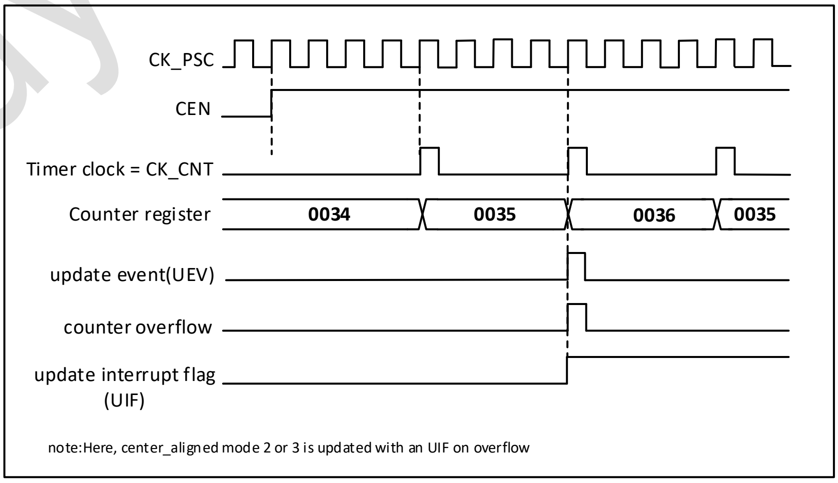


Figure 22-17 Counter timing diagram, internal clock divided by 4, TIM3_ARR = 0x36

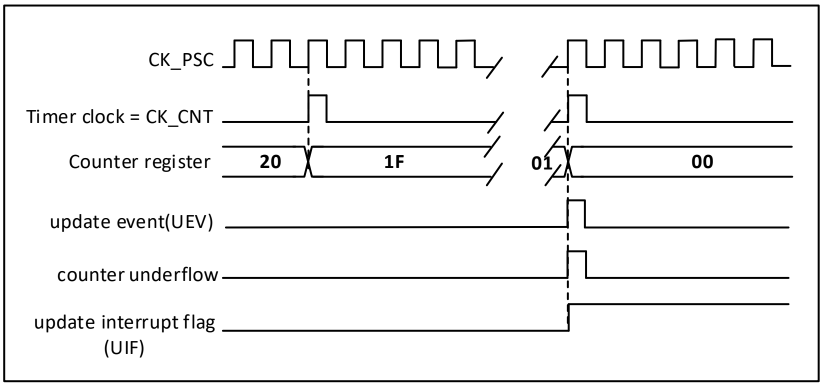


Figure 22-18 Counter timing diagram, internal clock divided by N

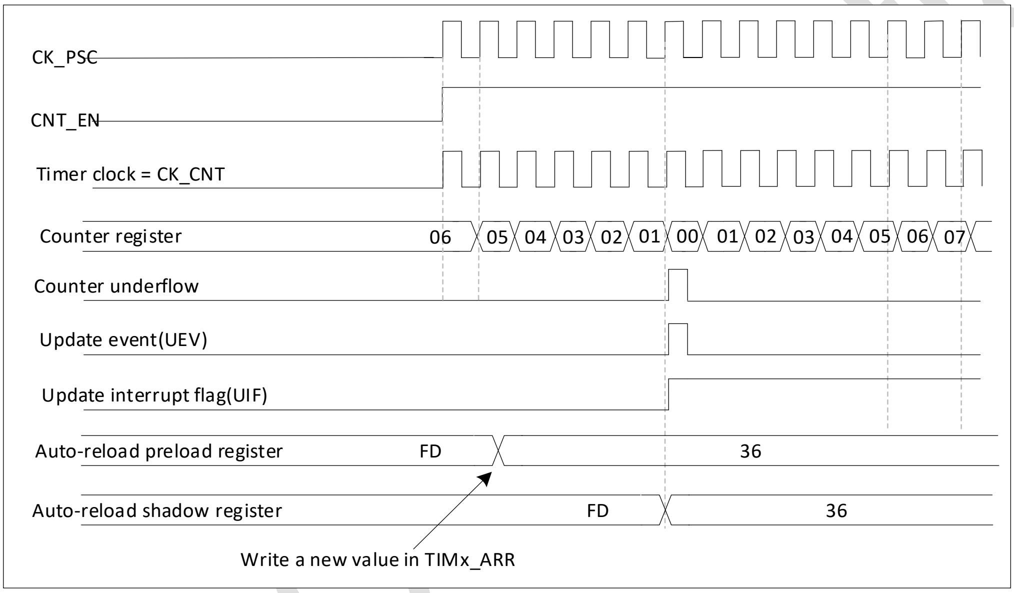


Figure 22-19 Counter timing diagram, Update event with ARPE = 1 (counter underflow)

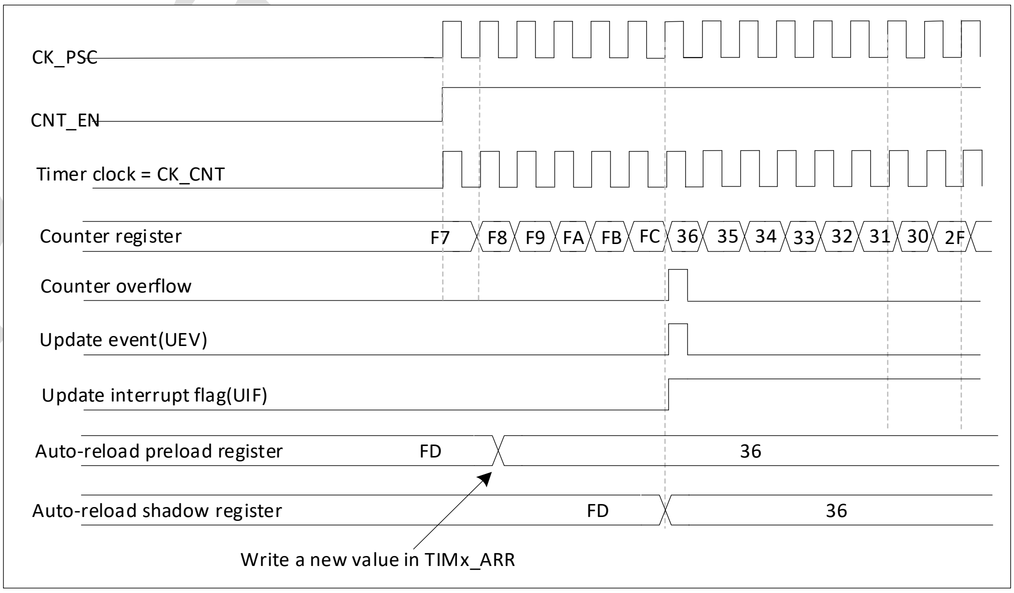


Figure 22-20 Counter timing diagram, Update event with ARPE = 1 (counter overflow)

22.3.3. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin (TIx)
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, Timer 1 can be configured to act as a prescaler for Timer 3.

Internal clock source (CK_INT)

If the slave mode controller is disabled, then the CEN, DIR (in the TIM3_CR1 register) and UG bits (in the TIM3_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.

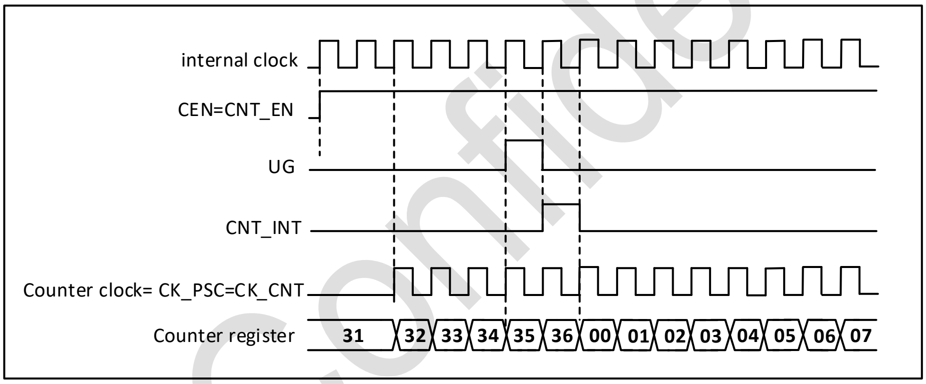


Figure 22-21 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

This mode is selected when SMS = 111 in the TIM3_SMCR register. The counter can count at each rising or falling edge on a selected input.

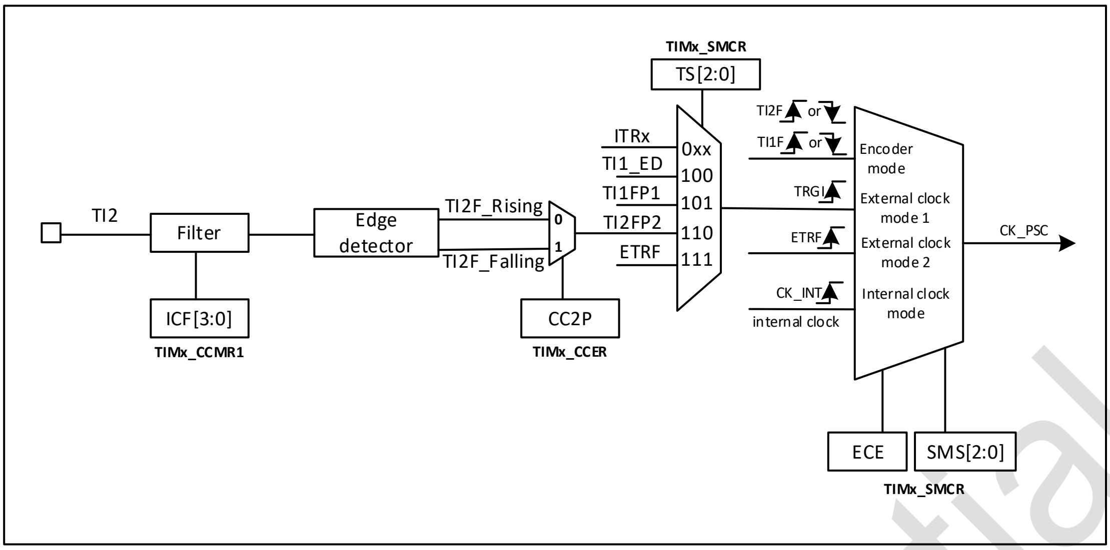


Figure 22-22 TI2 external clock connection example

For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = 01 in the TIM3_CCMR1 register.

2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIM3_CCMR1 register (if no filter is needed, keep IC2F = 0000).

3. Select rising edge polarity by writing CC2P = 0 in the TIM3_CCER register.

4. Configure the timer in external clock mode 1 by writing SMS = 111 in the TIM3_SMCR register.

5. Select TI2 as the input source by writing TS = 110 in the TIM3_SMCR register.

6. Enable the counter by writing CEN = 1 in the TIM3_CR1 register.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

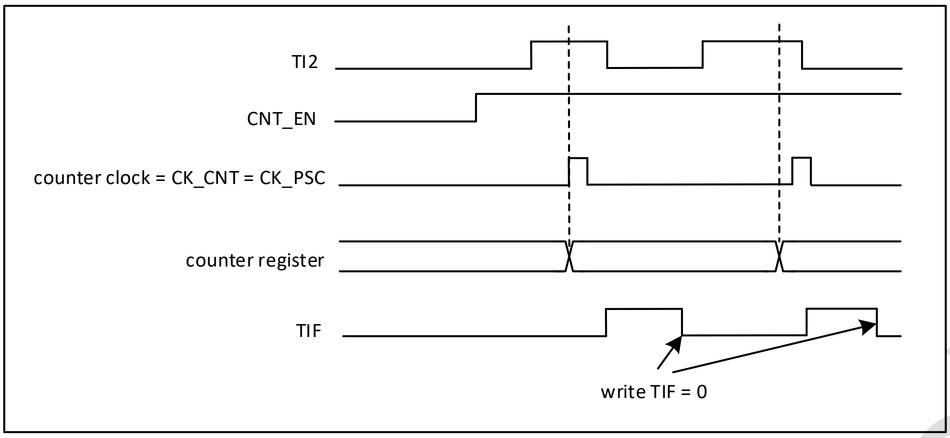


Figure 22-23 Control circuit in external clock mode 1

22.3.4. 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).

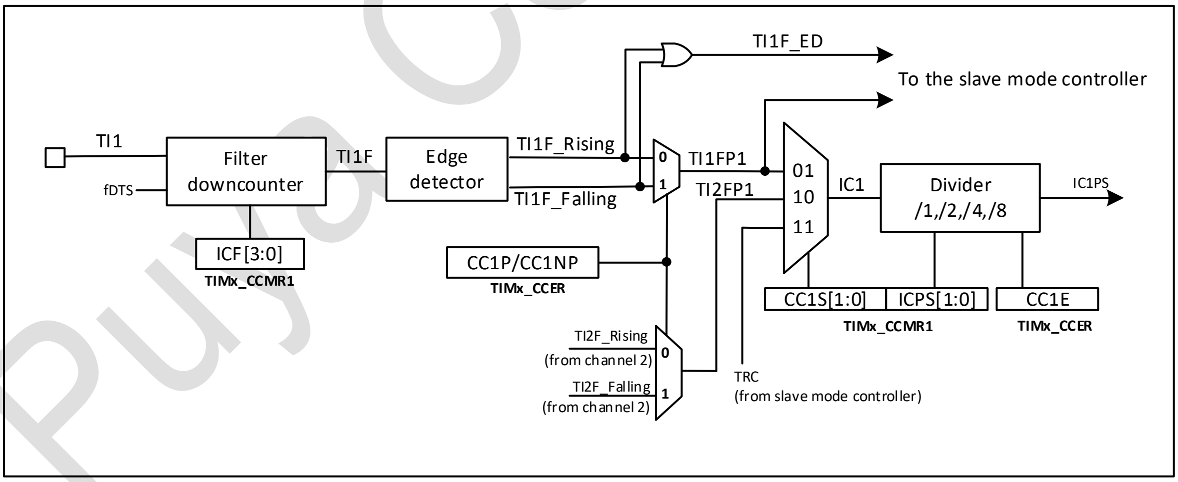


Figure 22-24 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference of OCxRef (active high). The polarity acts at the end of the chain.

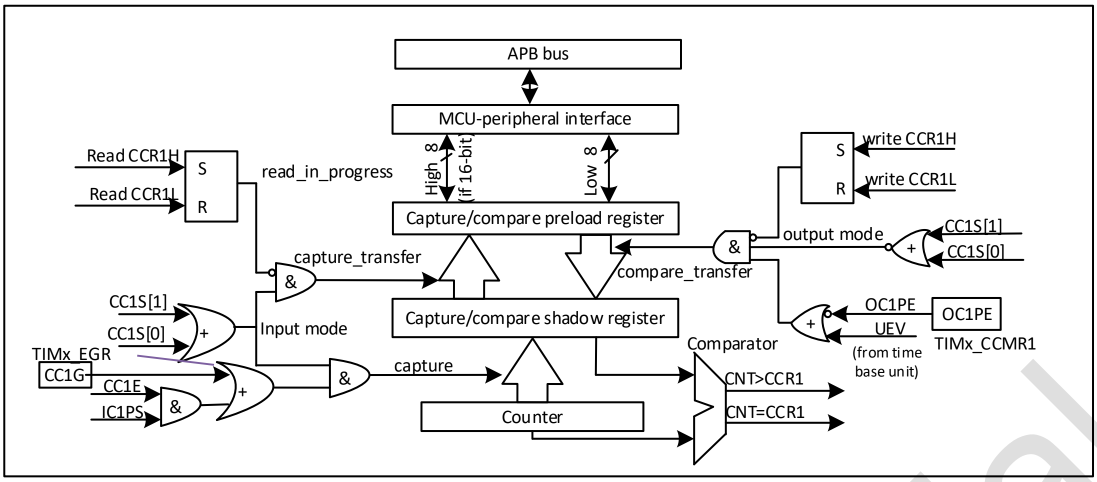


Figure 22-25 Capture/compare channel 1 main circuit

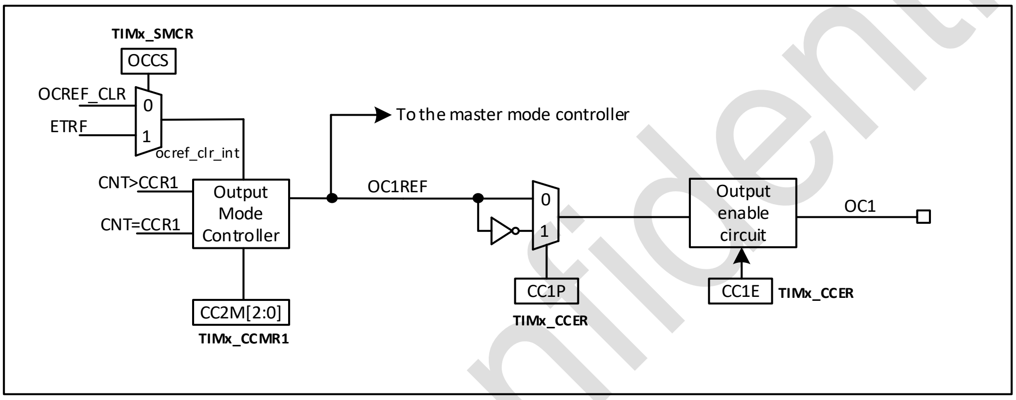


Figure 22-26 Output stage of capture/compare channel (channel 1)

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

22.3.5. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM3_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIM3_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag

CCxOF (TIM3_SR register) is set. CCxIF can be cleared by software by writing it to 0 or by reading the captured data stored in the TIM3_CCRx register. CCxOF is cleared when it is written with 0.

The following example shows how to capture the counter value in TIM3_CCR1 when TI1 input 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 TIM3_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM3_CCR1 register becomes read-only.
- Program the appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIM3_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 TIM3_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing the CC1P and CC1NP bits to 0 in the TIM3_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 TIM3_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIM3_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIM3_DIER register, and/or the DMA request by setting the CC1DE bit in the TIM3_DIER register.

When an input capture occurs:

- The TIM3_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.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIM3_EGR register.

22.3.6. 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.
- The 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, one can measure the period (in TIM3_CCR1 register) and the duty cycle (in TIM3_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 TIM3_CCR1: write the CC1S bits to 01 in the TIM3_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIM3_CCR1 and counter clear): write the CC1P to ‘0’(active on rising edge).
- Select the active input for TIM3_CCR2: write the CC2S bits to 10 in the TIM3_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIM3_CCR2): write the CC2P bit to ‘1’(active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIM3_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIM3_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1 in the TIM3_CCER register.

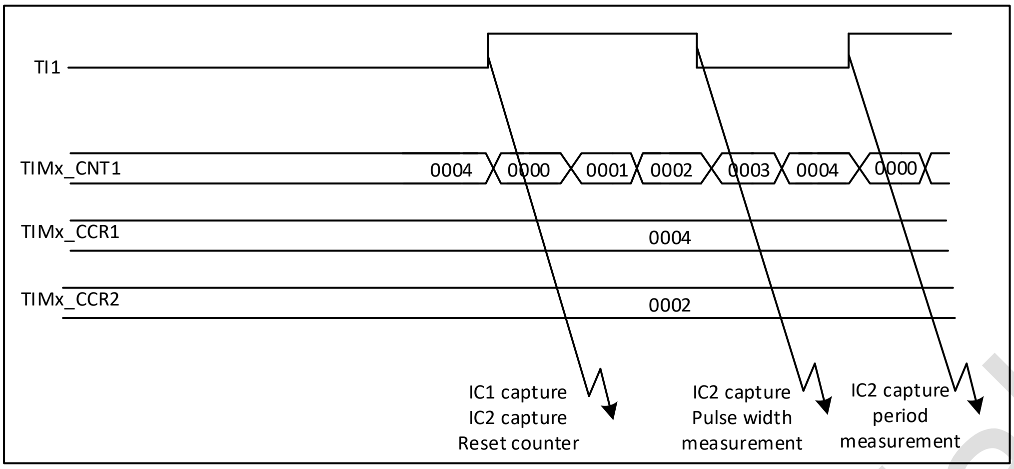


Figure 22-27 PWM input mode timing

22.3.7. Force output mode

In output mode (CCxS bits = 00 in the TIM3_CCMRx register), each output compare signal (OCxREF and then OCx) 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, one just needs to write 101 in the OCxM bits in the corresponding TIM3_CCMRx register. Thus OCxREF is forced high(OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

Like, CCxP = 0 (OCx active high) = > OCx is forced to high level.

OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM3_CCMRx register.

Anyway, the comparison between the TIM3_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.

22.3.8. 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:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM3_CCMRx register) and the output polarity (CCxP bit in the

TIM3_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 TIM3_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM3_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIM3_DIER register, CCDS bit in the TIM3_CR2 register for the DMA request selection).

The TIM3_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM3_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 TIM3_ARR and TIM3_CCRx registers.

3. Set the CCxIE bits if an interrupt request is to be generated.

4. Select the output mode. For example: one must write OCxM = 011, OCxPE = 0, CCxP = 0 and CCxE = 1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.

5. Enable the counter by setting the CEN bit in the TIM3_CR1 register.

The TIM3_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 TIM3_CCRx shadow register is updated only at the next update event UEV). An example is given.

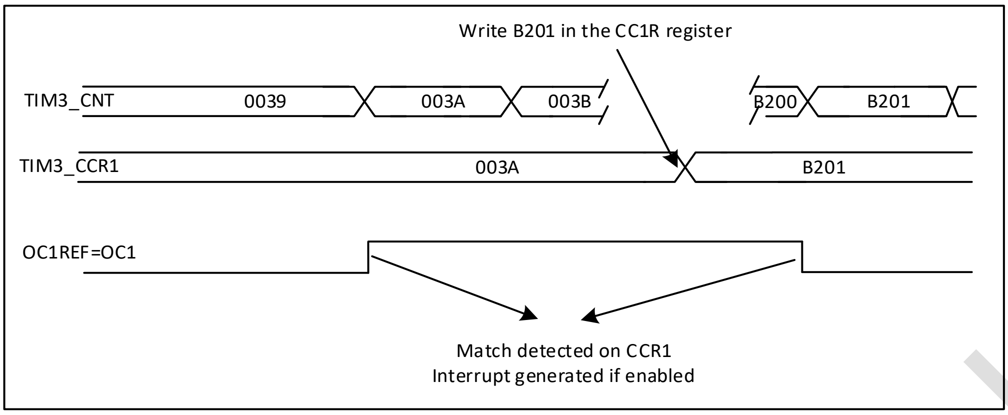


Figure 22-28 Output compare mode, toggle on OC1

22.3.9. Pulse Width Modulation(PWM) mode

Pulse width modulation mode allows to generate a signal with a frequency determined by the value of the TIM3_ARR register and a duty cycle determined by the value of the TIM3_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 TIM3_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM3_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 TIM3_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 the CCxE, CCxNE, MOE, OSSI and OSSR bit in the TIM3_CCER and TIM3_BDTR register. Refer to the TIM3_CCER register description for more details.

In PWM mode (1 or 2), TIM3_CNT and TIM3_CCRx are always compared to determine whether TIM3_CCRx≤TIM3_CNT or TIM3_CNT≤TIM3_CCRx (depending on the direction of the counter).

However, to comply with the OCREF_CLR functionality, the OCREF signal is asserted only:

- When the result of the comparison changes.

- When the output compare mode (OCxM bits in TIM3_CCMRx register) switches from the “frozen” configuration (no comparison, OCxM = ‘000) to one of the PWM modes (OCxM = ‘110 or ‘111).

This forces the PWM by software 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 TIM3_CR1 register.

PWM edge-aligned mode

- **Upcounting configuration**

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to the following example of PWM Mode 1. The reference PWM signal OCxREF is high as long as TIM3_CNT < TIM3_CCRx else it becomes low. If the compare value in TIM3_CCRx is greater than the auto-reload value (in TIM3_ARR) then OCxREF is held at ‘1. If the compare value is 0 then OCxREF is held at ‘0. The following figure shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

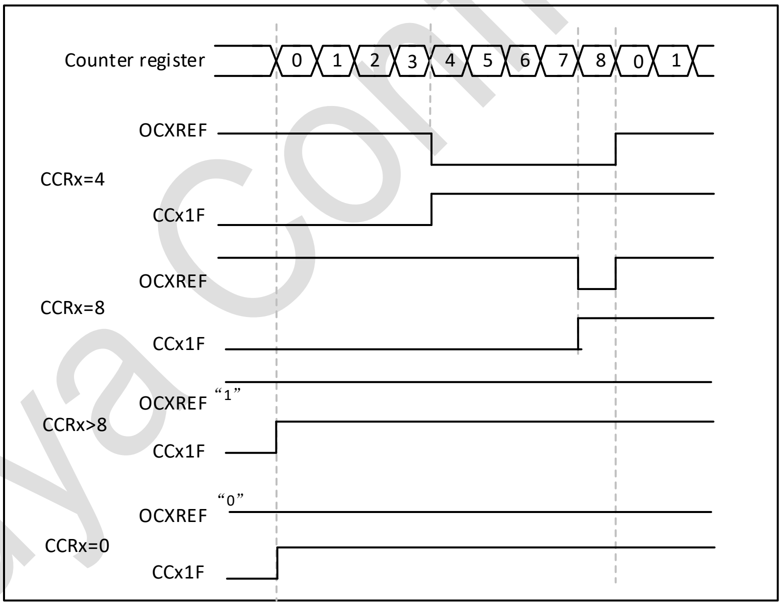


Figure 22-29 Edge-aligned PWM waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low as long as TIM3_CNT > TIM3_CCRx else it becomes high. If the compare value in TIM3_CCRx is greater than the auto-reload value in TIM3_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 TIM3_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 TIM3_CR1 register is updated by hardware and must not be changed by software. Refer to the following example of the Center-aligned mode.

- 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 TIM3_CR1 register.

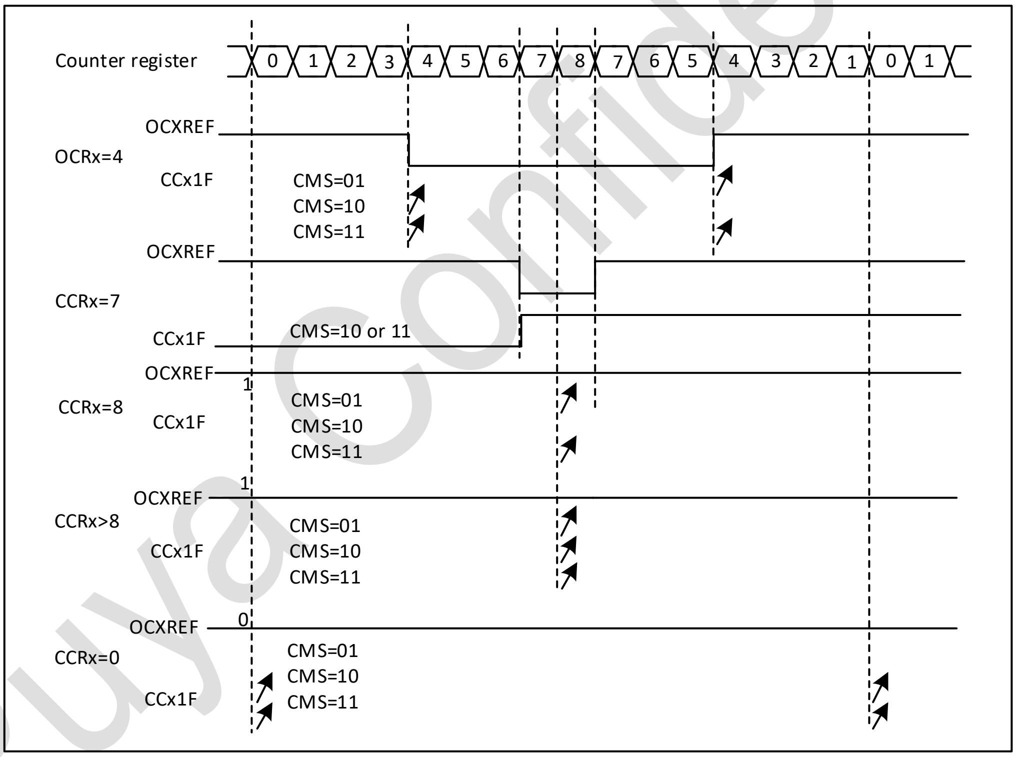


Figure 22-30 Center-aligned PWM waveforms (ARR = 8)

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 TIM3_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 a value greater than the auto-reload value is written in the counter (TIM3_CNT > TIM3_ARR). For example, if the counter was counting up, it continues to count up. ─ The direction is updated if 0 or the TIM3_ARR value is written 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 TIM3_EGR register) just before starting the counter and not to write the counter while it is running.

22.3.10. 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. One-pulse mode is selected by setting the OPM bit in the TIM3_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:

(1) Upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx),

(2) Downcounting: CNT > CCRx.

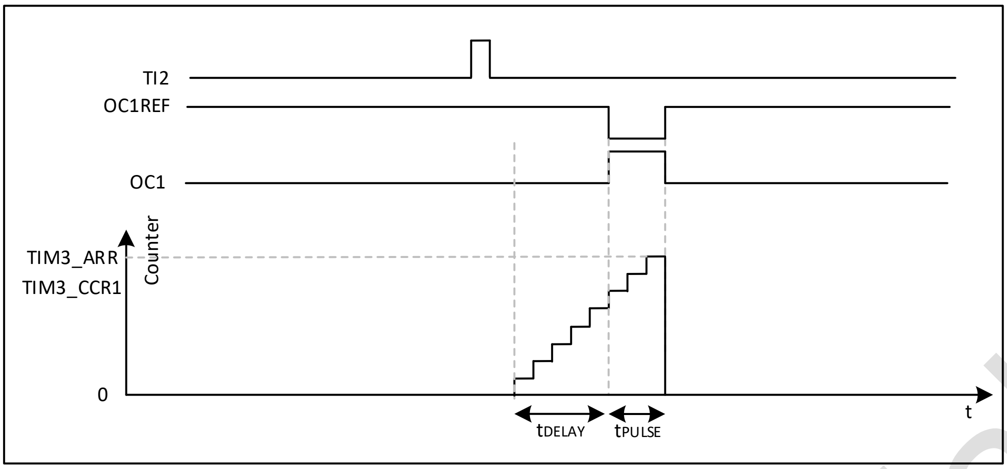


Figure 22-31 Example of one-pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PLUSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Use TI2FP2 as trigger 1:

- Map TI2FP2 on TI2 by writing CC2S = 01 in the TIM3_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P = 0 in the TIM3_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS = 110 in the TIM3_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to ‘110 in the TIM3_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 TIM3_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIM3_ARR - TIM3_CCR1 + 1).
- When the user 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 TIM3_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE = 1 in the TIM3_CCMR1 register and ARPE in the TIM3_CR1 register. In this case one has to write the compare value in the TIM3_CCR1

register, the auto-reload value in the TIM3_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 the example, the DIR and CMS bits in the TIM3_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIM3_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}.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIM3_CCMRx register. Then OCxRef (and OCx) are 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 PWM2 mode.

22.3.11. Encoder interface mode

To select Encoder Interface mode write SMS = ‘001 in the TIM3_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 TIM3_CCER register. When needed, the input filter can be programmed as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table73, The counter is clocked by each valid transition on TI1FP1 or TI2FP2 assuming that it is enabled (CEN bit in TIM3_CR1 register written to ‘1). TI1 and TI2 after input filter and polarity selection, TI1FP1 = TI1 if not filtered and not inverted, TI2FP2 = TI2 if not filtered and not inverted. 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

TIM3_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 TIM3_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIM3_ARR must be configured before starting. In the same way, the capture, compare, prescaler, trigger output features continue to work as normal. In this mode, the counter is modified automatically following the speed and the direction of the incremental 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 22-1 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	Up	Up	Up	Down
	Low	Down	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be 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.

Figure 18-32 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 (TIM3_CCMR1 register, TI1FP1 mapped on TI1)
- CC2S = 01 (TIM3_CCMR2 register, TI2FP2 mapped on TI2)
- CC1P = 0, CC1NP = ‘0’ (TIM3_CCER register, TI1FP1 noninverted, TI1FP1 = TI1)
- CC2P = 0, CC2NP = ‘0’ (TIM3_CCER register, TI2FP2 noninverted, TI2FP2 = TI2)
- SMS = 011 (TIM3_SMCR register, both inputs are active on both rising and falling edges)
- CEN = 1 (TIM3_CR1 register, Counter is enabled)

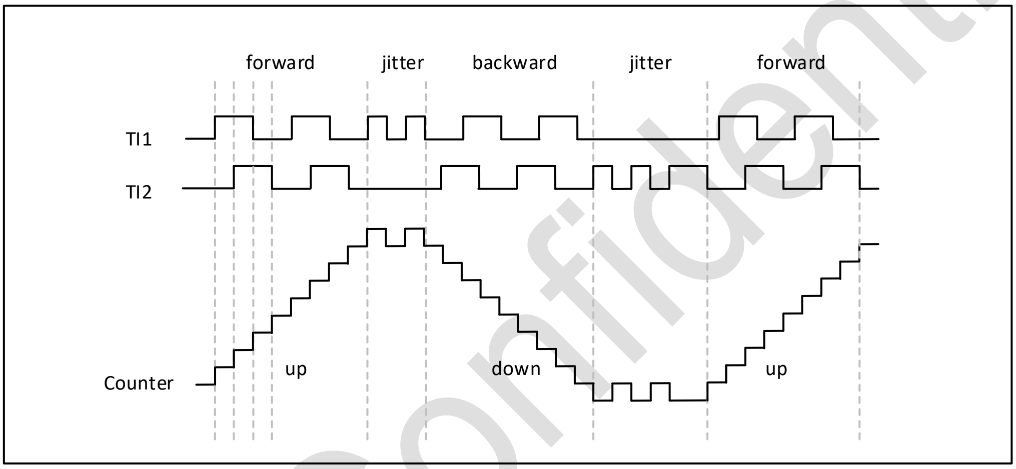


Figure 22-32 Example of counter operation in encoder interface mode

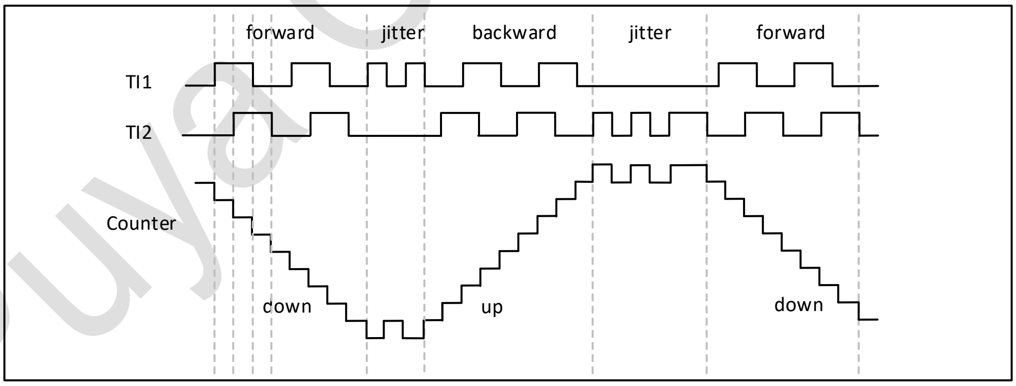


Figure 22-33 Example of encoder interface mode with TI1FP1 polarity inverted

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. This can be done by latching the counter value into a third input capture register if available (then the capture signal must be periodic and can be generated by another timer). when available, it is also possible to read its value through a DMA request generated by a Real-Time clock.

22.3.12. Timer input XOR function

The TI1S bit in the TIM1_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIM3_CH1 to TIM3_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

An example of this feature used to interface Hall sensors.

22.3.13. Timers and external trigger synchronization

The TIM3 Timers 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 UDIS bit from the TIM3_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIM3_ARR, TIM3_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 the TIM3_CCMR1 register. Write CC1P = 0 in TIM3_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS = 100 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.
- Start the counter by writing CEN = 1 in the TIM3_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. In the meantime, the trigger flag is set

(TIF bit in the TIM3_SR register) and an interrupt request, or a DMA request can be sent if enabled (depending on the TIE and TDE bits in TIM3_DIER register).

The following figure shows this behavior when the auto-reload register TIM3_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.

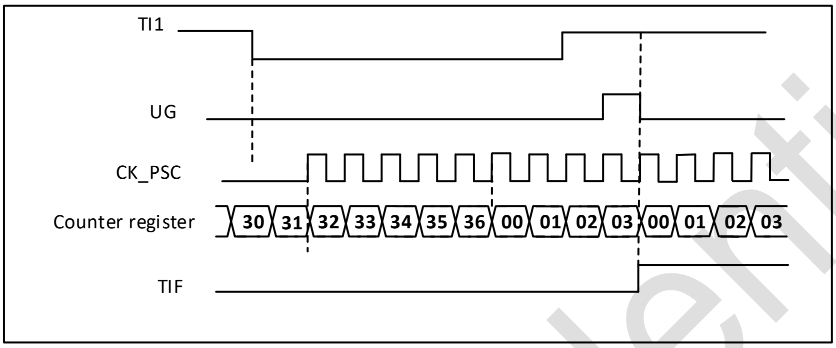


Figure 22-34 Control circuit in reset mode

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 TIM3_CCMR1 register. Write CC1P = 1 in TIM3_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS = 101 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.
- Enable the counter by writing CEN = 1 in the TIM3_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.

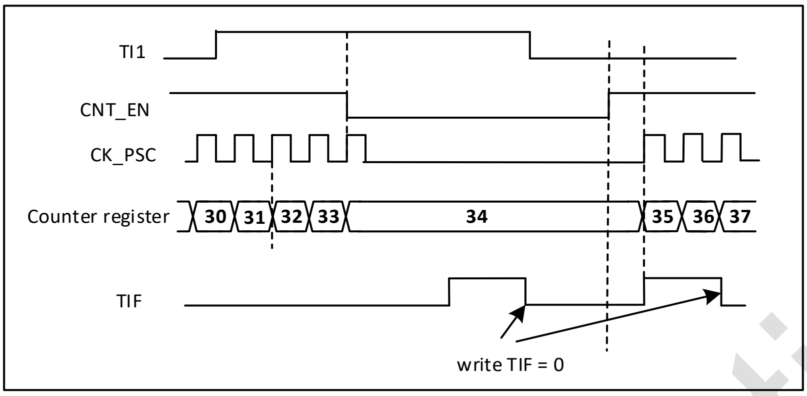


Figure 22-35 Control circuit in gated mode

Slave mode: Trigger mode

The counter can start in response to an event on a selected input.

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. CC2S bits are selecting the input capture source only, CC2S = 01 in TIM3_CCMR1 register. Write CC2P = 1 and CC2NP = 0 in TIM3_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS = 110 in TIM3_SMCR register. Select TI2 as the input source by writing TS = 110 in TIM3_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

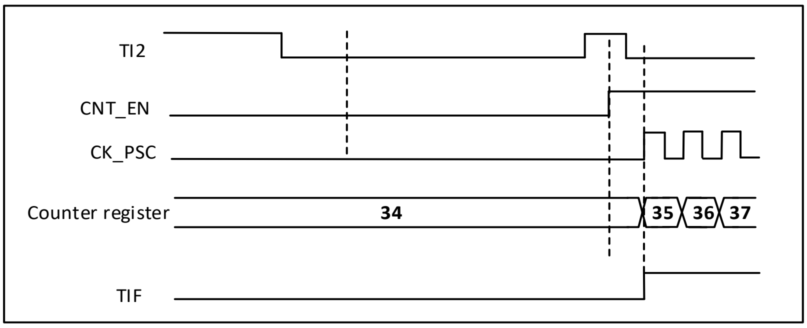


Figure 22-36 Control circuit in trigger mode

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 TIM3_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:

- 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.
- Configure the channel 1 as follows, to detect rising edges on TI:
  - IC1F = 0000: no filter.
  - The capture prescaler is not used for triggering and does not need to be configured.
  - CC1S = 01 in TIM3_CCMR1 register to select only the input capture source
  - CC1P = 0 in TIM3_CCER register to validate the polarity (and detect rising edge only).
- Configure the timer in trigger mode by writing SMS = 110 in TIM3_SMCR register. Select TI1 as the input source by writing TS = 101 in TIM3_SMCR register.

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.

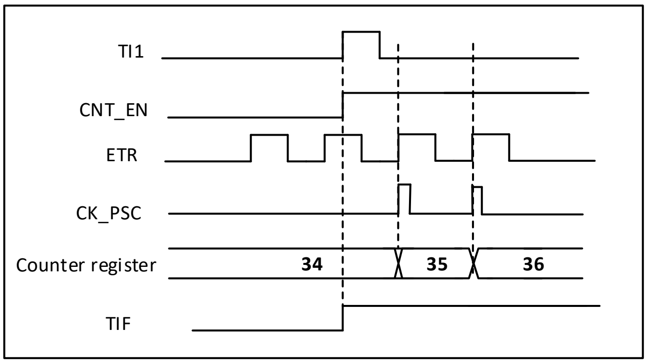


Figure 22-37 Control circuit in external clock mode 2 + trigger mode

22.3.14. Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master Mode, it can reset, start, stop or clock the counter of another Timer configured in Slave Mode.

The following figure presents an overview of the trigger selection and the master mode selection blocks.

Using one timer as prescaler for another

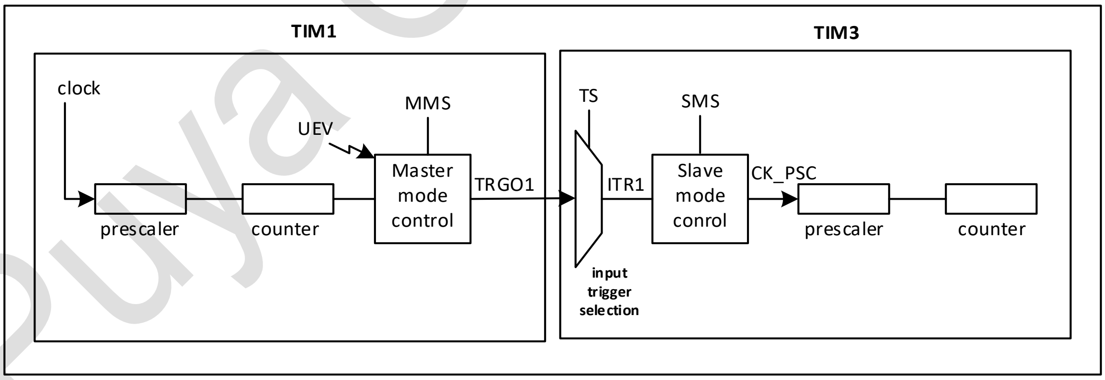


Figure 22-38 Master/Slave timer example

For example, Timer 1 can be configured to act as a prescaler for Timer 3. To do this:

- Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV. If MMS = 010 is written in the TIM1_CR2 register, a rising edge is output on TRGO1 each time an update event is generated.

- To connect the TRGO1 output of Timer 1 to Timer 3, Timer 3 must be configured in slave mode using ITR1 as internal trigger. This is selected through the TS bits in the TIM3_SMCR register (writing TS = 000).
- Then the Timer3's slave mode controller should be configured in external clock mode 1 (write SMS = 111 in the TIM3_SMCR register). This causes Timer 3 to be clocked by the rising edge of the periodic Timer 1 trigger signal (which correspond to the timer 1 counter overflow).
- Finally both timers must be enabled by setting their respective CEN bits within their respective TIMx_CR1 registers. Make sure to enable Timer3 before enabling Timer1.

Note: If OCx is selected on Timer 1 as trigger output (MMS = 1xx), its rising edge is used to clock the counter of timer 2.

Using one timer to enable another timer

In this example, we control the enable of Timer 3 with the output compare 1 of Timer 1. Refer to above Figure for connections. Timer 3 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure Timer 1 master mode to enable the slave timer(MMS = 001 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register).
- Enable Timer 3 by writing ‘1 in the CEN bit (TIM3_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).

Note: The counter 3 clock is not synchronized with counter 1, this mode only affects the Timer 3 counter enable signal.

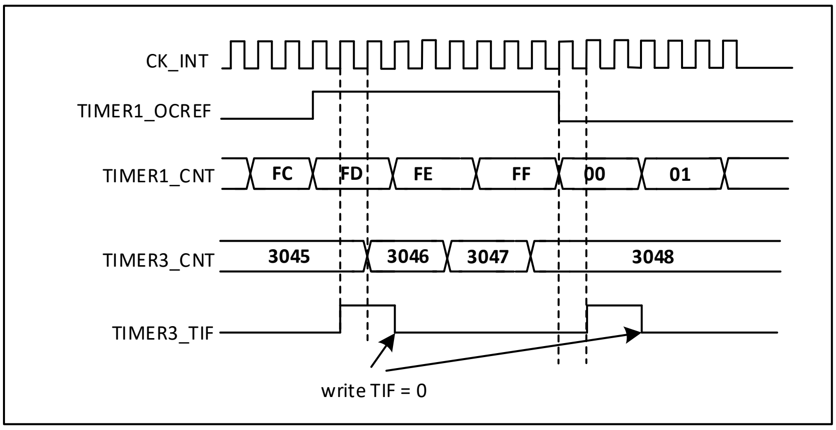


Figure 22-39 Gating timer 3 with OC1REF of timer 1

In the example in figure 22-39, the Timer 3 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 1. Then any value can be written in the timer counters. The timers can easily be reset by software using the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 3. Timer 1 is the master and starts from 0. Timer 3 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 3 stops when Timer 1 is disabled by writing ‘0 to the CEN bit in the TIM1_CR1 register:

- Configure Timer 1 master mode to send its Counter Enable signal (CNT_EN) as a trigger output (MMS = 001 in the TIM1_CR2 register).
- Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register).
- Configure Timer 2 to get the input trigger from Timer 1 (TS = 000 in the TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS = 101 in TIM2_SMCR register).
- Reset Timer 1 by writing ‘1 in UG bit (TIM1_EGR register).
- Reset Timer 2 by writing ‘1 in UG bit (TIM2_EGR register).
- Initialize Timer 3 to 0xE7 by writing ‘0xE7’ in the timer 2 counter (TIM2_CNT).
- Enable Timer 2 by writing ‘1 in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing ‘0 in the CEN bit (TIM1_CR1 register).

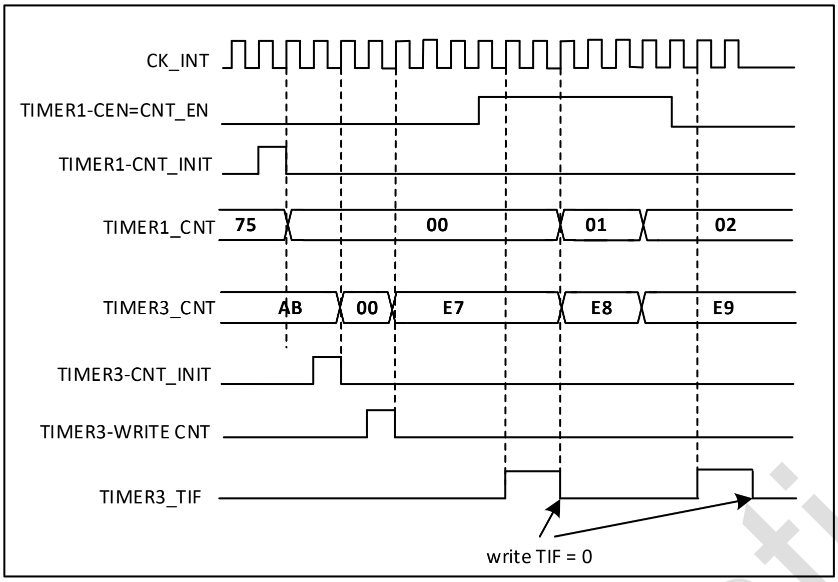


Figure 22-40 Gating timer 2 with Enable of timer 1

Using one timer to start another timer

In this example, we set the enable of Timer 3 with the update event of Timer 1. Refer to Figure 132 for connections. Timer 3 starts counting from its current value (which can be nonzero) on the divided internal clock as soon as the update event is generated by Timer 1. When Timer 3 receives the trigger signal its CEN bit is automatically set and the counter counts until we write '0' to the CEN bit in the TIM3_CR1 register. Both counter clock frequencies are divided by 3 by the pre-scaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT}/3$).

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS = 010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register).
- Start Timer 1 by writing '1 in the CEN bit (TIM1_CR1 register).

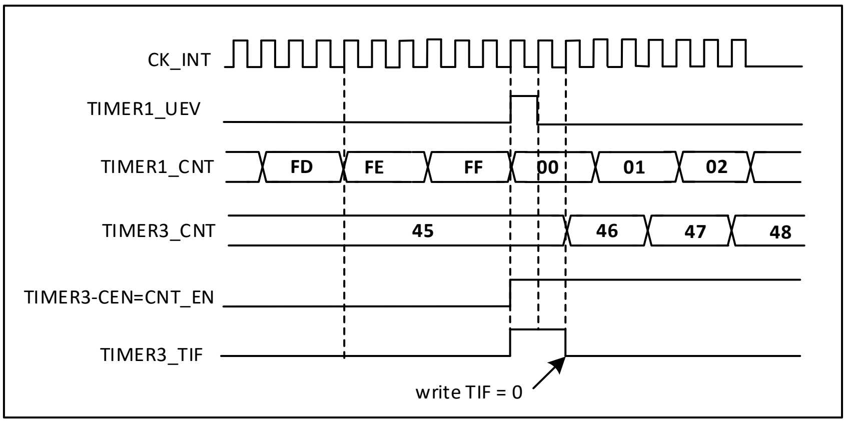


Figure 22-41 Triggering timer 3 with update of timer 1

As in the previous example, both counters can be initialized before starting counting. The following Figure shows the behavior with the same configuration as in the previous Figure but in trigger mode instead of gated mode (SMS = 110 in the TIM3_SMCR register).

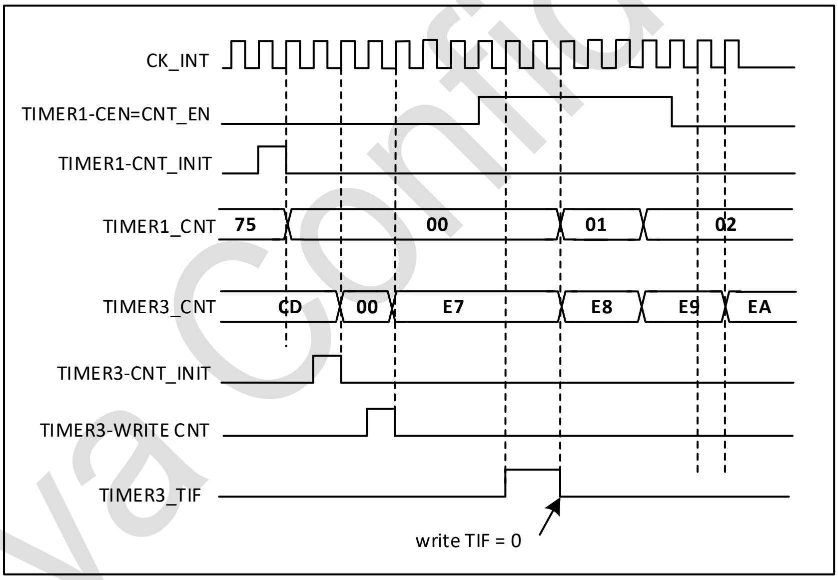


Figure 22-42 Triggering timer 3 with Enable of timer 1

Use an external trigger to start 2 timers synchronously

In this example, we set the enable of timer 1 when its TI1 input rises, and the enable of Timer 3 with the enable of Timer 1. Refer to Figure 132 for connections. To ensure the counters are aligned, Timer 1 must be configured in Master/Slave mode (slave with respect to TI1, master with respect to Timer 3):

- Configure Timer 1 master mode to send its Enable as trigger output (MMS = 001 in the TIM1_CR2 register).
- Configure Timer 1 slave mode to get the input trigger from TI1 (TS = 100 in the TIM1_SMCR register).
- Configure Timer 1 in trigger mode (SMS = 110 in the TIM1_SMCR register).
- Configure the Timer 1 in Master/Slave mode by writing MSM = 1 (TIM1_SMCR register).
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register).
- Configure Timer 3 in trigger mode (SMS = 110 in the TIM3_SMCR register).

When a rising edge occurs on TI1 (Timer 1), both counters starts counting synchronously on the internal clock and both TIF flags are set.

Note: In this example both timers are initialized before starting (by setting their respective UG bits). Both counters starts from 0, but an offset can easily be inserted between them by writing any of the counter registers (TIMx_CNT). One can see that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer1.

22.3.15. Debug mode

When the microcontroller enters debug mode, the TIM3 counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBGMCU module.

22.4. Register Descriptions

TIM2 register base address: 0x4000 0000

TIM3 register base address: 0x4000 0400

22.4.1. TIM2/3 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	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			
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx) 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved
7	ARPE	RW	0	Auto-reload preload enable 0: TIM3_ARR register is not buffered 1: TIM3_ARR register is buffered
6:5	CMS[1:0]	RW	00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit(DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS = 00 in TIM3_CCMRx register) are set both when the counter is counting up or 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

Bit	Name	R/W	Reset Value	Function
				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: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: − Counter overflow/underflow − Setting the UG bit − Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
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.

22.4.2. TIM2/3 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	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	-	0	Reserved, must be kept at reset value.
7	TI1S	RW	0	TI1 selection 0: The TIM3_CH1 pin is connected to TI1 input 1: The TIM3_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6:4	MMS[2:0]	RW	000	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 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 OR 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 TIM3_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)

Bit	Name	R/W	Reset Value	Function
				110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2:0	Reseved	-	0	Reserved, must be kept at reset value.

22.4.3. TIM2/3 slave mode control register (TIMx_SMCR)

Address offset:0x08

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	-	0	Reserved, must be kept at reset value.
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or the inversion of ETR is to be used as the trigger operation 0: ETR is not inverted and a high level or rising edge is active; 1: ETR is inverted, a low level or falling edge is active.
14	ECE	RW	0	External clock enable This bit enables external clock mode 2 0: disables external clock mode 2; 1: enables external clock mode 2; 1: Enables 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 TRGI to ETRF (SMS=111 and TS=111). Note 2: The following slave modes can be used in conjunction with external clock mode 2: reset mode, gated mode and trigger mode; however, the

Bit	Name	R/W	Reset Value	Function
				TRGI cannot be connected to the ETRF in this case (the TS bit cannot be '111'). Note 3: When external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13:12	ETPS[1:0]	RW	0	External trigger prescaler The frequency of the external trigger signal ETRP must be at most 1/4 of the TIMxCLK frequency. Prescaling can be used to reduce the frequency of ETRP when a faster external clock is input. 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[3:0]	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 effect, the digital filter is an event counter which records to N events and then produces a jump in the output. 0000: no filter, sampled at 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：采样频率 fSAMPLING=fDTS/8， N=6 1001：采样频率 fSAMPLING=fDTS/8, N=8 1010：采样频率 fSAMPLING=fDTS/16, N=5 1011：采样频率 fSAMPLING=fDTS/16， N=6 1100：采样频率 fSAMPLING=fDTS/16， N=8 1101：采样频率 fSAMPLING=fDTS/32， N=5 1110：采样频率 fSAMPLING=fDTS/32， N=6 1111：采样频率 fSAMPLING=fDTS/32， N=8
7	MSM	RW	0	Master/Slave mode 0: No action

Bit	Name	R/W	Reset Value	Function
				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	RW	0	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0). 001: 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 For more details on ITRx, refer to Table 5-1 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 the OCREF_CLR input 1: OCREF_CLR_INT is connected to ETRF
2:0	SMS	RW	0	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to thepolarity 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 TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - 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.

Bit	Name	R/W	Reset Value	Function
				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_ED 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 22-2 TIM3 internal trigger connection

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM2	TIM1	TIM15	TIM3	TIM14_OC
TIM3	TIM1	TIM2	TIM15	TIM14_OC

22.4.4. TIM2/3 DMA/Interrupt enable register (TIMx_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
14	TDE	RW	0	TDE: Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled.
13	Reserved	-	0	Reserved, must be kept at reset value.
12	CC4DE	RW	0	CC4DE: Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled. 1: CC4 DMA request enabled.

Bit	Name	R/W	Reset Value	Function
11	CC3DE	RW	0	CC3DE: Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled. 1: CC3 DMA request enabled.
10	CC2DE	RW	0	CC2DE: Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
9	CC1DE	RW	0	CC1DE: Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled.
7	Reserved	-	0	Reserved, must be kept at reset value.
6	TIE	RW	0	TIE: Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled.
5	Reserved	-	0	Reserved, must be kept at reset value.
4	CC4IE	RW	0	CC4IE: Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled. 1: CC4 interrupt enabled.
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

22.4.5. TIM2/3 status register (TIMx_SR)

22.4.6. 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
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
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
31:2	Reserved	-	0	Reserved, must be kept at reset value.
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 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. It is cleared by software or by writing TIMx_CCR1 by writing it to 0. 0: no repeat captures has been detected; 1: Falling edge capture event occurs
19	IC4IR	RC_W0	0	Rising edge capture 4 flag Refer to IC1IF description
18	IC3IR	RC_W0	0	Rising edge capture 3 flag Refer to IC1IF description
17	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IF description
16	IC1IR	RC_W0	0	Rising edge capture 4 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. It is cleared by software or by writing TIMx_CCR1 by writing it to 0. 0: no repeat captures has been detected; 1: Rising edge capture event occurs

Bit	Name	R/W	Reset Value	Function
15:13	Reserved	-	0	Reserved, must be kept at reset value.
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	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 TIM3_CCR1 register while CC1IF flag was already set
8:07	Reserved	-	-	Reserved, must be kept at reset value.
6	TIF	RC_W0	0	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode. It is set when the counter starts or stops when gated mode is selected). It is cleared by software. 0: No trigger event occurred 1: Trigger interrupt pending
5	Reserved	-	-	Reserved, must be kept at reset value
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 flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match 1: The content of the counter TIM3_CNT matches the content of the

Bit	Name	R/W	Reset Value	Function
				TIM3_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 TIM3_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM3_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 and if UDIS = 0 in the TIM3_CR1 register. − When CNT is reinitialized by software using the UG bit in TIM3_EGR register, if URS = 0 andUDIS = 0 in the TIM3_CR1 register. − When CNT is reinitialized by a trigger event (refer to the slave mode control register(TIM3_SMCR) description), if URS = 0 and UDIS = 0 in the TIM3_CR1 register.

22.4.7. TIM2/3 event generation register (TIMx_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
31:16	Reserved	-	0	Reserved, must be kept at reset value.
15:7	Reserved	-	0	Reserved, must be kept at reset value.
6	TG	W	0	Trigger generation

Bit	Name	R/W	Reset Value	Function
				This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	Reserved	-	0	Reserved, must be kept at reset value.
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 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates 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 selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIM3_ARR) if DIR = 1 (downcounting).

22.4.8. TIM2/3 capture/compare mode register 1 (TIMx_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**
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, must be kept at reset value.
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 (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 is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM3_CCER).
7	OC1CE	RW	0	Output Compare 1 Clear Enable 0: OC1REF is not affected by the ETRF input 1: OC1REF is cleared as soon as a High level is detected on ETRF input
6:4	OC1M[2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference sig-nal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits.

Bit	Name	R/W	Reset Value	Function
				000: Frozen - The comparison between the output compare register TIM3_CCR1 and the counter TIM3_CNT has no effect on the outputs. 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 TIM3_CNT = TIM3_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 TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = ‘0) as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = 1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: 1: 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). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.
3	OC1PE	RW	0	Output compare 1 preload enable 0: Preload register on TIM3_CCR1 disabled. TIM3_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM3_CCR1 enabled. Read/Write operations access the preload register. TIM3_CCR1 preload value is loaded in the active register at each update event. Note: 1: 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).

Bit	Name	R/W	Reset Value	Function
				2: Only in one pulse mode, PWM mode can be used without confirming the preload register, otherwise its behavior is undefined.
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 CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. 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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM3_CCER).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
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 (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.

Bit	Name	R/W	Reset Value	Function
				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 is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = 0 in TIM3_CCER).
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[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	Capture/Compare 1 selection

Bit	Name	R/W	Reset Value	Function
				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 is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIMx_CCER).

22.4.9. TIM2/3 capture/compare mode register 2 (TIMx_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	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**
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, must be kept at reset value.
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 (input/output) as well as the used input.

Bit	Name	R/W	Reset Value	Function
				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 TS bit (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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIMx_CCER).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		Reserved, must be kept at reset value.
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 (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

Bit	Name	R/W	Reset Value	Function
				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 TS bit (TIM3_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 3 filter
3:2	IC3PSC[1:0]	RW	00	Input capture 3 prescaler
1:0	CC3S[1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/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 is working only if an internal trigger input is selected through TS bit (TIM3_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = 0 in TIM3_CCER).

22.4.10. TIM2/3 capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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	-	0	Reserved, must be kept at reset value.
15	CC4NP	RW	0	Capture/Compare 4 output Polarity. Refer to CC1NP description
14	Reserved	-	0	Reserved, must be kept at reset value.
13	CC4P	RW	0	Capture/Compare 4 output Polarity.

Bit	Name	R/W	Reset Value	Function
				Refer to CC1P description
12	Reserved	-	0	Capture/Compare 4 output enable. Refer to CC1E description
11	CC3NP	RW	0	Capture/Compare 3 output Polarity. Refer to CC1NP description
10	Reserved	-	0	Reserved, must be kept at reset value.
9	CC3P	RW	0	Capture/Compare 3 output Polarity. Refer to CC1P description
8	CC3E	RW	0	Capture/Compare 3 output enable. Refer to CC1E description
7	CC2NP	RW	0	Capture/Compare 2 output Polarity. Refer to CC1NP description
6	Reserved	-	0	Reserved, must be kept at reset value.
5	CC2P	RW	0	Capture/Compare 2 output Polarity. Refer to CC1P description
4	CC2E	RW	0	Capture/Compare 2 output enable. Refer to CC1E description
3	CC1NP	RW	0	Capture/Compare 1 output Polarity 0: OC1N active high 1: OC1N active low This bit is used in conjunction with CC1P to define TI1FP1/TI2FP1 polarity. refer to CC1P description.
2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1P	RW	0	Capture/Compare 1 output Polarity. CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode). 01: inverted/falling edge Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode). 10: reserved, do not use this configuration.

Bit	Name	R/W	Reset Value	Function
				11: noninverted/both edges
0	CC1E	RW	0	Capture/Compare 1 output enable. CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled 1: Capture enabled

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

22.4.11. TIM2/3 counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
15:0	CNT[15:0]	RW	0	counter value

22.4.12. TIM2/3 prescaler (TIMx_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, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency CK_CNT is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event.

22.4.13. TIM 2/3 auto-reload register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
15:0	ARR[15:0]	RW	FFFF	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to Section 12.4.1: Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

22.4.14. TIM2/3 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
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 TIM3_CCMR1 register (bit OC1PE). Otherwise 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 TIM3_CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

22.4.15. TIM2/3 capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
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).

Bit	Name	R/W	Reset Value	Function
				It is loaded permanently if the preload feature is not selected in the TIM3_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 TIM3_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

22.4.16. TIM2/3 capture/compare register 3 (TIMx_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR3[15:0]	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 TIM3_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 TIM3_CNT and signaled on OC3 output. If channel CC3is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

22.4.17. TIM2/3 capture/compare register 4 (TIMx_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			Reserved, must be kept at reset value.
15:0	CCR4[15:0]	RW	0	Capture/Compare 4 value If CC4 channel 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 TIMx_CCMR4 register (bit OC4PE). Otherwise , the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If CC4 channel is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

22.4.18. TIM2/3 DMA control register (TIMx_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, must be kept at reset value.

Bit	Name	R/W	Reset Value	Function
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the number of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIMx_DMAR address). 00000: 1 transfer, 00001: 2 transfers, 00010: 3 transfers, ... 10001: 18 transfers. Example: Let's consider a transmission like this: DBL=7, DBA=TIM2_CR1 - If DBL = 7 and DBA = TIM2_CR1 denotes the address of the data to be transferred, then the address of the transfer is given by the following equation (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL where (address of TIMx_CR1) + DBA plus 7 gives the address where the data will be written or read, so that the transfer of data will take place in the 7 registers starting at address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following may occur: - If the data is set to half-word (16 bits), then the data is transferred to all 7 registers. - If the data is set to byte, the data is still transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte and so on. For timers, therefore, the user must specify the width of the data to be transferred by the DMA.
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address This 5-bit vector defines the base-address for DMA transfers (when read/write access are done through the TIM3_DMAR address). DBA is defined as an offset starting from the address of the TIM1_CR1 register. Example: 00000: TIMx_CR1, 00001: TIMx_CR2, 00010: TIMx_SMCR, ...

22.4.19. TIM2/3 DMA address for full transfer (TIMx_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 register for burst accesses A read or write operation to the DMAR register accesses the register located at the address (TIMx_CR1 address) + (DBA + DMA index) x 4 where TIMx_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIMx_DCR register, DMA index is automatically controlled by the DMA transfer, and ranges from 0 to DBL (DBL configured in TIMx_DCR).

22.4.20. TIM2/3 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIMx_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD [1:0]		ARPE	CMS [1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x04	TIMx_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS [2:0]			CCDS	Res.	Res.	Res.
	Reset value																									0	0	0	0	0			
0x08	TIMx_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETP.	ECE.	ETPS.		ETF.				MSM	TS[2:0]			OCCS	SMS[2:0]		
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIMx_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																		0		0	0	0	0	0		0		0	0	0	0	0
0x10	TIMx_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4IF.	IC3IF.	IC2IF.	IC1IF.	IC4IR.	IC3IR.	IC2IR.	IC1IR.	Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	0	0			0		0	0	0	0	0
0x14	TIMx_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																										0		0	0	0	0	0
0x18	TIMx_CCMR1(output	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M [2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	compare mode)																																
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIMx_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIMx_CCMR2(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIMx_CCMR2(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIMx_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
	Reset value																									0		0	0	0		0	0
0x24	TIMx_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIMx_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIMx_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIMx_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIMx_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIMx_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIMx_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x48	TIMx_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.			DBA[4:0]					
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	TIMx_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23. General-purpose timer (TIM6/7)

23.1. Introduction of TIME6 and TIM7

The general-purpose timers TIM6 and TIM7 each contain a 16-bit auto-load counter driven by the respective programmable prescaler.

They can be used as general purpose timers to provide a time reference and in particular to clock digital-to-analogue converters (DACs). In fact, they are connected directly to the DAC inside the chip and drive it directly via the trigger output.

TIM6 and TIM7 are both completely independent and do not share any resources with each other

23.2. TIME6 and TIM7 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”)
- Trigger DAC synchronisation circuits
- Generate interrupts/DMAs when update events occur

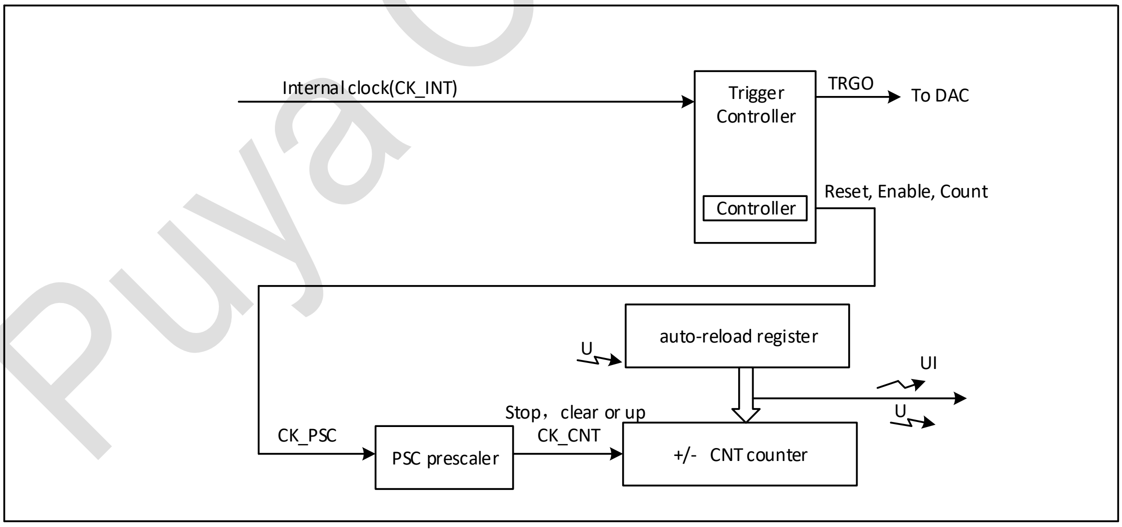


Figure 23-1 General-purpose timer block diagram (TIM6/7)

23.3. TIME6/TIM7 functional description

23.3.1. Time-base unit

The main block of the programmable general-purpose timer is a 16-bit upcounter 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 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.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. 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.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly.

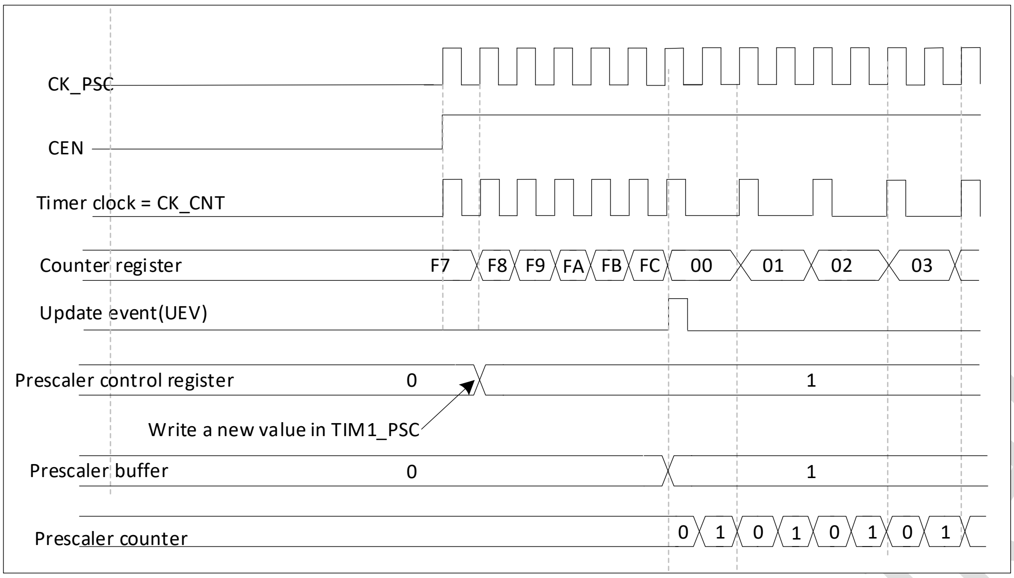


Figure 23-2 Counter timing diagram with prescaler division change from 1 to 2

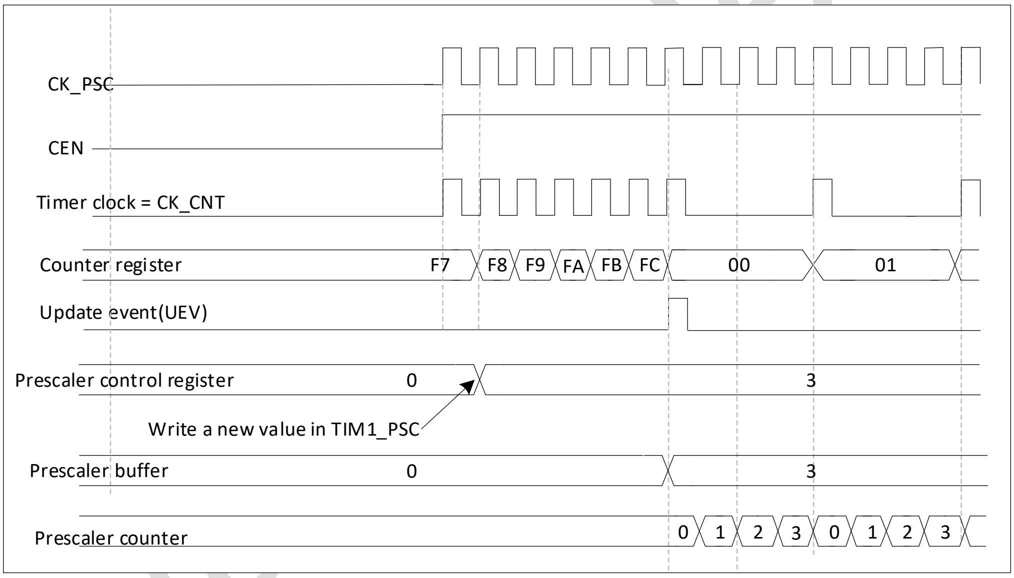


Figure 23-3 Counter timing diagram with prescaler division change from 1 to 4

Upcounter mode

The counter counts from 0 to the auto-reload value (content of the TIM14_ARR register), then restarts from 0 and generates a counter overflow event.

Every time the count overflows, an update event is generated. Setting the UG bit in the TIM14_EGR register also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM14_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 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.

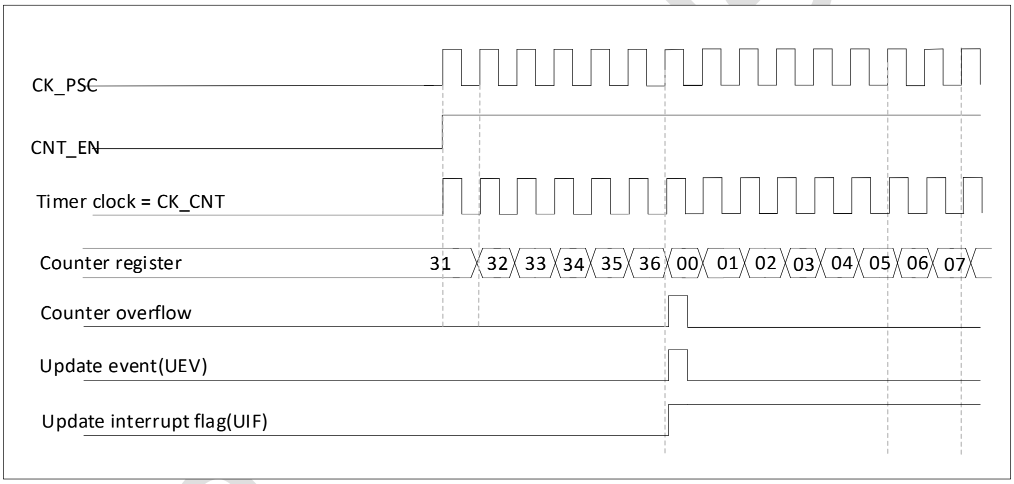


Figure 23-4 Counter timing diagram, internal clock divided by 1

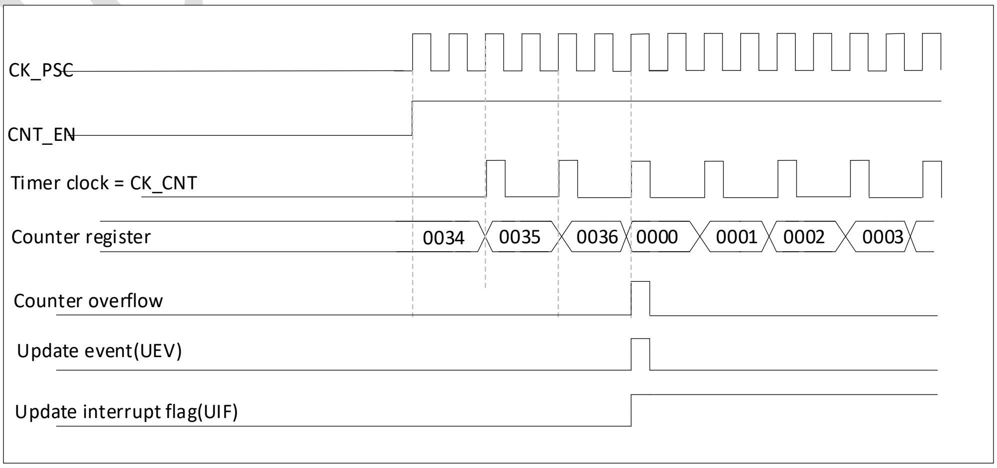


Figure 23-5 Counter timing diagram, internal clock divided by 2

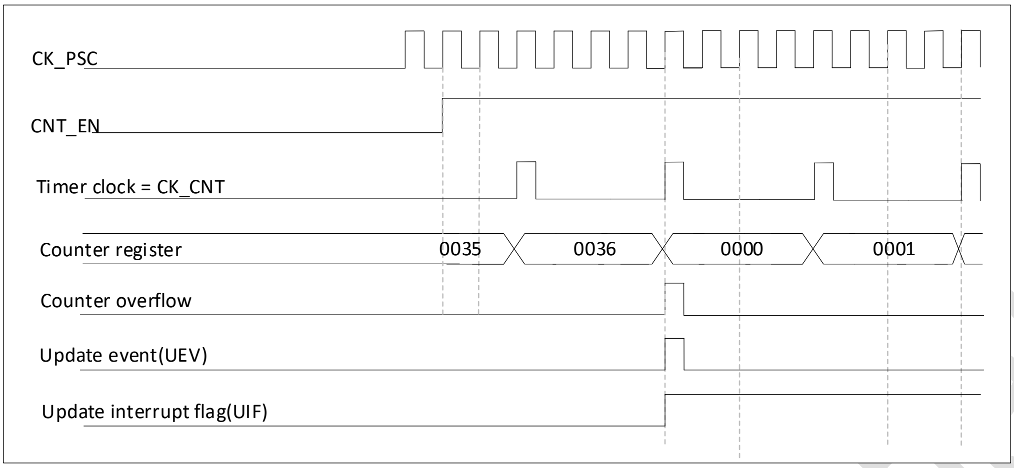


Figure 23-6 Counter timing diagram, internal clock divided by 4

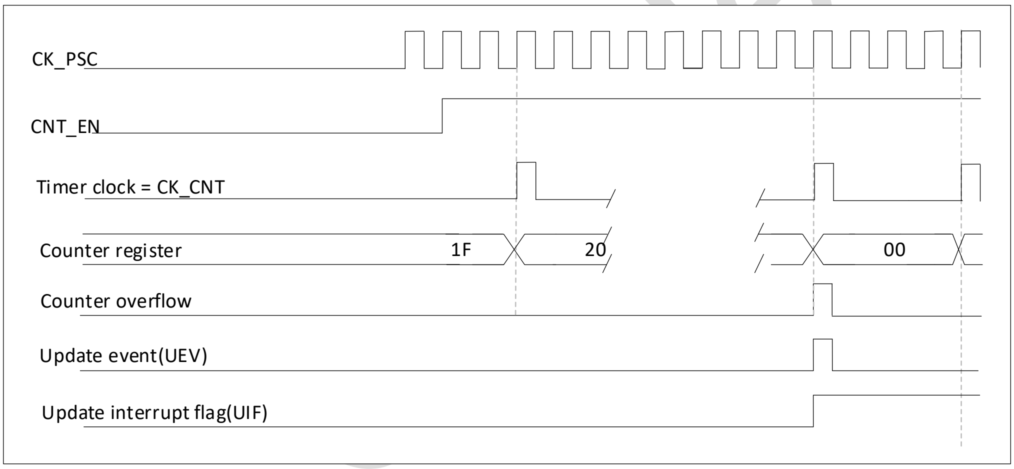


Figure 23-7 Counter timing diagram, internal clock divided by N

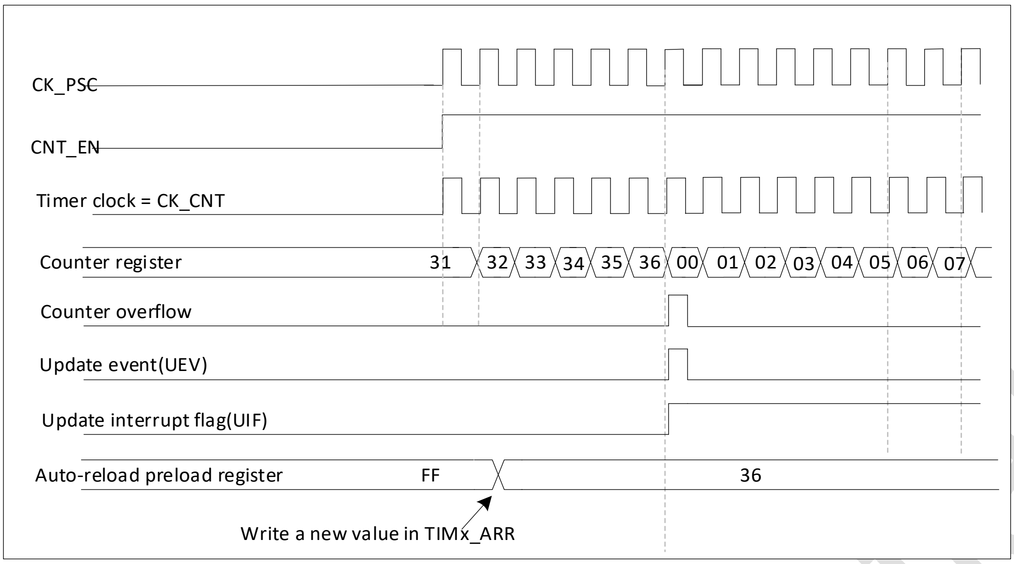


Figure 23-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR not preloaded)

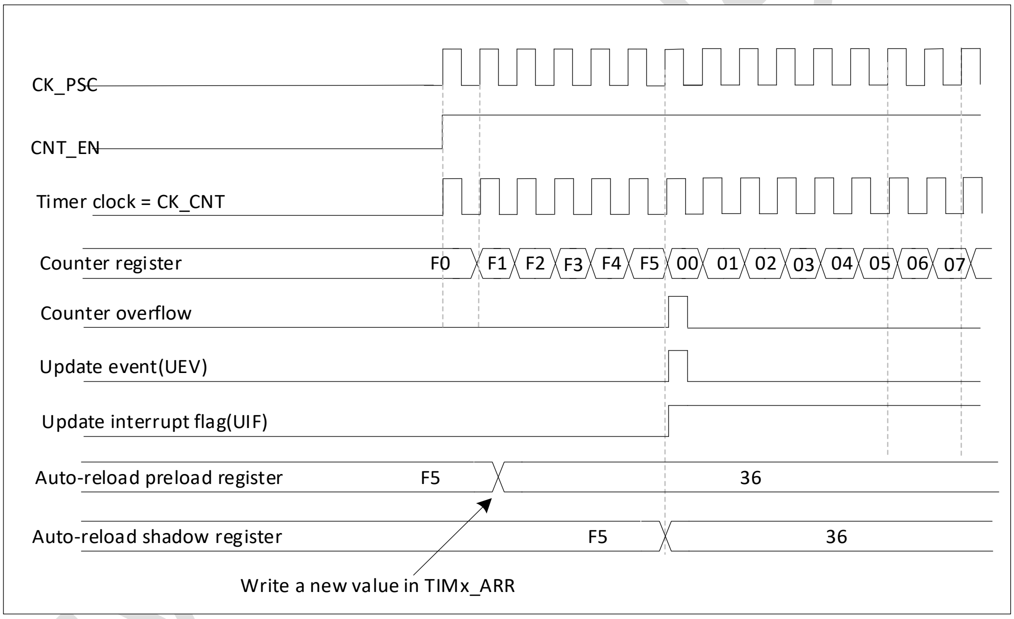


Figure 23-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

23.3.2. Clock source

The counter clock is provided by the Internal clock (CK_INT) source. The CEN (in the TIMx_CR1 register) and UG bits (in the TIM14_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

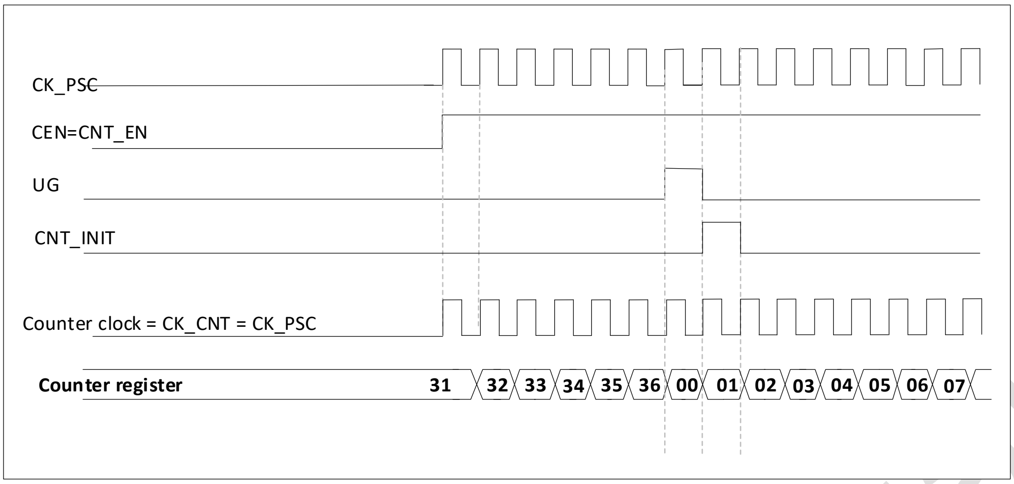


Figure 23-10 Control circuit in normal mode, internal clock divided by 1

23.3.3. Debug mode

When the microcontroller enters debug mode (M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

23.4. TIM6/TIM7 registers

23.4.1. TIM6/7 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	Res		ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
-	-	-	-	-	-	-		RW	-	-	-	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	0	Reserved, must be kept at reset value.
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:4	Reserved	-	0	Reserved, must be kept at reset value.
3	OPM	RW	0	One pulse mode 0: Counter is not stopped at update event

Bit	Name	R/W	Reset Value	Function
				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: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
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 only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware.

23.4.2. TIM6/7 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	Res		Res	MMS[2:0]			Res	Res	Res	Res
-	-	-	-	-	-	-		-	RW			-	-	-	-

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	0	Reserved, must be kept at reset value.
6:4	MMS[2:0]	RW	0	Main mode selection These 3 bits are used to select the synchronisation message (TRGO) sent to the slave timer in master mode. The possible combinations are as follows: 000: Reset - The UG bit of the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by a trigger input (from the mode controller being in reset mode), there will be a delay in the signal on TRGO relative to the actual reset. 001: Enable - The counter enable signal CNT_EN is used as a trigger output (TRGO). Sometimes it is necessary to start multiple timers at the same time or to control the enabling of slave timers over a period of time. The counter enable signal is generated by a logical or of the trigger input signal in CEN control bit and gated mode. When the counter enable signal is controlled by the trigger input, there is a delay on the TRGO unless Master/Slave mode is selected (see description of the MSM bit in the TIMx_SMCR register). 010: Update - The update event is selected as the trigger input (TRGO). For example, the clock of a master timer can be used as a prescaler for a slave timer. Note: The clock of the slave timer and ADC must be enabled first to receive the signal from the master timer and not changed when it is received.
3:0	Reserved	-	0	Reserved, must be kept at reset value.

23.4.3. TIM6/7 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	Res	Res	Res	Res	Res	Res	UDE	Res	Res	Res	Res	Res	Res	Res	UIE
-	-	-	-	-	-	-	RW	-	-	-	-	-	-	-	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved, must be kept at reset value.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled.
7:1	Reserved	-	-	Reserved, must be kept at reset value.
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

23.4.4. 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	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved, must be kept at reset value.
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 and if UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by software using the UG bit in the TIMx_EGR register, if URS = 0 and UDIS = 0 in the TIMx_CR1 register.

23.4.5. TIM6/7 event generation register (TIMx_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															UG
-															W

Bit	Name	R/W	Reset Value	Function
31：1	Reserved	-	0	Reserved, must be kept at reset value
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action. 1: Re-initializes the timer counter and generates an up-date of the registers. Note that the prescaler counter is cleared too (but the prescaler ratio is not affected).

23.4.6. TIM6/7 counter (TIMx_CNT)

Address offset:0x24

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

23.4.7. TIM6/7 prescaler (TIMx_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, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

23.4.8. TIM6/7 auto-reload register (TIMx_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	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, must be kept at reset value.
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.

23.4.9. TIM6/7 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIMx_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		ARPE	Res.		Res.	OPM.	URS	UDIS	CEN
	Reset value																									0				0	0	0	0
0x04	TIMx_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMS[2:0]			Res.	Res.	Res.	Res.
	Reset value																										0	0	0				
0x0C	TIMx_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
	Reset value																								0								0
0x10	TIMx_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
	Reset value																																0
0x14	TIMx_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
	Reset value																																0
0x18	TIM14_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x24	TIM14_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

24. General-purpose timer (TIM14)

24.1. TIM14 introduction

The TIM14 general-purpose timer consists 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 timer is completely independent, and does not share any resources. It can be synchronized together as described in TIM3.

24.2. 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

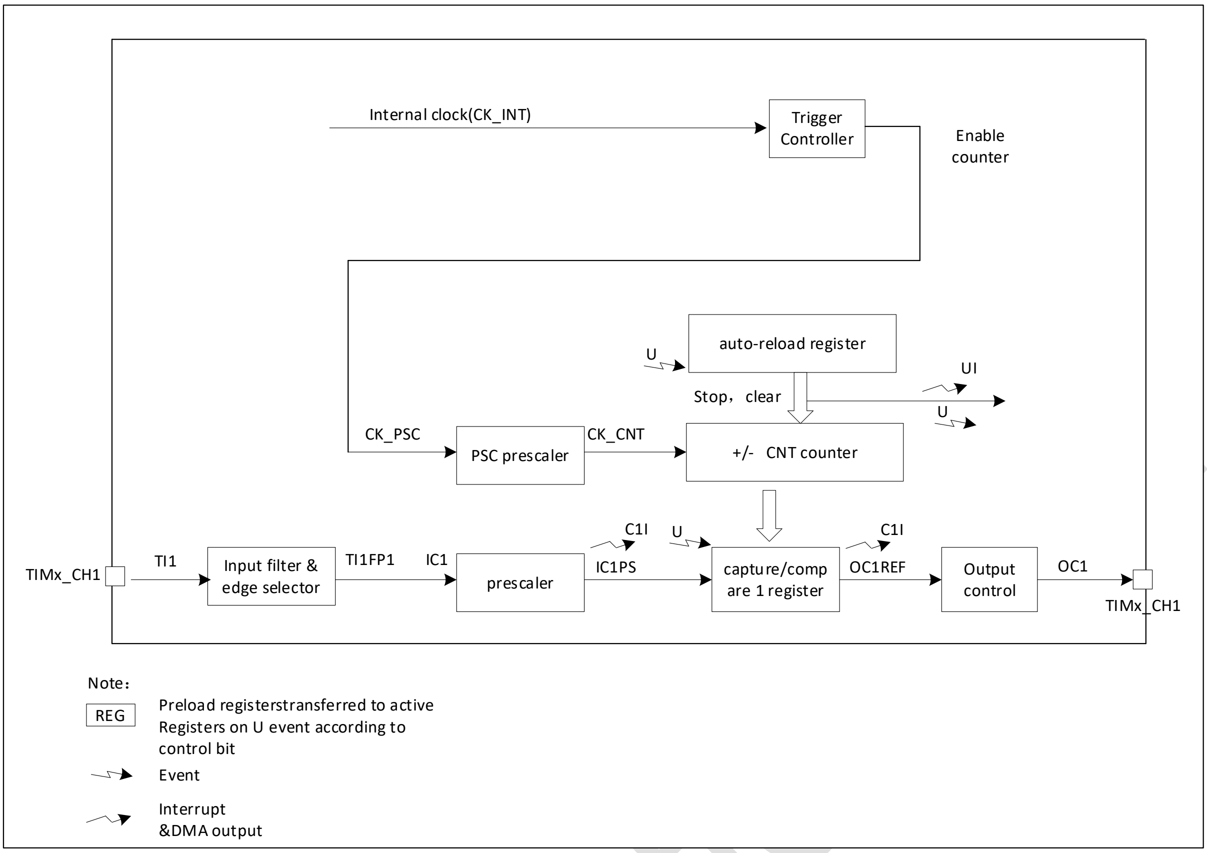


Figure 24-1 General-purpose timer block diagram (TIM14)

24.3. TIM14 functional description

24.3.1. Time-base unit

The main block of the programmable general-purpose timer is a 16-bit upcounter 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 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.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. 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.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly.

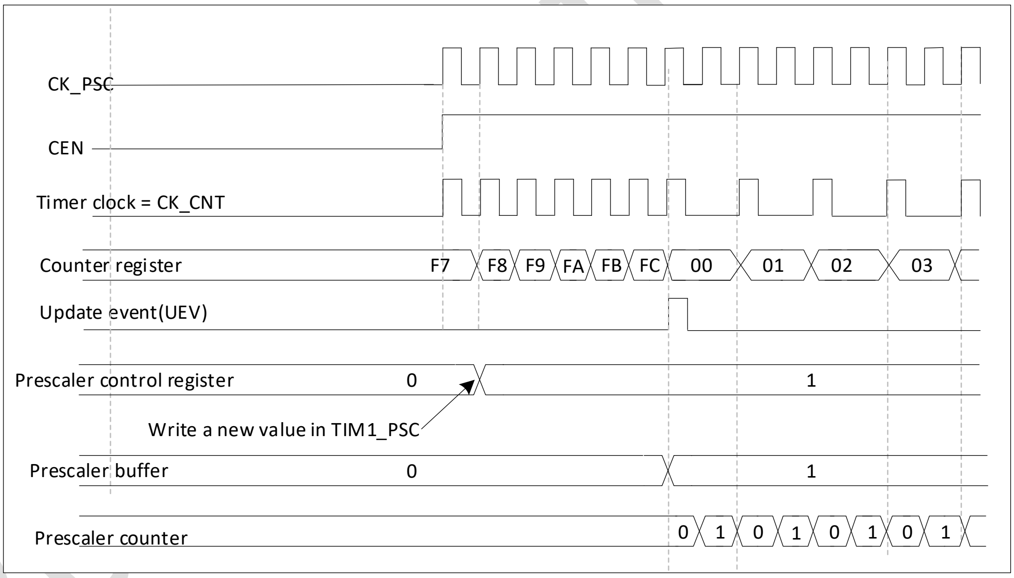


Figure 24-2 Counter timing diagram with prescaler division change from 1 to 2

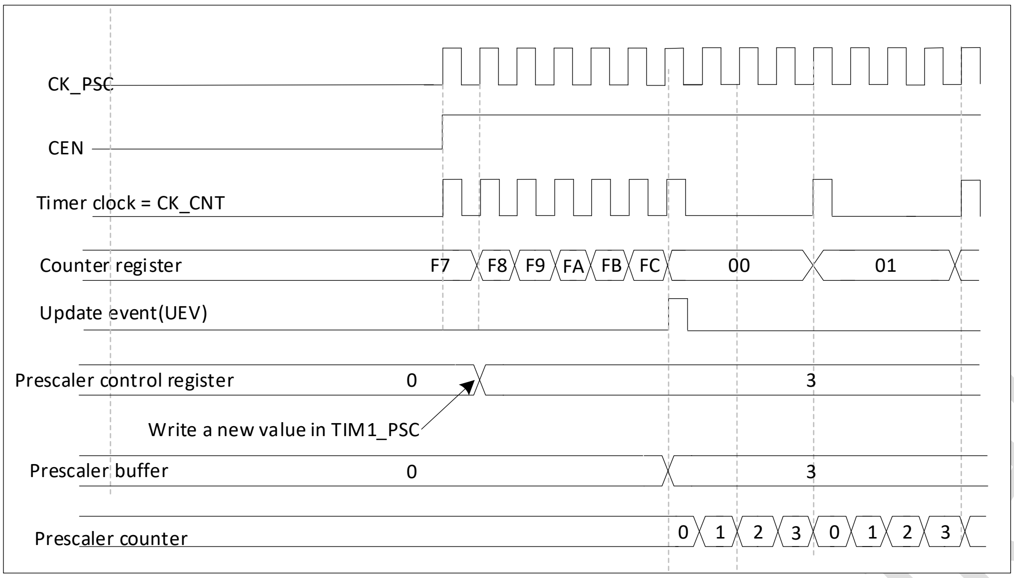


Figure 24-3 Counter timing diagram with prescaler division change from 1 to 4

Upcounter mode

The counter counts from 0 to the auto-reload value (content of the TIM14_ARR register), then restarts from 0 and generates a counter overflow event.

Every time the count overflows, an update event is generated. Setting the UG bit in the TIM14_EGR register also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM14_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 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.

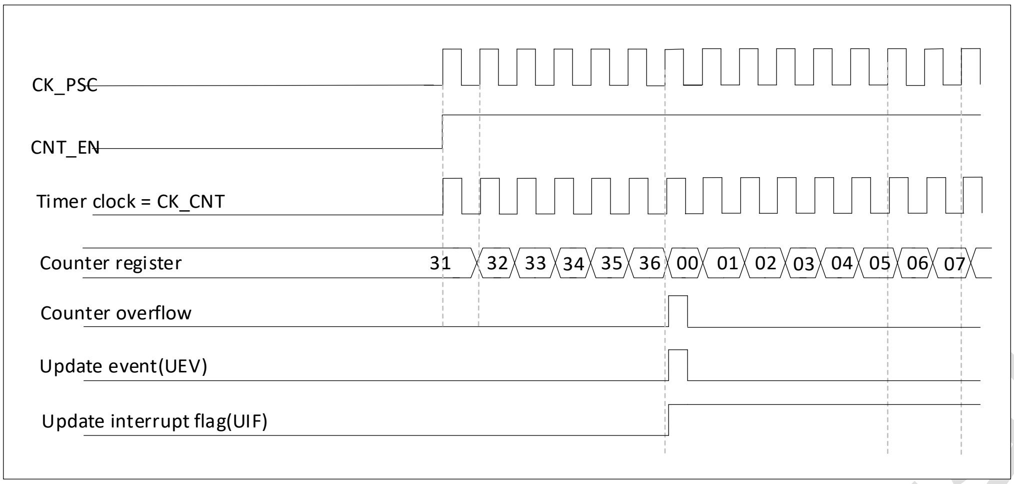


Figure 24-4 Counter timing diagram, internal clock divided by 1

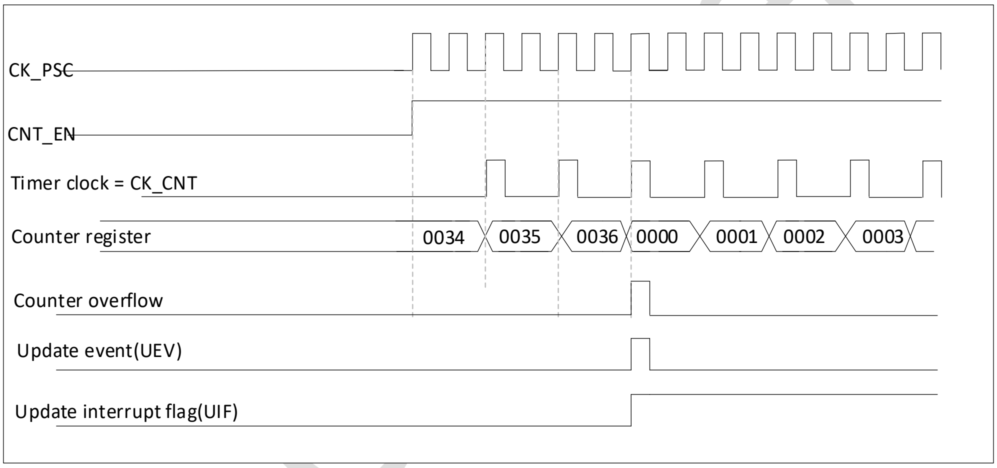


Figure 24-5 Counter timing diagram, internal clock divided by 2

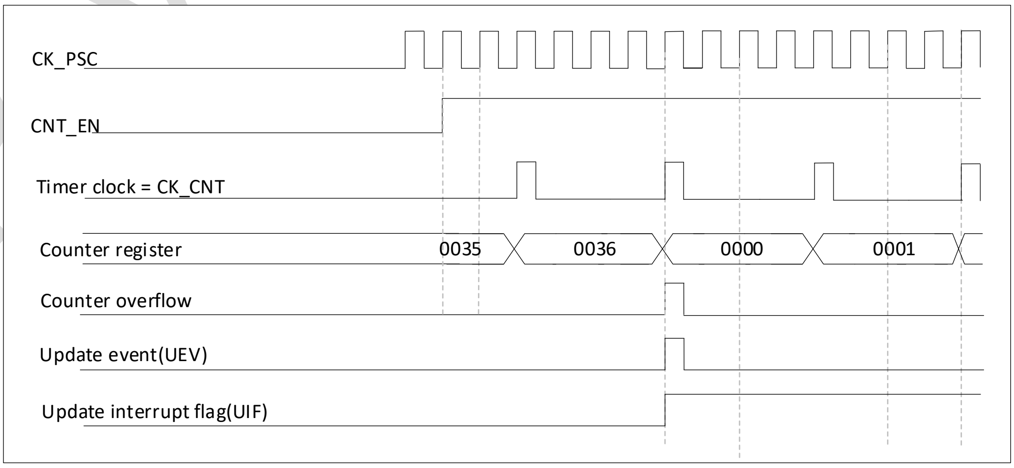


Figure 24-6 Counter timing diagram, internal clock divided by 4

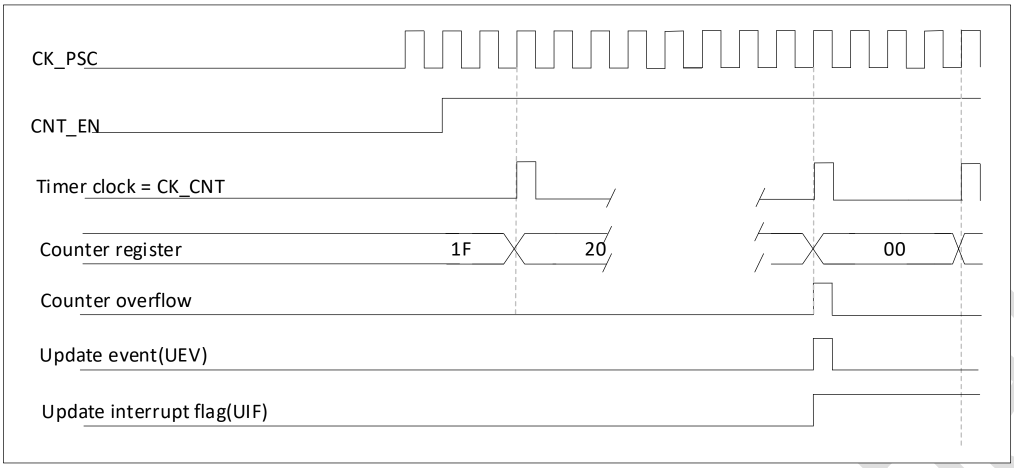


Figure 24-7 Counter timing diagram, internal clock divided by N

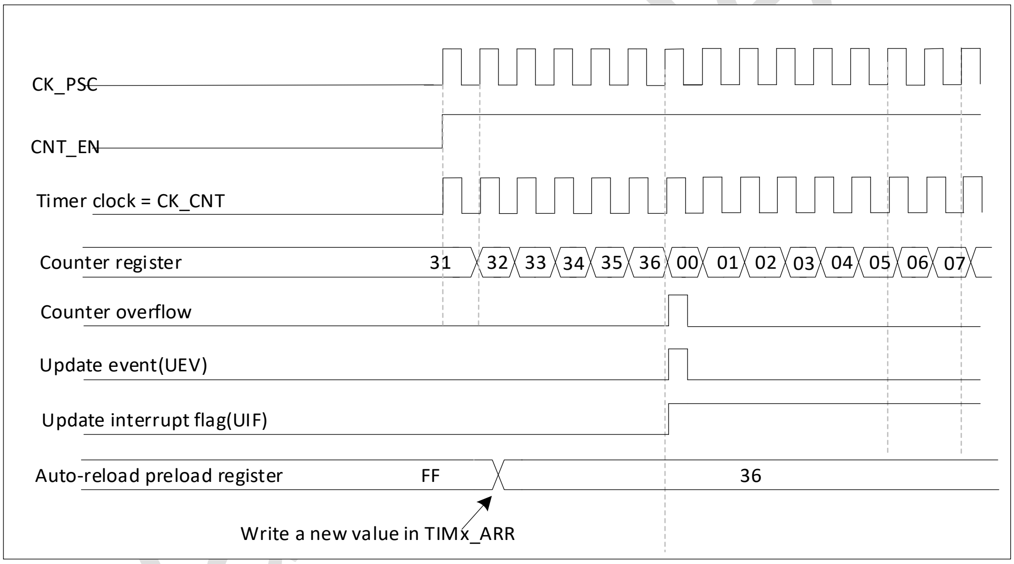


Figure 24-8 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR not preloaded)

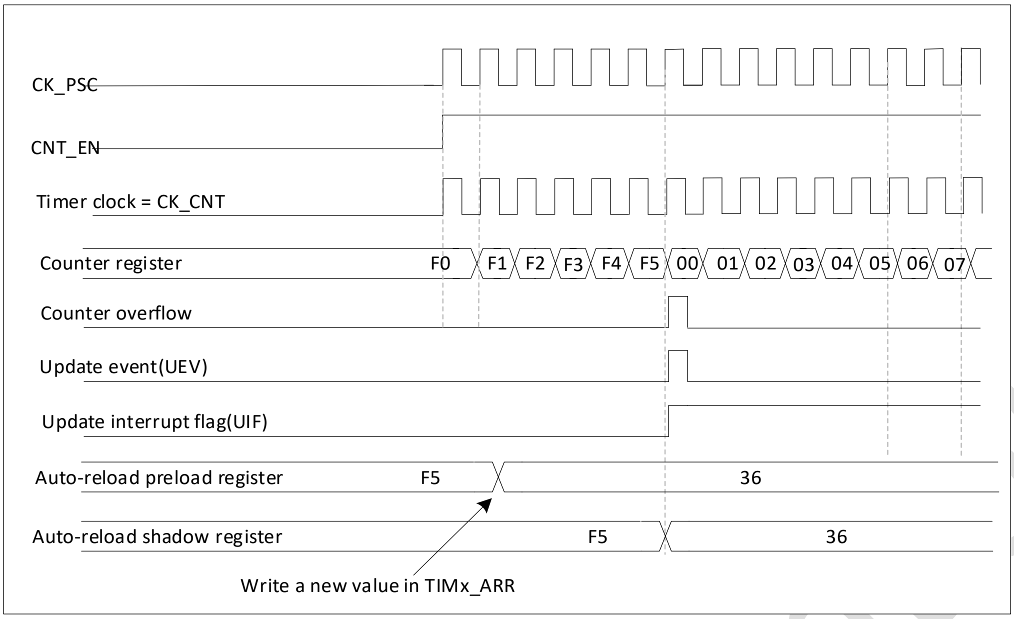


Figure 24-9 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

24.3.2. Clock source

The counter clock is provided by the Internal clock (CK_INT) source. The CEN (in the TIMx_CR1 register) and UG bits (in the TIM14_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

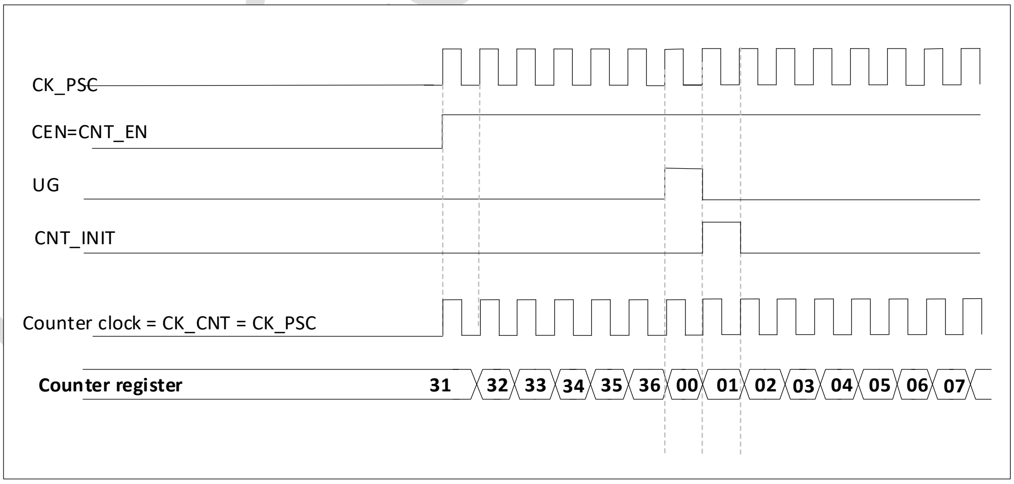


Figure 24-10 Control circuit in normal mode, internal clock divided by 1

24.3.3. 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).

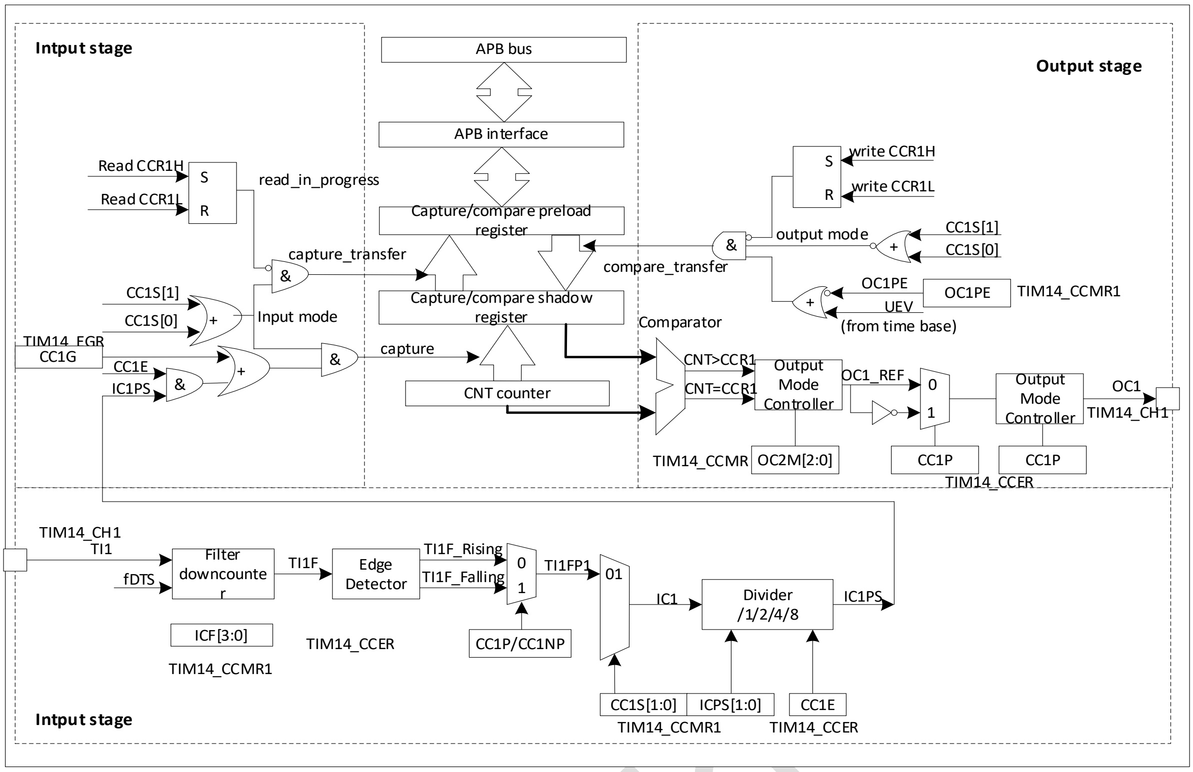


Figure 24-11 TIM14 Capture/compare channel

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).

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.

The capture/compare block is made of one preload register and one shadow register. Write and read only 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

24.3.4. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM14_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIM14_SR register) is set. 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 it is written with 0.

The following example shows how to capture the counter value in TIM14_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIM14_CCR1 must be linked to the TI1 input, so write the CC1S bits to ‘01’ in the TIM14_CCMR1 register. As soon as CC1S becomes different from ‘00’, the channel is configured in input mode and the TIM14_CCR1 register becomes readonly.
- Program the appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIM14_CCMRx register). 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 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.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER 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). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.

- In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

24.3.5. Forced output mode

In output mode (CCxS bits = ‘00’ in the TIM14_CCMRx register), each output compare signal (OCxREF and then OCx) 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, one just needs to write ‘101’ 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.

The comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the output compare mode section below.

24.3.6. Output compare mode

This function is used to control an output waveform or to indicate 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 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 TIM14_DIER register).

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).(This is meaningless, no OPM)

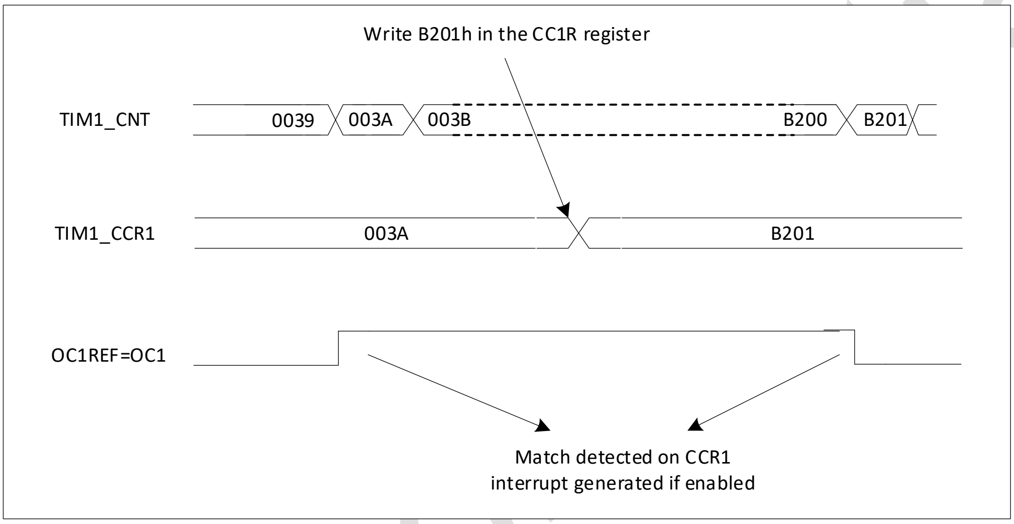


Figure 24-12 Output compare mode, toggle on OC1

24.3.7. PWM mode

Pulse Width Modulation mode allows 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. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register 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.

The 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 OCx output is enabled by the CCxE bit in the TIM14_CCER register.

In PWM mode (1 or 2), TIM14_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 < TIM14_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'.

The following figure shows some edgealigned PWM waveforms in an example where TIMx_ARR = 8.

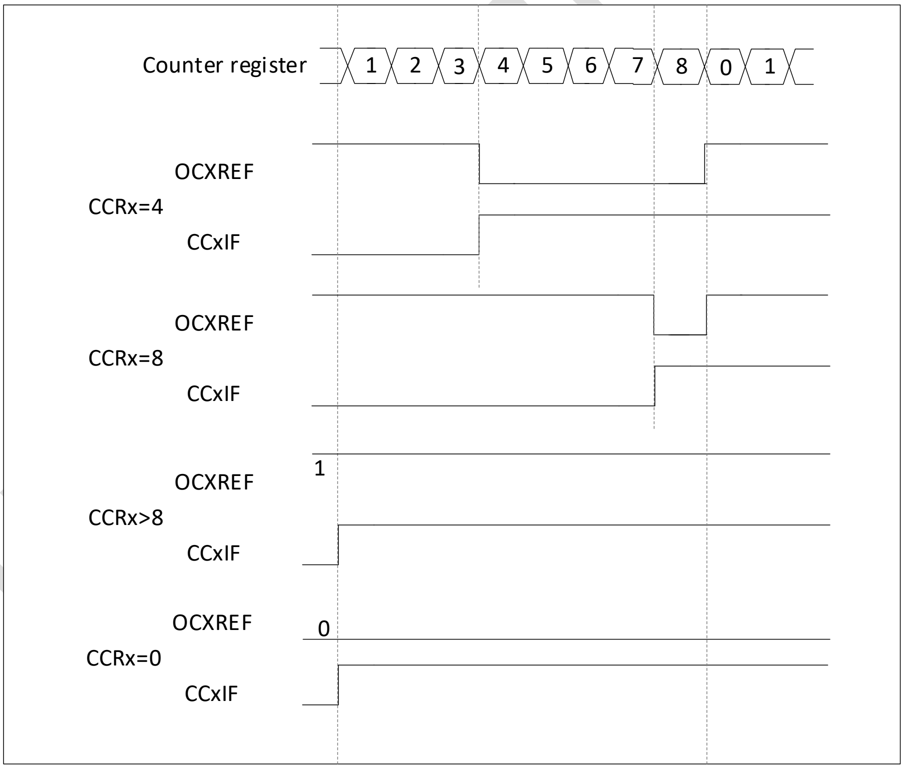


Figure 24-13 Edge-aligned PWM waveforms (ARR = 8)

24.3.8. One pulse mode

The one pulse mode (OPM) is a special case of one of the many modes described earlier. This mode allows the counter to respond to an excitation and produce a pulse with a programmable pulse width after a programmable delay.

The counter can be started from the mode controller to generate a waveform in either output compare mode or PWM mode. Setting the OPM bit in the TIMx_CR1 register will select the one pulse mode, which allows the counter to automatically stop when the next update event UEV is generated.

A pulse can only be generated if the comparison value is different from the initial value of the counter. Before starting (when the timer is waiting to be triggered), the following configuration must be made:

- Upward counting mode: counter CNT < CCRx ≤ ARR (in particular, 0 < CCRx)

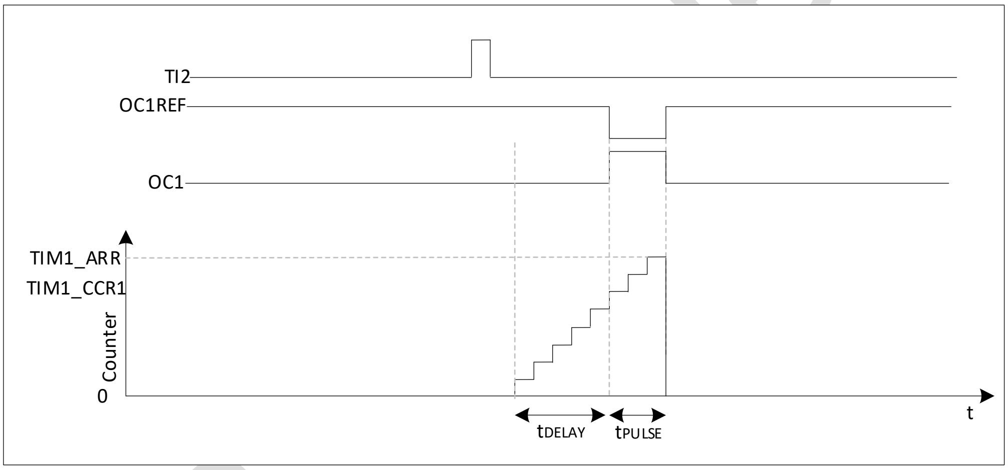


Figure 24-14 Example of a single pulse mode

For example, you need to generate a positive pulse of length tPULSE on OC1 after a delay of tDELAY, starting from a rising edge detected on the TI2 input pin.

Assume TI2FP2 as the trigger.

- Set CC2S=01 in the TIMx_CCMR1 register to map TI2FP2 to TI2.
- Set CC2P=0 in the TIMx_CCER register to enable the TI2FP2 to detect rising edges.
- Set TS=110 in the TIMx_SMCR register to trigger the TI2FP2 as a slave mode controller (TRGI).

- Set SMS=110 in the TIMx_SMCR register (trigger mode) and the TI2FP2 is used to start the counter.

The OPM waveform is determined by the value written to the compare register (taking into account the clock frequency and the counter prescaler)

- tDELAY is defined by the value in the TIMx_CCR1 register.
- tPULSE is defined by the difference between the auto-load value and the compare value (TIMx_ARR - TIMx_CCR1).
- Assume 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 in the TIMx_CCMR1 register to enter PWM mode 2; selectively enable the preload registers as required: set OC1PE = 1 in TIMx_CCMR1 and TIMx_CR1 register; then fill in the comparison value in the TIMx_CCR1 register and the auto-load value in the TIMx_ARR register, set the UG bit to generate an update event and wait for an external trigger event on TI2. In this example, CC1P = 0.

In this example, the DIR and CMS bits in the TIMx_CR1 register should be set low.

Since only one pulse is required, OPM=1 in the TIMx_CR1 register must be set to stop counting on the next update event (when the counter flips from the auto-load value to 0). When OPM = 0, repeat mode is selected.

Timer synchronisation

All TIMx timers are internally linked for timer synchronisation or linking. When a timer is in master mode, it can reset, start, stop or provide a clock to the counter of another timer in slave mode. tim10/11/13/14 act as master timer output oc to the slave timer TIM9/12 in the corresponding configuration.

24.3.9. Debug mode

When the microcontroller enters debug mode (M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

24.4. TIM14 registers

24.4.1. 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	-	0	Reserved, must be kept at reset value.
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx), 00: tDTS = tCK_INT 01: tDTS = 2 × tCK_INT 10: tDTS = 4 × tCK_INT 11: Reserved
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:3	Reserved	-	0	Reserved, must be kept at reset value.
2	URS	RW	0	Update request source This bit is set by software to select the update interrupt (UEV) sources. 0: Any of the following events generate an UEV or a DMA request if enabled: – Counter overflow – Setting the UG bit 1: Only counter overflow generates an UEV or a DMA re-quest if enabled.
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable update interrupt (UEV) event generation.

Bit	Name	R/W	Reset Value	Function
				0: UEV enabled. An UEV is generated by one of the following events: – Counter overflow – Setting the UG bit. Buffered registers are then loaded with their preload values. 1: UEV disabled. No UEV is generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and the prescaler are reinitialized if the UG bit is set.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware.

24.4.2. 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, must be kept at reset value.
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

24.4.3. 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	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						CC1OF	Res							CC1IF	UIF
-						Rc_w0	-							Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	0	Reserved, must be kept at reset value.
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 TIM14_CCR1 register while CC1IF flag was already set
8:2	Res	Rc_w0	0	Reserved, must be kept at reset value.
1	CC1IF	Rc_w0	0	Capture/compare 1 interrupt flag Condtion: channel CC1 is configured as output This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match. 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. Condition: 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.

Bit	Name	R/W	Reset Value	Function
				– When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS = '0' and UDIS = '0' in the TIMx_CR1 register.

24.4.4. 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	-	0	Reserved, must be kept at reset value.
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 1: Condition: channel CC1 is configured as output CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. Condition: channel CC1 is configured as input The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared.

24.4.5. 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, must be kept at reset value.
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, OC1N is derived. OC1REF is active high whereas OC1, OC1N active level depends on CC1P, CC1NP bit. 000: Frozen. The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. 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 - when upcounting, Channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. when downcounting, Channel 1 is inactive(OC1REF = 0) as long as TIMx_CNT > TIMx_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 output compare mode switches from frozen to PWM mode.

Bit	Name	R/W	Reset Value	Function
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	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 CC1 output. OC is then set to the compare level independently of the result of the comparison. Delay to sample the trigger input and to activate CC1 output is re-duced 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, must be kept at reset value.
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[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	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 Other: Reserved

Bit	Name	R/W	Reset Value	Function
				Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM14_CCER).

24.4.6. 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	-	0	Reserved, must be kept at reset value.
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 define TI1FP1 polarity (refer to CC1P description).
2	Reserved	-	0	Reserved, must be kept at reset value.
1	CC1P	RW	0	Input/Capture 1 output Polarity. Condition: CC1 channel configured as output 0: OC1 active high 1: OC1 active low Condition: CC1 channel configured as input The CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge Circuit is sensitive to TI1FP1 rising edge (capture mode, reset trigger, external clock or trigger mode), TI1FP1 is not inverted(gate mode, code mode). 01: inverted/falling edge Circuit is sensitive to TI1FP1 falling edge (capture mode, reset trigger, external clock or trigger mode), TI1FP1 is inverted(gate mode, code mode). 10: reserved, do not use this configuration. 11: noninverted/both edges

Bit	Name	R/W	Reset Value	Function
0	CC1E	RW	0	Input/Capture 1 output enable. Condition: CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin Condition: CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture register 1 (TIMx_CCR1) or not. 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

24.4.7. 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, must be kept at reset value.
15:0	CNT[15:0]	RW	0	Counter value

24.4.8. 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, must be kept at reset value.
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency CK_CNT is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event. Cleared to 0 by the UG bit in TIM_EGR or by a slave con-troller operating in reset mode.

24.4.9. 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, must be kept at reset value.
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the Time-base unit for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

24.4.10. 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, must be kept at reset value.
15:0	CCR1[15:0]	RW	0	Capture/Compare 1 value Condition: 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. Condition: channel CC1is configured as input CCR1 is the counter value transferred by the last input capture 1 event (IC1).

24.4.11. TIM14 option register (TIM14_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	-	0	Reserved, must be kept at reset value.
1:0	TI1_RMP	RW	0	Timer Input 1 remap Set and cleared by software. 00: TIM14 Channel1 is connected to the GPIO. Refer to the alternate function mapping in the device datasheets. 01: TIM14 Channel1 is connected to the RTCCLK.

Bit	Name	R/W	Reset Value	Function
				10: TIM14 Channel1 is connected to the HSE/32 Clock. 11: TIM14 Channel1 is connected to the microcontroller clock output (MCO), this selection is controlled by the MCO[2:0] bits of the Clock configuration register (RCC_CFGR)

24.4.12. TIM14 register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM14_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.		Res.	OPM.	URS	UDIS	CEN
	Reset value																							0	0	0				0	0	0	0
0x0C	TIM14_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1I	UIE
	Reset value																															0	0
0x10	TIM14_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1IF	Res.	Res.	IC1IR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1O	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	UIF
	Reset value												0			0								0								0	0
0x14	TIM14_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1G	UG
	Reset value																															0	0
0x18	TIM14_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	Res.	CC1S[1:0]	
	Reset value																										0	0	0	0		0	0
0x18	TIM14_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																									0	0	0	0	0	0	0	0
0x20	TIM14_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	Res.	CC1P	CC1E
	Reset value																													0		0	0
0x24	TIM14_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x34	TIM14_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	TIMx_OR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1_RMP[1:0]	
	Reset value																															0	0

25. Advanced Control timers (TIM15/16/17)

25.1. Introduction

The advanced timer (TIM15_16_17) consists of a 16-bit auto-load counter driven by a programmable divider. It can be used in a variety of scenarios including: pulse length measurement of the input signal (input capture) or to generate output waveforms (output compare, output PWM, complementary PWM with deadband insertion).

Pulse length and waveform period can be modulated from microseconds to milliseconds using the timer divider and the RCC clock control divider. The advanced timer (TIM15_16_17) and the general purpose (TIMx) timer are completely independent and do not share any resources. They can be synchronised together.

25.2. TIM15 main features

- 16-bit up, down or up-down auto-reload upcounter
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65536
- Up two channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge-aligned mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time(only channel 1)
- Synchronous circuits for controlling timers and timer interconnections using external signals
- Repetition counter to update the timer registers only after a given number of cycles of the counter
- Break input to put the timer’s output signals in the reset state or a known state
- Interrupt/DMA generation on the following events:
- Update: counter overflow up and down, counter initialisation (via software or internal or external trigger)

- Trigger event
- Input capture
- Output compare
- Break input

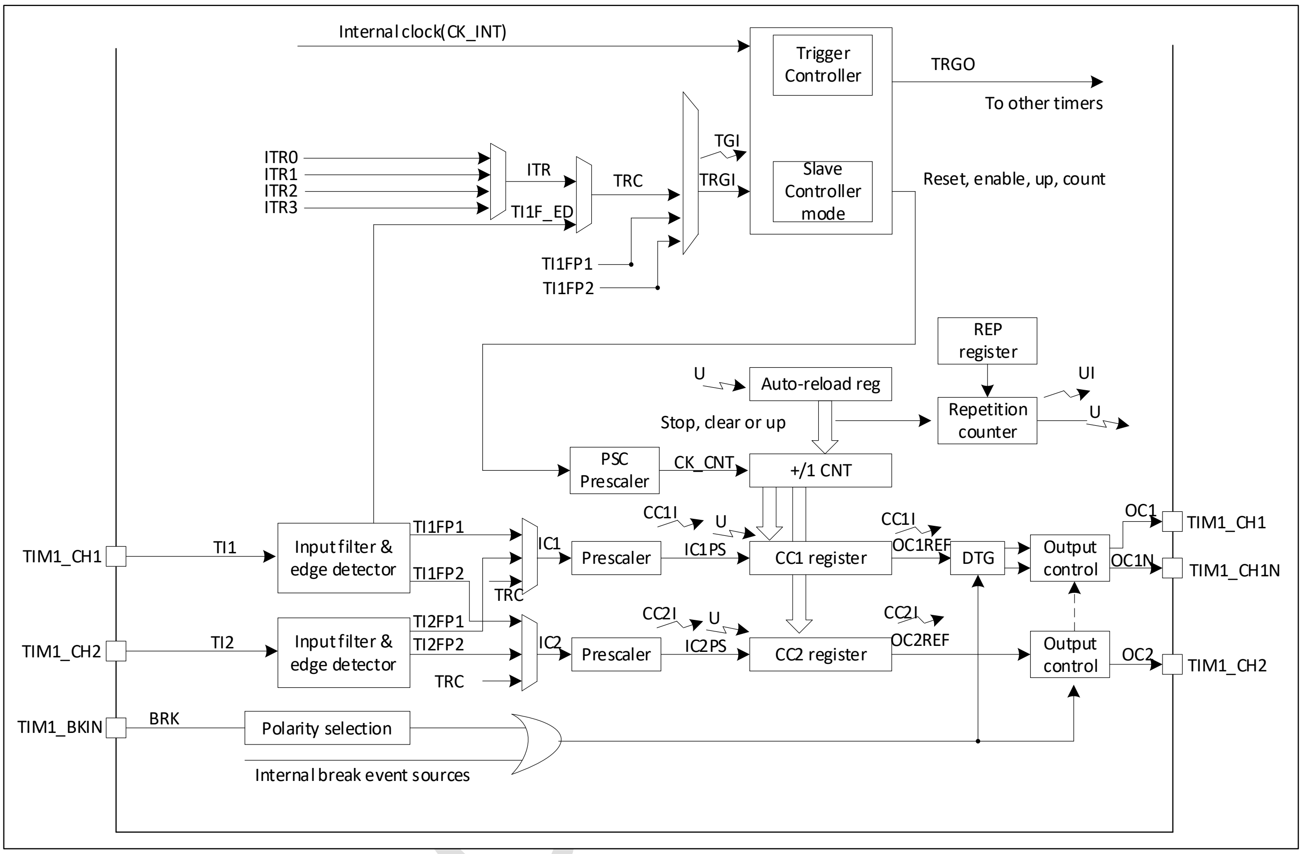


Figure 25-1 Block Diagram of advanced Control Timer (TMI15)

25.3. TIM16 and TIM17 main features

- 16-bit auto-reload upcounter
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65536
- One independent channel for:
  - Input capture
  - Output compare
  - PWM generation (Edge-aligned mode)
  - One-pulse mode output
- Complementary outputs with programmable dead-time

- Repetition counter to update the timer registers only after a given number of cycles of the counter
- Break input to put the timer's output signals in the reset state or a known state
- Interrupt/DMA generation on the following events:
  - Update: counter overflow
  - Input capture
  - Output compare
  - Break input

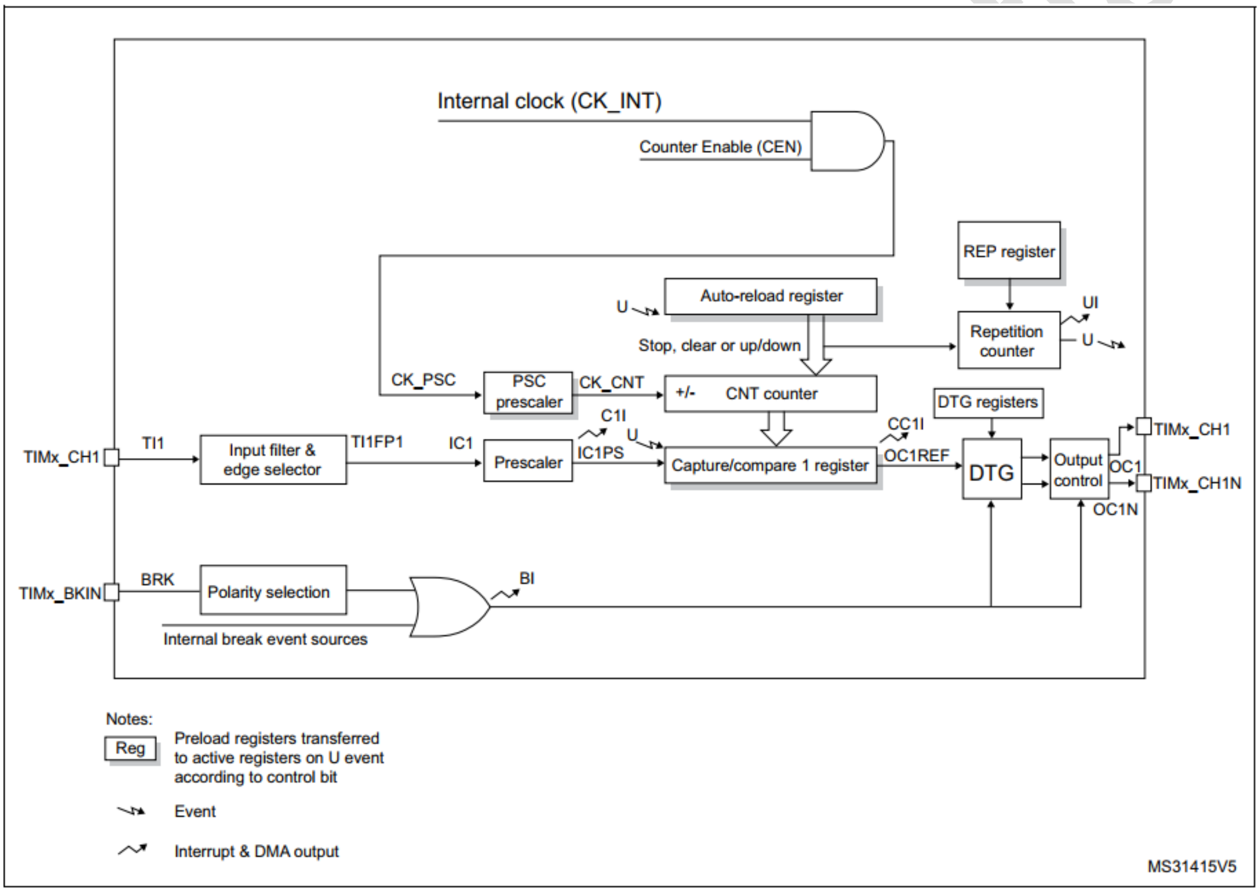


Figure 25-2 TIM16 and TIM17 block diagram

25.4. TIM15_16_17 functional description

25.4.1. Time-base unit

The main block of the programmable general purpose timer is a 16-bit upcounter 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 (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

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 TIMx_CR1 register. The update event is sent when the counter reaches the overflow 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_CR1 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.

The following figures give some examples of the counter behavior when the prescaler ratio is changedy:

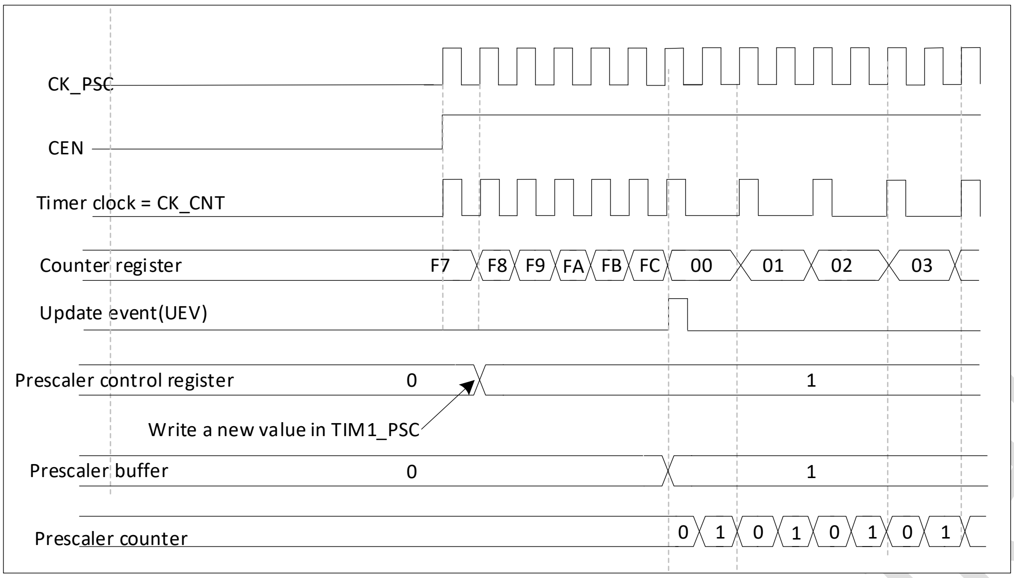


Figure 25-3 Counter timing diagram with prescaler division change from 1 to 2

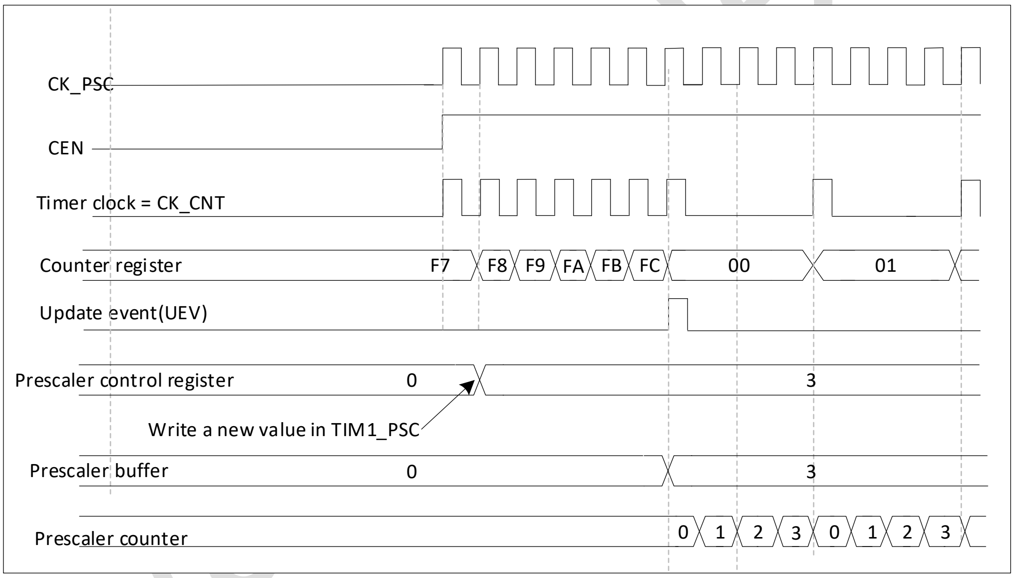


Figure 25-4 Counter timing diagram with prescaler division change from 1 to 4

25.4.2. Counter operation

Upward counting mode

The counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), 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 (TIMx_RCR). 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_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 auto-reload 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.

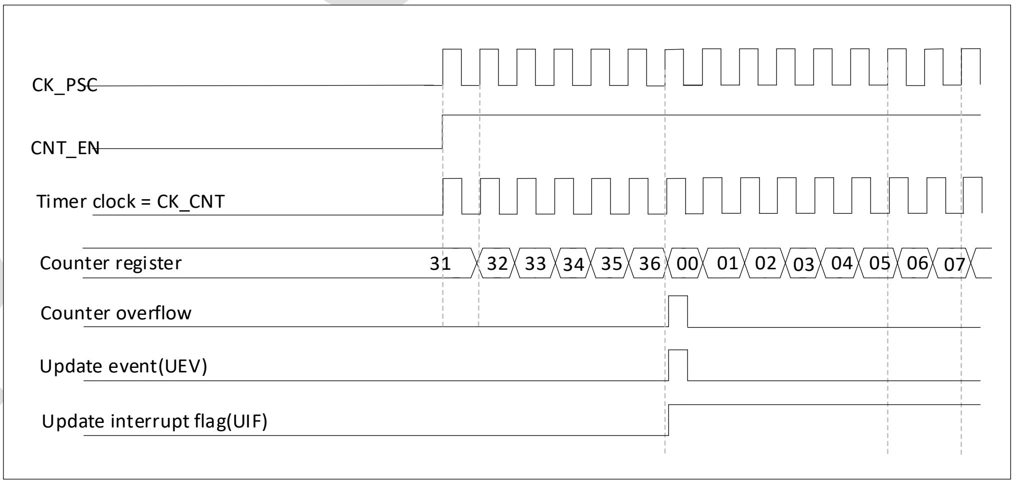


Figure 25-5 Counter timing diagram, internal clock divided by 1

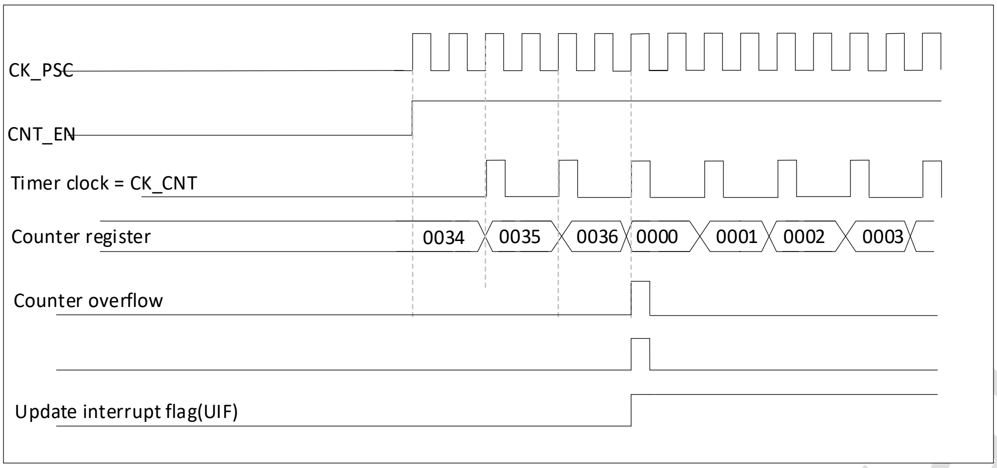


Figure 25-6 Counter timing diagram, internal clock divided by 2

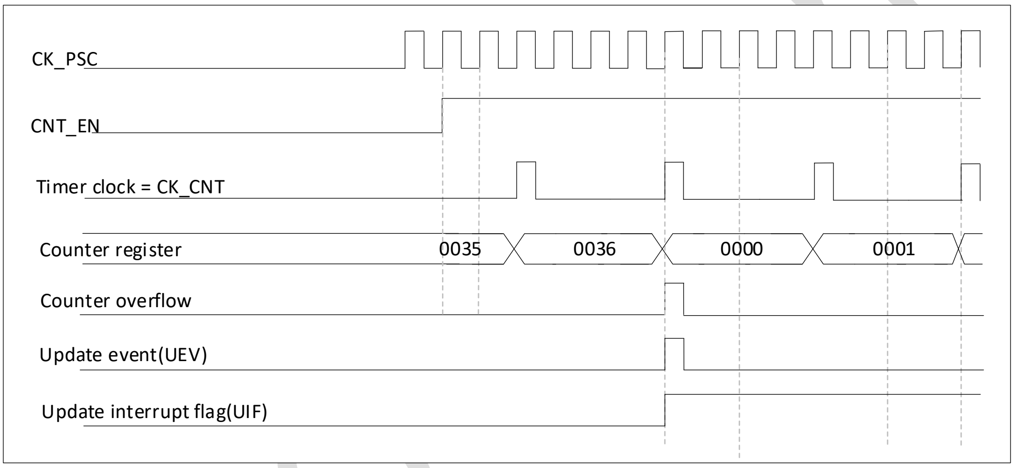


Figure 25-7 Counter timing diagram, internal clock divided by 4

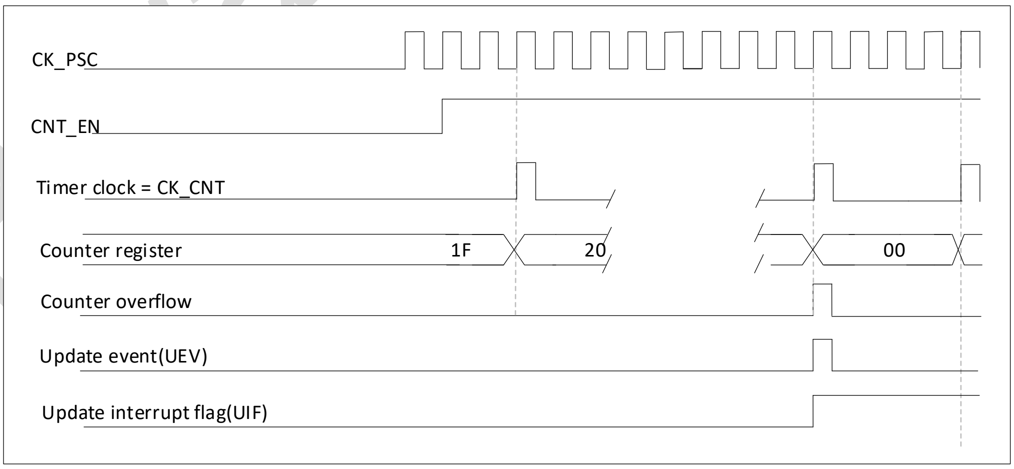


Figure 25-8 Counter timing diagram, internal clock divided by N

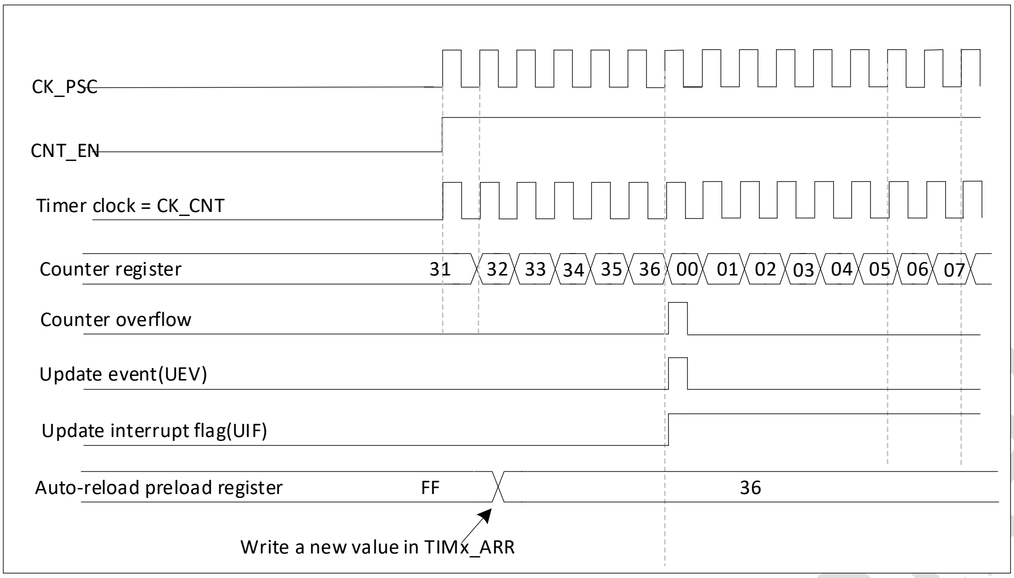


Figure 25-9 Counter timing diagram, update event when ARPE = 0 (TIMx_ARR not preloaded)

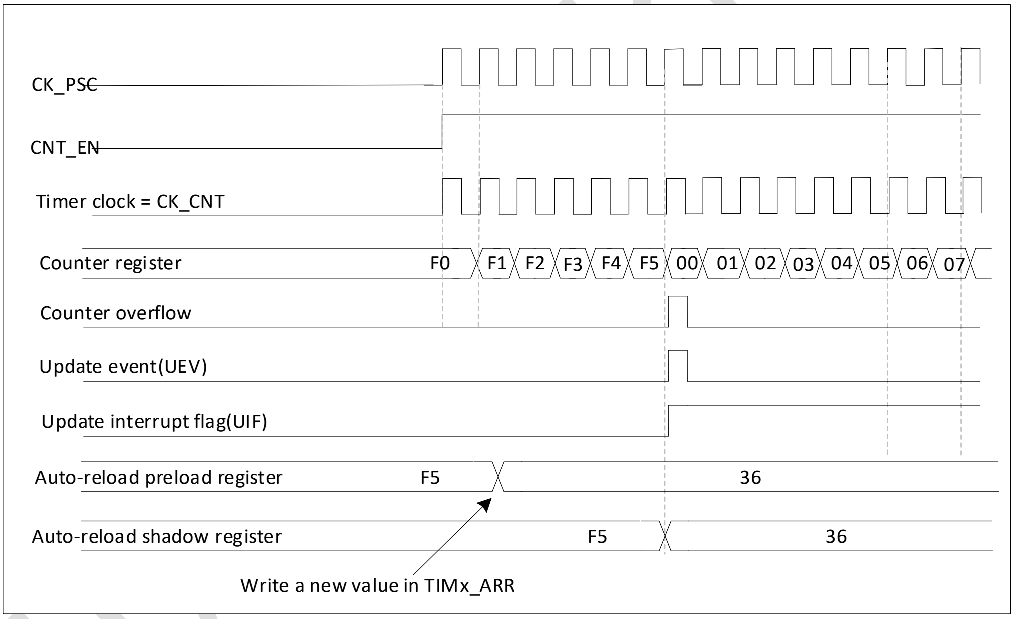


Figure 25-10 Counter timing diagram, update event when ARPE = 1 (TIMx_ARR preloaded)

Downward counting mode

Count down mode, starts counting down to 0 from the auto-loaded value, then restarts counting down from the auto-loaded value and generates a down overflow event.

If the repetition counter is used, the update event (UEV) is generated when the count down is repeated the number of times set in the repeat count register (TIMx_RCR), 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_CR1 register. 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 restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 shadow register is updated with the preload value (TIMx_ARR).

Note: Autoload is updated before the counter is reloaded, so the next cycle will be the expected value.

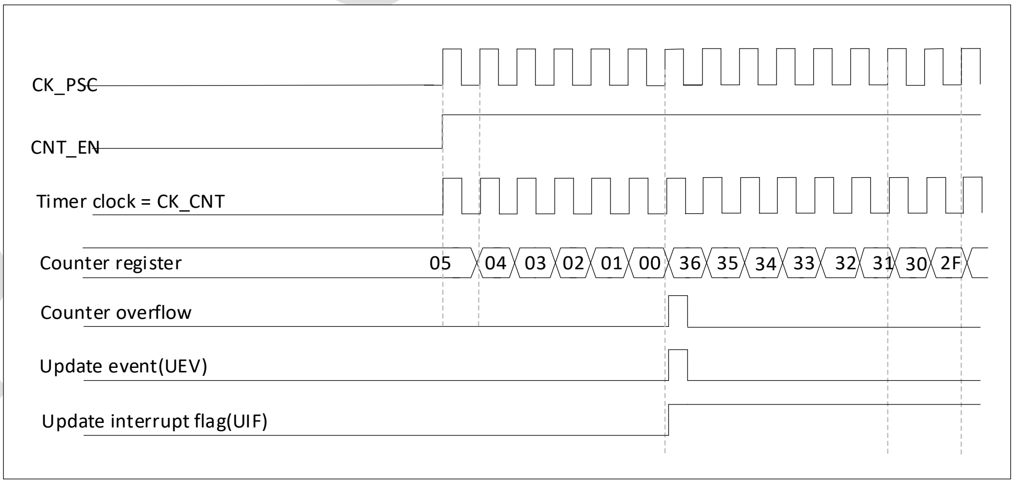


Figure 25-11 Counter timing diagram, internal clock divided by 1

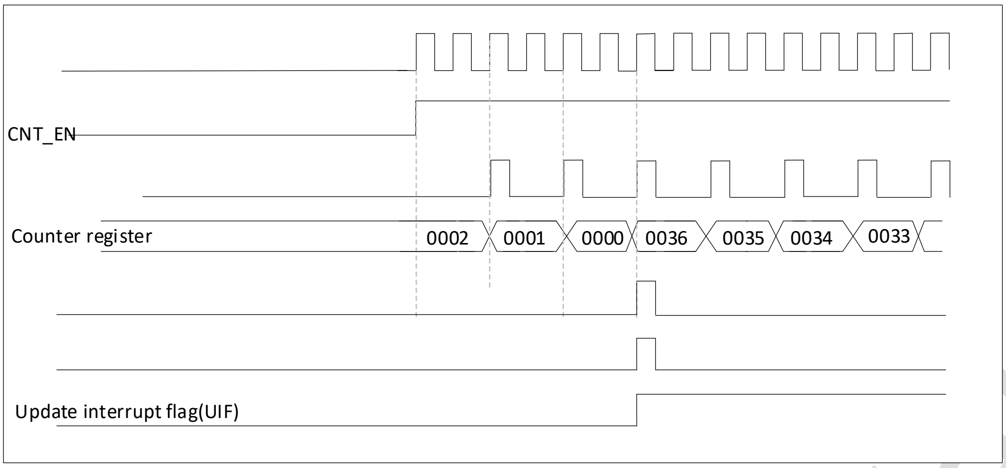


Figure 25-12 Counter timing diagram, internal clock divided by 2

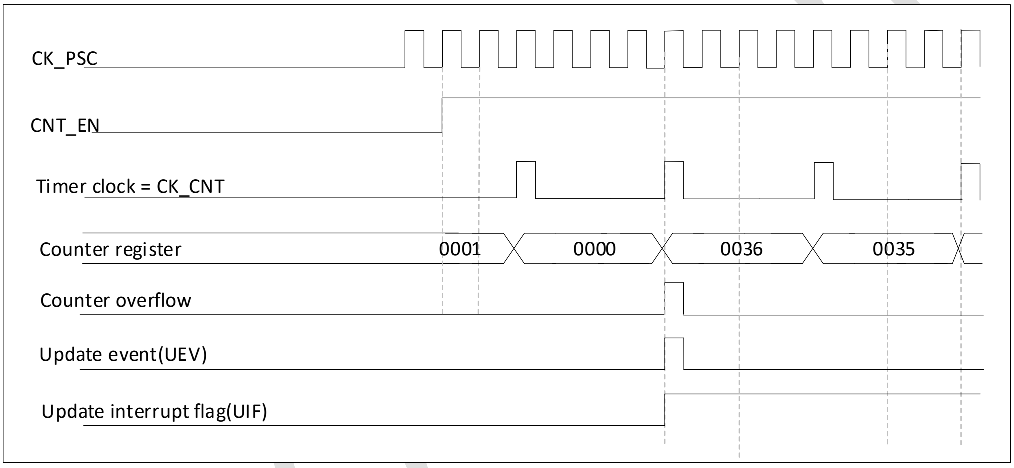


Figure 25-13 Counter timing diagram, internal clock divided by 4

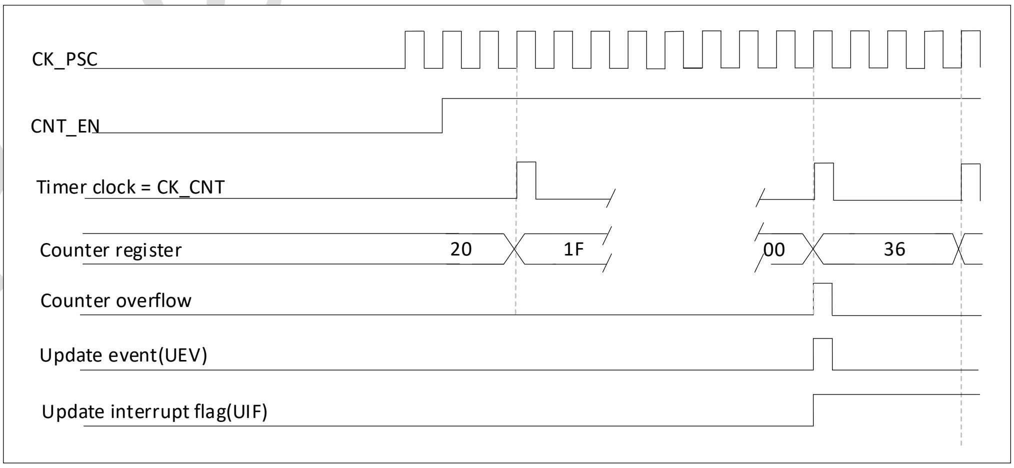


Figure 25-14 Counter timing diagram, internal clock divided by N

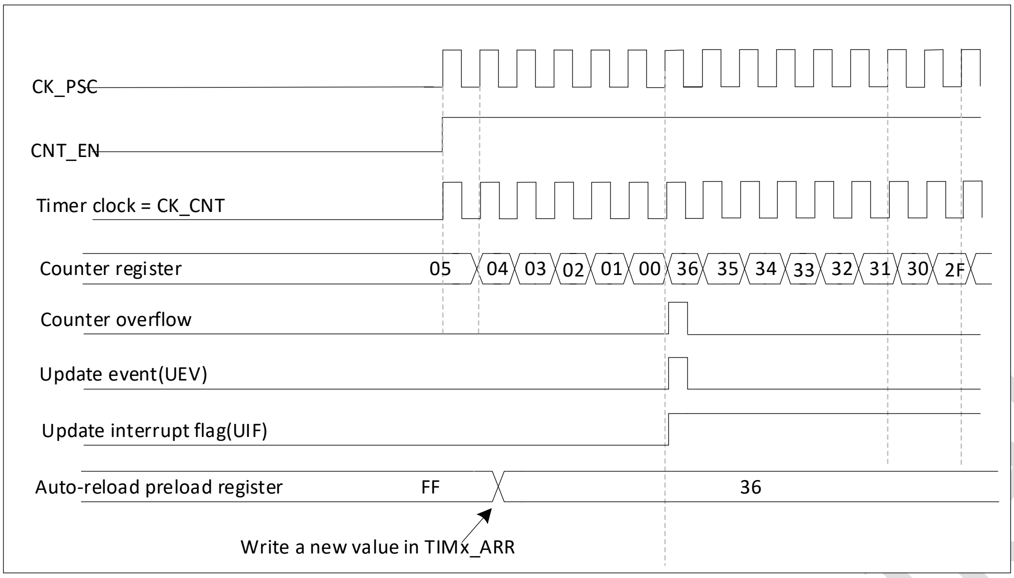


Figure 25-15 Counter timing diagram, update events when no cycle counter is used

Central alignment mode (counting up/down)

In central alignment mode, the counter counts from 0 to the auto-load value (TIMx_ARR register) -1, generating a counter overflow event, then counts down to 1 and generates a counter underflow event; it then counts again from 0.

The central alignment mode is valid when the CMS in the TIMx_CR1 register is not equal to 0. When the channel is configured in output mode, the output compare interrupt flag will be set when: counting down (central alignment mode 1, CMS="01"), counting up (central alignment mode 2, CMS="10") counting up and down (Central alignment mode 3, CMS="11").

In this mode, the DIR direction bit in TIMx_CR1 cannot be written. It is updated by hardware and indicates the current counting direction.

An update event can be generated on each count overflow and on each count underflow; it can also be generated by setting (in software or using the slave mode controller) the UG bit in the TIMx_EGR register. The counter then starts counting from 0 again and the prescaler also starts counting from 0 again.

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 the UDIS bit has been written to 0. However, the counter will still continue to count up or down, depending 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 update event UEV but without setting the UIF flag (thus no interrupt or DMA request is 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 shadow register is updated with the preload value (TIMx_ARR).

Note: If an update is generated 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 with the new value)

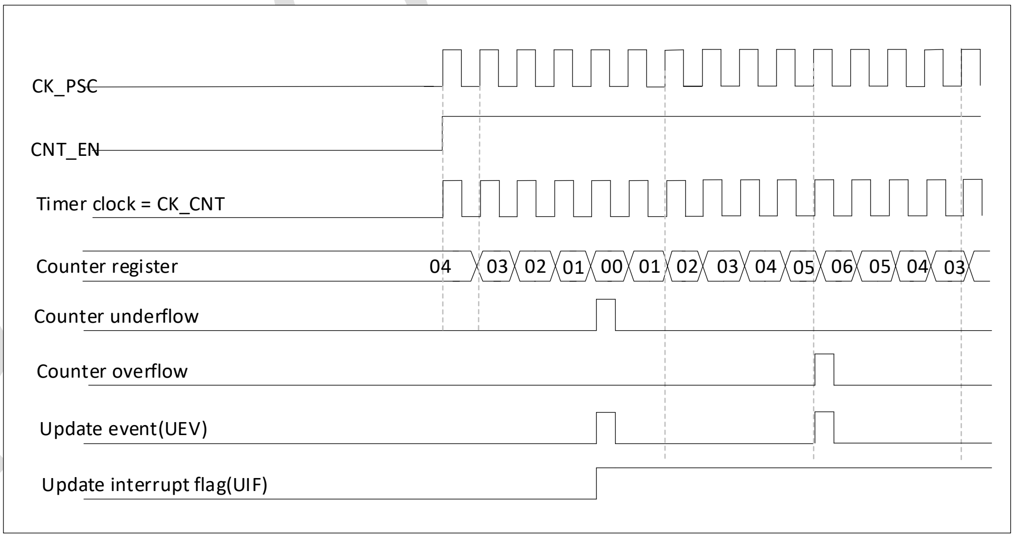


Figure 25-16 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

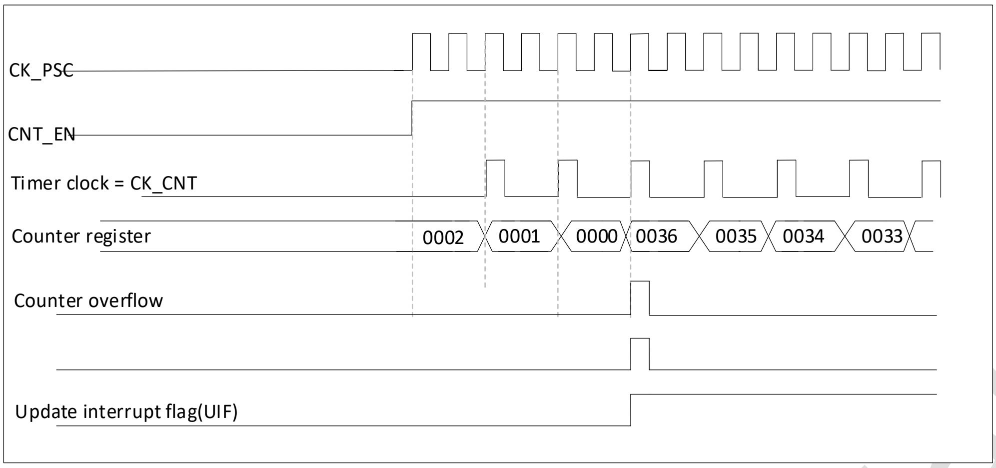


Figure 25-17 Counter timing diagram, internal clock divided by 2, TIMx_ARR=0x36

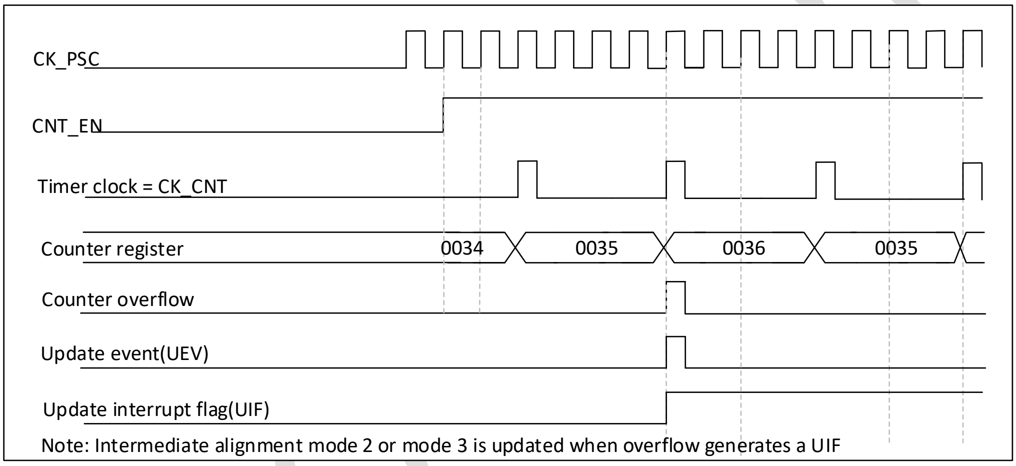


Figure 25-18 Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x36

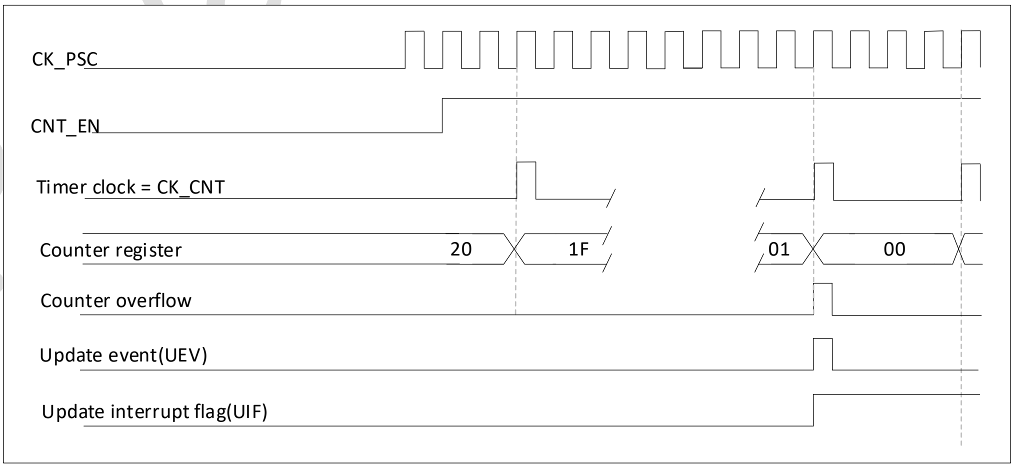


Figure 25-19 Counter timing diagram, internal clock divided by N

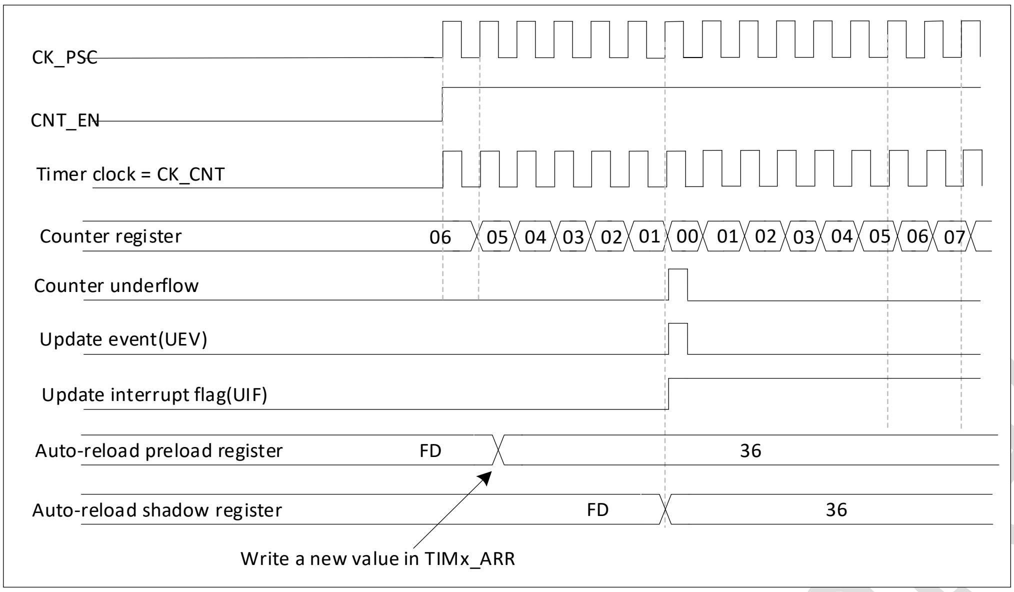


Figure 25-20 Counter timing diagram, update event when ARPE=1 (counter underflow)

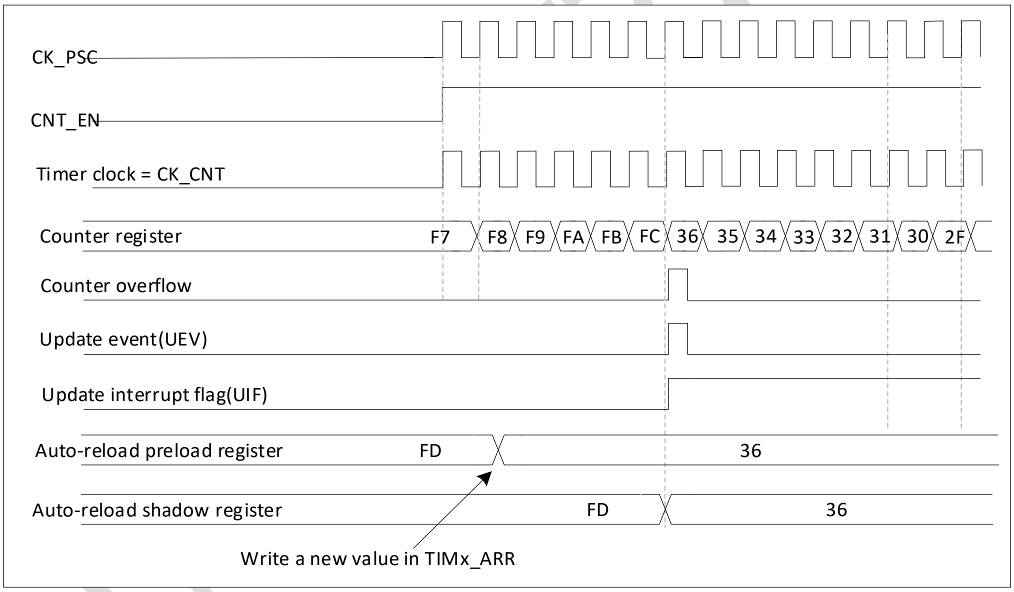


Figure 25-21 Counter timing diagram, update event when ARPE = 1 (Counter overflow)

25.4.3. Repeating down counter

Time-base unit describes how the update event (UEV) is generated withrespect to the counter overflows/underflows. It is actually generated only when the repetitioncounter 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 counter overflows or underflows, where N is the value in the TIMx_RCR repetition counter register.

The repeat counter is decremented at any of the following conditions.

- Each counter overflow in upcounting mode.
- Each counter underflow in downward counting mode.
- Each counter overflow and each underflow in central alignment mode

Although this limits the PWM to a maximum cycle period of 128 bits, it is able to update the duty cycle 2 times per PWM cycle. In central alignment mode, because the waveform is symmetrical, the maximum resolution is 2xTck if the comparison register is only refreshed once in each 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 central alignment mode, for odd values of the RCR, depending on when the RCR register is written and when the counter is started, an overflow or underflow occurs and an update event is generated. If the RCR is written before the counter is started, an update event is generated on an overflow. For example, for RCR = 3, the update event is generated on the 4th overflow or underflow event (depending on the value of the RCR being written).

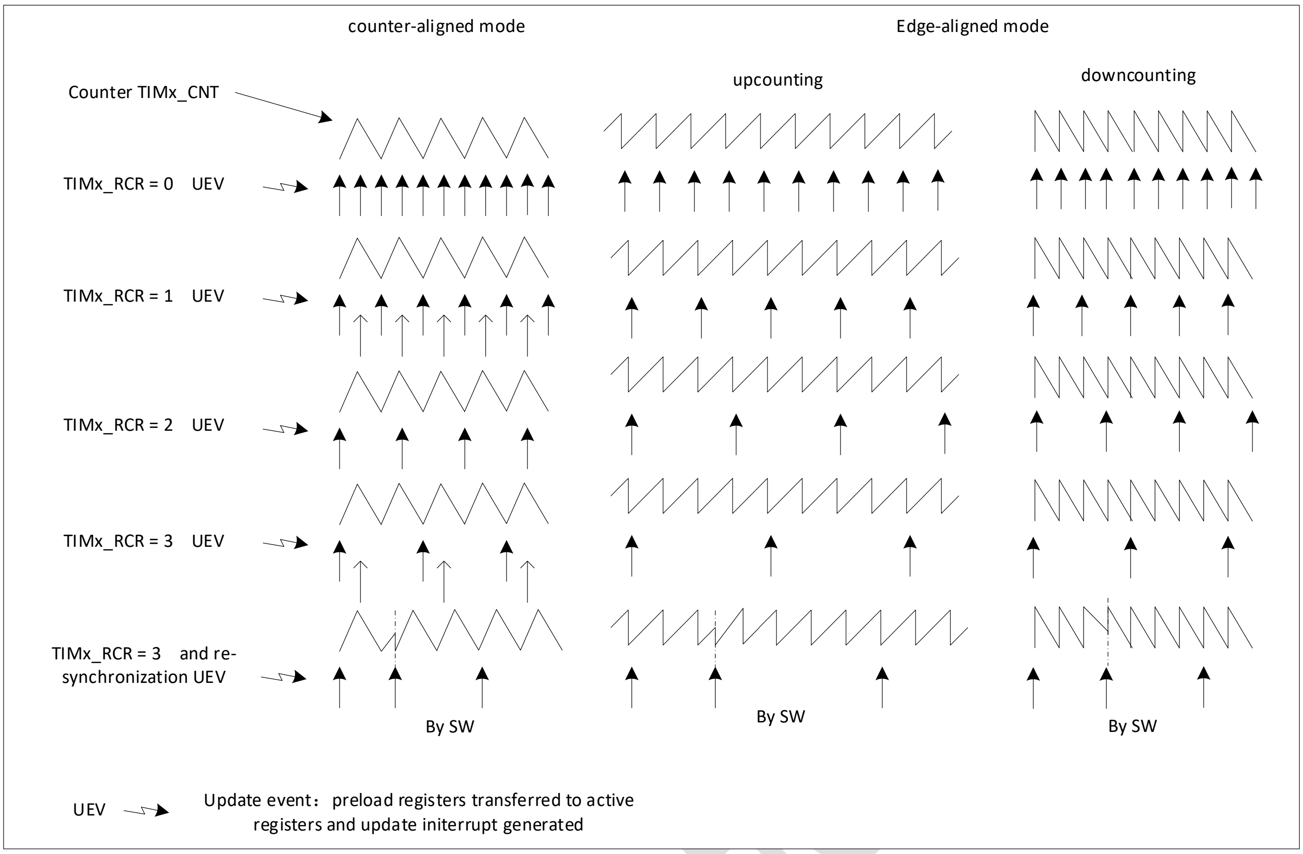


Figure 25-22 Update rate examples depending on mode and TIMx_RCR register settings

25.4.4. Clock sources

The counter clock can be provided:

- Internal clock (CK_INT).
- External clock mode 1: external input pins (TMI15 only)
- Internal trigger input (ITRx): uses one timer as a prescaler for another timer. For example, a timer Timer1 can be configured to act as a prescaler for another timer Timer2. (TMI15 only).

Internal clock source (CK_INT)

If the slave mode controller is disabled, the CEN and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

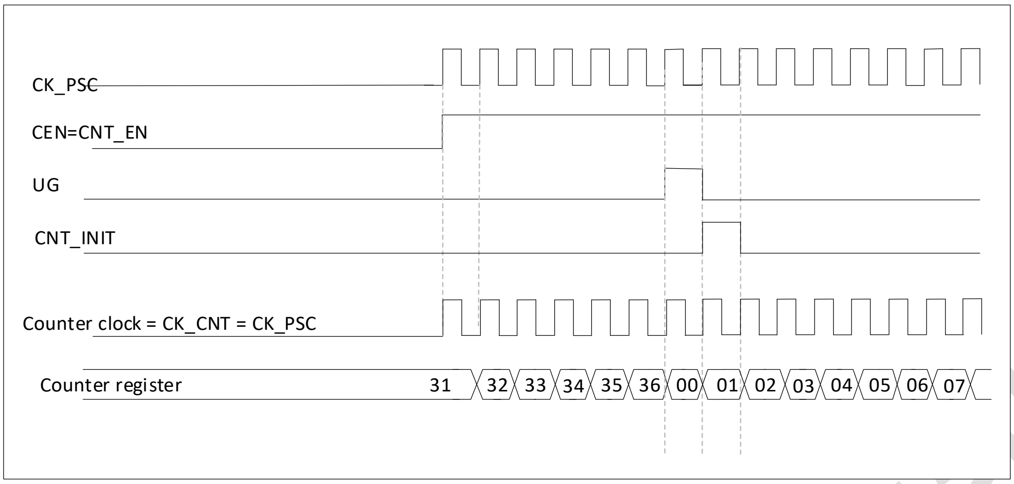


Figure 25-23 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1 (TIM15 only)

This mode is selected when SMS = 111 in the TIMx_SMCR register. The counter can count on each rising or falling edge of the selected input.

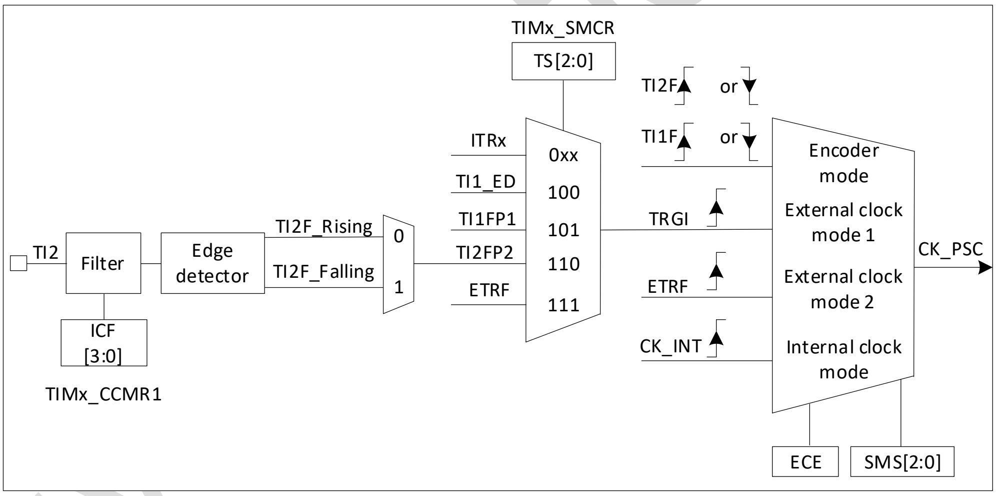


Figure 25-24 Example of an external clock connection

For example, to configure the counter to count on the rising edge of the T12 input upwards, use the following steps:

1. Configure the TIMx_CCMR1 register CC2S=01 so that channel 2 detects the rising edge of the TI2 input;
2. Configure the TIMx_CCMR1 register IC2F[3:0] to select the input filter bandwidth (if no filter is required, keep IC2F=0000);

3. Configure the TIMx_CCER register with CC2P=0 to select the rising edge polarity;

4. Configure the TIMx_SMCR register with SMS=111 to select the timer for external clock mode 1;

5. Configuring TS=110 in the TIMx_SMCR register to select TI2 as the trigger input source;

6. Set CEN=1 in the TIMx_CR1 register to start the counter.

Note: The capture prescaler is not used for triggering, so there is no need to configure it

When a rising edge occurs at TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge at TI2 and the actual counter clock depends on the resynchronisation circuit at the TI2 input.

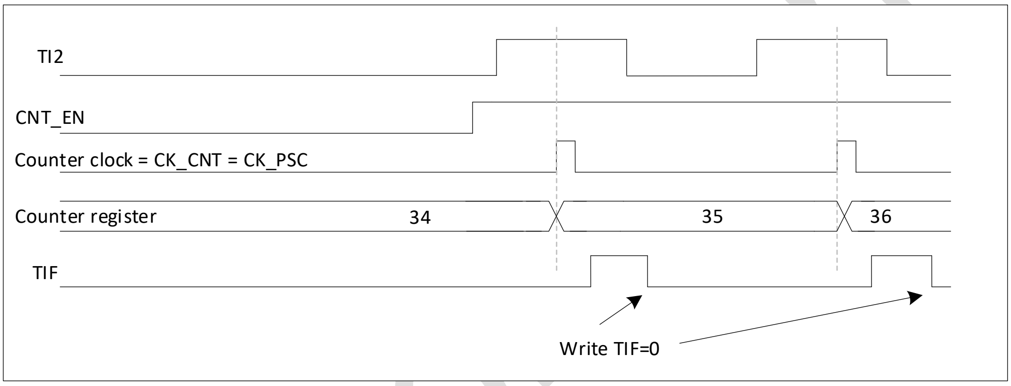


Figure 25-25 Control circuit in external clock mode 1

25.4.5. 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 following figures give an overview of one Capture/Compare channel.

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).

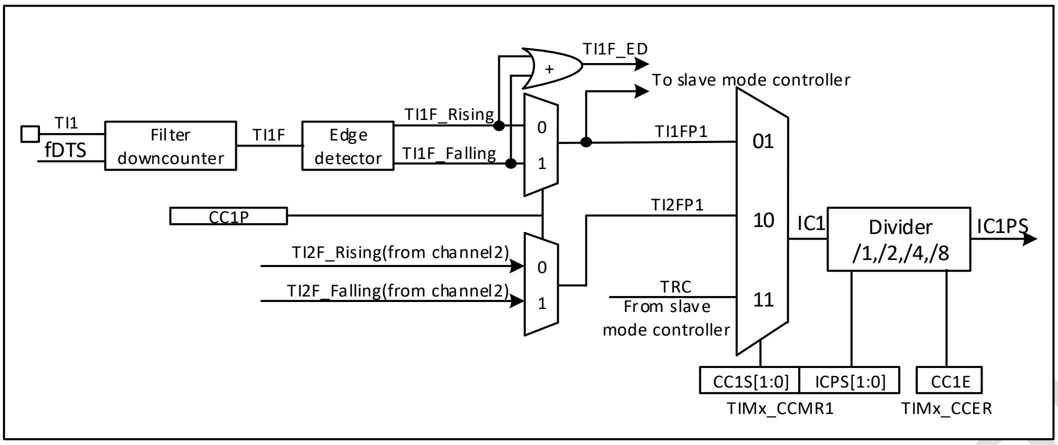


Figure 25-26 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

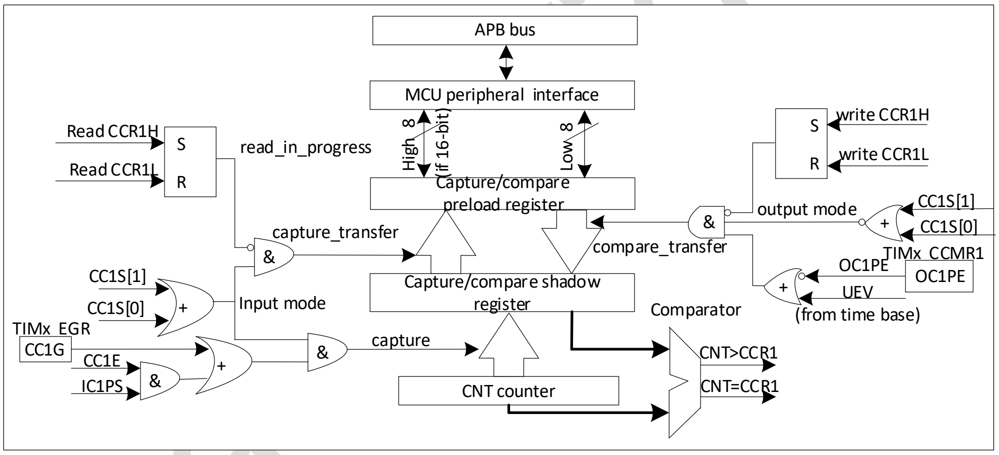


Figure 25-27 Capture/compare channel 1 main circuit

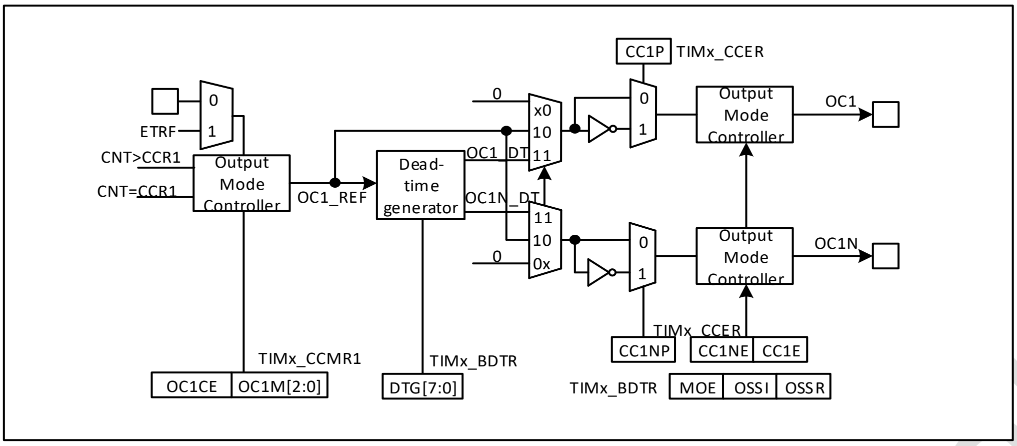


Figure 25-28 Output stage of capture/compare channel (channel 1 to 3)

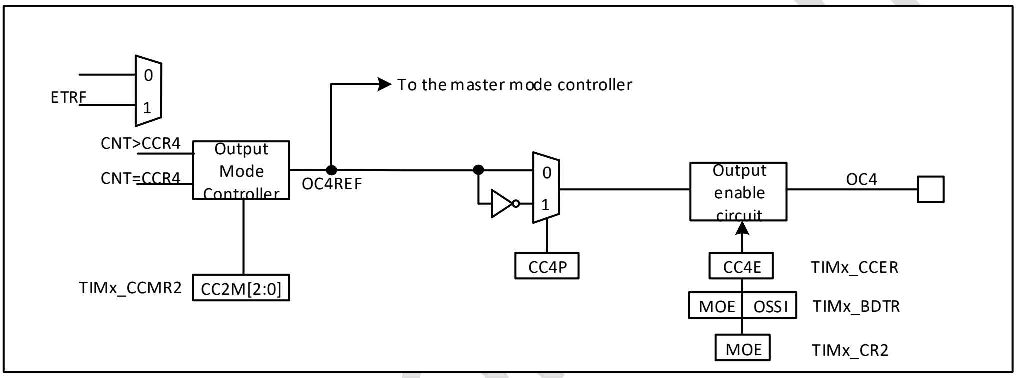


Figure 25-29 Output stage of capture/compare channel (channel 4)

The capture/compare block is made of one preload register and one shadow register. Write and read only 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.

25.4.6. Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set. 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 it is written with 0.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input 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 appropriate input filter duration in relation with the signal connected to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register). Let’s imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the 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 needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request 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.
- A DMA request is generated depending on the CC1DE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

25.4.7. Input capture mode（PWM input mode） （onlyTIM15）

This mode is a special case of input capture mode and operates the same as input capture mode except for the following differences:

- Both Icx signals are mapped to the same Tix input.
- The 2 Icx signals are edge valid but of opposite polarity.
- One of the TixFP signals is used as the trigger input signal and the slave mode controller is configured to reset mode. For example, when it is necessary to measure the length (TIMx_CCR1 register) and duty cycle (TIMx_CCR2 register) of the PWM signal input to TI1, proceed as follows (depending on the frequency of CK_INT and the value of the prescaler).
- Select a valid input for TIMx_CCR1: Set CC1S=01 in the TIMx_CCMR1 register (TI1 is selected).
- Selects valid polarity of TI1FP1 (used to capture data into TIMx_CCR1 and clear the counter): Set CC1P=0 (rising edge valid).
- Select a valid input for TIMx_CCR2: Set CC2S=10 in the TIMx_CCMR1 register (TI1 selected).
- Select valid polarity for TI1FP2 (capture data to TIMx_CCR2): set CC2P=1 (falling edge valid).
- Select a valid trigger input signal: Set TS=101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller to reset mode: Set SMS=100 in TIMx_SMCR.
- Enable Capture: Set CC1E=1 and CC2E=1 in the TIMx_CCER register.

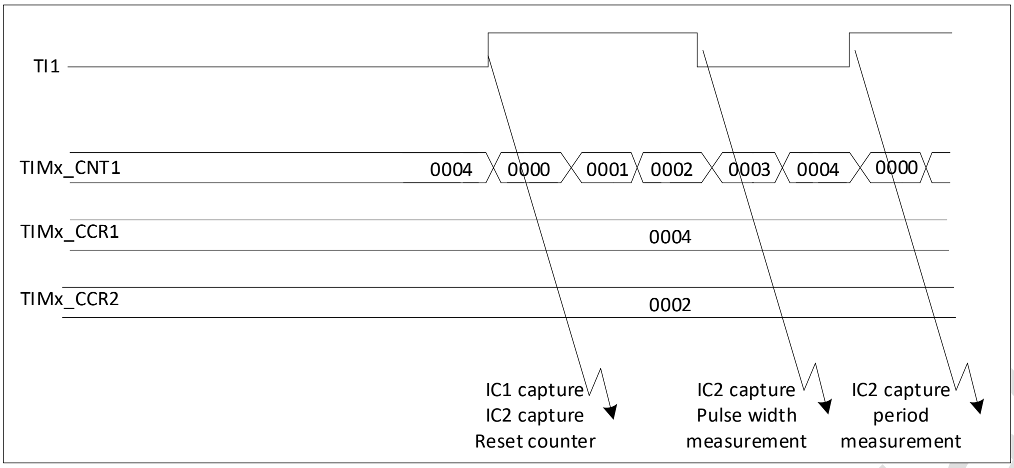


Figure 25-30 PWM Input Mode Timing

25.4.8. Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx) 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, one just needs to write 101 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.

25.4.9. 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).

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 following figure.

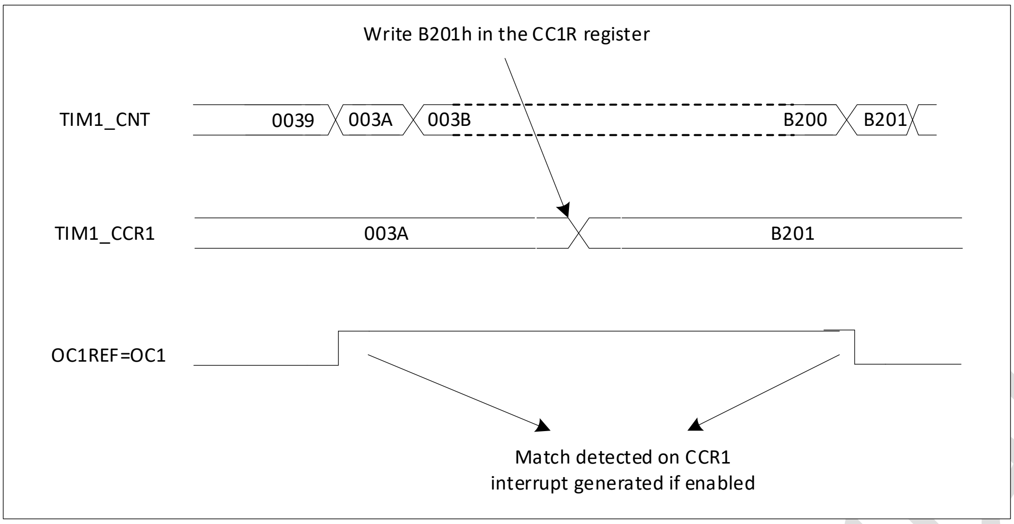


Figure 25-31 Output compare mode, toggle on OC1

25.4.10. PWM mode

Pulse Width Modulation mode allows 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 corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register 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 (in the TIMx_CCER and TIMx_BDTR registers). See the description of the TIMx_CCER register for details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared (depending on the counting direction of the counter) to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx.

The timer is able to generate PWM either in edge-aligned mode or in centrally-aligned mode, Depending on the status of the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

- **Upward Counting Configuration**

Perform up-count 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'. The following figure shows some edgealigned PWM waveforms in an example where TIMx_ARR = 8.

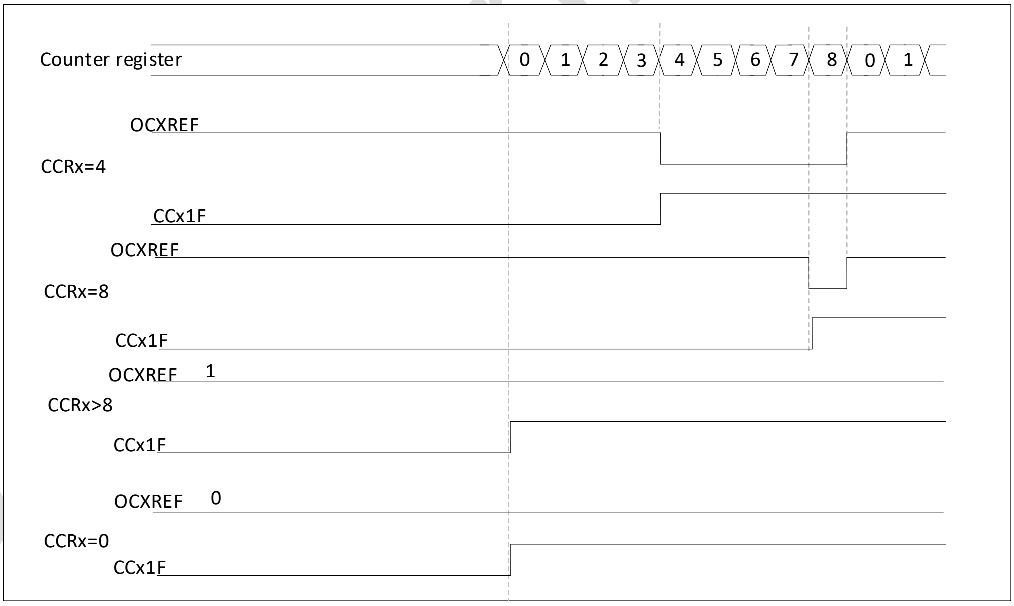


Figure 25-32 Edge-aligned PWM waveforms (ARR = 8)

- **Downward Counting Configuration**

Perform down counting when the DIR bit of the TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low when TIMx_CNT>TIMx_CCRx, otherwise it is high. If the comparison value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, OCxREF is held at '1'. A 0% PWM waveform cannot be generated in this mode.

PWM central alignment mode

Central alignment mode is when the CMS bit in the TIMx_CR1 register is not '00' (all other configurations have the same effect on the OCxREF/OCx signals). Depending on the CMS bit setting, the compare flag can be set to 1 when the counter is counting up, 1 when the counter is counting down, or 1 when the counter is counting up and down. the count direction bit (DIR) in the TIMx_CR1 register is updated by hardware, do not modify it by software.

The following diagram gives some examples of centrally aligned PWM waveforms

- TIMx_ARR = 8
- PWM mode 1
- CMS = 01 in the TIMx_CR1 register, which sets the compare flag when the counter counts down in central alignment mode.

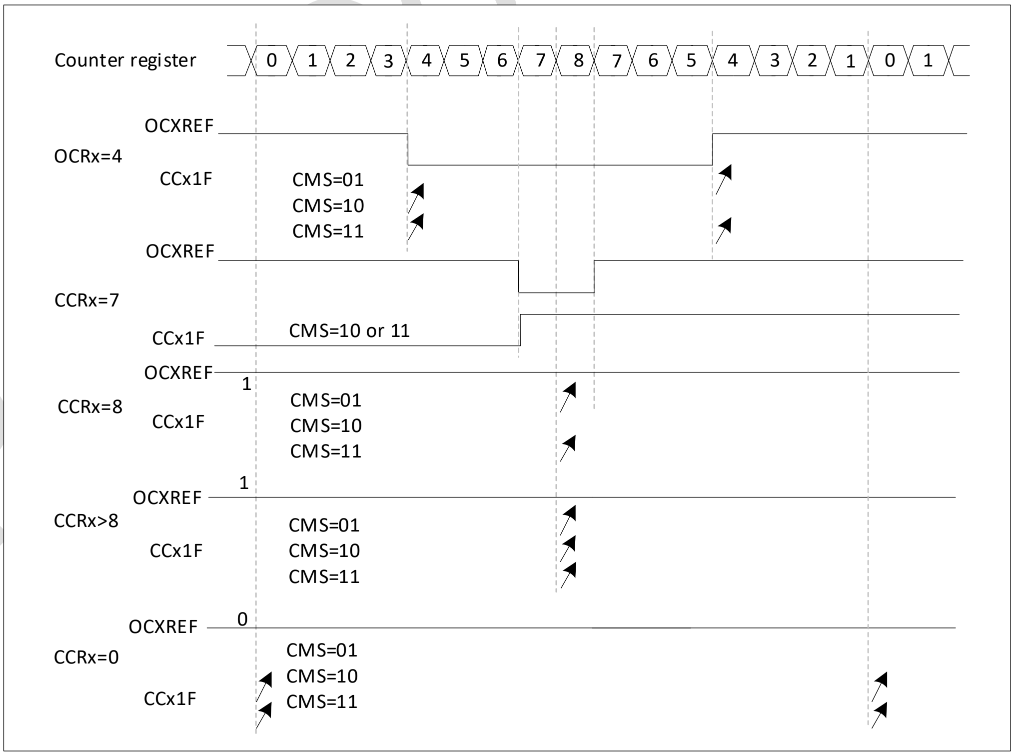


Figure 25-33 Centrally aligned PWM waveform (APR=8)

Hints for using the central alignment mode:

- When entering central alignment mode, the current up/down count configuration is used; this means that whether the counter counts up or down depends on the current value of the DIR bit in the TIMx_CR1 register. In addition, software cannot modify both the DIR and CMS bits.
- It is not recommended to rewrite the counter when running in central alignment mode, as this can have unpredictable results. In particular: - If the value written to the counter is greater than the value of the automatic reload (TIMx_CNT>TIMx_ARR), the direction will not be updated. For example, if the counter is counting up, it will continue to count up. ─ If a value of 0 or TIMx_ARR is written to the counter, the direction is updated, but no update event UEV is generated.
- The safest way to use the central alignment mode is to generate a software update (setting the UG bit in the TIMx_EGR bit) before starting the counter, and not to modify the counter value while counting is in progress.

25.4.11. Complementary outputs and dead-time insertion

The TIM16/17 general-purpose timers can output one complementary signal and manage the switching-off and switching-on of the outputs. This time is generally known as dead-time and it has to be adjusted depending on the devices that are connected to the outputs and their characteristics (intrinsic delays of levelshifters, delays due to power switches).

The polarity of the outputs (main output OCx or complementary OCxN) can be selected 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. Refer to Table xx Output control bits for complementary OCx and OCxN channels with break feature for more details. 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 DTG[7:0] 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).

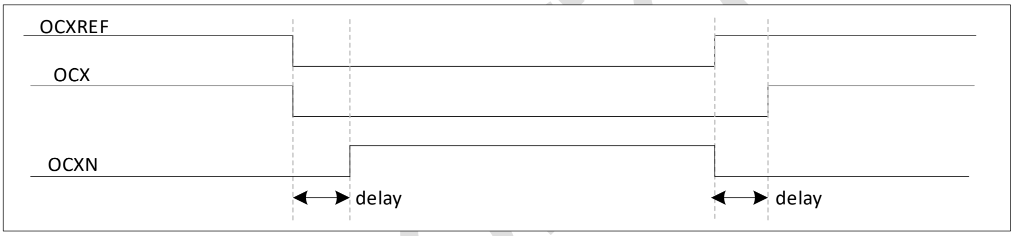


Figure 25-34 Complementary output with dead-time insertion

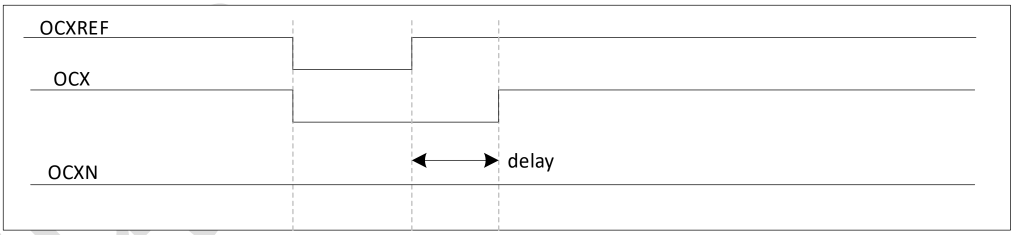


Figure 25-35 Dead-time waveforms with delay greater than the negative pulse

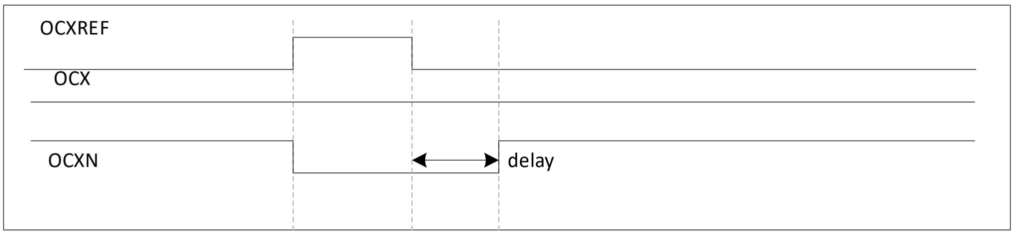


Figure 25-36 Dead-time waveforms with delay greater than the positive pulse

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 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 deadtime.

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.

25.4.12. Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits 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 source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- The core LOCKUP output
- The PVD output
- A clock failure event generated by the CSS detector

When exiting from reset, the break circuit is disabled and the MOE bit is low. The break function can be enabled 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 1 APB clock period to correctly read back the bit after the write operation.

Because MOE falling edge can be asynchronous, 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 MOE is set to 1 whereas it was low, a delay must be inserted (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 outputs in inactive state, idle state or in 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 because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
- 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. A DMA request can be generated if the BDE 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 again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written with 1 again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs is acting 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 cannot 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.

There are two ways to generate break:

- BKR input with programmable polarity while enable BKE in TIMx_BDTR register.
- Set the BG bit in TIMx_EGR 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 protection can be selected among 3 levels with the LOCK bits in the TIMx_BDTR register. Refer to TIM1 break and dead-time register (TIM1x_BDTR). The LOCK bits can be written only once after an MCU reset.

The following figure shows an example of behavior of the outputs in response to a break.

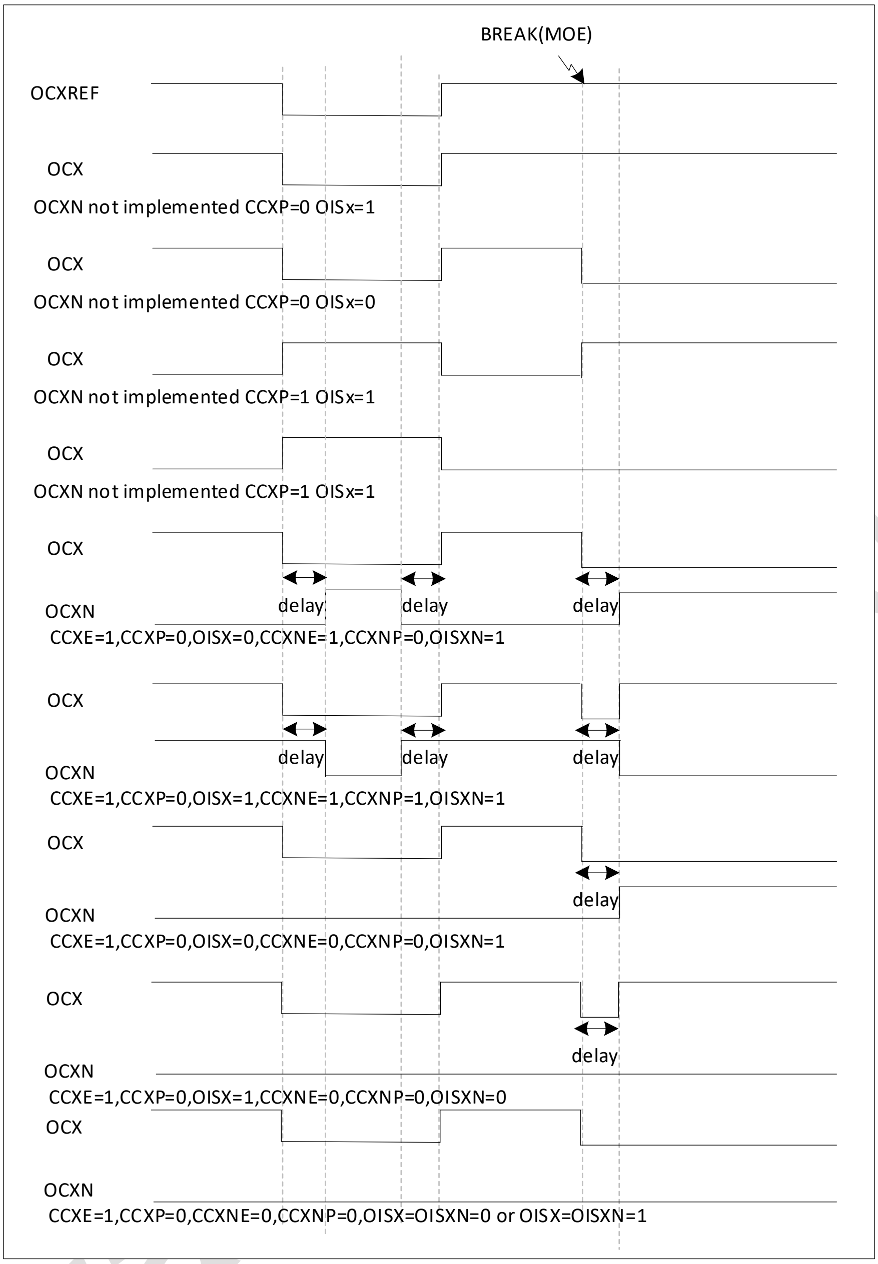


Figure 25-37 Output behavior in response to a break

25.4.13. 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

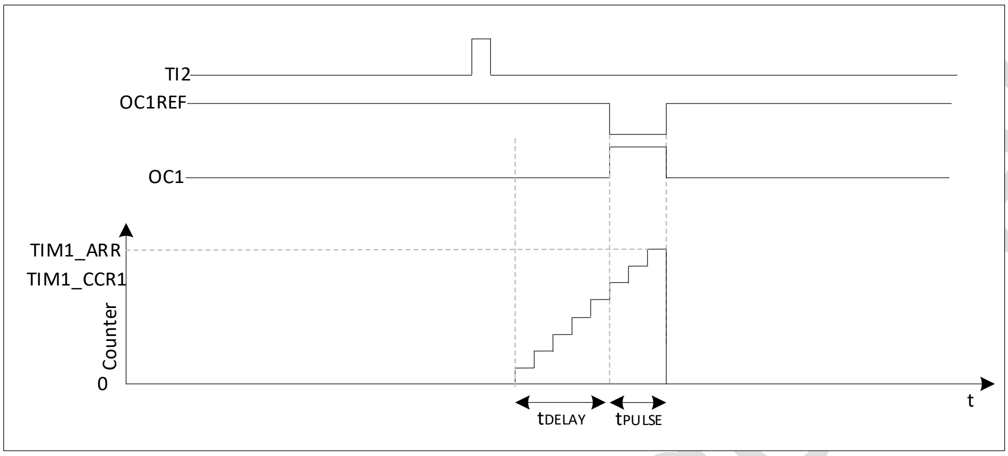


Figure 25-38 Example of One-pulse mode

For example, when it is necessary to generate a positive pulse of length tPULSE on OC1 after a delay of tDELAY starting from a rising edge detected on the TI2 input pin.

Use TI2FP2 as trigger 1.

- Set CC2S=01 in the TIMx_CCMR1 register to map TI2FP2 to TI2.
- Set CC2P=0 in the TIMx_CCER register to enable the TI2FP2 to detect rising edges.
- Set TS=110 in the TIMx_SMCR register to trigger the TI2FP2 as a slave mode controller (TRGI).
- sets SMS=110 in the TIMx_SMCR register (trigger mode) and the TI2FP2 is used to start the counter.

The OPM waveform is determined by the value written to the compare register (taking into account the clock frequency and the counter prescaler)

- tDELAY is defined by the value in the TIMx_CCR1 register.
- tPULSE is defined by the difference between the auto-load value and the compare value (TIMx_ARR - TIMx_CCR1).

- Assume 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 in the TIMx_CCMR1 register to enter PWM mode 2; selectively enable the preload registers as required: set OC1PE = in TIMx_CCMR1 1 in TIMx_CCMR1 and ARPE in TIMx_CR1 register; then fill in the comparison value in TIMx_CCR1 register and the auto-load value in TIMx_ARR register, set the UG bit to generate an update event, and wait for an external trigger event on TI2. In this example, CC1P = 0.

In this example, the DIR and CMS bits in the TIMx_CR1 register should be set low.

Since only one pulse is required, OPM = 1 in the TIMx_CR1 register must be set to stop counting on the next update event (when the counter flips from the auto-load value 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) are 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 PWM2 mode.

25.5. Synchronisation of TIMx timers and external triggers (TIM15 only)

The TIMx timer can be synchronised with an external trigger in several modes: reset mode, gated mode and trigger mode.

25.5.1. Slave mode: reset mode

When a trigger input event occurs, the counter and its prescaler can be reinitialised; at the same time, an update event UEV is also generated if the UDIS bit of the TIMx_CR1 register is low; all the preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the rising edge of the TI1 input causes the up counter to be cleared to zero:

- Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filter is needed, so keep IC1F = 0000). No capture prescaler is used in the trigger operation, so no configuration is required. the CC1S bit selects only the input capture source, i.e. CC1S=01 in the TIMx_CCMR1 register. set CC1P=0 in the TIMx_CCER register to determine polarity (rising edge detection only).
- Set SMS=100 in TIMx_SMCR register to configure the timer to reset mode; set TS=101 in TIMx_SMCR register to select TI1 as input source.
- Set CEN=1 in TIMx_CR1 register to start the counter.

The counter starts counting according to the internal clock and then operates normally until a rising edge appears on TI1; at this point, the counter is cleared and starts counting from 0 again. At the same time, the trigger flag (TIF bit in the TIMx_SR register) is set, generating an interrupt request or a DMA request depending on the setting of the TIE (interrupt enable) and TDE (DMA enable) bits in the TIMx_DIER register.

The diagram below shows the action when the Auto Reload register TIMx_ARR=0x36. The delay between the rising edge of TI1 and the actual reset of the counter depends on the resynchronisation circuitry at the TI1 input.

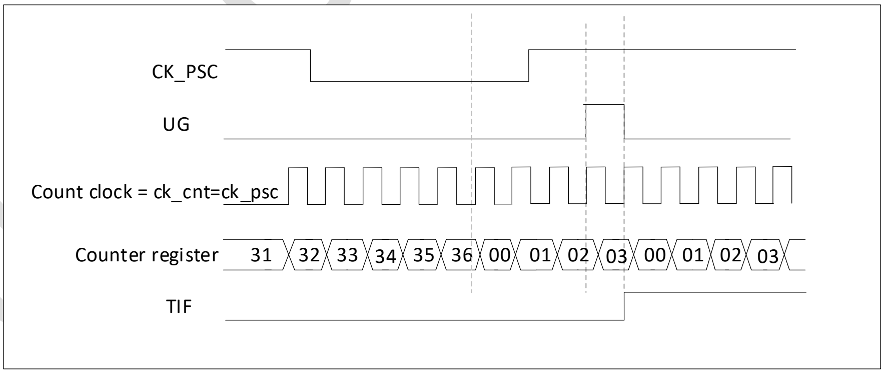


Figure 25-39 Control circuit in reset mode

25.5.2. Slave mode: Gated mode

The counter is enabled according to the level of the selected input.

In the following example, the counter only counts up when TI1 is low:

- Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (in this example, no filtering is required, so keep IC1F = 0000). No capture prescaler is used in the trigger operation, so no configuration is required. The CC1S bit is used to select the input capture source by setting CC1S=01 in the TIMx_CCMR1 register. set CC1P=1 in the TIMx_CCER register to determine the polarity (low level only is detected).
- Set SMS=101 in the TIMx_SMCR register to configure the timer in gated mode; set TS=101 in the TIMx_SMCR register to select TI1 as the input source.
- Set CEN=1 in TIMx_CR1 register to start the counter. In gated mode, if CEN=0, the counter cannot be started, regardless of the trigger input level.

The counter starts counting according to the internal clock as long as TI1 is low, and stops counting once TI1 goes high. The TIF flag in TIMx_SR is set when the counter is started or stopped.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronisation circuitry at the TI1 input.

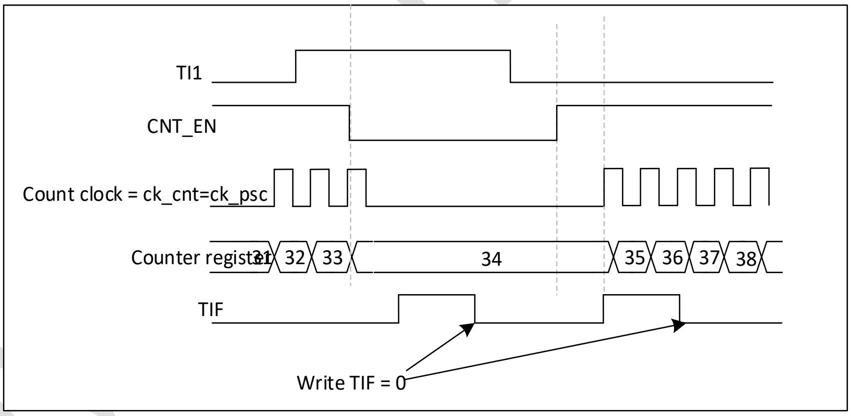


Figure 25-40 Control circuit in gated mode

25.5.3. Slave mode: trigger mode

The event selected on the input enables the counter.

In the following example, the counter starts counting up at the rising edge of the TI2 input:

- Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is needed and IC2F = 0000 is held). No capture prescaler is used in the trigger operation and no configuration is required. the CC2S bit is only used to select the input capture source, set CC2S=01 in the TIMx_CCMR1 register. set CC2P=1 in the TIMx_CCER register to determine the polarity (only low levels are detected).

- Set SMS=110 in TIMx_SMCR register to configure the timer to trigger mode; set TS=110 in TIMx_SMCR register to select TI2 as input source.

When there is a rising edge of TI2, the counter starts counting driven by the internal clock and the TIF flag is set at the same time.

The delay between the rising edge of TI2 and the counter starting to count depends on the resynchronisation circuit at the TI2 input.

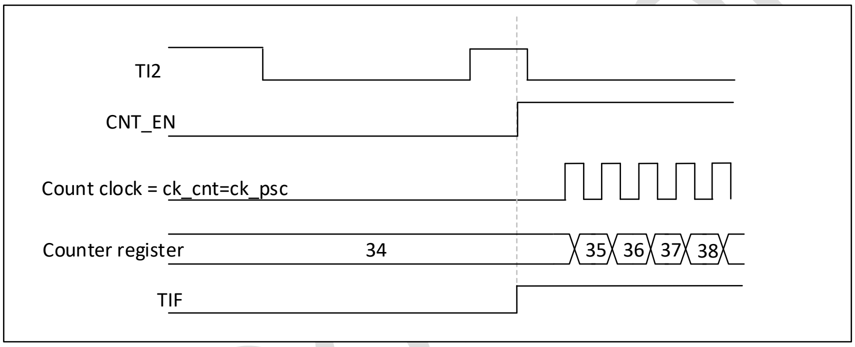


Figure 25-41 Control circuit in flip-flop mode

25.5.4. Slave mode: External clock mode 2 + Trigger mode

External clock mode 2 can be used in conjunction with another slave mode (except external clock mode 1 and encoder mode). In this case the ETR signal is used as an input to the external clock and another input can be selected as a trigger input in reset mode, gated mode or trigger mode. It is not recommended to use the TS bit of the TIMx_SMCR register to select ETR as TRGI.

In the following example, the counter counts up once on each rising edge of ETR once a rising edge occurs on TI1:

1. To configure the external trigger input circuit via the TIMx_SMCR register:

- ETF=0000: no filtering
- ETPS=00: no prescaler
- ETP=0: detect the rising edge of ETR, set ECE=1 to enable external clock mode 2. 2.

2. Configure channel 1 as follows, detecting the rising edge of TI:

- IC1F=0000: no filtering
- No capture prescaler is used in the trigger operation, no configuration is needed
- Set CC1S=01 in the TIMx_CCMR1 register to select the input capture source
- Set CC1P=0 in the TIMx_CCER register to determine polarity (rising edge detection only)

3. Set SMS=110 in TIMx_SMCR register to configure the timer to trigger mode. Set TS=101 in the TIMx_SMCR register to select TI1 as the input source.

When a rising edge occurs on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR.

The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronisation circuit at the ETRP input.

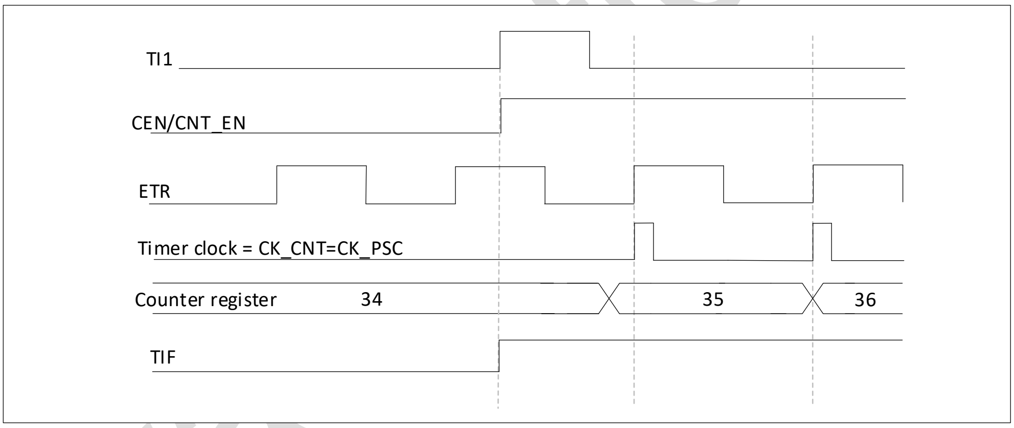


Figure 25-42 Control circuit in external clock mode 2 + trigger mode

25.5.5. TIM and external trigger synchronisation

The TIMx timer can be synchronised with an external trigger in several modes: reset mode, gated mode and trigger mode.

Slave mode: reset mode

When a trigger input event occurs, the counter and its prescaler can be reinitialised; at the same time, an update event UEV is also generated if the URS bit of the TIMx_CR1 register is low; all the preload registers (TIMx_ARR, TIMx_CCRx) are then updated.

In the following example, the rising edge of the TI1 input causes the up counter to be cleared to zero:

- Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filter is needed, so keep IC1F = 0000). No capture prescaler is used in the trigger operation, so no configuration is required. the CC1S bit selects only the input capture source, i.e. CC1S=01 in the TIMx_CCMR1 register. set CC1P=0 in the TIMx_CCER register to determine the polarity (only rising edges are detected).
- Set SMS=100 in TIMx_SMCR register to configure the timer to reset mode; set TS=101 in TIMx_SMCR register to select TI1 as input source.
- Set CEN=1 in TIMx_CR1 register to start the counter.

The counter starts counting according to the internal clock and then runs normally until a rising edge appears on TI1; at this point, the counter is cleared and starts counting again from 0. At the same time, the trigger flag (TIF bit in the TIMx_SR register) is set, generating an interrupt request or a DMA request depending on the setting of the TIE (interrupt enable) and TDE (DMA enable) bits in the TIMx_DIER register.

The diagram below shows the action when the Auto Reload register TIMx_ARR=0x36. The delay between the rising edge of TI1 and the actual reset of the counter depends on the resynchronisation circuitry at the TI1 input.

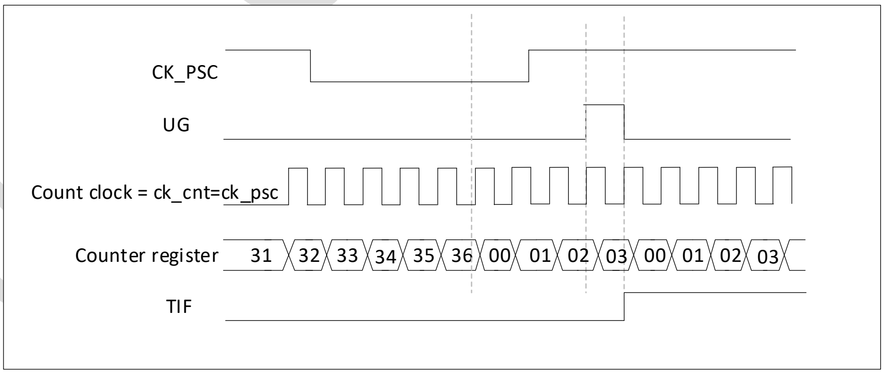


Figure 25-43 Control circuit in reset mode

Slave mode: Gated mode

The counter is enabled according to the level of the selected input.

In the following example, the counter only counts up when TI1 is low:

- Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (in this example, no filtering is required, so keep IC1F = 0000). No capture prescaler is used in the trigger operation, so no configuration is required. the CC1S bit is used to select the input capture source, set CC1S=01 in the TIMx_CCMR1 register. set CC1P=1 in the TIMx_CCER register to determine the polarity (low level only is detected).
- Set SMS=101 in TIMx_SMCR register to configure the timer for gated mode; set TS=101 in TIMx_SMCR register to select TI1 as input source.
- Set CEN=1 in TIMx_CR1 register to start the counter. In gated mode, if CEN=0, the counter cannot be started, regardless of the trigger input level.

The counter starts counting according to the internal clock as long as TI1 is low, and stops counting once TI1 goes high. The TIF flag in TIMx_SR is set when the counter is started or stopped.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronisation circuitry at the TI1 input.

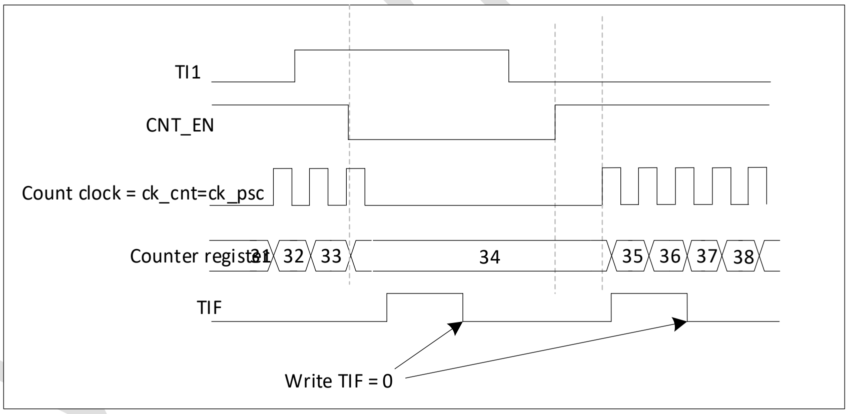


Figure 25-44 Control circuit in gated mode

The event selected on the input enables the counter.

In the following example, the counter starts counting up at the rising edge of the TI2 input:

1. Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is needed and IC2F = 0000 is held). No capture prescaler is used in the trigger operation and no configuration is required. the CC2S bit is only used to select the input capture source,

set CC2S=01 in the TIMx_CCMR1 register. set CC2P=1 in the TIMx_CCER register to determine the polarity (only low levels are detected).

2. Set SMS=110 in TIMx_SMCR register to configure the timer to trigger mode; set TS=110 in TIMx_SMCR register to select TI2 as input source.

When there is a rising edge of TI2, the counter starts counting driven by the internal clock and the TIF flag is set at the same time. The delay between the rising edge of TI2 and the counter starting to count depends on the resynchronisation circuit at the TI2 input.

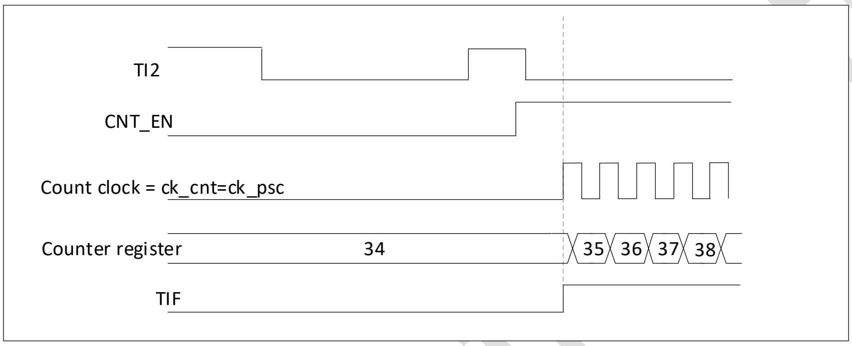


Figure 25-45 Control circuit in gated mode

Slave mode: External clock mode 2 + Trigger mode

External clock mode 2 can be used in conjunction with another slave mode (except external clock mode 1 and encoder mode). In this case the ETR signal is used as an input to the external clock and another input can be selected as a trigger input in reset mode, gated mode or trigger mode. It is not recommended to use the TS bit of the TIMx_SMCR register to select ETR as TRGI.

In the following example, the counter counts up once on each rising edge of ETR once a rising edge occurs on TI1:

- Configuring the external trigger input circuit via the TIMx_SMCR register:
  - ETF=0000: no filtering
  - ETPS=00: no prescaler
  - ETP=0: detect the rising edge of ETR, set ECE=1 to enable external clock mode 2.
- Configure channel 1 as follows, detecting the rising edge of TI:
  - IC1F=0000: no filtering

- No capture prescaler is used in the trigger operation, no configuration is needed
- Set CC1S=01 in the TIMx_CCMR1 register to select the input capture source
- Set CC1P=0 in the TIMx_CCER register to determine the polarity (rising edge detection only)

- Set SMS=110 in TIMx_SMCR register to configure the timer to trigger mode. Set TS=101 in the TIMx_SMCR register to select TI1 as the input source.

When a rising edge appears on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR. The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronisation circuitry at the ETRP input.

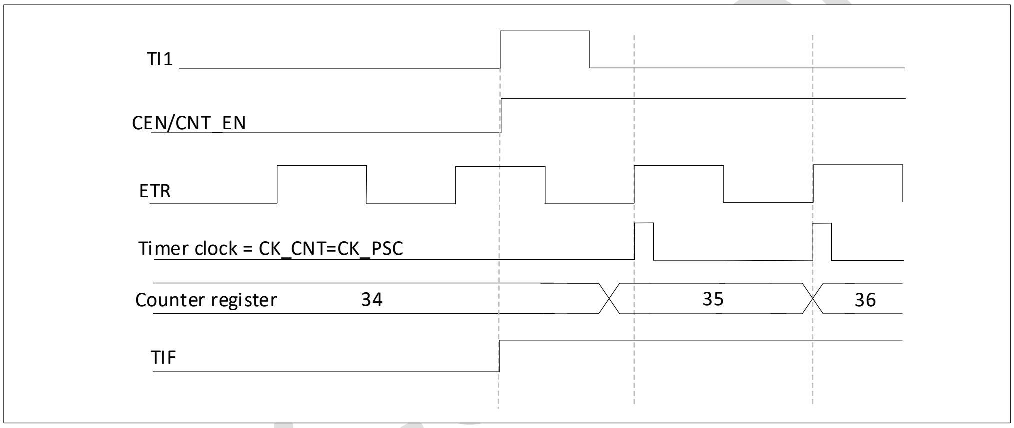


Figure 25-46 External clock mode 2 + control circuit in trigger mode

25.6. Timer synchronisation (only TIM15)

All TIMx timers are linked internally for timer synchronisation or linking. When a timer is in master mode, it can reset, start, stop or provide a clock to the counter of another timer in slave mode.

The diagram below shows an overview of the trigger selection and master mode selection modules.

25.6.1. Using a timer as a prescaler for another timer

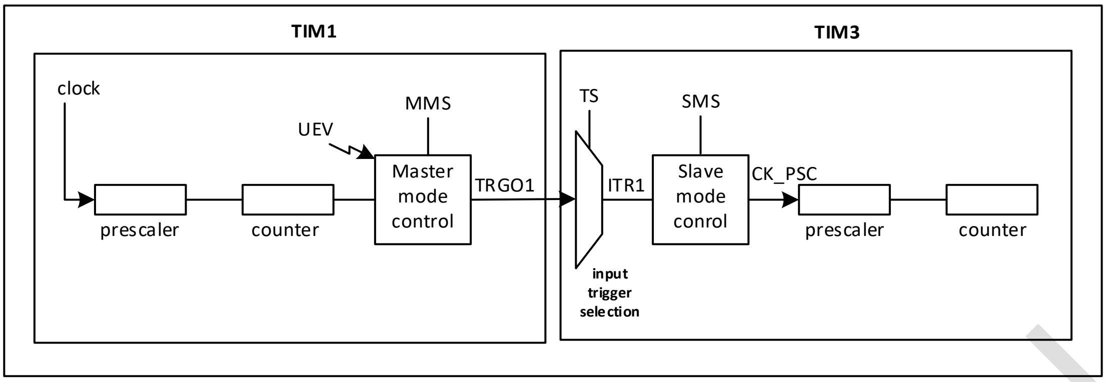


Figure 25-47 Example of a master/slave timer

E.g. Timer 1 can be configured to act as a prescaler for Timer 2. Referring to Figure 4-48, perform the following operations:

- Configure Timer 1 to be the master mode, which can output a periodic trigger signal at each update event UEV. With MMS='010' in the TIM15_16_17_CR2 register, a rising edge signal is output on TRGO1 whenever an update event is generated.
- Connect the TRGO1 output of Timer 1 to Timer 2, set TS='000' of the TIM2_SMCR register and configure Timer 2 as a slave mode using ITR1 as the internal trigger.
- The slave mode controller is then placed in external clock mode 1 (SMS=111 in the TIM2_SMCR register); this allows timer 2 to be driven by the periodic rising edge (i.e. timer 1's counter overflow) signal of timer 1.
- Finally, the CEN bit of the corresponding (TIMx_CR1 register) must be set to start each of the two timers.

Note: If OCx has been selected as the trigger output of timer 1 (MMS=1xx), its rising edge is used to drive the counter of timer 2.

25.6.2. Using one timer to enable another timer

In this example, the enable of Timer 2 is controlled by the comparison of the output of Timer 1. Refer to Figure 4-48 for the connection. Timer 2 counts the divided internal clock only when the OC1REF of Timer 1 is high. The clock frequency of both timers is obtained by dividing CK_INT by 3 (fCK_CNT = fCK_INT/3) from the prescaler.

- Configure Timer 1 as the master mode and send its output comparison reference signal (OC1REF) as the trigger output (MMS=100 of TIM15_16_17_CR2 register)
- Configure the OC1REF waveform of Timer 1 (TIM15_16_17_CCMR1 register)
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 of TIM2_SMCR register)
- Configure Timer 2 to be in gated mode (SMS=101 in TIM2_SMCR register)
- Set CEN=1 of TIM2_CR1 register to enable timer 2
- Set CEN=1 of TIM15_16_17_CR1 register to enable Timer 1

Note: Timer 2's clock is not synchronised with Timer 1's clock, this mode only affects the Timer 2 counter enable signal.

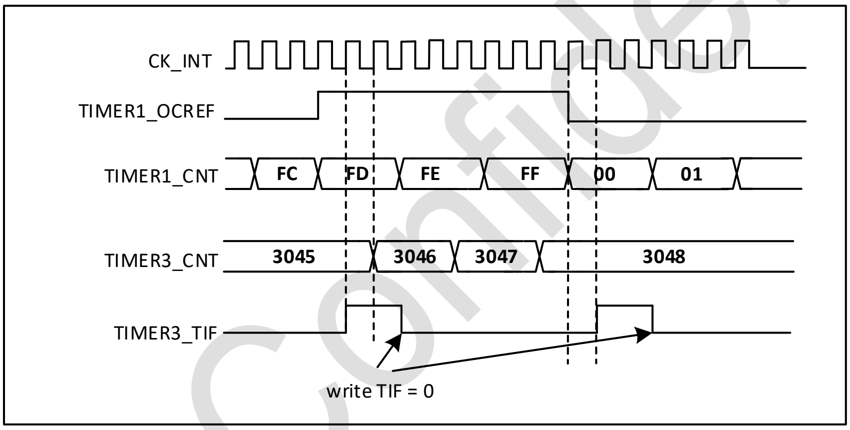


Figure 25-48 OC1REF control timer 2 of timer 1

In the example in Figure 4-47, before Timer 2 is started, their counters and prescaler are not initialised, so they start counting from the current value. It is possible to reset the 2 timers before starting Timer 1 so that they start from a given value, i.e. write any value required to the timer counter. The timers can be reset by writing the UG bit of the TIMx_EGR register.

In the next example it is necessary to synchronise Timer 1 and Timer 2. Timer 1 is the main mode and starts from 0, Timer 2 is the slave mode and starts from 0xE7; both timers have the same prescaler factor. Writing '0' to the CEN bit of TIM15_16_17_CR1 will disable timer 1 and timer 2 will then stop.

- Configure Timer 1 to be the master mode and send the output compare 1 reference signal (OC1REF) as the trigger output (MMS=100 in TIM15_16_17_CR2 register).
- Configure the OC1REF waveform for Timer 1 (TIM15_16_17_CCMR1 register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 of TIM2_SMCR register)
- Configure Timer 2 to be in gated mode (SMS=101 of TIM2_SMCR register)
- Set UG='1' of TIM15_16_17_EGR register to reset timer 1.
- Set UG='1' of TIM2_EGR register to reset timer 2.
- Write '0XE7' to Timer 2's counter (TIM2_CNT), initialize it to 0xE7.
- Set CEN='1' of TIM2_CR1 register to enable timer 2.
- Set CEN='1' of TIM15_16_17_CR1 register to enable timer 1.
- Set CEN='0' of TIM15_16_17_CR1 register to stop timer 1

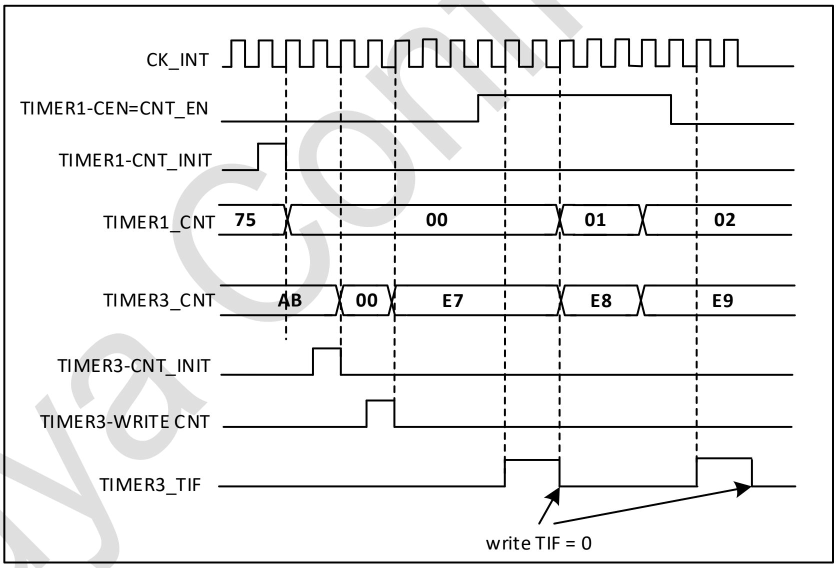


Figure 25-49 Timer 2 can be controlled by enabling Timer 1

25.6.3. Using a timer to start another timer

In this example, Timer 2 is enabled using an update event from Timer 1.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. On receipt of a trigger signal, the CEN bit of Timer 2 is automatically set to

'1' and the counter starts counting until a '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 to CK_INT (fCK_CNT = fCK_INT/3).

- Configure timer 1 to be the master mode and send its update event (UEV) as a trigger output (MMS=010 in TIM15_16_17_CR2 register).
- Configure the period of timer 1 (TIM15_16_17_ARR register).
- Configure timer 2 to get input trigger from timer 1 (TS=000 of TIM2_SMCR register)
- Configure Timer 2 to be in trigger mode (SMS=110 in TIM2_SMCR register)
- Set CEN=1 in TIM15_16_17_CR1 register to start Timer 1.

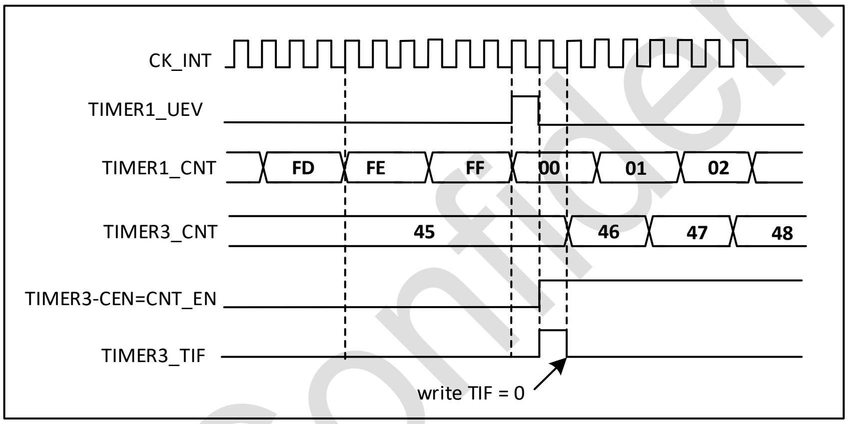


Figure 25-50 Trigger Timer 2 with Timer 1 update

In the previous example, both counters can be initialised before starting the count. Shows the action in the same configuration as 0, using the trigger mode instead of the gated mode (SMS=110 in the TIM2_SMCR register).

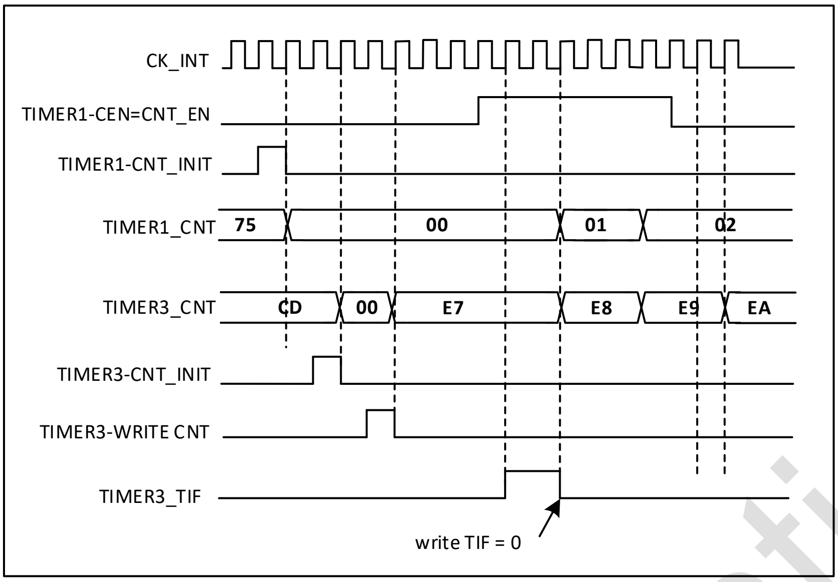


Figure 25-51 Trigger Timer 2 with Timer 1 enable

25.6.4. Use an external trigger to start 2 timers synchronously

This example enables Timer 1 when Timer 1's TI1 input rises, and enables Timer 1 while enabling Timer 2. To ensure counter alignment, Timer 1 must be configured in master/slave mode (corresponding to TI1 as slave and Timer 2 as master):

- Configure Timer 1 as master mode and send its enable as a trigger output (MMS=001 in TIM15_16_17_CR2 register).
- Configure Timer 1 as slave mode to get input trigger from TI1 (TS=100 of TIM15_16_17_SMCR register).
- Configure Timer 1 as trigger mode (SMS=110 in TIM15_16_17_SMCR register).
- Configure Timer 1 in Master/Slave mode with MSM=1 in the TIM15_16_17_SMCR register.
- Configure Timer 2 to get input triggering from Timer 1 (TS=000 of TIM2_SMCR register)
- Configure Timer 2 for trigger mode (SMS=110 of TIM2_SMCR register).

When a rising edge appears on TI1 of Timer 1, both timers start counting synchronously according to the internal clock and both TIF flags are set at the same time.

Note: In this example, both timers are initialised (setting the corresponding UG bits) before start-up and both counters start from 0, but an offset can be inserted between the timers by writing to either counter register (TIMx_CNT). In the diagram below you can see a delay between CNT_EN and CK_PSC for timer 1 in master/slave mode.

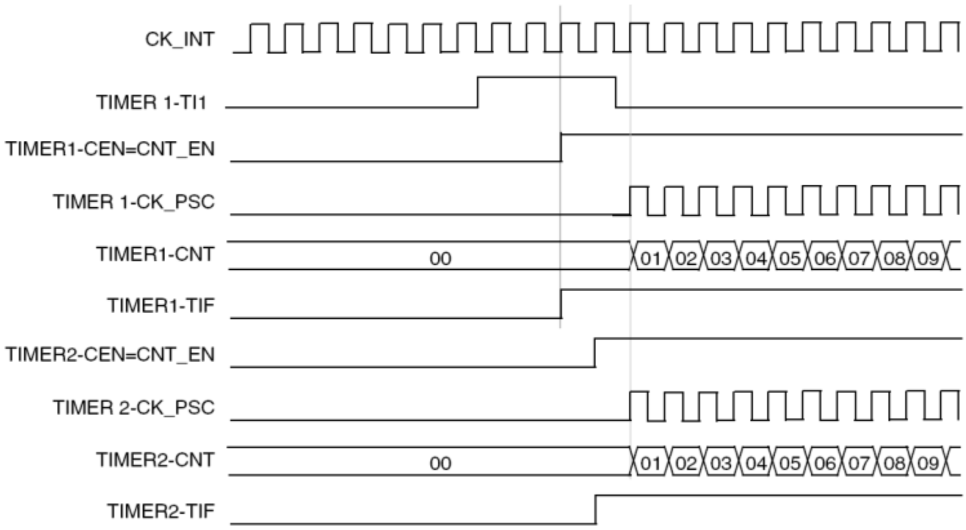


Figure 25-52 Use the TI1 input of Timer 1 to trigger Timer 1 and Timer 2

25.6.5. Debug mode

When the chip is in debug mode, the TIMx counter can continue to work normally or stop working depending on the setting of DBG_TIMx_STOP in the DBG module.

25.7. TIM15 registers

0x4001 4800 - 0x4001 4BFF TIM17

0x4001 4400 - 0x4001 47FF TIM16

0x4001 4000 - 0x4001 43FF TIM15

25.7.1. TIM15 control register 1 (TIMx_CR1)

Address offset:0x00

Reset value:0x0000 0000

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				OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	0	Reserved, must be kept at reset value.
9:8	CKD[1:0]	RW	00	Clock division

Bit	Name	R/W	Reset Value	Function
				This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and sampling clock used by the digital filters (ETR, TIx), 00: tDTS = tCK_INT 01: tDTS = 2 × tCK_INT 10: tDTS = 4 × tCK_INT 11: Reserved, do not use this configuration
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6:4	-	-	-	-
3	OPM	RW	0	Single pulse mode 0: Counter does not stop when an update event occurs 1: Counter stops when the next update event occurs (clear CEN bit)
2	URS	RW	0	Update request source This bit is set by software to select the update interrupt (UEV) sources. 0: If an update interrupt or DMA request is allowed, then any of the following events generate an UEV or a DMA request if enabled: – Counter overflow – Setting the UG bit 1: If an update interrupt or DMA request is allowed, only the counter overflow/underflow generates an update interrupt or DMA request interrupt or DMA request
1	UDIS	RW	0	Update disable This bit is set and cleared by software to enable/disable update interrupt (UEV) event generation. 0: UEV enabled. An UEV is generated by one of the following events: – Counter overflow – Setting the UG bit. Buffered registers are then loaded with their preload values. 1: UEV disabled. No UEV is generated, shadow registers keep their value (ARR, PSC, CCRx). The counter and the prescaler are reinitialized if the UG bit is set.
0	CEN	RW	0	Counter enable 0: Counter disabled 1: Counter enabled

Bit	Name	R/W	Reset Value	Function
				Note: External clock, gated mode and encoder mode can only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware

25.7.2. TIM15 control register 2 (TIMx_CR2)

Address offset:0x04

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					OIS2	OIS1N	OIS1		MMS[2:0]			CCDS	CCUS		CCPC
					RW	RW	RW		RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
15:11	Reserved	-	0	Reserved, must be kept at reset value
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 cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR 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 cannot be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register)
7	Reserved	-	-	-
6:4	MMS[2:0]	RW	000	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.

Bit	Name	R/W	Reset Value	Function
				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 enable. The Counter Enable signal is generated by a logic OR 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 TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: 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	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
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 COMG bit only 1: When capture/compare control bits are preloaded (CCPC = 1), they are updated by setting the COMG bit or when an rising edge occurs on TRGI *Note: This bit acts only on channels that have a complementary output.*
1	Res	-	0	Reserved, must be kept at reset value.
0	CCPC	RW	0	CCPC: Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded

Bit	Name	R/W	Reset Value	Function
				1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a communication event (COM) occurs (COMG bit set or rising edge detected on TRGI, depending on the CCUS bit). Note: This bit acts only on channels that have a complementary output.

25.7.3. TIM15 slave mode control register (TIMx_SMCR)

Address offset:0x08

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								MSM		TS[2:0]				SMS[2:0]	
								RW		RW				RW	

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	Reserved, must be kept at reset value.
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: Internal Trigger 0 (ITR0). 001: 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	Reserved	-	0	Reserved, must be kept at reset value
2：0	SMS[2:0]	RW	000	Slave mode selection

Bit	Name	R/W	Reset Value	Function
				When external signals are selected the active edge of the trigger signal (TRGI) is linked to thepolarity 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 TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - 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_ED 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.

TIM1 internal trigger connection

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	reserved	reserved	TIM3	TIM17 OC1

25.7.4. TIM15 DMA/Interrupt enable register (TIMx_DIER)

Address offset:0x0C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TDE	COMDE			CC2DE	CC1DE	UDE	BIE	TIE	COMIE			CC2IE	CC1IE	UIE
-	RW	RW			RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved		0	Reserved, must be kept at reset value.
14	TDE	RW	0	TDE: Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled
13	COMDE	RW	0	COMDE：COM DMA request is enable 0：COM DMA request is disabled 1：COM DMA request is disabled
11:12	Reserved	-	-	-
10	CC2DE	RW	0	CC2DE: Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
9	CC1DE	RW	0	CC1DE: Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request 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
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4:3	Reserved	-	0	Reserved, must be kept at reset value
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

25.7.5. TIM15 status register (TIMx_SR)

Address offset:0x010

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res			CC2OF	CC1OF	Res	BIF	TIF	COMIF			CC2IF	CC1F	UIF
-	-	-			Rc_w0	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0			Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
15:11	Reserved	-	0	Reserved, must be kept at reset value
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 TIMx_CCR1 register while CC1IF flag was already set
8	Res	Rc_w0	0	Reserved, must be kept at reset value.
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 This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode.It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending
5	COMIF	Rc_w0	0	COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software by writing it to '0'. 0: No COM event occurred.

				1: COM interrupt pending.
4:3	Reserved	-	0	Reserved, must be kept at reset value.
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 flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. When the contents of TIMx_CCR1 are greater than the contents of TIMx_ARR, the CC1IF bit goes high on the counter overflow (in upcounting and up/down-counting modes) or underflow (in downcounting mode) **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) Note: When CEN is turned on, this bit is also 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 repetition counter = 0) and if the UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS = 0 and UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by a trigger event (refer to TIM1 slave mode control register (TIM1_SMCR)), if URS = 0 and UDIS = 0 in the TIMx_CR1 register. (Refe to Slave Mode Control Register (TIMx_SMCR))

25.7.6. TIM15 event generation register (TIMx_EGR)

Address offset:0x14

Reset value:0x0000 0000

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			CC2G	CC1G	UG
-	-	-	--	-	-	-	-	W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	Reserved, must be kept at reset value.
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. MOE bit is cleared and BIF flag is set. If the corresponding interrupt is enabled, the correspond-ing interrupt 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: The TIF flag is set in TIMx_SR register. Related inter-rupt or DMA transfer can occur if enabled.
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 having a complemen-tary output.
4:3	Reserved	-	0	Reserved, must be kept at reset value.
2	CC2G	W	0	CC2G: 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 1: **If channel CC1 is configured as output:**

Bit	Name	R/W	Reset Value	Function
				CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. **If channel CC1 is configured as input:** The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
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). The counter is cleared if the center-aligned mode is selected or if DIR = 0 (upcounting), else it takes the auto-reload value (TIMx_ARR) if DIR = 1 (downcounting).

25.7.7. TIM15 capture/compare mode register1 (TIMx_CCMR1)

Address offset:0x18

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]			OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]															
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
15	Reserved	-	0	Reserved, must be kept at reset value
14:12	OC2M[2:0]	RW	0	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

Bit	Name	R/W	Reset Value	Function
				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 is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7	Reserved	-	0	Reserved, must be kept at reset value.
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 active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs (this mode is used to generate a timing base). 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 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = ‘0’) as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = ’1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: 1: 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). 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the

Bit	Name	R/W	Reset Value	Function
				output compare mode switches from “frozen” mode to “PWM” mode. 3: On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the OC1M active bits take the new value from the preloaded bits only when a COM event is generated.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note: 1: 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). 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). 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 CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. 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

Bit	Name	R/W	Reset Value	Function
				11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
15:12	IC2F	RW	0	Input capture 2 filter
11:10	IC2PSC[1:0]	RW	0	Input capture 2 prescaler
9:8	CC2S[1:0]	RW	0	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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER)
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6

Bit	Name	R/W	Reset Value	Function
				1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8 Note: Care must be taken that fDTS is replaced in the formula by CK_INT when ICxF[3:0] = 1, 2 or 3.
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’ (TIMx_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	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 is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

25.7.8. TIM15 capture/compare enable register (TIMx_CCER)

Address offset:0x20

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res							CC2NP		CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
-	-							RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:14	Reserved	-	0	Reserved, must be kept at reset value.

Bit	Name	R/W	Reset Value	Function
13:8	Reserved	-	0	Reserved, must be kept at reset value.
7	CC2NP	RW	0	Capture/Compare 2 complementary output polarity refer to CC1NP description
6	Reserved	-	0	Reserved, must be kept at reset value.
5	CC2P	RW	0	Capture/Compare 2 output polarity refer to CC1P description
4	CC2E	RW	0	Capture/Compare 2 output enable refer to CC1E description
3	CC1NP	RW	0	Capture/Compare 1 complementary output polarity 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 TIMx_BDTR register) and CC1S = "00" (the channel is configured in output).
2	CC1NE	RW	0	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and 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	Capture/Compare 1 output polarity **CC1 channel configured as output:** 0: OC1 active high 1: OC1 active low **CC1 channel configured as input:** CC1NP/CC1P bits select the active polarity of TI1FP1 and TI2FP1 for trigger or capture operations. 00: non-inverted/rising edge The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: inverted/falling edge The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode), TIxFP1 is inverted (trigger operation in gated mode or en-coder mode). 10: reserved, do not use this configuration.

Bit	Name	R/W	Reset Value	Function
				11: non-inverted/both edges The circuit is sensitive to both TIxFP1 rising and falling edges (capture or trigger operations in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Note: 1. On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bits only when a Commutation event is generated. 2. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	RW	0	Capture/Compare 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. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled. Note: On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bits only when a Commutation event is generated.

Table 25-1 Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states(1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output state	OCxN output state
1	x	0	0	1	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	1	0	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1

Control bits					Output states(1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output state	OCxN output state
		0	1	1	OCxREF + Polarity OCx = OCxREF xor CCxP, OCx_EN = 1	Output Disabled (not driven by the timer) OCxN = 0, OCxN_EN = 0
		1	0	0	OCREF + Polarity + dead-time OCx_EN = 1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN = 1
		1	0	1	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	1	0	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity OCx = OCxREF xor CCxP, OCx_EN = 1	Off-State (output enabled with inactive state) OCxN = CCxNP, OCxN_EN = 1
		0	0	0	OCREF + Polarity + dead-time OCx_EN = 1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer) Asynchronously: OCx = CCxP, OCx_EN = 0, OCxN = CCxNP, OCxN_EN = 0 Then if the clock is present: 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		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1 Then if the clock is present: 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		

1. When both outputs of a channel are not used (CCxE = CCxNE = 0), the OISx, OISxN, CCxP and CCxNP bits must be kept cleared.

Note: The state of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel state and the GPIOand AFIO registers.

25.7.9. TIM15 counter (TIMx_CNT)

Address offset:0x24

Reset value:0x0000 0000

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
15:0	CNT[15:0]	RW	0	Counter value

25.7.10. TIM15 prescaler (TIMx_PSC)

Address offset:0x28

Reset value:0x0000 0000

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
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

25.7.11. TIM15 auto-reload register (TIMx_ARR)

Address offset: 0x2c

Reset value:0x0000 FFFF

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
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

25.7.12. TIM15 repetition counter register (TIMx_RCR)

Address offset: 0x30

Reset value:0x0000 0000

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
15:8	Reserved			Reserved, must be kept at reset value.
7:0	REP[7:0]	RW	0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. 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.

25.7.13. TIM15 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value:0x0000 0000

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
15：0	CCR1[15:0]	RW/RO	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).

25.7.14. TIM15 capture/compare register 2 (TIMx_CCR2)

Address offset: 0x38

Reset value:0x0000 0000

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
15:0	CCR2[15:0]	RW/RO	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 TIMx_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 TIMx_CNT and signalled on OC2 output. **If channel CC2 is configured as input:** CCR2 is the counter value transferred by the last input capture 2 event (IC2).

25.7.15. TIM15 break and dead-time register (TIMx_BDTR)

Address offset: 0x44

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Note: Depending on the lock setting, the AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] bits can be write-protected and it is necessary to configure them when writing to the TIMx_BDTR register for the first time.

Bit	Name	R/W	Reset Value	Function
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically 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 TIMx_CCER register).
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 the break input is not be active) Note: This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_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 programmed (LOCK bits in TIMx_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
12	BKE	RW	0	Break enable 0: Break inputs (BRK and CCS clock failure event) disabled 1, Break inputs (BRK and CCS clock failure event) enabled

Bit	Name	R/W	Reset Value	Function
				Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
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 is not implemented if no complementary output is implemented in the timer. 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. Then, OC/OCN enable output signal = 1 Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are forced first with 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 soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_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 bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register and BKE/BKP/AOE bits in TIMx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_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 TIMx_CCMRx registers, as

Bit	Name	R/W	Reset Value	Function
				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 re-set. Once the TIMx_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]x tdtg with tdtg = tDTS. DTG[7:5] = 10x = > DT = (64+DTG[5:0])xtdtg with Tdtg = 2xtDTS. DTG[7:5] = 110 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 8xtDTS. DTG[7:5] = 111 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 16xtDTS. Example if TDTS = 125 ns (8 MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 us to 31750 ns by 250 ns steps, 32 us to 63 us by 1 us steps, 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 TIMx_BDTR register).

25.7.16. TIM15 DMA control register (TIMx_DCR)

Address offset: 0x48

Reset value:0x0000 0000

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
15:13	Reserved			Reserved, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the number of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIMx_DMAR address)

Bit	Name	R/W	Reset Value	Function
				00000: 1 transfer 00001: 2 transfers 00010: 3 transfers ... 10001: 18 transfers Example: Let us consider this transfer: DBL=7, DBA=TIM2_CR1 - If DBL=7 and DBA=TIM2_CR1 indicates the address of the data to be transferred, then the address of the transfer is given by (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL where (address of TIMx_CR1) + DBA plus 7 gives the address where the data will be written or read, so that the transfer of data will take place in the 7 registers starting at address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following may occur: - If the data is set to half-word (16 bits), then the data is transferred to all 7 registers. - If the data is set to byte, the data is still transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte and so on. For timers, therefore, the user must specify the width of the data to be transferred by the DMA.
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DBA [4:0]: DMA base address This 5-bit vector defines the base-address for DMA transfers (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. Example: 00000: TIMx_CR1, 00001: TIMx_CR2, 00010: TIMx_SMCR, ...

25.7.17. TIM15 DMA address for full transfer (TIMx_DMAR)

Address offset: 0x4C

Reset value:0x0000 0000

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
15：0	DMAB[31:0]	RW	0	DMA register for full transfer A read or write operation to the DMAR register accesses the register located at the following address TIMx_CR1 address) + (DBA + DMA index) , where TIMx_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIMx_DCR register, DMA index is automatically controlled by the DMA transfer, depending on the DBL defined in the TIMx_DCR register.

Note: When using the DMA continuous transfer function, the value of the CNDTR register of the corresponding channel in the DMA must correspond to the value of DBL in the TIMx_DCR register, otherwise this will not work properly.

25.7.18. TIM15 register map

<table>
<tr><th>Offset</th><th>Bit Width</th><th>Register</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="3">0x00</td><td rowspan="3">32</td><td>TIMx_CR1</td><td colspan="22" rowspan="2">Reserved</td><td colspan="2">CKD</td><td>ARPE</td><td colspan="3" rowspan="2">Reserved</td><td>OPM</td><td>URS</td><td>UDIS</td><td>CEN</td></tr>
<tr><td>Read/Write</td><td colspan="2">rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr>
<tr><td>Reset Value</td><td colspan="22">0</td><td colspan="2">0</td><td>0</td><td colspan="3">0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td rowspan="3">0x04</td><td rowspan="3">32</td><td>TIMx_CR2</td><td colspan="21" rowspan="2">Reserved</td><td>OIS2</td><td>OIS1N</td><td>OIS1</td><td rowspan="2">Reserved</td><td colspan="3">MMS</td><td>CCDS</td><td>CCUS</td><td rowspan="2">Reserved</td><td>CCPC</td></tr>
<tr><td>Read/Write</td><td>rw</td><td>rw</td><td>rw</td><td colspan="3">rw</td><td>rw</td><td>rw</td><td>rw</td></tr>
<tr><td>Reset Value</td><td colspan="21">0</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3">0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td rowspan="3">0x08</td><td rowspan="3">32</td><td>TIMx_SMCR</td><td colspan="24" rowspan="2">Reserved</td><td>MSM</td><td colspan="3">TS</td><td rowspan="2">Reserved</td><td colspan="3">SMS</td></tr>
<tr><td>Read/Write</td><td>rw</td><td colspan="3">rw</td><td colspan="3">rw</td></tr>
<tr><td>Reset Value</td><td colspan="24">0</td><td>0</td><td colspan="3">0</td><td>0</td><td colspan="3">0</td></tr>
<tr><td>0x0C</td><td>32</td><td>TIMx_DIER</td><td colspan="17">Reserved</td><td>TDE</td><td colspan="3">Reserved</td><td>CC2DE</td><td>CC1DE</td><td>UDE</td><td>BIE</td><td>TIE</td><td>COMIE</td><td colspan="2">Reserved</td><td>CC2IE</td><td>CC1IE</td><td>UIE</td></tr>
</table>

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Read/Write																		rw				rw	rw	rw	rw	rw	rw			rw	rw	rw
		Reset Value	0																	0	0			0	0	0	0	0	0	0		0	0	0
0x10	32	TIMx_SR	Reserved										IC2IR	IC1IR	Reserved		IC2IF	IC1IF	Reserved					CC2OF	CC1OF	Reserved	BIF	TIF	COMIF	Reserved		CC2IF	CC1IF	UIF
		Read/Write											rc_w0	rc_w0			rc_w0	rc_w0						rc_w0	rc_w0		rc_w0	rc_w0	rc_w0			rc_w0	rc_w0	rc_w0
		Reset Value											0	0			0	0						0	0	0	0	0	0			0	0	0
0x14	32	TIMx_EGR	Reserved																								BG	TG	COMG	Reserved		CC2G	CC1G	UG
		Read/Write																									w	w	w			w	w	w
		Reset Value	0																								0	0	0			0	0	0
0x18	32	TIMx_CCMR1:OUTPUT	Reserved																Reserved	OC2M			OC2PE	OC2FE	CC2S		Reserved	OC1M			OC1PE	OC1FE	CC1S	
		Read/Write																		rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	
		Reset Value	0																0	0	0	0	0	0	0		0	0	0	0	0	0	0	
0x18	32	TIMx_CCMR1:INPUT																	IC2F				IC2PSC		CC2S		IC1F				IC1PSC		CC1S	
		Read/Write																	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	
		Reset Value	0																0	0	0	0	0	0	0		0	0	0	0	0	0	0	
0x20	32	TIMx_CCER	Reserved																								CC2NP	Reserved	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		Read/Write																									rw		rw	rw	rw	rw	rw	rw
		Reset Value																									0	0	0	0	0	0	0	0
0x	32	TIMx_CNT	Reserved																CNT															

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24		Read/Write																	rw															
		Reset Value	0																0															
0x28	32	TIMx_PSC	Reserved																PSC															
		Read/Write																	rw															
		Reset Value	0																0															
0x2C	32	TIMx_ARR	Reserved																ARR															
		Read/Write																	rw															
		Reset Value	0																0xFFFF															
0x30	32	TIMx_RCR	Reserved																								REP							
		Read/Write																									rw							
		Reset Value	0																								0							
0x34	32	TIMx_CCR1	Reserved																CCR1															
		Read/Write																	rw/ro															
		Reset Value	0																0															
0x38	32	TIMx_CCR2	Reserved																CCR2															
		Read/Write																	rw/ro															
		Reset Value	0																0															
0x44	32	TIMx_BDTR	Reserved																MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
		Read/Write																	rw	rw	rw	rw	rw	rw	rw		rw							
		Reset Value	0																0	0	0	0	0	0	0		0							
0x	32	TIMx_DCR	Reserved																			DBL					Re-served			DBA				

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
48		Read/Write																				rw								rw				
		Reset Value	0																			0					0			0				
0x4C	32	TIMx_DMAR	Reserved																DMAB															
		Read/Write																	rw															
		Reset Value	0																0															

25.8. TIM16/TIM17 registers

0x4001 4800 - 0x4001 4BFF TIM17

0x4001 4400 - 0x4001 47FF TIM16

0x4001 4000 - 0x4001 43FF TIM15

25.8.1. TIM16/17 control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

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				OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15 :10	Reserved	-	0	Reserved, must be kept at reset value.
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (tDTS)used by the dead-time generators and the digital filters (TIx), 00: tDTS = tCK_INT 01: tDTS = 2*tCK_INT 10: tDTS = 4*tCK_INT 11: Reserved, do not program this value
7	ARPE	RW	0	Auto-reload preload enable 0: TIMx_ARR register is not buffered

Bit	Name	R/W	Reset Value	Function
				1: TIMx_ARR register is buffered
6:4	-	-	-	-
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: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
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 only work after the CEN bit is set by software. Trigger mode can automatically set the CEN bit by hardware.

25.8.2. TIM16/17 control register 2 (TIMx_CR2)

Address offset: 0x04

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						OIS1N	OIS1					CCDS	CCUS	Res	CCPC
Rw						RW	RW					Rw	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved	-	0	Reserved, must be kept at reset value
14:10	-	-	-	
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 TIMx_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 TIMx_BKR register).
7:4	-	-	-	-
3	CCDS	RW	0	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
2	CCUS	RW	0	Capture/compare control update selection 0: If the capture/compare control bits are pre-loaded (CCPC=1), they can only be updated by setting the COM bit. 1: If the capture/compare control bits are pre-loaded (CCPC=1), they can be updated by setting the COM bit or a rising edge on the TRGI or on a rising edge of TRGI to update them. Note: This bit will only work for channels with complementary outputs.
1	Res	-	0	Reserved, must be kept at reset value.
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded

Bit	Name	R/W	Reset Value	Function
				1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when COMG bit is set. Note: This bit acts only on channels that have a complementary output.

25.8.3. TIM16/17 DMA/interrupt enable register (TIM16/17_DIER)

Address offset: 0x0C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						**CC1DE**	**UDE**	**BIE**		**COMIE**				**CC1IE**	**UIE**
-						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved		0	Reserved, must be kept at reset value.
14:10	-	-	-	-
9	CC1DE	RW	0	CC1DE: capture/compare 1 DMA request enable 0: capture/compare 1 DMA request disabled. 1: capture/compare 1 DMA request enabled.
8	UDE	RW	0	UDE: Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled.
7	BIE	RW	0	BIE: break interrupt enable 0: break interrupt disabled. 1: break interrupt enabled.
6	-	-	-	-
5	COMIE	RW	0	COMIE: COM interrupt enable 0: COM interrupt disabled. 1: COM interrupt enabled.
4:2	-	-	-	-
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.

25.8.4. TIM16/17 status register (TIMx_SR)

Address offset: 0x010

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res				CC1OF		BIF		COMIF				CC1F	UIF
-	-	-				Rc_w0		Rc_w0		Rc_w0				Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
15：13	Reserved	-	0	Reserved, must be kept at reset value.
12:10	-	-	-	-
9	CC1OF	Rc_w0	0	Capture/compare 1 over capture flag This flag can be set by hardware only when the corresponding channel is configured for input capture. Writing 0 clears this bit. 0: No overcapture is generated, 1: When CC1IF is set, the counter value has been captured into the TIMx_CCR1 register.
8	Res	Rc_w0	0	Reserved, must be kept at reset value.
7	BIF	Rc_w0	0	Break Interrupt flag This bit is set by hardware once the break input is vaild. This bit can be cleared by software if the break input is invaild. 0: No break event is generated, 1: A valid level is detected on the break input.
6	-	-	-	-
5	COMIF	Rc_w0	0	COM interrupt flag This bit is set by hardware once a COM event is generated (when CcxE, CcxNE, OCxM have been updated). It is cleared by software. 0: No COM event is generated, 1: COM interrupt waiting for response
4:2	-	-	-	-
1	CC1IF	Rc_w0	0	Capture/Compare 1 Interrupt Flag If channel CC1 is configured in output mode: This bit is set by hardware when the counter value matches the compare value, except in center-align mode (refer to the CMS bit in the TIM1_CR1 register). It is cleared by software. 0: no match occurs,

Bit	Name	R/W	Reset Value	Function
				1: The value of TIMx_CNT matches the value of TIMx_CCR1. If channel CC1 is configured in input mode: This bit is set by hardware when a capture event occurs, it is cleared by software or cleared by reading TIMx_CCR1. 0: No input capture is generated, 1: Input capture occurred and the counter value has been loaded into TIMx_CCR1 (edge detected on IC1 with the same polarity as selected).
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 and if UDIS = 0 in the TIMx_CR1 register. –When CNT is reinitialized by software using the UG bit in the TIMx_EGR register, if URS = 0 and UDIS = 0 in the TIMx_CR1 register. (Refer to Slave Mode Control Register (TIMx_SMCR))

25.8.5. TIM16/17 event generation register (TIMx_EGR)

Address offset： 0x14

Reset value:0x0000 0000

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		COMG				CC1G	UG
-	-	-	--	-	-	-	-	W		W				W	W

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	Reserved, must be kept at reset value
7	BG	W	0	generate break event This bit is set by software to generate a break event and is automatically cleared by hardware. 0: no action, 1: Generate a break event. At this time, MOE = 0, BIF = 1, if the corresponding interrupt is enabled, the correspond-ing interrupt will be generated.

Bit	Name	R/W	Reset Value	Function
6	-	-	-	-
5	COMG	W	0	Capture/compare events, generate control update This bit is set by software and automatically cleared by hardware. 0: no action, 1: When CCPC = 1, the CcxE, CcxNE, OCxM bits are allowed to be updated. Note: This bit is only valid for channels with complementary output.
4:2	-	-	-	-
1	CC1G	W	0	Generate capture/compare 1 event This bit is set by software to generate a capture/compare event and is automatically cleared by hardware. 0: no action, 1: Generate a capture/compare event on channel CC1: If channel CC1 is configured as an output: Set CC1IF = 1, if the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If channel CC1 is configured as an input: The current counter value is captured to the TIMx_CCR1 register, and CC1IF = 1 is set. If the corresponding interrupt and DMA are enabled, the corresponding interrupt and DMA will be generated. If CC1IF is already 1, set CC1OF = 1.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action. 1: Re-initializes the timer counter and generates an update of the registers. Note that the prescaler counter is cleared too (but the prescaler ratio is not affected). If in center-align mode or DIR = 0 (upcounter), the counter will be cleared to 0, if DIR = 1 (downcounter), the counter will take the value of TIMx_ARR.

25.8.6. TIM16/17 capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 0x18

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

RES	RES			RES	RES	RES		RES	OC1M[2:0]			OC1PE	OC1FE	CC1S[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
15:7	Reserved	-		Reserved, must be kept at reset value.
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 active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. 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 - In upcounting, channel 1 is active as long as TIMx_CNT < TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF = ‘0’) as long as TIMx_CNT > TIMx_CCR1 else active (OC1REF = ’1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT > TIMx_CCR1 else inactive. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S = ’00’ (the channel is configured in output). Note: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Name	R/W	Reset Value	Function
3	OC1PE	RW	0	Output Compare 1 Preload Enable 0: The preload function of the TIM1_CCR1 register is disabled, the TIM1_CCR1 register can be written at any time, and the new value takes effect immediately. 1: Enable the preload function of the TIM1_CCR1 register. The read and write operations are only performed on the preload register. The preload value of TIM1_CCR1 is loaded into the current register when the update event arrives. Note 1: This bit cannot be modified once the LOCK level is set to 3 (LOCK bit in the TIMx_BDTR register) and CC1S = 00 (the channel is configured as an output). Note 2: Only in single pulse mode, PWM mode can be used without confirming the preload register, otherwise its action is undefined.
2	OC1FE	RW	0	Output Compare 1 Fast Enable This bit is used to speed up the CC output's response to trigger input events. 0: CC1 operates normally according to the value of the counter and CCR1, even if the flip-flop is on. When the flip-flop input has an active edge, the minimum delay to activate the CC1 output is 5 clock cycles. 1: The active edge input to the flip-flop acts as if a compare match had occurred. Therefore, OC is set to the comparison level regardless of the comparison result. The delay between the active edge of the sampling flip-flop and the CC1 output is shortened to 3 clock cycles. OC1FE only works when the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S[1:0]	RW	00	Capture/Compare 1 Select. These 2 bits define the direction of the channel (input/output), and the selection of the input pins: 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 off (CC1E = 0 in TIM16/17_CCER register).

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
15:12	IC2F	RW	0	Input capture 2filter
11:10	IC2PSC[1:0]	RW	0	Input capture 2 prescaler
9:8	CC2S[1:0]	RW	0	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 is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIMx_CCER).
7:4	IC1F[3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS / 2, N = 6 0101: fSAMPLING = fDTS / 2, N = 8 0110: fSAMPLING = fDTS / 4, N = 6 0111: fSAMPLING = fDTS / 4, N = 8 1000: fSAMPLING = fDTS / 8, N = 6 1001: fSAMPLING = fDTS / 8, N = 8 1010: fSAMPLING = fDTS / 16, N = 5 1011: fSAMPLING = fDTS / 16, N = 6 1100: fSAMPLING = fDTS / 16, N = 8 1101: fSAMPLING = fDTS / 32, N = 5 1110: fSAMPLING = fDTS / 32, N = 6 1111: fSAMPLING = fDTS / 32, N = 8
3:2	IC1PSC[1:0]	RW	00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1).

Bit	Name	R/W	Reset Value	Function
				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	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 Other: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = 0 in TIM16/17_CCER).

25.8.7. TIM16/17 capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value:0x0000 0000

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
15:4	Reserved	-	0	Reserved, must be kept at reset value.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity 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 TIMx_BDTR register) and CC1S = "00" (the channel is configured in 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, OIS1N and 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

Bit	Name	R/W	Reset Value	Function
				The CC1 channel is configured as an output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: The two bits of CC1NP/CC1P select whether the polarity signal of TI1FP1 or TI2FP1 is used as the trigger or capture signal. 00: Not inverted/rising edge: Capture occurs on the rising edge of TixFP1 (capture, reset trigger, external clock or trigger mode), TixFP1 is not inverted (gate trigger mode, code mode). 01: Inversion/Falling Edge: Not Inverted/Rising Edge: Capture occurs on the falling edge of TixFP1 (capture, reset trigger, external clock or trigger mode), TixFP1 is inverted (gate trigger mode, code mode). 10: Reserved, invalid configuration. 11: No reverse, double edge. Note: This bit cannot be modified once the LOCK level (LOCK bit in the TIMx_BDTR register) is set to 3 or 2 and CC1S = 00 (channel configured as output).
0	CC1E	RW	0	Input/Capture 1 Output Enable The CC1 channel is configured as an output: 0: Off - OC1 output is disabled, so the output level of OC1 depends on the value 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 value of the MOE, OSSI, OSSR, OIS1, OIS1N, CC1NE bits. The CC1 channel is configured as an input: This bit determines whether the value of the counter can capture the TIMx_CCR1 register. 0: Capture disabled 1: Capture enable

Table 25-2 output control bits for complementary OCx and OCxN channels with break feature

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 the timer)，OCx=0, OCx_EN=0	Output disabled (Not driven by the timer)，OCxN=0, OCxN_EN=0

Control bits					Output state	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
		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 异或 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	OCREF Complementary value（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 and disabled), OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF+Polarity + dead-time OCx_EN=1	OCREF Complementary value（not OCREF）+ polarity + dead-time OCN_EN=1
0	0	X	0	0	Output disabled(Not driven by the timer) Asynchronous: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 If the clock is present: after a dead time, it is assumed that OISx and OISxN do not both correspond to valid levels of OCx and OCxN, OCx=OISx and OCxN=OISxN.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	Off state (output enabled and at an invalid level) Asynchronous: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 If the clock is present: after a dead time, it is assumed that OISx and OISxN do not both correspond to valid levels of OCx and OCxN, OCx=OISx and OCxN=OISxN	
	1		1	0		
	1		1	1		

If neither of the 2 outputs of a channel is used (CCxE = CCxNE = 0) then OISx, OISxN, CCxP and CCxNP must all be cleared to zero.

Note: The status of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel status and the GPIO and AFIO registers.

25.8.8. TIM16/17 counter (TIMx_CNT)

Address offset: 0x24

Reset value:0x0000 0000

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
15:0	CNT[15:0]	RW	0	Counter value

25.8.9. TIM16/17 prescaler (TIMx_PSC)

Address offset: 0x28

Reset value:0x0000 0000

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
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active pre-scaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).

25.8.10. TIM16/17 auto-reload register (TIMx_ARR)

Address offset: 0x2c

Reset value:0x0000 FFFF

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
15:0	ARR[15:0]	RW	FFFF	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.

25.8.11. TIM16/17 repetition counter register (TIMx_RCR)

Address offset: 0x30

Reset value:0x0000 0000

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
15:8	Reserved			Reserved, must be kept at reset value.
7:0	REP[7:0]	RW	0	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to the number of PWM periods in edgealigned mode, the number of PWM half periods in center-aligned mode.

25.8.12. TIM16/17 capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value:0x0000 0000

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
15:0	CCR1[15:0]	RW/RO	0	Capture/Compare 1 value Condition: 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. Condition: channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

25.8.13. TIM16/17 break and dead-time register (TIMx_BDTR)

Address offset: 0x44

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Note: Depending on the lock setting, the AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] bits can be write-protected and it is necessary to configure them when writing to the TIMx_BDTR register for the first time.

Bit	Name	R/W	Reset Value	Function
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically 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 TIMx_CCER register) See OC/OCN enable description for more details (TIMx 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 the break input is not be active)

Bit	Name	R/W	Reset Value	Function
				Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_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 programmed (LOCK bits in TIMx_BDTR register).
12	BKE	RW	0	Break enable 0: Break inputs (BRK and BRK_ACTH) disabled 1, Break inputs (BRK and BRK_ACTH) enabled Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK bits in TIMx_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 is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details. 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. Then, OC/OCN enable output signal = 1 Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE = 0 on channels configured as outputs. See OC/OCN enable description for more details. 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0) 1: When inactive, OC/OCN outputs are forced first with 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 soon as the LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration

Bit	Name	R/W	Reset Value	Function
				These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected 01: LOCK Level 1 = DTG bits in TIMx_BDTR register, OISx and OISxN bits in TIMx_CR2 register and BKE/BKP/AOE bits in TIMx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_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 TIMx_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 TIMx_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]x tdtg with tdtg = tDTS DTG[7:5] = 10x = > DT = (64+DTG[5:0])xtdtg with Tdtg = 2xtDTS DTG[7:5] = 110 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 8xtDTS DTG[7:5] = 111 = > DT = (32+DTG[4:0])xtdtg with Tdtg = 16xtDTS Example if TDTS = 125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps, 16 µs to 31750 ns by 250 ns steps, 32 µs to 63 µs by 1 µs steps, 64 µs to 126 µs by 2 µs steps Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

25.8.14. TIM16/17 DMA control register (TIMx_DCR)

Address offset: 0x48

Reset value:0x0000 0000

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
15:13	Reserved			Reserved, must be kept at reset value.
12:8	DBL[4:0]	RW	0 0000	DMA burst length This 5-bit vector defines the length of DMA transfers (the timer recognizes a burst transfer when a read or a write access is done to the TIM16/17_DMAR address), Transfers can be in half-words or in bytes. 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers ... 10001: 18 transfers. Example: Let's consider a transmission like this: DBL=7, DBA=TIM2_CR1 - If DBL=7 and DBA=TIM2_CR1 indicates the address of the data to be transferred, then the address of the transfer is given by (address of TIMx_CR1) + DBA + (DMA index), where DMA index = DBL where (address of TIMx_CR1) + DBA plus 7 gives the address where the data will be written or read, so that the transfer of data will take place in the 7 registers starting at address (address of TIMx_CR1) + DBA. Depending on the setting of the DMA data length, the following may occur: - If the data is set to half-word (16 bits), then the data is transferred to all 7 registers. - If the data is set to byte, the data is still transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte and so on. For timers, therefore, the user must specify the width of the data to be transferred by the DMA.
7:5	Reserved	RW	0	Reserved, must be kept at reset value.
4:0	DBA[4:0]	RW	0 0000	DMA base address

Bit	Name	R/W	Reset Value	Function
				This 5-bits vector defines the base-address for DMA transfers (when read/write access are done through the TIM16/17_DMAR address). DBA is defined as an offset starting from the address of the TIM16/17_CR1 register. Example: 00000: TIM16/17_CR1 00001: TIM16/17_CR2 00010: TIM16/17_SMCR ......

25.8.15. DMA address for full transfer (TIMx_DMAR)

Address offset: 0x4C

Reset value:0x0000 0000

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
15：0	DMAB[31:0]	RW	0	DMA register for burst accesses A read or write access to the DMAR register accesses the register located at the Address: “(TIM16/17_CR1 address) + DBA + (DMA index)” in which: TIM16/17_CR1 address is the address of the control register 1, DBA is the DMA base address configured in TIM16/17_DCR register, DMA index is the offset automatically controlled by the DMA transfer, depending on the length of the transfer DBL in the TIM16/17_DCR register.

Note: When using the DMA continuous transfer function, the value of the CNDTR register of the corresponding channel in the DMA must correspond to the value of DBL in the TIMx_DCR register, otherwise this will not work properly.

25.8.16. TIM16/17 register map

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	32	TIMx_CR1	Reserved																						CKD		ARPE	Reserved			OPM	URS	UDIS	CEN
		Read/Write																							rw		rw				rw	rw	rw	rw
		Reset Value	0																						0		0	0			0	0	0	0
0x04	32	TIMx_CR2	Reserved																						OIS1N	OIS1	Reserved				CCDS	CCUS	Reserved	CCPC
		Read/Write																							rw	rw					rw	rw		rw
		Reset Value																							0	0					0	0	0	0
0x0C	32	TIMx_DIER	Reserved																						CC1DE	UDE	BIE	Reserved	COMIE	Reserved			CC1IE	UIE
		Read/Write																							rw	rw	rw		rw				rw	rw
		Reset Value																							0	0	0	0	0				0	0
0x10	32	TIMx_SR	Reserved											IC1IR	Reserved			IC1IF	Reserved						CC1OF	Reserved	BIF	Reserved	COMIF	Reserved			CC1IF	UIF
		Read/Write												rc_w0				rc_w0							rc_w0		rc_w0		rc_w0				rc_w0	rc_w0
		Reset Value												0				0							0	0	0	0	0				0	0
0x14	32	TIMx_EGR	Reserved																								BG	Reserved	COMG	Reserved			CC1G	UG
		Read/Write																									w		w				w	w
		Reset Value	0																								0	0	0				0	0
0x18	32	TIMx_CCMR1:OUTPUT	Reserved																									OC1M			OC1PE	Reserved	CC1S	
		Read/Write																										rw	rw	rw	rw		rw	
		Reset Value																										0	0	0	0	0	0	
0x18	32	TIMx_CCMR1:I	Reserved																								IC1F				IC1PSC		CC1S	

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		NPUT																																
		Read/Write																									rw	rw	rw	rw	rw	rw	rw	
		Reset Value																									0	0	0	0	0	0	0	
0x20	32	TIMx_CCER	Reserved																												CC1NP	CC1NE	CC1P	CC1E
		Read/Write																													rw	rw	rw	rw
		Reset Value																													0	0	0	0
0x24	32	TIMx_CNT	Reserved																CNT															
		Read/Write																	rw															
		Reset Value	0																0															
0x28	32	TIMx_PSC	Reserved																PSC															
		Read/Write																	rw															
		Reset Value	0																0															
0x2C	32	TIMx_ARR	Reserved																ARR															
		Read/Write																	rw															
		Reset Value	0																0xFFFF															
0x30	32	TIMx_RCR	Reserved																								REP							
		Read/Write																									rw							
		Reset Value	0																								0							
0x34	32	TIMx_CCR1	Reserved																CCR1															
		Read/Write																	rw/ro															
		Reset	0																0															

Offset	Bit Width	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Value																																
0x44	32	TIMx_BDTR	Reserved																MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
		Read/Write																	rw	rw	rw	rw	rw	rw	rw		rw							
		Reset Value	0																0	0	0	0	0	0	0		0							
0x48	32	TIMx_DCR	Reserved																			DBL					Reserved			DBA				
		Read/Write																				rw								rw				
		Reset Value	0																			0					0			0				
0x4C	32	TIMx_DMAR	Reserved																DMAB															
		Read/Write																	rw															
		Reset Value	0																0															

26. Low power timer (LPTIM)

26.1. Introduction

LPTIM is a 16-bit timer. The ability of LPTIM to wake up the system from a low-power mode makes it suitable for implementing low-power applications.

LPTIM introduces a flexible clocking scheme that provides the required functionality and performance while minimizing power consumption.

26.2. LPTIM main features

- 16-bit up counter
- 3-bit prescaler with 8 possible division factors (1, 2, 4, 8, 16, 32, 64, 128)
- Optional clock
- Internal clock source: LSE, LSI or APB clock
- 16-bit ARR reload register
- Continuous /Single mode

26.3. LPTIM functional description

26.3.1. LPTIM block diagram

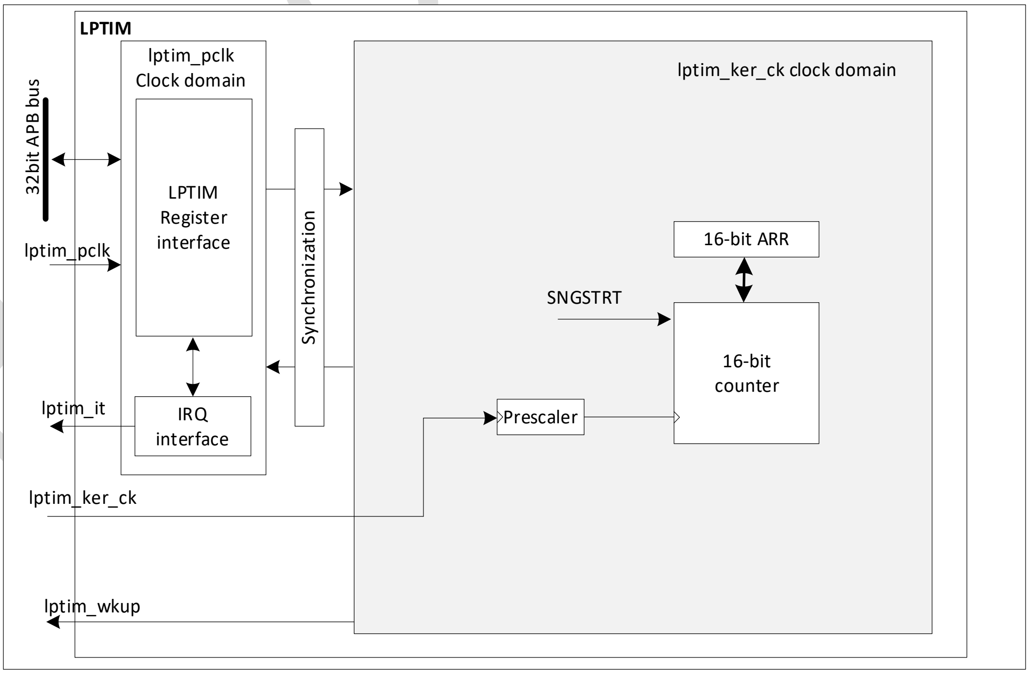


Figure 26-1 LPTIM block diagram

26.3.2. LPTIM reset and clock

LPTIM can use multiple clock sources for count.

Through the RCC module, it can be clocked with an internal clock signal (the clock signal can be selected among APB, LSI, LSE sources).

26.3.3. Prescaler

LPTIM 16-bit counter driven by a configurable 2 power prescaler control. The prescaler division ratio is controlled by PRESC[2:0].

The following table lists all cases:

Table 26-1 prescale factor

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

26.3.4. Operating mode

LPTIM has only one use timer mode.

- **Single mode**: The timer starts from a trigger event and stops when the ARR value is reached.

To enable single counting, the SNGSTRT bit must be set.

A new trigger event will restart the timer. Any trigger events after the counter has started and before the ARR is reached will be ignored.

- **Continuous mode:** The timer runs freely, starting from the trigger event and not stopping until

the timer is disable. To enable continuous counting, the LPTIM_CR.CNTSTRT bit must be set to 1. Setting LPTIM_CR.CNTSTRT will start the counter for continuous counting.

Change from single mode "on the fly" to continuous mode:

- If continuous mode was previously selected, setting LPTIM_CR.SNGSTRT switches the LPTIM to single mode. The counter (if active) will stop as soon as it reaches ARR.

- If single mode was previously selected, setting LPTIM_CR.CNTSTRT switches LPTIM to continuous mode. The counter (if active) is restarted as soon as it reaches the ARR.

26.3.5. Register update

The PRELOAD bit controls how the LPTIM_ARR register is updated:

- When the PRELOAD bit is reset to '0': The LPTIM_ARR register is updated immediately after any write access.
- When the PRELOAD bit is set to '1': If the timer has already started, LPTIM_ARR will be updated at the end of the current period.

The LPTIM APB interface and the LPTIM Kernel logic use different clocks, so there is a certain delay when the APB is written and the written value is applied to the counter comparator. During this delay period, any additional writes to these registers must be avoided.

26.3.6. Counter mode

LPTIM supports internal clock counting only, enabling the timer. The ENABLE bit in the LPTIM_CR register is used to enable/disable the LPTIM core logic. When the ENABLE bit is set, a delay of two counter clocks is required to enable LPTIM. The LPTIM_CFGR and LPTIM_IER registers can only be modified when LPTIM is disabled.

26.3.7. Counter reset

In order to reset the contents of the LPTIM_CNT register, a reset mechanism is provided:

Asynchronous reset mechanism:

Asynchronous reset is controlled by the RSTARE bit in the LPTIM_CR register. After setting the CONURST bit to 1, the reset signal is sent to the Kernel clock domain of the LPTIM. It is therefore important to note that several clock cycles have passed in the logic circuitry of the LPTIM Kernel clock domain before the reset takes effect. This will add a few extra counts to the LPTIM counter from the time the reset is triggered until the reset takes effect.

As 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 synchronise the reset signal from the APB clock domain when writing a 1 to the COUNTRST bit.

26.3.8. Debug mode

When the microcontroller enters debug mode, the LPTIM counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

26.4. LPTIM low power mode

Table 26-2 The difference between different low power modes of LPTIM Table

Mode	Description
Sleep	No effect. LPTIM interrupts cause the device to exit Sleep mode.
Stop	No effect when LPTIM is clocked by LSE or LSI. LPTIM interrupts cause the device to exit Stop.

26.5. LPTIM interrupt

The following events will generate interrupt/wake-up events if they are enabled in the LPTIM_IER register:

- Auto-reload match

Note: If the corresponding bit in the LPTIM_IER register (interrupt enable register) is set to 1 after the corresponding flag in the LPTIM_ISR register (status register) is set to 1, no interrupt will be generated.

Interrupt event	Description
Auto-reload match	When the content of the counter register (LPTIM_CNT) matches the content of the auto-reload register (LPTIM_ARR), the interrupt flag is set

26.6. LPTIM register

26.6.1. 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
31：7	Reserved	-	0	-
4	ARROK	R	0	Automatic reload register update OK.

Bit	Name	R/W	Reset Value	Function
				ARROK is set by hardware to notify the application that the APB bus write to LPTIM_ARR has completed successfully. Writing a 1 to LPTIM_ICR.ARROKCF clears the ARROK flag.
1	ARRM	R	0	Auto-reload match ARRM is set by hardware to inform the application that the LPTIM_CNT register value matches the LPTIM_ARR register value. Writing a 1 to the ARRMCF bit of the LPTIM_ICR register clears the ARRM flag.
0	Reserved	-	-	-

26.6.2. 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	ARRMCF	Res
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	-	Reserved
4	ARROKCF	RW	0	Auto-reload register update OK Clear Flag. Writing a 1 to this bit clears the ARROK flag in the LPTIM_ISR register
3：2	Reserved	-	-	Reserved
1	ARRMCF	RW	0	Auto-reload match clear flag Writing a 1 to this bit clears the ARRM flag in the LPTIM_ISR register
0	Reserved	-	-	

26.6.3. 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	ARRMOKIE	Res	Res	ARRMIE	Res
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	Reserved
4	ARROKIE	RW	0	Auto-reload register update OK interrupt enable 0:ARROK interrupt disabled 1:ARROK interrupt enabled
3：2	Reserved	-	-	Reserved
1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: ARRM interrupt disabled 1: ARRM interrupt enabled
0	Reserved	-	-	Reserved

26.6.4. 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	Res	Res	Res	Res	PRELOAD	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	PRESC[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	Res
				RW	RW	RW									

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	PRELOAD	RW	0	Register update mode The preload bit controls the LPTIM_ARR register update mode 0: Update registers after each APB bus write access 1: Registers are updated at the end of the current LPTIM period
21:12	Reserved	-	-	Reserved
11:9	PRESC[2:0]	RW	0	clock prescaler The PRESC bits configure the prescaler division factor. It can be a factor in the following divisions:

Bit	Name	R/W	Reset Value	Function
				000:/1 001:/2 010:/4 011:/8 100:/16 101:/32 110:/64 111:/128
8:0	Reserved	-	-	Reserved

26.6.5. 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	R STARE	COUNT RST	CNT STRT	SNG STRT	EN ABLE
											RW	RS	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	RSTARE	RW	0	Reset enable after read This bit is set and cleared by software. When RSTARE is set to '1', any read access to the LPTIM_CNT register will asynchronously reset the LPTIM_CNT register contents.
3	COUNTRST	RS	0	Counter Reset. This bit is set to 1 by software and cleared to 0 by hardware. when set to "1", this bit triggers a synchronous reset of the LPTIM_CNT counter register. Due to the synchronous nature of this reset, it only needs to be released after a synchronous delay of 3 LPTIM core clock cycles (the LPTIM core clock may be different from the APB clock). Note: Software must not set COUNTRST to "1" until it has been cleared to "0" by hardware. Therefore, the software should check that the COUNTRST bit is cleared to "0" before attempting to set it to "1".
2	CNTSTRT	RW	0	The timer starts continuous mode.

Bit	Name	R/W	Reset Value	Function
				This bit is set by software and this position 1 will start the LPTIM in continuous mode. If this bit is set to 1 while doing a single count mode count, the timer will not stop on the next pulse mode count with the LPTIM_ARR and LPTIM_CNT registers matched. the LPTIM counter remains counting in continuous mode. Note: This bit can only be set to 1 when LPTIM is enabled. it will be automatically reset by hardware.
1	SNGSTRT	RW	0	LPTIM starts single mode. This bit is set by software and cleared by hardware. Setting this bit will start LPTIM in single-pulse mode. Note: This bit can only be set when LPTIM is enabled. It will be reset automatically by hardware.
0	ENABLE	RW	0	LPTIM enable bit, set and cleared by software 0: LPTIM disabled 1: LPTIM enabled

26.6.6. LPTIM auto-reload 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	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	ARR	RW	0x0001	Auto-reload value ARR is the auto-reload value of LPTIM This register can only be updated when LPTIM is enabled

26.6.7. 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	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	-	0	Reserved
15: 0	CNT	R	0	counter value When LPTIM is running on an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two values returned are the same. A read access can be considered reliable when the values of two consecutive read accesses are equal.

26.6.8. LPTIM register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	LPTIM_ISR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARROK	Res	Res	ARRM	Res
	Reset value																												0			0	
0x04	LPTIM_ICR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARROKCF	Res	Res	ARRMCF	Res
	Reset value																												0			0	

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x08	LPTIM_IER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARROKIE	Res	Res	ARRMIEE	Res
	Reset value																												0			0	
0x0C	LPTIM_CFGR	Res	Res	Res	Res	Res	Res	Res	Res	Res	PRELOAD	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PRESC [2:0]			Res	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value										0											0	0	0									
0x10	LPTIM_CR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RSTARE	COUNTRST	CNTSTRT	SNGSTRT	ENABLE
	Reset value																												0	0	0	0	0
0x18	LPTIM_ARR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ARR[15:0]															

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	LPTIM_CNT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

27. Independent watchdog (IWDG)

27.1. Introduction

Independent watchdog is integrated in the chip, and this module has the characteristics of high-security level, accurate timing and flexible use. IWDG finds and resolves functional malfunctions due to software failure and triggers system reset when the counter reaches the specified timeout value.

The IWDG is clocked by LSI and thus stay active even if the main clock fails.

The IWDG is the best suited to applications that require the watchdog to run as a totally independent process outside the main application, without having high timing accuracy constraints.

27.2. IWDG main features

- Free-running downcounter
- Clocked from an independent RC oscillator (can operate in Stop modes)
- supports the RCC module to automatically enable the LSI as the IWDG clock after the CPU has configured the IWDG module start.
- When the watchdog is activated, a reset is generated when the counter counts to 0x000

27.3. IWDG functional description

27.3.1. IWDG block diagram

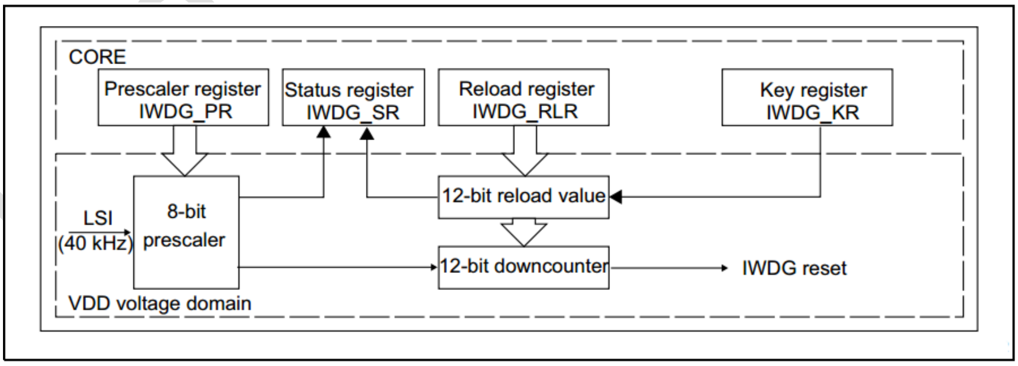


Figure 27-1 IWDG block diagram

Note: The watchdog function is in the VDD supply area, i.e. it still works in shutdown and standby mode.

When the independent watchdog is started by writing the value 0x0000 CCCC in the 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 the key value 0x0000 AAAA is written in the IWDG_KR register, the IWDG_RLR value is reloaded in the counter and the watchdog reset is prevented.

Table 27-1 Watchdog timeout 32kHz input clock (LSI)

Prescaler factor	PR[2:0]	Minimum time (ms) RL[11:0]=0x000	Maximum time (ms) RL[11:0]=0xFFF
/4	0	0.125	511.875
/8	1	0.25	1023.75
/16	2	0.5	2047.5
/32	3	1	4095
/64	4	2	8190
/128	5	4	16380
/256	(6 or 7)	8	32760

Note: These times are given according to the 32kHz clock. In reality, the internal RC frequency of the MCU will vary between 25kHz and 40kHz. Furthermore, even if the RC oscillator frequency is accurate, the exact timing still depends on the phase difference between the APB interface clock and the RC oscillator clock, so there will always be a full RC period that is uncertain. A relatively accurate watchdog timeout can be obtained by calibrating the LSI.

27.3.2. Hardware watchdog

If the “Hardware watchdog” feature is enabled through the device option bits, the watchdog is automatically enabled at power-on, and generates a reset unless the Key register is written by the software before the counter reaches end of count.

27.3.3. Register access protection

Write accesses to registers IWDG prescale, IWDG reload are 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 break the timing, e.g. writing 0x0000AAAA loading, the registers will be protected again.

If the value of the prescaler register, reload register is being updated, the status register is going to reflect it.

27.3.4. Debug mode and STOP mode

This function can only be used if the system supports DBG_MCU.

If the CPU enters debug mode, whether the IWDG continues to count or enters stop mode depends on the configuration of DBG_IWDG_STOP in the DBG module.

STOP mode is the IWDG_STOP bit in the option byte, which controls whether the IWDG continues to count normally or enters deepsleep mode when the CPU enters deepsleep mode, depending on the configuration of IWDG_STOP in the option byte in the FMC.

The configuration for IWDG_STOP in the Option byte is as follows:

Set the timer running state of the iwdg in stop mode

0: freeze timer

1: normal operation

The default IWDG_STOP in Option byte is " 1 ".

27.4. IWDG registers

These peripheral registers can be operated with half-words (16-bit) or words (32-bit).

27.4.1. Key register (IWDG_KR)

Address offset: 0x00

Reset value: 0x0000 0000 (reset by Standby mode)

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
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. Writing the key value 0xCCCC starts the watchdog (except if the hardware watchdog option is selected)

27.4.2. 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	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:0	PR[2:0]	RW	0	Prescaler divider. They are select the prescaler divider feeding the counter clock by set this register. PVU bit of 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

27.4.3. Reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF(reset by Standby mode)

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	RL[11:0]	RW	0	IWDG counter reload value. RL value will be loaded in the counter each time when the value 0xAAAA is writtern in the IWDG_KR register. The watch-dog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. The RVU bit in the IWDG_SR register must be reset in order to be able to change the reload value.

27.4.4. Status register (IWDG_SR)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	RVU	R	0	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the re-load 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 prescaler value is ongoing. It is reset by hardware when the prescaler update operation is completed.

Note: It is mandatory to wait until IWDG_PVU、IWDG_SR.RVU bit is reset before changing the IWDG_PR、IWDG_SR.RLR. However, after updating the IWDG_PR and/or the IWDG_RLR, it is not necessary to wait until IWDG_SR.PVU or IWDG_SR.RVU is reset before continuing code execution

27.4.5. IWDG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[2:0]		
																															0	0	0
0x08	IWDG_RLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RL[11:0]											
																						1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
																															0	0	0

28. System window watchdog (WWDG)

28.1. Introduction

The system window watchdog (WWDG) is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared.

An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value. This implies that the counter must be refreshed in a limited window.

The WWDG clock is prescaled from the APB clock and has a configurable time-window that can be programmed to detect abnormally late or early application behavior.

The WWDG is best suited for applications which require the watchdog to react within an accurate timing window.

28.2. WWDG main features

- Programmable free-running downcounte
- Conditional reset
  - Reset when the downcounter value becomes less than 0x40
  - Reset if the downcounter is reloaded outside the window
- Early wakeup interrupt(EWI): triggered (if enabled and the watchdog activated) when the downcounter is equal to 0x40.

28.3. WWDG functional description

If the watchdog is activated (the WDGA bit is set in the WWDG_CR register) and when the 7-bit downcounter (T[6:0] bits) is decremented from 0x40 to 0x3F (T6 becomes cleared), it initiates a reset. If the software reloads the counter while the counter is greater than the value stored in the window register, then a reset is generated.

The application program must write in the WWDG_CR register at regular intervals during normal operation to prevent an MCU reset. This operation must occur only when the counter value is lower than the window register value and higher than 0x3F. The value to be stored in the WWDG_CR register must be between 0xFF and 0xC0.

28.3.1. WWDG block diagram

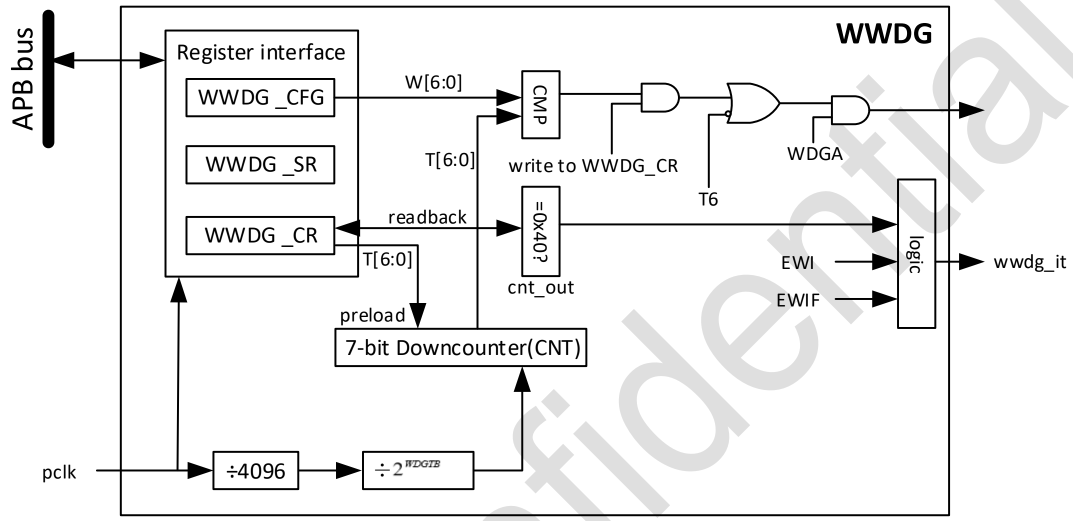


Figure 28-1 Window watchdog block diagram

28.3.2. Enabling the watchdog

When the user selects "software WWDG" in the WWDG_SW bit in the option byte, the watchdog is usually disabled after reset. Then by setting the WDGA bit in the WWDG_CR register, the WWDG module is enabled and then cannot be disabled unless a reset occurs.

In the option byte there is the wwdg_sw register, which also starts the watchdog, with the following values

0: hardware watchdog

1: software watchdog

In addition the watchdog will be started if either the hardware or software start is set.

28.3.3. Controlling the downcounter

This downcounter is free-running, counting down even if the watchdog is disabled. When the watchdog is enabled, the T6 bit must be set to prevent generating an immediate reset.

The T[5:0] bits contain the number of increments which represents the time delay before the watch-dog produces a reset.

The configuration register (WWDG_CFR) contains the high limit of the window: to prevent a reset, the downcounter must be reloaded when its value is lower than the window register value and greater than 0x3F.

Another way to reload the counter is to use the Early Wakeup Interrupt (EWI). This interrupt is ena-bled by setting the EWI bit in the WWDG_CFR register. When the down counter reaches 0x40, this interrupt is generated, and the corresponding interrupt service routine (ISR) can be used to load the counter to prevent the WWDG from being reset. This interrupt can be cleared by writing '0' in the WWDG_SR register.

Note: A software reset can be generated using the T6 bit (set WDGA bit to '1' and T6 bit to '0')

28.3.4. Advanced watchdog interrupt feature

The Early Wakeup Interrupt (EWI) can be used if specific safety operations or data logging must be per-formed before the actual reset is generated. The EWI interrupt is enabled by setting the EWI bit in the WWDG_CFR register. When the downcounter reaches the value 0x40, an EWI interrupt is generated and the corresponding interrupt service routine (ISR) can be used to trigger specific ac-tions.

28.3.5. How to program the watchdog timeout

When writing to the WWDG_CR register, always write 1 in the T6 bit to avoid generating an immedi-ate reset

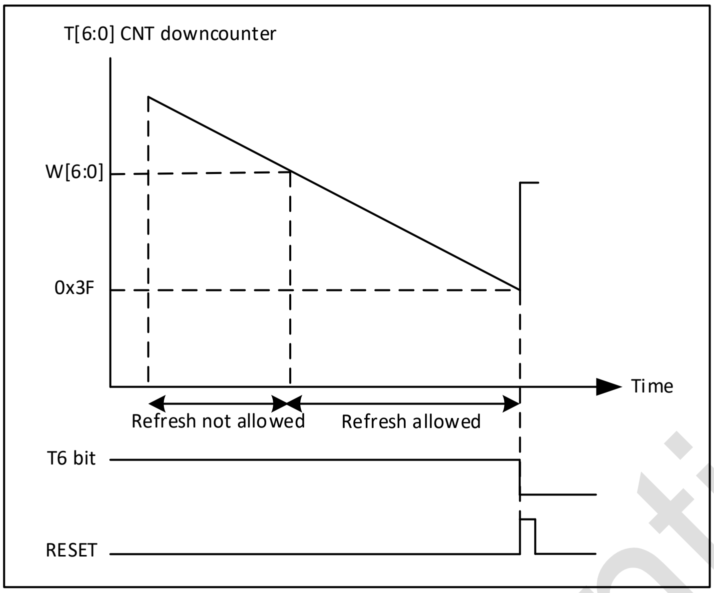


Figure 28-2 Window watchdog timing diagram

The formula to calculate the timeout value is given by:

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WWDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

28.3.6. Debug mode

When the microcontroller is in debug mode (Cortex-M4 core stopped), the WWDG counter can continue or stop depending on the status of the DBG_WWDG_STOP configuration bit in the debug module. See the section on debug mode for details.

28.4. WWDG registers

28.4.1. 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 by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: Watchdog disabled 1: Watchdog enabled
6:0	T[6:0]	RW	7F	7-bit counter (MSB to LSB) These bits contain the value of the watchdog counter. It is decremented every ($4096x2^{WDGTB}$) PCLK cycles. A reset is produced when it is decremented from 0x40 to 0x3F (T6 becomes cleared).

28.4.2. 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	RES	-	Reserved
9	EWI	RS	0	Early wakeup interrupt When set, an interrupt occurs whenever the counter reaches the value 0x40. This interrupt is only cleared by hardware after a reset.
8:7	WDGTB[1:0]	RW	2'b0	Timer base The time base of the prescaler can be modified as follows: 00: CK Counter Clock (PCLK div 4096) div 1 01: CK Counter Clock (PCLK div 4096) div 2 10: CK Counter Clock (PCLK div 4096) div 4 11: CK Counter Clock (PCLK div 4096) div 8
6:0	W[6:0]	RW	7'h7F	7-bit window value

Bit	Name	R/W	Reset Value	Function
				These bits contain the window value to be compared to the downcounter.

28.4.3. 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	RES	-	Reserved
0	EWIF	RC_W0	0	Early wakeup interrupt flag This bit is set by hardware when the counter has reached the value 40h. It must be cleared by software by writing ‘0’. A write of ‘1’ has no effect. This bit is also set if the interrupt is not enabled.

28.4.4. WWDG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	WWDG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T[6:0]						
	Reset value																									0	1	1	1	1	1	1	1
0x04	WWDG_	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB[1:0]		W[6:0]						

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CFR																																
	Reset value																							0	0	0	1	1	1	1	1	1	1
0x08	WWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
	Reset value																																0

29. Real-time clock (RTC)

29.1. Introduction

The real-time clock (RTC) is an independent timer/counter. The RTC module has a set of continuous counting counters, which can provide the function of clock-calendar under the corresponding software configuration. Modifying the value of the counter can reset the current time and date of the system.

29.2. RTC main features

- Programmable prescale factor: up to 2^{20}
- 32-bit programmable counter for longer time period measurement
- Two separate clocks: PCLK and RTC clock for APB interface (PCLK clock frequency is more than four times faster than RTC clock frequency)
- Three RTC clock sources:
  - HSE clock divided by 128
  - LSE oscillator clock
  - LSI oscillator clock
- 2 independent reset types:
  - APB1 interface reset by the system;
  - The RTC cores (prescaler, alarm, counter and divider) can only be reset by the backup domain.
- Three dedicated 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 rolls back to 0

29.3. RTC functional description

29.3.1. Overview

The RTC consists of two main parts (see diagram below). The first part (APB interface) is used to connect with the APB bus. The other part (RTC core) consists of a set of programmable counters, divided into two main modules:

The first module is the prescaler module of the RTC, which can be programmed to generate the RTC time base TR_CLK up to 1 second. The RTC prescaler module contains a 20-bit programmable divider (RTC prescaler). The RTC generates an interrupt (seconds interrupt) in every TR_CLK cycle if the corresponding enable bit is set in the RTC_CR register.

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 with the programmable time stored in the RTC_ALR register. If the corresponding enable bit is set in the RTC_CR control register, an alarm interrupt will be generated on a compare match.

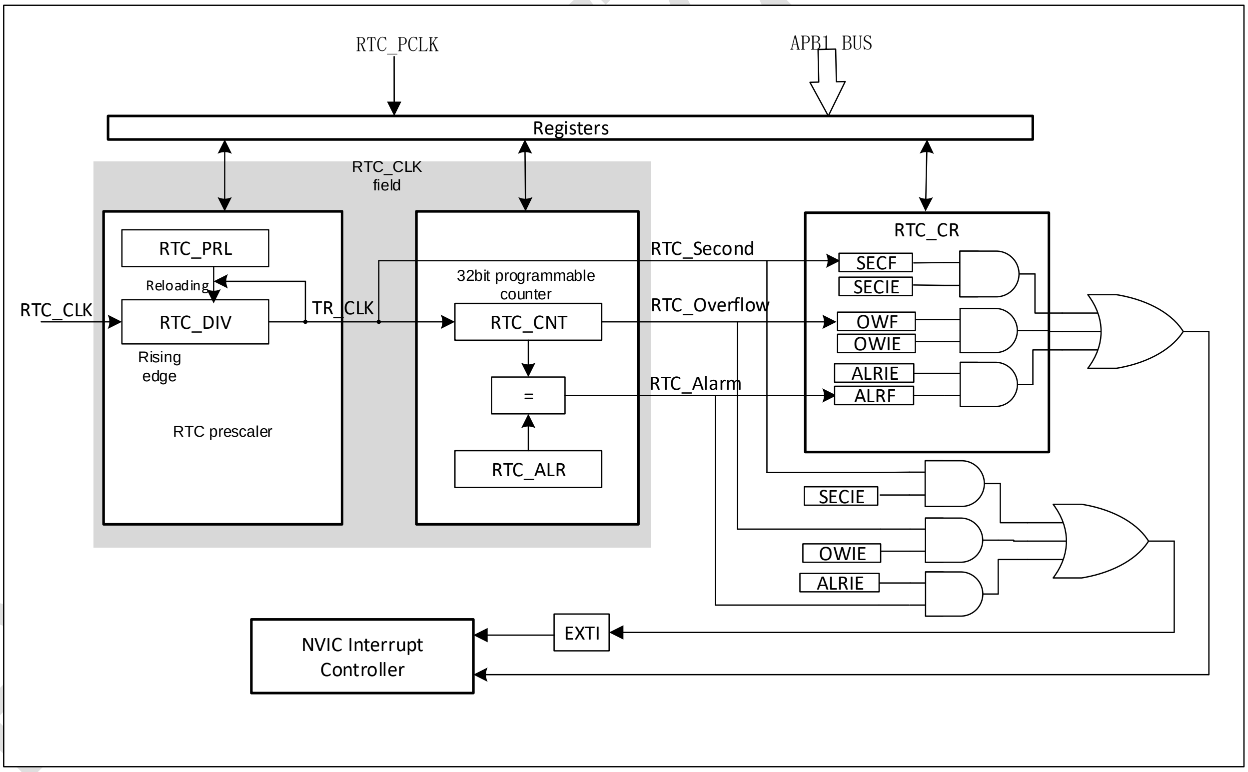


Figure 29-1 RTC block diagram

The DBP bit of the PWR_CR1 register is used to control the write protection disable to the RTC register. By default, DBP = 0, the RTC register cannot be written and accessed, and the RTC register can be written only after the software sets the DBP.

29.3.2. Resetting RTC register

All registers of RTC are reset from power-on reset (POR/PDR/BOR) and software reset of RTC (RCC_BDCR.16), and the rest of reset sources (NRST/IWDG/WWDG/OBL/SYSRESETREQ) have no effect.

Note that when the reset source for the RTC module cannot be generated, not only the RTC module cannot be reset, but also the clock source enable signal sent to the RTC module cannot be reset, nor the clock source selection control sent to the RTC module.

29.3.3. Reading RTC register

The RTC core and the APB clock are completely independent. The values of the RTC_DIV, RTC_CNT, and RTC_ALR registers accessed by the configuration software are all through the APB interface, but the update of the relevant readable registers is internally first passed through the rising edge of the RTC clock, and then synchronized again using the APB clock. The same is true for the flag register of the RTC.

If the APB interface has been disabled before, and the read operation is performed immediately after the APB interface is enabled (at this time, before the register is updated for the first time), the read operation may be corrupted (usually read returns 0). This situation will occur in the following situations:

- System reset or power-on reset.
- The system has just woken up from standby mode
- The system has just woken up from shutdown mode (in this case the count value is just not updated because the CPU is not working and the system clock is stopped, but the RTC counts normally and the count value is not synchronised to the VDD area)

In all the above-mentioned situations, the RTC core circuit keeps running, and the APB interface is turned off (reset or no clock).

Correspondingly, when reading the RTC register, after the RTC APB interface has been closed, the software must wait for the RSF bit (register synchronization flag) of the RTC_CRL register to be set by hardware.

Note: The APB1 interface of the RTC is not affected by low power modes such as WFI and WFE.

Description:

- CPU-readable registers include RTC_CR, RTC_CNT and RTC_DIV;
- RTC_CR register for the RTC_PCLK field, which can be read by the CPU at any time to a stable value;
- RTC_CNT and RTC_DIV are derived from the RTC_CLK domain. The RTC_DIV register is updated on the rising edge of each RTC_CLK after the RTC is operational; the RTC_CNT and the flags derived from the RTC_CLK clock field are also updated using the same signal as the RTC_DIV register, although this does not change the value of the RTC_CNT on every update;
- RSF is implemented in the RTC_PCLK domain and is set when the pulse signal is valid after the RTC_CLK is synchronised to RTC_PCLK;
- RSF controls only the timing of RTC_CNT and RTC_DIV reads (hardware will not control).

29.3.4. Configuring RTC register

The RTC_PRL, RTC_CNT, RTC_ALR registers can only be written to by entering the configuration mode by setting the CNF bit of the RTC_CRL register.

In addition, writing to any RTC register is only enabled when the previous write operation is finished. To enable the software to detect this condition, the chip provides RTOFF status bit in the RTC_CR register, which is used to display the update status of the register. A new value can be written to the RTC register only when the RTOFF status bit is 1.

Configuration process

1. Poll RTOFF until the bit goes high (indicating that the previous configuration has been completed).
2. Set CNF bit to enter configuration mode (the RTOFF bit remains at 1 at this point to ensure that RTOFF = 1 when the CPU is writing registers).
3. Write 1 or more RTC registers (RTOFF is still 1 in this process, RTC_CLK domain registers are written to buffer registers in this step).

4. Clear CNF bit to exit configuration mode (hardware detects that CNF is clear and starts to perform the write register operation in the previous step and starts to write to the RTC_CLK domain registers; at the same time RTOFF is cleared to zero).
5. Poll RTOFF, wait until it changes to 1 (RTOFF is set after the buffer register is written to the RTC_CLK field).

The write operation is performed only when the CNF bit is cleared, and at least 3 RTC_CLK cycles are required to complete the write operation. (The configuration cannot be restarted 3 RTC_CLK after the CNF flag bit is cleared, otherwise a configuration error will occur (controlled at this point by RTOFF=0))

Description:

1. RTOFF=1 when the CPU writes a register during this procedure;
2. CPU write cycle from CNF=1 to CNF=0, configuring other registers between these two operations;
3. Writing CNF to 1 first and then clearing CNF to zero, this operation clears RTOFF; writing only CNF=0 or writing CNF=1 and then not clearing it does not clear RTOFF;
4. Write CNF to 1 and then clear CNF to zero; this operation initiates the process of writing the buffer register to the RTC_CLK domain register;
5. RTOFF implementation in the RTC_PCLK domain;

29.3.5. RTC flag assertion

On every RTC clock cycle, before changing the RTC counter, the hardware sets the RTC seconds flag (SECF). On the last RTC clock cycle before the counter reaches 0x0000, the RTC overflow flag (OWF) is set.

RTC_Alarm and RTC alarm flag (ALRF) are set in the RTC clock cycle before the value of the counter reaches the value of the alarm register plus 1 (RTC_ALR+1). Writes to the RTC alarm must be synchronized with the RTC seconds flag using one of the following procedures:

(1) Use the RTC alarm interrupt and modify the RTC alarm and/or 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.

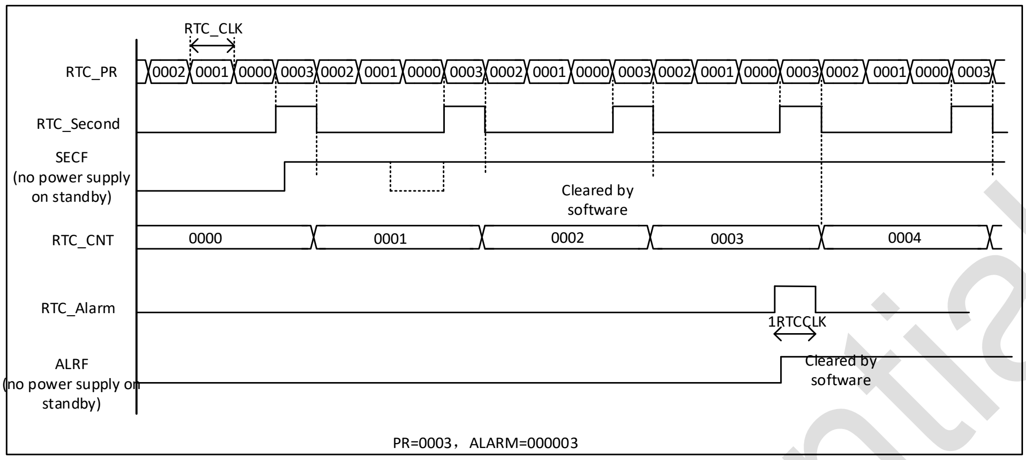


Figure 29-2 Example of RTC seconds and alarm waveforms, PR = 0003, ALARM = 00004

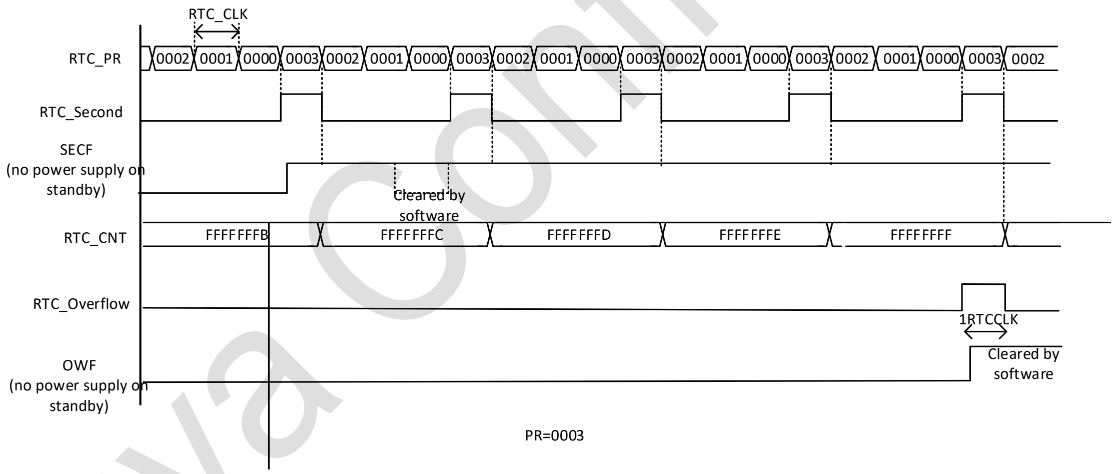


Figure 29-3 RTC overflow waveform example, PR = 0003

29.3.6. RTC calibration

For measurement purposes, the RTC clock divided by 64 can be output on the IO pin (PA4). This function is achieved by setting the CCO bit (RTCCR register).

By configuring the CAL[6:0] bits, the clock can be slowed down to 121 PPM.

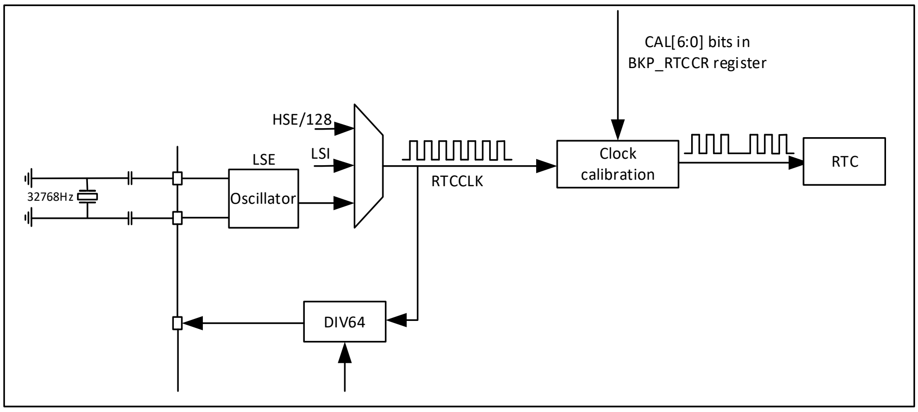


Figure 29-4 RTC calibration chart

29.4. RTC registers

29.4.1. RTC control register high bit (RTC_CRH)

Address offset: 0x00

Reset value: 0x0000

When PWR_CR1.DBP is 1, it is allowed to write to this register.

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved			
2	OWIE	RW	0	Overflow interrupt enable bit 0: Overflow interrupt disable 1: Overflow interrupt enable
1	ALRIE	RW	0	Alarm interrupt enable bit 0: Alarm interrupt disabled 1: Alarm interrupt enabled
0	SECIE	RW	0	Second interrupt enable bit 0: Second interrupt disable 1: Second interrupt enable

These bits are used to mask interrupt requests. Note that, all interrupts are masked after system reset, so it is possible to ensure that there are no pending interrupt requests after initialization by writing to the RTC register. When the peripheral is completing the previous write operation (the flag bit RTOFF = 0), the RTC_CRH register cannot be written.

The RTC function is controlled by this control register. Some certain bits must use a dedicated configuration flow to be written.

29.4.2. RTC control registe low bit (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. This bit is used by the RTC module to indicate the status of the last operation on its registers (indicating whether the operation is complete). If this bit is '0', it means that no RTC register can be written. 0: The last write operation to the RTC register is still in progress 1: The last write operation to the RTC register has been completed
4	CNF	RW	0	Configuration flag This bit must be set to '1' by software to enter configuration mode, allowing new values to be written to the RTC_CNT, RTC_ALR or RTC_PRL registers. Only after

Bit	Name	R/W	Reset Value	Function
				this bit is set to '1' and cleared to '0' again by software, the write operation will be performed. 0: Exit configuration mode (start to update 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 this bit to '1', and the software clears this bit. This bit must be cleared to '0' by software after an APB reset, or after the APB clock is stopped. Before performing any read operations, the user program must wait for this bit to be set to '1' by hardware to ensure that RTC_CNT, RTC_ALR or RTC_PRL have been synchronized. 0: Register has not been synchronized 1: Register has been synchronized
2	OWF	RC_W0	0	Overflow flag This bit is set to '1' by hardware when the 32-bit programmable counter overflows. If OWIE = 1 in the RTC_CRH register, an interrupt will be generated. This bit can only be cleared to '0' by software, writing '1' has no effect. 0: No overflow 1: 32-bit programmable counter overflow
1	ALRF	RC_W0	0	Alarm flag When the 32-bit programmable counter reaches the predetermined value set by the RTC_ALR register, this bit is set to '1' by hardware. An interrupt is generated if ALRIE = 1 in the RTC_CRH register. This bit can only be cleared to '0' by software, writing '1' has no effect. 0: No alarm clock 1: There is an alarm clock
0	SECF	RC_W0	0	Second flag When the 32-bit programmable prescaler overflows, this bit is set to '1' by hardware and the RTC counter is incremented by 1. Therefore, this flag provides a periodic signal (usually 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, writing '1' has no effect. 0: The second mark condition is not established 1: The second mark condition is established

The function of the RTC is controlled by this control register. When the peripheral is continuing the last write operation (RTOFF = 0), the RTC_CR register cannot be written.

Notes:

1. Any flag bit will remain pending until the appropriate RTC_CR request bit is reset by software, indicating that the requested interrupt has been accepted.

2. On reset all interrupts are disabled and no pending interrupt requests can be made to the RTC register for write operations (meaning writes from the buffer register to the RTC_CLK field register).

3. When APB1 clock is not running, OWF, ALRF, SECF and RSF bits are not updated (cannot be synchronized).

4. The OWF, ALRF, SECF and RSF bits can only be set by hardware and cleared by software.

5. If ALRF = 1 and ALRIE = 1, RTC global interrupts are allowed to be generated. If the EXTI Line 17 interrupt is allowed to be generated in the EXTI controller, the RTC Global Interrupt and RTC Alarm Interrupt are allowed to be generated. 6.

6. If ALRF=1, the RTC alarm interrupt is allowed to be generated if the interrupt mode of EXTI line 17 is set in the EXTI controller; if the event mode of EXTI line 17 is set in the EXTI controller, a pulse is generated on this line (no RTC alarm interrupt is generated).

29.4.3. RTC prescaler reload value high (RTC_PRLH)

The PRL register holds the periodic count value of the RTC prescaler. This register is write-protected by RTOFF bit of RTC_CR register, only RTOFF = 1 can write operation.

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 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 , the RTC interrupt and flag bit cannot be generated correctly

29.4.4. RTC prescaler reload value low (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 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 , the RTC interrupt and flag bit cannot be generated correctly

29.4.5. RTC prescaler divide register high (RTC_DIVH)

At each TR_CLK cycle, the value of the RTC_PRL register is reloaded into the RTC prescaler counter. In order to get an accurate clock, it is possible to read the current value of the prescaler counter (without stopping it), which is stored in the RTC_DIV register.

This register is read-only, when the value of the RTC_PRL or RTC_CNT register changes, it will be reloaded by (RTC_PRL).

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	Reverved			
3:0	RTC_DIV[19:16]	R	0	RTC clock dividers

29.4.6. RTC prescaler divide factor 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	Reverved	-	-	Reverved
15:0	DIV[15:0]	R	0x8000	RTC clock dividers

29.4.7. RTC count register high (RTC_CNTH)

The RTC module has a 32-bit programmable counter, which is accessed through two 16-bit registers and counts based on the TR_CLK generated by the prescaler.

The RTC_CNT register holds the count value of this counter. The registers are write-protected and can only be written when RTOFF = 1. Write to the high 16bit RTC_CNTH or low 16bit RTC_CNTL register, directly load it into the corresponding programmable counter, and reload the RTC prescaler. When a read operation occurs, the current value of the counter (system date) 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 high 16bits of the RTC core counter When reading the RTC_CNTH register, it returns the high 16bits of the current value of the RTC counter register. This register can only be written to by entering configuration mode.

29.4.8. RTC count 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	The lower 16 bits of the RTC core counter When reading the RTC_CNTL register, it returns the lower 16 bits of the current value of the RTC counter register. This register can only be written to by entering configuration mode.

29.4.9. RTC alarm register high (RTC_ALRH)

When the programmable counter (count) reaches the 32bit 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 allowed only when 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]	W	0xFFFF	RTC alarm high 16bit Software can write the high 16bit of Alarm time. Writing to this register must enter configuration mode.

29.4.10. RTC alarm register low (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 low 16bit Software can write the low 16bit of Alarm time. Writing to this register must enter configuration mode.

29.4.11. RTC clock calibration register (BKP_RTCCR)

Address offset: 0x2C

Reset value: 0x0000(only can be reset by por and bdcr soft 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	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			
9	ASOS	RW	0	Alarm or second output selection When the ASOE bit is set, the ASOS bit can be used to select whether the output on the Pin is RTC second pulse or Alarm pulse signal 0: RTC Alarm pulse signal 1: RTC second pulse signal
8	ASOE	RW	0	Alarm or second output enable When this bit is set, the ASOS bit determines whether the output on the pin is the RTC second pulse or the Alarm pulse signal.
7	CCO	RW	0	Calibration clock output 0: No effect 1: When this bit is set, the 64 frequency division of the RTC clock is output on the pin
6:0	CAL[6:0]	RW	0	Calibration value This value shows the negligible number of clock pulses per 2^{20} clock pulses, which allows the

Bit	Name	R/W	Reset Value	Function
				RTC to calibrate to slow down the clock in steps of 1000000/2^{20} PPM. RTC clock can be slowed down from 0 to 121PPM

29.4.12. RTC register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RTC_CRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OWIE	ALRIE	SECIE
	Reset value																														0	0	0
0x04	RTC_CRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	SECF
	Reset value																											1	0	0	0	0	0
0x08	RTC_PRLH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[19:16]			
	Reset value																													0	0	0	0
0x0C	RTC_PRLL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	RTC_DIVH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_DIV [19:16]			
	Reset																													0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value																																
0x14	RTC_DIVL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	RTC_CNTH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[31:16]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	RTC_CNTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	RTC_ALRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[31:16]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x24	RTC_ALRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	BKP_RTCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASOS	ASOE	CCO	CAL[6:0]						
	Reset value																							0	0	0	0	0	0	0	0	0	0

30. Inter-integrated circuit (I2C) interface

30.1. Introduction

The I2C (inter-integrated circuit) bus interface handles communications between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports Standard-mode (Sm), Fast-mode (Fm) and Fast-mode Plus (Fm+).

Depending on the needs of a particular device, DMA can be used to lighten the load on the CPU.

30.2. I2C main features

- Slave and master modes
- Multimaster capability: Can be master or slave
- Support different communication speeds
  - Standard-mode: up to 100 kHz
  - Fast-mode: up to 400 kHz
- As Master
  - issue clock
  - issue Start & Stop clock
- As slave
  - Programmable I2C address detection
  - Stop bit discovery
- 7-bit addressing mode
- General call
- Status flag bit
  - Transmit/receive mode flag bit
  - Byte transfer completion flag bit
  - I2C busy flag bit
- error flag bit

- Host Arbitration Lost
- ACK failure after address/data transfer
- Start/Stop error
- Overrun/Underrun(Clock stretch function disabled)

■ Optional clock stretching

■ Single-byte buffer with DMA capability

■ Software reset

■ Analog noise filter function

■ Configurable PEC (packet error checking) generation and verification

- PEC value can be sent in the last bytes in Tx mode
- The last byte does PEC error checking

■ SMBus compatible

- 25ms clock low timeout delay
- 10ms cumulative master clock low scaling time
- 25ms cumulative clock low scaling time for slave devices
- Hardware PEC generation/checking with ACK control
- Address Resolution Protocol (ARP) support.

30.3. I2C functional description

30.3.1. I2C block diagram

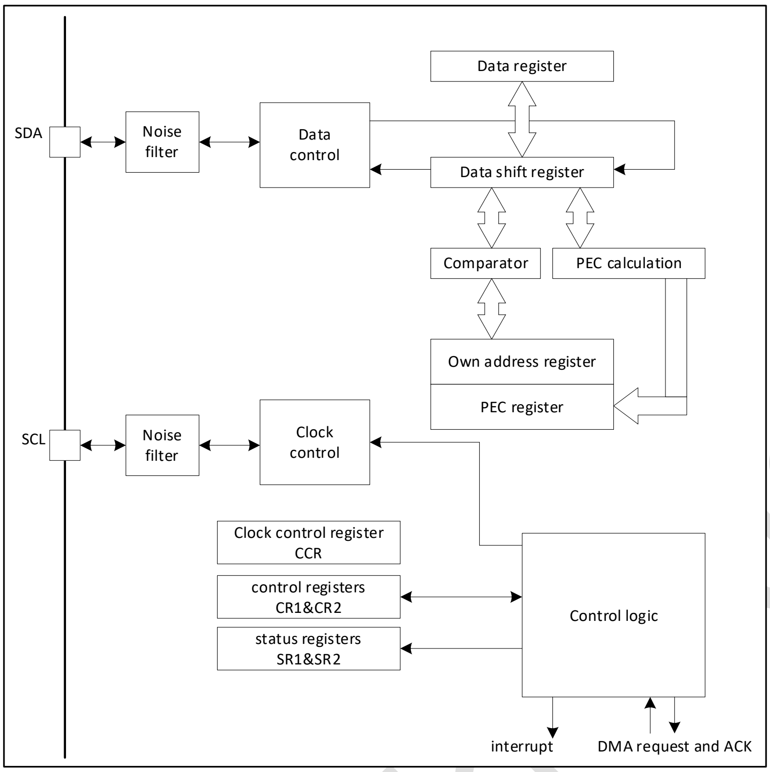


Figure 30-1 I2C block diagram

30.3.2. Mode selection

The interface can operate in one of the four following modes:

1) Slave transmitter

2) Slave receiver

3) Master transmitter

4) Master receiver

By default, it operates in slave mode. The interface automatically switches from slave to master when it generates a START condition, and from master to slave if an arbitration loss or a STOP generation occurs, allowing multimaster capability.

30.3.2.1. Communication flow

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a START condition and ends with a STOP condition. Both START and STOP conditions are generated in master mode by software.

In Slave mode, the interface is capable of recognizing its own addresses (7 or 10-bit), and the General Call address. The General Call address detection can be enabled or disabled by software. The reserved SMBus addresses can also be enabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the START condition contain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter. Refer to the following figure.

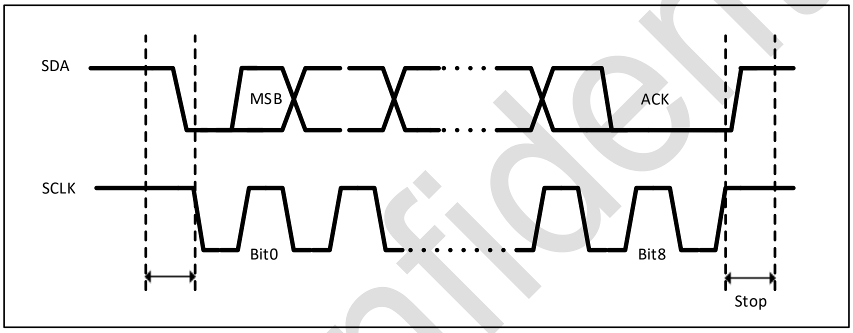


Figure 30-2 I2C bus protocol

Acknowledge can be enabled or disabled by software. The I2C interface addresses can be selected by software.

30.3.3. I2C initialization

30.3.3.1. Enabling and disabling the peripheral

The I2C peripheral clock must be configured and enabled the bit of I2C_EN in the RCC_APBENR1 register. Then the I2C can be enabled by setting the PE bit in the I2C_CR1 register.

30.3.3.2. I2C timings

The timings must be configured in order to guarantee a correct data hold and setup time, used in master and slave modes. This is done by programming the I2C_CCR and I2C_TRISE register.

30.3.4. I2C slave mode

By default, the I2C interface always works in slave mode. To switch from slave mode to master mode, a start condition needs to be generated.

In order to generate the correct timing, the input clock to this module must be programmed in the I2C_CR2 register. The frequency of the input clock must be at least:

Standard mode: 2 MHz

Fast mode: 4 MHz

If start condition is detected,the address received on the SDA line is sent to the shift register and is combined with the chip's address OAR1 or general The call address (if ENGC = 1) is compared.

Header or address mismatch:

The I2C interface ignores it and waits for another start condition.

Address match:

The I2C interface generates the following timings:

● If ACK is set to '1' by software, an acknowledge pulse is generated.

● The ADDR bit is set by hardware, and if the ITEVTEN bit is set, an interrupt is generated.

In slave mode, the TRA bit indicates that it is currently in receiver mode or transmitter mode.

30.3.4.1. Slave transmitter

After receiving the address and clearing the ADDR bit, (if the least significant bit of the address byte is 1) the Slave sends the data (byte) from the DR register to the SDA by the internal shift register.

Slave pulls SCL low until the ADDR bit is cleared and the data to be transmitted has been written to the DR register.

When an acknowledge pulse is received: The TxE bit is set by hardware and an interrupt is generated if the ITEVTEN and ITBUFEN bits are set.

If the TxE bit is set, but no new data is written to the I2C_DR register before the end of the next data transmission, the BTF bit is set. Slave pulls SCL low until the BTF bit is cleared by software (after reading I2C_SR1, then writing to the I2C_DR register).

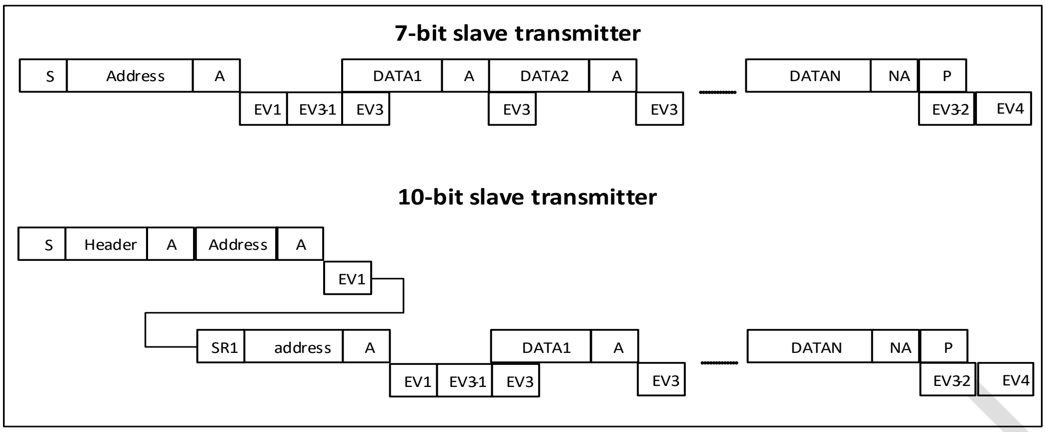


Figure 30-3 Transmission sequence diagram from the sender

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, NA = Non-acknowledge, EVx = Event (interrupt when ITEVFEN = 1)

EV1: ADDR = 1, by first reading the SR1 register, then reading the SR2 register to clear the ADDR bit

EV3-1: TxE = 1, shift register empty, data register empty, write Data1 to DR register

EV3: TxE = 1, shift register is not empty, data register is empty, write to DR register (Data2) to clear TxE

EV3-2: AF = 1, software write 0 to AF bit to clear this bit

EV4: STOPF=1, clearing this bit by reading the SR1 register first, then writing the CR1 register.

30.3.4.2. Slave receiver

After receiving the address and clearing ADDR, (if the least significant bit of the address byte is 0) the slave will store the byte received from the SDA line into the DR register by the internal shift register. The I2C interface performs the following actions after each byte is received:

- If the ACK bit is set, an acknowledge pulse is generated
- The hardware sets RxNE = 1. An interrupt is generated if the ITEVTEN and ITBUFEN bits are set.

If RxNE is set and the DR register is not read before the end of receiving new data, the BTF bit is set, and the slave keeps pulling SCL low until the BTF is cleared (the I2C_DR register is read after I2C_SR1). (See below).

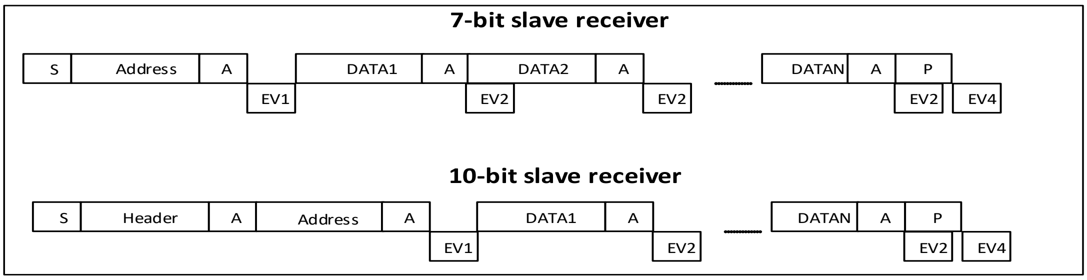


Figure 30-4 Slave receiver transmission sequence diagram

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledged, EVx = Event (with interrupt if ITEVFEN = 1)

EV1: ADDR = 1, by reading SR1 first, then SR2, the ADDR is cleared

EV2: RxNE = 1, read the DR register to clear this bit

EV4: STOPF = 1, clear this bit by first reading the SR1 register and then writing the CR1 register.

Note:

1) EV1 event pulls down SCL until the end of the corresponding software sequence.

2) EV2 software sequence must be completed before the current byte transfer is completed.

3) After checking the contents of the SR1 register, the user should perform a complete clearing sequence for each flag set that is found. For example, the ADDR and STOPF flags need to use the following sequence:

If ADDR = 1, read SR first, then read SR2,if STOPF = 1, read SR1 first, and then write CR1.

The purpose of this is to ensure that if both ADDR and STOPF are found to be set, they can both be cleared.

30.3.4.3. Close communication

After the last data byte is transmitted, the master generates a stop condition. When the slave detects this condition:

The hardware sets STOPF, and if the ITEVTEN bit is set, an interrupt is generated.

By first reading SR1 and then writing CR1, the STOPF bit is cleared. (See EV4 in the figure above).

30.3.5. I2C master mode

In Master mode, the I2C interface starts data transfer and generates a clock signal. Serial data transfers always begin with a START condition and end with a STOP condition.

When a START condition is generated on the bus via the START bit, the device enters master mode.

The following is the sequence of operations required for master mode:

- Set the input clock to the module in the I2C_CR2 register to generate the correct timing
- Configure the clock control register
- Configure the rise time register
- Program the I2C_CR1 register to start the peripheral
- Set the START bit in the I2C_CR1 register to 1 to generate a start condition

The input clock frequency to the I2C module must be at least:

- In standard mode: 2 MHz
- In fast mode: 4 MHz

30.3.5.1. The host generates clock

The CCR register generates high and low levels of SCL with rising or falling edges. Since the slave may stretch the SCL signal, after the rising edge of SCL occurs, the master checks the SCL signal from the bus when the time programmed in the TRISE register arrives.

- If SCL is low, it means that the slave is stretching the SCL bus, and the high-level counter stops counting until SCL is detected high. This is to ensure a minimum high time for the SCL parameter.
- If SCL is high, the high-level counter keeps counting.

In fact, even if the slave does not stretch SCL, it will take some time for such a feedback loop to occur from the rising edge of SCL to the detection of the rising edge of SCL. The time of this loop is related to the rise time of SCL (VIH data detection of SCL), plus the analog noise filtering of the SCL input path, and the SCL synchronization inside the chip due to the use of the APB clock. The maximum time for the feedback loop is programmed in the TRISE register, so the frequency of SCL remains stable regardless of the rise time of SCL.

30.3.5.2. Start condition

When BUSY = 0, set START = 1, the I2C interface will generate a Start condition and switch to master mode (MSL is set).

Note: Setting the START bit in master mode will generate a ReStart condition by hardware after the current byte is transmitted.

if Start condition is issued:

● The SB bit is set by hardware and an interrupt is generated if the ITEVTEN bit is set.

The master reads the SR1 register, and then writes the slave address to the DR register. (Transfer sequence EV5)

30.3.5.3. Slave address sending

The slave address is sent to the SDA line through the internal shift register.

- When in 10-bit address mode, sending a header sequence generates the following events.

– The ADDR10 bit is set in hardware and an interrupt is generated if the ITEVTEN bit is set.

The master device then waits for a read of the SR1 register, followed by a second address byte written to the DR set register.

The ADDR bit is set by hardware and an interrupt is generated if the TEVFEN bit is set.

The master device then waits for one read of the SR1 register, followed by a read of the SR2 register.

In 7-bit address mode, send out an address byte.

If the address byte is sent out

➢ The ADDR bit is set by hardware and an interrupt is generated if the ITEVTEN bit is set.

Then the Master reads the SR1 register, followed by the SR2 register.

According to the lowest bit of the sent slave address, the master decides to enter the transmitter mode or the receiver mode.

- In 7-bit address mode
  - To enter transmitter mode, the master device sets the least significant bit to '0' when sending the slave address.
  - To enter receiver mode, the master device sends the slave address with the least significant bit set to '1'.

The TRA bit indicates whether the master is in receiver mode or transmitter mode.

- When in 10-bit address mode.
  - To enter transmitter mode, the master device sends the header byte (11110xx0) first, then the slave address with LSB bit equal to 0. (The xx in the header byte is the highest 2 bits of the 10-bit address).
  - To enter receiver mode, the master sends the header byte (11110xx0) followed by the slave address with the LSB bit equal to 0. The master then sends the slave address with the LSB bit equal to 0. Then a start condition is re-sent followed by the header byte (11110xx1).

(The xx in the header byte is the highest 2 bits of the 10-bit address.) The TRA bit indicates whether the master device is in receiver mode or transmitter mode.

30.3.5.4. Master transmitter

After sending the address and clearing the ADDR bit, the master device sends the byte from the DR register to the SDA line through the internal shift register.

The Master waits until the first data byte is written to the DR register (see EV8_1).

When an ACK pulse is received, the TxE bit is set by hardware and an interrupt is generated if the INEVFEN and ITBUFEN bits are set.

If TxE is set and no new data byte is written to the DR register before the end of the last data transmission, BTF is set by hardware. The I2C interface will keep SCL low until the BTF is cleared (after reading I2C_SR1, then writing the I2C_DR register).

Closing Communication.

After writing the last byte in the DR register, a STOP condition is generated by setting the STOP bit (see EV8_2 in Figure), then the I2C interface will automatically return to slave mode (MLS bit cleared).

Note: When the TxE or BTF bit is set, the stop condition should be scheduled on the EV8_2 event.

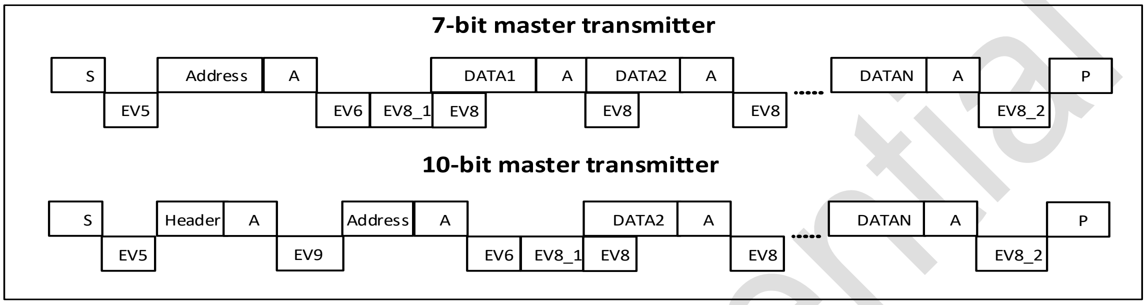


Figure 30-5 Master transmitter transmission sequence diagram

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event (with interrupt if ITEVFEN = 1)

EV5: SB = 1, by reading SR1, and then writing data to the DR register, the bit is cleared

EV6: ADDR = 1, by reading SR1, and then reading SR2, the bit is cleared

EV8_1: TxE = 1, shift register is empty, data register is empty, write Data1 to DR register

EV8: TxE = 1, shift register is not empty, data register is empty, write Data2 to DR register, this bit is cleared

EV8_2: TxE = 1, BTF = 1, Write the Stop bit register, when the hardware sends the Stop bit, TxE and BTF are cleared

EV9: ADDR10=1,read SR1 then write DR register to clear the event.

Note:

1) EV5, EV6, EV8_1 and EV8_2 events, stretch the low level of SCL until the corresponding software sequence execution ends.

2) EV8 software The sequence must complete before the current byte is sent. If the EV8 software sequence cannot be completed before the end of the currently transmitted byte, it is recommended to use BTF instead of TxE, which has the disadvantage of slowing down the communication.

30.3.5.5. Master receiver

After sending the address and clearing the ADDR, the I2C interface enters the master receiver mode. In this mode, the I2C interface receives data bytes from the SDA line and sends them to the DR register through the internal shift register. After each byte, the I2C interface performs the following operations in sequence:

- If the ACK bit is set, issue an acknowledge pulse.
- The hardware sets RxNE = 1, if the INEVFEN and ITBUFEN bits are set, an interrupt will be generated

If the RxNE bit is set and the data in the DR register is not read before the end of receiving new data, the hardware will set BTF = 1, and the I2C interface will keep SCL low before clearing BTF, after reading I2C_SR1 Reading the I2C_DR register again will clear the BTF bit.

Close communication:

Method 1: The application scenario of this method is: when I2C is set as the highest priority interrupt in the application

The Master sends a NACK after receiving the last byte from the Slave. After receiving the NACK, the Slave releases control of the SCL and SDA lines. The Master can then send a Stop/Restart condition.

1) In order to generate a NACK pulse after the last byte is received, the ACK bit must be cleared after reading the second-to-last data byte (after the second-to-last RxNE event).

2) To generate a STOP/RESTART condition, software must set the STOP/START bit after reading the second-to-last data byte (after the second-to-last RxNE event).

3) When a single byte is received, the closing acknowledge and stop conditions should be generated just after EV6 (EV6_1, after clearing ADDR). After a stop condition is generated, the I2C interface automatically returns to slave mode (MSL bit is cleared).

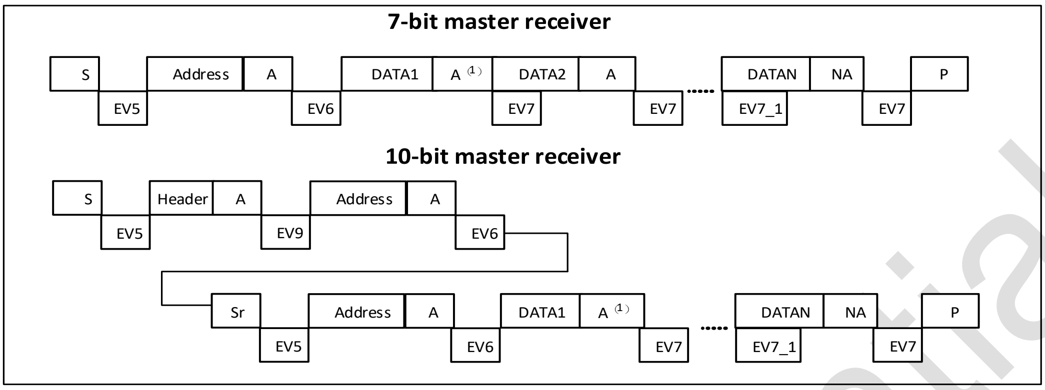


Figure 30-6 Method 1: Timing when master mode transmits

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, read SR1, then write DR register, this bit is cleared

EV6:ADDR = 1, read SR1, then read SR2, this bit is cleared

EV6_1: No related flag event, only used for 1 byte reception.

EV7: RxNE = 1, read DR register, this bit is cleared

EV7_1: RxNE = 1, read DR register, write ACK = 0 and set STOP

EV9: ADDR10=1, read SR1 then write DR register to clear the event.

1) If a single byte is received, the above marked as (1) The place will be NA

2) EV5, EV6 events, stretch the low level of SCL until the corresponding software sequence execution ends

3) EV7 software sequence must be executed before the current byte is sent. In EV7, the software sequence cannot be managed until the currently transferred byte has been transferred. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication.

4) The software sequence of EV6_1 or EV7_1 must be completed before the ACK of the current byte transmission.

Method 2: The application scenario of this method is: the I2C interrupt is not the highest priority in the application, or the query method.

Use this method, DataN-2 is not read, so after DataN-1, the communication is stretched (RxNE and BTF is set). Then, before reading DataN-2 of the DR register, clear the ACK bit to ensure that it is cleared before DataN ACKs. After this, after reading DataN-2, set the STOP/START bit and read DataN-1. After RxNE is set, read DataN

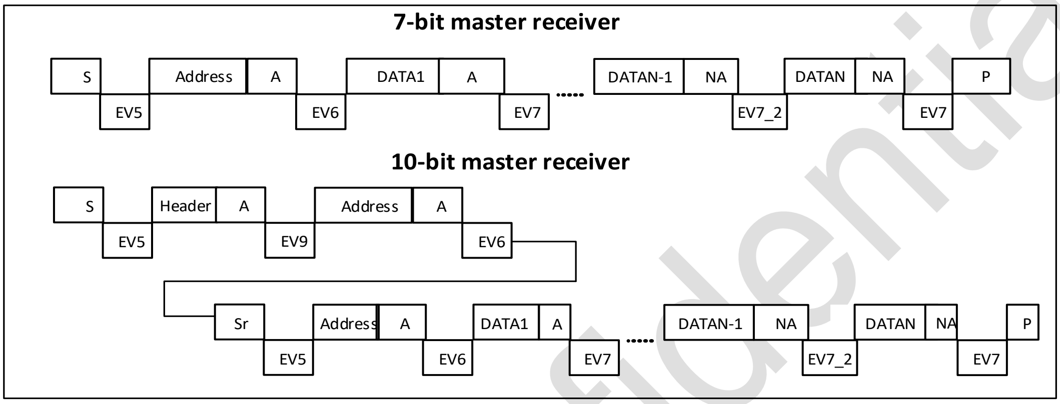


Figure 30-7 Method 2: Timing for master mode transmission when N > 2

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6: ADDR, first read SR1, then read SR2, clear this bit

EV7: RxNE = 1, read DR register to clear this bit

EV7_2: BTF = 1, DataN-2 is stored in DR register, DataN-1 is stored in shift register, write ACK = 0, read DataN-2 in the DR register. Set STOP, read DataN-1

EV9: ADDR10=1, read SR1 then write DR register to clear the event.

Note:

- EV5, EV6 events, stretch the low level of SCL until the corresponding software sequence execution ends.
- EV7 software sequence must be executed before the completion of the current byte transmission. In EV7, the software sequence cannot be managed until the currently transferred byte has been

transferred. It is recommended to use BTF instead of RXNE, which has the disadvantage of slowing down communication.

When 3 bytes are to be read:

- RxNE = 1 = > Nothing (DataN-2 not read).
- DataN-1 received
- BTF = 1, shift and data registers are full: DR register stores DataN-2, shift register stores DataN-1 = > SCL is pulled low: there is no other data to be received on the bus
- Clear the ACK bit
- Read DataN-2 in DR register = > This will start the reception of DataN in shift register
- DataN reception complete (with a NACK)
- Write START or STOP bit
- Read DataN-1
- RxNE = 1
- Read DataN

The above process is a description for N > 2. The reception of 1 byte and 2 bytes uses different processing methods, see the following description:

- **In the case of 2 bytes reception**

1. Set the POS and ACK bits
2. Wait for ADDR to be set
3. Clear the ADDR bit
4. Clear the ACK bit
5. Wait for BTF to be set
6. Write STOP bit
7. Read DR twice

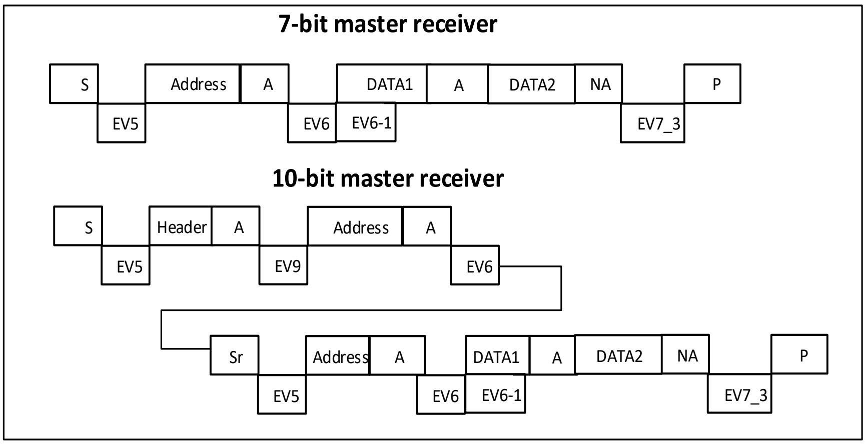


Figure 30-8 Method 2: timing when master mode transmits when N = 2

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event (with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6: ADDR = 1, first read the SR1 register, then read the SR2 register, clear the ADDR bit

EV6_1: no related flag events. After EV6, that is, after the address is cleared, ACK should be cleared

EV7_3: BTF = 1, write STOP = 1, then read DR twice (Data1 and Data2)

EV9: ADDR10=1, read SR1 then write DR register to clear the event

Note:

- EV5, EV6 events, stretch SCL
- The software sequence of EV6_1 must be completed before the ACK of the current byte transmission
- **In the case of single byte reception**
  1. In ADDR event, clear the ACK bit
  2. Clear ADDR
  3. Write STOP or START bit
  4. Read data after RxNE flag is set

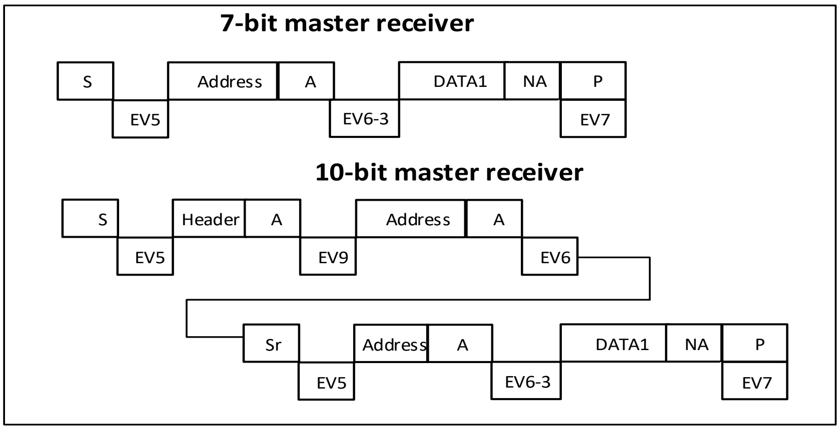


Figure 30-9 Method 2: timing when master mode transmits when N = 1

Legend: S = Start, Sr = Repeated Start, P = Stop, A = Acknowledge, EVx = Event(with interrupt if ITEVFEN = 1)

EV5: SB = 1, first read SR1 register, then write DR register, clear this bit

EV6_3: ADDR = 1, write ACK = 0. First read the SR1 register, then read the SR2 register, clear the ADDR bit. After ADDR is cleared, write STOP = 1

EV7: RxNE = 1, read DR register to clear this bit

EV9: ADDR10=1, read SR1 then write DR register to clear the event.

Note:

EV5 event will stretch the low level of SCL until the corresponding software sequence execution ends.

30.3.6. Error stage

30.3.6.1. Bus error(BERR)

A bus error is generated when the I2C interface detects an external stop or start condition during an address or data byte transfer. at this time:

- The BERR bit is set to '1', if the ITERREN bit is set, an interrupt is generated
- In slave mode: the data is discarded, the hardware releases the bus:
  - If it is a wrong Start condition, the slave considers a Restart and waits for an address or a stop condition

- If it is a wrong Stop condition, the slave operates according to the normal stop condition, and the hardware releases the bus at the same time

- In master mode: the hardware does not release the bus, and does not affect the current transmission status. At this point it is up to the software to decide whether to abort the current transfer.

30.3.6.2. ACK Failed(AF)

An acknowledge error occurs when the interface detects a no acknowledge bit. at this time:

- The AF bit is set, and an interrupt is generated if the ITERREN bit is set
- When the transmitter receives a NACK, the communication must be reset:
  - If it is in slave mode, the hardware releases the bus.
  - If in master mode, software must generate a stop condition or repeated start.

30.3.6.3. Arbitration loss (ARLO)

Arbitration loss error occurs when I2C interface detects arbitration loss, at this time:

1. The ARLO bit is set by hardware and an interrupt is generated if the ITERREN bit is set
2. The I2C interface automatically returns to slave mode (MSL bit is cleared). When the I2C interface loses arbitration, it cannot respond to its slave address in the same transfer, but it can respond after the master that wins the bus sends a repeated start condition
3. Hardware release bus

30.3.6.4. Overrun/Underrun error (OVR)

In slave mode, if the clock extension is disabled and the I2C interface is receiving data, when it has received a byte (RxNE = 1), but the previous byte data in the DR register has not been read, an overrun occurs. mistake.

at this time:

1. The last received data is discarded

2. On overrun error, software should clear the RxNE bit and the transmitter should resend the last transmitted byte

In slave mode, if the clock extension is disabled and the I2C interface is sending data, before the clock of the next byte arrives, the new data has not been written to the DR register (TxE = 1), an underrun error occurs. at this time:

- The previous byte in the DR register will be repeated
- The user should make sure that when an underrun error occurs, the receiver should discard the repeatedly received data. The transmitter should update the DR register at the specified time according to the I2C bus standard

When sending the first byte, the DR register must be written to after clearing ADDR and before the first rising edge of SCL, if this is not possible, the receiver should discard the first data.

30.3.7. SDA/SCL control

- If clock stretching is allowed:
  - Transmitter mode: If TxE = 1 and BTF = 1: The I2C interface keeps the clock line low before transmission, waiting for software to read SR1, and then write the data into the data register (DR and shift registers are both empty).
  - Receiver mode: if RxNE = 1 and BTF = 1: I2C interface keeps clock line low after receiving data byte, waiting for software to read SR1, then read data register DR (DR and shift registers are both full).
- If clock stretching is disabled in slave mode:
  - If RxNE = 1, the DR has not been read before the next byte is received, an overun occurs. The last byte received is lost.
  - If TxE = 1, an underrun occurs when no new data is written into the DR before the next byte must be sent. The same bytes will be sent repeatedly.
  - Hardware does not implement control of write collisions.

30.3.8. DMA requests

DMA requests (when enabled) are only used for data transfers. When the data register becomes empty when sending, or the data register becomes full when receiving, a DMA request is generated. DMA must be initialized and enabled before the end of the current byte transfer. The DMAEN bit (in the I2C_CR2 register) must be enabled before an ADDR event occurs.

In master or slave mode, when clock stretching is enabled, the DMAEN bit can be set during an ADDR event before clearing ADDR. A DMA request must be responded to before the current byte transfer is complete. When the length of the DMA transfer data reaches the value set by the DMA, the DMA controller sends EOT (End of transfer) to the 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 disabled, and then the stop condition is 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 sends a hardware signal EOT_1, which corresponds to DMA transmission (byte number - 1). If the LAST bit is set in the I2C_CR2 register, the hardware will automatically send a NACK after sending the next byte after EOT_1. With the interrupt enabled, the user can generate a stop condition in the interrupt service routine when the DMA transfer is complete.

30.3.8.1. Transmission using DMA

DMA mode can be enabled by setting the DMAEN bit in the I2C_CR2 register. As long as the TxE bit is set, the data will be loaded into the I2C_DR register from the preset memory area by DMA. To assign a DMA channel to I2C, 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 registers. Data is transferred from this bank to I2C_DR after each TxE event.

3. Set the desired number of bytes to transfer in the DMA_CNDTRx registers. This value will be decremented after each TxE event.

4. Configure the channel priority using the PL[0:1] bits in the DMA_CCRx register.

5. Set the DIR bit in the DMA_CCRx register, and can configure the interrupt request when the entire transfer is half or fully completed according to the application requirements.

6. Activate the channel by setting the EN bit on the DMA_CCTx register.

When the number of data transfers set in the DMA controller has been completed, the DMA controller sends an EOT/EOT_1 signal that the transfer is over to the I2C interface. When interrupts are enabled, a DMA interrupt will be generated.

Note: Do not set the ITBUFEN bit in the I2C_CR2 register if using DMA for transmission.

30.3.8.2. Reception using DMA

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 in the I2C_DR register will be transferred to the set memory area by the DMA (refer to the DMA description). To set up a DMA channel for I2C reception, 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 memory area after each RxNE event.

2. Set the bank address in the DMA_CMARx registers. Data will be transferred from the I2C_DR register to this bank after each RxNE event.

3. Set the desired number of bytes to transfer in the DMA_CNDTRx registers. This value will be decremented after each RxNE event.

4. Configure the channel priority using PL[0:1] in the DMA_CCRx register.

5. Clear the DIR bit in the DMA_CCRx register. According to the application requirements, it can be set to issue an interrupt request when half or all of the data transfer is completed.

6. Set the EN bit in the DMA_CCRx register to activate the channel.

When the number of data transfers set in the DMA controller has been completed, the DMA controller sends an EOT/EOT_1 signal that the transfer is over to the I2C interface. When interrupts are enabled, a DMA interrupt will be generated.

Note: If using DMA for reception, do not set the I2C_CR2 register's ITBUFEN bit.

30.3.9. SMBus

The System Management Bus (SMBus) is a two-wire interface. Through it, devices can communicate with each other and with other parts of the system. It is based on the I2C operating principle and SMBus provides a control bus for system and power management related tasks. A system using SMBus can pass information to and from several devices without using separate control lines.

The System Management Bus (SMBus) standard involves three types of devices. Slave devices, which receive or respond to commands. Master devices, which are used to issue commands, generate clocks and terminate transmissions. The host, a dedicated master device that provides the master interface to the system CPU. The host must have master-slave functionality and must support the SMBus notification protocol. Only one host is allowed in a system.

The I2C1 module in this project supports SMbus/PMbus functionality.

30.3.9.1. Similarities between SMBus and I2C

- 2 lines of bus protocol (1 clock, 1 data) + optional SMBus reminder line
- Master-slave communication, master device provides clock
- Multi-host functionality
- SMBus data format similar to I2C 7-bit address format.

30.3.9.2. Differences between SMBus and I2C

The following table shows the differences between SMBus and I2C:

Table 30-1 Comparison between SMBus and I2C

SMBus	I2C
Maximum transmission speed 100kHz	Maximum transmission speed 400kHz
Minimum transmission speed 10kHz	No minimum transmission speed
35ms clock low timeout	No clock timeout
Fixed logic level	Logic level determined by VDD
Different address types (reserved, dynamic, etc.)	7-bit, 10-bit and broadcast call slave address types
Different bus protocols (fast command, call handling, etc.)	No bus protocol

30.3.9.3. SMBus applications

Using the system management bus, the device can provide manufacturer information, tell the system its model/part number, save the status of suspended events, report different types of errors, receive control parameters, and return its status. The SMBus provides a control bus for system and power management related tasks.

30.3.9.4. Address Resolution Protocol (ARP)

SMBus slave address conflicts can be resolved by dynamically assigning a new unique address to each slave device.

ARP has the following properties:

1. address allocation utilises the standard SMBus physical layer arbitration mechanism
2. the allocated address remains unchanged while the device maintains power, allowing the device to retain its address in the event of a power failure
3. there is no additional SMBus packing overhead after the address has been allocated (i.e. accessing a device with an allocated address takes the same amount of time as accessing a device with a fixed address)
4. Any SMBus master device can traverse the bus.

30.3.9.5. SMBus Alert Mode (ALERT)

SMBus alert is an optional signal with an interrupt line for devices that wish to extend their control capability at the expense of a pin.SMBALERT is a line-and-signal like the SCL and SDA signals.SMBALERT is normally used in conjunction with the SMBus broadcast call address. The message associated with SMBus is 2 bytes.

A single slave device can signal to the host that it wishes to communicate via SMBALERT, which can be achieved by setting the ALERT bit on the I2C_CR1 register. The host handles the interrupt and accesses all SMBALERT devices via the Alert Response Address ARA (address 0001100x). This status is identified by the SMBALERT status flag in the I2C_SR1 register. The host performs a modified receive byte operation. The 7-bit device address provided by the sender device is placed on the 7 highest bits of the byte, the eighth bit can be 0 or 1.

If multiple devices pull SMBALERT low, the highest priority device (the smallest address) will win the right to communicate during the address transfer via standard arbitration. After acknowledging the slave address, this device must not pull its SMBALERT low again. if the host still sees SMBALERT low when the message transfer is complete, it knows it needs to read the ARA again. hosts that do not perform the SMBALERT signal may periodically access the ARA. for more detailed information on the SMBus reminder mode, please refer to the SMBus specification in version 2.0.

30.3.9.6. Bus protocols

The SMBus specification supports nine bus protocols. For detailed information on these protocols and the SMBus address types, please refer to the SMBus specification version 2.0. These protocols are implemented by the user's software.

30.3.9.7. Timeout error (TIMEOUT)

There are many differences between I2C and SMBus in terms of timing specifications.

SMBus defines a clock low timeout of 35 ms. SMBus specifies TLOW:SEXT as the cumulative clock low extension time for slave devices. sMBus specifies TLOW:MEXT as the cumulative clock low extension time for master devices. Please refer to the SMBus specification in version 2.0 for more details on timeouts.

The status flag Timeout or Tlow error in I2C_SR1 indicates the status of this feature.

30.3.9.8. PMBus

PMbus is based on SMbus and the transmission logic is identical to SMbus, the difference being that PMbus defines some functions related to power management (done by software).

30.4. I2C interrupts

Table 30-2 Table 30-3 I2C interrupt requests

Interrupt event	Event flag	Interrupt enable control bit
START has been sent (Master)	SB	ITEVTEN
Address has been sent(Master) / Address matched(Slave)	ADDR	
The 10-bit header has been sent	ADDR10	
Stop detection interrupt flag(Slave)	STOPF	
Transfer Complete Reload	BTF	
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error	BERR	ITERREN
Arbitration loss(Master)	ARLO	
ACK Failed	AF	
Overrun/Underrun	OVR	
PEC error	PECERR	

30.5. I2C registers

Registers can be accessed half-word or word.

30.5.1. I2C 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
SWR ST	Re s.	ALE RT	PE C	PO S	AC K	ST OP	STA RT	NO STRET CH	EN GC	ENP EC	ENA RP	SMB- TYPE	Re s.	SMB US	P E
RW		RW	R W	R W	R W	RW	RW	RW	RW	RW	RW	RW		RW	R W

Bit	Name	R/W	Reset Value	Function
15	SWRST	RW	0	Software reset. When set, I2C is in reset state. Before the reset release, make sure that the I2C pins are re-leased and the bus is in an idle state. 0: I2C module is not in reset state 1: I2C module is in reset state Note: This bit can be used to reinitialize I2C in error or locked state. If the BUSY bit is 1, no stop condition is detected on the bus.
14	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
13	ALERT	RW	0	SMBus reminder. 0: Release the SMBAlert pin to make it high. Reminder response address header immediately following the NACK signal; 1: Drive the SMBAlert pin to make it low. Reminder response address header immediately following the ACK signal; Cleared by hardware when PE=0.
12	PEC	RW	0	Packet error checking software can set and clear this bit, hardware can clear this bit in the following cases: when a PEC has been transmitted, a start condition, or a stop condition, or when PE = 0; 0: no PEC transmission 1: PEC transmission (in transmit or receive mode) Note: The calculation of the PEC is invalidated when arbitration is lost.
11	POS	RW	0	ACK/PEC position (for data reception), this register can be set/cleared by software, or cleared by hardware when PE = 0. 0: The ACK bit controls the (N)ACK of the byte currently being received in the shift register. The PEC bit indicates that the byte in the current shift register is PEC 1: The ACK bit controls the (N)ACK of the next byte received in the shift register. The PEC bit indicates that the next byte received in the shift register is a PEC Note: The POS bit can only be used in a 2-byte receive configuration and must be configured before receiving data. In order to NACK the 2nd byte, the ACK bit must be cleared after clearing ADDR. In order to detect the PEC of the second byte, the PEC bit must be set during the ADDR stretch event after the POS bit is configured.
10	ACK	RW	0	Acknowledgment enable. This register can be set/cleared by software, or cleared by hardware when PE = 0. 0: no response returned

Bit	Name	R/W	Reset Value	Function
				1: Return an acknowledgment after a byte has been received. (matching address or data)
9	STOP	RW	0	When a stop condition occurs, software can set/clear this register, or when a stop condition is detected, it is cleared by hardware, when a timeout error is detected, hardware is set. In Master Mode: 0: No stop generation 1: Generate a stop condition after the current byte transfer or after the current start condition is issued In Slave Mode: 0: No stop condition is generated 1: Release the SCL and SDA lines after the cur-rent byte transfer
8	START	RW	0	The starting condition is generated. This register can be set/cleared by software, or cleared by hardware when a START condition is issued or when PE = 0. Master mode: 0: No start generation 1: Restart/Start condition Slave Mode: 0: No start generation 1: When the bus is idle, generate a start genera-tion (and automatically switch to Master Mode by hardware)
7	NOSTRETCH	RW	0	Clock stretching (Slave) is disabled. When the ADDR or BTF flag is set, this bit is used by the slave to disable clock stretching until reset by software. 0: allow clock stretching 1: Disable clock stretching
6	ENGC	RW	0	General call enabled. 0: Disable general call. Respond to address 00h with NACK 1: Allow general calls. Responds to address 00h with an ACK
5	ENPEC	RW	0	PEC enable. 0: PEC calculation disabled 1: PEC calculation enabled

Bit	Name	R/W	Reset Value	Function
4	ENARP	RW	0	ARP enable. 0: ARP disabled; 1: ARP enabled; If SMBTYPE=0, the default address of the SMBus device is used; If SMBTYPE=1, the primary address of the SMBus is used.
3	SMBTYPE	RW	0	SMBus type. 0: SMBus device; 1: SMBus host;
2	Reserved	-	-	Reserved
1	SMBUS	RW	0	SMBus mode. 0: I2C mode; 1: SMBus mode;
0	PE	RW	0	I2C module enabled. 0: Disable 1: I2C enable Note: If a communication is in progress when this bit is cleared, after the current communication ends, the I2C module is disabled and returns to the idle state. Since PE = 0 after the communication ends, all bits are cleared. In master mode, this bit must never be cleared until the communication has ended.

30.5.2. I2C 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	Res	FREQ[5:0]					
			RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:13	Reserved	RES	-	Reserved

Bit	Name	R/W	Reset Value	Function
12	LAST	RW	0	DMA last transfer. 0: EOT of next DMA is not the last transfer 1: The EOT of the next DMA is the last transfer Note: This bit is used in master receive mode to allow a NACK to be generated on the last re-ceived data.
11	DMAEN	RW	0	DMA request enabled. 0: Disable DMA request, 1: DMA requests are enabled when TxE = 1 or RxNE = 1.
10	ITBUFEN	RW	0	Buffer interrupt enable. 0: No interrupt is generated when TxE = 1 or RxNE = 1 1: When TxE = 1 or RxNE = 1, an event interrupt is generated (regardless of the value of DMAEN)
9	ITEVTEN	RW	0	Event interrupt enable. 0: Disable 1: Enable event interrupt This interrupt will be generated under thefollow-ing conditions: 1. SB = 1 (main mode), 2. ADDR = 1 (Master/Slave mode) 3. STOPF = 1 (slave mode) 4. BTF = 1, but no TxE or RxNE events 5. If ITBUFFEN = 1, TxE event is 1 6. If ITBUFEN = 1, the RxNE event is 1
8	ITERREN	RW	0	Error interrupt enable. 0: Disable error interrupt, 1: Enable error interrupt, This interrupt will be generated under the follow-ing conditions: ✓ BERR = 1 ✓ ARLO = 1 ✓ AF = 1 ✓ OVR = 1 ✓ PECERR = 1
7:6	Reserved	RES	-	Reserved
5:0	FREQ	RW	0	I2C module clock frequency. This register must be configured with the value of the APB clock frequency to generate data setup and hold times that are compatible with the I2C protocol.

Bit	Name	R/W	Reset Value	Function
				The minimum allowable frequency that can be set is 4 MHz (standard mode, ie 100 k), 12 MHz (400 k), and the maximum frequency is the highest APB clock frequency of the chip. 000000: Forbidden 000001: Forbidden 000100: 4 MHz … 100100: 36 MHz … 110000: 48 MHz Greater than 100100: Forbidden.

30.5.3. I2C own address register 1 (I2C_OAR1)

Address offset:0x08

Reset value:0x4000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD-MODE	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
15	ADDMODE	RW	0	Addressing mode (slave mode). 0: 7-bit slave address (does not respond to a 10-bit address); 1: 10-bit slave address (not responding to a 7-bit address);
14:10	Reserved	-	-	Reserved
9:8	ADD[9:8]	RW	0	Interface address. 7-bit address mode This register is not relevant. 10-bit address mode Bits 9 to 8 of the bit address.
7:1	ADD[7:1]	RW	0	Bit 7:1 of address
0	Reserved	-	-	Reserved

30.5.4. I2C own address register 2 (I2C_OAR2)

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
15:8	Reserved	-	-	Reserved
7:1	ADD2[7:1]	RW	0	Bits 7 to 1 of the interface address. Bits 7~1 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 recognised; 1: In 7-bit address mode, both OAR1 and OAR2 are recognised.

30.5.5. I2C 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
15:8	Reserved	RES	-	Reserved

7:0	DR[7:0]	RW	0	8-bit data register, two independent buffers inside the chip share an address, which are used to store the received data (RX_DR) and place the data to be sent to the bus (TX_DR). **Transmitter Mode:** Data transfer is automatically started when a byte is written to the DR register (actually written to TX_DR). Once the transmission starts (TxE = 1), if the next data to be transmitted can be written into the DR register in time, the I2C module will maintain a continuous data flow. **Receiver Mode:** The received byte is copied to the DR register (actually RX_DR) (RxNE = 1). Read out the data register before receiving the next byte (RxNE = 1) to realize continuous data reception. Note: 1) In slave mode, the address will not be copied into the data register DR 2) The hardware does not handle write conflicts (if TxE = 0, the data register can still be written) 3) If the ARLO event occurs while processing the ACK pulse, the received byte will not be copied to the data register, so it cannot be read

30.5.6. I2C stage 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
15	SMBALERT	RC_W0	0	SMBus alert status. SMBus host mode: 0: no SMBus alert; 1: SMBAlert alert event generated at the pin; SMBus slave mode: 0: no SMBAlert response address header sequence;

Bit	Name	R/W	Reset Value	Function
				1: SMBAlert response address header sequence received until SMBAlert goes low. This bit is cleared by software writing 0 or by hardware when PE=0.
14	TIMEOUT	RC_W0	0	Timeout or Tlow error. 0: no timeout error; 1: the duration for which SCL is low has reached 25ms (timeout); or the cumulative clock extension time of the master low exceeds 10ms (Tlow:next); or the cumulative clock extension time of the slave low exceeds 25ms (Tlow:sext). When this bit is set in slave mode: the slave device resets communication and the hardware releases the bus; when this bit is set in master mode: hardware issues a stop condition; This bit is cleared by a software write 0, or by hardware when PE=0. Note: This function is only available in SMBUS mode.
13	Reserved	-	-	Reserved
12	PECERR	RC_W0	0	A PEC error occurred while receiving. 0: No PEC error, return ACK after receiving PEC (if ACK = 1) 1: There is a PEC error, and NACK is returned after receiving PEC (regardless of the value of ACK) This bit is cleared by software by writing 0, or by hardware when PE = 0
11	OVR	RC_W0	0	Overload/Underload flag. 0: no overload/underload, 1: Overload/underload occurred. When NOSTRETCH = 1, this bit is set by hardware in slave mode, In the receive mode, when a new byte is received (including the ACK response pulse), and the contents of the data register have not been read out, the newly received byte will be lost. In transmit mode when a new byte is to be sent, but no new data is written to the data register, the same byte will be sent twice.

Bit	Name	R/W	Reset Value	Function
				This bit is cleared by software by writing 0, or by hardware when PE = 0. Note: If a write to the data register occurs very close to the rising edge of SCL, the transmitted data is indeterminate and a hold time error occurs.
10	AF	RC_W0	0	Reply failure flag. 0: no response failed, 1: Reply failed. Hardware will set this register when no acknowledgement is returned. This bit is cleared by software by writing 0, or by hardware when PE = 0
9	ARLO	RC_W0	0	Arbitration lost (main mode). 0: no arbitration loss is detected. 1: Arbitration loss detected. This register is set by hardware when the interface loses control of the bus to another host. This bit is cleared by software by writing 0, or by hardware when PE = 0. After an ARLO event, the I2C interface automatically switches back to slave mode (M/SL = 0).
8	BERR	RC_W0	0	Bus error flag. 0: No start or stop condition error. 1: Error in start or stop condition. This bit is set by hardware when the interface detects a false start or stop condition. This bit is cleared by software by writing 0, or by hardware when PE = 0.
7	TxE	R	0	The data register is empty (when transmitting) flag. 0: The data register is not empty. 1: The data register is empty. When sending data, this bit is set to 1 when the data register is empty, and this bit is not set during the sending address phase. This bit can be cleared by software writing data to the DR register, or automatically by hardware after a START or STOP condition occurs, or when PE = 0.

Bit	Name	R/W	Reset Value	Function
				This bit is not set if a NACK is received, or if the next byte to be sent is PEC (PEC = 1). Note: After writing the first data to be sent, or writing data when BTF is set, the TxE bit cannot be cleared, because the data register is empty at this time.
6	RxNE	R	0	Data register not empty (on reception) flag. 0: The data register is empty. 1: The data register is not empty. On reception, this register is set when the data register is not empty. During the receive address phase, this register is not set. This register is cleared by software reads and writes to the data register, or by hardware when PE = 0. Note: When BTF is set, reading data does not clear the RxNE bit because the data register is still full.
5	Reserved	-	-	Reserved
4	STOPF	R	0	Stop condition detection bit (slave mode). 0: No stop bit detected. 1: Stop condition detected. After an acknowledgment (if ACK = 1), the hardware sets this bit to 1 when the slave device detects a stop condition on the bus. After the software reads the I2C_SR1 register, a write to the I2C_CR1 register will clear this bit, or when PE = 0, the hardware will clear this bit. Note: The STOPF bit is not set after a NACK is received.
3	ADD10	R	0	10-bit address header sequence has been sent (master mode). 0: No ADD10 event has occurred. 1: The master device has sent the first address byte out. In 10-bit address mode, when the master device will have sent the first byte out, the hardware sets the location 1. A write operation to the I2C_DR register will clear this bit after the software reads the I2C_SR1 register, or when PE=0, the hardware clears this bit.

Bit	Name	R/W	Reset Value	Function
				Main: This register is not set after a NACK is received.
2	BTF	R	0	End of byte transfer flag. 0: Byte transfer not complete 1: Byte transfer ended successfully The hardware will set this register in the following cases (when slave mode, NOSTRETCH = 0, master mode, regardless of NOSTRETCH): 1. On reception, when a new byte is received (including the ACK pulse) and the data register has not been read (RxNE = 1). 2. When transmitting, when a new data should be transmitted and the data register has not been written with new data (TxE = 1). This bit is cleared by a read or write to the data register after software reads the I2C_SR1 register, or by hardware after sending a start or stop condition, or when PE = 0. Note: After receiving a NACK, the BTF bit is not set.
1	ADDR	R	0	Address has been sent (Master mode)/Address matched (Slave mode). After the software reads the I2C_SR1 register, reading the I2C_SR2 register will clear this bit, when PE = 0, it will be cleared by hardware. Address matching (Slave): 0: The address does not match or the address was not received, 1: The received address matches. Hardware will set this bit when the received slave address matches the OAR register or general call address. Note: In slave mode, it is recommended to perform a complete clearing sequence, that is, after ADDR is set, read the SR1 register first, and then read the SR2 register. Address has been sent (Master): 0: Address sending has not ended, 1: Address sending ends. For a 7-bit address, it is set when the ACK byte is received.

Bit	Name	R/W	Reset Value	Function
				Note: This register will not be set after receiving a NACK.
0	SB	R	0	Start bit flag (master mode). 0: start condition not sent, 1: The start condition has been sent, — This register is set when a START condition is sent. — After the software reads the I2C_SR1 register, a write to the data register will clear this bit, or when PE = 0, it will be cleared by hardware.

30.5.7. I2C stage registor 2 (I2C_SR2)

Address offset: 0x18

Reset value: 0x0000

Note: Even if the ADDR flag is set after reading the I2C_SR1 register, reading the I2C_SR2 register after reading I2C_SR1 will clear the ADDR flag. Therefore, the I2C_SR2 register must be read only when the ADDR bit of the I2C_SR1 register is found to be set or the STOPF bit is cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUALF	SMBHOST	SMBDEF-AULT	GENCALL	Res.	TRA	BUSY	MSL
R	R	R	R	R	R	R	R	R	R	R	R		R	R	R

Bit	Name	R/W	Reset Value	Function
15:8	PEC	R	0	Packet error detection register. When ENPEC=1, this register holds the value of the internal PEC.
7	DUALF	R	0	Dual address flag (slave mode) . 0: the received address matches OAR1; 1: Received address matches OAR2. This register is cleared by hardware when a stop condition or a repeated start condition is generated, or when PE=0.
6	SMBHOST	R	0	SMBus host header sequence received (slave mode) flag. 0: SMBus host address not received;

Bit	Name	R/W	Reset Value	Function
				1: SMBus host address received when SMB-TYPE=1 and ENARP=1. Hardware clears this register when a stop condition or a repeating start condition is generated, or when PE=0.
5	SMBDEFAULT	R	0	SMBus slave device default address (slave mode). 0: default address of SMBus device not received; 1: the default address of the SMBus device is received when ENARP=1. Hardware clears this register when a stop condition or a repeating start condition is generated, or when PE=0.
4	GENCALL	R	0	General call address (slave mode). 0: No broadcast call address received, 1: When ENGC = 1, the address of the general call is received. Hardware clears this register when a STOP condition or a repeated START condition occurs, or when PE = 0.
3	Reserved	-	-	Reserved
2	TRA	R	0	Send/receive flag. 0: data received 1: Data has been sent At the end of the entire address transfer phase, this register is set according to the R/W bit of the address byte. Hardware clears this register when a STOP condition is detected (STOPF = 1), or a repeated-START condition, or bus arbitration is lost (ARLO = 1), or when PE = 0.
1	BUSY	R	0	Bus busy flag. 0: No data communication on the bus 1: Polarity data communication is in progress on the bus Set by hardware when SDA or SCL is detected low. Hardware clears when a stop condition is detected.

Bit	Name	R/W	Reset Value	Function
				This register indicates the current bus communication in progress, this information is still updated when the interface is disabled (PE = 0).
0	MSL	R	0	Master-slave mode. 0: slave 1: master — Set by hardware when the interface is in master mode (SB = 1), — Hardware cleared when a stop condition is detected on the bus (STOPF = 1), arbitration lost (ARLO = 1), or when PE = 0.

30.5.8. 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	Res	Res	CCR[11:0]											
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: Standard mode 1: Fast Mode
14	DUTY	RW	0	Duty cycle in fast mode. 0: In fast mode: T_{low}/T_{high} = 2 1: In fast mode: T_{low}/T_{high} = 16/9
13:12	Reserved	RES	-	Reserved
11:0	CCR[11:0]	RW	0	Clock control division factor in fast/standard mode (master mode). This division factor is used to set the SCL clock in master mode. ■ Standard Mode: (1) T_{high} = CCR x Tpclk (2) T_{low} = CCR x Tpclk ■ Express Mode: (3) DUTY = 0: T_{high} = CCR x Tpclk Tlow = 2 x CCR x Tpclk (4) DUTY = 1 (to reach 400KHz):

Bit	Name	R/W	Reset Value	Function
				T_{high} = 9 x CCR x Tpclk Tlow = 16 x $_{CCR}$ x Tpclk Note: 1) The minimum allowed setting is 0x04, and the minimum allowed in fast DUTY mode is 0x01 2) $T_{high} = t_{r(SCL)} + t_{w(SCLH)}$ 3) $T_{low} = t_{r(SCL)} + t_{w(SCLL)}$ 4) These delays have no filter 5) This register can only be configured when PE = 0, 6) f_{CK} should be an integer multiple of 10 MHz, so that the fast 400 kHz can be correctly generated at which

30.5.9. I2C TRISE register (I2C_TRISE)

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	Res	Res	Res	Res	TRISE[5:0]					
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:6	Reserved	RES	-	Reserved
5:0	TRISE	RW	0	Maximum rise time in fast/standard mode (master mode). These bits should provide the maximum duration of the S CL feedback loop in master mode. The purpose of this is to maintain a stable frequency of SCL regardless of the duration of the rising edge of SCL. These bits must be set to the maximum SCL rise time given in the I2C bus specification in 1 increments. For example: the maximum allowable SCL rise time in standard mode is 1000ns. If the value in FREQ [5:0] in the I2C_CR2 register is equal to

Bit	Name	R/W	Reset Value	Function
				0x08, Tpclk = 125ns, then TRISE is configured as 0x09 (1000ns/125ns = 8 + 1 = 9). Filter values can also be added to TRISE. If the result is not an integer, the integer part is written to TRISE to ensure the t_{HIGH} parameter. Note: This register can only be set when PE = 0.

30.5.10. I2C register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	I2C_CR1	Reserved																SWRST	Reserved	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	Reserved	SMBUS	PE
	Reset value																	0		0	0	0	0	0	0	0	0	0	0	0		0	0
0x04	I2C_CR2	Reserved																Reserved			LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Reserved		FREQ[5:0]					
	Reset value																				0	0	0	0	0			0	0	0	0	0	0
0x08	I2C_OAR1	Reserved																ADDMODE	Reserved					ADD[9:8]		ADD[7:1]							ADD0
	Reset value																	0						0	0	0	0	0	0	0	0	0	0
0x0C	I2C_OAR2	Reserved																								ADD2[7:1]							ENDUAL
	Reset value																									0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x10	I2C_DR	Reserved																Reserved								DR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x14	I2C_SR1	Reserved																SMBALERT	TIMEOUT	Reserved	PECERR	OVR	AF	ARLO	BERR	TXE	RXNE	Reserved	STOPF	ADD10	BTF	ADDR	SB
	Reset value																	0	0		0	0	0	0	0	0	0		0	0	0	0	0
0x18	I2C_SR2	Reserved																PEC[7:0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Reserved	TRA	BUSY	MSL
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
0x1C	I2C_CCR	Reserved																F/S	DUTY	Reserved		CCR[11:0]											
	Reset value																	0	0			0	0	0	0	0	0	0	0	0	0	0	0
0x20	I2C_TRISE	Reserved																Reserved										TRISE[5:0]					
	Reset value																											0	0	0	0	1	0

31. Serial peripheral interface (SPI)

This project designs and implements two SPI modules, both of which have exactly the same functions.

31.1. Introduction

Serial Peripheral Interface (SPI) allows the chip to communicate with external devices in half-duplex, full-duplex, and simplex synchronous serial communication. This interface can be configured in master mode and provides the communication clock (SCK) for external slave devices. The interface can also work in a multi-master configuration.

It can be used for a variety of purposes, including two-wire simplex simultaneous transmission using one bidirectional data line. and also for reliable communication using CRC checksum. Reliable communication with CRC checksum is also available.

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 can operate in 2 modes, master and slave, in half-duplex communication. When acting as a master, it provides a clock signal to an external slave device via the interface.

31.2. Main features

31.2.1. SPI main features

- Master or slave operation
- Full-duplex synchronous transfers on three lines
- Half-duplex synchronous transfer on two lines (with bidirectional data line)
- Simplex synchronous transfers on two lines (with unidirectional data line)
- 8-bit to 16-bit data size selection
- Multimaster mode capability
- 8 master mode baud rate prescalers up to fPCLK/4.
- Slave mode frequency up to fPCLK/4.

- NSS management by hardware or software for both master and slave: dynamic change of master/slave operations
- Programmable clock polarity and phase
- Programmable data order with MSB-first or LSB-first shifting
- Dedicated transmission and reception flags with interrupt capability
- SPI bus busy status flag
- Hardware CRC support for reliable communication
  - CRC value can be sent as the last byte in transmit mode
  - automatic CRC checksum of the last byte received in full duplex mode
- Master mode faults, overloads and CRC error flags that can cause interruptions
- 2 DMA-capable Rx and Tx FIFOs with a depth of 4 and a width of 16 bits (8 bits when the data frame is set to 8 bits)

31.2.2. IIS main features

- Simplex communication (send or receive only)
- Master or slave operation
- 8-bit linear programmable divider for precise audio sampling frequencies (8KHz to +96KHz)
- Data format can be 16-bit, 24-bit or 32-bit
- Fixed packet format for audio channels in 16-bit (16-bit data frames) or 32-bit (16-bit, 24-bit 32-bit data frames)
- Programmable clock polarity (steady state)
- Underflow flag bit in slave transmit mode and overflow flag bit in master/slave receive mode
- 16-bit data registers for transmit and receive, one register present at each end of the channel
- I2S protocol support:
- I2S Philips standard
- MSB alignment standard (left-aligned)
- LSB alignment standard (right-aligned)

- PCM standard (16-bit channel frame with long or short frame synchronization on a 16-bit channel frame or 16-bit data frame extended to a 32-bit channel frame)
- Data reversal always MSB first
- DMA capability for both transmit and receive
- Master clock can be output to external audio devices with a fixed ratio of 256xFs (Fs is the audio sampling frequency)

31.3. SPI function description

31.3.1. Overview

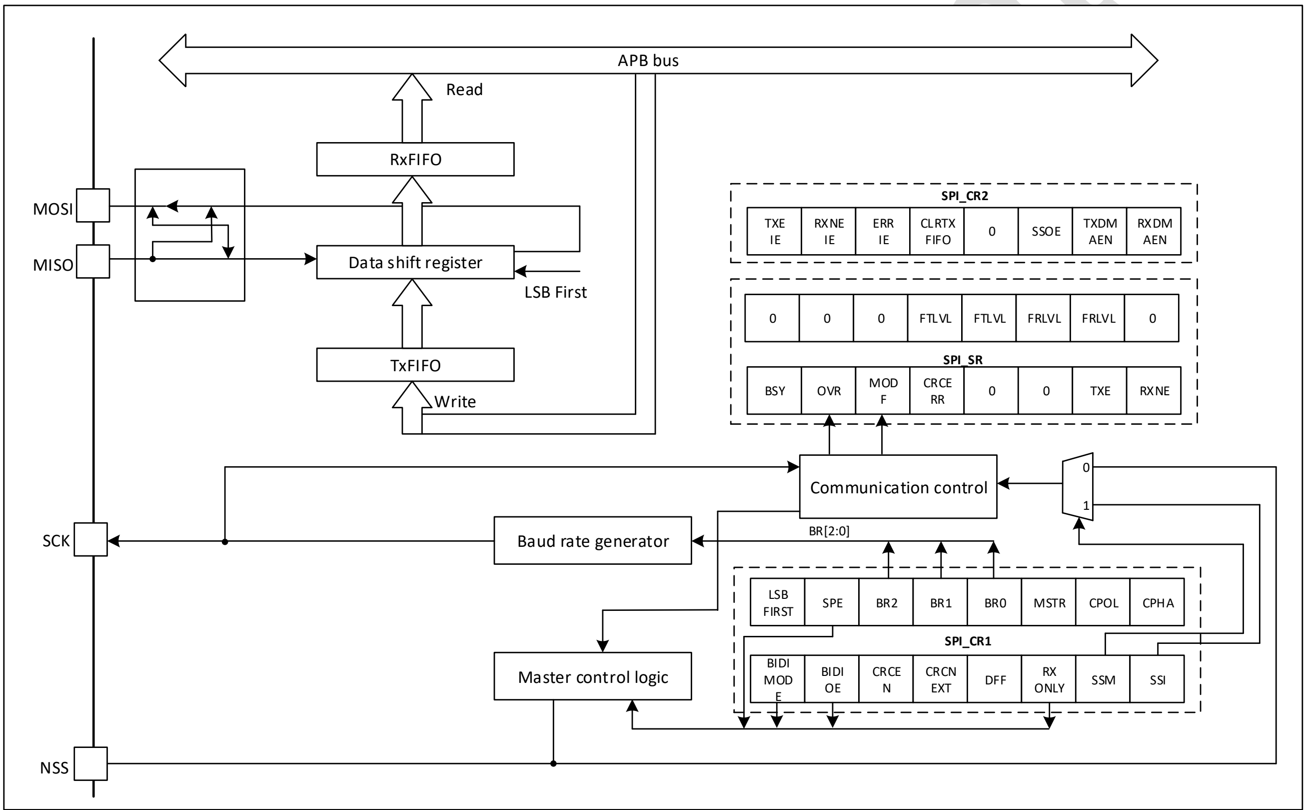


Figure 31-1 SPI block diagram

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 or
- 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

31.3.2. 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 or 3 wires (with software NSS management) or 3 or 4 wires (with hardware NSS management). Communication is always initiated by the master.

31.3.2.1. 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 linked using two unidirectional lines between the MOSI and the MISO pins. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master transmits the data to be sent to the slave via the MOSI line and receives data from the slave via the MISO line. When the data frame transfer is complete (all the bits are shifted) the information between the master and slave is exchanged.

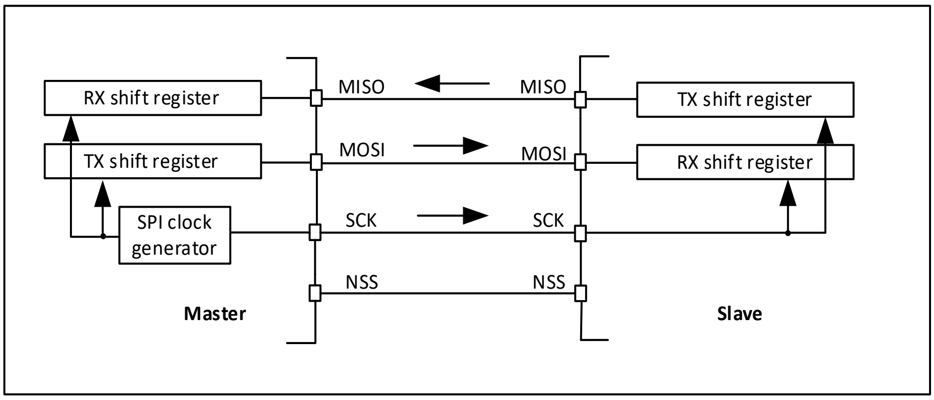


Figure 31-2 Full-duplex single master/slave application

31.3.2.2. Half-duplex communication

The SPI can communicate in half-duplex mode by setting the BIDIMODE bit in the SPIx_CR1 register. In this configuration, one single cross connection line is used to link the shift registers of the master and slave together. During this communication, the data is synchronously shifted between the shift registers on the SCK clock edge in the transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPIx_CR1 registers. In this configuration, the master's MISO pin and the slave's MOSI pin are free for other application uses and act as GPIOs.

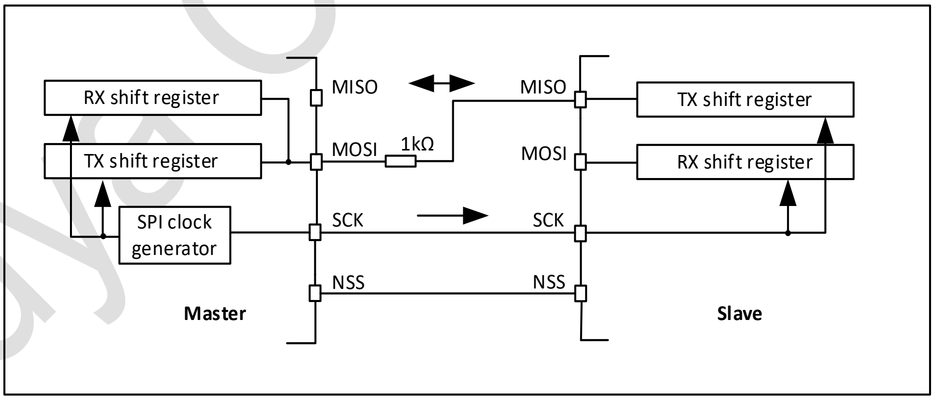


Figure 31-3 Half-duplex single master/slave application

The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, the pins can be left unused by the peripheral. Then the flow has to be handled internally for both master and slave.

31.3.2.3. Simplex communications

The SPI can communicate in simplex mode by setting the SPI in transmit-only or in receiveonly using the RXONLY bit in the SPIx_CR2 register. In this configuration, only one line is used for the transfer between the shift registers of the master and slave. The remaining MISO and MOSI pins pair is not used for communication and can be used as standard GPIOs.

1. **Transmit-only mode (RXONLY = 0)**: The configuration settings are the same as for fullduplex. The application has to ignore the information captured on the unused input pin. This pin can be used as a standard GPIO.

2. **Receive-only mode (RXONLY = 1)**: The application can disable the SPI output function by setting the RXONLY bit. In slave configuration, the MISO output is disabled and the pin can be used as a GPIO. The slave continues to receive data from the MOSI pin while its slave select signal is active. Received data events appear depending on the data buffer configuration. In the master configuration, the MOSI output is disabled and the pin can be used as a GPIO. The clock signal is generated continuously as long as the SPI is enabled. The only way to stop the clock is to clear the RXONLY bit or the SPE bit and wait until the incoming pattern from the MISO pin is finished.

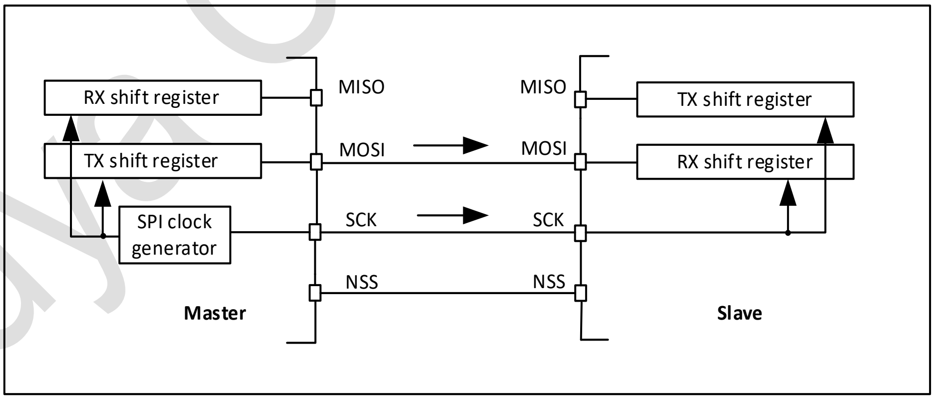


Figure 31-4 Simplex single master/single slave application(master in transmit-only/slave in receive-only mode)

1. The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, the pins can be left unused by the peripheral. Then the flow has to be handled internally for both master and slave. For more details see Section 28.5.5: Slave select (NSS) pin management.

2. An accidental input information is captured at the input of transmitter Rx shift register. All the events associated with the transmitter receive flow must be ignored in standard transmit only mode.
3. In this configuration, both the MISO pins can be used as GPIOs.

simplex communication can be replaced by half -duplex communication by setting the transfer direction (the bidirectional mode is enabled when the BI DIO E bit is not changed).

31.3.3. 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 a slave by pulling the connected slave NSS low. When this is done, standard master and dedicated slave communication is established.

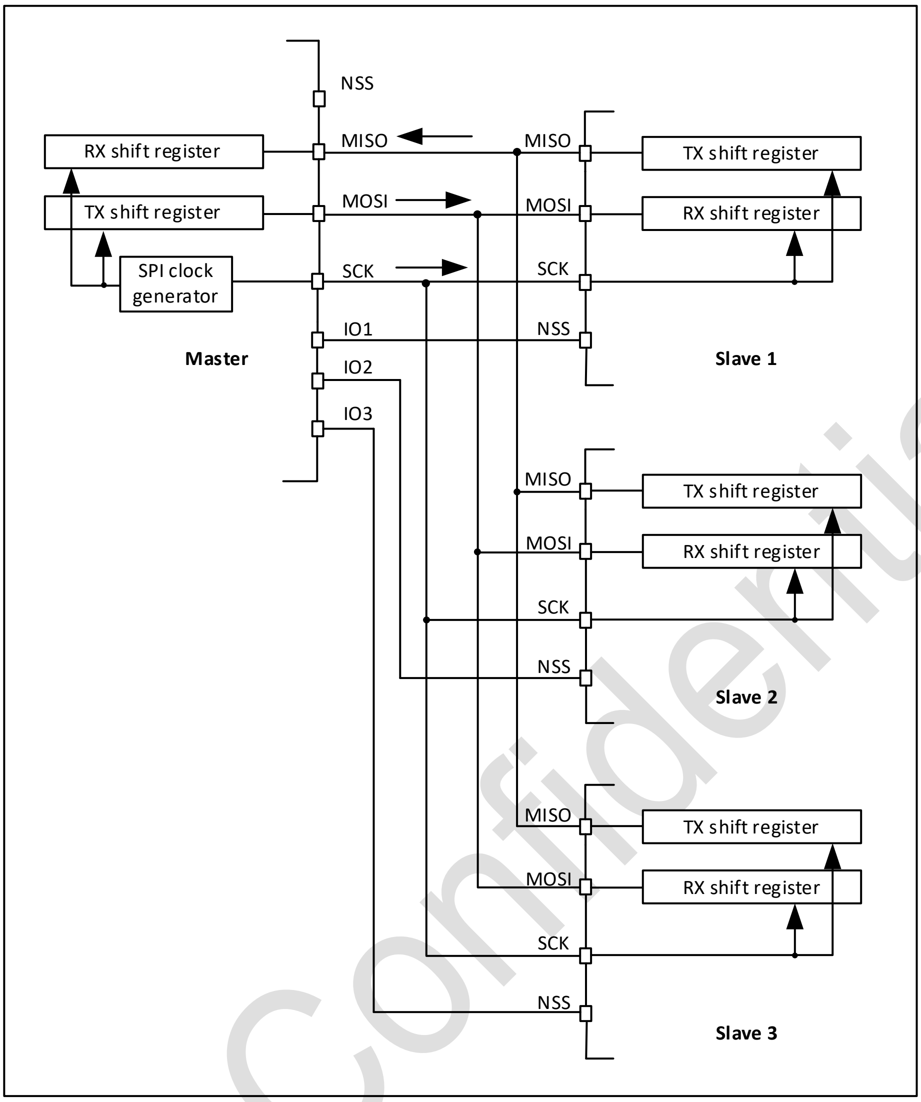


Figure 31-5 Master communicates with three independent slaves

NSS is not used on the host side in this configuration. Any MODF errors must be prevented by SSM = 1, SSI = 1.

Since the MISOs of the slaves are connected together, all slaves must configure their MISO's GPIO as AF open-drain.

31.3.4. 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 session is completed, the active slave select signal is released and the node mastering the bus temporary returns back to passive slave mode waiting for next session start. If potentially both nodes raised their mastering request at the same time a bus conflict event appears (see mode fault MODF event). Then the user can apply some simple arbitration process (e.g. to postpone next attempt by predefined different time-outs applied at both nodes).

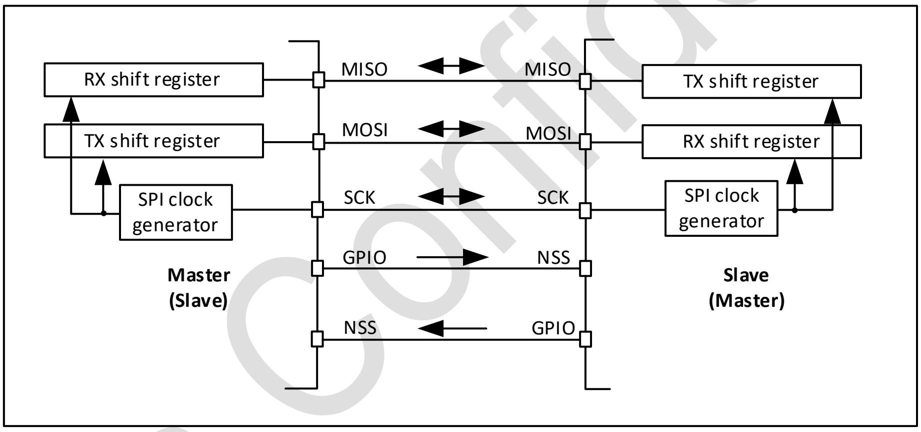


Figure 31-6 Multi-master application

The NSS pin is configured at hardware input mode at both nodes. Its active level enables the MISO line output control as the passive node is configured as a slave.

31.3.5. 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. s an input it can prevent multimaster bus collision, and as an output it can drive a slave elect signal of a single slave.

Hardware or software slave select management can be set using the SSM bit in the SPIx_CR1 register:

3. Software NSS management (SSM = 1): in this configuration, slave select information is driven internally by the SSI bit value in register SPIx_CR1. The external NSS pin is free for other application uses.
4. Hardware NSS management (SSM = 0): in this case, there are two possible configurations. The configuration used depends on the NSS output configuration

1. NSS output enable (SSM = 0,SSOE = 1): this configuration is only used when the MCU is set as master. 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). A pulse can be generated between continuous communications if NSS pulse mode is activated (NSSP = 1). 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.

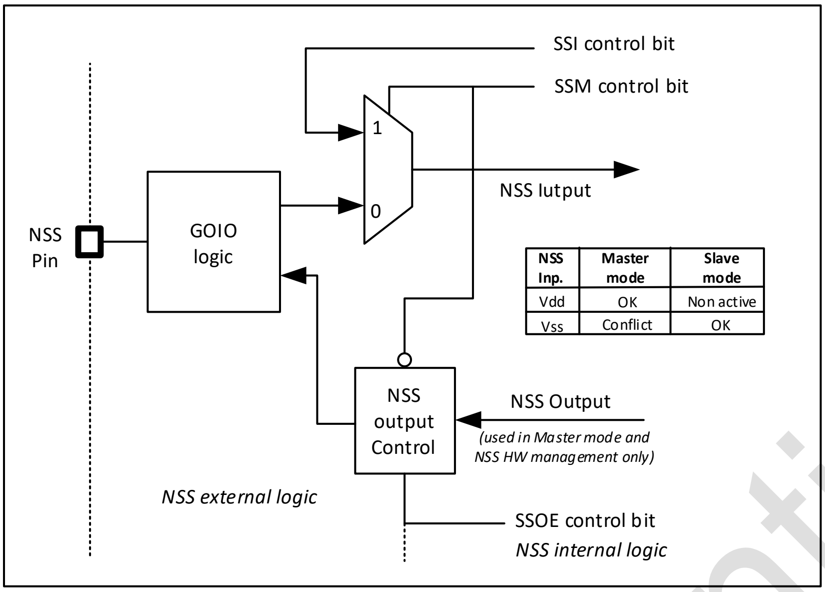


Figure 31-7 Hardware/software slave select management

31.3.6. Communication formats

During SPI communication, receive and transmit operations are performed simultaneously. The serial clock (SCK) synchronizes the shifting and sampling of the information on the data lines. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slaves devices must follow the same communication format.

31.3.6.1. Clock phase and polarity controls

There are 4 possible timings that can be configured by software through the CPOL and CPHA bits (SPI_CR1 register). CPOL (clock polarity) controls the IDLE state of the clock when no data is being transmitted. This bit affects both master and slave. If CPOL is reset, the SCK pin has a low state. If CPOL is set, the SCK pin has a high IDLE state.

If CPHA is set, the second edge of SCK captures the first data bit transmitted (falling edge if CPOL is reset, rising edge otherwise). On the occurrence of clock change type, the data is latched. If CPHA is reset, the first edge of SCK captures the first transmitted data bit (falling edge if CPOL is set, rising edge otherwise). Data is latched when this type of clock change occurs.

The combination of CPOL and CPHA selects the data capture clock edge.

SPI must be disabled (SPE = 0) before CPOL/CPHA is changed.

The IDLE state of SCK must correspond to the polarity selected by the SPI_CR1 register.

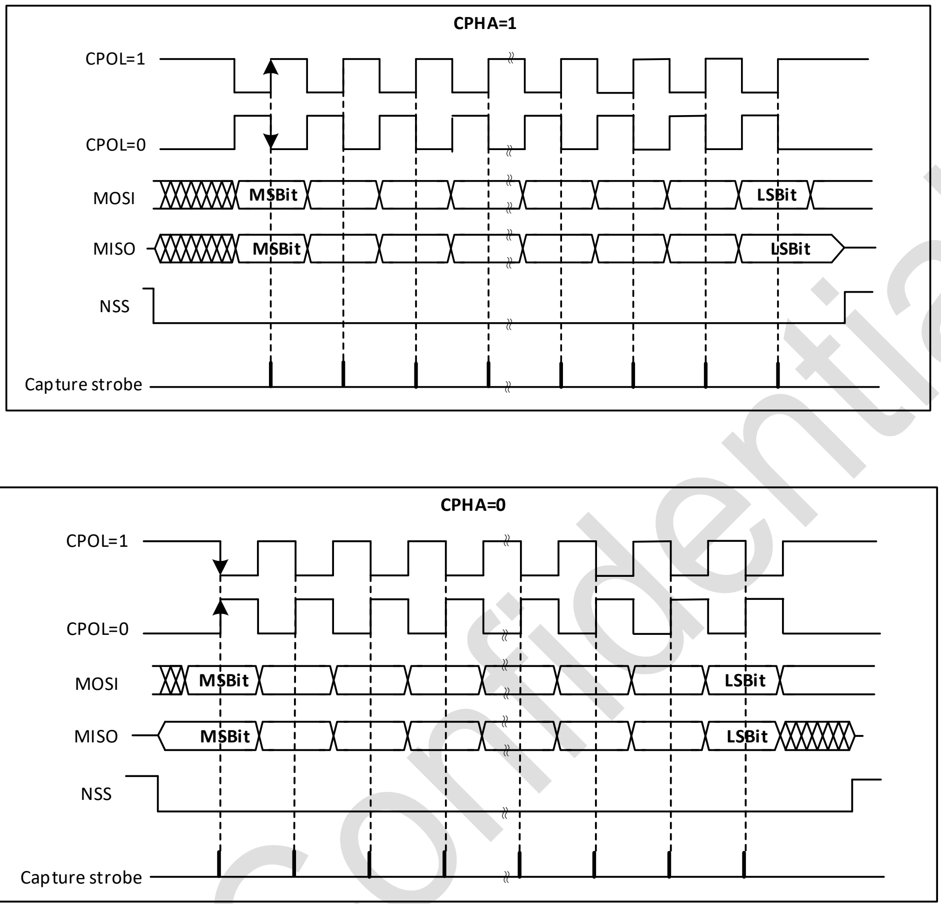


Figure 31-8 Data clock timing diagram

The order of data bits depends on LSBFIRST bit setting.

31.3.6.2. Data frame format

Through the LSBFIRST bit (SPI_CR 1 register), the SPI sh ift register can be set to MSB-FIRST or LSB-FIRST. Select the number of bits in the data frame by using the DFF bit (SPI_CR2 register). It can be selected as 8-bit or 16-bit length, and this setting applies to both sending and receiving.

31.3.7. SPI configuration

The configuration procedure is almost the same for master and slave. For specific mode setups, follow the dedicated sections. When a standard communication is to be initialized, perform these steps:

- Write related GPIO registers: configure MOSI, MISO and SCK pins
- Write SPI_CR1 register

- Configure the clock baud rate via BR[2:0] (not required for slave mode)
- Configure CPOL and CPHA
- simplex or half -duplex mode by RXONLY or B IDIMODE and BIDIOE (RXONLY and BIDIMODE cannot be active at the same time)
- Configure LSBFIRST
- Configure SSM and SSI
- Configure the MSTR bit (in multi-master In NSS configuration, if the host is configured to prevent MODF errors, avoid NSS conflict state)
- Write SPI_CR2 register
  - Configure DS bit, select the number of data frame bits
  - Configure SSOE (not required for slave mode)
  - Configure the FRXTH bit. RXFIFO thresholds must be aligned with the number of bits accessed to the SPI_DR register
- Write the corresponding DMA register: configure the SPI Tx and Rx channels of the DMA

31.3.8. 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 data register of the slave must already contain data to be sent before starting communication with the master (either on the first edge of the communication clock, or before the end of the ongoing communication if the clock signal is continuous). The SCK signal must be settled at an idle state level corresponding to the selected polarity before the SPI slave is enabled.

The master at full-duplex (or in any transmit-only mode) starts to communicate when the SPI is enabled and TXFIFO is not empty, or with the next write to TXFIFO.

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 handling DMA, follow the dedicated section.

31.3.9. Data transmission and reception procedures

31.3.9.1. RXFIFO and TXFIFO

All SPI data transactions pass through the 32-bit embedded FIFOs. 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 except for receiver-only mode (slave or master).

The handling of FIFOs depends on the data exchange mode (duplex, simplex), data frame format (number of bits in the frame), access size performed on the FIFO data registers (8-bit or 16-bit).

A read access to the SPIx_DR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPIx_DR stores the written data in the TXFIFO at the end of a send queue. The read access must be always aligned with the RXFIFO threshold configured by the FRXTH bit in SPIx_CR2 register. FTLVL [1:0] and FRLVL [1:0] bits indicate the current occupancy level of both FIFOs.

A read access to the SPIx_DR register must be managed by the RXNE event. This event is triggered when data is stored in RXFIFO and the threshold (defined by FRXTH bit) is reached. 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, RXFIFO can store up to four data frames.

Both TXE and RXNE events can be handled by queries, interrupts and DMA.

When the RXFIFO is full, an overrun event is generated if the next data is received. overrun events can be handled by means of queries and interrupts.

The BSY bit that is set indicates that communication is in progress for 1 current data frame. When the clock signal is provided continuously, the BSY flag remains set between two data frames on the master side. However, between each data frame transmission on the slave side, the BSY remains low for a minimum of 1 SPI Clock width.

In some application scenarios, when data is written to the TXFIFO, the TXFIFO data can be cleared by setting the CLRTXFIFO bit so that new data can be written to the TXFIFO again to resume communication.

31.3.9.2. Sequence handling

A few data frames can be passed at single sequence to complete a message. When transmission is enabled, a sequence begins and continues while any data is present in the TXFIFO of the master. The clock signal is provided continuously by the master until TXFIFO becomes empty, then it stops waiting for additional data.

In receive-only modes, half-duplex (BIDIMODE = 1, BIDIOE = 0) or simplex (BIDIMODE = 0, RXONLY = 1) the master starts the sequence immediately when both SPI is enabled and receive-only mode is activated. The clock signal is provided by the master and it does not stop until either SPI or receive-only mode is disabled by the master. The master receives data frames continuously up to this moment.

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. Be aware there is no underflow error signal for master or slave in SPI mode, and data from the slave is always transacted and processed by the master even if the slave could not prepare it correctly in time. It is preferable for the slave to use DMA, especially when data frames are shorter and bus rate is high.

Each sequence must be encased by the NSS pulse in parallel with the multislave system to select just one of the slaves for communication. In a single slave system it is not necessary to control the slave with NSS, but it is often better to provide the pulse here too, to synchronize the slave with the beginning of each data sequence. NSS can be managed by both software and hardware (see Section 28.5.5: Slave select (NSS) pin management).

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.

31.3.9.3. Procedure for disabling the SPI

When SPI is disabled, it is mandatory to follow the disable procedures described in this paragraph. 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.

Master in full-duplex or transmit only mode can finish any transaction when it stops providing data for transmission. In this case, the clock stops after the last data transaction. Special care must be taken in packing mode when an odd number of data frames are transacted to prevent some dummy byte exchange (refer to Data packing section). Before the SPI is disabled in these modes, the user must follow standard disable procedure. 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 having unread data, ensure that RXFIFO is empty when disabling the SPI, by using the correct disabling procedure, or by initializing all the SPI registers with a software reset via the control of a specific register dedicated to peripheral reset.

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.

- When transactions' streams from DMA or FIFO are completed while the last data frame or CRC frame transaction is still ongoing in the peripheral bus.

The correct disable procedure is (except when receive only mode is used):

1. Wait until FTLVL [1:0] = 00 (no more data to transmit).

2. Wait until BSY = 0 (the last data frame is processed).

3. Disable the SPI (SPE = 0).

4. Read data until FRLVL [1:0] = 00 (read all the received data).

The correct disable procedure for certain receive only modes is:

1. Interrupt the receive flow by disabling SPI (SPE = 0) in the specific time window while the last data frame is ongoing.

2. Wait until BSY = 0 (the last data frame is processed).

3. Read data until FRLVL [1:0] = 00 (read all the received data).

31.3.9.4. Using DMA communication

To work at maximum speed and to speed up the data read/write process to avoid overrun, the SPI has the DMA capability of the simple request/response protocol.

When TXE or RXNE is set, a DMA request is generated. there are separate requests for the Tx and Rx buffer.

1. When sending, each time TXE is set to 1, a DMA request is generated. The DMA will then write data to the SPI_DR register.
2. When receiving, each time RXNE is set to 1, a DMA request is generated. The DMA then reads the data from the SPI_DR register.

When the SPI is being used to send data only, it is possible to enable the SPI Tx DMA channel only. In this case, the OVR flag bit is set because the data being received is not read away. When the SPI is being used for receive data only, the SPI Rx DMA channel can be enabled.

When the DMA has been written with all the data to be sent (the TCIF flag bit of the DMA_ISR register is set) during transmit, the BSY flag can be monitored to ensure that SPI communication is complete. This is used to avoid that the final transfer is corrupted when the SPI is switched off or when it enters stop mode. The software must first wait for FTLVL[1:0]=00 and then for BSY=0.

When starting to use DMA communication, the following steps must be followed in order to avoid error events in the DMA channel tube:

1) Enable DMA Rx buffer (RXDMAEN bit of SPI_CR2) (if Rx DMA is used)
2) Enable Tx Rx DMA streams (in the DMA register) (if steams is used)
3) Enable DMA Tx buffer (in the TXDMAEN bit of the SPI_CR2 register) (if Tx DMA is used)
4) Enable SPI via the SPE bit

Force the communication to be switched off with the following steps:

a) Shutting down DMA Tx Rx streams (in the DMA register) (if stream is used)
b) Shutting down the SPI via the SPI shutdown process
c) Close the DMA Tx and Rx buffer (if DMA Tx and Rx are used) by clearing TXDMAEN and RXDMAEN (SPI_CR2 registers).

31.3.9.5. Communication Timing

This section describes some typical timings that are valid for queries, interrupts or DMAs. For simplicity, it is assumed that LSBFIRST=0, CPOL=0, CPHA=1. The full configuration of DMA operation is also not provided.

a) When NSS is active, SPI is enabled and the slave starts to control MISO; the slave loses control of MISO when NSS is released or SPI is turned off. For the slave, sufficient time must be provided to the master before the transfer starts so that the data can be prepared in advance.

On the host side, the SPI peripheral controls the MOSI and SCK signals (also the NSS signal) only when the SPI is enabled. If the SPI is switched off, the SPI peripheral is disconnected from the GPIO, so the level value on these lines depends on the GPIO setting.

b) On the master side, BSY remains active between frames if the communication is continuous. On the slave side, the BSY signal usually becomes low for at least one clock cycle between data frames.

c) The TXE signal is only cleared when the TXFIFO is full.

d) As soon as the TXDMAEN bit is set, the DMA arbitration process starts. After TXEIE is set, the TXE interrupt is generated. When the TXE signal is valid, data transfer to the TxFIFO starts until the TxFIFO becomes full, or the DMA transfer is completed.

e) If all the data to be sent can fit into the TxFIFO, the DMA Tx TCIF flag is pulled high prior to the SPI bus transfer. This flag remains high until the SPI interaction is complete.

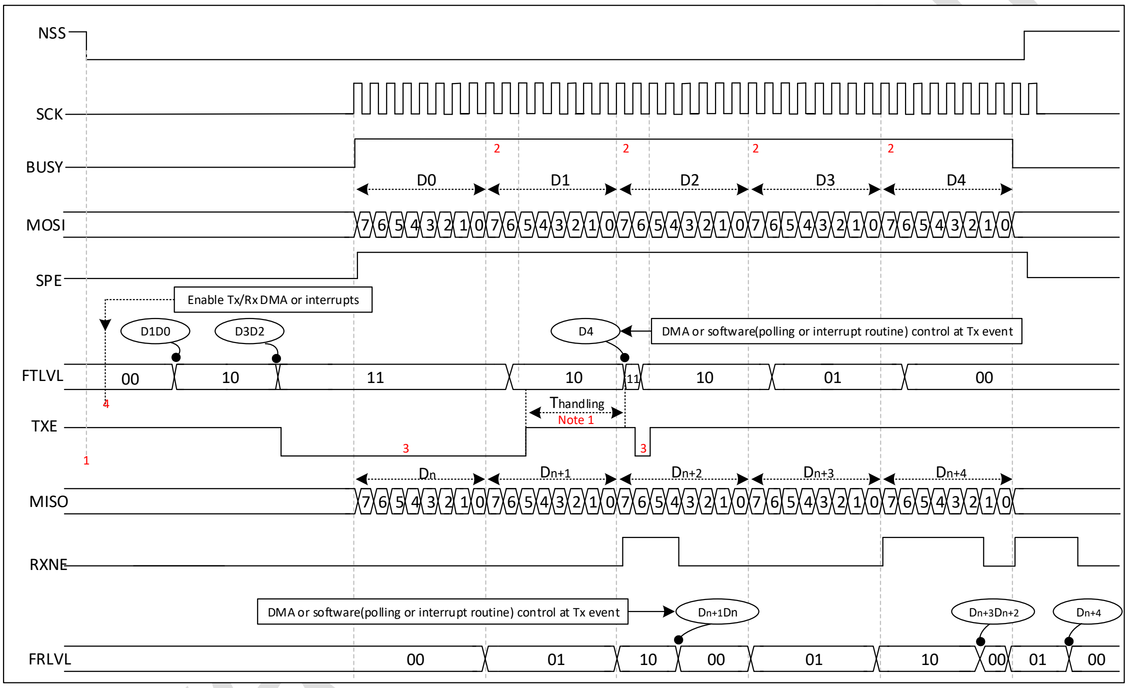


Figure 31-9 Host full duplex timing (data width = 8)

31.3.10. Status flags

Three status flags are provided for the application to completely monitor the state of the SPI bus.

31.3.10.1. Tx 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 SPIx_CR2 register is set. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

31.3.10.2. Rx buffer not empty (RXNE)

The RXNE flag is set depending on the FRXTH bit value in the SPIx_CR2 register:

1. If FRXTH is set, RXNE goes high and stays high until the RXFIFO level is greater or equal to 1/4 FIFO.
2. If FRXTH is cleared, RXNE goes high and stays high until the RXFIFO level is greater than or equal to 1/2 (16-bit).

An interrupt can be generated if the RXNEIE bit in the SPIx_CR2 register is set.

The RXNE is cleared by hardware automatically when the above conditions are no longer true.

31.3.10.3. Busy flag (BSY)

The BSY flag is set and cleared by hardware (writing to this flag has no effect). This flag indicates the state of the SPI communication layer.

When it is set to '1', it indicates that the SPI is busy communicating, with one exception: in the bidirectional receive mode of master mode (MSTR = 1, BDM = 1 and BDOE = 0), the BSY flag is held during reception to low.

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, so it needs to strictly follow the procedure below.

The BSY flag is also useful for preventing write collisions in a multimaster system.

Except for the bidirectional receive 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 the 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: it is recommended to use always the TXE and RXNE flags (instead of the BSY flags) to handle data transmission or reception operations.

31.3.11. Error flags

31.3.11.1. Mode fault (MODF)

Mode fault occurs when the master device has its internal NSS signal (NSS pin in NSS hardware mode, or SSI bit in NSS software mode) pulled low. 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 SPIx_SR register while the MODF bit is set.

2. Then write to the SPIx_CR1 register.

To avoid any multiple slave conflicts in a system comprising several MCUs, the NSS pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits can be restored to their original state after this clearing sequence. As a security, hardware does not allow the SPE and MSTR bits to be set while the MODF bit is set. In a slave device the MODF bit cannot be set except as the result of a previous multimaster conflict.

31.3.11.2. Overrun flag (OVR)

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 or the DMA did not have enough time to read the previously received data (stored in the RXFIFO) or when space for data storage is limited.

When an overrun condition occurs, the newly received value does not overwrite the previous one in the 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.

31.3.12. SPI interrupts

Table 31-1 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

31.3.13. CRC calculations

CRC checks are used to ensure reliable communication. Separate CRC calculators are used for data transmission and data reception. The CRC is calculated by performing a programmable polynomial operation on each received bit. the SPI provides a CRC8 or CRC16 calculation independent of the frame data length, which can be fixed to 8 or 16 bits. For all other data frame lengths, no CRC is available.

31.3.13.1. CRC Principle

Before the SPI is enabled (SPE = 1), CRC calculation is enabled by setting the CRCEN bit in the SPIx_CR1 register. The CRC value is calculated using an odd number of programmable polynomials on each bit. This calculation is processed on the sample 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 from the received data and the CRC sent by the transmitter, the CRCERR flag is set to indicate a data error. The correct procedure for handling CRC calculations depends on the SPI configuration and the selected transmission management method.

31.3.13.2. CPU-controlled CRC transmission

Begins and communicates continuously until the last data frame in the SPIx_DR register is sent or received. The CRCNEXT bit must then be set in the SPIx_CR1 register to indicate the start of CRC frame transmission after the currently processed data frame. The CRCNEXT bit must be set before the end of the last data frame transmission. The CRC calculation is stopped during the CRC transmission.

The received CRC data is stored in the RXFIFO in the form of bytes or words. This is why in CRC mode only, the receive buffer must be treated as a single 16-bit buffer for receiving only one data frame at a time. a CRC format transmission usually requires one more data frame at the end of the data transmission. However, when setting up an 8-bit data frame that passes the 16-bit CRC checksum, two more frames are required to send the full CRC value.

When the last CRC data is received, an automatic checksum is performed to compare the re-ceived value with the SPIx_RXCRC register. The software must check the CRCERR flag in the SPIx_SR register to determine if there is an error in the data transfer. After CRC reception, the CRC value is stored in the RXFIFO and the SPIx_DR register must be read to clear the RXNE flag.

31.3.13.3. DMA Controlled CRC Transfer

When SPI communication is enabled using DMA mode with CRC, the sending and receiving 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 handled by software. The counter for the SPI transfer DMA channel must be set to the number of data frames to be transferred, which does not include the CRC frames. On the receiver side, the received CRC value is automatically han-dled 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 stored in the RXFIFO. In full duplex mode, the counter of the receive DMA channel can be set to the number of data frames to be received in-cluding the CRC.

In receive only mode, the DMA receive channel counter should only contain the amount of data transferred and not the CRC calculation. A full DMA based transfer is then made, as it works as a single buffer in this mode, so all CRC values must be read back from the FIFO by software. At the end of the data and CRC transfer, the CRCERR flag in the SPIx_SR register is set if an error occurs during the transfer.

31.4. IIS functional description

31.4.1. Overview

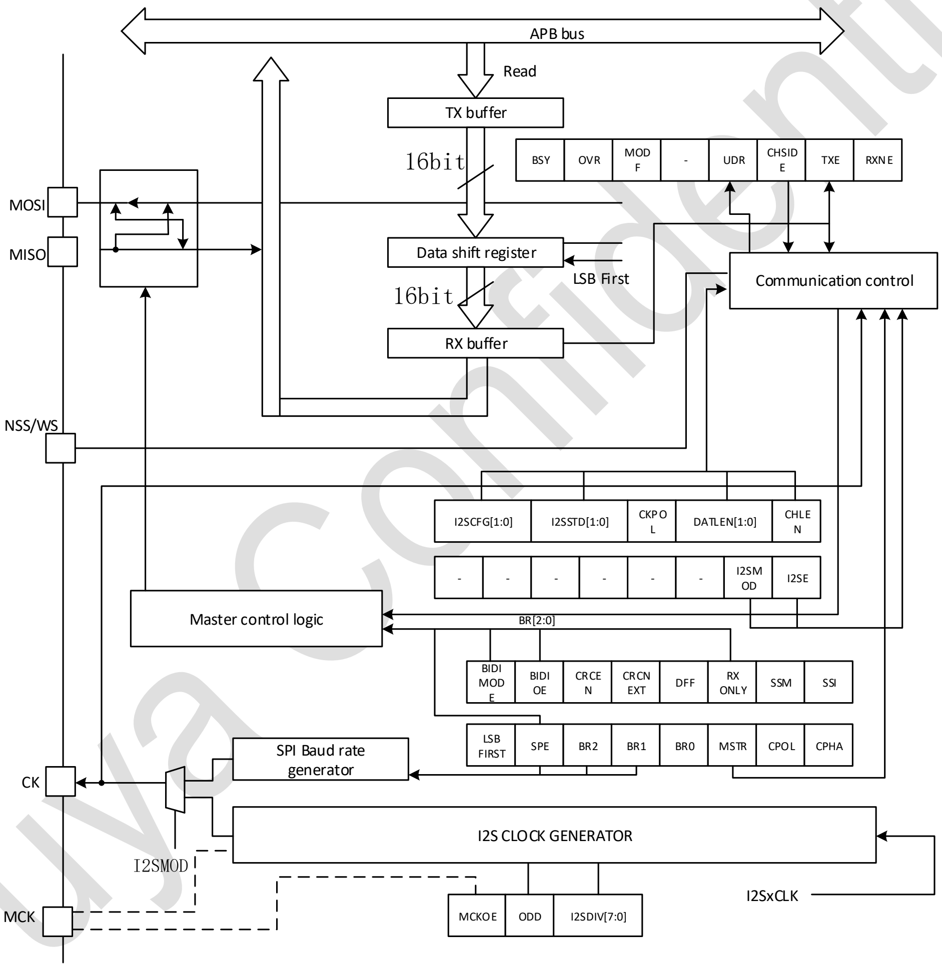


Figure 31-10 IIS Block Diagram

The I2S function can be enabled by setting the I2SMOD position of register SPI_I2SCFGR to '1'. At this point, the SPI module can be used as an I2S audio interface. The I2S interface uses approximately the same pins, flags and interrupts as the SPI interface.

I2S and SPI share 3 pins:

a) SD: serial data (mapped to the MOSI pin), used to send and receive data from the 2-way time division multiplexed channel;

b) WS: word select (mapped to the NSS pin), used as a data control signal output in master mode and as an input in slave mode;

c) CK: serial clock (mapped to the SCK pin), output as a clock signal in master mode and as an input in slave mode.

An additional pin can be used to output the clock when a master clock is required for certain external audio devices:

d) MCK: master clock (independently mapped), used as output additional clock signal pin 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 pre-set to 256 × Fs, where Fs is the sampling frequency of the audio signal.

When set to master mode, the I2S uses its own clock generator to generate the clock signal for communication. This clock generator is also the clock source for 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 sets parameters such as audio standard, slave/master mode, data format, packet frame, clock polarity, etc.).

The register SPI_CR1 and all CRC registers are not used in I2S mode. Similarly, the SSOE bit of register SPI_CR2, and the MODF and CRCERR bits of register SPI_SR are not used in I2S mode.

I2S uses the same register SPI_DR as SPI for 16-bit wide mode data transfer.

31.4.2. Audio protocol

The 3-wire bus supports time-division multiplexing of audio data on 2 channels: left and right, but only one 16-bit register is used for transmit or receive. Therefore, when writing data to the data register, the software must write the corresponding data according to the channel currently being transmitted; similarly, when reading the register data, the CHSIDE bit of the SPI_SR register is checked to determine which channel the received data belongs to. The left channel always sends data before the right channel (the CHSIDE bit is meaningless under the PCM protocol). There are four available combinations of data and packet frames. Data can be sent in the following four data formats:

1. 16-bit data packed into a 16-bit frame
2. 16-bit data packed into 32-bit frames
3. 24-bit data packed into 32-bit frames
4. 32-bit data packed into 32-bit frames

When using 16-bit data expansion into 32-bit frames, the first 16 bits (MSB) are meaningful data and the last 16 bits (LSB) are forced to 0. This operation requires no software intervention and no DMA requests (only one read/write operation is required). 24-bit and 32-bit data frames require 2 CPU read or write operations to register SPI_DR, and when using DMA, 2 DMA transfers. For 24-bit data, the lowest 8 bits are set to 0 by hardware after expansion to 32-bit. 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] bits and the PCMSYNC bits of register SPI_I2SCFGR.

31.4.2.1. I2S Philips protocol

With this standard, pin WS is used to indicate which channel the data being sent belongs to. This pin is active 1 clock cycle before the first data (MSB) is sent.

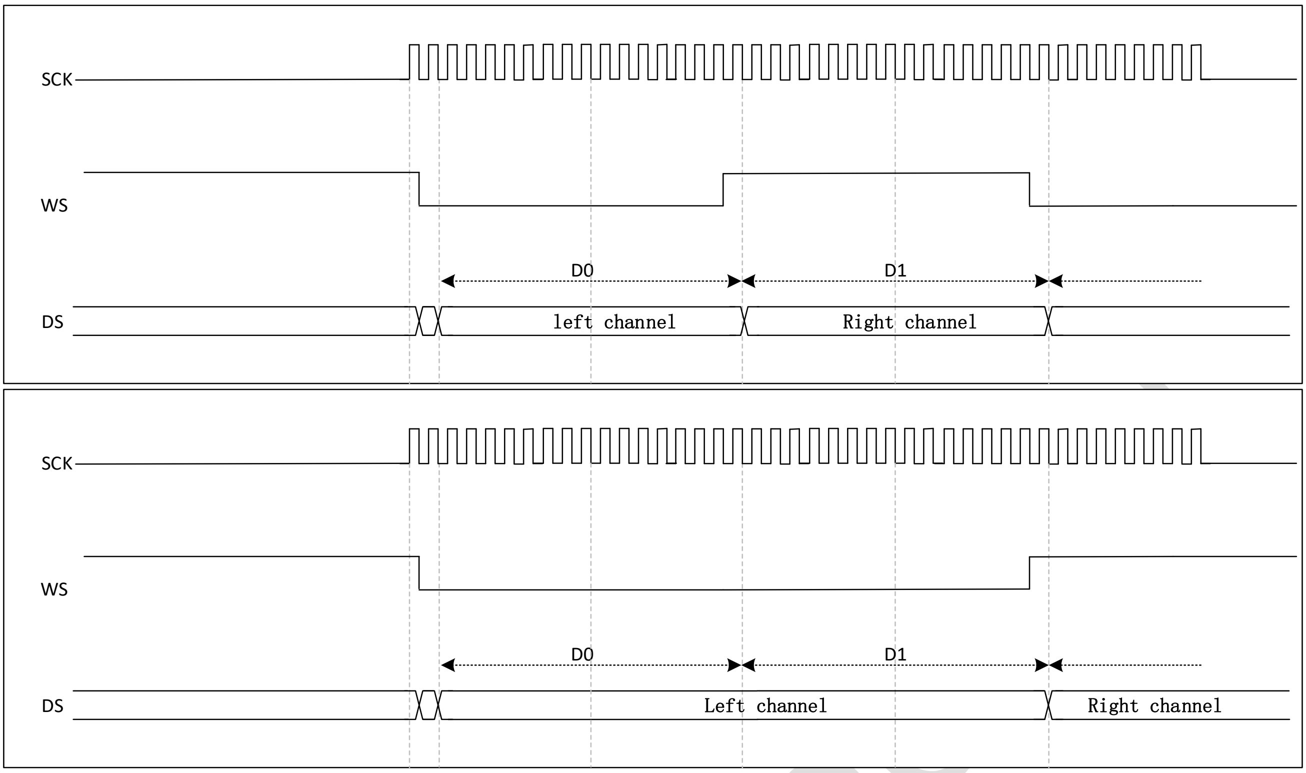


Figure 31-11 I2S Philips protocol waveform (16/32 bit full precision, CKPOL = 0)

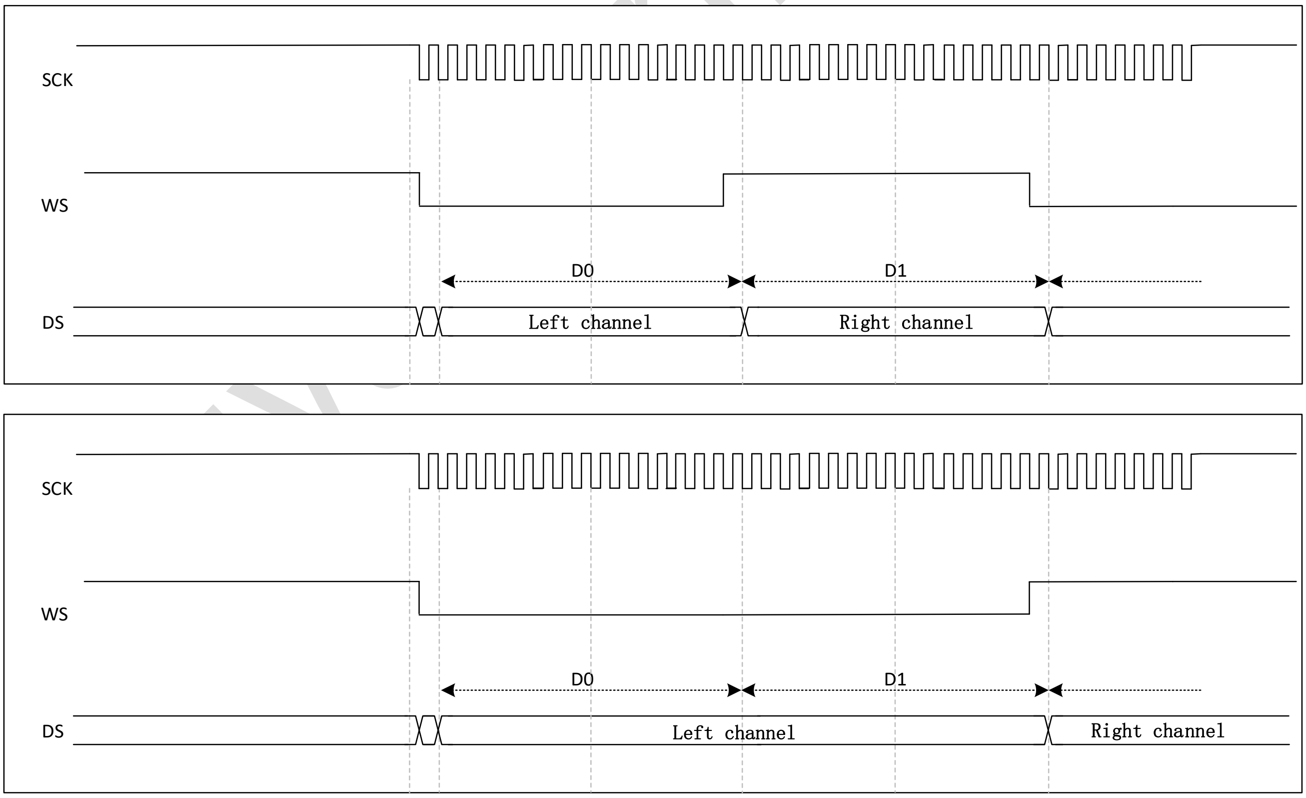


Figure 31-12 I2S Philips protocol waveform (16/32 bit full precision, CKPOL = 1)

The sender changes the data on the falling edge of the clock signal (CK) and the receiver reads the data on the rising edge. The WS signal also changes on the falling edge of the clock signal.

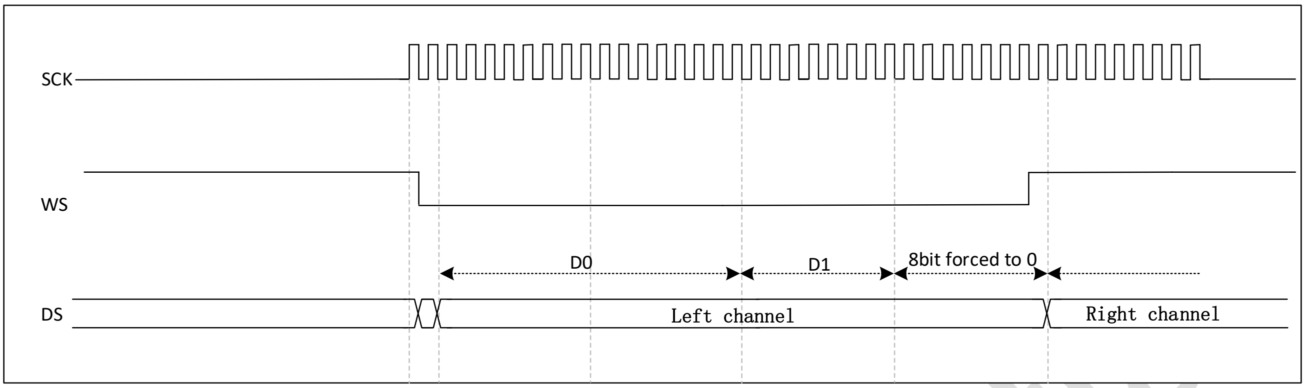


Figure 31-13 I2S Philips protocol standard waveform (24-bit frame, CKPOL = 0)

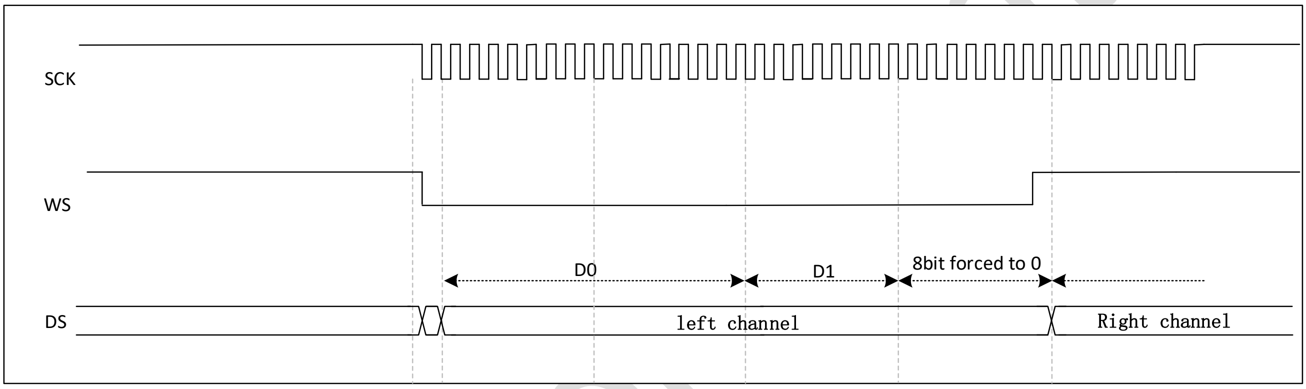


Figure 31-14 I2S Philips protocol standard waveform (24-bit frame, CKPOL = 1)

This mode requires 2 read or write operations on register SPI_DR.

In transmit mode: If 0x8EAA33 (24 bits) is required to be sent:

- The first write to register 0x8EAA; the second 0x33XX is written to the register. (only the high 8 bits are sent to complete the 24-bit data, the low 8 bits are meaningless and can be any value)

In receive mode: If receiving 0x8EAA33:

- The first read out of the data register is 0x8EAA; the second read out of the data register is 0x3300. (Only the high 8 bits are valid, the low 8 bits are always 0)

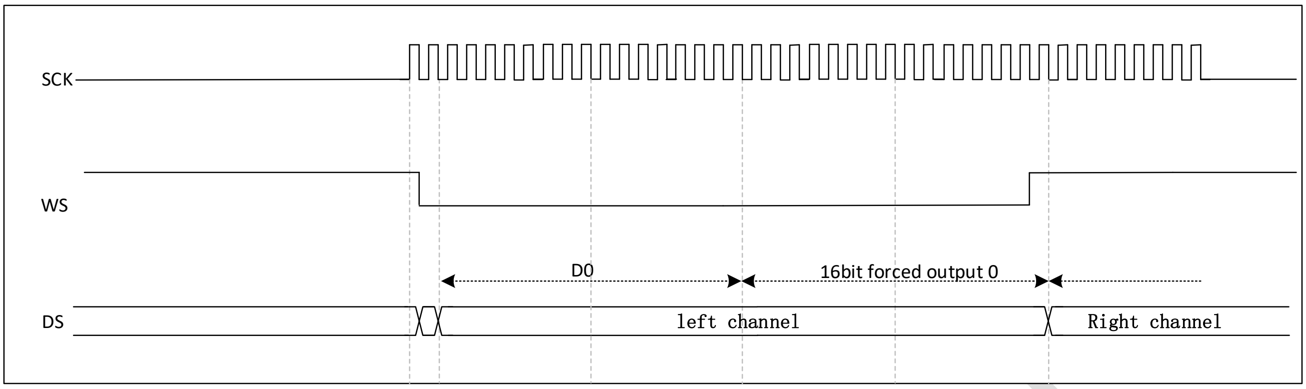


Figure 31-15 I2S Philips protocol standard waveform (16-bit expansion to 32-bit packet frames, CKPOL = 0)

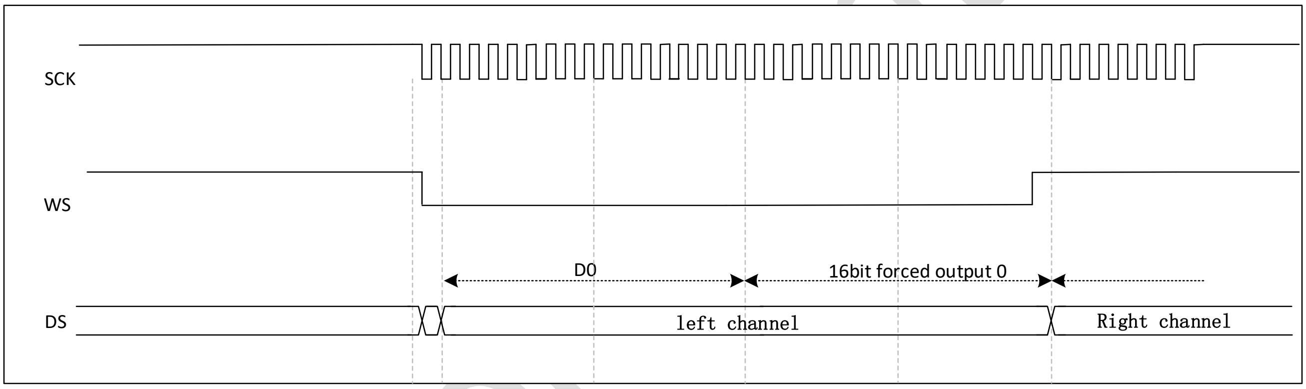


Figure 31-16 I2S Philips protocol standard waveform (16-bit expansion to 32-bit packet frames, CKPOL = 1)

During the I2S configuration phase, if you choose to extend 16-bit data to 32-bit sound frames, you only need to access the SPI_DR register once. the lower 16 bits used to extend to 32-bit are set to 0x0000 in hardware.

If the data to be transmitted or received is 0x76A3 (expansion to 32 bits is 0x76A30000), and then operate SPI_DR(0x76A3) only once

The MSB needs to be written to register SPI_DR when sending; a flag bit TXE of '1' indicates that new data can be written and an interrupt can be generated if the corresponding interrupt is allowed. Sending is done by hardware, and TXE is set and the corresponding interrupt is generated even if the last 16 bits of 0x0000 have not yet been sent. On reception, the flag bit RXNE is set to '1' each time a high 16-bit half-word (MSB) is received, and an interrupt can be generated if the corresponding interrupt is allowed.

This allows more time between the 2 reads and writes and prevents underflow or overflow from occurring.

31.4.2.2. MSB alignment standards

In this standard, the WS signal and the first data bit, the highest bit (MSB), are generated at the same time.

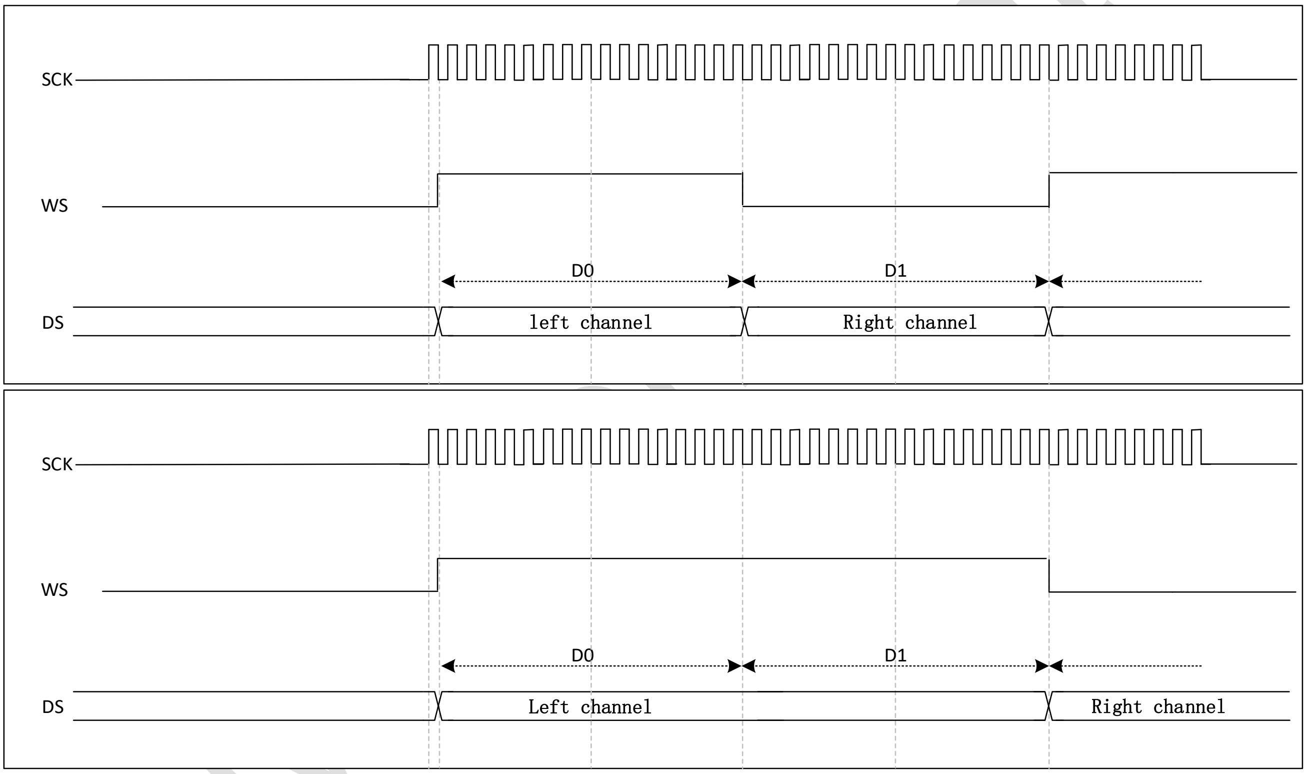


Figure 31-17 MSB-aligned 16-bit or 32-bit full precision,CPKOL = 0

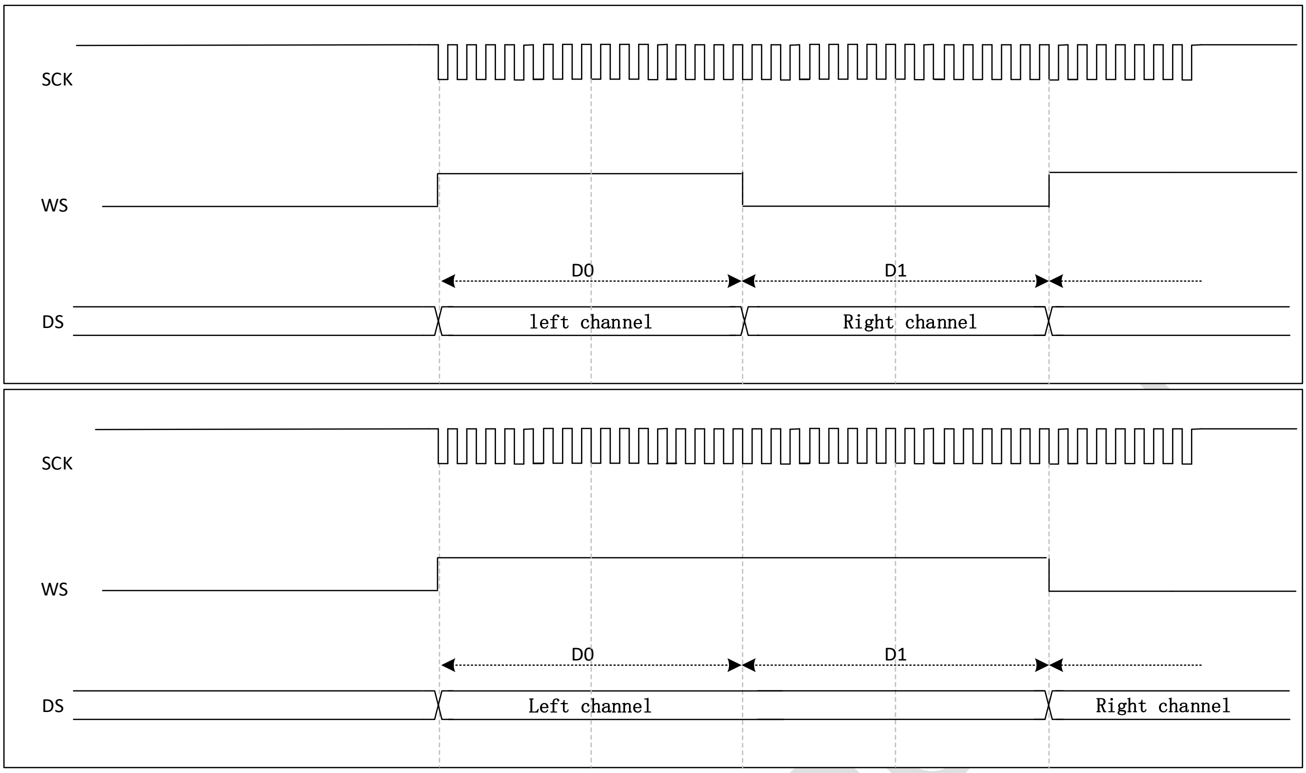


Figure 31-18 MSB-aligned 16-bit or 32-bit full precision ,CPKOL = 1

The sender changes the data on the falling edge of the clock signal (CK); the receiver is reading the data on the rising edge.

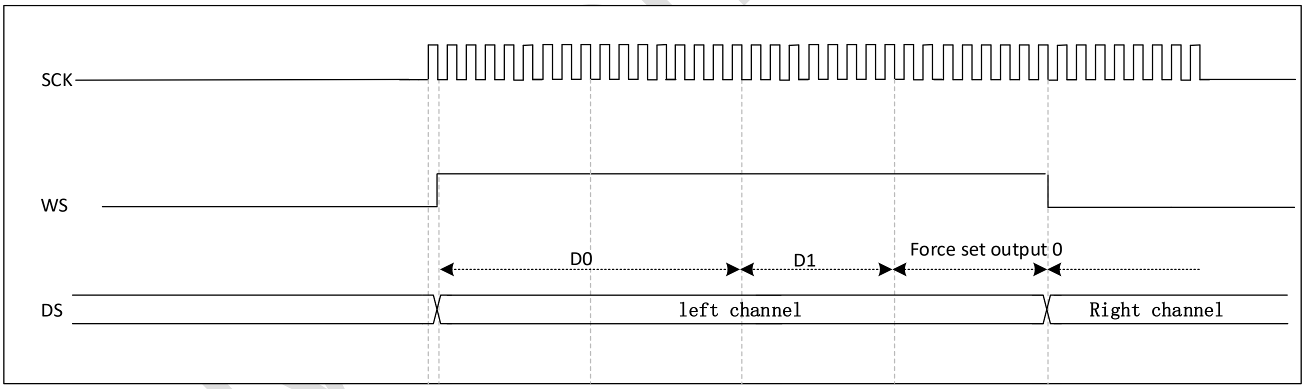


Figure 31-19 MSB-aligned 24-bit data， CKPOL = 0

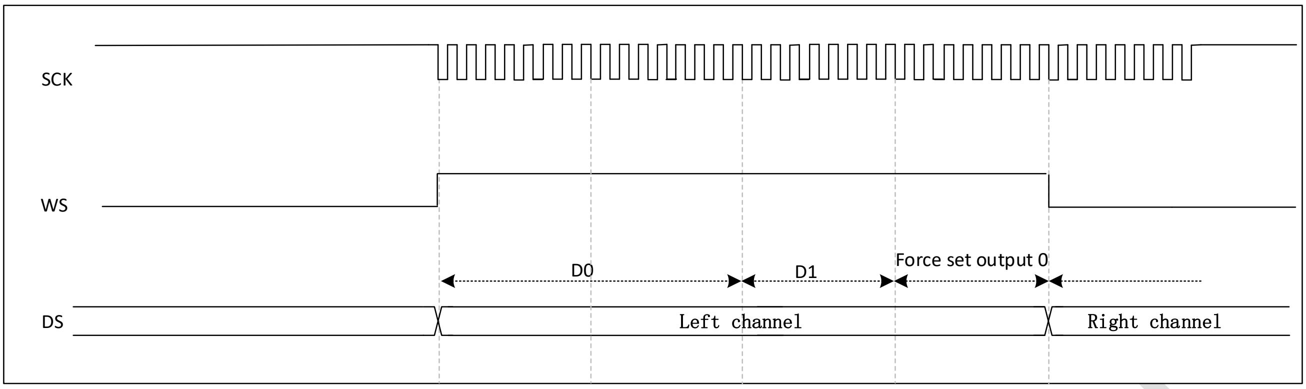


Figure 31-20 MSB-aligned 24-bit data， CKPOL = 1

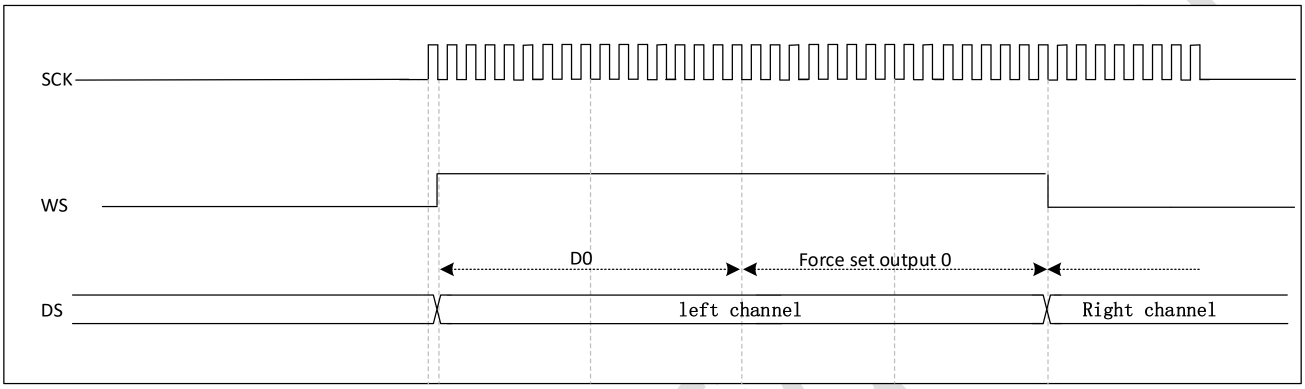


Figure 31-21 MSB aligned 16-bit data extension to 32-bit packet frames， CKPOL = 0

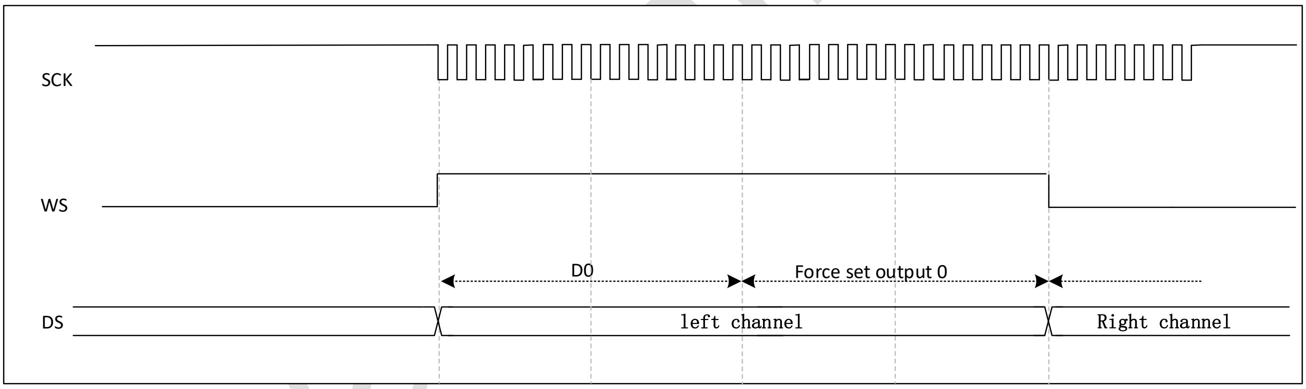


Figure 31-22 MSB aligned 16-bit data extension to 32-bit packet frames， CKPOL = 1

The next TXE event occurs as soon as valid data starts to be sent out from the SD pin. On receive, the RXNE event occurs as soon as valid data is received (not the 0x0000 part).

31.4.2.3. LSB alignment standards

This standard is similar to the MSB alignment standard (no difference in 16-bit or 32-bit full precision frame formats).

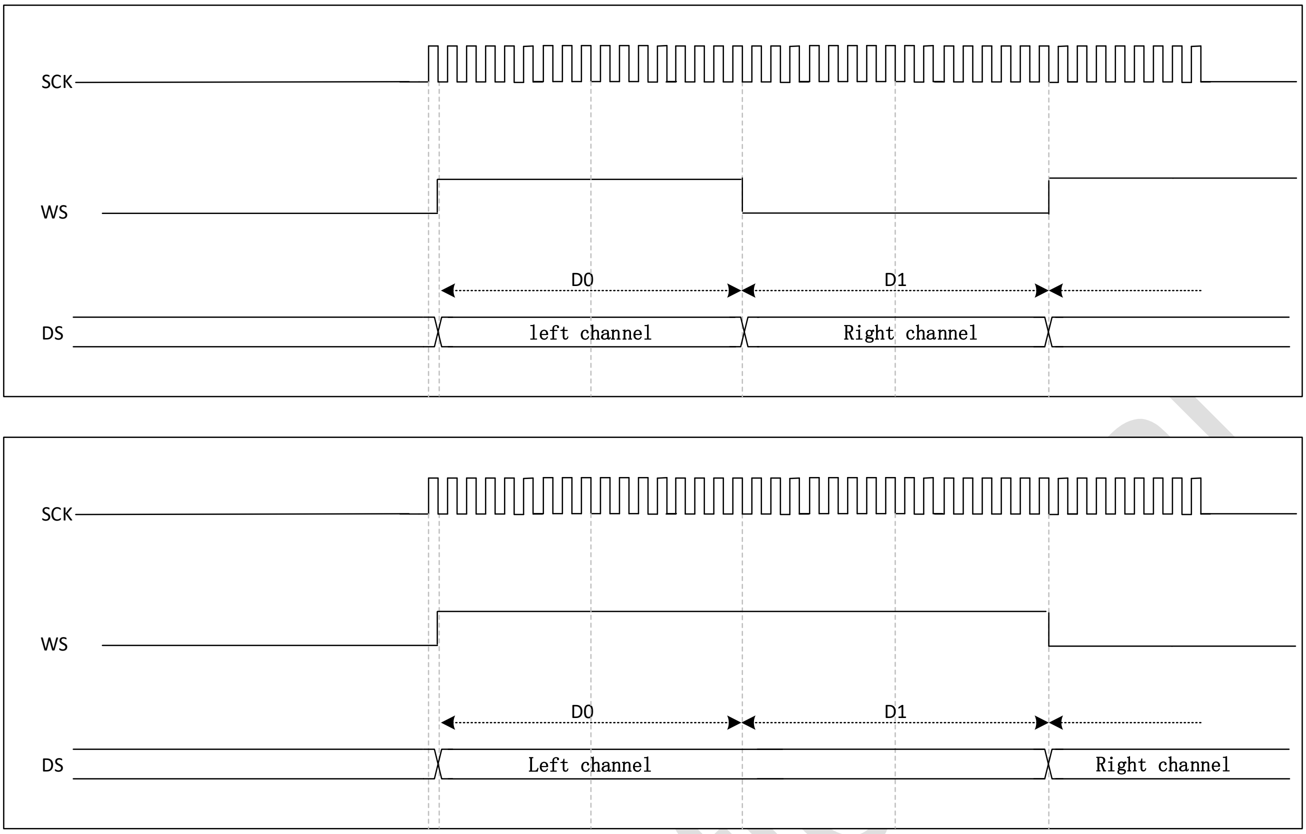


Figure 31-23 LSB-aligned 16-bit or 32-bit full precision， CKPOL = 0

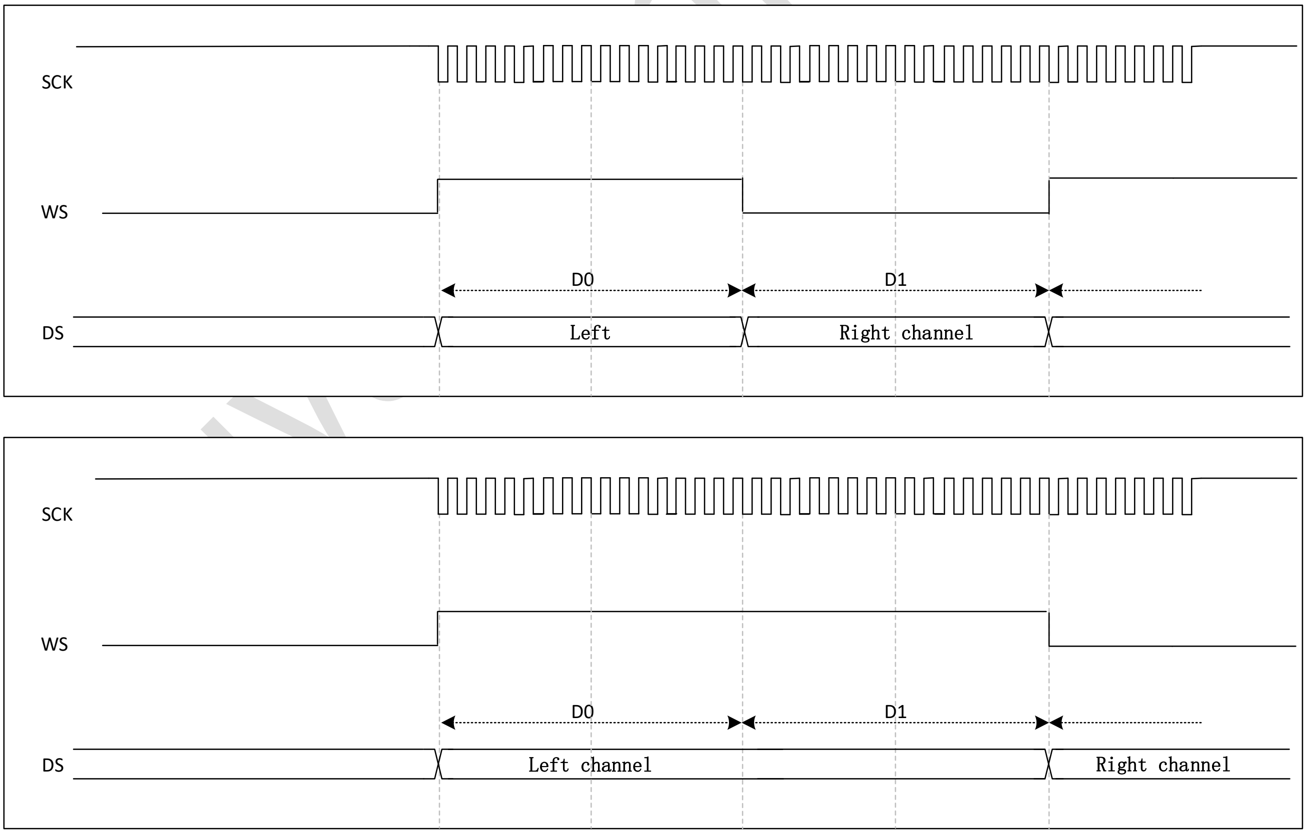


Figure 31-24 LSB-aligned 16-bit or 32-bit full precision， CKPOL = 1

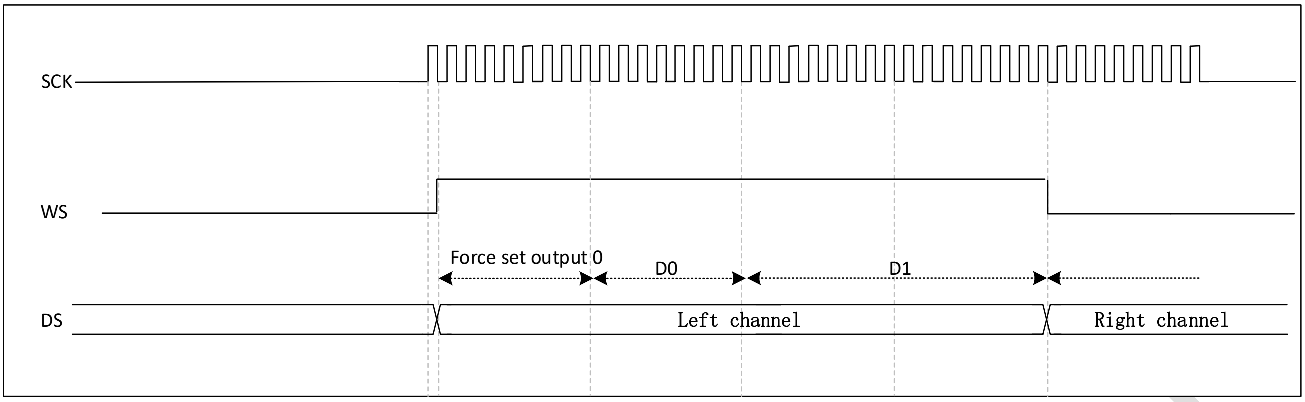


Figure 31-25 LSB-aligned 24-bit data， CKPOL = 0

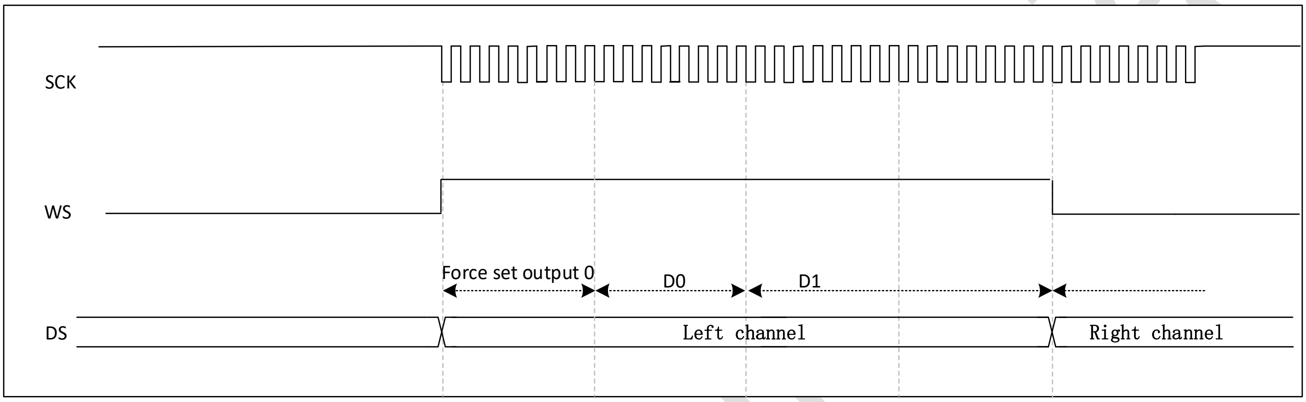


Figure 31-26 LSB-aligned 24-bit data， CKPOL = 1

1. In transmit mode

   To send data 0x3478AE, 2 write operations are required to the register SPI_DR by software or DMA. The operation to request sending 0x3478AE is as follows.

   1. TXE=1 when the first write to the data register 0xXX34. (Only the low 8 bits in the half word have meaning, the high 8 bits are forced to 0)
   2. TXE=1 when second write to data register 0x78AE

2. In reception mode

   To receive data 0x3478AE, it is required to perform 1 read operation on each of the 2 consecutive RXNE events to register SPI_DR. The operation required to receive 0x3478AE is as follows.

   1. TXE=1 when the data register 0x0034 is read for the first time.(Only the low 8 bits in the half word have meaning, the high 8 bits are forced to 0)
   2. TXE=1 when the second read of data register 0x78AE

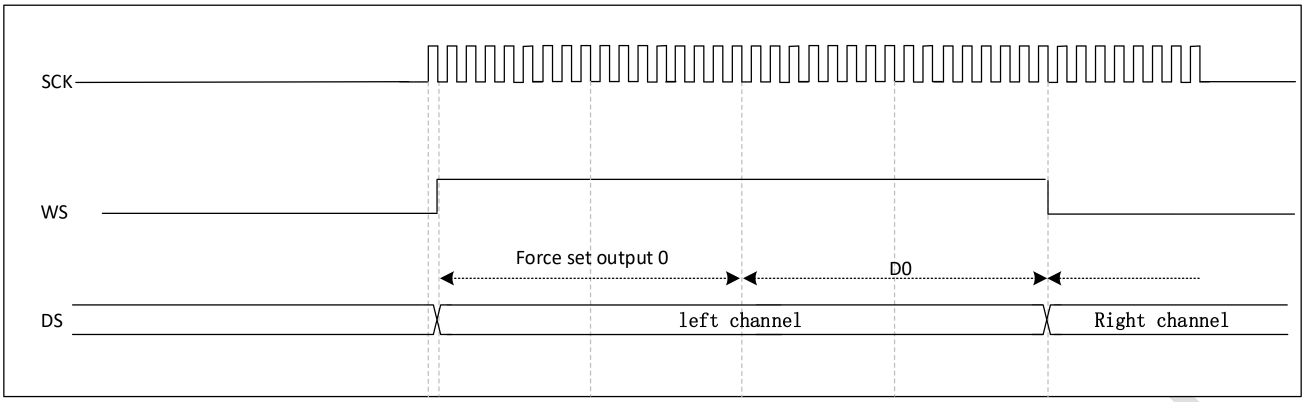


Figure 31-27 LSB-aligned 16-bit data extension to 32-bit packet frames， CKPOL = 0

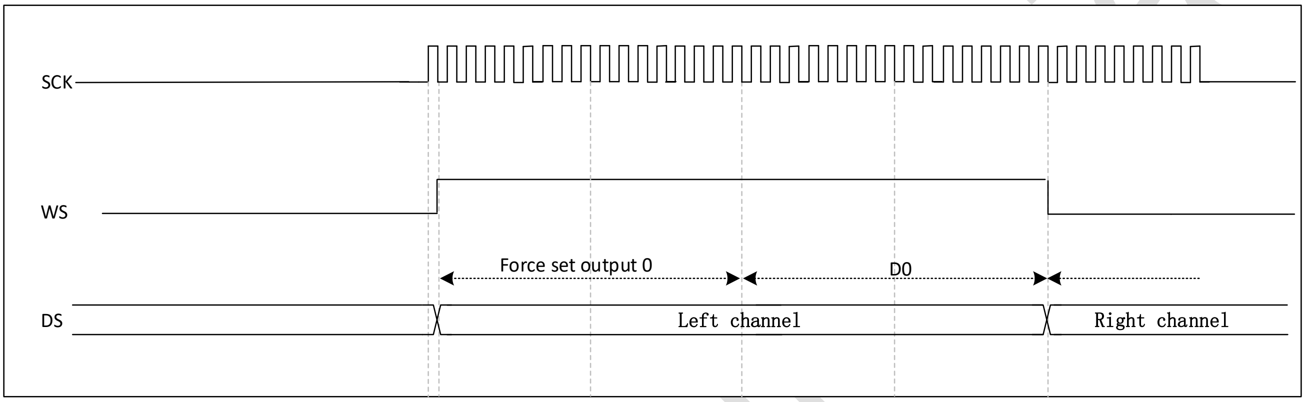


Figure 31-28 LSB-aligned 16-bit data extension to 32-bit packet frames， CKPOL = 1

During the I2S configuration phase, if you choose to extend the 16-bit data to a 32-channel frame, you only need to access the SPI_DR register once. At this point, the high half word (16-bit MSB) after the extension to 32 bits is set to 0x0000 by hardware.

If the data to be transmitted or received is 0x76A3 (extended to 32 bits is 0x0000 76A3), the required operation is shown below.

1. Only need to operate SPI_DR.(0x76A3) once.

When sending, if TXE is '1', the user needs to write the data to be sent (i.e. 0x76A3). The 0x0000, used to extend to 32 bits, is partially sent out by hardware first, and the next TXE event occurs once valid data starts to be sent out from the SD pin.

On receive, the RXNE event occurs as soon as valid data is received (not the 0x0000 part).

This allows more time between the 2 reads and writes and prevents underflow or overflow from occurring.

31.4.2.4. PCM standards

With the PCM standard, there is no information about the sound channel selection. The PCM standard has 2 available frame structures, short frame or long frame, which can be selected by setting the PCMSYNC bit in register SPI_I2SCFGR.

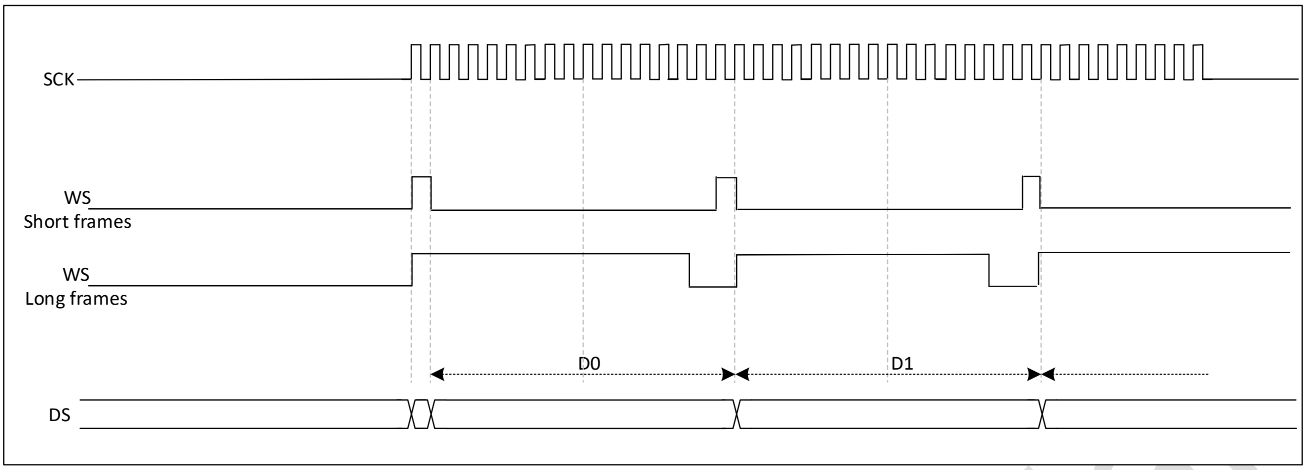


Figure 31-29 PCM standard waveform (16-bit， CKPOL=0)

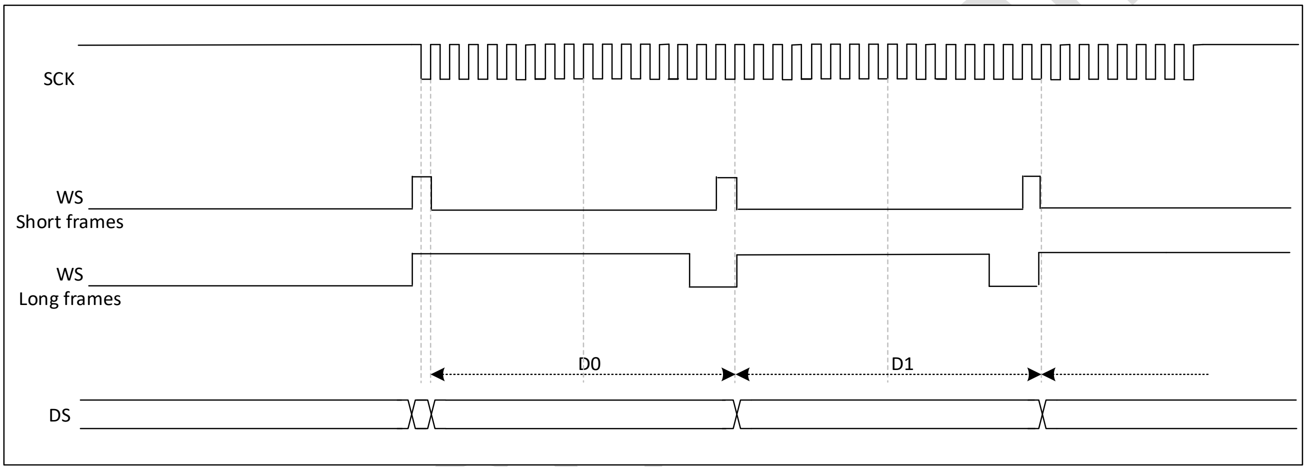


Figure 31-30 PCM standard waveform (16-bit， CKPOL=1)

For long frames, the WS signal used for synchronisation in master mode is valid for a fixed duration of 13 bits.

For short frames, the WS signal used to synchronise is only 1 bit long.

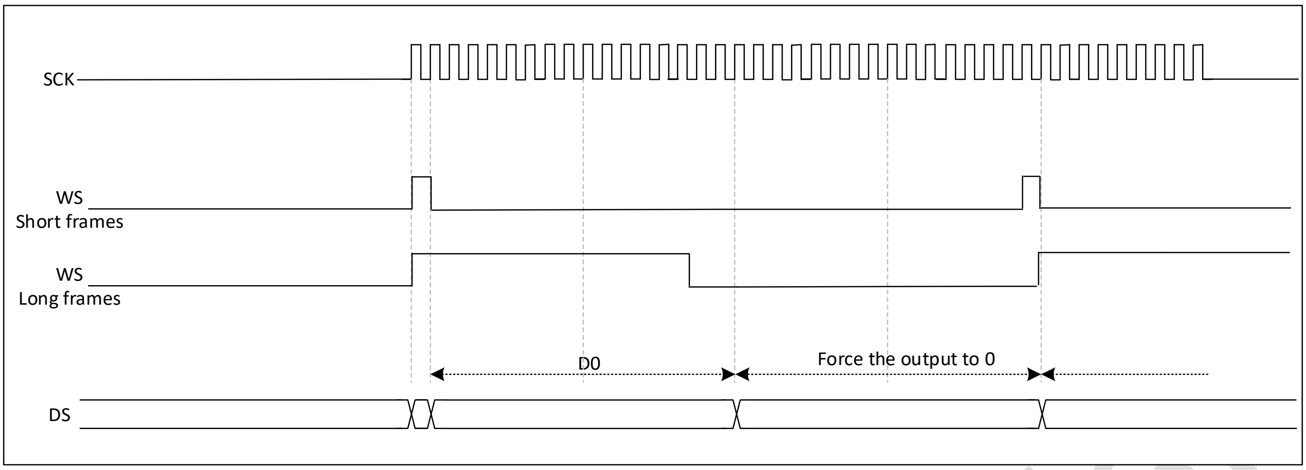


Figure 31-31 PCM standard waveforms (16-bit extended to 32-bit packet frames，CKPOL=0)

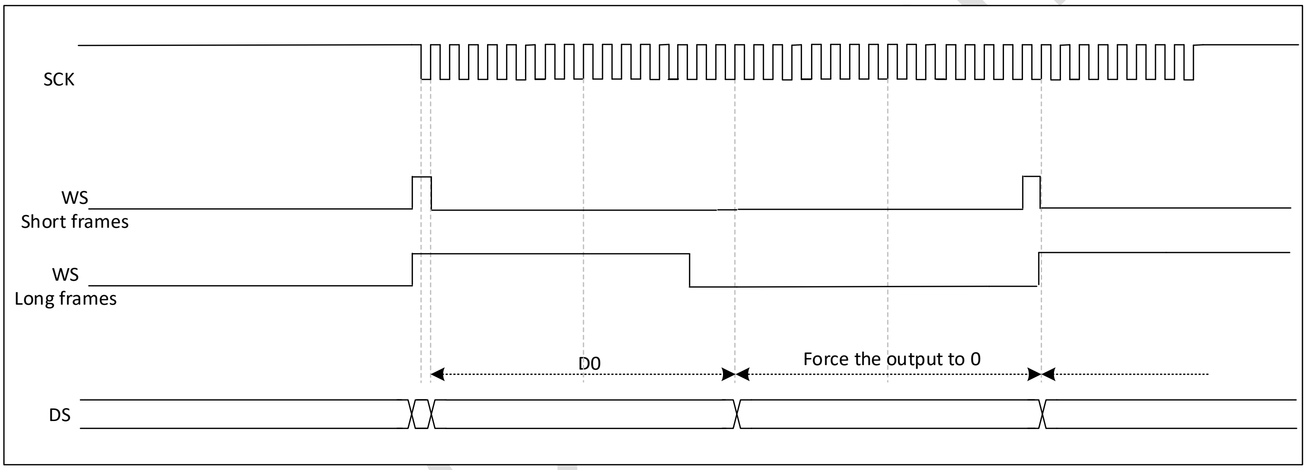


Figure 31-32 PCM standard waveforms (16-bit extended to 32-bit packet frames，CKPOL=1)

Note: Regardless of the mode (master or slave) and the synchronisation method (short or long frame), the time difference between two consecutive frames of data and between the two synchronisation signals (even in slave mode) needs to be determined by setting the DATLEN and CHLEN bits in the SPI_I2SCFGR register.

31.4.3. Clock generators

The bit rate of I2S i.e. determines the data flow on the I2S data line and the frequency of the I2S clock signal. That is

I2S bit rate = number of bits per channel x number of channels x audio sampling frequency

For an audio signal with left and right channels and 16 bits, the I2S bit rate is calculated as follows

I2S bit rate = 16 x 2 x Fs;

If the packet length is 32 bits, then we have: I2S bit rate = 32 x 2 x Fs

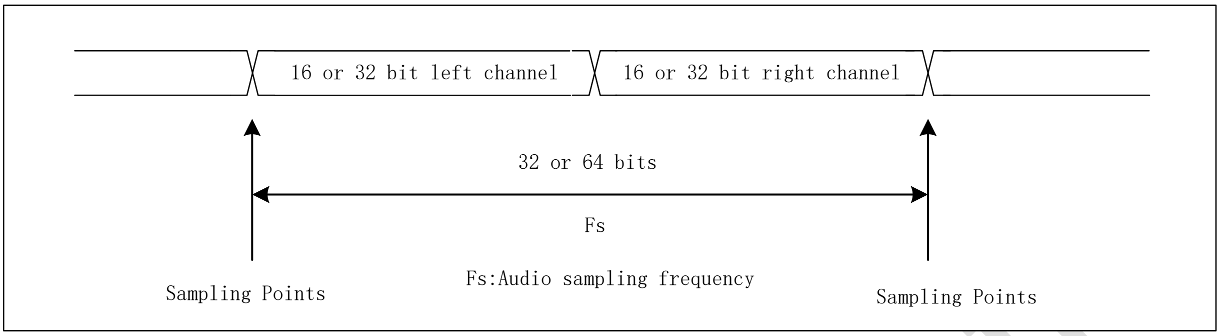


Figure 31-33 Audio Sample Definition

In the main mode, the linear crossover needs to be set correctly in order to obtain the desired audio frequency.

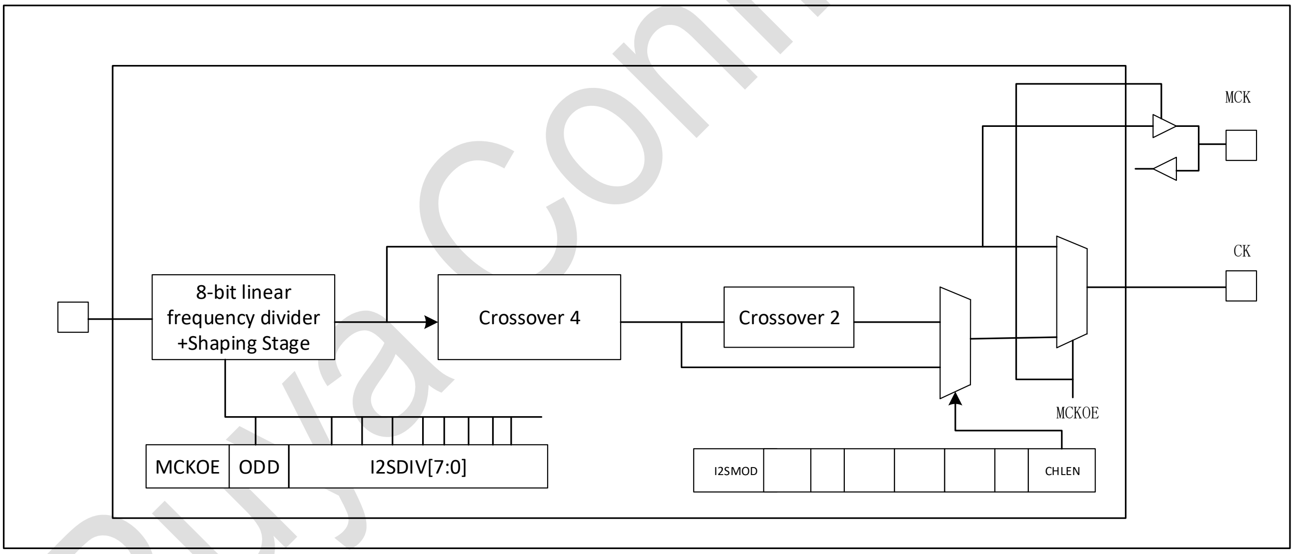


Figure 31-34 I2S clock generator architecture

The audio sampling frequency can be 96kHz, 48kHz, 44.1kHz, 32kHz, 22.05kHz, 16kHz, 11.025kHz or 8kHz (or any value in this range). To obtain the required frequency, a linear divider is set according to the following formula, when a master clock is to be generated (the MCKOE bit of register SPI_I2SPR is '1'):

When the frame length of the sound channel is 16 bits, Fs = I2SxCLK/[(16*2)*((2*I2SDIV)+ODD)*8]

When the frame length of the sound 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 sound channel is 16 bits, Fs = I2SxCLK/[(16*2)*((2*I2SDIV)+ODD)]

When the frame length of the sound channel is 32 bits, Fs=I2SxCLK/[(32*2)*((2*I2SDIV)+ODD)]

Accurate audio frequencies are obtained using the standard 8MHz HSE clock, see the following table:

Table 31-35 Frequency of audio

SYSCLK(MHZ)	I2S_DIV		I2S_ODD		MCLK	Expectation Fs(Hz)	Actual Fs(Hz)			
	16 bit	32 bit	16 bit	32 bit			16 bit	32 bit	16 bit	32 bit
72	11	6	1	0	None	96000	97826.09	93750	1.90%	2.34%
72	23	11	1	1	None	48000	47872.34	48913.04	0.27%	1.90%
72	25	13	1	0	None	44100	44117.65	43269.23	0.04%	1.88%
72	35	17	0	1	None	32000	32142.86	32142.86	0.45%	0.45%
72	51	25	0	1	None	22050	22058.82	22058.82	0.04%	0.04%
72	70	35	1	0	None	16000	15957.45	16071.43	0.27%	0.45%
72	102	51	0	0	None	11025	11029.41	11029.41	0.04%	0.04%
72	140	70	1	1	None	8000	8007.117	7978.723	0.09%	0.27%
72	2	2	0	0	Yes	96000	70312.5	70312.5	26.76%	26.76%
72	3	3	0	0	Yes	48000	46875	46875	2.34%	2.34%
72	3	3	0	0	Yes	44100	46875	46875	6.29%	6.29%
72	4	4	1	1	Yes	32000	31250	31250	2.34%	2.34%
72	6	6	1	1	Yes	22050	21634.62	21634.62	1.88%	1.88%
72	9	9	0	0	Yes	16000	15625	15625	2.34%	2.34%
72	13	13	0	0	Yes	11025	10817.31	10817.31	1.88%	1.88%
72	17	17	1	1	Yes	8000	8035.714	8035.714	0.45%	0.45%

31.4.4. I2S main mode

Set the I2S to operate in master mode, with the serial clock output from pin CK and the word select signal generated from pin WS. The master clock (MCK) can be selected to be output or not by setting the MCKOE bit of register SPI_I2SPR.

1) Set register I2SDIV[7:0] of SPI_I2SPR to define the serial clock baud rate that matches the audio sampling frequency. Also define the ODD bit of register SPI_I2SPR.
2) Set the CKPOL bits to define the level state of the clock for communication when it is idle. If a

master clock MCK is required to be supplied to an external DAC/ADC audio device, set the MCKOE position of register SPI_I2SPR to '1' and calculate the values of I2SDIV and ODD according to the different MCK output states.

3) Set the I2SMOD bit of register SPI_I2SCFGR to '1' to activate the I2S function, set the I2SSTD[1:0] and PCMSYNC bits to select the I2S standard used, and set CHLEN to select the number of data bits for each channel. Also set register I2SCFG[1:0] of SPI_I2SCFGR to select the I2S master mode and direction.
4) If required, the desired interrupt function and DMA function can be switched on by setting register SPI_CR2.
5) The I2SE position of register SPI_I2SCFGR must be set to '1'.
6) Pins WS and CK need to be configured for output mode. If the MCKOE bit of register SPI_I2SPR is '1', pin MCK should also be configured to output mode.

- Send flow

The transmit process starts when 1 half word (16 bits) of data is written to the transmit cache.

It is assumed that the first data written to the transmit buffer corresponds to the left channel data. When the data is moved from the transmit cache to the shift register, the flag bit TXE is set to '1', at which point the data corresponding to the right channel is to be written to the transmit cache. The flag bit CHSIDE indicates which channel corresponds to the data currently to be transferred. The value of the flag bit CHSIDE is updated when TXE is '1', so it has meaning when TXE is '1'. A complete data frame is not considered to be complete until all the data for the left channel and then the right channel have been transmitted. It is not possible to transmit only partial data frames, e.g. data for the left channel only.

While the first bit of data is sent, the half-word data is transferred in parallel to the 16-bit shift register and then the subsequent bits are sent from pin MOSI/SD in the order of high first. Each time data is moved from the transmit buffer to the shift register, the flag bit TXE is set to '1' and an interrupt is generated if the TXEIE bit in register SPI_CR2 is '1'.

The operation of writing data depends on the selected I2S standard. To ensure continuous audio data transfer, it is recommended to write the next data to be transferred to register SPI_DR before the current transfer is completed.

It is recommended to wait for the flag bits TXE=1 and BSY=0 before clearing the I2SE bit '0' when the I2S function is to be switched off.

- Receive flow

The configuration steps for the receive flow are the same as for the transmit flow except for point 3 (see "Transmit flow" above), which is done by configuring

I2SCFG[1:0] to select the main receive mode.

Audio data is always received in 16-bit packets, regardless of the data and channel length. That is, the flag bit RXNE is set to '1' each time the receive buffer is filled, and an interrupt is generated if the RXNEIE bit in register SPI_CR2 is '1'. Depending on the configured data and channel length, receiving data for the left or right channel will require one or two transfers of data into the receive buffer. The RXNE flag is cleared by a read of the SPI_DR register. The CHSIDE is updated after each reception and its value depends on the WS signal generated by the I2S unit. The read data operation depends on the selected I2S standard.

If the previous received data has not yet been read and new data is received, i.e. an overflow occurs, 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 .

To switch off the I2S function, a special operation needs to be performed to ensure that the I2S module can complete the transfer cycle normally without starting a new data transfer. The operation procedure is related to the data configuration and channel length, as well as to 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)
- Wait for penultimate (n-1) RXNE=1;

- Wait 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 bit) 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, turn off I2S using the following:
- Wait for the penultimate (n-1) RXNE = 1;
- Wait for one I2S clock cycle (using software delay);
- turn off I2S (I2SE=0).

Note: The BSY flag is always low during transmission.

31.4.5. I2S slave mode

In slave mode, I2S can be set to transmit and receive modes. The configuration of the slave mode follows basically the same procedure as the configuration of the master mode. In slave mode, the I2S interface is not required to provide a clock. Both the clock signal and the WS signal are provided by the external master I2S device, connected to the corresponding pins. Therefore there is no need for the user to configure the clock.

- Set the I2SMOD bits of register SPI_I2SCFGR to activate the I2S function; set 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. Set register I2SCFG[1:0] of SPI_I2SCFGR to select the data direction of the I2S slave mode.
- Set register SPI_CR2 to turn on the required interrupt function and DMA function, as required.
- The I2SE bit of register SPI_I2SCFGR must be set to '1'.
- Transmit Flow

The transmit process starts when the external master device sends the clock signal and when the NSS_WS signal requests data transfer. The slave device must be enabled and the I2S data register written before the external master device can start communicating.

For the MSB-aligned and LSB-aligned modes of I2S, the first data item written to the data register corresponds to the data of the left channel. When communication starts, data is transferred from the transmit buffer to the shift register and then the flag bit TXE is set to '1'; at this point, the data item corresponding to the right channel is to be written to the I2S data register.

The flag bit CHSIDE indicates which channel corresponds to the data currently to be transmitted. In contrast to the transmit process in master mode, in slave mode CHSIDE depends on the WS signal from the external master I2S. This means that the slave I2S prepares the first data to be sent before it receives the clock signal generated by the master. A WS signal of '1' means that the left channel is sent first.

Note: The time to set the I2SE bit to '1' should be at least 2 PCLK clock cycles earlier than the master I2S clock signal on the CK pin.

When the first bit of data is sent, the half-word data is transferred in parallel via the I2S internal bus to the 16-bit shift register, and then the other bits are sent from pin MOSI/SD in order of higher first. Each time data is transferred from the transmit buffer to the shift register, the flag bit TXE is set to '1' and an interrupt is generated if the TXEIE bit in register SPI_CR2 is '1'. Note that the flag bit TXE is confirmed to be '1' before writing data to the transmit buffer. The operation of writing data depends on the selected I2S standard.

To ensure continuous audio data transfer, it is recommended to write the next data to be transferred to register SPI_DR before the current transfer is completed. If the 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 a software send data error. If the ERRIE bit of register SPI_CR2 is '1', an interrupt will be generated when the flag bit UDR of register

SPI_SR is high. It is recommended to switch off I2S at this point 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 turn off I2S.

- Receive flow

The configuration steps are the same as the transmit process except for point 1. The main receive mode needs to be selected by configuring I2SCFG[1:0].

Regardless of the data and channel length, audio data is always received in 16-bit packets, i.e. each time the receive buffer is filled, the flag bit RXNE is set to '1' and an interrupt is generated if the RXNEIE bit in register SPI_CR2 is '1'. Depending on the data and channel length settings, receiving left or right channel data will require one or two transfers of data to the receive buffer. The CHSIDE is updated each time data is received (to be read from SPI_DR), which corresponds to the WS signal generated by the I2S unit. Reading the SPI_DR register will clear the RXNE bit. The read data operation depends on the selected I2S standard, see section 5.2 for details.

An overflow is generated when new data is received before the previous received data has been read, and the flag bit OVR is set to '1'; if the ERRIE bit of 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 via the audio channel.

31.4.6. Status flag bits

There are 3 status flag bits for the user to monitor the status of the I2S bus.

31.4.6.1. Busy flag bit (BSY)

The BSY flag is set and cleared by hardware (writing this bit has no effect) and this flag bit indicates the status of the I2S communication layer. A '1' indicates that I2S communication is in progress, with one exception: in main receive mode (I2SCFG=11), the BSY flag is always low during receive. The BSY flag can be used to detect the end of a transmission before the software wants to shut down the

SPI module. This will avoid corrupting the last transmission and therefore the following procedure needs to be followed strictly. The BSY flag is set to '1' when a transmission is started, unless the I2S module is in main receive mode.

This flag bit is cleared in the following cases:

a) when the transmission is finished (except in the main transmit mode, in which case the communication is continuous);

b) when the I2S module is switched off.

c) When communication is continuous:

d) in master transmit mode, the BSY flag is always high during the entire transmission;

e) In slave mode, the BSY flag becomes low for 1 I2S clock cycle between each data item transmission.

31.4.6.2. Transmit buffer empty flag bit (TXE)

This flag bit is '1' to indicate that the transmit buffer is empty and that new data to be sent can be written to the transmit buffer. The flag bit is cleared to '0' when there is already data in the transmit buffer. The flag bit is also '0' when I2S is switched off (I2SE bit is '0').

31.4.6.3. Receive buffer not empty flag bit (RXNE)

This flag position '1' indicates that there is valid data received in the receive cache. This bit is cleared to '0' when the SPI_DR register is read.

31.4.6.4. Sound channel flag bit (CHSIDE)

In transmit mode, this flag bit is refreshed when TXE is high, indicating the sound channel on which the data sent from the SD pin is located. If an underflow error occurs in slave mode, the value of this flag bit is invalid and the I2S needs to be turned off and on again before communication can be restarted. In receive mode, this flag bit is refreshed when data is received in register SPI_DR, indicating the channel where the received data is located. If an error occurs (e.g. overflow OVR), this flag bit is meaningless and the I2S needs to be turned off and then on again (also, if necessary, the I2S configuration needs to be modified).

Under the PCM standard, this flag bit is meaningless in either short or long frame format.

If the flag bit OVR or UDR of register SPI_SR is '1' and the ERRIE bit of register SPI_CR2 is '1', an interrupt will be generated and the interrupt flag can later be cleared by reading register SPI_SR.

31.4.7. Error flag bits

The I2S unit has 2 error flag bits.

31.4.7.1. Underflow flag bit (UDR)

In slave transmit mode, this flag bit is set to '1' if new data is still not written to the SPI_DR register when the first clock edge of the data transfer arrives. This flag bit is valid after the I2SMOD position '1' of register SPI_I2SCFGR. If the ERRIE bit of register SPI_CR2 is '1', an interrupt will be generated. This flag bit is cleared by performing a read operation on register SPI_SR.

31.4.7.2. Overflow flag bit (OVR)

If new data is received while the previous received data has not been read, an overflow is generated, this flag bit is '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 point, the contents of the receive cache, will not be refreshed to the new data sent from the transmitting device. A read operation on register SPI_DR returns the last correctly received data. All other 16-bit data sent by the transmitting device after the overflow has occurred is lost. This flag bit is cleared by reading register SPI_SR and then register SPI_DR.

31.4.8. I2C interrupts

Table 31-2 I2C interrupt requests

Interrupt event	Event flag bit	Enable flag bit
Transmit buffer empty	TxE	TXEIE
Receive buffer not empty	RxNE	RXNEIE
Overflow	OVR	ERRIE
Underflow	UDR	ERRIE

31.4.9. DMA function

Same as SPI except for CRC function, as there is no data transfer protection system in I2S mode.

31.5. SPI and I2S registers

The SPI counterpart registers can be accessed by16-bit.

31.5.1. SPI control register 1 (SPI_CR1) (not used in I2S mode)

Address offset:0x00

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW	~~RW~~	~~RW~~	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: selects the "two-wire bidirectional" mode; 1: Selects "single line bi-directional" mode. Note: Not used in I2S mode.
14	BIDIOE	RW	0	Output enable in bi-directional mode Together with the BIDIMODE bit determines the direction of data output in "single line bidirectional" mode 0: output disable (receive-only mode); 1: output enable (send-only mode). This "single wire" data line is the MOSI pin on the master side and the MISO pin on the slave side. Note: Not used in I2S mode.
13	CRCEN	RW	0	Hardware CRC checksum enable 0: disables CRC calculation; 1: Enable CRC calculation. Note: This bit can only be written when SPI is disabled (SPE=0), otherwise an error occurs. This bit can only be used in full duplex mode. Note: Not used in I2S mode.
12	CRCNEXT	RW	0	Next send CRC 0: The next sent value comes from the send buffer.

Bit	Name	R/W	Reset Value	Function
				1: The next send value comes from the send CRC register. Note: This bit should be set immediately after the last data is written to the SPI_DR register. Note: Not used in I2S mode.
11	DFF	RW	0	Data frame format 0: transmit/receive using an 8-bit data frame format; 1: transmit/receive using 16-bit data frame format. Note: This bit can only be written when SPI is disabled (SPE=0), otherwise an error occurs. Note: Not used in I2S mode.
10	RXONLY	RW	0	Receive only This bit, together with the BIDIMODE bit, determines the direction of transmission in "two-wire unidirectional" mode. In a multiple slave configuration, this bit is set to 1 on a slave device that is not being accessed, so that only the accessed slave device has an output and therefore no data conflicts on the data lines are caused. 0: full duplex (transmit and receive) 1: output disabled (receive only mode) Note: Not used in I2S mode.
9	SSM	RW	0	Software Slave Device Management When SSM is set, the level on the NSS pin is determined by the value of the SSI bit. 0: Software slave management disabled 1: Enables software slave device management Note: Not used in I2S mode.
8	SSI	RW	0	Internal slave select This register is only valid when SSM=1. This register determines the level on the NSS, and I/O operations on the NSS pin are not valid. Note: Not used in I2S mode.
7	LSBFIRST	RW	0	Frame format 0: MSB sent first 1: LSB is sent first The value of this register cannot be changed while communication is in progress. Note: Not used in I2S mode.

Bit	Name	R/W	Reset Value	Function
6	SPE	RW	0	SPI enable 0: SPI disabled 1: SPI enable Note: Not used in I2S 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 The value of this register cannot be changed while communication is in progress. Note: Not used in I2S mode.
2	MSTR	RW	0	Master device selection 0: Configured as a slave device 1: configured as a master device The value of this register cannot be changed while communication is in progress. Note: Not used in I2S mode.
1	CPOL	RW	0	Clock polarity 0: SCK low in idle state 1: SCK is held high in the idle state The value of this register cannot be changed while communication is in progress. Note: Not used in I2S mode.
0	CPHA	RW	0	Clock phase 0: data sampled from the first clock edge 1: Data is sampled from the second clock edge The value of this register cannot be changed while communication is in progress. Note: Not used in I2S mode.

31.5.2. SPI Control Register 2 (SPI_CR2)

Address offset:0x04

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	TXDMAEN	RXDMAEN
								RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	-	Reserved
7	TXEIE	RW	0	Transmit buffer interrupt enable 0: TXE interrupt disabled 1: TXE interrupt enable. interrupt request is generated when TXE=1.
6	RXNEIE	RW	0	Receive buffer off-air interrupt enable 0: disable RXNE interrupt 1: Enable RXNE interrupt. interrupt request is generated when RXNE=1.
5	ERRIE	RW	0	Error interrupt enable This bit controls whether an interrupt is generated when an error (CRCERR, OVR, MODF) is generated 0: error interrupt disabled 1: Enables the error interrupt.
4	CLRTXFIFO	RW	0	Clear TXFIFO Software set, hardware reset 0: No effect 1: Clear TXFIFO Note: This bit can only be written when SPI is disabled (SPE=0), otherwise it is not valid.
3	Reserved	-	-	Reserved
2	SSOE	RW	0	SS output enable 0: disable SS output in master mode, the device can work in multi-master mode 1: Enable SS output in master mode, the device cannot work in multi-master device mode.
1	TXDMAEN	RW	0	Transmit Buffer DMA Enable When this bit is set, a DMA request is issued once the TXE flag is set 0: Disable transmit buffer DMA 1: Enables transmit buffer DMA.
0	RXDMAEN	RW	0	Receive buffer DMA enable

Bit	Name	R/W	Reset Value	Function
				When this bit is set, a DMA request is issued once the RXNE flag is set 0: Receive buffer DMA is disabled 1: Enables receive buffer DMA.

31.5.3. 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			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 send level. hardware set, hardware clear 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full (when FIFO threshold is greater than 1/2, it is considered full) Note: This bit is not used in I2S mode.
10:9	FRLVL	R	0	FIFO receive level. hardware set, hardware clear 00: FIFO empty 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO full Note: These bits are not used in I²S mode and SPI Receive Only mode with CRC checksum.
8	Reserved	-	-	Reserved
7	BSY	R	0	Busy flag. 0: SPI is not busy; 1: SPI is in communication, or the transmit buffer is not empty. This bit is set or reset by hardware.
6	OVR	R	0	Overflow flags 0: no overflow error has occurred; 1: an overflow error has occurred.

Bit	Name	R/W	Reset Value	Function
				This bit is set by hardware and reset by the software sequence. (The up and down overflow sequences are different).
5	MODF	R	0	Mode errors 0: no mode error has occurred; 1: A mode error has occurred. This bit is set by hardware and reset by software sequence. Note: Not used in I2S mode.
4	CRCERR	RC_W0	0	CRC error flag 0: The received CRC value matches the value in the SPI_RXCRCR register; 1: The received CRC value and the value in the SPI_RXCRCR register do not match. This bit is set by hardware and reset by a software write '0'. Note: Not used in I2S mode.
3	UDR	R	0	Underrun flag 0: underrun has not occurred; 1: Underrun has occurred. This flag is set to '1' by hardware and cleared to '0' by a software sequence. Note: Not used in SPI mode.
2	CHSIDE	R	0	Sound channel 0: transmission or reception of the left channel is required; 1: Transmission or reception of the right channel is required. Note: Not used in SPI mode. Not meaningful in PCM mode.
1	TXE	R	1	The send buffer is empty. 0: Send buffer non-empty 1: Send buffer empty
0	RXNE	R	0	Receive buffer is non-empty. 0: Receive buffer non-empty 1: Receive buffer is empty

31.5.4. 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															

Bit	Name	R/W	Reset Value	Function
15:0	DR[15:0]	RW	0	Data register. The data to be sent or received. The data registers act as an interface between the RxFIFO and the TxFIFO. When data is to be read, the RxFIFO is actually accessed, while to write data, the TxFIFO is actually accessed. Note: Data is always right-aligned. Unused bits are ignored when writing to registers and read to zero when reading registers. The Rx threshold setting must always correspond to the read access currently in use.

31.5.5. 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															

Bit	Name	R/W	Reset Value	Function
15:0	CRCPOLY[15:0]	RW	0x7	CRC polynomial register This register contains the polynomials used in CRC calculations. The reset value is 0x0007, other values can be set depending on the application. Note: Not used in I2S mode. Note: Polynomial values can only be odd, even values are not supported.

31.5.6. SPI Rx CRC register (SPI_RXCRCR)

Address offset:0x14

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

RxCRC[15:0]
R

Bit	Name	R/W	Reset Value	Function
15:0	RxCRC[15:0]	R	0	Receive CRC register When CRC calculation is enabled, RXCRC[15:0] contains the CRC value calculated based on the received byte. This register is reset when a '1' is written to the CRCEN bit of SPI_CR1. The CRC is calculated using the polynomial in SPI_CRCPR. When the data frame format is set to 8 bits, only the lower 8 bits are involved in the calculation and follow the CRC8 method; when the data frame format is 16 bits, all 16 bits in the register are involved in the calculation and follow the CRC16 standard. Note: Reading this register when the BSY flag is '1' will likely read an incorrect value. Note: Not used in I2S mode.

31.5.7. SPI Tx CRC register (SPI_TXCRCR)

Address offset:0x18

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
15:0	TxCRC[15:0]	R	0	Send CRC register When CRC calculation is enabled, TXCRC[15:0] contains the CRC value calculated based on the byte to be sent. This register is reset when a '1' is written to the CRCEN bit in SPI_CR1. The CRC is calculated using the polynomial in SPI_CRCPR. When the data frame format is set to 8 bits, only the lower 8 bits are involved in the calculation and follow the CRC8 method; when the data frame format is 16 bits, all 16 bits in the register

Bit	Name	R/W	Reset Value	Function
				are involved in the calculation and follow the CRC16 standard. Note: Reading this register when the BSY flag is '1' will likely read an incorrect value. Note: Not used in I2S mode.

31.5.8. 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				I2SMOD	I2SE	I2SCFG		PCMSYNC	Res	I2SSTD		CKPOL	DATLEN		CHLEN
				RW	RW	RW		RW		RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
15:12	Reserved	-	-	Reserved
11	I2SMOD	RW	0	I2S mode selection 0: Selects SPI mode; 1: Selects I2S mode. Note: This bit can only be set when SPI or I2S is switched off.
10	I2SE	RW	0	I2S enable 0: I2S off; 1: I2S enable. Note: Not used in SPI mode.
9:8	I2SCFG	RW	0	I2S mode setting 00: transmission from the device; 01: slave device receiving; 10: Master device transmit; 11: Master device receive. Note: This bit can only be set when I2S is switched off. It is not used in SPI mode.
7	PCMSYNC	RW	0	PCM frame synchronization 0: short frame synchronisation; 1: Long frame synchronisation. Note: This bit is only relevant when I2SSTD = 11 (using PCM standard). It is not used in SPI mode.

Bit	Name	R/W	Reset Value	Function
6	Reserved	-	-	Reserved
5:4	I2SSTD	RW	0	I2S standard selection 00: I2S Philips standard; 01: High byte alignment standard (left-aligned); 10: Low byte alignment standard (right-aligned); 11: PCM standard. Note: For correct operation, this bit can only be set when I2S is switched off. Not used in SPI mode.
3	CKPOL	RW	0	Stationary clock polarity 0: I2S clock quiescent low; 1: I2S clock quiescent state is high. Note: For correct operation, this bit can only be set when I2S is switched off. It is not used in SPI mode.
2:1	DATLEN	RW	0	Length of data to be transferred 00: 16-bit data length; 01: 24-bit data length; 10: 32-bit data length; 11: Not allowed. Note: For correct operation, this bit can only be set when I2S is switched off. It is not used in SPI mode.
0	CHLEN	RW	0	Channel length (number of data bits per audio channel) 0: 16 bits wide; 1: 32 bits wide. The write operation of this bit is only meaningful if DATLEN = 00, otherwise the channel lengths are all fixed by hardware to 32 bits. Note: For correct operation, this bit can only be set when I2S is switched off. It is not used in SPI mode.

31.5.9. SPI_I2S Prescaler Register (SPI_I2SPR)

Address offset:0x20

Reset value:0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						MCKOE	ODD	I2SDIV							

	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	-	Reserved
9	MCKOE	RW	0	Master device clock output enable 0: disables the master device clock output; 1: Master device clock output enable. Note: For correct operation, this bit can only be set when I2S is switched off. This bit is only used in I2S master device mode. Not used in SPI mode.
8	ODD	RW	0	Odd factor prescaling 0: actual crossover factor = I2SDIV * 2; 1: Actual crossover factor = (I2SDIV * 2) + 1. Note: For correct operation, this bit can only be set when I2S is switched off. This bit is only used in I2S master device mode. Not used in SPI mode.
7:0	I2SDIV	RW	0	I2S linear prescaling Disable setting I2SDIV [7:0] = 0 or I2SDIV [7:0] = 1 Note: For correct operation, this bit can only be set when I2S is switched off. This bit is only used in I2S master device mode. Not used in SPI mode.

31.5.10. SPI register map

Off-set	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SPI_CR1	BI-	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPI_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	CLRTXFIF	Res.	SSOE	TXDMAE	RXDMAEN
	Reset value									0	0	0	0		0	0	0
0x08	SPI_SR	Res.	Res.	Res.	FTLVL[1:0]		FRLVL[1:0]		Res.	BSY	OVR	MODEF	CRCERR.	UDR.	CHSIDE.	TXE	RXNE
	Reset value				0	0	0	0		0	0	0	0	0	0	1	0

Off-set	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0C	SPI_DR	DR[15:0]															
	Reset value	0															
0x10	SPI_CRCPR	CRCPOLY[15:0]															
	Reset value	0															
0x14	SPI_RxCRC	RxCRC[15:0]															
	Reset value	0															
0x18	SPI_TxCRC	TxCRC[15:0]															
	Reset value	0															
0x1C	SPI_CFGR	Res	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG		PCM-SYNC	Res.	I2SSTD		CKPOL	DATLEN		CHLEN
	Reset value					0	0	0		0		0		0	0		0
0x20	SPI_I2SPR	Res	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV							
	Reset value							0	0	0							

32. Universal synchronous asynchronous receiver transmitter (USART)

32.1. Introduction

The universal synchronous asynchronous receiver transmitter (USART) offers a flexible means of Full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The USART offers a very wide range of baud rates using a programmable baud rate generator.

It supports synchronous one-way communication and Half-duplex Single-wire communication, as well as multiprocessor communications.

High speed data communication is possible by using the DMA (direct memory access) for multibuffer configuration.

32.2. USART main features

- 2 USARTs supporting full functionality (USART1 and USART2), two USARTs not supporting lin, scen, irda (USART3 and USART4)
- Full duplex asynchronous communication
- Shared programmable baud rate for transmit and receive up to 4.5 Mbit/s
- Configurable data word length (8 or 9 bits)
- Configurable stop bits - 0.5, 1, 1.5 or 2 stop bits supported
- Sender provides clock for synchronous transmission
- Single line half duplex communication
- Separate transmitter and receiver enable bits
- Parity control
  - Transmit parity bit
  - Checksumming of received data
- Detect flag

  - Receive buffer full
  - Send buffer empty
  - End-of-transmission flag
- Ability of Lin master to send synchronous disconnects and Lin slave to detect disconnects
  - When USART hardware is configured as LIN, 13-bit disconnects are generated and 10/11 disconnects are detected
- IRDA SIR encoder decoder
  - Supports 3/16 bit duration in normal mode
- Smartcard emulation function
  - Smart card interface supports the asynchronous smart card protocol as defined in ISO7816-3
  - 0.5 and 1.5 stop bits for smart cards
- Four error detection flags
  - Overflow error
  - Noise error
  - Frame error
  - Checksum error
- 10 flagged interrupt sources
  - CTS change
  - LIN disconnector detection
  - Send data register empty
  - Send completed
  - Receive data register full
  - Bus idle detected
  - Overflow error

  - Frame error
  - Noise error
  - Checksum error
- Multiprocessor communication. Silent mode if address does not match
- Wake up from silent mode (by idle bus detection or address flag detection)

Two ways of waking up the receiver: address bit (MSB, bit 9), bus idle.

32.3. USART function description

USART interface is connected with other devices through three pins. Any USART bidirectional communication requires a minimum of two pins: Receive data In (RX) and Transmit data Out (TX):

RX: Receive data Input. This is the serial data 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 I/O port configuration. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level. In Single-wire and Smartcard modes, this I/O is used to transmit and receive the data.

- The bus should be idle before transmitting or receiving
- A start bit
- A data word (8 or 9 bits), with the least significant bit first
- 0.5, 1.5, 2 stop bits, thus indicating the end of the data frame
- Use of a fractional baud rate generator: 12-bit integer and 4-bit decimal representation
- A status register (USART_SR)
- A data register (USART_DR)
- A baud rate register (USART_BRR), 12-bit integer and 4-digit decimal
- A protection time register in smart card mode (USART_GTPR)

The following pins are required in synchronous mode:

CK: Transmitter clock output.

Clock output. This pin outputs the transmitter data clock for synchronous transmission corresponding to SPI master mode (no clock pulses on start bit and stop it, and a software option to end a clock pulse on the last data bit). In parallel, data an be received synchronously on RX. This can be used to control peripherals that ave shift registers (e.g. LCD drivers). The clock phase and polarity are software rogrammable.

The following pins are required in RS232 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 data (when low).

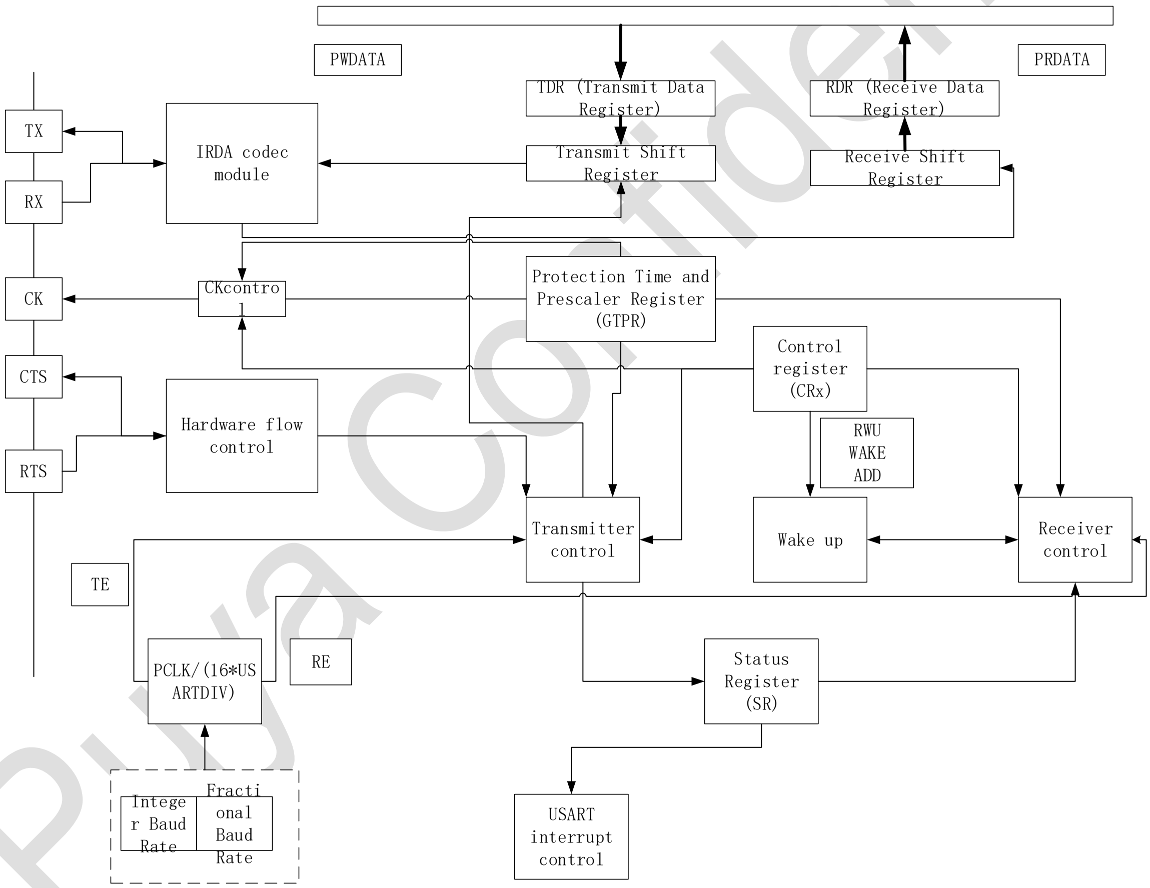


Figure 32-1 USART block diagram

32.3.1. USART character description

World length may be selected as being either 8 or 9 bits by programming the M bits in the USART_CR1 register. The TX pin is low during the start bit and high during the stop bit.

An **Idle character** is interpreted as an entire frame of “1”s followed by the start bit of the next frame containing the data(the number of “1”s includes the number of stop bits)

A **Break character** is interpreted on receiving “0”s for a frame period(including the stop bit period, which is also '0'). At the end of the break frame, the transmitter inserts 1 or 2 stop bits.

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.

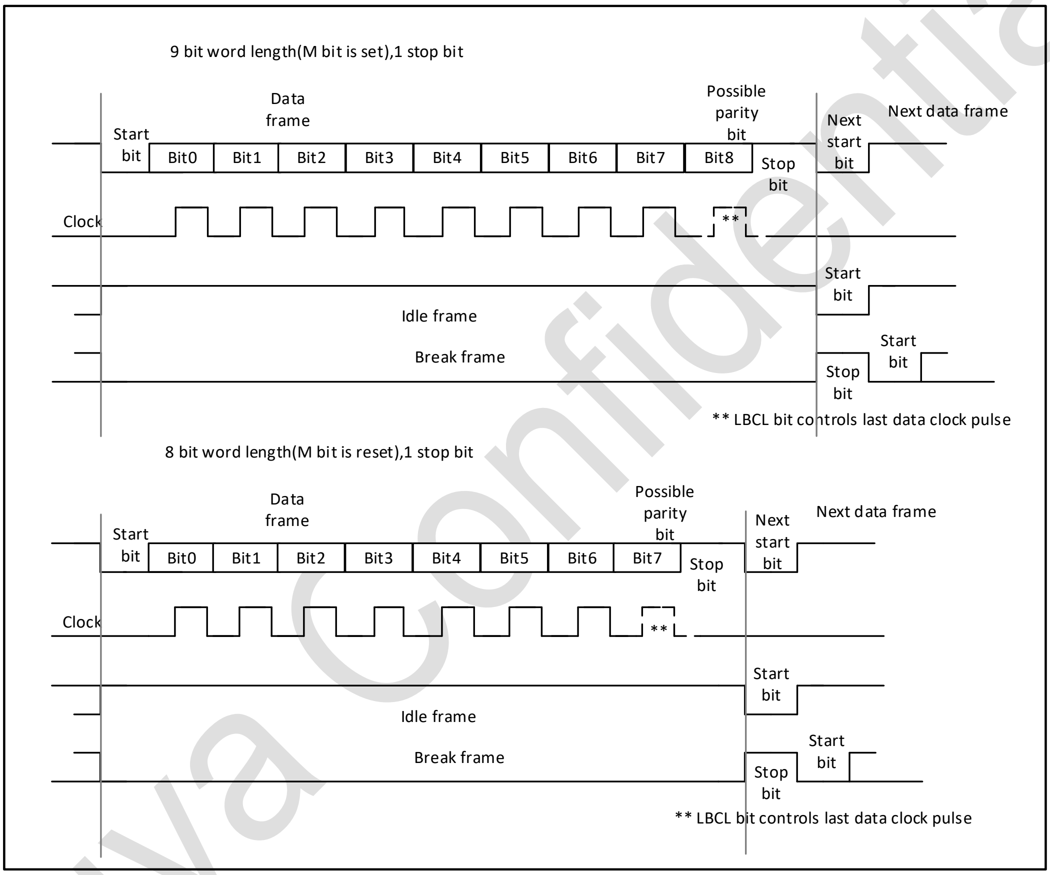


Figure 32-2 Word length programming

32.3.2. Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bits status. The Transmit Enable bit (TE) must be set in order to activate the transmitter function. The data in the transmit shift register is output on the TX pin and the corresponding clock pulses are output on the CK pin.

32.3.2.1. Character transmission

During an USART transmission, data shifts out least significant bit first (default configuration) on the TX pin. In this mode, the USART_DR register consists of a buffer (DR) between the internal bus and the transmit shift register.

Every character is preceded by a start bit which is a logic level low for one bit period. The character is terminated by a configurable number of stop bits. The USART supports multiple stop bit configurations: 1 and 2 stop bits.

Note:

The TE bit should not be reset during transmission of data. Resetting the TE bit during the transmission will corrupt the data on the TX pin as the baud rate counters will get frozen. The current data being transmitted will be lost.

An idle frame will be sent after the TE bit is enabled.

32.3.2.2. Configurable stop bits

The number of stop bits to be transmitted with every character can be programmed in Control register 2, bits 13,12.

1） 1 stop bit: This is the default value of number of stop bits.

2） 2 stop bits: This will be supported by normal USART, Single-wire and Modem modes. An idle frame transmission will include the stop bits.

A break transmission will be 10 low bits (when m = 0) or 11 low bits (when m = 1) followed by 2 stop bits. It is not possible to transmit long breaks (break of length greater than 9/10 low bits).

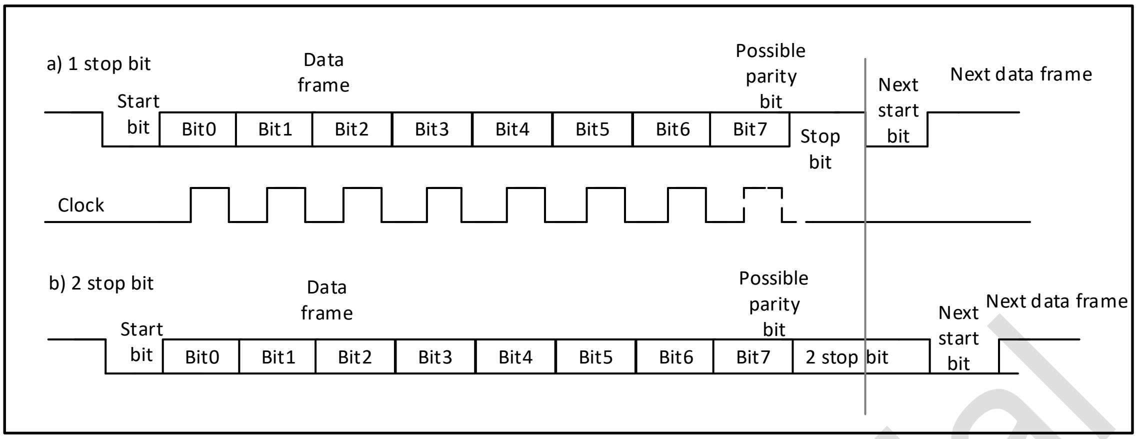


Figure 32-3 Configurable stop bits

Character transmission procedure:

1) Enable the USART by writing the UE bit in USART_CR1 register to 1.

2) Program the M bits 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 is to take place. Configure the DMA register as explained in multibuffer communication.

5) Select the desired baud rate using the USART_BRR register.

6) Set the TE bit in USART_CR1 to send an idle frame as first transmission.

7) Write the data to 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.

32.3.2.3. Single byte communication

Clearing the TXE bit is always performed by a write to the transmit data register. The TXE bit is set by hardware and it indicates:

- The data has been moved from the USART_TDR register to the shift register and the data transmission has started.
- The USART_TDR register is empty.
- The next data can be written in the USART_TDR 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 DR register, next, the data is copied in the shift register at the end of the currently ongoing transmission.

When no transmission is taking place, a write instruction to the USART_DR register places the data in the shift register, the data transmission starts, and the TXE bit is 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 in the USART_TDR 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 can also be cleared by software by writing '0' to it. This clearing method is only recommended for use in multi-buffer communication mode.

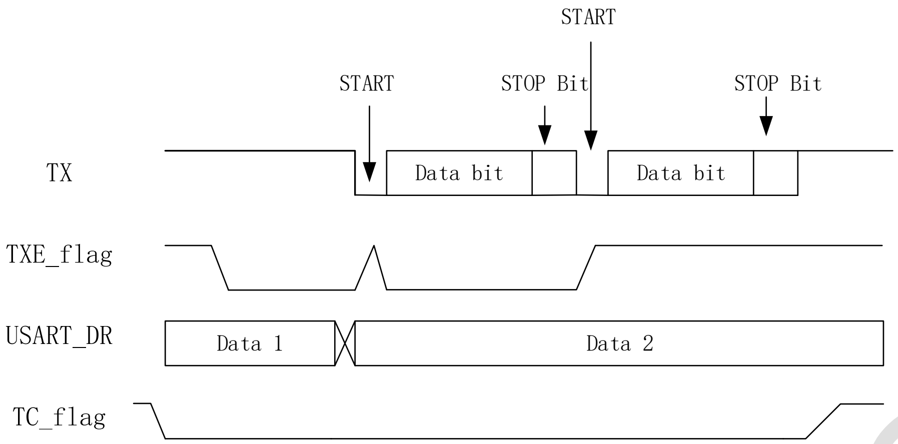


Figure 32-4 TC/TXE behavior when transmitting

32.3.2.4. Break characters

Setting the SBK bit transmits a break character. The break frame length depends on the M bits. If a '1' is written to the SBK bit, a break character is sent on the TX line after completing the current character transmission. The SBK bit is set by the write operation and it is reset by hardware when the break character is completed (during the stop bits after the break character). The USART inserts a logic 1 signal (STOP) for the duration of at the end of the break frame to guarantee the recognition of the start bit of the next frame.

If software resets the SBK bit before starting to transmit the break frame, the break symbol will not be sent. If two consecutive break frames are to be sent, the SBK bit should be set after the stop bit of the previous break symbol.

32.3.2.5. Idle characters

Setting the TE bit drives the USART to send an idle frame before the first data frame.

32.3.3. Recevier

The USART can receive data words of either 7, 8 or 9 bits depending on the M bits in the USART_CR1 register.

32.3.3.1. 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 0 X 0 0 0 0.

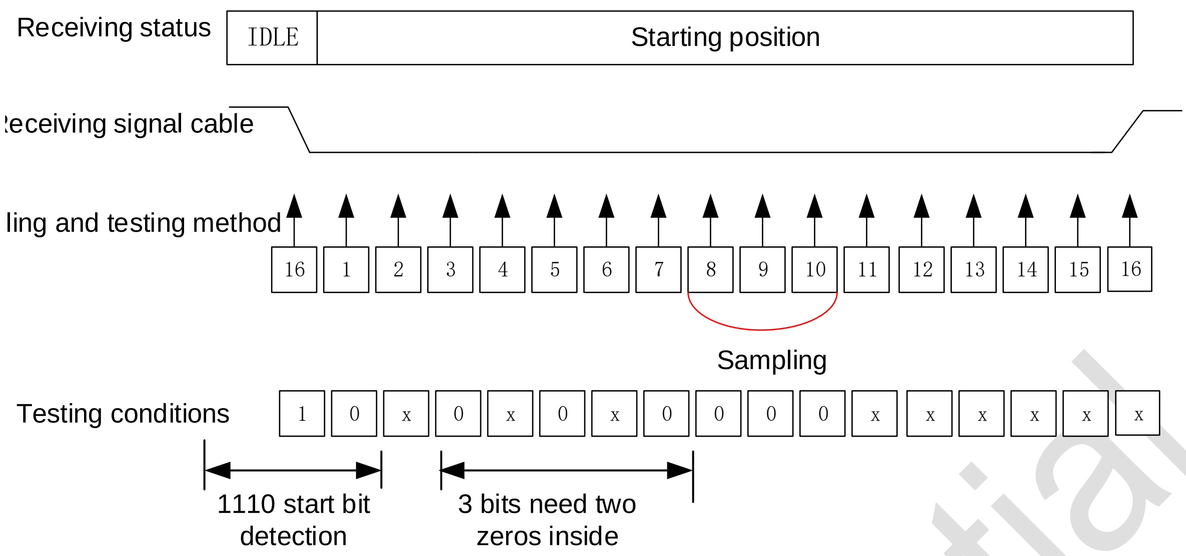


Figure 32-5 Start bit detection

If the sequence is incomplete, the receiver will exit the start bit detection and return to the idle state *(* without setting the flag *)* to wait for a falling edge. If all *3* sample points are *'0' (* the first sample at bits *3, 5,* and *7, and the second sample at bits 8*, *9*, and *10* are all *'0')*, then acknowledge receipt Start bit, then set *the RXNE* flag bit, if *RXNEIE = 1*, an interrupt will be generated.

If only *2 of the 3 sample points* are *'0' twice (* the *3rd, 5th, 7th* sample point and the *8th, 9th, 10th* sample point *),* then the start bit is still valid, but The *NE* noise flag is set. If this condition cannot be met, the detection process of the start bit is aborted, and the receiver will return to the idle state *(* the flag bit is not set *).*

If at one time only *2 of the 3 sample points* are *'0' (* the *3rd, 5th, 7th* sample point or the *8th, 9th, 10th* sample point *),* then the start bit is still valid, but the *NE* noise flag is set.

32.3.3.2. Character reception

During USART reception, the least significant bit of data is shifted in first from the RX pin. In this mode, the USART_DR register contains a buffer between the internal bus and the receive shift register.

Configuration steps:

1. Set UE in USART_CR1 register to 1 to activate USART.

2. Program the M bits of USART_CR1 to define the word length

3. Write the number of stop bits in USART_CR2

4. If multi-buffer communication is required, select the DMA enable bit (DMAR) in USART_CR3. 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 of USART_CR1. Activate the receiver to start looking for the start bit.

When a character is received :

- The RXNE bit is set. It indicates that the contents of the shift register are transferred to the RDR. In other words, the data has been received and can be read (including error flags associated with it).
- If the RXNEIE bit is set, an interrupt is generated.
- If a frame error, noise or overflow error is detected during reception, the error flag will be set
- In multi-buffer communication, RXNE is set up after each byte is received, and is cleared by the DMA read operation of the data register.
- In single buffer mode, the RXNE bit is cleared by software reading the USART_DR register. The RXNE flag can also be cleared by writing 0 to it. The RXNE bit must be cleared before the end of the next character reception to avoid overrun errors. Note: The *RE* bit should not be reset while receiving data. If the *RE* bit is cleared on reception, the reception of the current byte is lost.

32.3.3.3. Break character

When a break character is received, the USART handles it as a framing error.

32.3.3.4. Idle character

When an idle frame is detected, there is the same procedure as for a received data character plus an interrupt if the IDLEIE bit is set.

32.3.3.5. 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 or the previous DMA request has not been serviced.

When an overrun error occurs: The ORE bit is set.

- The RDR content will not be lost. The previous data is available when a read to USART_RDR is performed.
- The shift register will be oveRWritten. After that point, any data received during overrun is lost.
- An interrupt is generated if either the RXNEIE bit is set or both the EIE and DMAR bits are set.
- Sequential read operations of USART_SR and USART_DR registers can reset the ORE bit

Note: When the *ORE* bit is set, it indicates that at least *1* data has been lost. There are two possibilities:

- If RXNE = 1, the last valid data is still in the receive register RDR and can be read.
- If RXNE = 0, it means that the last valid data has been read, and there is nothing to read in RDR. This can happen when new (ie lost) data is received while the last valid data is being read in the RDR. This can also happen when new data is received during the read sequence (between the USART_SR register read access and the USART_DR read access).

32.3.3.6. Noise error

Data recovery is performed by distinguishing between valid input data and noise using oversampling techniques (except synchronous mode)

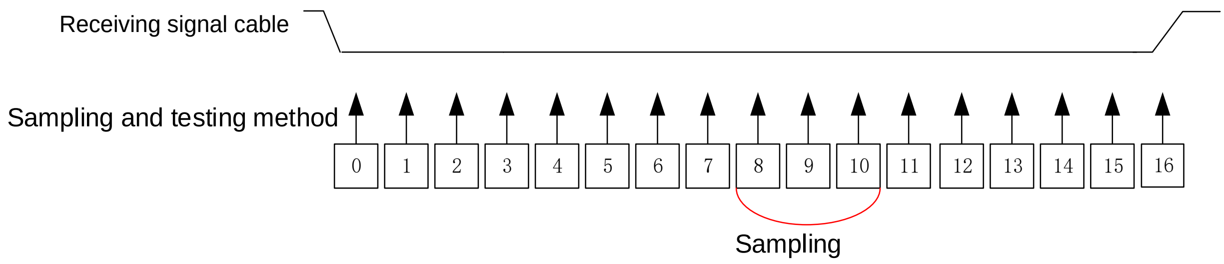


Figure 32-6 Data sampling for noise detection

Table 32-1 Noise detection from sampled data

Sample value	NE state	Bit value received	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 the received frame:

- Set the NE flag on the rising edge of the RXNE bit.
- Invalid data is transferred from the shift register to the USART_DR register.
- In the case of single-byte communication, no interrupt is generated. However, since the NE flag and the RXNE flag are set at the same time, RXNE will generate an interrupt. In the case of multi-buffer communication, an interrupt will be generated if the EIE bit in the USART_CR3 register has been set.
- First read USART_SR, then read USART_DR register, will clear the NE flag bit.

32.3.3.7. Framing error

A framing error is detected when the stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.

When the framing error is detected:

- The FE bit is set by hardware
- The invalid data is transferred from the Shift register to the USART_RDR register.
- No interrupt is generated in case of single byte communication. However this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt will be issued if the EIE bit is set in the USART_CR3 register.
- Sequential reads of the USART_SR and USART_DR registers reset the FE bit.

32.3.3.8. Configurable stop bits during reception

The number of stop bits to be received can be configured via the control bits in control register 2, which can be 1 or 2 in normal mode, and may be 0.5 or 1.5 in smart card mode.

- 0.5 stop bits (receive in smart card mode): 0.5 stop bits are not sampled. Therefore, if 0.5 stop bits are selected, frame errors and broken frames cannot be detected.
- 1 stop bit: 1 stop bit is sampled at sample points 8, 9 and 10.
- 1.5 stop bits (smart card mode): When sending in smart card mode, the device must check that the data has been sent out correctly. Therefore the receiver function block must be activated (RE = 1 in the USART_CR1 register) and the signal on the data line must be sampled during the sending of the stop bit. If a checksum error occurs, the smart card will indicate a framing error by pulling the data line low when the sender samples the NACK signal, i.e. at the time corresponding to the stop bit on the bus. FE is set up at the end of the 1.5 stop bits along with RXNE. The 1.5 stop bits are sampled at sample points 16, 17 and 18. The 1.5 stop bits can be divided into 2 parts: one is 0.5 clock cycles, during which nothing is done. This is followed by 1 clock cycle of stop bits, which are sampled at the midpoint of this period.
- 2 stop bits: The sampling of the 2 stop bits is done at the 8th, 9th and 10th sample points of the first stop bit. If a frame error is detected during the first stop bit, the frame error flag is set. The second stop bit is no longer checked for frame errors. The RXNE flag will be set at the end of the first stop bit.

32.3.4. USART baud rate generation

The baud rate for the receiver and transmitter (Rx and Tx) are both set to the same value as programmed in the USART_BRR register.

Tx / Rx baud = *fCK* /(16**USARTDIV*)

Here *fCK* is the clock to the peripheral USARTDIV is an unsigned fixed-point number. The 12-bit value is set in the USART_BRR register.

Note: After writing to *USART_BRR,* the baud rate counter is replaced by the new value of the baud rate register. Therefore, do not change the value of the baud rate register while communication is in progress.

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:

To program USARTDIV = 25.62, then:

DIV_Fraction = 16*0.62 = 9.92

The nearest integer is: 10 = 0x0A

DIV_Mantissa = mantissa (25.620) = 25 = 0x19

Then, USART_BRR = 0x19A

Example 3:

To program USARTDIV = 50.99, then:

DIV_Fraction = 16*0.99 = 15.84

The nearest integer is: 16 = 0x10 = > DIV_frac[3:0] overflow = > carry must be added to the fractional part

DIV_Mantissa = mantissa (50.990 + carry) = 51 = 0x33

Then, USART_BRR = 0x330, USARTDIV = 51

Baud rate		F_{PCLK} = 36 MHz			F_{PCLK} = 72 MHz		
S.No	**Kbps**	**Actual**	**BRR**	**Error(%)**	**Actual**	**BRR**	**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	N.A	N.A	N.A	4500	1	0%

Note: The lower the CPU clock the lower the accuracy for a particular baud rate. The upper limit of the achievable baud rate can be fixed with these data.

32.3.5. USART receiver's tolerance to clock deviation

The asynchronous receiver of the USART works 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's 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-tolow transition timing)

DTRA + DQUANT + DREC + DTCL < USART receiver's tolerance.

For normal reception of data, the tolerance of the USART receiver is equal to the maximum tolerable variation, which depends on the following choices:

- 10- or 11-bit character length defined by the M bits of the USART_CR1 register

- whether to use fractional baud rate to generate

Table 32-2 USART receiver tolerance when DIV_Fraction is 0

M bit	NF is an error	NF is don't care
0	3.75%	4.375%
1	3.41%	3.97%

Table 32-3 USART receiver tolerance when DIV_Franction is different from 0

M bit	NF is an error	NF is don't care
0	3.33%	3.88%
1	3.03%	3.53%

32.3.6. USART auto baud rate detection

The USART is able to detect and automatically set the USART_BRR register value based on the reception of one character. Automatic baud rate detection is useful under two 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 (oversampling by 16 must be selected and baudrate between fCK/65535 and fCK/16.

Before activating the auto baud rate detection, the auto baud rate detection mode must be chosen. There are various modes based on different character patterns(They can be chosen through the ABRMOD[1:0] field in the USART_CR3 register). 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:

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, ensuring better accuracy in the case of slow signal slopes.

In parallel, another check is performed for each intermediate transition of RX line. An error is generated if the transitions on RX are not sufficiently synchronized with the receiver (the receiver being based on the baud rate calculated on bit 0).

Prior to activating auto baud rate detection, the USART_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 USART_CR2 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 USART_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 (oversampling by 16) and not between 8 and 65536 clock periods (oversampling by 8)).

The RXNE interrupt will signal the end of the operation. At any later time, the auto baud rate detection may be relaunched by resetting the ABRF flag (by writing a 0).

Note: If the USART is disabled (UE = 0) during an auto baud rate operation, the BRR value may be corrupted.

32.3.7. Multiprocessor communication using USART

It is possible to perform multiprocessor communication with the USART (with several USARTs connected in a network). For instance one of the USARTs can be the master, its TX output connected to the RX inputs of the other USARTs. The others are slaves, their respective TX outputs are 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 should actively receive the full message contents, thus reducing redundant USART service overhead for all non addressed receivers.

The non addressed devices may be placed in mute mode by means of the muting function.In mute mode:

- None of the reception status bits can be set.
- All the receive interrupts are inhibited.
- The RWU bit in USART_ISR register is set to 1. RWU can be controlled automatically by hardware or by software, through the MMRQ bit in the USART_RQR register, under certain conditions. The USART can enter or exit from mute mode using one of two methods, depending on the WAKE bit in the USART_CR1 register:
- Idle Line detection if the WAKE bit is reset.
- Address Mark detection if the WAKE bit is set.

32.3.7.1. Idle line 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_ISR register. An example of mute mode behavior using Idle line detection is given in the following figure.

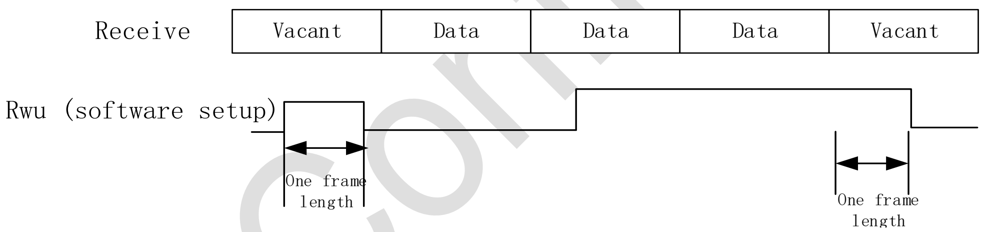


Figure 32-7 Mute mode using Idle line detection

32.3.7.2. Address mark check (WAKE = 1)

In this mode, bytes are recognized as addresses if their MSB is a ‘1’ otherwise they are considered as data. In an address byte, the address of the targeted receiver is put in the 4 LSBs. The choice of 4-bit address detection is done using the ADDM7 bit. This 4-bit/7-bit word is compared by the receiver with its own address which is programmed in the ADD bits in the USART_CR2 register.

If the received byte does not match its programmed address, the USART enters silent mode. At this point, the hardware sets the RWU bit.

Receiving this byte will neither set the RXNE flag nor generate an interrupt or issue a DMA request because the USART is already in silent mode.

When the received byte matches the programmed address in the receiver, the USART exits silent mode. Then the RWU bit is cleared and subsequent bytes are received normally. The RXNE bit will be set when this matching address byte is received because the RWU bit has been cleared.

When the receive buffer contains no data (RXNE = 0 in USART_SR), the RWU bit can be written to 0 or 1. Otherwise , the write operation is ignored. The figure below shows an example of using address mark detection to wake up and enter silent mode.

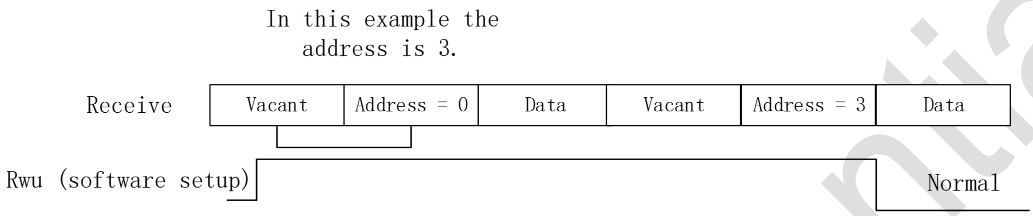


Figure 32-8 Slient mode with address tag detection

32.3.7.3. Check control

Setting the PCE bit on the USART_CR1 register enables parity control (generates a parity bit when transmitting, and performs parity checking when receiving). The possible USART frame formats are listed in the table below according to the frame length defined by the M bits.

Table 32-4 Frame format

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 a device with an address tag, the address is matched only considering the *MSB* bits of the data, not the parity bits. *(MSB* is the last sent out of the data bits, followed by the parity bit or stop bit*)*

32.3.7.4. Even parity

The parity bit is calculated to obtain an even number of “1s” inside the frame of the 7 or 8 LSB bits (depending on M bits values) and the parity bit.

As an example, if data = 00110101, and 4 bits are set, then the parity bit will be 0 if even parity is selected (PS bit in USART_CR1 = 0).

32.3.7.5. Odd parity

The parity bit is calculated to obtain an odd number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on M bits values) and the parity bit.

As an example, if data = 00110101 and 4 bits set, then the parity bit will be 1 if odd parity is selected (PS bit in USART_CR1 = 1).

32.3.7.6. Transfer mode

If the PCE bit is set in USART_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_ISR register and an interrupt is generated if PEIE is set in the USART_CR1 register. The PE flag is cleared by software riting 1 to the PECF in the USART_ICR register.

32.3.8. LIN (Local Area Internet) mode

Configure USART_CR2.LINEN=1 to select the LIN mode. In LIN mode, the following bits must remain bit 0:

- CLKEN of the USART_CR2 register.
- STOP/SCEN/HDSEL/IREN of the USART_CR3 register.

32.3.8.1. Sending

Compared to a normal USART send, a LIN send has the following differences:

M=0 and a data length of 8 bits;

It is necessary to configure USART_CR2.LINEN=1. 13bit 0 will be sent as a break frame after setting SBK. A "1" is then sent to allow detection of the next start bit.

32.3.8.2. Receive

When LIN mode is enabled, the break symbol detection circuit is activated. This detection is completely independent of the USART receiver. The break symbol is detected as soon as it appears, either when the bus is idle or during the sending of a data frame which is not yet complete and the sending of another break symbol is inserted.

When the receiver is activated (RE=1 for USART_CR1), the circuit monitors the start signal on RX. Monitoring the start bit is done in the same way as detecting the break symbol or data. When the start bit is detected, the circuit samples each subsequent bit at the 8th, 9th and 10th oversampling clock point of each bit. If 10 (when LBDL = 0 for USART_CR2) or 11 (when LBDL = 1 for USART_CR2) consecutive bits are 0 and are followed by a delimiter, the LBD flag of USART_SR is set. If the LBDIE bit = 1, the interrupt is generated. Check the delimiter before acknowledging the break symbol, as it means that the RX line has gone back to high.

If a 1 is sampled before the 10th or 11th sample point, the detection circuit cancels the current detection and re-finds the start bit. If LIN mode is disabled, the receiver continues to operate as a normal USART and does not need to consider detecting the break sign.

If LIN mode is not activated (LINEN=0), the receiver continues to operate normally in USART mode and no disconnect detection is performed.

If LIN mode is activated (LINEN=1), as soon as a frame error occurs (i.e. the stop bit detects a '0', which occurs in a break frame), the receiver stops until the break symbol detection circuit receives a 1 (which occurs when the break symbol is not sent in full), or a delimiter (This occurs when a complete disconnected symbol has been detected).

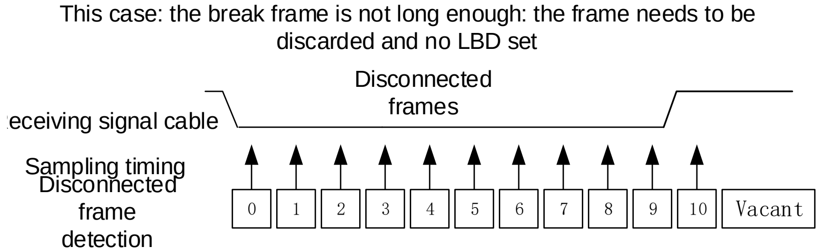


Figure 32-9 Insufficient break frame length in LIN mode

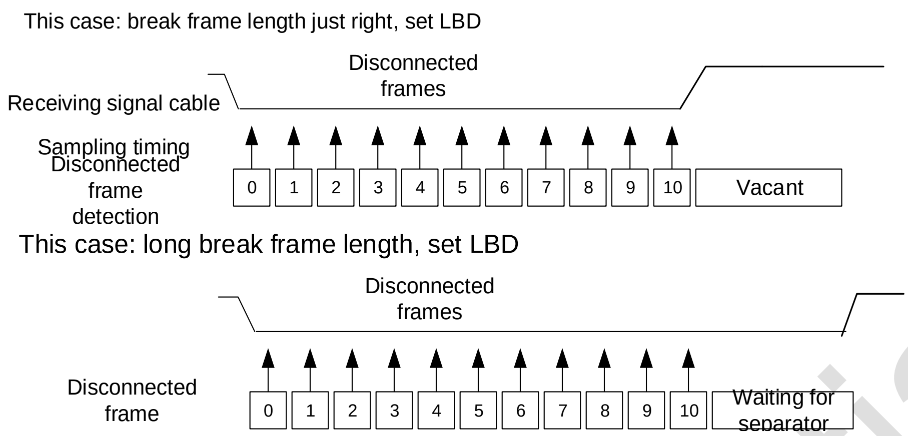


Figure 32-10 LIN mode with enough break frame length

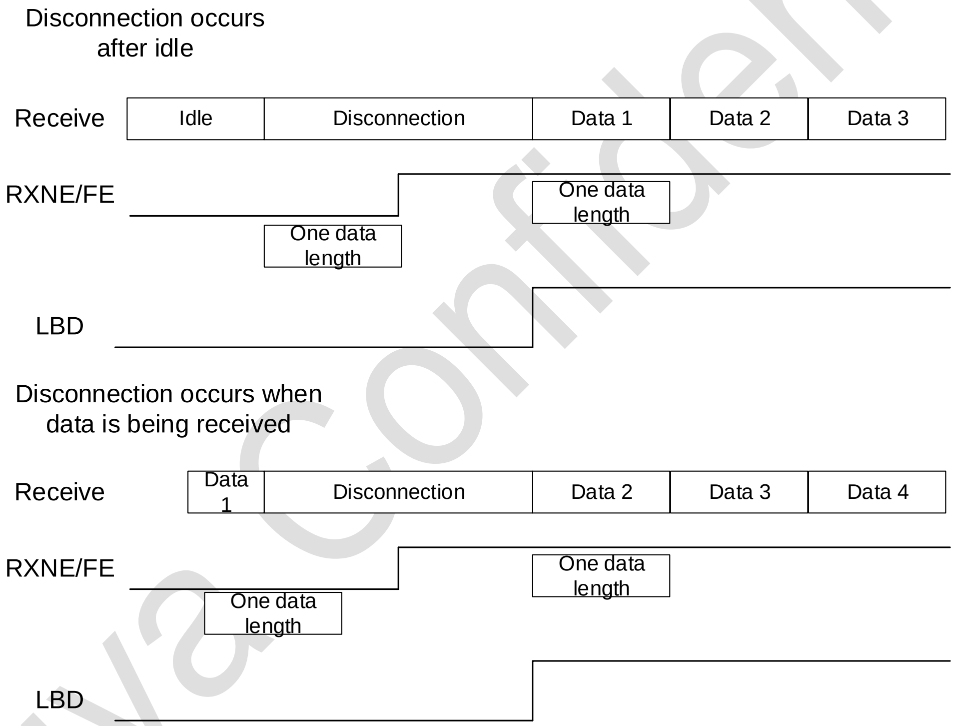


Figure 32-11 Disconnection detection in LIN mode and detection of frame errors

32.3.9. USART synchronous mode

The synchronous mode is selected by writing the CLKEN bit of the USART_CR2 register to 1. In synchronous mode, the following bits must remain clear:

- LINEN bit in the USART_CR2 register
- SCEN, HDSEL and IREN bits in the USART_CR3 register

The USART allows the user to control bi-directional synchronous serial communication in master mode. pin CK is the output of the USART transmitter clock. There are no clock pulses on the CK pin during the start and stop bits. Depending on the state of the LBCL bit in the USART_CR2 register, it is determined whether a clock pulse is generated or not during the last valid data bit. the CPOL bit in the USART_CR2 register allows the user to select the clock polarity and the CPHA bit on the USART_CR2 register allows the user to select the phase of the external clock.

The external CK clock is not activated during bus idle, before the actual data arrives and when sending a break symbol.

In synchronous mode, the USART transmitter works exactly the same as in asynchronous mode. However, because the CK is synchronised with the TX (according to CPOL and CPHA), the data on the TX is sent out synchronously with the CK.

The USART receiver in synchronous mode works differently than in asynchronous mode. If RE=1, the data is sampled on CK (on the rising or falling edge according to CPOL and CPHA) without any oversampling. However, the build-up time and duration (depending on baud rate, 1/16 bit time) must be taken into account.

Notice:

The CK pin works in conjunction with the TX pin. Thus, the clock is only provided when the transmitter is enabled (TE = 1) and data is sent (writing data to the USART_DR register). This means that it is not possible to receive a synchronous data when no data is being sent.

The LBCL, CPOL and CPHA bits should be correctly configured when both the transmitter and receiver are disabled; these bits cannot be changed when the transmitter or receiver is enabled.

It is recommended to set TE and RE in the same command to reduce the build time and hold time of the receiver.

USART only supports the master mode: it cannot receive or send data with an input clock from another device (CK is always an output).

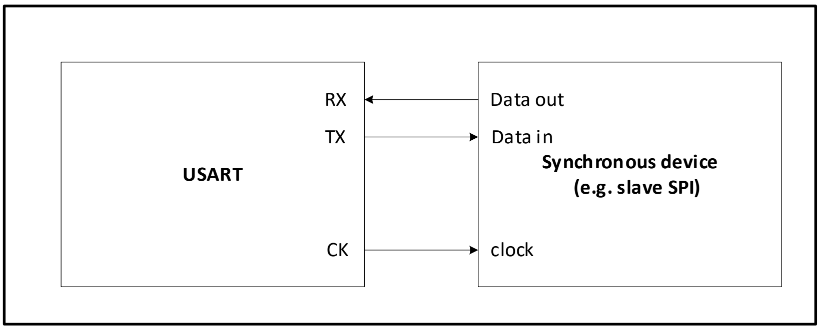


Figure 32-12 Example of USART isochronous transmission

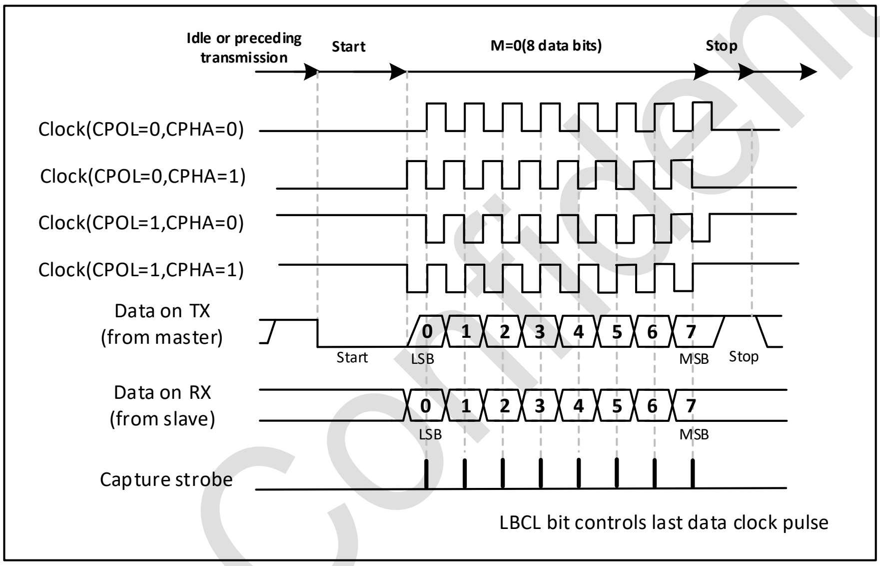


Figure 32-13 USART data clock timing example (M = 0)

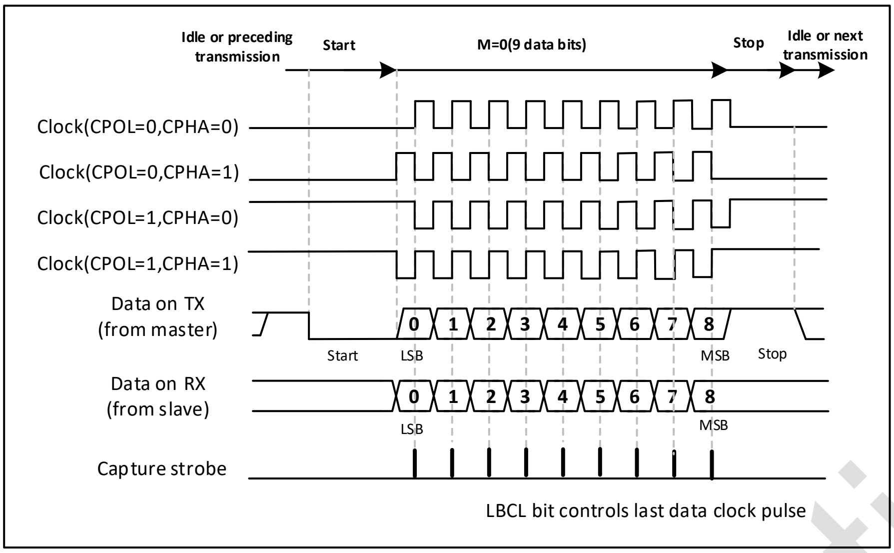


Figure 32-14 USART data clock timing example(M = 1)

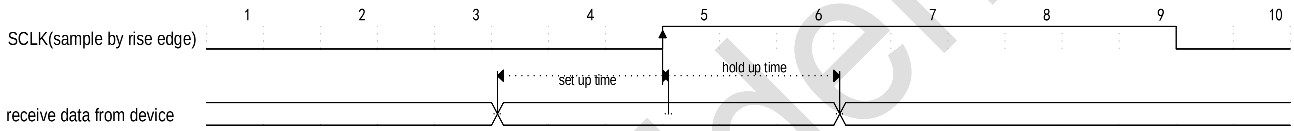


Figure 32-15 RX data sample/hold time

32.3.10. USART single-wire half-duplex communication

The single-wire semi-bidirectional mode is selected by setting the HDSEL bit of the USARTx_CR3 register. In this mode, the following bits must remain clear:

- CLKEN bit of the USARTx_CR2 register
- SCEN and IREN bits of the USART_CR3 register

The USART can be configured to follow a single-wire half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are interconnected internally on the chip. Half and full duplex communication is selected using the control bit 'HALF DUPLEX SEL' (HDSEL bit in USARTx_CR3).

When HDSEL is '1'

- RX is no longer used
- TX is always released when no data is being transmitted. Therefore, it behaves as a standard I/O port in the idle state or in the receive state. This means that this I/O must be configured as a dangling input (or an open-drain output high) when not driven by the USART.

Other than this, communication is similar to normal USART mode. It is up to the software to manage conflicts on the line (e.g. by using a central arbiter). In particular, the transmit from is not blocked by the hardware. When the TE bit is set, the transmission continues as soon as the data is written to the data register.

32.3.11. Smart card

Set the SCEN bit of the USART_CR3 register to select the smart card mode. In smart card mode, the following bits must remain clear:

- LINEN bit of the USART_CR2 register
- HDSEL bit and IREN bit of the USART_CR3 register

In addition, the CLKEN bit can be set to provide a clock to the smart card.

The interface is ISO7816-3 compliant and supports the smart card asynchronous protocol. The USART should be set to:

1. 8 bits of data plus a parity bit: in this case M=1 and PCE=1 in the USART_CR1 register
2. 1.5 stop bits for transmit and receive: i.e. STOP=11 in the USART_CR2 register

Note: It is also possible to select 0.5 stop bits for receive, but to avoid switching between the two configurations it is recommended to use 1.5 stop bits for both transmit and receive.

The example given below illustrates the signal on the data line, with and without a checksum error.

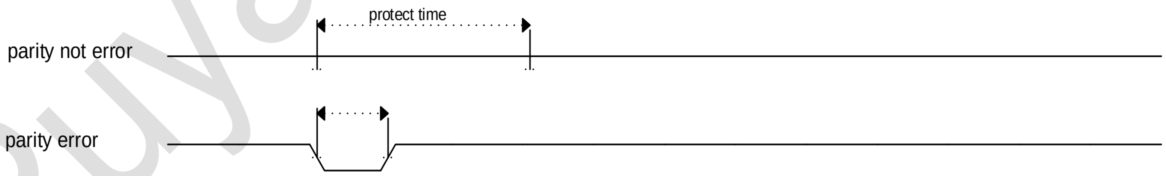


Figure 32-16 ISO7816-3 asynchronous protocol

When connected to a smart card, the TX of the USART drives a bi-directional line that is also driven by the smart card. In order to do this, SW_RX must be connected to the same I/O port as TX. The transmitter's output enable bit TX_EN is set up during the sending of start bits and data bytes and released during the sending of stop bits (weak pull-up), so that the receiver can pull the data line low if a checksum error is detected. If TX_EN is not used, TX is pulled high during the stop bit: in this case the receiver can also drive the line as long as TX is configured for open drain.

The smart card is a single wire half duplex communication protocol

1) Sending data out of the transmit shift register is delayed by a minimum of 1/2 baud clock. During normal operation, a full transmit shift register will start shifting data out at the next baud clock edge. In smart card mode, this send is delayed by 1/2 baud clock.

2) If a parity error is detected during the reception of a data frame set to 0.5 or 1.5 stop bits, the transmit line is pulled down one baud clock cycle after the reception of the frame is complete (i.e. when the stop bit ends). This tells the smart card that the data sent to the USART has not been received correctly. This NACK signal (pulling down the transmit line by one baud clock cycle) will generate a frame error on the transmitter side (which is configured for 1.5 stop bits). The application can handle resending the data according to the protocol. If the NACK control bit is set, the receiver will give a NACK signal in case of a checksum error; otherwise no NACK will be sent.

3) The setting of the TC flag can be delayed by programming the protection time register. During normal operation, the TC is set when the transmit shift register becomes empty and no new transmit requests are made. In smart card mode, an empty transmit shift register will trigger the protection time counter to start counting up until the value in the protection time register is reached. The TC is forced low during this time. When the protection time counter reaches the value in the protection time register, the TC is set high.

4) The TC flag is withdrawn independent of the smart card mode.

5) If the transmitter detects a frame error (receipt of a NACK signal from the receiver), the receiver function module of the transmitter does not detect the NACK as a start bit. The duration of the received NACK can be 1 or 2 baud clock cycles according to the ISO protocol.

6) On the receiver side, if a checksum error is detected and a NACK is sent, the receiver will not detect the NACK as a start bit.

Notice:

1) The disconnect symbol has no meaning in smart card mode. A 00h data with a framing error will be treated as data and not as a disconnect symbol.

2) No IDLE frames are sent when the TE bit is toggled back and forth. The ISO protocol does not define IDLE frames.

The diagram below details how the USART samples the NACK signal. In this example, the USART is sending data and is configured for 1.5 stop bits. In order to check the integrity of the data and the NACK signal, the receive function block of the USART is activated.

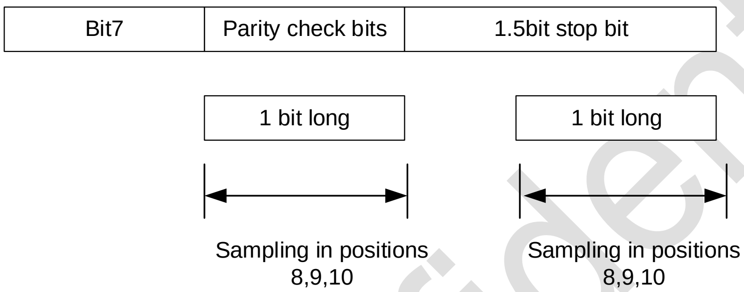


Figure 32-17 Parity error detection using 1.5 stop bits

The USART can provide a clock for the smart card via the CK output. In smart card mode, CK is not directly linked to communication, but is first simply used to drive the smart card clock from the internal peripheral input clock via a 5-bit prescaler. The dividing frequency is configured in the prescaler register USART_GTPR. The CK frequency can range from fCK/2 to fCK/62, where fCK is the peripheral input clock.

32.3.12. IrDA SIR ENDEC function module

The IrDA mode is selected by setting the IREN bit of the USART_CR3 register. In IRDA mode, the following bits must remain clear:

- LINEN, STOP and CLKEN bits of the USART_CR2 register
- The SCEN and HDSEL bits of the USART_CR3 register.

32.3.12.1. IrDA normal mode

The IrDA SIR physical layer specifies the use of an inverted zero modulation scheme (RZI), which uses an infrared light pulse to represent a logic '0' (see Figure 4-12). The SIR transmits an encoder to modulate the NRZ (non-zeroed) bit stream from the USART output. The output pulse stream is transmitted to an external output driver and IR LED. The USART for SIR ENDEC only supports up to 115.2Kbps rate. In normal mode, the pulse width is specified as 3/16 of a bit period.

The SIR receive decoder demodulates the zeroed bit stream from the IR receiver and outputs the received NRZ serial bit stream to the USART. in the idle state the decoder input is normally high (marking state). The polarity of the transmit encoder output is opposite to that of the decoder input. When the decoder input is low, a start bit is detected.

- IrDA is a half-duplex communication protocol. If the transmitter is busy (i.e. the USART is sending data to the IrDA encoder), any data on the IrDA receive line will be ignored by the IrDA decoder. If the receiver is busy (i.e. the USART is receiving decoded data from the IrDA decoder), the data on the TX from the USART to the IrDA will not be encoded by the IrDA. When receiving data, sending should be avoided as the data that will be sent may be corrupted.
- The SIR transmit logic sends '0' as a high pulse and '1' as a low level. The width of the pulse is specified as 3/16 of the bit period in normal mode.
- The SIR receive logic interprets the high state as a '1' and the low pulse as a '0'.
- The transmit encoder output has the opposite polarity to the decoder input. When idle, the SIR output is in the low state.
- The SIR decoder converts the IrDA compatible receive signal into a bit stream for the USART.
- The IrDA specification requires pulses wider than 1.41us. The pulse width is programmable. The spike pulse detection logic at the receiver side filters out pulses less than 2 PSC cycles wide (PSC is a prescaler value programmed in the IrDA low power baud rate register USART_GTPR). Pulses with a width of less than 1 PSC period must be filtered out, but those with a width greater than 1 and less than 2 PSC periods may be received or filtered out, and those with a width greater than 2 periods will be treated as a valid pulse. When PSC = 0, the IrDA encoder/decoder does not operate.
- The receiver can communicate with a low-power transmitter.
- In IrDA mode, the STOP bit on the USART_CR2 register must be configured as a stop bit.

32.3.12.2. IrDA low power mode

Transmitter:

In low-power mode, the pulse width no longer lasts for 3/16 of a bit period. Instead, the width of the pulse is 3 times the low power baud rate, which can be as low as 1.42 MHz. usually this value is 1.8432 MHz (1.42 MHz < PSC < 2.12 MHz). A low-power mode programmable divider divides the system clock to achieve this value.

Receiver:

The low-power mode reception is similar to the normal mode reception. To filter out spikes, the USART should filter out pulses shorter than 1 PSC in width. Only low level signals with a duration greater than 2 cycles of the IrDA low power baud rate clock (PSC in USART_GTPR) are accepted as valid.

Notice:

1. pulses with a width of less than 2 greater than 1 PSC period may or may not be filtered out.

2. The build-up time of the receiver should be managed by software. The IrDA physical layer specification specifies a minimum delay of 10ms between transmit and receive (IrDA is a half duplex protocol).

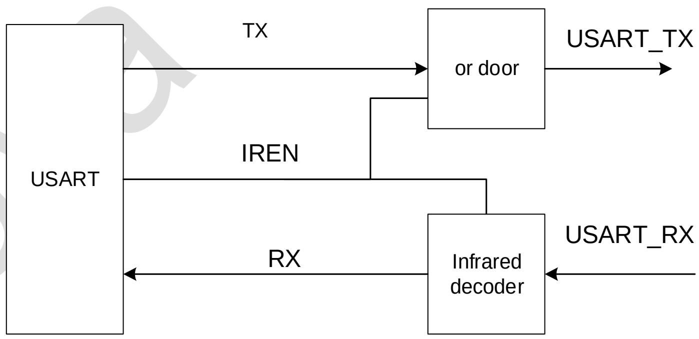


Figure 32-18 IrDA SIR ENDEC block diagram

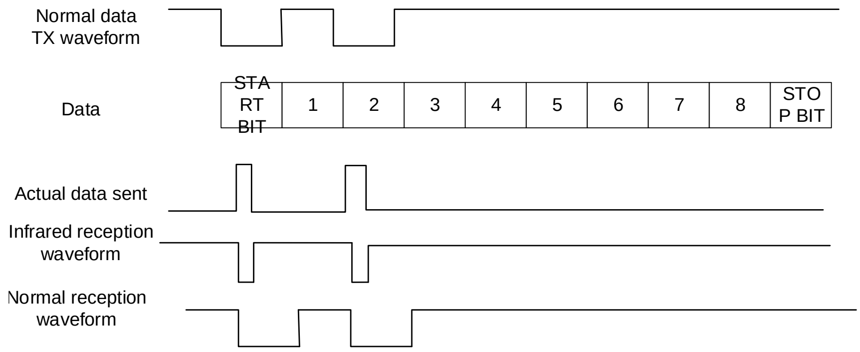


Figure 32-19 IrDA data modulation (3/16) - normal mode

32.3.13. USART continuous communication in DMA mode

The USART is capable of performing continuous communication using the DMA. The DMA requests for Rx buffer and Tx buffer are generated independently.

32.3.13.1. Transmission using DMA

DMA mode can be enabled for transmission by setting DMAT bit in the USART_CR3 register. Data is loaded from a SRAM area configured using the DMA peripheral to the USART_DR register whenever the TXE bit is set. To map a DMA channel for USART transmission, use the following procedure (x denotes the channel number):

1) Write the USART_TDR register address in the DMA control register to configure it as the destination of the transfer. The data is moved to this address from memory after each TXE event.

2) Write the memory address in the DMA control register to configure it as the source of the transfer. The data is loaded into the USART_TDR register from this memory area after each TXE event.

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) Configure DMA interrupt generation after half/ full transfer as required by the application.

6) Clear the TC flag in the USART_ISR register by setting the TCCF bit in the USART_ICR register.

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 transmission mode, once the DMA has written all the data to be transmitted, the TC flag can be monitored to make sure that the USART communication is complete. This is required to avoid corrupting the last transmission before disabling the USART or entering Stop mode. software needs to wait for TXE = 1 first, then wait for TC = 1.

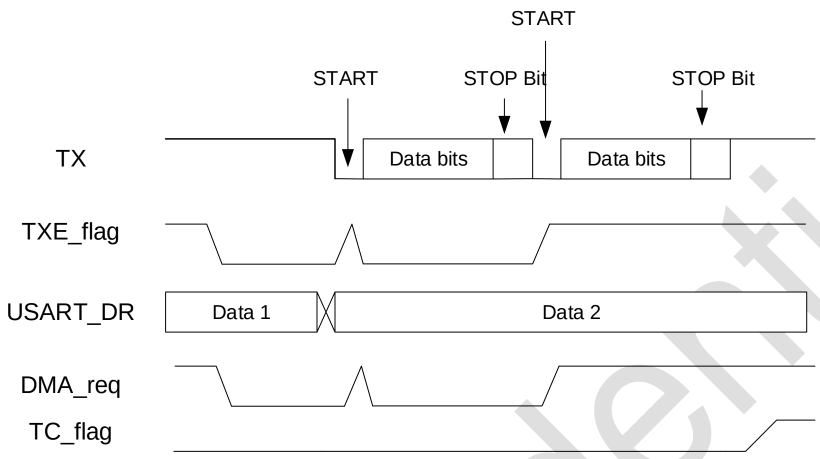


Figure 32-20 Send using DMA

32.3.13.2. Reception using DMA

DMA can be activated by setting the DMAR bit of the USART_CR3 register for receiving. Each time a byte is received, the DMA controller will transfer the data from the USART_DR register to the specified SRAM area (refer to DMA related instructions). The steps to assign a DMA channel for USART reception are as follows (x represents the channel number):

1) Configure the USART_DR register address as the source address of the transfer through the DMA control register. After each RXNE event, data will be read from this address and transferred to memory.

2) Configure the memory address as the destination address of the transfer through the DMA control register. Data will be transferred from USART_DR to this memory area after each RXNE event.

3) Configure the total number of bytes to be transferred in the DMA control register.

4) Configure the channel priority on the DMA register.

5) According to the requirements of the application, configure the DMA interrupt to be generated when the transfer is half or fully completed.

6) Activate the channel on the DMA control register.

When receiving the transfer amount specified by the DMA controller, the DMA controller generates an interrupt on the interrupt vector of the DMA channel.

32.3.13.3. Error flags and interrupt generation in multi-buffer communication

In the case of multi-buffer communication, if any error occurs during the communication, the error flag will be set after the current byte has been transferred. An interrupt will be generated if the interrupt enable bit is set. In the case of a single byte reception, the framing error, overflow error and noise flags that are set together with RXNE have separate error flag interrupt enable bits, if set, an interrupt will be generated after the current byte transmission is completed.

32.3.14. Hardware flow control

Using the nCTS input and nRTS output. The figure below shows how to connect 2 devices in this mode.

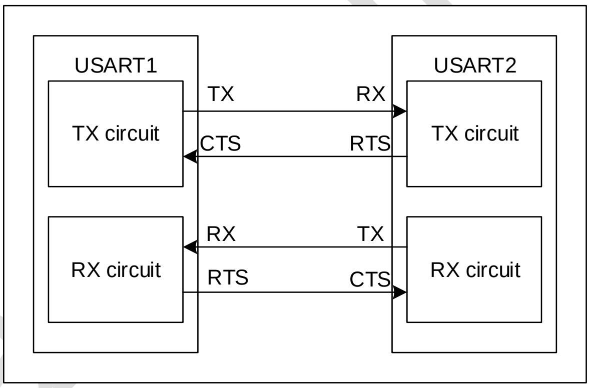


Figure 32-21 Hardware flow control between two USARTs

32.3.14.1. RTS flow control

If RTS flow control is enabled (RTSE = 1), nRTS becomes active (connected low) as soon as the USART receiver is ready to receive new data. When data arrives in the receive register, nRTS is released, thereby indicating that data transmission is to be stopped at the end of the current frame.

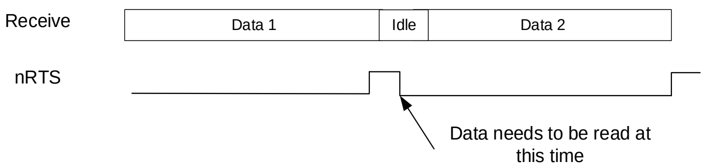


Figure 32-22 RTS flow control

32.3.14.2. CTS flow control

If CTS flow control is enabled (CTSE = 1), the transmitter checks the nCTS input before sending the next frame. If nCTS is valid (pulled to a low level), the next data is sent (assuming that data is ready to be sent, that is, TXE = 0), otherwise the next frame of data is not sent. If the nCTS is invalidated during transmission, the transmission stops after the current transmission is completed.

When CTSE = 1, as long as the nCTS input changes state, the hardware automatically sets the CTSIF status bit. It indicates whether the receiver is ready to communicate. An interrupt is generated if the CTSIE bit in the USART_CT3 register is set.

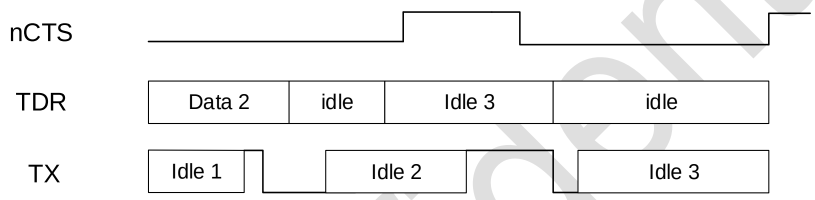


Figure 32-23 CTS flow control

32.4. USART interrupt request

Serial num-ber	Interrupt event	Event flag	Enable bit	Send/receive
1	Send data register empty	TXE	TXEIE	Send
2	CTS (Clear to Send)interrupt	CTSIF	CTSIE	Send
3	Transmission completed	TC	TCIE	Send
4	The receive register is not empty (read data is ready)	RXNE	RXNEIE	Take over
5	Overrun error	ORE		Take over
6	Idle frame	IDLE	IDLEIE	Take over
7	Parity error	PE	PEIE	Take over
8	Noise, overrun and frame errors when communicating with multiple processors	NR/ORE/FE	EIE	Take over

All USART interrupts share the same interrupt vector.

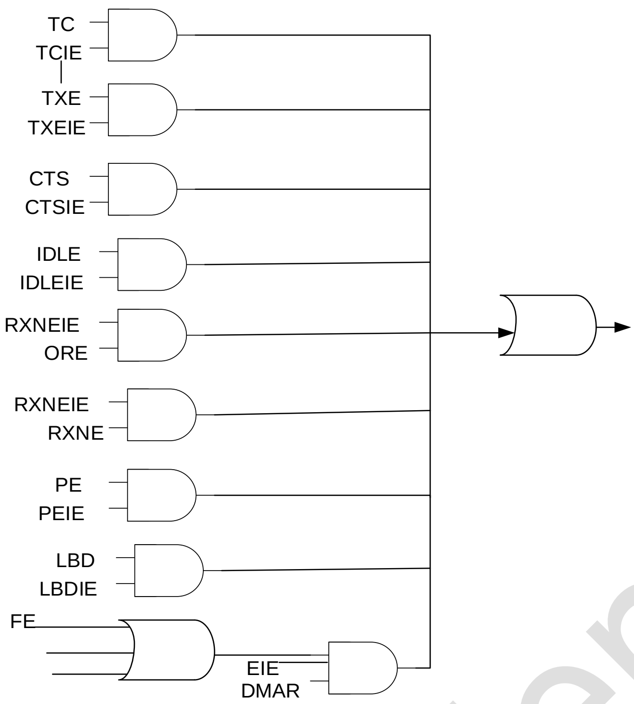


Figure 32-24 USART interrupt map

32.5. USART register

32.5.1. 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	Res	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
			W	R	R	RC_W0		R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	RES	-	Reserved
12	ABRRQ	W	0	Automatic baud rate request Writing 1 to this bit resets the ABRF flag and requests automatic baud rate detection for the next frame.
11	ABRE	R	0	Autobaud error flag.

Bit	Name	R/W	Reset Value	Function
				This register is set by hardware when there is an error in automatic baud rate detection (baud rate out of range or character comparison error). Software clears this bit by writing a 1 to the ABRRQ register.
10	ABRF	R	0	Automatic baud rate detection flag. This bit is set to 1 by hardware when auto-baud rate is set (set RXNE = 1 at the same time, an interrupt will be generated when the interrupt is enabled), or when an error occurs in the auto-baud rate detection operation (ABRE = 1, RXNE = 1, FE = 1). Software clears this bit by writing a 1 to the ABRRQ bit in the USART_RQR register.
9	CTS	RC_W0	0	When CTS input toggle, do not CTSE = 1, this register is 1. Software writes 0 to clear. When CTSIE = 1, a CTS interrupt is generated. 0: CTS line value unchanged 1: CTS line value change
8	LBD	Rc_W0		LBD: LIN break detection flag When a LIN break is detected, this bit is set to '1' by hardware and cleared to '0' by software (write to this bit). 0). If LBDIE = 1 in USART_CR3, an interrupt is generated. 0: no LIN disconnection detected; 1: LIN disconnection detected. Note: If LBDIE = 1, interrupt to be generated when LBD is '1'
7	TXE	R	1	Transfer register empty flag. This register is set by hardware when the USART_DR register data is transferred to the shift register. When TXEIE = 1, an interrupt is generated. Writing to the USART_DR register will clear this bit 0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	RC_W0	1	Transmission complete flag. After the transmission of the data frame is completed, and TXE = 1, the hardware will set this register. An interrupt is generated when TCIE = 1.

Bit	Name	R/W	Reset Value	Function
				Software reading the USART_SR register first and then writing the USART_DR register will clear this bit (for multiprocessor communication). Software can also write 0 to clear. 0: Transmission not completed 1: Transmission completed
5	RXNE	RC_W0	0	The read data register is not empty flag. This register is set by hardware when the shift register value is transferred to the USART_DR register. Software reads the USART_DR register, or writes 0 to clear this bit. When RXNEIE = 1, an interrupt is generated. 0: No data received 1: Receive data ready
4	IDLE	R	0	Idle sign. Detect IDLE line, the hardware sets this register. An interrupt is generated when IDLEIE = 1. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: IDLE not detected line 1: IDLE detected line
3	ORE	R	0	Overrun error flag. When RXNE = 1, the hardware sets this bit when the data received in the shift register is about to be transferred to the RDR register. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. When RXNEIE = 1, an interrupt is generated. 0: No Overrun error is generated 1: Generate Overrun error Note: When this register is set, the contents of the RDR register are not lost, but the contents of the shift register are oveRWritten. When EIE = 1, an ORE interrupt is generated.
2	NE	R	0	Noise error sign. This register is set by hardware when the data frame receives noise.

Bit	Name	R/W	Reset Value	Function
				Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: No noise error detected 1: Noise error detected Note: When RXNE and NE are generated at the same time, no interrupt is generated when NE = 1, but an interrupt is generated when the RXNE flag is set. In multi-buffer communication mode, NE = 1 will generate an interrupt when EIE = 1.
1	FE	R	0	Framing error flag. This bit is set by hardware when out-of-sync, excessive noise, or abort characters are detected. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. 0: no frame error detected 1: Framing 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 currently transmitted data has both a frame error and an overload error, the hardware will continue to transmit the data and only set the ORE flag. In multi-buffer communication mode, FE = 1 will generate an interrupt when EIE = 1.
0	PE	R	0	Checksum error. This register is set by hardware when the parity value is incorrect during reception. Software can clear this bit by reading the USART_SR register first and then the USART_DR register. But software must wait for RXNE = 1 before clearing this bit. When PEIE, an interrupt is generated. 0: No parity error is generated 1: Generate parity error

32.5.2. UASRT 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	RES	-	Reserved
8:0	DR[8:0]	RW	undefined	Receive/transmit data register. Depending on the read or write operation, the former is the received data and the latter is the transmitted data. The DR register is physically composed of two registers (one is the transmitted T DR, the other is the received R DR), so the DR register implements two functions of reading and writing. T DR register provides a parallel interface between the internal bus and the output shift register, and the R DR register provides a parallel interface between the input shift register and the internal bus. When parity is enabled for transmit operation, writing the M SB bit (bit7 or bit8) has no effect because it has been replaced by the parity bit. When the parity enable is turned on for a receive operation, the read MSB bit is the received parity bit.

32.5.3. 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 auto-baud detection mode, hardware updates this register.

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RES	-	Reserved
15:4	DIV_Mantissa[15:4]	RW	0	12bit integer
3:0	DIV_Fraction[3:0]	RW	0	4bit decimal

32.5.4. 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	SBK
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	RES	-	Reserved
13	UE	RW	0	USART enabled. When this bit is cleared, the USART module will immediately stop the current operation. This bit is set and cleared by software. 0: USART prescaler and output disabled, low-power mode 1: USART enable The software needs to wait for USART_ISR.TC to be set before clearing the UE bit and entering the low power mode, Also, the DMA channel needs to be disabled before clearing the UE bit.
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. How to wake up from mute mode. Set or cleared by software. 0: Idle line wake up 1: address wake-up
10	PCE	RW	0	Parity control. 0: Parity check disabled

Bit	Name	R/W	Reset Value	Function
				1: Parity check enabled Parity bit: 9th bit of 9bit, 8th bit of 8bit.
9	PS	RW	0	Parity check selection. Set and cleared by software. 0: Even parity 1: odd parity
8	PEIE	RW	0	PE interrupt enable. Set and cleared by software. 0: Disable 1: PE interrupt enable
7	TXEIE	RW	0	TXE interrupt enable. Set and cleared by software. 0: Disable 1: TX E interrupt enable
6	TCIE	RW	0	End of transfer interrupt enable. Set and cleared by software. 0: Disable 1: TC interrupt enable
5	RXNEIE	RW	0	RXNE interrupt enable, set and cleared by software. 0: Disable 1: ORE or RXNE interrupt enable
4	IDLEIE	RW	0	IDLE interrupt enable. Set and cleared by software. 0: Disable 1: IDLE interrupt enable
3	TE	RW	0	Transmission enable. 0: Transmission prohibited 1: Transmission enable
2	RE	RW	0	Receive enable. 0: Reception prohibited 1:Receive enable, start to detect start bit
1	RWU	RW	0	Receive wakeup. This bit indicates whether the USART is in mute mode. This register is set when a mute mode sequence is received, this register is cleared when a wake-up sequence is received. The specific wake-up sequence (address or IDLE) is controlled by the register USART_CR1.WAKEbit.

Bit	Name	R/W	Reset Value	Function
				0: The receiver is in working mode 1: The receiver is in silent mode Note 1: Before setting this bit to enter mute mode, the USART must have received a data byte, otherwise in mute mode, it cannot be woken up by idle bus detection. Note 2: When configured as address mark detection wake-up (WAKE = 1), the RWU bit cannot be modified by software when RXNE is set.
0	SBK	RW	0	Send break frame. Software sets this register to send the break byte. This register is cleared by hardware after the stop bit of the break frame is sent. 0: do not send break bytes 1: send break byte

32.5.5. 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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	RES	-	Reserved
14	LINEN	RW	0	LIN mode enable. Software set and cleared. 0: LIN mode; 1: LIN mode enabled; LIN mode sends LIN sync by enabling the SBK bit breaks (13 low), and detecting the Lin sync break character.

Bit	Name	R/W	Reset Value	Function
13：12	STOP[1:0]	RW	0	Stop bit configuration. 00: 1 stop bit; 01: 0.5 stop; 10: 2 stop bit; 11: 1.5stop
11	CLKEN	RW	0	CK pin enable. 0: disabled; 1: CK pin enable; This bit is reserved when synchronous mode is not supported.
10	CPOL	RW	0	Clock polarity. Synchronous mode, CK pin output clock polarity. 0: outside the transmission window, CK pin is stable low; 1: outside the transmission window, CK pin is stable high;
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 re-quired 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 data is not output at the CK pin; 1: the clock pulse of the last bit of data is output at the CK pin;
7	Reserved	RES	-	Reserved
6	LBDIE	RW	0	LIN break interrupt enable. 0: disable; 1: interrupt generation; This controls the LBD in the USART_SR regis-ter, so that LBD to 1 to generate an interrupt
5	LBDL	RW	0	LIN break detection length. 0: 10bits detection break; 1: 11bits detection break;

Bit	Name	R/W	Reset Value	Function
4	Reserved	RES		Reserved
3:0	ADD[3:0]	RW	4’b0	USART address. This register is used in multiprocessor mute mode and is used as the address when waking up with a 4bit address.

32.5.6. 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	NACK	HDSEL	IRLP	IREN	EIE
	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	RES	-	Reserved
14:13	ABRMOD[1:0]	RW	2’b0	Automatic baud rate detection mode. 00: measure the baud rate from the start bit 01: Falling edge to falling edge measurement 10: Reserved 11: Reserved When ABREN = 0 or UE = 0, this register is write-only.
12	ABREN	RW	0	Auto-baud rate enabled. 0: Disable 1: Auto baud rate enabled
11	OVER8	RW	0	Oversampling mode. 0: Oversampling by 16 1: Oversampling by 8 can only be written when U E = 0.
10	CTSIE	RW	0	CTS interrupt enable. 0: Forbidden,

Bit	Name	R/W	Reset Value	Function
				1: CTSIF interrupt enable,
9	CTSE	RW	0	CTS enabled. 0: CTS hardware flow control is disabled, 1: CTS mode enabled. Data is only transmitted when the CTS input is 0. At this point, after the data is written into the data register, the transmission will not be started until the CTS is valid.
8	RTSE	RW	0	RTS enabled. 0: RTS hardware flow control is disabled, 1: The RTS output is enabled, and the next data is requested only when the receive buffer is not full. After the current data is sent, the sending operation is suspended. If data can be received, set RTS to valid (0).
7	DMAT	RW	0	Enable DMA when transferring. 0: Forbidden, 1: Enable DMA during transfer.
6	DMAR	RW	0	DMA is enabled when receiving. 0: Forbidden, 1: Enable DMA when receiving.
5	SCEN	RW	0	Smart card mode enable. 0: disabled; 1: enable;
4	NACK	RW	0	Smart card NACK enable. 0: send NACK disable in case of parity error; 1: NACK enable sent in case of parity error;
3	HDSEL	RW	0	Half-duplex option. 0: Non-half duplex mode, 1: Half-duplex mode selection.
2	IRLP	RW	0	IrDA Low Power. 0: Normal mode 1: IrDA low power mode
1	IREN	RW	0	IrDA mode enable. Software enables and clears this register. 0: IrDA disable 1: IrDA enable
0	EIE	RW	0	Error interrupt enable. 0: Forbidden, 1: Frame error FE, overrun error ORE, noise NF interrupt enable.

32.5.7. Protection time and prescaler (USART_GTPR)

Address offset:0x18

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	RES	0	Reserved
15：8	GT[7:0]	RW	0	Guard time value This field defines the protection time in baud clock units. In smart card mode, this function is required. The send complete flag is set only when the protection time has elapsed.
7：0	PSC[7:0]	RW	0	Prescaler value - In IR (IrDA) low-power mode: PSC[7:0] = IR low-power baud rate. To divide the system clock to obtain the frequency in low-power mode: The source clock is divided by the value in the register (only 8 bits are valid) 00000000: reserved - do not write this value; 00000001: dividing the frequency of the source clock 1; 00000010: frequency division of the source clock 2; ...... - In normal mode for IR (IrDA): PSC can only be set to 00000001. - In smart card mode: PSC[4:0]: prescaler value The system clock is divided to provide the clock for the smart card. The value given in the register (the lower 5 bits are valid) is multiplied by 2 and used as a dividing factor for the source clock.

Bit	Name	R/W	Reset Value	Function
				00000: Reserved - do not write this value; 00001: divide the source clock by 2; 00010: divide the source clock by 4; 00011: divide the source clock by 6; ...... Note: Bits [7:5] have no meaning in smart card mode.

32.5.8. USART register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	USART_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABRRQ	ABRE	ABRF	CTS	LBD.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
	Reset value																				0	0	0	0	0	1	1	0	0	0	0	0	0
0x04	USART_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x08	USART_BRR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_Mantissa[11:0]												DIV_Faction[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	USART_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x10	USART_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINEN.	STOP		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE.	LBDL.	Res.	ADD[3:0]			
	Reset value																		0	0		0	0	0	0		0	0		0	0	0	0
0x14	USART_CR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABRMOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	USART_GTPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GT[7:0]							PSC[7:0]						
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0

33. CAN2.0 Controller (CAN2.0B)

33.1. Introduction

The CAN (Controller Area Network) bus is a bus standard that enables microprocessors or devices to communicate with each other without a host.

The CAN controller follows the CAN bus CAN2.0B protocol.

The CAN bus controller handles the sending and receiving of data on the bus and in this product the CAN controller has 12 filter groups. The filters are used to select the messages to be received for the application.

The application in the CAN controller can send transmit data to the bus via one high-priority Primary Transmit Buffer (hereinafter referred to as PTB) and three Secondary Transmit Buffers (hereinafter referred to as STB), with the transmit scheduler determining the order in which mailboxes are sent. The three STBs and the three RBs can be understood as two 3-stage FIFOs, which are fully controlled by the hardware.

The CAN bus controller can also support time-trigger communication.

33.2. CAN main features

- Fully supports the CAN2.0B protocol.
- CAN2.0 supports a maximum communication baud rate of 1Mbit/s
- Supports baud rate prescaling from 1 to 1/32, flexible configuration of baud rate
- 3 receive buffers
  - FIFO mode
  - Errors or non-received data do not overwrite stored messages
- 1 high priority master transmit buffer PTB
- 3 sub-send buffers STB
  - FIFO method
  - Priority arbitration method
- 12 independent filter groups

  - ➢ 11-bit standard ID and 29-bit extended ID support
  - ➢ Programmable ID CODE bits and MASK bits

- PTB/STB both support single send mode
- Silent mode support
- Supports loopback mode
- Support for capturing the type of error transmitted and locating the location of the arbitration failure
- Programmable error warning values
- Support for ISO 11898-4 time triggered CAN and receive timestamps

33.3. CAN functional description

33.3.1. Module block diagram

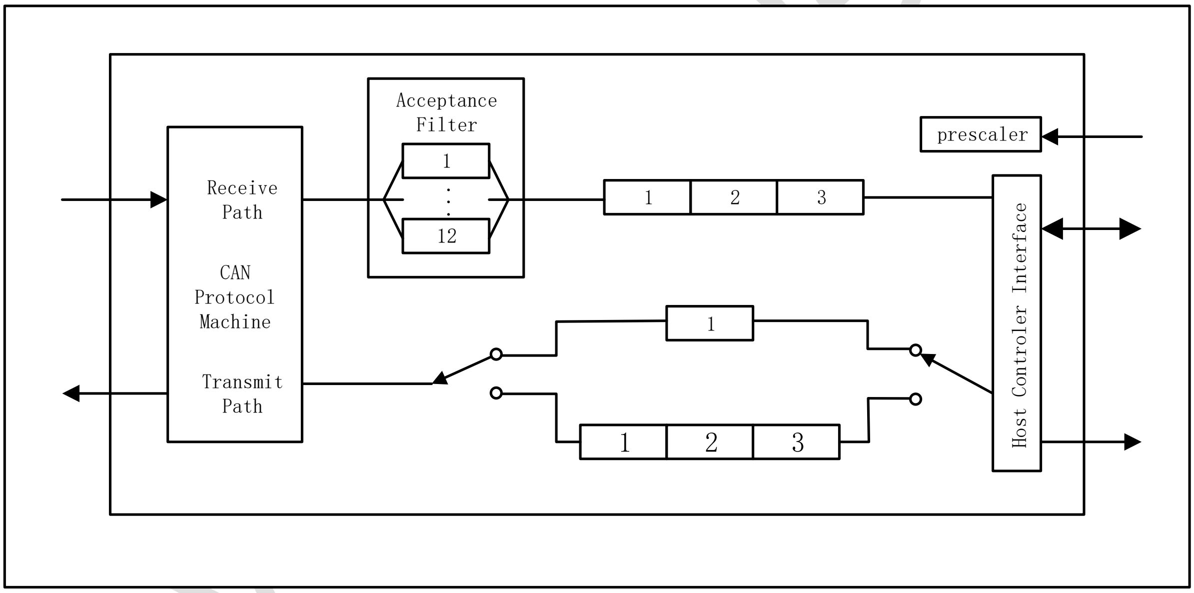


Figure 33-1 CAN module block diagram

33.3.2. Action modes

There are two modes of operation of the CAN controller, reset mode (CAN_MCR.RESET=1) and action mode (CAN_MCR.RESET=0). When the module is initialized, the registers that can only be operated in the reset mode should first be set in the reset mode (see the chapter Software Reset Function for details), then exit the reset mode and operate the remaining registers in the action mode.

33.3.3. Baud rate setting

CAN communication uses the clock can_clk whose clock source is an external high speed oscillator HSE or PLL. Before using the CAN module, the CAN communication clock needs to be set in the RCC chapter.

The diagram below shows the definition of the CAN bit times. The upper part of the dotted line is the bit time defined by the CAN protocol and the lower part of the dotted line is the bit time defined by this CAN controller CAN-CTRL. The CAN_ACBTR register can only be set when CAN_MCR.RESET=1, i.e. when the CAN software is reset.

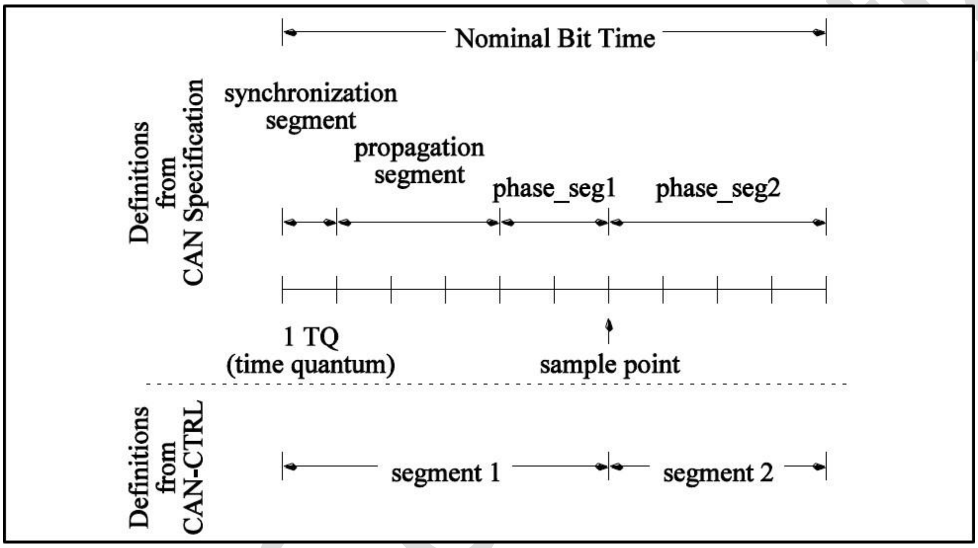


Figure 33-2 CAN bit time definitions

The TQ is calculated using the following formula, where PRESC is set via the PRESC bit of the CAN_RLSSP register. $fcan\_clk$ is the CAN communication clock frequency.

$$TQ = \frac{S\_PRESC+1}{fcan\_clk}$$

For the calculation of the bit time please refer to the following equation, where segment1 and segment2 are set via the AC_SEG_1 and AC_SEG_2 bits of the CAN_ACBTR register.

$$BT = tS\_segment1 + tS\_segment2 = ((AC\_SEG\_1 + 2) + (AC\_SEG\_2 + 1)) \times TQ$$

Table 33-1 CAN time setting rules

Bit	Setting range			Rule
AC_SEG_1 bit of the CAN_ACBTR register	[0..511]	CAN2.0 bits	(slow)	SEG_1 ≥ SEG_2 + 1

Bit	Setting range			Rule
AC_SEG_2bit of the CAN_ACBTR register	[0..127]	CAN2.0 bits	(slow)	
AC_SJW bit of the CAN_ACBTR register	[0..127]	CAN2.0 bits	(slow)	

Table 33-2 Recommended baud rate setting at 20MHz communication clock

Bit Rate [Mbit/s]	PSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25（Arbitration）	80	1	80	64	16	16	-
0.5（Arbitration）	80	1	40	32	8	8	-
0.5	80	1	40	32	8	8	-
1	80	1	20	16	4	4	16
2	80	1	10	8	2	2	8
4	80	1	5	4	1	1	4
5	75	1	4	3	1	1	3

Table 33-3 Recommended baud rate setting at 40MHz communication clock

Bit Rate [Mbit/s]	PSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN Communication clocks]
0.25（Arbitration）	80	2	80	64	16	16	-
0.5（Arbitration）	80	1	80	64	16	16	-
0.5	80	2	40	32	8	8	-
1	80	1	40	32	8	8	32
2	80	1	20	16	4	4	16
4	80	1	10	8	2	2	8

Bit Rate [Mbit/s]	PSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN Communi-cation clocks]
5	75	1	8	6	2	2	6
8	80	1	5	4	1	1	4

Table 33-4 Recommended baud rate setting at 80MHz communication clock

Bit Rate [Mbit/s]	PSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN communi-cation clock]
0.25（Arbi-tration）	80	4	80	64	16	16	-
0.5（Arbitra-tion）	80	2	80	64	16	16	-
0.5	80	4	40	32	8	8	-
1	80	2	40	32	8	8	64
2	80	2	20	16	4	4	32
4	80	1	20	16	4	4	16
5	75	1	16	12	4	4	12
8	80	1	10	8	2	2	8

33.3.4. Transmit buffers

The CAN controller provides two transmit buffers for sending data, the primary transmit buffer PTB and the secondary transmit buffer STB. PTB has the highest priority but can only buffer one frame of data, STB has a lower priority than PTB but can buffer three frames of data and the three frames of data in STB can work in FIFO mode or priority arbitration mode. All 3 frames in the STB can be sent by setting the TSALL bit in the CAN_MCR register to 1. In FIFO mode, the first data written is sent first, in priority mode, the data with the lowest ID is sent first.

The data in the PTB has the highest priority, so PTB sends can defer STB sends, but STBs that have already won arbitration and started sending cannot be deferred by PTB sends.

The PTB and STB can be accessed via the CAN_TBUF register. The next SLOT in the STB is selected via the TBSEL bit of the CAN_MCR register, TBSEL=0 for PTB and TBSEL=1 for STB. The correspondence is shown in the following diagram:

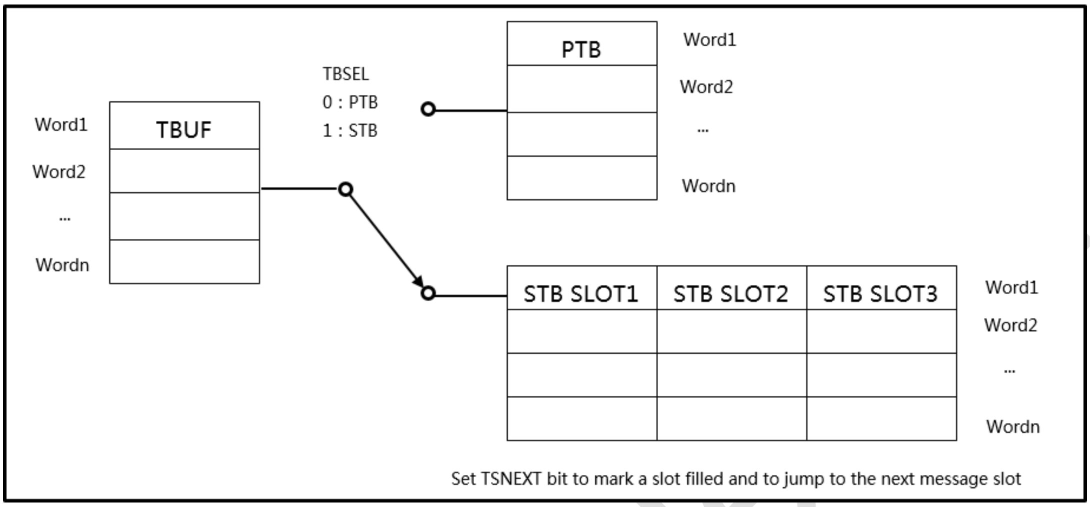


Figure 33-3 CAN TBUF register write send buffer and schematic

33.3.5. Receive buffers

The CAN controller provides three SLOT receive buffers for storing the received data, which operate in FIFO mode The RB SLOT reads the received data via the RBUF register, always reading the earliest received data first, and releases the RB SLOT already read via the RCTRL register with RREL set to 1 and point to the next RB SLOT.

The diagram for reading RB SLOT via CAN_RBUF is shown below.

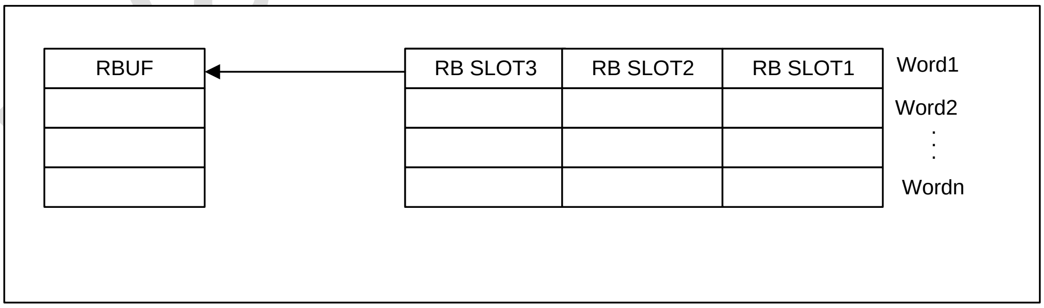


Figure 33-4 CAN RBUF register read receive buffer schematic

33.3.6. Receive filter register groups

The CAN controller provides 12 groups of 32-bit filters to filter the received data and thus reduce the CPU load, which can support either the standard format 11-bit ID or the extended format 29-bit ID.

The CODE register is used to compare the received CAN frames (for frame format please refer to the LLC frame format definition), while the MASK register is used as a mask for the CODE register and the corresponding bit in the CODE register is ignored if the corresponding MASK bit is 1.

The received data is received as long as it passes any of the 12 groups of filters and the received data is stored in the RB, otherwise the data is not received and not stored.

Each group of filters is enabled or disabled via the AE_n bit of the CAN_ACFCR register. ID CODE and ID MASK are accessed via the CAN_ACFC register and CAN_ACFM. The filters are selected via the ACFADR bit of the CAN_ACFCR register. The ACF register accesses the filter register group in the following figure.

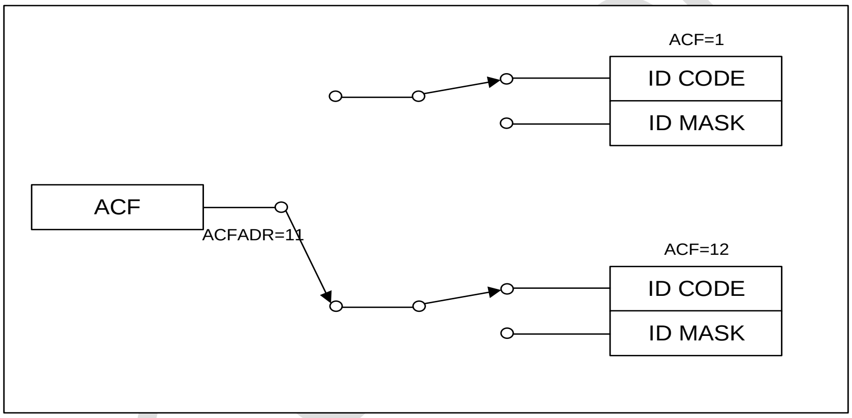


Figure 33-5 Schematic diagram of the CAN ACF register access filter group

33.3.7. Definition of the LLC frame format

The following table shows the definition of a logical link control frame (LLC) containing a time stamp. This uniform definition is used for sending frames (stored in the TBUF), receiving frames (read from the RBUF) and configuring the receiving filters (ACFC and ACFM)

Address offset:	**register**																
0x00	CAN_ID	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		Res.	Res.	Res.	ID[28:16]												
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		ID[15:0]															
0x04	CAN_FORMAT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Address offset:	register																
		Res.	Res.	Res.	LBF	Res.	KOER[2:0]			Res.	Res	Res.	RMF.	Res	Res	FDF.	IDE
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Res.	Res.	Res.	Res.	Res.	DLC[10:0]										
0x08	CAN_TYPE	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		HANDLE[7:0]								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.															
0x18	CAN_TTCAN	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
		CYCLE_TIME[15:0]															
0x1c	CAN_DATA1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		Data[31:16]															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Data[15:0]															
0x20	CAN_DATA2	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		Data[31:16]															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Data[15:0]															

Bit	Description
ID	Frame identifier
DLC	Data Length Code The DLC defines the number of bytes of payload within the frame. Classic CAN 2.0B frames have a DLC length of 4 bits (DLC(3:0)). For classic CAN 2.0B, it is possible to choose to transmit remote frames where the DLC is not meaningful. The explanation of the bit RMF provides more details.
IDE	ID format 0 - Standard format: ID(28:18) 1 - Extended format: ID(28:0)
FDF	This bit should be set to 0 to support the CAN 2.0 frame
RMF	Remote frames 0 - Data frames

Bit	Description
	1 - Remote frame Remote frames carry a zero byte payload. The value of the DLC is transmitted as is, but has no effect on the frame size. Therefore, the DLC value of a remote frame may carry some encoding information. Only classic CAN 2.0B frames can be remote frames.
KOER	Error type The KOER of the receive frame has the same meaning as the KOER bit in register EALCAP. If RBALL=1, the KOER of the received frame becomes meaningful. Note that if RBALL=1, the filter set is normally disabled.
LBF	Loopback frames If loopback mode is activated and the CAN controller has received its own transmission frame, the LBF of the received frame is set to 1. This is useful if LBME = 1 and other nodes in the network also transmit.
HANDLE	Frame identification HANDLE The purpose of the handle is to identify the frame using TSTAT. HANDLE is not used in MAC frames. It is recommended that the host application write the value of the software counter to HANDLE. Such a software counter can be incremented with each new frame to be transmitted and should be flipped. Note: To avoid uninitialised memory problems during simulation, the host application should always define HANDLE for each frame to be transmitted.
CYCLE_TIME	CYCLE_TIME (timestamp of the TTCAN) CYCLE_TIME will store only the received frames. This is the cycle time at the SOF of the frame. The cycle time of the reference message is always 0.
Data	Payload data for the frame. Classic CAN 2.0B has a maximum of 8 bytes.

33.3.8. Data transmission

At least one frame of data must be loaded in the PTB or STB before transmission is started, the TPE is locked during PTB transmission and the STB filling can be confirmed by the TSSTAT bit. To set up the transmit data, proceed as follows:

- Set TBSEL to select the BUF to be sent from the PTB and STB
- Write the data to be sent via the CAN_TBUF register. 3.
- If STB is selected, set TSNEXT=1 to complete the loading of the STB SLOT. 4.
- Send enable
- PTB send using TPE
- STB send using TSALL or TSONE
- Send completion status confirmation
- PTB send completion using TPIF, TPIE for TPIF enable

- STB Use TSONE to send completion with TSIF, TSIE to enable TSIF
- STB uses TSIF for TSALL transmission completion, when all STB SLOT data to be set are sent before TSIF is set and TSIE is used to enable TSIF

33.3.9. Single data transmission

The single transmission trigger is intended for exclusive use in time windows where a message needs to be transmitted in single transmission mode. The selected message is defined by TTPTR.

The single shot mode is automatically used regardless of the status of RELIM. The register RELIM is ignored and remains unchanged.

The single shot trigger is intended to be used for proprietary time windows. For this purpose, ISO 11898-4 defines a transmit enable window of up to 16 cycles of time ticks. Register bits TEW(3:0)+1 define

The number of scales. If the bus is occupied by another frame, a frame cannot be started. This should not happen in an exclusive time window, but the send enable window ensures that there is no delay in starting, which would result in a violation of the next time window. If the send enable window is closed and the frame cannot be started, it is aborted. As a result, the TB time slot of the frame will be marked empty and if AIE is set, AIF will be set. The frame data in the TB slot will not be touched, so if the same data is next attempted to be transmitted, the slot will simply be marked as filled again.

If TT_TRIG is lower than the actual cycle time, TEIF is set and no action is performed.

33.3.10. Cancellation of data transmission

A data transmission that has been requested but not yet executed can be cancelled via TPA or TSA. Cancellation of a data transmission can occur in the following cases:

1. in arbitration

The node fails arbitration, the data transmission is cancelled.

If arbitration of the node is successful, the transmission continues.

2. Data is being sent

- data is successfully sent and ACK is received, the corresponding flag and status are set normally. The data send is not cancelled.
- Data sent successfully but ACK not received, data send cancelled and error counter increased.
- TSALL=1 set to send data, STB SLOT data being sent is sent normally, STB SLOT that has not started sending is cancelled.

The cancellation of data transmission results in the following two cases.

1. The TPA releases the PTB and makes TPE=0.

2. The TSA releases one STB SLOT or all STB SLOTs depending on whether the transmission is enabled by TSONE or TSALL.

33.3.11. Data reception

The receive filter group reduces the CPU load by filtering out unwanted receive data and reducing the occurrence of interrupts and RB reads. To set up the receive data, proceed as follows:

1. Set the filter group.

2. Set RFIE, RAFIE and AFWL. 3.

3. Wait for RFIF or RAFIF. 4.

4. Read the first received data from the RB FIFO via CAN_RBUF.

5. Set RREL=1 and select the next RB SLOT.

6. Repeat 4, 5 until RB is confirmed empty by RSTAT.

33.3.12. Error handling

The CAN controller can, on the one hand, handle some errors automatically, e.g. by automatically retransmitting data or discarding received frames containing errors, and on the other hand, report errors to the CPU via interrupts.

The CAN node has the following three error states:

- Error active: the node automatically sends an active error flag when it detects an error.
- Error Passive: The node automatically sends a passive error flag when it detects an error.

- Node off: In the off state the node no longer affects the entire CAN network.

The CAN controller provides two counters TECNT and RECNT in the CAN_WECR register for error counting; the TECNT and RECNT counters are incremented and decremented according to the rules specified in the CAN protocol. A programmable CAN error warning LIMIT register is also provided to generate an error interrupt to notify the CPU.

There are five types of errors during CAN communication, which can be identified by the KOER bit of the CAN_WECR register.

- Bit errors
- Form errors
- Padding error
- Answer error
- CRC error

33.3.13. Node shutdown

When the number of transmit errors is greater than 255, the CAN node automatically enters the Node Off state and does not participate in CAN communication until it returns to the Error Active state. The CAN node shutdown state can be confirmed by the BUSOFF bit in the CAN_MCR register, and an EIF interrupt is generated when BUSOFF is set.

The CAN can be restored from the node off state to the error active state in two ways:

- Power-on reset
- Receiving a sequence of 128 consecutive 11-bit invisible bits (recovery sequence)

In the node off state, the TECNT value remains unchanged and RECNT is used to count the recovery sequence. After recovery from the node off state, TECNT and RECNT are reset to 0.

33.3.14. Arbitration failure location capture

The CAN controller is capable of capturing the exact position of the arbitration failure bit and reflecting it in the WECR.ALC register.The WECR.ALC register holds the position of the last arbitration failure bit and the WECR.ALC bit is not updated if the node wins the arbitration.The ALC value is defined as follows

After the SOF bit, the first ID data bit ALC is 0, the second ID data bit ALC is 1, and so on. The maximum value of ALC is 31 since arbitration only occurs within the arbitration field. e.g. if a standard format remote frame and an extended frame arbitrate and the extended frame fails at the IDE bit, then ALC = 12.

33.3.15. Loopback modes

The CAN controller supports the following two loopback modes:

- Internal loopback
- External loopback

Both loopback modes can receive their own data frames and are mainly used for test purposes.

In internal loopback mode, the module internally connects the receive data line to the transmit data line and the transmit data is not output. In internal loopback mode, the node generates a self-answer signal to avoid ACK errors.

In external loopback mode the connection to the transceiver is maintained so that the sent data is still present on the CAN bus and the CAN receives its own sent data with the help of the transceiver. The external loopback mode can be determined by the SACK bit in the CAN_WECR register, which does not generate a self-answer signal when SACK=0 and generates a self-answer signal when SACK=1.

In external loopback mode, when SACK=0, the following two situations occur:

- Other nodes also receive the data frame sent by this node and send an answer signal, in this case this node can successfully send and receive data.

- If no other node returns an answer signal, an answer error will be generated and the data will be resent and the error counter will be increased. In this case it is recommended to use the single send mode.

When returning from loopback mode to normal mode, in addition to clearing the mode bit, a software reset of the CAN controller is required.

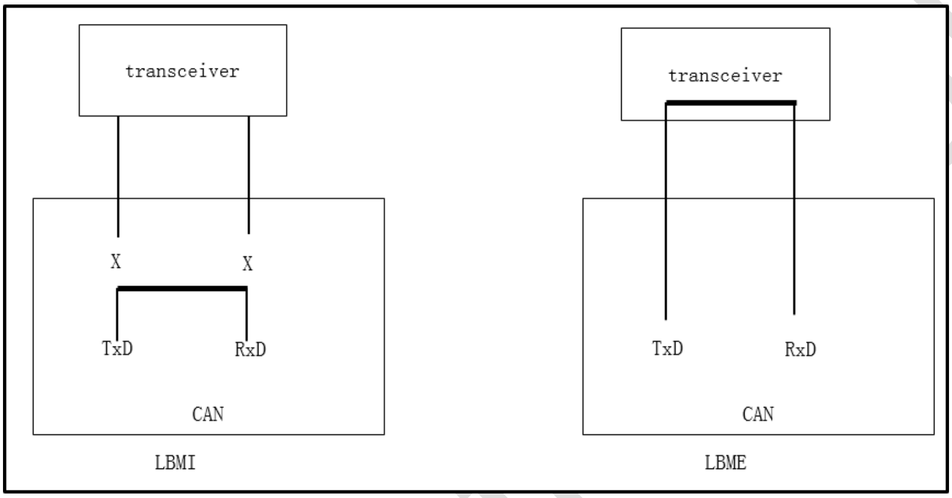


Figure 33-6 Diagram of CAN internal loopback LBMI and external loopback LBME

33.3.16. Silent Mode

The silent mode can be used to monitor the CAN network data. In silent mode, data can be received from the CAN bus and no data is sent to the bus. Set the LOM in the CAN_MCR register to 1 to put the CAN bus controller into silent mode and clear it to 0 to leave silent mode.

The external loopback mode can be combined with the silent mode to form the external loopback silent mode, where the CAN can be considered a quiet receiver, but can send data when necessary. In external loopback silent mode, frames containing self-answering signals are allowed to be sent, but the node does not generate error flags and overloaded frames.

33.3.17. Software reset function

The software reset function is implemented by setting the RESET bit of the CAN_MCR register to 1. The reset range of the software reset function is shown in the table below.

Table 33-5 Software Reset Range Table

Register Bit Name	Software Reset	Remarks	Register Bit Name	Software Reset	Remarks
ACFADR	NO	-	AC_SEG_1	YES	Writable only during software reset
ACODE	NO	Writable only during software reset	AC_SEG_2	YES	Writable only during software reset
AE_x	NO	-	AC_SJW	YES	Writable only during software reset
AFWL	NO	-	KOER	YES	-
AIF	YES	-	LBME	YES	-
ALC	YES	-	LBMI	YES	-
ALIE	NO	-	RACTIVE	YES	Receiving stops immediately and no ACK is generated
ALIF	YES	-	RAFIE	NO	-
AMASK	NO	Writable only during software reset	RAFIF	YES	-
BEIE	NO	-	RBALL	YES	-
BEIF	YES	-	RBUF	YES	RB is marked as empty, value variable
BUSOFF	NO	Clear by writing 1	RECNT	NO	Zeroing via BUSOFF write 1
EIE	NO	-	REF_ID	NO	-
EIF	NO	-	REF_IDE	NO	-
EPASS	NO	-	RFIE	NO	-
EPIE	NO	-	RFIF	YES	-
EPIF	YES	-	RIE	NO	-
EWARN	NO	-	RIF	YES	-
EWL	YES	-	ROIE	NO	-
FRESC	NO	Writable only during software reset	ROIF	YES	-
ROV	YES	-	ROM	NO	
RREL	YES	-	TSMODE	NO	
RSTAT	YES		TSNEXT	YES	-

Register Bit Name	Software Reset	Remarks	Register Bit Name	Software Reset	Remarks
SACK	YES	-	TSONE	YES	-
SELMASK	NO	-	TPIE	NO	-
SSPOFF	YES	-	TPIF	YES	-
TACTIVE	YES	Sending stops immediately	TPSS	YES	-
TBE	YES	-	TSFF	YES	All STB SLOTs are marked as empty
TBF	NO	-	TSIE	NO	-
TBPTR	NO	-	TSIF	YES	-
TBSEL	YES	-			
TBUF	YES	STB is marked as empty, pointing to PTB	TSSTAT	YES	All STB SLOTs are marked as empty
TDCEN	YES	-	TTEN	YES	-
TECNT	NO	Can be cleared by BUSOFF=1	TTIF	YES	-
TEIF	YES	-	TTIE	NO	-
TPA	YES	-	TTPTR	NO	-
TPE	YES	-	TTTBM	NO	-
TSA	YES	-	TTYPE	NO	-
TSALL	YES	-	TT_TRIG	NO	-
WTIE	NO	-	TT_WTRIG	NO	-
WTIF	YES		T_PRESC	NO	-

33.3.18. Time-triggered TTCAN

CAN-CTRL provides partial (lever 1) hardware support for the time-triggered communication method specified in ISO 11898-4. This section describes the TTCAN functionality in the following 5 sections.

33.3.18.1. TBUF behaviour in TTCAN mode

TTTBM=1

When TTTBM=1, the PTB and STB SLOT alike form a TB SLOT, which is specified by the TBPTR bit of the CAN_TTCR register to send a BUF, where TBPTR=0 points to the PTB, TBPTR=1 is to the STB SLOT1 and so on. The host can use the TBE and TBF registers to mark the transmit BUF SLOT, where the TBSEL and TSNEXT registers have no meaning and can be ignored.

When TTTBM=1, the PTB does not have any special attributes and, like the STB SLOT, the transmission completion flag is TSIF.

In TTCAN mode, the transmit BUF has no FIFO mode and no priority arbitration mode, and only one selected SLOT can send data.

In TTCAN mode, the start of transmission is time-triggered and TPE, TSONE, TSALL and TPA are fixed to 0 and ignored.

TTTBM=0

When TTTBM=0, a combination of event driven communication and receive timestamp functions are used. In this mode, the PTB and STB functions are the same as when TTEN=0, so the PTB always has the highest priority, while the STB can operate in FIFO mode or arbitration mode.

33.3.18.2. TTCAN function

After power-up, the Time Master needs to be initialized according to the ISO 11898-4 protocol. There can be up to 8 potential Time Masters in a CAN network, each with its own reference message ID (last 3 digits of the ID). After TTEN = 1, the 16-bit counter starts working and when the reference message is successfully received or the Time Master successfully sends a reference message, the CAN controller copies the Sync_Mark to the Ref_Mark, which sets the cycle time to 0. A successful reception of a reference message sets the RIF flag and a successful transmission of a reference message sets the TPIF or TSIF flag. At this point the host needs to prepare the trigger condition for the next action.

The trigger condition can be a receive trigger. This trigger only triggers an interrupt that can be used to detect if the expected message has been received. The trigger condition can also be a send trigger. This trigger starts sending the data in the TBUF SLOT specified by the TTPTR register. If the selected TBUF SLOT is marked empty, no sending is started, but the interrupt flag is set.

33.3.18.3. TTCAN Timing

The CAN controller supports ISO 11898-4 level 1. contains a 16-bit counter operating at the bit times defined by AC_PRESC, AC_SEG_1, AC_SEG_2. If TTEN = 1, there is an additional prescaler T_PRESC.

The value of the counter at SOF of a frame of data is Sync_Mark. if the frame of data is a reference message, Sync_Mark is copied to Ref_Mark. cycle time is equal to the value of the counter minus Ref_Mark. this time is used as a timestamp for the received message or as a trigger time reference for the sent message.

33.3.18.4. TTCAN triggering method

The TTCAN trigger method is defined via the TTYPE register, the TTPTR register specifies the sending SLOT, and TT_TRIG specifies the cycle time of the trigger.

The following five trigger methods are included:

- Immediate trigger
- Time trigger
- Single send trigger
- Send start trigger
- Send Stop Trigger

All triggers use the TTIF flag except for the immediate trigger method, which only supports the time trigger method when TTTBM=1.

Immediate trigger

The trigger is started by writing the high bit of TT_TRIG (regardless of the value written). In this mode, the data in the TBUF SLOT selected by TTPTR is sent immediately. TTIF is not set.

Time triggering

The time-triggered mode only generates an interrupt by setting the TTIF flag and has no other function. If a node expects to receive the expected data within a specific time window, the time triggered method can be used. If the TT_TRIG value is less than the actual cycle time, then TEIF is set and no other action is taken.

Single send trigger

The single send trigger method is used to send data within the execution time window.

If the data does not start to be sent within the specified send enable time window, the frame is discarded. The corresponding transmit BUF SLOT is marked empty and the AIF is set. The data in the corresponding transmit BUF is not rewritten as it can be sent again by setting the TPF.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no other action is taken.

Send start trigger

The send start trigger method is used within the arbitration time window to participate in arbitration. If the message specified in the TTPTR register is not successfully sent, the send stop trigger can be used to stop the send.

If the TT_TRIG value is less than the actual cycle time, TEIF is set and no other action is taken.

Send Stop Trigger

The send stop trigger is used to stop a send that has been started by the send start trigger. If a transmission is stopped, the transmission frame is discarded, AIF is set and the selected TBUF SLOT is marked empty, but the data in the TBUF SLOT is not rewritten and can be sent again by setting the TBF.

If the TT_TRIG value is less than the actual cycle time then TEIF is set and execution stops.

33.3.18.5. TTCAN trigger watch time

The TTCAN trigger watch time function is similar to the watchdog function and is used when TTTBM=1. It is used to see how long the gate has been in operation since the last successful reception of a reference message. The reference message can be received during the cycle time or after an event and the application should set the appropriate watchdog time for the specific case.

If cycle count is equal to TT_WTRIG, WTIF is set. write 0 via WTIE to turn off the watchdog trigger.

If TT_WTRIG is smaller than the actual cycle time, then TEIF is set.

33.3.19. Interruptions

Interruption flags	Description
RIF	Receiving interruptions
ROIF	Receive overflow interrupt
RFIF	Receive BUF full interrupt
RAFIF	Receive BUF will be full interrupt
TPIF	PTB transmit interrupt
TSIF	STB send interrupt
EIF	Error interruption
AIF	Cancel send interrupt
EPIE	Error Passive Interruption
ALIF	Failed arbitration interruption
BEIF	Bus error interrupt
WTIF	Trigger watchdog interrupt
TEIF	Trigger an error interrupt
TTIF	Time-triggered interruptions

33.4. Register Descriptions

33.4.1. Node Configuration Register (CAN_TSNCR)

Address offset: 0x00

Reset value: 0x02010801

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.	ROP	Res.
														rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VERSION[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
31：18	Reserved			
17	ROP	rw	0	Restricted operations

Bit	Name	R/W	Reset Value	Function
				0 - Restricted operation not enabled 1 - Restricted operation enabled ROP cannot be changed while the transmission is active. If ROP is enabled, data frame transmission cannot be initiated.
16	CES	rw	1	CAN error signals 0-Disable the error signal 1-Enables the error signal If CES=1, an error is signalled using the error flag and the error counter is incremented. This behaviour is equivalent to as defined in ISO 11898-1:2015 (classic CAN). Otherwise , if CES=0, the error will result in a protocol exception event and will not modify the error counter.
15:0	Version [15:0]	r		CAN-CTRL version. VER_1 holds the major version, VER_0 holds the minor version. Example: Version 5x15N00S00 is represented by VER_1=5 and VER_0=15=0x0f.

33.4.2. Bit Timing Configuration Register (CAN_ACBTR)

Address offset: 0x04

Reset value: 0x05050008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	AC_SJW[6:0]							Res.	AC_SEG_2[6:0]						
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	AC_SEG_1[8:0]								
							rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
30:24	AC_SJW	rw	7’h05	Synchronous jump width The sync jump width $tSJW$ = (AC_SJW + 1) · TQ is the maximum time to shorten or lengthen the resync bit time by the maximum time.
22:16	AC_SEG_2	rw	7’h05	Bit Timing Segment 2 Time SEG_2 = (AC_SEG_2 + 1) · TQ End from sample point in place
8:0	AC_SEG_1	rw	9’h08	Bit Timing Segment 1

Bit	Name	R/W	Reset Value	Function
				The sample point will be set after the start of the bit time to $tSEG\_1 = (AC\_SEG\_1 + 2) \cdot TQ$.

33.4.3. Limit and Prescaler Configuration Register (CAN_RLSSP)

Address offset: 0x10

Reset value: 0x77000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	RETLIM[2:0]			Res.	REALIM[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	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.	PRESC[4:0]				
											rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
30:28	RETLIM	rw	3'h07	Limit on the number of automatic retransmissions 111: no limit (limited only by sending error counter TECNT) 110: 7 attempts ... 000: 1 attempt (no retransmission) If any errors occur during transmission, the CAN node can automatically retry when the bus is idle. The number of re-transmission attempts can be limited using RETLIM. RETLIM can be updated at any time, but after the update, the contents of the registers take a few clocks to be syn-chronised to the CAN protocol machine, during which time REALIM cannot be written to again
26:24	REALIM	rw	3'h07	Limit on the number of re-arbitrations 111: No limit 110: 7 attempts ... 000: 1 attempt (no automatic arbitration) If two or more CAN nodes try to transmit frames at the same time, the lower priority frame loses arbitration and backs off silently without interrupting the higher priority frame. CAN nodes can automatically retry when the bus is idle. Using REALIM you can limit the number of re-arbitra-tion attempts.

Bit	Name	R/W	Reset Value	Function
				RELIM can be updated at any time, but after the update, the contents of the registers take a few clocks to synchronise to the CAN protocol machine, during which time REALIM cannot be written to again.
23:5	Reserved			
4:0	PRESC	rw	0	Prescaler (PRESC+1) divides the input clock of the module to obtain TQ

33.4.4. Status Register (CAN_IFR)

Address offset: 0x14

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EWARN	EPASS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
r	r														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	WTIF	TEIF	TTIF	EPIF	ALIF	BEIF	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31	EWARN	r	0	The set ERROR WARNING LIMIT is reached 0: RECNT or TECNT is less than the EWL setting 1: RECNT or TECNT is greater than or equal to the EWL setting
30	EPASS	r	0	Error Passive mode active 0: node is an active error node 1: Node is passive error node
29:14	Reserved			
13	WTIF	rw	0	TTCAN: Watch Trigger Interrupt Flag If the cycle count reaches the limit defined by TT_WTRIG and WTIE is set, WTIF will be set.
12	TEIF	rw	0	TTCAN: Trigger error interrupt flag The conditions for setting TEIF are described in the TTCAN chapter; there are no bits to enable or disable TEIF processing
11	TTIF	rw	0	Time-triggered interrupt flag

Bit	Name	R/W	Reset Value	Function
				TTIF is set when CYCLECOUNT value = TT_TRIG set value and TTIE = 1. If TT_TRIG is not updated, TTIF is only set once and will not be set for the next count cycle, even if the CYCLECOUNT value = TT_TRIG. Clear this flag bit by writing 1 to the application.
10	EPIF	rw	0	Error Passive Interrupt Flag. EPIF will be set if EPASS changes and EPIE=1. Clear this flag bit by writing 1 to the application.
9	ALIF	rw	0	Arbitration Lost Interrupt Flag 0: Arbitration successful 1: Arbitration failed Clear this flag bit by writing 1 to the application.
8	BEIF	rw	0	Bus Error Interrupt Flag 0: No bus error 1: Bus error Clear this flag bit by writing 1 to the application.
7	RIF	rw	0	Receive Interrupt Flag 0: No data frame received 1: A valid data frame or remote frame was received The flag is cleared by writing 1 to the application.
6	ROIF	rw	0	Receive Overrun Interrupt Flag 0: No RBs are overwritten (overwrite) 1: At least one of the RBs is overwritten The flag is cleared by writing 1 to the application.
5	RFIF	rw	0	Receive BUF Full Interrupt Flag (RB Full Interrupt Flag) 0: RB FIFO is not full 1: RB FIFO full Clear this flag bit by writing 1 to the application.
4	RAFIF	rw	0	Receive BUF Almost Full Interrupt Flag 0: The number of RB SLOTs filled is less than the AFWL setting 1: The number of RB SLOTs filled is greater than or equal to the AFWL setting The flag is cleared by writing 1 to the application.
3	TPIF	rw	0	PTB transmit interrupt flag (Transmission Primary Interrupt Flag) 0: No PTB transmission completed 1: The requested PTB transmission completed success-fully by the application writing 1 to clear this flag bit.

Bit	Name	R/W	Reset Value	Function
				Note: TPIF is not valid in TTCAN mode, only the TSIF flag applies
2	TSIF	rw	0	STB Transmission Secondary Interrupt Flag 0: No STB transmission completed 1: The requested STB transmission is successfully completed by clearing this flag bit by writing 1 to the application. Note: TPIF is not valid in TTCAN mode, only the TSIF flag is used
1	EIF	rw	0	Error interrupt flag (Error Interrupt Flag) 0: The BUSOFF bit has not changed, or the value of the error counter has not changed in relation to the ERROR warning limit setting. 1: The BUSOFF bit has changed, or the value of the error counter has changed in relation to the ERROR warning limit setting. For example, the value of the error counter changes from less than the set value to more than the set value, or from more than the set value to less than the set value. The flag bit is cleared by writing 1 to the application.
0	AIF	rw	0	Abort Interrupt Flag 0: No transmission data cancelled 1: The send message requested via TPA or TSA has been successfully cancelled. This flag bit is cleared by an application write 1.

33.4.5. Interrupt Enable Register (CAN_IER)

Address offset: 0x18

Reset value: 0x000468FE

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.	WTIE	Res.	TTIE	EPIE	ALIE	BEIE	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	Res.
		rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
13	WTIE	rw	1	Watch Trigger Interrupt Enable 0: Disable 1: Enabled
11	TTIE	rw	1	Time Trigger Interrupt Enable 0: Disable 1: Enabled
10	EPIE	rw	0	The set ERROR WARNING LIMIT is reached 0: RECNT or TECNT is less than the EWL setting 1: RECNT or TECNT is greater than or equal to the EWL setting
9	ALIE	rw	0	Arbitration Lost Interrupt Enable 0: Disable 1: Enabled
8	BEIE	rw	0	Bus Error Interrupt Enable 0: Disable 1: Enabled
7	RIE	rw	0	Receive Interrupt Enable 0: Disable 1: Enabled
6	ROIE	rw	0	Receive Overrun Interrupt Enable 0: Disable 1: Enabled
5	RFIE	rw	0	Receive BUF Full Interrupt Enable (RB Full Interrupt Enable) 0: Disable 1: Enabled
4	RAFIE	rw	0	Receive BUF Almost Full Interrupt Enable (RB Almost Full Interrupt Enable) 0: Disable 1: Enabled
3	TPIE	rw	0	PTB Transmission Primary Interrupt Enable 0: Disable 1: Enabled
2	TSIE	rw	0	STB Transmission Secondary Interrupt Enable 0: Disable 1: Enabled
1	EIE	rw	0	Error Interrupt Enable 0: Disable 1: Enabled

33.4.6. Transmission Status Register (CAN_TSR)

Address offset: 0x1C

Reset value: 0x00000000

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

Bit	Name	R/W	Reset Value	Function
26:24	TSTAT_H	rw	0	Status code of a transmitted frame that has completed transmission
23:16	HANDLE_H	rw	0	The value of the handle of the transmitted frame that has completed transmission
10:8	TSTAT_L	rw	0	Status code of the currently transmitting frame
7:0	HANDLE_L	rw	0	Handle value of the currently transmitting frame

33.4.7. Global Configuration Register (CAN_MCR)

Address offset:0x28

Reset value:0x00900080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SACK	ROM	ROV	RREL	RBALL	Res.	RSTAT[1:0]		Res.	TSNEXT	TSMODE	TTTBM	Res.	TSFF	TSSTAT[1:0]	
rw	rw	r	rw	rw		r	r	rw	rw	rw	rw		r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA	RESET	LBME	LBMI	Res.	Res.	Res.	Res.	BUSOFF
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw					rw

Bit	Name	R/W	Reset Value	Function
31	SACK	rw	0	Self-ACKnowledge 0: No self-ACKnowledge 1: Self-ACKnowledge is enabled when LBME=1

Bit	Name	R/W	Reset Value	Function
30	ROM	rw	0	Receive buffer overflow mode set position (Receive buffer overflow mode) 0: The earliest received data is overwritten 1: Newly received data is not stored
29	ROV	r	0	Receive BUF overflow flag bit (Receive buffer OVerflow) 0: No overflow 1: Overflow, at least one data loss Cleared by writing RREL to 1.
28	RREL	rw	0	Release receive BUF (Receive buffer RELease) 0: not released 1: indicates that the receive BUF has already been read and the RBUF register points to the next RB SLOT.
27	RBALL	rw	0	Receive Buffer stores ALL data frames 0: normal mode 1: stores all data including those with errors.
25:24	RSTAT	r	0	Self-ACKnowledge 0: No self-ACKnowledge 1: Self-ACKnowledge is enabled when LBME=1
23	Reserved			
22	TSNEXT	rw	0	Next STB (Transmit buffer Secondary NEXT) 0: No action 1: Current STB SLOT is filled, pointing to the next SLOT After the application has written the data in the TBUF, the application identifies that the current STB SLOT has been filled by setting the TSNEXT bit, so that the hardware can point the TBUF to the next STB SLOT. The data in the STB SLOT identified by the TSNEXT bit can be sent via the TSONE or TSALL bit. This bit is cleared by the application writing 1 and the hardware clearing zero. After all STB SLOTs have been filled, TSNEXT remains at 1 until an STB SLOT is released. Note: This bit is fixed to 0 in TTCAN mode.
21	TSMODE	rw	0	STB transmit mode (Transmit buffer Secondary operation MODE) 0: FIFO mode 1: Priority mode FIFO mode sends data frames in the order in which they are written.

Bit	Name	R/W	Reset Value	Function
				The priority mode is automatically determined by the ID, the smaller the ID the higher the priority. Regardless of the mode, the PTB has the highest priority. Note: The TSMODE bit can only be set when the STB is empty.
20	TTTBM	rw	1	TTCAN BUF Mode (TTCAN Transmit Buffer Mode) TTTBM is ignored when TTEN=0. 0: determined by TSMODE, PTB and STB 1: set by TBPTR and TTPTR This bit can be set to 0 when only the receive timestamp function is required in TTCAN mode, and the use of PTB or STB is determined by TSMODE. Note: The TSMODE bit can only be set when the STB is empty.
18	TSFF	r	0	TTEN=0 or TTTBM=0: STB Full Flag (Transmit Secondary buffer Full Flag) 0: STB SLOT is not fully filled 1: STB SLOT is fully filled TTEN=1 and TTTBM=1: TB Full Flag (Transmit buffer Full Flag) 0: TBPTR selected transmit BUF is not fully filled 1: TBPTR selected transmit BUF is fully filled
17:16	TSSTAT	r	0	STB status (Transmission Secondary Status bits) TTEN=0 or TTEN=1 &TTTBM=0 00: STB empty 01: STB less than or equal to half full 10: STB greater than half full 11: STB full TTEN=1 and TTTBM=1 00: PTB and STB empty 01: PTB and STB not full 10: Reserved 11: PTB and STB full
15	TBSEL	rw	0	Transmit BUF Select bit (Transmit Buffer Select) 0: PTB 1: STB When TTEN=1 & TTTBM=1, TBSEL is reset to the reset value. Note: When writing the TBUF register or TSNEXT bit, this bit needs to be held at a fixed value.

Bit	Name	R/W	Reset Value	Function
14	LOM	rw	0	Silent mode enable bit (Listen Only Mode) 0: Silent mode disabled 1: silent mode enabled LOM=1 & LBME=0 forbids transmission. When LOM=1&LBME=1, answering the corresponding received frames and error frames is prohibited, but data can be sent. Note: Setting this bit is prohibited in communication.
13	STBY	rw	0	Transceiver standby mode 0 - Disable 1 - enable This register bit is connected to the output signal stby and can be used to control the standby mode of the transceiver. STBY cannot be set to 1 if TPE=1, TSONE=1 or TSALL=1. If the host sets STBY to 0, then before the host can request a new transmission, the host needs to wait for the time required for the transceiver to start up.
12	TPE	rw	0	PTB transmit enable bit (Transmit Primary Enable) 0: Disable PTB transmission 1: Enables PTB transmission When this bit is enabled, the Mailbox in the PTB will be sent at the next available location. The STB transmission already started will continue, but the next pending STB transmission will be delayed until the PTB transmission is complete. This bit will remain at 1 after a write 1 until the PTB send is complete or until the send is cancelled via TPA. Software cannot clear this bit by writing a 0. TPE is reset to a reset value by hardware in the following cases: RESET=1 BUSOFF=1 LOM=1&LBME=0 TTEN=1&TTTBM=1
11	TPA	rw	0	PTB Transmit Primary Abort bit 0: No cancellation 1: Cancels a PTB transmission that has been requested via TPE set to 1 but not yet started

Bit	Name	R/W	Reset Value	Function
				This bit is written to 1 by software but cleared by hardware. The TPE bit can be cleared by writing a 1, so it is written 1 at the same time as the TPE. the TPE is reset to the reset value by hardware in the following cases: RESET=1 BUSOFF=1 TTEN=1&TTTBM=1
10	TSONE	rw	0	Transmit Secondary ONE frame (STB data positioning) 0: Do not send 1: Send one frame of STB data In FIFO mode, the first data written is sent, in priority mode the highest priority data is sent. This bit will remain at 1 after a 1 is written until the STB is sent or until the transmission is cancelled by TSA. This bit cannot be cleared by software by writing a 0. TSONE is reset to the reset value by hardware in the following cases: RESET=1 BUFOFF=1 LOM=1&LBME=0 TTEN=1&TTTBM=1
9	TSALL	rw	0	Transmit Secondary ALL frame 0: Do not send 1: Send all data in STB This bit will remain at 1 after a 1 is written until the STB is sent or until it is cancelled by TSA. This bit cannot be cleared by software by writing a 0. TSALL is reset to a reset value by hardware in the following cases: RESET=1 BUSOFF=1 LOM=1&LBME=0 TTEN=1&TTTBM=1
8	TSA	rw	0	STB Transmit Secondary Abort bit 0: No cancellation 1: Cancels an STB transmission that has been requested via TSONE or TSALL set to 1 but not yet started This bit is written to 1 by software but cleared by hardware. Writing 1 clears the TSONE or TSALL bit. TSA is reset to the reset value by hardware in the following cases: RESET=1

Bit	Name	R/W	Reset Value	Function
				BUSOFF=1
7	RESET	rw	1	Reset request bit 0: No partial reset requested 1: Local reset is requested Some registers can only be written when RESET=1, please refer to the software reset function, when the node enters the BUS OFF state, the hardware automatically sets RESET position 1. Please note that when RESET=0 it takes 11 CAN bit times for the node to participate in communication.
6	LBME	rw	0	External loopback mode enable bit 0: Disable external loopback mode 1: Enables external loopback mode Note: Setting this bit is prohibited in communication.
5	LBMI	rw	0	Internal loopback mode enable bit 0: Disable internal loopback mode 1: Enables internal loopback mode Note: Setting this bit is prohibited in communication.
0	BUSOFF	rw	0	Bus off state 0: Bus active 1: Bus off status Note: Writing 1 clears the TECNT and RECNT registers and is generally only used for debugging purposes

33.4.8. Error Warning Register (CAN_WECR)

Address offset:0x2C

Reset value:0x0000001B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TECNT[7:0]								RECNT[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
KOER[2:0]			ALC[4:0]					AFWL[3:0]				EWL[3:0]			
r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:24	TECNT	r	0	Transmit Error Count (TEC) The transmit error counter is increased or decreased according to the error count specified by the CAN protocol.

Bit	Name	R/W	Reset Value	Function
				The counter does not overflow and 255 is the maximum value.
23:16	RECNT	r	0	Receive Error Counter (Receive Error Count) The receive error counter increases or decreases according to the error count specified by the CAN protocol. The counter does not overflow and 255 is the maximum value.
15:13	KOER	rw	0	Error Category (Kind Of Error) 000: No error 001: Bit error 010: Form error 011: Filling error 100: Response error 101: CRC error 110: Other errors 111: Reserved The KOER bit is updated when there is an error, the KOER bit remains unchanged when sending and receiving normally.
12:8	ALC	r	0	Arbitration Lost Capture (ALC) When arbitration fails ALC records the position in a frame of data at the time of the arbitration failure.
7:4	AFWL	rw	4'h01	Receive BUF will be full Warning Limit (receive buffer Almost Full Warning Limit) Set value range is 1~3. AFWL=0 is meaningless and is treated as AFWL=1.
3:0	EWL	rw	4'h0B	Error Waring Limit programmed value (Programmable Error Warning Limit) Error Waring Limit = (EWL+1)*8. This register setting affects the EIF flag.

33.4.9. Reference ID register (CAN_REFMSG)

Address offset:0x30

Reset value:0x0

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

Bit	Name	R/W	Reset Value	Function
31	REF_IDE	rw	0	IDE bit of the reference message (REFerence message IDE bit) 0: Standard format 1: Extended format
28:0	REF_ID	rw	0	ID bits of the reference message (REFerence message IDentifier) REF_IDE=0: REF_ID[10:0] is valid REF_IDE=1: REF_ID[28:0] is valid REF_ID is used to detect the reference message and is applicable for both transmit and receive. When a reference message is detected, the Sync_Mark of the current frame becomes the Ref_Mark. REF_ID[2:0] is fixed to 0 and its value is not checked so that up to 8 potential time masters can be supported. After the highest byte of the REF_MSG has been written, it is necessary to wait 6 CAN clock cycles to complete the transfer of the REF_MSG to the CAN clock domain.

33.4.10. TTCAN configuration register (CAN_TTCR)

Address offset:0x34

Reset value:0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	T_PRESC[1:0]		TTEN	TBE	TBF	TBPTR[5:0]					
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
TEW[3:0]				Res.	TTYPE[2:0]			Res.	Res.	TTPTR[5:0]					
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Name	R/W	Reset Value	Function
26:25	T_PRESC	rw	0	TTCAN counter prescaler 00b: 1 division of the bit time set in the ACBTR register 01b: 2 divisions of the bit time set by the ACBTR register 10b: 4 divisions of the bit time set in the ACBTR register 11b: 8 divisions of the bit time set by the ACBTR register The TTCAN time base is the CAN bit time defined by PRESC, AC_SEG_1 and AC_SEG_2. An additional prescaling factor of 1, 2, 4 or 8 is defined using T_PRESC.

Bit	Name	R/W	Reset Value	Function
				T_PRESC can only be modified when TTEN=0. It is also possible to modify T_PRESC and set TTEN with one write access at the same time.
24	TTEN	rw	0	TTCAN Enable (Time Trigger Enable) 0: disabled 1: TTCAN is enabled and the counter starts counting.
23	TBE	rw	0	Set TB slot to "empty" (set TB slot to "empty") 0: No operation 1: The SLOT selected by TBPTR is marked as empty When the SLOT is marked empty and TSFF=0, the TBE is automatically reset to 0. If this bit is set to 1, TBE = 1 if there is data being sent in the selected SLOT, then TBE is reset to 0 when sending is complete, sending is incorrect or sending is cancelled. TBE has a higher priority than TBF.
22	TBF	rw	0	Set TB slot to "Filled" (set TB slot to "Filled") 0: No operation 1: The SLOT selected by TBPTR is marked as filled When SLOT is marked as filled and TSFF=1, TBE is automatically reset to 0.
21:16	TBPTR	rw	0	TB SLOT pointer (Pointer to a TB message slot) 000: points to PTB 001: points to STB SLOT1 010: points to STB SLOT2 011: points to STB SLOT3 Other: invalid setting The pointed TB SLOT can be accessed read or write via TBUF and can be marked as populated or not by TBE and TBF. TBSEL and TSNEXT registers are not valid in TTCAN mode Note: write operations to this bit are only possible when TSFF = 0
15:12	TEW	rw	0	Transmit Enable Window Single Shot Transmit Trigger for the TTCAN, a window of TEW+1 cycle time can be set and transmissions are only allowed within this window.
10:8	TTYPE	rw	0	Trigger Type 000: Immediate Trigger for immediate transmission 001: Time Trigger for receive triggers

Bit	Name	R/W	Reset Value	Function
				010: Single Shot Transmit Trigger for exclusive time windows 011: Transmit Start Trigger for merged arbitrating time windows 100: Transmit Stop Trigger for merged arbitrating time windows Other: Reserved The trigger time is set via the TT_TRIG register and the TB Slot is selected via TTPTR.
5:0	TTPTR	rw	0	Transmit Trigger TB slot Pointer 000: Pointing to PTB 001: points to STB SLOT1 010: points to STB SLOT2 011: points to STB SLOT3 Other: set disable If the pointing TB SLOT is marked empty, TEIF is set when the trigger time is reached.

33.4.11. TTCAN Trigger Register (CAN_TTTR)

Address offset:0x38

Reset value:0x02020003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TT_WTRIG[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
TT_TRIG[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	TT_WTRIG	rw	16'hffff	Cycle time of the watchdog trigger When the highest byte of TT_WTRIG is operated, the TT_WTRIG value starts to be passed to the CAN clock field, so if BYTE operation, the low byte needs to be written first before the high byte.
15:0	TT_TRIG	rw	0	Trigger Time Used to specify the cycle time of the trigger, for send triggers the send SOF time is approximately the TT_TRIG setting + 1 When the highest byte of TT_WTRIG is operated, the TT_WTRIG value starts to pass to the CAN clock domain,

				so if BYTE operation, the low byte needs to be written first and then the high byte.

33.4.12. Memory Status Register (CAN_SCMS)

Address offset: 0x3C

Reset value: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	HELOC[1:0]		TXB	TXS	ACFA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
			r		r	r	rw								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FSTIM[2:0]			Res.
												rw			

Bit	Name	R/W	Reset Value	Function
28:27	HELOC	r	16'h 0	Host-side memory error location 00 - No error when accessing from the host side 01 - Error on access from host side in TBUF 10 - Error accessing from host side in RBUF 11 - Error on access from host side in ACF During the read access from the host side, HELOC will be updated with each new error. This is sufficient, because read errors during read access from the CAN side will be signalled by the ACFA, TXS and TXB, TXS and TXB are signalled. The HELOC will only be updated in the event of an error, but not in the event of a warning caused by a corrected unit error The HELOC will only be updated in the event of an error, but not in the event of a warning caused by a corrected unit error.
26	TXB	r	16'h0	Transmission block 0 - normal operation 1 - transmission blocked If MDEIF or MAEIF is set due to an error while the CAN protocol machine is reading data for transmission is set due to an error, the transmission is blocked immedi-ately. If SEIF is set, then the transmission is also blocked imme-diately.

Bit	Name	R/W	Reset Value	Function
				If TXB=1, the error counter is frozen. If RESET=1, the TXB is reset.
25	TXS	r	16’h0	Transmission stop 0 - Normal operation 1 - Transfer stopped If MDEIF or MAEIF is set due to an error when prioritising machine access to memory then any new transfers are stopped. If there is an active transfer, this will complete before stopping, but if an error occurs during this transfer, no re-transfer will be started. If RESET=1, the TXS is reset.
24	ACFA	rw	16’h0	Filter Enable 0 - ACF normal operation 1 - ACF disabled: all received frames are accepted If MDEIF or MAEIF is set due to an address range error in ACF, then set ACFA. Accept filtering is then disabled and all frames are accepted. ACFA can be reset by writing 1 to it like an interrupt flag. However, since ACFA will be set while reception is still active, it needs to be reset after reception is complete, e.g. If RIF is set. If RESET=1, ACFA will also be reset.
3:1	FSTIM	rw	16’h0	Fault excitation Fault excitation is only possible when XMREN = 1.

33.4.13. Filter Group Control Register (CAN_ACFCR)

Address offset:0x44

Reset value:0x00010000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	AE_11	AE_10	AE_9	AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0
r	r	r	r	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.	ACFADR			
r	r	r	r	r	r	r	r	r	r	r	r	rw			

Bit	Name	R/W	Reset Value	Function
31:16	AE_x	rw	16'h01	ACF enable 1 - Enabled 0 - Disable Each acceptance filter (ACFC / ACFM) can be individually enabled or disabled. After a hardware reset, only filter number 0 is enabled by default.
3:0	ACFADR	rw	16'h00	Filter Address ACFADR points to a specific acceptance filter. The registers ACFC and ACFM can be used to access the the selected filter. The value of ACFADR>ACF_NUMBER-1 is nonsensical and is automatically treated as the value ACF_NUMBER-1

33.4.14. Filter group code register (CAN_ACFC)

Address offset:0x48

Reset value:0xXXXXXXXX

0x00	CAN_ID
0x04	CAN_FORMAT
0x08	CAN_TYPE

Note: Please refer to 33.3.7 LLC Frame Format Definition for the control meaning of the above registers

33.4.15. Filter groupmask register (CAN_ACFM)

Address offset:0x58

Reset value:0xXXXXXXXX

0x00	CAN_ID
0x04	CAN_FORMAT
0x08	CAN_TYPE

Note: Please refer to 33.3.7 LLC Frame Format Definition for the control meaning of the above registers

33.4.16. CAN Receive BUF register (CAN_RBUF)

Address offset:0x70

Reset value:0xXXXXXXXX

Note: Please refer to 33.3.7 LLC Frame Format Definition for the control meaning of the above registers

33.4.17. CAN transmit BUF register (CAN_TBUF)

Address offset:0x94

Reset value:0xXXXXXXXX

Note: Please refer to 33.3.7 LLC Frame Format Definition for the control meaning of the above registers

33.4.18. CAN register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	[illegible]
0x00	CAN_TSNCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ROP	CES	Version [15:0]															
	Reset value	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1																
0x04	CAN_ AC-	Res.	AC_SJW[6:0]							Res.	AC_SEG_2[6:0]							Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	[illegible]
	Reset value	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0x10	CAN_RLSSP	Res.	RETLIM[2:0]			Res.	RE-ALIM[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[19:16]			
	Reset value	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	CAN_IFR	EWARN	EPASS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WTIF	TEIF	TTIF	EPIF	ALIF	BEIF	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	[illegible]
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	CAN_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WTIE	Res.	TTIE	EPIE	ALIE	BEIE	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	[illegible]
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	1	1	1	1	1	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1C	CAN_TSR	Res.	Res.	Res.	Res.	Res.	TSTAT_H[2:0]			HANDLE_H[7:0]								Res.	Res.	Res.	Res.	Res.	TSTAT_L[2:0]			HANDLE_L[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	CAN_MCR	SACK	ROM	ROV	RREL	RBALL	Res.	RSTAT[1:0]		Res.	TSNEXT	TSMODE	TTTBM	Res.	TSFF	TSSTAT[1:0]		TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA	RESET	LBME	LBMI	Res.	Res.	Res.	Res.	BUSOFF
	Reset value	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0x2C	CAN_WECR	TECNT[7:0]								RECNT[7:0]								KOER[2:0]			ALC[4:0]					AFWL[3:0]				EWL[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1
0x30	CAN_REFMSG	REF_IDE	Res.	Res.	REF_ID[28:0]																												
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	CAN_TTCR	Res.	Res.	Res.	Res.	Res.	T_PRESC[1:0]		TTEN	TBE	TBF	TBPTR[5:0]						TEW[3:0]				Res.	TTYPE[2:0]			Res.	Res.	TTPTR[5:0]					
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	CAN_TTTR	TT_WTRIG[15:0]																TT_TRIG[15:0]															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	CANFD SCMS				HELOC[1:0]		TXB	TXS	ACFA																					FSTIM[2:0]			
	Reset value				0		0	0	0																					0			
0x44	CAN_ACFCR	Res.	Res.	Res.	Res.	AE_11	AE_10	AE_9	AE_8	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ACFADR			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34. USB Full Speed Device Interface (USBD)

34.1. Introduction

The USBD peripheral implements the interface between the USB2.0 full speed bus and the APB1 bus.

The USBD peripheral supports USB suspend/resume operations and can stop the device clock for low power consumption.

34.2. Key USB features

- Conforms to the USB2.0 full-speed device specification
- Configurable with 1 to 6 USB endpoints (endpoint 0 ~ endpoint 5)
- Dedicated 1024 byte packet cache storage
- CRC (cyclic redundancy check) generation/checking, reverse non-zero (NRZI) encoding/decoding and bit padding
- Support for controlled/synchronous/batch/interrupt transfers
- Support for double buffer mechanism for bulk/synchronous endpoints
- Support for USB suspend/resume operations
- Frame lock clock pulse generation

34.3. USB device block diagram

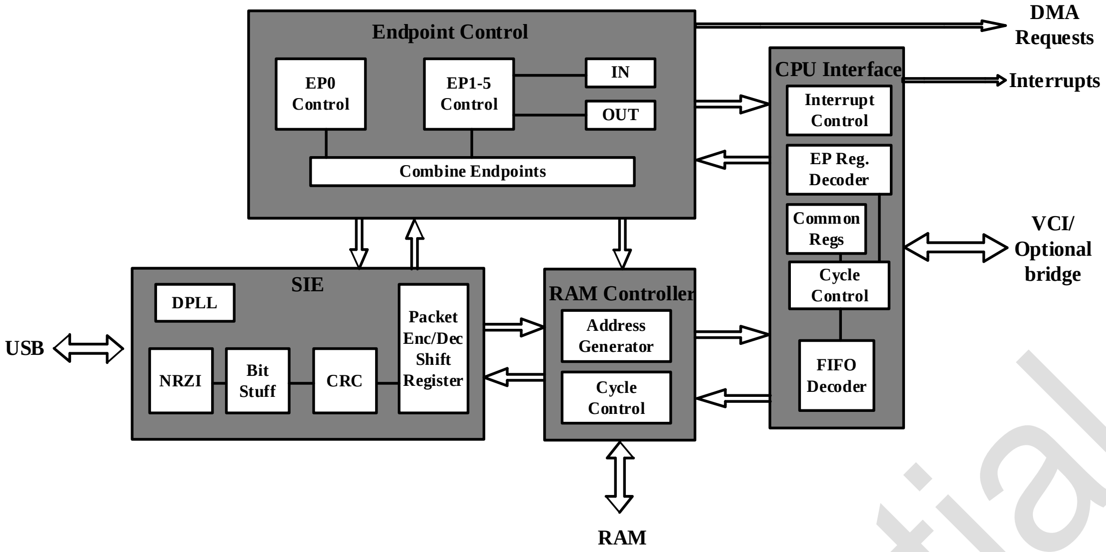


Figure 34-1 USB device block diagram

34.4. Functional description

The USB module provides a USB specification compliant communication link between the PC host and the functions implemented by the microcontroller. data transfer between the PC host and the microcontroller is done via a dedicated data buffer that can be accessed directly by the USB peripheral. The size of this dedicated data buffer is fixed, with endpoint 0 using a maximum 64 byte buffer, EP1 using a maximum 512 byte buffer, EP2, EP3 and EP4 using a 128 byte buffer and EP5 using a 64 byte buffer, with the IN and OUT buffers of the same endpoint being shared. The USB module communicates with the PC host, detecting token packets, processing data sent/received and handling handshake packets according to the USB specification.

When a valid token group is recognised by the USB module, the associated data transfer takes place (if data transfer is required and the endpoints are configured) and the USB module implements the exchange of data between the port and the dedicated buffer via an internal register. After all data transfers have been completed, the appropriate handshake packet is sent or received if required, depending on the direction of the transfer.

At the end of the data transfer, the USB module triggers an interrupt associated with the endpoint and by reading the status register or using the different interrupt handlers, the microcontroller can determine:

- which endpoint needs to be serviced
- which type of transfer was in progress when errors such as bit fill, format, CRC, protocol, missing ACK, buffer overflow/buffer not full, etc. were generated.

The USB module can be put in low power mode (SUSPEND mode) by writing the control registers at any time when the USB module is not needed. In this mode, the USB internal data clock is stopped. The USB module can be woken up in low-power mode by detecting the data transfer on the USB line.

The USB module uses a fixed clock, which is defined by the USB standard as 48MHz.

34.4.1. Description of the functional module

The USB module implements all the features of the standard USB interface and is composed of the following parts:

- Serial Interface Engine (SIE):The SIE handles NRZI encoding/decoding, bit padding/unpadding and CRC generation/checking. It generates a 12MHz USB clock from the input 48MHz clock and the data stream is synchronised under this clock when data is received from the USB host. It generates headers for the transmitted packets and decodes the packets as they are received.
- Endpoint Controller:Uses two state controllers, one for control transfers on endpoint 0 and the other for bulk, interrupt or synchronous transfer transactions on endpoints 1-5.
- CPU Interface:The CPU interface allows access to the control/status registers and FIFOs of each endpoint. It generates an interrupt to the CPU when a packet is successfully transmitted or received, or when the host enters or recovers from SUSPEND mode.
- RAM controller:The RAM controller provides an interface to a single block of synchronous single-port RAM that is used to cache packets between the CPU and the USB. It takes FIFO pointers from the endpoint controller and converts them to address pointers within the RAM block and generates RAM access control signals.

34.4.2. System reset and power-on reset

When a system reset or power-on reset occurs, the first thing the application needs to do is to provide the clock signal required by the USB module and then clear the reset signal to allow the program to access the registers of the USB module. The initialisation process following a reset is described below:

Firstly, the clock of the register unit is activated by the application;

secondly, the relevant control bits of the device clock management logic unit are then configured;

Finally, the reset signal is cleared.

When the system is reset, the application should initialise all the registers required to enable the USB module to generate normal interrupts and complete data transfers. All registers not related to endpoints need to be initialised according to the needs of the application (e.g. selection of interrupt enable, etc.). Next, the USB reset state is entered.

34.4.3. USB reset (RESET interrupt)

When a USB reset occurs, the USB module enters the state described below:

- position 0 of the ADD[6:0] of the USB_CR register;
- INDEX position 0 of the USB_FRAME register;
- clearing all data from the FIFOs;
- clearing all control/status registers;
- enable all interrupts except suspend;
- Generate a USB reset interrupt.

When the software receives a reset interrupt, it should close all open pipes and wait for the bus enumeration to start

34.4.4. USB suspend/wake mode

The USB module will enter suspend mode when the USB device is idle for all 3ms and the enable suspend bit in the USB_CR register has been set to 1. If the suspend interrupt is enabled (EN_Suspend bit in the USB_INTRE register) then a suspend interrupt will be generated at this point.

When the USB enters suspend mode, the internal 12MHz data clock will be stopped to reduce power consumption. At the same time, the incoming 48MHz system clock needs to be maintained so that the USB module can detect signals on the bus. It is of course possible to stop the system clock to further reduce power consumption, but if the system clock is stopped the USB module will not be able to detect signals on the bus and the user will have to restart the system clock.

When the USB module is in suspend mode, it can wake up the device when it detects the K status on the bus, thus exiting suspend mode and generating a resume interrupt. It is of course possible to force the device out of suspend mode by setting the Resume bit in the USB control register by the CPU, in which case the CPU should clear the bit after 10 ms (max. 15 ms). At this point the USB module is completely out of suspend mode and no resume interrupts are generated.

34.4.5. IN grouping (for data transmission)

The FIFO sizes for IN endpoints 1-5 are 512bytes, 128bytes, 128bytes, 128bytes and 64bytes respectively, but the maximum packet size for IN transmission is configurable and is determined by the InMaxP register written to each endpoint.

As each packet to be sent is loaded into the IN FIFO, the InPktRdy bit in the USB_INEPCSR register needs to be set. If the AutoSet bit in the USB_INEPCSR register is set, the InPktRdy bit is automatically set when the maximum packet is loaded in the FIFO. For packets smaller than the maximum value, InPktRdy is to be set manually (by the CPU).

When the InPktRdy bit is set, the FIFONotEmpty bit in the USB_INEPCSR register is also set, and the packet is then ready to be sent.

When the packet is successfully sent, the InPktRdy bit and the FIFONotEmpty bit are cleared and if the corresponding interrupt is enabled, then an interrupt is generated for the corresponding IN endpoint. The next packet can then be loaded into the FIFO.

34.4.5.1. IN packet double packet caching function

If the FIFO size of the IN endpoint is at least twice the maximum packet size of this endpoint (depending on the size of InMaxP), then both packets can be cached in the FIFO.

When the first packet is loaded into the FIFO, InPktRdy is set and then immediately cleared, generating the corresponding IN endpoint interrupt. The second packet can then be loaded into the FIFO and InPktRdy will be set again (manually or automatically) so that both packets can be sent.

When the first packet is successfully sent, the clearing of the InPktRdy bit and the generation of the corresponding IN endpoint interrupt indicates that the other packet can now be loaded into the FIFO. The status of the FIFONotEmpty bit at this point indicates how many packets may have been loaded. If the FIFONotEmpty bit is set, then there will be another packet in the FIFO and only one packet will be loaded. If the FIFONotEmpty bit is reset, then there are no packets in the FIFO and more than two packets may be loaded.

34.4.6. OUT packets (for data reception)

The FIFO sizes for OUT endpoints 1-5 are 512bytes, 128bytes, 128bytes, 128bytes and 64bytes respectively, but the maximum packet size transmitted by OUT is programmable and is determined by the OutMaxP register written to each endpoint.

When a packet is received and placed in the OUT FIFO, the OutPktRdy bit and the FIFOFull bit in USB_OUTEPCSR are set and the appropriate OUT endpoint interrupt is generated to indicate that a packet is now ready to be read from the FIFO. After the packet has been read, the OutPktRdy bit needs to be cleared in order to receive more packets. If the AutoClear bit in USB_OUTEPCSR is set and the largest packet is read away from the FIFO, the OutPktRdy bit is automatically cleared. the FIFOFull bit is also cleared. For packets smaller than the maximum packet size, OutPktRdy must be cleared manually (i.e. by the CPU).

34.4.6.1. OUT packet double packet cache function

The OUT FIFO can cache two packets if the size of the OUT endpoint FIFO is at least twice the maximum packet size for that endpoint (set in the OutMaxP register).

When the first packet to be received is loaded into the OUT FIFO, the OutPktRdy bit in USB_OUTEPxCSR is set and the corresponding OUT endpoint interrupt is generated to indicate that the packet is now ready to be read from the FIFO.

Note: The FIFOFull bit in USB_OUTEPCSR is not set at this point, it is only set when the second packet is received and loaded into the OUT FIFO.

After reading away the first packet, the OutPktRdy bit needs to be cleared in order to receive more packets. If the AutoClear bit in USB_OUTEPCSR is set and the maximum size packet is read away from the FIFO, the OutPktRdy bit will be automatically cleared. For packets smaller than the maximum packet size, OutPktRdy must be cleared manually (i.e. by the CPU).

If the FIFOFull bit is set to 1 when OutPktRdy is cleared, the FIFOFull bit will be cleared first. OutPktRdy is then set again to indicate that there is another packet in the FIFO waiting to be read away.

34.4.7. Controlling the transfer

Endpoint 0 is the main control endpoint of the USB. As such, the routines required to drive endpoint 0 are more complex than those required to drive the other endpoints.

Software can receive and process all standard device requests through endpoint 0. These are described in the USB Bus Specification. The protocol for these device requests involves a different number and type of transactions per transfer. To accommodate this, the CPU needs to use a state machine approach to decode and process the commands. Standard device requests can be divided into three categories: zero data requests, write requests and read requests. This section describes the sequence of events that must be executed by the software to process the different types of device requests.

Note: The Setup package associated with any standard device request should include an 8-byte command. Any Setup package that contains more than 8 bytes of command characters will be automatically rejected by the USB host.

34.4.7.1. Zero Data Requests

All information for a Zero Data request is contained in the 8 byte command and no additional data needs to be transferred. Examples of standard device requests for zero data are:SET_FEATURE, CLEAR_FEATURE, SET_ADDRESS, SET_CONFIGURATION, SET_INTERFACE.

As with all requests, the event sequence will start when the software receives an endpoint 0 interrupt. the OutPktRdy bit (in the USB_EP0CSR register) will also be set. the 8 byte command should be read from the endpoint 0 FIFO, decoded and the appropriate action taken. For example, if the command is SET_ADDRESS, the 7-bit address value contained in the command should be written to the FAddr register.

The USB_EP0CSR register should then be written to set the ServicedOutPktRdy bit (indicating that the command has been read from the FIFO) and to set the DataEnd bit (indicating that no further data is required for the request).

When the host moves to the status stage of the request, a second endpoint 0 interrupt is generated to indicate that the request has completed. No further action is required by the software; the second interrupt simply acknowledges the successful completion of the request.

If the command is an unrecognisable command, or cannot be executed for some other reason, then when it has been decoded the USB_EP0CSR register should be written to set the ServicedOutPktRdy bit and set the SendStall bit. When the host moves to the status stage of the request, the device will send a STALL to tell the host that the request is not being executed. A second endpoint 0 interrupt will be generated and the SentStall bit will be set.

If the host sends more data after setting the DataEnd bit, then the device will send a STALL. an endpoint 0 interrupt will be generated and the SentStall bit will be set.

34.4.7.2. Write request

A write request consists of one (or more) additional packets that are sent from the host after the 8-byte command. An example of a write standard device request is :SET_DESCRIPTOR.

As with all requests, the event sequence will start when the software receives an endpoint 0 interrupt. the OutPktRdy bit (in the USB_EP0CSR register) will also be set. the 8-byte command should be read and decoded from the endpoint 0 FIFO.

As with the zero data request, the USB_EP0CSR register should then be written to set the ServicedOutPktRdy bit (indicating that the command has been read from the FIFO), but in this case the DataEnd bit should not be set (indicating that more data is required).

When the second endpoint 0 interrupt is received, the USB_EP0CSR register should be read to check the endpoint status. the OutPktRdy bit should be set to indicate that a packet has been received. The COUNT0 register should then be read to determine the size of this packet. Packets can be read from the endpoint 0 FIFO.

If the length of the data associated with the request (indicated by the wLength field in the command) is greater than the maximum packet size for endpoint 0, further packets will be sent. In this case, USB_EP0CSR should be written to set the ServicedOutPktRdy bit, but the DataEnd bit should not be set.

The ServicedOutPktRdy bit and the DataEnd bit in the USB_EP0CSR register should be set when all expected packets have been received (indicating that no more data is required).

When the host moves to the status stage of the request, another endpoint 0 interrupt will be generated to indicate that the request has been completed. No further action is required by the software; the interrupt is simply an acknowledgement that the request has completed successfully.

If the command is an unrecognisable command, or cannot be executed for some other reason, the ServicedOutPktRdy bit and the SendStall bit in the USB_EP0CSR register should be set when it has been decoded. When the host sends more data, the USB device will send a STALL to tell the host that the request was not executed. An endpoint 0 interrupt will be generated and the SendStall bit will be set.

If the host sends more data after setting DataEnd, then the USB device will send a STALL. an endpoint 0 interrupt will be generated and the SentStall bit will be set up.

34.4.7.3. Read request

After the 8 byte command, the read request has one (or more) packets sent from the function to the host. Examples of standard device requests are: GET_CONFIGURATION, GET_INTERFACE, GET_DESCRIPTOR, GET_STATUS, SYNCH_FRAME.

As with all requests, the event sequence will start when the software receives an endpoint 0 interrupt. the OutPktRdy bit (in the USB_EP0CSR register) will also be set. the 8 byte command should be read from the endpoint 0 FIFO and decoded. The ServicedOutPktRdy bit in the USB_EP0CSR register should then be set (indicating that the command has been read from the FIFO).

The data sent to the host should be written to the FIFO at endpoint 0. If the data to be sent is larger than the maximum packet size at endpoint 0, only the maximum packet size should be written to the FIFO. The InPktRdy bit in register USB_EP0CSR should then be set (indicating that there is a packet in the FIFO to be sent). When the packet is sent to the host, another endpoint 0 interrupt will be generated and the next packet can be written to the FIFO.

When the last packet is written to the FIFO, the InPktRdy bit and the DataEnd bit in the USB_EP0CSR register should be set (indicating that there is no more data after this packet).

When the host moves to the status phase of the request, another endpoint 0 interrupt will be generated to indicate that the request has been completed. No further action is required by the software; the interrupt is simply an acknowledgement of the successful completion of the request.

If the command is an unrecognised command, or cannot be executed for some other reason, the ServicedOutPktRdy bit and SendStall bit in the USB_EP0CSR register should be set when it has been decoded. When the host requests data, the USB device will send a STALL to tell the host that the request is not being executed. An endpoint 0 interrupt will then be generated and the SendStall bit will be set.

If the host requests more data after DataEnd has been set, then the USB device will send a STALL. an endpoint 0 interrupt will be generated and the SentStall bit will be set.

34.4.7.4. Error handling

A control transfer may be aborted due to a protocol error on the USB, an early termination of the transfer by the host, or if the software function controller wishes to terminate the transfer (e.g. because it cannot process the command). the USB will automatically detect a protocol error and send a STALL to the host in the following cases.

- The host sends more data than specified in the command during the OUT data phase of the write request. This condition will be detected when the host sends an OUT token with the DataEnd bit set.
- The host requests more data than specified in the command during the IN data phase of a read request. This condition is detected when the host sends an IN token after the DataEnd bit in the USB_EP0CSR register has been set.
- The host sends an OUT packet with more than MaxP data bytes.
- The host sends a non-zero length DATA1 packet during the STATUS phase of the read request.

  When the USB device sends a STALL packet, it sets the SentStall bit and generates an interrupt. When the software receives an endpoint 0 interrupt with the SentStall bit set, it should abort the current transmission, clear the SentStall bit and return to the initial state.

If the host enters the state phase before transmitting all of the requested data, or if it sends a new SETUP packet before completing the current transmission, thus ending the transmission early, then the SetupEnd bit will be set and an Endpoint 0 interrupt will be generated. When software receives an endpoint 0 interrupt with the SetupEnd bit set, it should abort the current transmission, set the ServicedSetupEnd bit, and return to the initial state. If the OutPktRdy bit is set, this indicates that the host has sent another SEPUP packet and the software should then process this command.

If the software wishes to terminate the current transfer because it cannot process the command or has other internal errors, then it should set the SendStall bit. The USB device will then send a STALL packet to the host, setting the SendStall bit and generating an endpoint 0 interrupt.

34.4.8. Synchronous transmission

The USB standard defines a method of transmission that requires a fixed and precise data rate to be maintained at full speed: synchronous transmission. Synchronous transmission is generally used for the transmission of audio streams, compressed video streams and other data that has strict data rate requirements. If an endpoint is defined as a "synchronous endpoint" during enumeration, the USB host allocates a fixed amount of bandwidth to each frame and ensures that each frame transmits exactly one IN or OUT packet (the packet type is determined by the direction of the endpoint transmission). In order to meet the bandwidth requirements, there are no retransmissions with errors; this means that no handshake protocols, i.e. no ACK packets, are sent after the data packets are sent or received. Similarly, only packets with a PID (packet ID) of DATA0 are transmitted by synchronous transmission, and no data flipping mechanism is used.

34.4.8.1. Synchronous read transfer

The synchronous read transfer is used to transfer cyclic data from the USB module to the host. Three optional features can be used to synchronise the IN endpoints.

- Dual packet cache feature

Double packet buffering is automatically enabled when the value written to the InMaxP register is less than or equal to half the size of the FIFO allocated to the endpoint. When enabled, up to two packets can be stored in the FIFO waiting to be transferred to the host.

- DMA Functionality

If the endpoint has DMA enabled, a DMA request will be generated whenever the endpoint is able to receive another packet in its FIFO. This feature can be used to allow an external DMA controller to load packets into the FIFO without processor intervention. However, this feature is not particularly useful for synchronising endpoints, as the packets transferred are usually not the maximum packet size and the USB_INEPCSR register needs to be accessed after each packet to check for Underrun errors.

- AutoSet feature

When the AutoSet feature is enabled, the InPktRdy bit will be automatically set when an InMaxP byte packet is loaded into the FIFO. However, this feature is not particularly useful for synchronous endpoints, as the packets transferred are usually not of maximum packet size and require access to the USB_INEPCSR register after each packet to check for Underrun errors.

Before a synchronous IN endpoint can be used, the InMaxP register must be written with the maximum packet size (in bytes) for that endpoint. This value should be the same as the wMaxPacketSize field of the endpoint's standard endpoint descriptor. In addition, the relevant interrupt enable bit in the USB_INTRE register should be set to 1 and some of the bits in the USB_INEPCSR register should be set as follows.

Table 34-1 USB_INTRE register

AutoSet	ISO	Mode	DMAEnab	FrcDataTog
0/1	1	1	0/1	0

A synchronous endpoint does not support data retransmission, so if data errors are to be avoided then the data sent to the host must be loaded into the FIFO before the IN token is received. The host will send one IN token per frame, but the time within the frame is variable. If an IN token is received at the end of a frame and then at the beginning of the next frame, there will be little time to reload the FIFO. for this reason, a double cache is usually required in the synchronous IN endpoint.

The AutoSet function can be used to synchronise IN endpoints, but unless the data from the source arrives at an absolutely consistent rate and is synchronised with the host's frame clock, the size of the packet sent to the host will have to be increased or decreased frame by frame to match the source data rate. This means that the actual packet size is not always the InMaxP size, which renders the AutoSet function useless.

An interrupt is generated when a packet is sent to the host, which can be used by software to load the next packet into the FIFO and set the InPktRdy bit in the USB_INEPCSR register. As the interrupt can occur at almost any time within a frame, depending on when the host schedules the routine, this can lead to irregular timing of FIFO load requests. If the endpoint's data source is from some external hardware, it may be more convenient to wait for the end of each frame before loading the FIFO, as this will minimise the need for additional buffering. The above operation can be achieved by using a SOF interrupt or an external SOF_PULSE signal from the USB device to trigger the loading of the next packet. When a SOF packet is received, a SOF_PULSE signal is generated per frame (the USB also keeps an external frame counter, so it can still generate SOF_PULSE when a SOF packet is lost). This SOF_PULSE interrupt can still be used to set the InPktRdy bit in USB_INEPCSR and to check for data overflow/misses.

The synchronous IN pipeline with dual caching enabled can be a source of problems. Dual caching requires that packets are only transmitted after the frame has been loaded. If the function loads the first packet at least one frame before the host builds the pipeline (and then starts sending the IN token), there is no problem. However, if the host already starts sending the IN token when the first packet is loaded, the packet may be transmitted in the same frame as it is loaded, depending on whether it is loaded before or after the IN token is received. This potential problem can be avoided by setting the ISO Update bit in the USB_CR register. When this bit is set to 1, any packets loaded into the synchronous IN endpoint FIFO will not be sent until the next SOF packet is received, thus ensuring that packets are not sent prematurely.

If the endpoint has no data in the FIFO when the IN token is received, it will send an empty packet to the host and set the UnderRun bit in the USB_INEPxCSR register. This indicates that the software is not providing the host with data fast enough. It is up to the application to decide how to handle this error condition.

If the software finds that the InPktRdy bit in the USB_INEPxCSR register is set when it wants to load the next packet each frame, this indicates that the packet has not been sent yet (perhaps because an IN token packet from the host is corrupted). It depends on how the application handles this situation: it can choose to flush the unsent packet by setting the FlushFIFO bit in the USB_INEPxCSR register, or it can choose to skip the current packet.

34.4.8.2. Synchronous write transfer (ISOCHRONOUS OUT)

The synchronous OUT endpoint is used to transmit periodic data from the function controller to the host. Three optional features are available for the synchronous OUT endpoint.

1. Double packet cache function

Double packet buffering is automatically enabled when the value written to the OutMaxP register is less than or equal to half the size of the FIFO allocated to the endpoint. When enabled, up to two packets can be stored in the FIFO.

2. DMA function

If DMA is enabled for an endpoint, a DMA request will be generated whenever there is a packet in the endpoint's FIFO. This feature can be used to allow an external DMA controller to read packets from the FIFO without the need for processor intervention. However, this feature is not particularly useful for synchronous endpoints as the packets transferred are usually not the maximum packet size and the USB_OUTEPCSR register needs to be accessed after each packet to check for over-flow or CRC errors.

3. AutoClear function

When the AutoClear feature is enabled, the OutPktRdy bit is automatically cleared when packets of OutMaxP bytes are read from the FIFO. However, this feature is not particularly useful for synchro-nous endpoints, as the packets transferred are usually not of maximum packet size and the USB_OUTEPCSR register needs to be accessed after each packet to check for overflow or CRC errors.

The maximum packet size (in bytes) of the endpoint must be used to write to the OutMaxP register before the synchronous OUT endpoint can be used. This value should be the same as the wMax-PacketSize field of the endpoint's standard endpoint descriptor. In addition, the relevant interrupt en-able bit in the USB_INTRE register should be set to 1 and some of the bits in the USB_OUTEPxCSR register should be set as shown below.

Table 34-2 USB_OUTEPxCSR register

AutoClear	ISO	DMAEnab
0/1	1	0/1

A synchronous endpoint does not support data retransmission, so there must be room in the FIFO to receive packets if data overflow is to be avoided. The host will send one packet per frame, but the time within the frame can vary. If a packet is received at the end of one frame and another arrives at the beginning of the next frame, there is little time to read the data in the FIFO. Therefore, for synchronous OUT endpoints, a double packet caching feature is usually required.

The AutoClear feature can be used with synchronous OUT endpoints. However, unless the data receiver receives data at an absolutely consistent rate and is synchronised with the host's frame clock, the packet size sent by the host will have to increase or decrease frame by frame to match the required data rate. This means that the actual packet size is not always the OutMaxP size, which renders the AutoClear function useless.

When a packet is received from the host, an interrupt is generated which can be used by software to read away the packet from the FIFO and clear the OutPktRdy bit in the USB_OUTEPCSR register. As the interrupt may occur at almost any time within a frame, depending on when the host has scheduled the transaction, the timing of FIFO read data requests may be irregular. If the endpoint's data receiver is to transmit to some external hardware, it may be desirable to wait for the end of each frame before reading the FIFO, thus reducing the need for additional buffering. This can be done by using a SOF interrupt or an external SOF_PULSE signal from the USB device to trigger a packet read. The SOF_PULSE signal is generated once per frame when a SOF packet is received (the USB device also retains an external frame counter so that it can still generate SOF_PULSE when a SOF packet is lost). This interrupt can still be used to clear the OutPktRdy bit in USB_OUTEPxCSR and to check for data overflow/missing.

If there is no space in the FIFO to store packets received from the host, the overflow bit in the USB_OUTEPxCSR register will be set. This indicates that the software is not reading the data fast enough. It is up to the application to decide how to handle this error condition.

If the USB device finds a received packet with a CRC error, it still stores the packet in the FIFO and sets the OutPktRdy bit and the DataError bit. It is up to the application to decide how to handle this error condition.

34.4.9. Batch transfers

34.4.9.1. Bulk read transfer (BULK IN)

The Bulk IN endpoint is used to transfer irregular data from the function controller to the host. There are three optional features available for the Bulk IN endpoint.

- Dual packet cache feature

If the value written to the InMaxP register is less than or equal to half the size of the FIFO allocated to the endpoint, the double packet caching feature will be automatically enabled. When enabled, up to two packets can be stored in the FIFO for transmission to the host.

- DMA Functionality

If DMA is enabled for an endpoint, a DMA request will be generated whenever the endpoint is able to accept another packet in its FIFO. This feature can be used to allow an external DMA controller to load packets into the FIFO without processor intervention.

- AutoSet feature

When the AutoSet feature is enabled, the InPktRdy bit will be automatically set when an InMaxP byte packet is loaded into the FIFO. This is particularly useful when loading the FIFO using DMA, as it avoids the need for any processor intervention when loading individual packets during high volume transfers.

Before a Bulk IN endpoint can be used, the maximum packet size (in bytes) for that endpoint must be used to write to the InMaxP register. This value should be the same as the wMaxPacketSize field of the endpoint's standard endpoint descriptor. In addition, the relevant interrupt enable bit in the USB_INTRE register should be set to 1 and the USB_INEPxCSR register should be set as shown below.

Table 34-3 USB_INEPxCSR register

AutoSet	ISO	Mode	DMAEnab	FrcDataTog
0/1	0	1	0/1	0

When first configuring a Bulk IN endpoint, the USB_INEPCSR register should be written to set the ClrDataTog bit after the SET_CONFIGURATION or SET_INTERFACE command is executed on endpoint 0. This will ensure that the data switch is initiated in the correct state. In addition, if there are any packets in the FIFO (indicated by the set FIFONotEmpty bit) they should be flushed by setting the FlushFIFO bit.

Note:If double buffering is enabled, it may be necessary to set this bit twice in succession.

When data is to be transferred through the Bulk IN pipe, a packet needs to be loaded into the FIFO and written to the USB_INEPxCSR register to set the InPktRdy bit. When the packet has been sent, the InPktRdy bit is cleared and an interrupt is generated so that the next packet can be loaded into the FIFO. If dual packet caching is enabled, then as soon as the first packet is loaded and the InPktRdy bit is set, the InPktRdy bit will be cleared and an interrupt will be generated so that the second packet can be loaded into the FIFO. The software should operate in the same way, loading a packet when it receives an interrupt, whether or not the dual packet cache is enabled.

The packet size should not exceed the size specified in the InMaxP register. When a block of data larger than InMaxP is to be transmitted, it must be sent as multiple packets. The size of these packets should be InMaxP, except for the last packet. The host can determine that all data transmitted has been sent by knowing the total amount of data expected. Alternatively, when it receives a packet smaller than the size of InMaxP, it can infer that all the data has been sent. In the latter case, if the total size of the data block is a multiple of InMaxP, then the function needs to send an empty packet after all the data has been sent. This is done by setting InPktRdy when the next interrupt is received and not loading any data into the FIFO.

If a large amount of data is being transferred then by using DMA it is possible to avoid calling the interrupt service program to load each packet.

If the software wants to close the Bulk IN pipeline, it should set the SendStall bit. When the USB device receives the next IN token it will send a STALL to the host, setting the SendStall bit and generating an interrupt.

When the software receives an interrupt with the SentStall bit set, it should clear this SentStall bit. Of course, it should also leave the SendStall bit set until it is ready to re-enable the Bulk IN pipeline.

Note:If for some reason the host cannot receive the STALL packet, it will send another IN token, so it is recommended that the SendStall bit be left set until the software is ready to re-enable the Bulk IN pipeline. When the pipeline is re-enabled, the data switching sequence should be restarted by setting the ClrDataTog bit in the USB_INEPCSR register.

34.4.9.2. Bulk write transfer (BULK OUT)

The Bulk OUT endpoint is used to transfer unscheduled data from the host to the USB module. There are three optional features available for the Bulk OUT endpoint.

1) Double packet cache function

If the value written to the OutMaxP register is less than or equal to half the size of the FIFO allocated to the endpoint, double packet buffering will be automatically enabled. When enabled, up to two packets can be stored in the FIFO.

2) DMA Function

If DMA is enabled for an endpoint, a DMA request will be generated whenever there is a packet in the endpoint's FIFO. This feature can be used to allow an external DMA controller to read packets from the FIFO without the need for processor intervention

3) AutoSet feature

When the AutoClear feature is enabled, the OutPktRdy bit is automatically cleared when packets of OutMaxP bytes are unloaded from the FIFO. This is particularly useful when reading data from the FIFO using DMA. This is because it avoids the processor intervention required when reading individual packets during large Bulk transfers.

Before using the Bulk OUT endpoint, the maximum packet size (in bytes) of the endpoint must be used to write to the OutMaxP register. This value should be the same as the wMaxPacketSize field of the endpoint's standard endpoint descriptor. In addition, the relevant interrupt enable bit in the USB_INTRE register should be set to 1 (if this endpoint requires an interrupt) and the relevant bit in the USB_OUTEPxCSR register should be set as shown below.

Table 34-4 USB_OUTEPxCSR register

AutoClear	ISO	DMAEnab
0/1	0	0/1

When first configuring a Bulk OUT endpoint, the USB_OUTEPCSR should be written to set the ClrDataTog bit after the SET_CONFIGURATION or SET_INTERFACE command on endpoint 0. This will ensure that the data switch is initiated in the correct state. Similarly, if there are any packets in the FIFO (OutPktRdy bit is set) they should be flushed by setting the FlushFIFO bit.

Note:If double buffering is enabled, it may be necessary to set this bit twice in succession.

When a packet is received by a Bulk OUT endpoint, the OutPktRdy bit is set and an interrupt is generated. The software should read the OutCount register of the endpoint to determine the size of the packet. The packet should be read from the FIFO first and then the OutPktRdy bit should be cleared.

The packet size should not exceed the size specified in the OutMaxP register (as this should be the value set in the wMaxPacketSize field of the endpoint descriptor sent to the host). When a block of data larger than OutMaxP is to be sent to a function, it will be sent as multiple packets. The size of all packets is OutMaxP, except for the last packet, which will contain the remainder. Software can use application-specific methods to determine the total size of the block to determine when the last packet is received. Alternatively, when it receives a packet smaller than the size of OutMaxP, it may infer that the entire data has been received (if the total size of the data block is a multiple of OutMaxP, an empty packet will be sent at the end of the data, indicating that the transmission is complete).

If a large amount of data is being transferred, by using DMA it is possible to avoid calling the interrupt service program to read the data for each packet.

If the software wants to close the Bulk OUT pipeline, it should set the SendStall bit. When the USB device receives the next packet it will send a STALL to the host, setting the SendStall bit and generating an interrupt.

When the software receives an interrupt with the SentStall bit set, it should clear this SentStall bit. It should also leave the SendStall bit set until it is ready to re-enable the Bulk OUT pipeline.

Note:If for some reason the host fails to receive a STALL packet, it will send another packet, so it is recommended that the SendStall bit is retained until the software is ready to re-enable the Bulk OUT pipeline. When re-enabling the Bulk OUT pipeline, the data switching sequence should be restarted by setting the ClrDataTog bit in the USB_OUTEPxCSR register.

34.4.10. Interrupting a transfer

34.4.10.1. Interrupt Read Transfer (INTERRUPT IN)

The INTERRUPT IN endpoint is used to transfer periodic data from the function controller to the host.

The Interrupt IN endpoint uses the same protocol as the Bulk IN endpoint and can be used in the same way. Although DMA can be used, it offers few benefits as the interrupt endpoint normally expects all data to be transferred in a single packet.

Interrupt IN endpoints also support a feature not supported by Bulk IN endpoints, namely that they support continuous switching of data switching bits. This feature is enabled by setting the FrcDataTog bit in the USB_INEPCSR register. When this bit is set to 1, the USB device will assume that the packet has been successfully sent and toggle the data bits for the endpoint, regardless of whether an ACK is received from the host.

34.4.10.2. Interrupt Write Transfer (INTERRUPT OUT)

The Interrupt OUT endpoint is used to transfer periodic data from the host to the function controller.

The Interrupt OUT endpoint uses almost the same protocol as the Bulk OUT endpoint and can be used in the same way. Although DMA can be used with an Interrupt OUT endpoint, it usually provides very little benefit as the Interrupt endpoint usually expects all data to be transferred in one packet.

34.5. USB registers

Note: When writing to a USB register you can only operate on a byte (byte) or word (word) basis, and when reading to a USB register you can only operate on a byte (byte) basis.

34.5.1. USB Control Register (USB_CR)

Address offset:0x00

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**
ISO Update	RES			Re-set	Re-sume	Sus-pend Mode	Enable Sus-pend	Up-date	ADD						
RW				R	RW	RW	RW	RW	RW						

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	Reserved
15	ISO Update	RW	0	ISO Update: After this position 1, the USB controller needs to wait for a SOF after InPktRdy is set to 1 before sending a packet; if the IN Token is received before the SOF is received, the USB controller will send a zero packet. (This register is only used for ISO transfers)
14:12	Reserved	-	-	Reserved
11	Reset	R	0	Reset: USB 总线有复位信号时，该 Bit 置 1
10	Resume	RW	0	Resume: When the USB device is in Suspend mode, the software sets to 1 to generate the Resume wakeup signal; the software clears this bit after 10mS. (15mS maximum)

Bit	Name	R/W	Reset Value	Function
9	Suspend Mode	RW	0	Suspend_Mode: set to 1 by the USB device hardware when it enters Suspend mode; this bit is cleared when the interrupt register is read by software, or when the Resume register is written by software
8	Enable Suspend	RW	0	Enable_Suspend： Suspend function enabled
7	Update	RW	0	Update： Set to 1 when Func Addr is written; cleared when the address takes effect (at the end of the transfer);
6:0	ADD	RW	0	ADD： Function address

34.5.2. USB Interrupt Status Register (USB_INTR)

Address offset:0x04

Reset value:0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										EP1IN	EP4IN	EP3IN	EP2IN	EP5IN	EP0
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res		EP1 OUT	EP4 OUT	EP3 OUT	EP2 OUT	EP5 OUT	Res					SOF	Reset	Re-sume	Sus-pend
0		0	0	0	0	0	0					0	0	0	0

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	0	Reserved
21	EP1IN	R	0	IN Endpoint 1 interrupt
20	EP4IN	R	0	IN Endpoint 4interrupt
19	EP3IN	R	0	IN Endpoint 3 interrupt
18	EP2IN	R	0	IN Endpoint 2 interrupt
17	EP5IN	R	0	IN Endpoint 5 interrupt
16	EP0	R	0	Endpoint 0 Interrupt
15:14	Reserved	-	0	Reserved
13	EP1OUT	R	0	OUT endpoint 1 interrupt
12	EP4OUT	R	0	OUT endpoint 4 interrupt
11	EP3OUT	R	0	OUT endpoint 3 interrupt
10	EP2OUT	R	0	OUT endpoint 2 interrupt
9	EP5OUT	R	0	OUT endpoint 5 interrupt

Bit	Name	R/W	Reset Value	Function
8:4	Reserved	-	0	Reserved
3	SOF	R	0	SOF interrupt, set to 1 at the start of each frame
2	Reset	R	0	Reset interrupt, set to 1 when a reset signal is detected on the USB bus
1	Resume	R	0	Resume interrupt, set to 1 when the Resume signal is detected on the USB bus when the USB device is in Suspend mode
0	Suspend	R	0	Suspend interrupt, set to 1 when the Suspend signal is detected on the USB bus

34.5.3. USB interrupt enable register (USB_INTRE)

Address offset:0x08

Reset value: 0x003F3E06

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										EP1INE	EP4INE	EP3INE	EP2INE	EP5INE	EP0E
0										1	1	1	1	1	1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		EP1OUTE	EP4OUTE	EP3OUTE	EP2OUTE	EP5OUTE	Res					SOFE	ResetE	ResumeE	SuspendE
0		1	1	1	1	1	0					0	1	1	0

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	0	Reserved
21	EP1INE	RW	1	IN Endpoint 1 interrupt enable
20	EP4INE	RW	1	IN Endpoint 4 interrupt enable
19	EP3INE	RW	1	IN Endpoint 3 interrupt enable
18	EP2INE	RW	1	IN Endpoint 2 interrupt enable
17	EP5INE	RW	1	IN Endpoint 5 interrupt enable
16	EP0E	RW	1	Endpoint 0 Interrupt enable
15:14	Reserved	-	0	Reserved
13	EP1OUTE	RW	1	OUT Endpoint 1 Interrupt enable
12	EP4OUTE	RW	1	OUT Endpoint 4 Interrupt enable
11	EP3OUTE	RW	1	OUT Endpoint 3 Interrupt enable
10	EP2OUTE	RW	1	OUT Endpoint 2 Interrupt enable

Bit	Name	R/W	Reset Value	Function
9	EP5OUTE	RW	1	OUT Endpoint 5 Interrupt enable
8:4	Reserved	-	0	Reserved
3	SOFE	RW	0	SOF interrupt enable
2	ResetE	RW	1	Reset interrupt enable
1	ResumeE	RW	1	Resume interrupt enable
0	SuspendE	RW	0	Suspend interrupt enable

34.5.4. USB frame register (USB_FRAME)

Address offset:0x0C

Reset value: 0x00

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

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	0	Reserved
19:16	INDEX	R	0	Endpoint selection
15:11	Reserved	-	0	Reserved
10:0	FRAMENUM	R	0	Last received frame number

34.5.5. USB Endpoint 0 Control Register (USB_EP0CSR)

Address offset:0x10

Reset value: 0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	COUNT0							Serviced-SetupEnd	Ser-vicedOut-PktRdy	Send-Stall	Set-upEnd	Da-taEnd	Sent-Stall	InPk-tRdy	Out-Pkt Rdy
0	0							0	0	0	0	0	0	0	0

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	0	Reserved
14:8	COUNT0	R	0	Length of data received by EP0, valid for reading when OutPktRdy is set to 1
7	ServicedSetupEnd	W1	0	Software write 1 to clear SetupEnd, this bit is automatically cleared
6	ServicedOutPktRdy	W1	0	Software write 1 clears OutPktRdy and the bit is automatically cleared
5	SendStall	W1	0	software write 1 terminates the current transmission; the STALL handshake is sent, after which the bit is automatically cleared to zero;
4	SetupEnd	R	0	This bit, at the end of the control transfer, DataEnd set 1 precedes 1, generates an interrupt and clears the FIFO;
3	DataEnd	W1	0	The software writes 1 and the bit is automatically cleared in the following cases; 1, after the last packet is sent and InPktRdy is set. 2, after the most recent packet has been read and OutPktRdy has been cleared to zero by software 3, after sending a zero packet and setting InPktRdy;
2	SentStall	RW0	0	Set to 1 after sending STALL handshake, cleared by software
1	InPktRdy	RW1	0	Software Write 1 when a packet is written to the FIFO. The packet is cleared when the packet transfer is complete and an interrupt is generated when it is cleared.
0	OutPktRdy	R	0	This bit is set to 1 upon receipt of a packet and generates an interrupt

34.5.6. USB IN Endpoint Control Register (USB_INEPxCSR)

Address offset:0x14

Reset value: 0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								INMAXP							
0								0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	ClrDataTog	SentSta	SendSta	FlushFIF	UnderR	FIFONotEmpty	InPktR	AutoS	ISO	Mode	DMAE	FrcDat	Res		

		ll	ll	O	un		dy	et				aTo g	
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	0	Reserved
23:16	INMAXP	RW	0	Maximum packet length of IN endpoint in 8bytes; one for each IN endpoint, except EP0
15	Reserved	-	0	Reserved
14	ClrDataTog	W1	0	Software write 1, reset IN EP data toggle to 0
13	SentStall	RW0	0	Set to 1 after sending the STALL handshake packet; at this point the FIFO should be cleared and InPktRdy should also be cleared, this bit is cleared by software
12	SendStall	RW	0	Software write 1 to send a STALL handshake upon receipt of the IN Token; software clear to terminate the Stall transmission, this bit is not valid for ISOs;
11	FlushFIFO	W1	0	Software write 1 clears the IN FIFO; only the next packet to be transmitted can be cleared, if there are two packets of data in the FIFO, two writes are required and the bit is automatically cleared;
10	UnderRun	R	0	In ISO mode, when a zero packet is sent after receiving an IN Token and InPktRdy is not set to 1, this position is 1; in Bulk and Int modes, when the USB device receives an IN Token and replies with a NAK, this bit is cleared by software
9	FIFONotEmpty	RW0	0	IN FIFO non-empty flag
8	InPktRdy	RW	0	When a packet is written to the FIFO, the software writes 1; the packet is cleared when the transfer is complete and an interrupt is generated after the clearing
7	AutoSet	RW	0	After setting 1, InPktRdy is automatically set to 1 when the maximum packet (InMaxP) is written in the IN FIFO
6	ISO	RW	0	Set to 1 to enable ISO transmission, clear to Bulk or Interrup transmission
5	Mode	RW	0	1: IN endpoint; 0: OUT endpoint.
4	DMAE	RW	0	In Endpoint DMA request enable

Bit	Name	R/W	Reset Value	Function
3	FrcDataTog	RW	0	Set to 1 to force the data toggle signal and clear the packets from the FIFO, regardless of whether an ACK is received
2:0	Reserved	-	0	Reserved

34.5.7. OUT endpoint control register (USB_OUTEPxCSR)

Address offset:0x14

Reset value: 0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								OUTMAXP							
0								0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ClrDataTog	SentStall	SendStall	FlushFIFO	DataError	OverRun	FIFOFull	OutPktRdy	AutoClear	ISO	DMAE	DMAMode	Res			
0	0	0	0	0	0	0	0	0	0	0	0	0			

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	0	Reserved
23:16	OUTMAXP	RW	0	The largest packet at the OUT endpoint, one for each OUT endpoint, except EP0;
15	ClrDataTog	W1	0	Software write 1 to reset the data toggle of the EP to 0;
14	SentStall	RW0	0	Set to 1 at the end of sending the STALL handshake; this bit is cleared by software;
13	SendStall	RW	0	Software write 1 to send STALL handshake and software clear to end STALL; Not valid in ISO mode;
12	FlushFIFO	W1	0	Clear the OUT FIFO, writing once to clear one packet of data;
11	DataError	R	0	Packets with CRC errors or bit-stuff errors when OutPktRdy is set to 1; automatically cleared when OutPktRdy is cleared to zero; valid only in ISO mode;
10	OverRun	RW0	0	OUT packets can no longer be written to the OUT FIFO and are cleared by software; Valid only in ISO mode;
9	FIFOFull	R	0	OUT FIFO full flag

Bit	Name	R/W	Reset Value	Function
8	OutPktRdy	RW0	0	Set to 1 upon receipt of the packet; an interrupt is generated when the data is read from the FIFO and cleared by software;
7	AutoClear	RW	0	When set to 1, OutPktRdy is automatically cleared when the data read from the OUT FIFO reaches the value of OutMaxP
6	ISO	RW	0	1: ISO mode; 0: Bulk or Interrupt mode;
5	DMAE	RW	0	DMA enable
4	DMAMode	RW	0	0: DMA requests are generated for all received packets and interrupts are generated; 1: DMA requests are generated on receipt of data from OutMaxP with no interrupts; other packet sizes, generate interrupts but not DMA requests;
3:0	Reserved	-	0	Reserved

34.5.8. USB OUT endpoint count register (USB_OUTCOUNT)

Address offset:0x1C

Reset value: 0x00

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

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	0	Reserved
10:0	OUTCOUNT	R	0	Length of data received, valid for reading when OutPktRdy is set to 1

34.5.9. USB FIFO register (USB_FIFO)

Address offset:0x20-0x33

Reset value: 0x00

Note: FIFO can only be operated by word or byte

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

FIFODATA															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIFODATA															
-															

Bit	Name	R/W	Reset Value	Function
31:0	FIFODATA	-	-	FIFODATA(0~5, each EP occupies 4 Byte addresses)

34.5.10. USB register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	USB_CR	Reserved																ISO Up-	Reserved			Reset	Resume	Suspend	Enable	Update	ADD						
	ResVal	0																0	0			0	0	0	0	0	0						
0x04	USB_INTR	Reserved									EP1IN		EP4IN	EP3IN	EP2IN	EP5IN	EP0	Reserved		EP1OUT	EP4OUT	EP3OUT	EP2OUT	EP5OUT	Reserved					SOF	Reset	Resume	Suspend
	ResVal	0									0		0	0	0	0	0	0		0	0	0	0	0	0					0	0	0	0
0x08	USB_INTRE	Reserved									EP1INE		EP4INE	EP3INE	EP2INE	EP5INE	EP0E	Reserved		EP1OUTE	EP4OUTE	EP3OUTE	EP2OUTE	EP5OUTE	Reserved					EN_SOF	EN_Reset	EN_Resume	EN_Suspend
	ResVal	0									1		1	1	1	1	1	0		1	1	1	1	1	0					0	1	1	0
0x0C	USB_FRAME	Reserved												INDEX				Reserved					FramNUM										
	ResVal	0												0				0					0										
0x10	USB_EP0CSR	Reserved																	COUNT0							ServicedSetupEnd	ServicedOutPktRdy	SendStall	SetupEnd	DataEnd	SentStall	InPktRdy	OutPktRdy
	ResVal	0																	0							0	0	0	0	0	0	0	0
0x14	USB_INEPxCSR	Reserved								INMAXP								Reserved	ClrDataTog	SentStall	SendStall	FlushFIFO	UnderRun	FIFONotEmpty	InPktRdy	AutoSet	ISO	Mode	DMAEnab	FrcDataTog	Reserved		
	ResVal	0								0								0	0	0	0	0	0	0	0	0	0	0	0	0	0		

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x18	USB_OUTEPxCSR	Reserved								OUTMAXP								ClrDataTog	SentStall	SendStall	FlushFIFO	DataError	OverRun	FIFOFull	OutPktRdy	AUTOClear	ISO	DMAEnab	DMAMode	Reserved			
	ResVal	0								0								0	0	0	0	0	0	0	0	0	0	0	0	0			
0x1C	USB_OUTCOUNT	Reserved																					OUTCOUNT										
	ResVal	0																					0										
0x20~0x33	USB_FIFODATA	FIFO-DATA(0~5,每个 EP 占用 4 个 Byte 地址)																															
	ResVal	0																															

35. Debug support

35.1. Overview

The chip is based on the Cortex-M0+ CPU, which includes advanced debug hardware extensions. The hardware debug module allows the kernel to stop when fetching fingers (instruction breakpoints) or accessing data (data breakpoints). When the kernel is stopped, both the internal state of the kernel and the external state of the system are available for query. After completing the query, the kernel and peripherals can be recovered and the program will continue to execute.

The debug function is used by the debug host when connecting and debugging the MCU, the interface for debugging is a serial wire, the debug function in the M0+ CPU Core is an ARM CoreSight Design kit.

M0+ provides integrated on-chip debug support consisting of the following components:

- SW-DP: serial wire
- BPU: Break point unit
- DWT: Data watchpoint trigger

The debug support also includes the debug integration features of this chip:

Flexible debug pin assignment, SWIO@PA13, SWCLK@PA14

MCU debug box (supports low power mode, control of peripheral clocks, etc.)

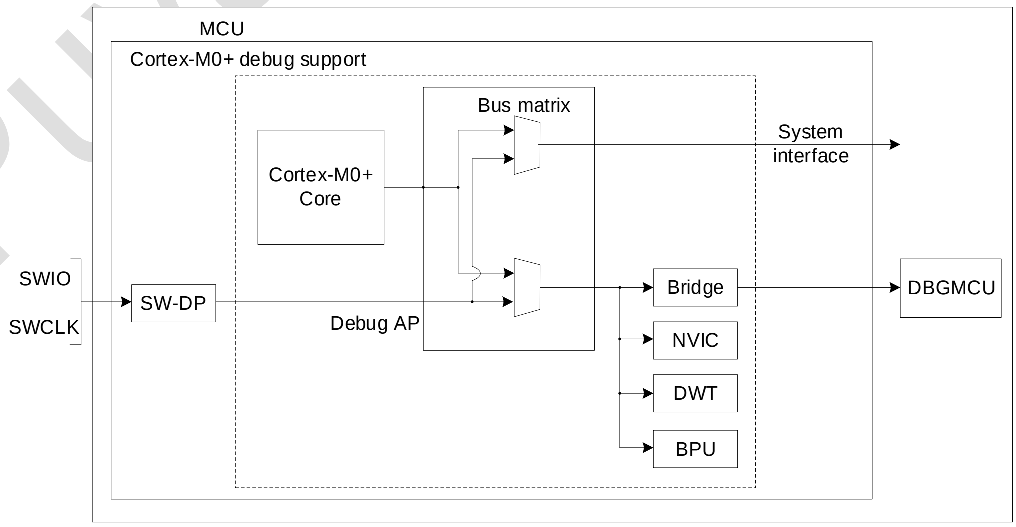


Figure 35-1 DBG Block Diagram

35.2. Pinouts and debug port pins

35.2.1. SWD debug ports

There are two ports associated with the debug function, which are visible in all package forms.

Table 35-1 DBG Block Diagram

SW-DP Port Pin Name	SW Debug Interface		Pin Assignment
	Type	Debugging function	
SWDIO	input/output	input/output Serial data	PA13
SWDCLK	input	Serial clock	PA14

35.2.2. SW-DP pin assignment

After reset (SYSRESETn or PORESETn), the pins used for the SW-DP are assigned as dedicated pins which are immediately usable by the debugger host.

However, the MCU offers the possibility to disable the SWD port and can then release the associated pins for general-purpose I/O (GPIO) usage

35.2.3. 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

Having embedded pull-up and pull-down resistors removes the need to add external resistors.

35.3. ID codes and locking mechanism

Here are several ID codes inside the MCU. It is recommended that Keil, IAR and other tools use this ID Code (located at 0x4001 5800) locks debugging.

After the chip is powered on, the hardware reads the 0x1FFF 0FF8 address of the flash 's factory config.byte and loads it into the DBG _IDCODE register.

35.4. SWD debug port

35.4.1. 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 turnaround time is one bit time, however this can be adjusted by configuring the SWCLK frequency.

35.4.2. SWD protocol sequence

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 35-2 Packet request (8 bits)

Bit	Name	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	Single bit parity of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as “1” by the target because of the pull-up

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drive the line.

Table 35-3 ACK response (3 bits)

Bit	Name	Description
[2:0]	ACK	001: FAULT

Bit	Name	Description
		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 35-4 DATA transfer (33 bits)

Bit	Name	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.

35.4.3. 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 (corresponding to Cortex®-M0).

35.4.4. DP and AP read/write accesses

- Read accesses to the DP are not posted: the target response can be immediate (if ACK = OK) or can be delayed (if 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 done is NOT an AP access, then the DP-RDBUFF register must be read to obtain the result. The READOK flag of the DP-CTRL/STAT register is updated on every AP read access or RDBUFF read request to know if the AP read access was successful.
- The SW-DP implements a write buffer (for both DP or AP writes), that enables it toaccept a write operation even when other transactions are still outstanding. If the write buffer is full, the target acknowledge response is “WAIT”. With the exception of IDCODE read or CTRL/STAT read or ABORT write which are accepted even if the write buffer is full.
- Because of the asynchronous clock domains SWCLK and HCLK, two extra SWCLK cycles are needed after a write transaction (after the parity bit) to make the write effective internally. These cycles should be applied while driving the line low (IDLE state) This is particularly important when writing the

CTRL/STAT for a power-up request. If the next transaction (requiring a power-up) occurs immediately, it will fail.

35.4.5. SW-DP registers

Access to these registers are initiated when APnDP = 0.

A[3:2]	R/W	CTRLSEL bit of SELECT register	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	-

35.4.6. SW-AP registers

Address	A[3:2] value	Description
0x0	00	Reserved
0x4	01	DP CTRL/STAT register. Used to: ■ Request a system or debug power-up ■ Configure the transfer operation for AP accesses ■ Control the pushed compare and pushed verify operations. ■ Read some status flags (overrun, power-up acknowledges)
0x8	10	DP SELECT register: Used to select the current access port and the active 4-words register window. ■ Bits 31:24: APSEL: select the current AP ■ Bits 23:8: reserved ■ Bits 7:4: APBANKSEL: select the active 4-words register window on the current AP ■ Bits 3:0: reserved
0xC	11	DP RDBUFF register: Used to allow the debugger to get the final result after a sequence of operations (without requesting new JTAG-DP operation)

35.5. Core debug

Core debug is accessed through the core debug registers. Debug access to these registers is by means of the debug access port. It consists of four registers:

Table 35-5 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 are only reset by a power-on reset. Refer to the Cortex®-M0+ TRM for further details. To Halt on reset, it is necessary to:

- Enable the bit0 (VC_CORRESET) of the Debug and Exception Monitor Control Register
- Enable the bit0 (C_DEBUGEN) of the Debug Halting Control and Status Register

35.6. Break point unit (BPU)

The Cortex-M0+ BPU implementation provides four breakpoint registers. The BPU is a subset of the Flash Patch and Breakpoint (FPB) block available in ARMv7-M (Cortex-M3 & Cortex-M4).

35.6.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.

Refer the ARMv6-M ARM and the ARM CoreSight Components Technical Reference Manual for more information about the BPU CoreSight identification registers, and their addresses and access types.

35.7. Data watchpoint (DWT)

The Cortex-M0 DWT implementation provides two watchpoint register sets

35.7.1. DWT functionality

The processor watchpoints implement both data address and PC based watchpoint functionality.

35.7.2. 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 provides coarse grained profiling. See the ARMv6-M ARM for more information.

The Cortex-M0 DWT_PCSR records both instructions that pass their condition codes and those that fail.

35.8. MCU debug component (DBGMCU)

The MCU debug component helps the debugger provide support for:

- Low-power modes
- Clock control for timers, watchdog and I2C during a breakpoint

35.8.1. 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 core does not allow FCLK or HCLK to be turned off during a debug session. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug software in low-power modes. For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior:

- In Sleep mode: FCLK and HCLK are still active. Consequently, this mode does not impose any restrictions on the standard debug features.In stop mode: The DBG_STOP bit must be set in advance by the debugger.
- In Stop/Standby mode, the DBG_STOP bit must be previously set by the debugger. This enables the internal RC oscillator clock to feed FCLK and HCLK in Stop mode.

35.8.2. Debug support for times, watchdog and IIC

During a breakpoint, it is necessary to choose how the counter of timers and watchdog should behave:

- They can continue to count inside a breakpoint. This is usually required when a PWM is controlling a motor, for example.
- They can stop to count inside a breakpoint. This is required for watchdog purposes. For the I2C, the user can choose to block the SMBUS timeout during a breakpoint.

35.9. DBG register

35.9.1. DBG device ID code register (DBG_IDCODE)

Address offset: 0x00

Only supports 32-bit address access, read only.

This register can be accessed via the software debug port (2 pin) or the user software.

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

Bit	Name	R/W	Reset Value	Function
31：0	REV_ID	R		Revision ID

35.9.2. Debug MCU configuration register (DBGMCU_CR)

This register configures the MCU low power mode in debug state.

This register is asynchronously reset by a power-on reset (not a 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 software user to write to these registers.

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			
1	DBG_STOP	RW	0	Debug Stop mode 0: (FCLK = Off, HCLK = Off) In STOP mode, the clock controller disables HCLK and FCLK. When exiting from STOP mode, the clock configuration is identical to the one after RESET (CPU clocked by the HSI). Consequently, the software must reprogram the clock controller to enable the clock configuration. 1: (FCLK = on, HCLK = on). When entering STOP mode, HSI will not be turned off, and FCLK and HCLK are generated by I. When exiting STOP mode, if the clock control needs to be changed, the software needs to be reconfigured.
0	DBG_SLEEP	RW	0	Debug sleep mode. 0: (FCLK on, HCLK off). In sleep mode, FCLK is provided by the originally configured system clock and HCLK is off. Since sleep mode does not reset the configured clock system, the software does not need to reconfigure the clock after exiting from sleep mode. 1: (FCLK on, HCLK on) In sleep mode, both the FCLK and HCLK clocks are provided by the originally configured system clock.

35.9.3. DBG APB freeze register 1 (DBG_APB_FZ1)

This register is used to configure the clock of timer, RTC, IWDG, WWDG under debug. This register is asynchronously reset by a power-on reset (not a 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	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res	DBG_CAN_STOP	Res	Res	Res
RW									R	RW		RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Res	Res	Res	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res	Res	Res	Res	DBG_TIM7_STOP	DBG_TIM6_STOP	Res	Res	DBG_TIM3_STOP	DBG_TIM2_STOP
			RW	RW	RW					rw				RW	RW

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM_STOP	RW	0	When the CPU is stopped, the counter clock control bit of the L PTIM 0: enable 1: Disable
30：23	Reserved			
22	DBG_I2C2_SMBUS_TIMEOUT	R	0	Fixed to 0
21	DBG_I2C1_SMBUS_TIMEOUT	RW	0	Controls whether the I2C1 SMBUS timeout is frozen when the CPU core is in the halt state. 0: same processing as normal mode; 1: SMBUS timeout count frozen.
20	reserved			
19	DBG_CAN_STOP	RW	0	Control CAN stop when CPU core is in halt state. 0: when the CPU halt, the CAN is in the same mode as normal; 1: CAN receive register disabled when CPU halt;
12	DBG_IWDG_STOP	RW	0	When the CPU is stopped, the clock control bit of the IWDG counter 0: enable 1: Disable
11	DBG_WWDG_STOP	RW	0	When the CPU is stopped, the clock control bit of the WWDG counter 0: enable 1: Disable
10	DBG_RTC_STOP	RW	0	When the CPU is stopped, the clock control bit of the RTC counter 0: enable 1: Disable
9：6	Reserved			
5	DBG_TIM7_STOP	RW	0	Controls the count clock of TIM7 when the CPU core is in the halt state.

Bit	Name	R/W	Reset Value	Function
				0: clock enable; 1: clock off;
4	DBG_TIM6_STOP	RW	0	Controls the count clock of TIM6 when the CPU core is in halt state. 0: clock enable; 1: clock off;
3:2	reserved			
1	DBG_TIM3_STOP	RW	0	TIM 3 counter when the CPU is stopped 0: enable 1: Disable
0	DBG_TIM2_STOP	RW	0	Controls the count clock of TIM2 when the CPU core is in halt state. 0: clock enable; 1: clock off;

35.9.4. DBG APB freeze register 2 (DBG_APB_FZ2)

This register is used to configure the clock control of timer under debug. This register is asynchronously reset by a power-on reset (not a system reset). It can be written by the debugger under system reset.

Address offset: 0x0C

Power on Reset value: 0x0000 0000

Only supports 32-bit address access, 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	DBG_TIM15_STOP
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBG_TIM14_STOP	Res	Res	Res	DBG_TIM1_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
RW				RW											

Bit	Name	R/W	Reset Value	Function
31:19	Reserved			

Bit	Name	R/W	Reset Value	Function
18	DBG_TIM17_STOP			When the CPU is stopped, the clock control bit of the TIM17 counter 0: Enable 1: Disable
17	DBG_TIM16_STOP			When the CPU is stopped, the clock control bit of the TIM16 counter 0: Enable 1: Disable
16	Reserved			
15	DBG_TIM14_STOP			When the CPU is stopped, the clock control bit of the TIM14 counter 0: Enable 1: Disable
14:12	Reserved			
11	DBG_TIM1_STOP			When the CPU is stopped, the clock control bit of the TIM1 counter 0: Enable 1: Disable
10:0	Reserved			

35.9.5. DBG register map

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DBG_ID-CODE	REV_ID																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	DBG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STO	DBG_SLE
	Reset value																															0	0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x08	DBG_APB_FZ1	DBG_LPTIM_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I2C2_SMBUS_T	DBG_I2C1_SMBUS_T	Res.	DBG_CAN_STOP	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	Res.	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
	Reset value	0									0	0		0							0	0	0					0				0	0
0x0C	DBG_APB_FZ2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM1_STOP	DBG_TIM14_STOP	Res.	Res.	Res.	DBG_TIM1_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value														0	0	0	0				0											

36. Updated History

Version	Date	Updated record
V0.1	2023.04.20	1. First edition

Puya Semiconductor Co., Ltd.

IMPORTANT NOTICE