PY32T020-B Reference Manual

32-bit ARM® Cortex®-M0+ Microcontrollers

Puya Semiconductor (Shanghai) Co., Ltd.

Contents

1. List of abbreviations for registers

Abbreviation	Description
Read/Write (RW)	Software can read and write to this bit.
Read-only (R)	Software can only read this bit.
Write-only (W)	Software can only write to this bit.
Read/Clear Write0 (RC_W0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on this bit.
Read/Clear Write1 (RC_W1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on this bit.
Read/Clear Write (RC_W)	The software can read and clear the bit by writing to the register. The value written to this bit is not important.
Read/Clear by read (RC_R)	Software can read this bit. Reading this bit automatically clears it to 0. Writing this bit has no effect on the bit value.
Read/Set by Read (RS_R)	Software can read this bit. Reading this bit automatically sets it to 1. Writing this bit has no effect on the bit value.
Read/Set (RS)	Software can read as well as set this bit by writing 1. Writing 0 has no effect on this bit.
Toggle (T)	The software can toggle this bit by writing 1. Writing 0 has no effect.
Reserved (Res.)	Reserved bit, must be kept at reset value.

2. System block diagram

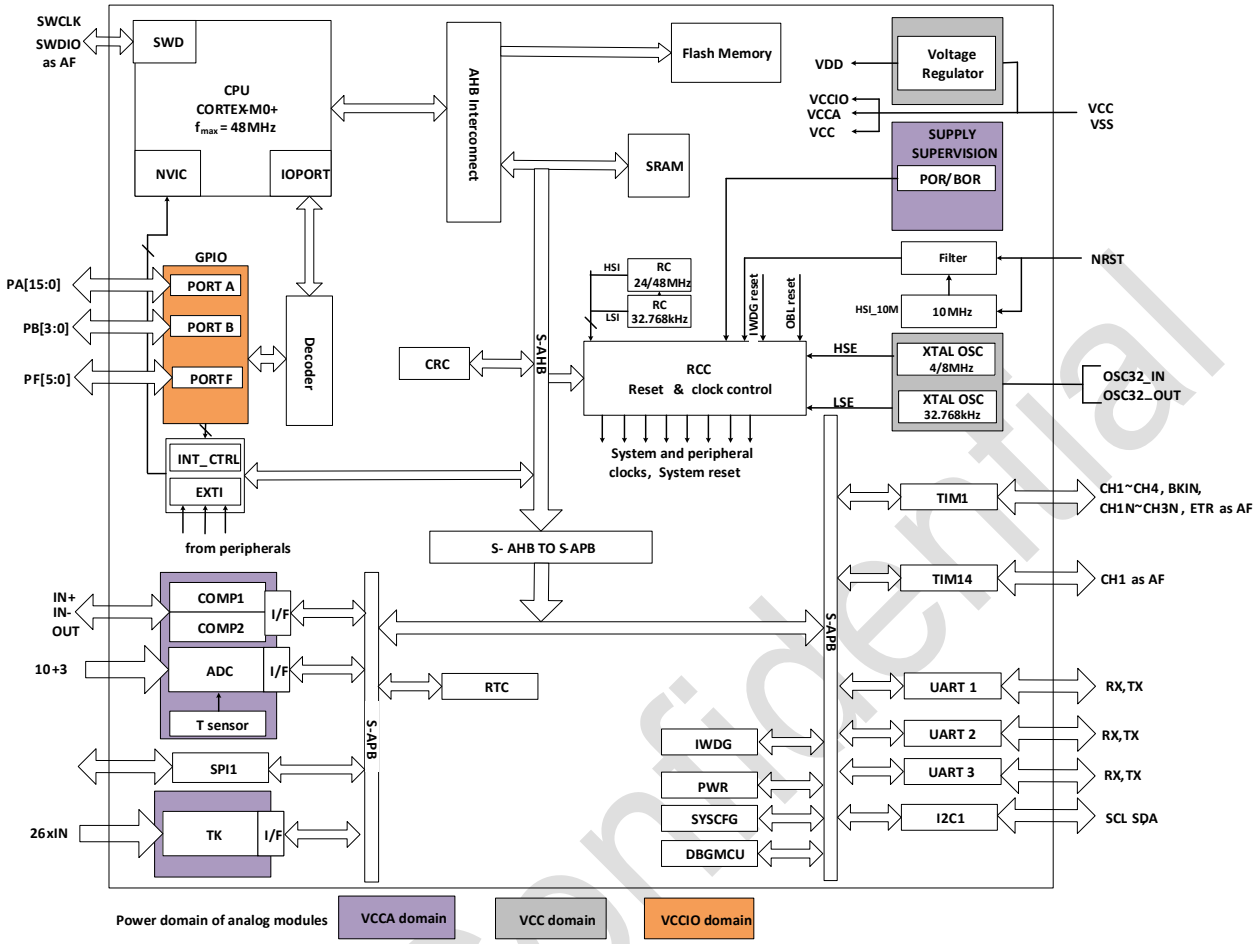


Figure 2-1 System block diagram

3. Memory and bus architecture

3.1. System architecture

The system consists of:

- A Master
  - Cortex-M0+
- Three Slaves
  - Internal SRAM
  - Internal Flash memory
  - Other AHB slave (including AHB-to-APB bridge, CRC, EXTI, RCC)

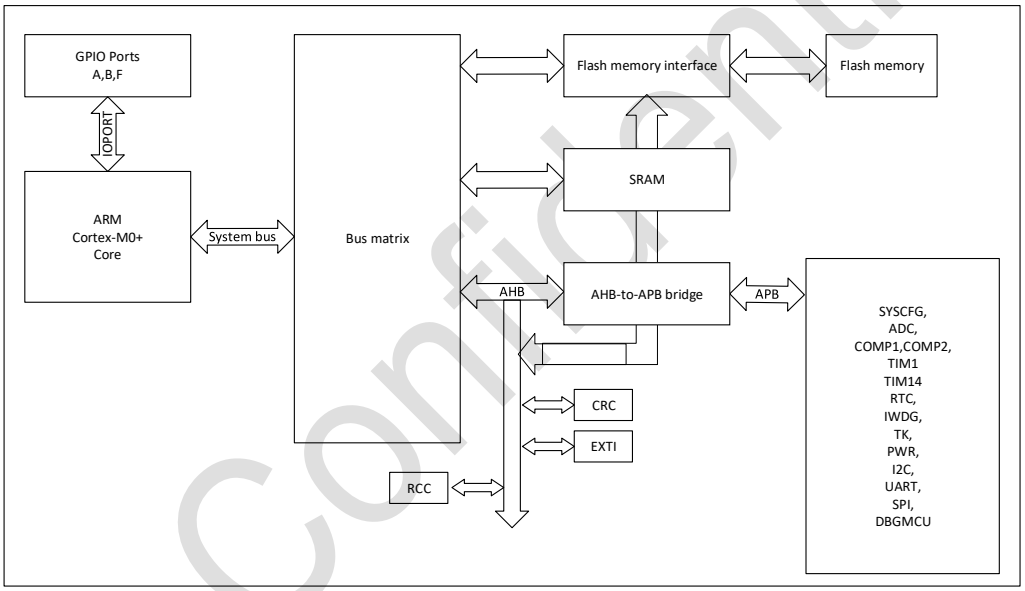


Figure 3-1 System architecture

- AHB

The system bus of the Cortex-M0 + is connected to the on-chip AHB bus through which the CPU accesses Flash, SRAM, CRC, EXTI, RCC and peripherals through the AHB-to-APB bridge.

- AHB-to-APB bridge (APB)

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

3.2. Memory organization

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

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

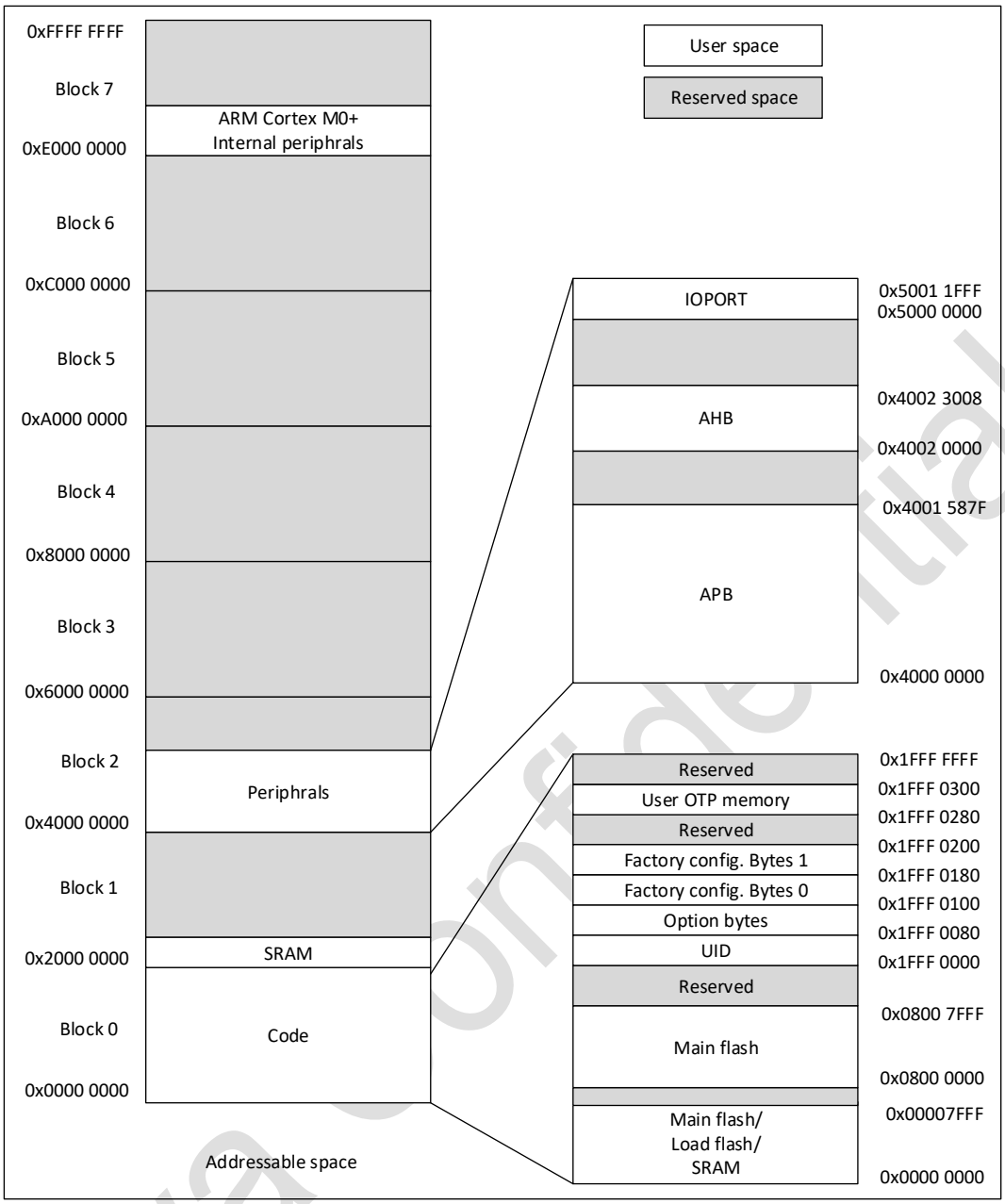


Figure 3-2 Memory map

Table 3-1 Memory Address

Type	Boundary Address	Size	Memory Area
SRAM	0x2000 1000-0x3FFF FFFF	-	Reserved
	0x2000 0000-0x2000 0FFF	4 KB	SRAM
Code	0x1FFF 0300-0x1FFF FFFF	-	Reserved
	0x1FFF 0280-0x1FFF 02FF	128 Bytes	User OTP memory
	0x1FFF 0180-0x1FFF 01FF	128 Bytes	Factory config. Bytes 1
	0x1FFF 0100-0x1FFF 017F	128 Bytes	Factory config. Bytes 0
	0x1FFF 0080-0x1FFF 00FF	128 Bytes	Option bytes
	0x1FFF 0000-0x1FFF 007F	128 Bytes	UID
	0x0800 8000-0x1FFE FFFF	-	Reserved
	0x0800 0000-0x0800 7FFF	32 KB	Main flash memory
	0x0000 8000-0x07FF FFFF	-	Reserved
	0x0000 0000-0x0000 7FFF	32 KB	Depending on the Boot configuration: 1. Main flash memory 2. Load flash 3. SRAM

Table 3-2 Peripheral register address

Bus	Boundary Address	Size	Peripheral(1)
	0xE000 0000-0xE00F FFFF	1 MB	M0+
IOPORT	0x5000 1800-0x5FFF FFFF	-	Reserved (1)
	0x5000 1400-0x5000 17FF	1 KB	GPIOF
	0x5000 0800-0x5000 13FF	-	Reserved (1)
	0x5000 0400-0x5000 07FF	1 KB	GPIOB
	0x5000 0000-0x5000 03FF	1 KB	GPIOA
AHB	0x4002 3400-0x4FFF FFFF	-	Reserved
	0x4002 3010-0x4002 33FF	1 KB	Reserved
	0x4002 3000-0x4002 300F		CRC
	0x4002 2400-0x4002 2FFF	-	Reserved
	0x4002 2000-0x4002 23FF	1 KB	Flash FMC
	0x4002 1C00-0x4002 1FFF	-	Reserved
	0x4002 1900-0x4002 1BFF	1 KB	Reserved
	0x4002 1800-0x4002 18FF		EXTI(2)
	0x4002 1400-0x4002 17FF	-	Reserved
	0x4002 1080-0x4002 13FF	1 KB	Reserved
	0x4002 1000-0x4002 107F		RCC(2)
	0x4002 0000-0x4002 0FFF	-	Reserved
APB	0x4001 5C00-0x4001 FFFF	-	Reserved
	0x4001 5880-0x4001 5BFF	1 KB	Reserved
	0x4001 5800-0x4001 587F		DBG
	0x4001 4800-0x4001 57FF	-	Reserved
	0x4001 4480-0x4001 47FF	1 KB	Reserved
	0x4001 4400-0x4001 447F		UART2
	0x4001 3C00-0x4001 43FF	-	Reserved
	0x4001 3880-0x4001 3BFF	1 KB	Reserved
	0x4001 3800-0x4001 387F		UART3
	0x4001 3400-0x4001 37FF	-	Reserved
	0x4001 3080-0x4001 33FF	1 KB	Reserved

Bus	Boundary Address	Size	Peripheral(1)
	0x4001 3000-0x4001 307F		SPI
	0x4001 2C80-0x4001 2FFF	1 KB	Reserved
	0x4001 2C00-0x4001 2C7F		TIM1
	0x4001 2800-0x4001 2BFF	-	Reserved
	0x4001 2400-0x4001 27FF	1 KB	ADC
	0x4001 0400-0x4001 23FF	-	Reserved
	0x4001 0220-0x4001 03FF	1 KB	Reserved
	0x4001 0200-0x4001 021F		CMP1/2
	0x4001 0000-0x4001 01FF		SYSCFG
	0x4000 7400-0x4000 FFFF	-	Reserved
	0x4000 7080-0x4000 73FF	1 KB	Reserved
	0x4000 7000-0x4000 707F		PWR(3)
	0x4000 5800-0x4000 6FFF	-	Reserved
	0x4000 5480-0x4000 57FF	1 KB	Reserved
	0x4000 5400-0x4000 547F		I^2C
	0x4000 4800-0x4000 53FF	-	Reserved
	0x4000 4480-0x4000 47FF	1 KB	Reserved
	0x4000 4400-0x4000 447F		UART1
	0x4000 3C00-0x4000 43FF	-	Reserved
	0x4000 3880-0x4000 3BFF	1 KB	Reserved
	0x4000 3800-0x4000 387F		TK
	0x4000 3400-0x4000 37FF	-	Reserved
	0x4000 3080-0x4000 33FF	1 KB	Reserved
	0x4000 3000-0x4000 307F		IWDG
	0x4000 2C00-0x4000 2FFF	-	Reserved
	0x4000 2880-0x4000 2BFF	1 KB	Reserved
	0x4000 2800-0x4000 287F		RTC
	0x4000 2400-0x4000 27FF	-	Reserved
	0x4000 2080-0x4000 23FF	1 KB	Reserved
	0x4000 2000-0x4000 207F		TIM14
	0x4000 0000-0x4000 1FFF	-	Reserved

1. In the above table, the reserved address cannot be written, read back is 0, and a HardFault is generated.
2. Not only supports 32-bit word access, but also supports half-word and byte access.
3. Not only supports 32-bit word access, but also supports half-word access.

3.3. Embedded SRAM

PY32T020-B series feature 4 KB SRAM. It is accessed by half-word (16 bits) or word (32 bits).

3.4. Flash memory

The Flash memory is composed of two distinct physical areas:

- Main flash block: 32 KB (8K x 32 bits), used to store user programs and user data. In addition, a maximum of 4 KB can be set as a User bootloader according to customer configuration.
- Information block: 0.75 KB (192 x 32 bits), including:
  - Factory config. Bytes 0: 128 Bytes for storing:
    Store trimming data (including HSI trimming)
  - Factory config. Bytes 1: 128 Bytes for storing:

Hardware Trimming configuration value

- UID: 128 Bytes, used to store the UID
- Option byte: 128 Bytes, used to store configuration values for hardware and storage protection
- OTP flash: 128 Bytes, used to store user data

The protection of Main flash area includes the following mechanisms:

- Read protection (RDP) blocks external access
- Write protection (WRP) prevents unintended writes (caused by confusion of program). The minimum protection unit for write protection is 4 KB.
- Option byte write protection is a special design for unlock.
- SDK protection.

3.5. Boot modes

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

Table 3-3 Boot configuration

Boot mode configuration		Mode	
nBOOT1 bit	**nBOOT0 pin**	**Boot memory size = = 0**	**Boot memory size !=0**
X	0	Boot from Main flash	Boot from Main flash
0	1	Boot from SRAM	Boot from SRAM
1	1	N/A	Boot from Load flash(1)

1. Products with 20 KB Flash cannot boot from Load flash.

It is up to the user to set boot mode configuration related to the required boot mode.

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

- Boot from Main flash: the Main flash memory is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x0800 0000). In other words, the Flash memory contents can be accessed starting from address 0x0000 0000 or 0x0800 0000.
- Boot from user bootloader: The user bootloader memory is aligned in the boot memory space 0x0000 0000, but can still be accessed from the following address space according to the user bootloader size setting.

Table 3-4 Boot configuration

User Bootloader	Access address
None	-
1 KB	0x0800 7C00~0x0800 7FFF
2 KB	0x0800 7800~0x0800 7FFF
3 KB	0x0800 7400~0x0800 7FFF
4 KB	0x0800 7000~0x0800 7FFF

- Boot from SRAM: the SRAM is aliased in the boot memory space (0x0000 0000), but it is still accessible from its original memory space (0x2000 0000).

3.5.1. Memory physical mapping

If the Boot mode is selected, the application software can modify the memory that is accessible in the program space. This modification is performed by programming the MEM_MODE bits in the SYSCFG configuration register 1 (SYSCFG_CFGR1).

4. Embedded Flash memory

4.1. Flash main features

The Flash memory unit page size is 128 Bytes and unit sector size is 4 KB.

The Flash memory is composed of two distinct physical areas:

- Main flash block: 32 KB (8K x 32 bits), used to store user programs and user data. In addition, a maximum of 4 KB can be set as a User bootloader according to customer configuration.
- Information block: 0.75 KB (192 x 32 bits)

The protection of Main flash area includes the following mechanisms:

- Read protection (RDP) blocks external access
- Write protection (WRP) prevents unintended writes (caused by confusion of program). The minimum protection unit for write protection is 4 KB.
- Option byte write protection is a special design for unlock.
- SDK protection.

4.2. Flash memory functional description

4.2.1. Flash memory organization

The Flash memory is organized as 32-bit wide memory cells that can be used for storing both code and data constants. The Page size is 128 Bytes and the Sector size is 4 KB. Functionally, Flash memory is divided into Main flash and Information flash. The former has a capacity of 32 KB and the latter has a capacity of 0.75 KB.

The Page erase and sector erase operations can be applied to areas of the Main flash that are not write protected.

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

Table 4-1 Flash structure and boundary address

Block	Sector	Page	Base address	Size
Main flash	Sector 0	Page 0-31	0x0800 0000-0x0800 0FFF	4 KB
	Sector 1	Page 32-63	0x0800 1000-0x0800 1FFF	4 KB
	Sector 2	Page 64-95	0x0800 2000-0x0800 2FFF	4 KB
	Sector 3	Page 96-127	0x0800 3000-0x0800 3FFF	4 KB
	Sector 4	Page128-159	0x0800 4000-0x0800 4FFF	4 KB
	Sector 5	Page160-191	0x0800 5000-0x0800 5FFF	4 KB
	Sector 6	Page192-223	0x0800 6000-0x0800 6FFF	4 KB
	Sector 7	Page224-255	0x0800 7000-0x0800 7FFF	4 KB
UID	-	Page 0	0x1FFF 0000-0x1FFF 007C	128 Bytes
option bytes	-	Page 1	0x1FFF 0080-0x1FFF 00FC	128 Bytes
Factory config 0	-	Page 2	0x1FFF 0100-0x1FFF 017C	128 Bytes
Factory config 1	-	Page 3	0x1FFF 0180-0x1FFF 01FC	128 Bytes

Reserved	-	Page 4	0x1FFF 0200-0x1FFF 027C	128 Bytes
USER OTP memory	-	Page 5	0x1FFF 0280-0x1FFF 02FC	128 Bytes

4.2.2. Flash read operation and access latency

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

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

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

4.2.3. Flash program and erase operations

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

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

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

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

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

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

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

- Acess control register (FLASH_ACR)
- KEY register (FLASH_KEYR)
- Option byte key register (FLASH_OPTKEYR)
- Flash status register (FLASH_SR)
- Flash control register (FLASH_CR)
- Flash option register (FLASH_OPTR)
- FLASH SDK address register (FLASH_SDKR)
- Flash boot control register (FLASH_BTCR)

- Flash write protection resister (FLASH_WRPR)
- FLASH sleep time config register (FLASH_STCR)
- Flash TS0 register (FLASH_TS0)
- Flash TS1 register (FLASH_TS1)
- Flash TS2P register (FLASH_TS2P)
- Flash TPS3 register (FLASH_TPS3)
- Flash TS3 register (FLASH_TS3)
- Flash page erase TPE register (FLASH_PERTPE)
- Flash sector/mass erase TPE register (FLASH_SMERTPE)
- Flash program TPE register (FLASH_PRGTPE)
- Flash pre-program TPE register (FLASH_PRETPE)

4.2.3.1. Unlocking the Flash memory

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

This is done in the following steps:

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

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

Any wrong sequence locks up the FLASH_CR register until the next reset. In the case of a wrong key sequence, a bus error is detected, and a 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 to 1.

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

4.2.3.2. Flash memory programming

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

If the address main Flash memory location is write-protected by the FLASH_WRPR register, the program operation is skipped and a warninf is issued by the WROERR bit in the FLASH_SR register.

In addition, when part of the Main flash area is used as Load flash, the program operation will be ignored for the selected area, and the WRPERR bit of the FLASH_SR register will also be set. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.

The Flash memory programming sequence is as follows:

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

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

4.2.3.3. Flash memory erase

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

Note: sector and mass erase do not work for information memory.

Page erase

When a page is write-protected, it will not be erased and the WRPERR bit is set. In addition, when part of the Main flash area is used as Load flash, the program operation will be ignored for the selected page, and the WRPERR bit will also be set.

A page erase operation needs to perform the following steps:

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

Mass erase

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

In addition, when some Main flash areas are used as Load flash, the mass erase function is invalid, no mass erase operation will be generated, and the WEPERR bit will also be set.

The steps for Mass erase are as follows:

- Check the BSY bit to confirm if there are ongoing Flash operations
- Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
- Set the MER bit and EOPIE bit in the FLASH_CR register
- Write any data (32-bit data) to any Main flash space
- Wait for the BSY bit to be cleared.
- Check that EOP flag bit is set
- Clear EOP flag

Sector erase

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

In addition, when part of the Main flash area is used as Load flash, if sector7 is selected as the erase object, sector7 will not be erase, and the WRPERR bit will also be set.

The steps for a sector erase are as follows:

- Check the BSY bit to confirm if there are ongoing Flash operations
- Write KEY1 and KEY2 to the FLASH_KEYR register to unlock the protection of the FLASH_CR register
- Set the SER bit and EOPIE bit in the FLASH_CR register
- Write arbitrary data to this sector
- Wait for the BSY bit to be cleared.
- Check that EOP flag bit is set
- Clear EOP flag

4.2.3.4. Write and erase time configuration

The time of Flash program and erase needs to be strictly controlled, otherwise the operation will fail. If you need to write and erase Flash, you need to correctly configure the Flash write and erase time control registers according to the HSI output frequency, refer to the descriptions of Flash_TS0, Flash_TS1, Flash_TS2P, Flash_TPS3, Flash_TS3, Flash_PERTPE, Flash_SMERTPE, Flash_PRGTPE, and Flash_PRETPE.

4.2.4. Flash option bytes

4.2.4.1. Flash option byte description

Part of the information area of flash is used as option bytes, which are configured by the end user depending on the application requirements. As a configuration example, the watchdog may be selected in hardware or software mode.

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

Table 4-2 Option byte format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Complemented 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 software can read the option bytes from these flash memory locations or from their corresponding option registers referenced in the table.

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

Table 4-3 Option byte organization

Address	Description
0x1FFF 0080	Flash user option byte and complemented code
0x1FFF 0084	BOR configuration and its complemented code
0x1FFF 0088	Option byte and its complemented code of address area of flash memory SDK
0x1FFF 008C	WRP address option byte and complemented code
0x1FFF 0090	Reserved
0x1FFF 0094	Reserved
…	Reserved
…	Reserved
…	Reserved
0x1FFF 00FC	Reserved

4.2.4.2. Option bytes for Flash user options

Flash memory address: 0x1FFF 0080

Production value: 0x6D55 92AA

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

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

Bit	Name	R/W	Function
31	~ IWDG_STOP	R	Complemented code of IWDG_STOP
30	~NRST_MODE	R	Complemented code of NRST_MODE
29	Reserved		
28	~IWDG_SW	R	Complemented code of IWDG_SW
27: 25	~BOR_LEV[2:0]	R	Complemented code of BOR_LEV
24	~BOR_EN	R	Complemented code of BOR_EN

Bit	Name	R/W	Function
23: 16	~ RDP	R	Complemented code of RDP
15	IWDG_STOP	R	Set the running state of IWDG timer in Stop mode 0: Freeze timer 1: Normal operation
14	NRST_MODE	R	0: Reset pin used as NRST 1: Reset pin used as normal GPIO
13	Reserved		-
12	IWDG_SW	R	0: Hardware window watchdog 1: Software window watchdog
11: 9	BOR_LEV[2:0]	R	001: BOR rising threshold is 1.99 V and falling threshold is 1.88 V 010: BOR rising threshold is 2.19 V and falling threshold is 2.10 V 011: BOR rising threshold is 2.39 V and falling threshold is 2.30 V 100: BOR rising threshold is 2.78 V and falling threshold is 2.69 V 101: BOR rising threshold is 3.08 V and falling threshold is 2.99 V 110: BOR rising threshold is 3.68 V and falling threshold is 3.58 V 111: BOR rising threshold is 4.20 V and falling threshold is 4.08 V
8	BOR_EN	R	BOR enable 0: BOR OFF 1: BOR enabled, BOR_LEV works
7: 0	RDP	R	0xAA: level 0, read protection off Non-0xAA: level 1, read protection enabled

4.2.4.3. Flash SDK area address option byte

Flash memory address: 0x1FFF 0084

Production value: 0xFFF0 000F

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

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

Bit	Name	R/W	Function
31: 28	Reserved		
27: 24	~SDK_END[3:0]	R	Complemented code of SDK_END
23: 20	Reserved		
19: 16	~SDK_STRT[3:0]	R	Complemented code of SDK_STRT
15: 12	Reserved		
11: 8	SDK_END[3:0]	R	SDK area end address, the corresponding STEP of each bit is 2 KB
7: 4	Reserved		
3: 0	SDK_STRT[3:0]	R	SDK area start address, the corresponding STEP of each bit is 2 KB

4.2.4.4. Option byte for FLASH boot control

Flash memory address: 0x1FFF 0088

Production value: 0xFEFF 0100

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ nBOOT1	~ BOOT0	Res.	Res.	Res.	Res.	~SWD_MODE[1:0]		Res.	Res.	Res.	Res.	Res.	~BOOT_SIZE [2:0]		
R	R					R	R						R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
nBOOT1	BOOT0.	Res.	Res.	Res.	Res.	SWD_MODE[1 :0]		Res.	Res.	Res.	Res.	Res.	BOOT_SIZE [2:0]		
R	R					R	R						R	R	R

Bit	Name	R/W	Function
31	~ nBOOT1	R	Complemented code of nBOOT1
30	~ BOOT0	R	Complemented code of BOOT0
29: 26	Reserved		
25:24	~SWD_MODE[1:0]	R	Complemented code of SWD_MODE [1: 0]
23: 19	Reserved		
18: 16	~BOOT_SIZE [2:0]	R	Complemented code of BOOT_SIZE
15	nBOOT1	R	Boot mode of nBOOT1 / BOOT0 X0: Boot from Main Flash 11: Boot from Load Flash 01: Boot from SRAM
14	BOOT0	R	
13: 10	Reserved		
9:8	SWD_MODE[1:0]	R	Select the IO port as GPIO mode or SWD mode PF3 PF4 PA13 PA14 00: SWCLK SWDIO GPIO GPIO 01: GPIO GPIO SWDIO SWCLK 10: GPIO SWDIO GPIO SWCLK 11: SWCLK GPIO SWDIO GPIO
7:3	Reserved		
2: 0	BOOT_SIZE [2:0]	R	Select Main flash area as Load flash 000: No Load flash area 001: 1 KB（0x0800 7C00~0x0800 7FFF） 010: 2 KB（0x0800 7800~0x0800 7FFF） 011: 3 KB（0x0800 7400~0x0800 7FFF） 1xx: 4 KB（0x0800 7000~0x0800 7FFF）

4.2.4.5. Flash WRP area address option byte

Flash address: 0x1FFF 008C

Production value: 0xFF00 00FF

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

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

Bit	Name	R/W	Function
31: 24	Reserved	-	-
23: 16	~ WRP	R	Complemented code of WRP

15: 8	Reserved	-	-
7: 0	WRP	R	0: sector [y] is protected 1: sector[y] is unprotected y=0~7

4.2.4.6. Flash option bytes

After reset, the bits associated with the option bytes in the FLASH_CR register are write-protected. The OPTLOCK bit in the FLASH_CR register must be cleared before relevant operations can be performed on the option byte (Operate in every program or erase progress).

The following steps are used to unlock the register:

1. Unlock write protection of the FLASH_CR register by unlocking timing
2. Write OPTKEY1 = 0x0819 2A3B to the FLASH_OPTKEYR register
3. Write OPTKEY2 = 0x4C5D 6E7F to the FLASH_OPTKEYR register

Any wrong sequence locks up the FLASH_CR register until the next reset. When the KEY timing sequence is wrong, a bus error state is generated and a HardFault interrupt is generated.

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

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

Modify user's option bytes

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

1. Clear the OPTLOCK bit using the previously described steps
2. Check the BSY bit to confirm that there are no ongoing Flash operations
3. Write the desired value (1 to 4 words) to the option byte register FLASH_OPTR/FLASH_SDKR/FLASH_WRPR (ignore this operation if writing is not word aligned)
4. Set OPTSTRT bit
5. Write any 32-bit data to 0x4002 2080 (trigger a formal write operation)
6. Wait for the BSY bit to be cleared.
7. Wait for the EOP to rise, the software clears

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

Reload option bytes

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

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

- When the OBL_LAUNCH bit in the FLASH_CR register is set
- After power-on reset (POR/PDR/BOR)

The operation performed by load option byte is to read the option byte in the information memory area, and then store the read data in the internal option register (FLASH_OPTR, FLASH_SDKR and FLASH_WRPR). These internal registers configure the system and can be read by software. Setting the OBL_LAUNCH bit generates a reset, so that the loading of option bytes can be performed under the system reset.

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

If the word and its complement are matching, the option byte is copied into the option register.

If the word and its complement are not matching, the OPTVERR status bit of the FLASH_SR register is set. The option register maintains its default value:

- For user options
  - BOR_LEV is written as 000 (lowest threshold)
  - The BOR_EN bit is written as 0 (BOR off)
  - The NRST_MODE bit is written as 0 (reset input only)
  - The RDP bit is written as 0xff (i.e. level 1)
  - The rest of the mismatched values are written as 1
- For the SDK area option, SDKR_STRT [3: 0] = 0x0, SDKR_END [3: 0] = 0xF, that is, all flash space is set to SDK
- For Flash boot control option
  - nBOOT1, BOOT0 bits are written as 00 (that is, Main flash is selected as the boot area)
  - The BOOT_SIZE bit is written as 0 (i.e., no Load flash area)
- For the WRP option, the mismatched value is the default unprotected
- After the system is reset, the contents of option bytes are copied to the following option register (software readable and writeable):
  - FLASH_OPTR
  - FLASH_SDKR
  - FLASH_BTCR
  - FLASH_WRPR

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

4.3. Flash option bytes

A partial section (2 pages in total) of the Flash information area in the device is used as the Factory config. byte uses.

Configure bytes to store information for software to read:

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

- V_{REFBUF} Trimming values corresponding to different output voltages

Table 4-4 Byte information configuration

Page	Word	Address	Description
2	0	0x1FFF 0100	Stores HSI 24 MHz frequency selection control and corresponding Trimming values
	1	0x1FFF 0104	Stores HSI 48 MHz frequency selection control and corresponding Trimming values
	2	0x1FFF 0108	Reserved
	3	0x1FFF 010C	Reserved
	4	0x1FFF 0110	Reserved
	5	0x1FFF 0114	Normal TS Data
	6	0x1FFF 0118	High TS Data
	7	0x1FFF 011C	Store the configuration values of the corresponding FLASH_TS0, FLASH_TS1 and FLASH_TS3 registers at the frequency of HSI 24 MHz
	8	0x1FFF 0120	Store the configuration values of corresponding FLASH_TS2P and FLASH_TPS3 registers at HSI 24 MHz frequency
	9	0x1FFF 0124	Stores the configuration value of the corresponding FLASH_PERTPE register at the frequency of HSI 24 MHz
	10	0x1FFF 0128	Stores the configuration value of the corresponding FLASH_SMERTPE register at the frequency of HSI 24 MHz
	11	0x1FFF 012C	Store the configuration values of the corresponding FLASH_PRGTPE and FLASH_PRETPE registers at HSI 24 MHz frequency
	12	0x1FFF 0130	Store the configuration values of the corresponding FLASH_TS0, FLASH_TS1 and FLASH_TS3 registers at the frequency of HSI 48 MHz
	13	0x1FFF 0134	Store the configuration values of the corresponding FLASH_TS2P and FLASH_TPS3 registers at HSI 48 MHz frequency
	14	0x1FFF 0138	Stores the configuration value of the corresponding FLASH_PERTPE register at the frequency of HSI 48 MHz
	15	0x1FFF 013C	Stores the configuration value of the corresponding FLASH_SMERTPE register at the frequency of HSI 48 MHz
	16	0x1FFF 0140	Store the configuration values of the corresponding FLASH_PRGTPE and FLASH_PRETPE registers at HSI 48 MHz frequency
	17	0x1FFF 0144	Store the Trimming value corresponding to the LSI 32.768 kHz frequency
	18	0x1FFF 0148-0x1FFF 017C	Reserved
3	0	0x1FFF 0180-0x1FFF 0FFF	Reserved

4.3.1. HSI_TRIMMING_FOR_USER

Address: 0x1FFF 0100 (24 MHz)/0x1FFF 0104 (48 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.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSI_FS[2:0]			HSI_TRIM[12:0]												
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

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

4.3.2. FLASH_SLEEPTIME_CONFIG

Address: 0x1FFF 0114

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

The software needs to read the data from this address and then write it to the corresponding [15: 8] of the FLASH_STCR register.

4.3.3. HSI_24M/48M_EPPARA0

Address: 0x1FFF 011C (24 MHz), 0x1FFF 0130 (48 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TS1 [9:0]										TS3 [8:7]	
				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**
TS3 [6:0]							TS0 [8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

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

4.3.4. HSI_24M/48M_EPPARA1

Address: 0x1FFF 0120 (24 MHz), 0x1FFF 0134 (48 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TPS3 [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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P [8:0]								
							R	R	R	R	R	R	R	R	R

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

4.3.5. HSI_24M/48M_EPPARA2

Address: 0x1FFF 0124 (24 MHz), 0x1FFF 0138 (48 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	PERTPE[17:16]	
														R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

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

4.3.6. HSI_24M/48M_EPPARA3

Address: 0x1FFF 0128 (24 MHz), 0x1FFF 013C (48 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.	SMERTPE [17:16]	
														R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

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

4.3.7. HSI_24M/48M_EPPARA4

Address: 0x1FFF 012C (24 MHz), 0x1FFF 0140 (48 MHz)

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

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

4.3.8. LSI_32.768K_TRIMMING

Address: 0x1FFF 0144 (32.768 kHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res	Res	Res.	Res.	Res	Res.	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	LSI_TRIM[8:0]								
							R	R	R	R	R	R	R	R	R

The software needs to read the data from this address and then write it to the corresponding LSI_TRIM [8: 0] of the RCC_ICSCR register to change the LSI frequency.

4.4. Flash USER OTP memory bytes

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

Table 4-5 USER OTP memory bytes organization

Page	Word	Address	Contents
5	0	0x1FFF 0280	Bit[31:16]: store user data Bit[15:0]: USER OTP MEMORY_LOCK
	1	0x1FFF 0284	Store user data
	2	0x1FFF 0288	Store user data
	…	…	Store user data
	…	…	Store user data
	…	…	Store user data
	31	0x1FFF 02FC	Store user data

This Page is configured in the information area, and the program and erase of this Page area are processed according to the method of Main flash. In addition, the mass erase of the Main flash area is not valid for this area.

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

This Page Write has the following protection.

Table 4-6 Flash USER OTP memory bytes write protection status

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

4.5. Flash protection

The protection of Main flash area includes the following mechanisms:

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

4.5.1. SDK area protection

The protection of Main flash area includes the following mechanisms:

Starting address

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

End address

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

When SDK protection is valid, mer cannot be used for mass erase. If the trigger address is located in the SDK area, a HardFault will be generated. If the trigger address is not located in the SDK area, the wrperr error flag will be set; When using ser for sector erase, if the trigger address is located in the sdk area, a HardFault will be generated. If the trigger address is not located in the sdk area, but the sector contains the sdk area, the wrperr error flag will be set.

When the protection is effective, when the FLASH_SDKR register is unprotected (writing SDK_STRT [3: 0] is greater than SDK_END [3: 0]), the hardware will first trigger mass erase (the protected program in the SDK area has been written before, and through mass erase It plays the role of protecting the SDK area program), and then updates the value of the SDK option in the flash option byte (the updated value at this time means that the SDK protection is invalid). The mass erase generated by the SDK protection rewrite will also erase the Load flash area.

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

4.5.2. Flash memory read protection

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

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

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

Table 4-7 Flash read protection status

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

Level 0: Unprotected

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

Level 1: read protection

If the RDP and its complement in the option byte contain any combination other than (0xAA, 0x55), Level 1 read protection takes effect, and Level 1 is the default protection level.

- User mode: A program executed in User mode (started from Main flash), which can do all operations on Main flash, option byte.
- Debug, boot from SRAM: In debug mode, or when booted from SRAM, the Main flash is not read-accessible. In these modes, a read or write access to the Main flash generates a bus error and a HardFault interrupt.

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

Table 4-8 Relationship of access status to protection level and execution mode

Area	READ Protection level	SDK Area Protection level	Boot From Main flash (CPU) Boot From Load flash						Debug/ excuted From RAM		
			User execution (From Non SDK Area)			User execution (From SDK Area)					
			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
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SDK Area		Disable	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
Non SDK Area	1	Disable	Yes	Yes	Yes	N/A	N/A	N/A	No	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
SDK Area		Disable	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
option bytes area	x	Disable	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Factory Bytes	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No
UID	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No

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

4.5.3. Flash write protection (WRP)

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

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

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

Note: Write protection only works on Main flash.

4.5.4. Load flash area protection

When the Load flash is active, the erase and write operation of the selected area is ignored, and the WRPRTERR bit of the FLASH_CR register is also set.

Modify BOOT_SIZE of FLASH_BTCR, and the hardware will perform a mass erase operation on the Main flash. The mass erase generated by rewriting will also erase the Load flash area.

4.5.5. Option byte write protection

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

4.6. Flash interrupt

Table 4-9 Flash interrupt request

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

Note: The following events do not have separate interrupt flags but generate a Hard fault:

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

4.7. Flash registers

4.7.1. Flash access control register (FLASH_ACR)

Address offset:0x00

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	-
1:0	LATENCY	RW	0	Waiting state corresponding to Flash read operation: 0: No waiting state for Flash read operation (AHB clock at 24 MHz and below) 01: The Flash read operation has one waiting state, that is, each Flash read requires two AHB clock cycles (the AHB clock is at 48 MHz) 10: Reserved 11: Reserved

4.7.2. Flash key register (Flash_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

All register bits are write-only, readout returns 0.

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

Bit	Name	R/W	Reset Value	Function
31: 0	KEY[31:0]	W	32'h0	The following values must be written consecutively to unlock the FLASH_CR register and allow Flash's program/erase operation KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.7.3. Flash option key register (Flash_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

All register bits are write-only, readout returns 0.

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

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

4.7.4. 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**

OPTV ERR		USRLOC K	Res .	Res .	Res .	Res .	Res .	Res .	Res .	Res .	WRP ERR	Res .	Res .	Res .	EOP
RC_W 1		R									RC_W 1				RC_W 1

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	BSY	R	0	Busy bit This bit indicates that the operation of the flash is in progress. This bit is set by the hardware at the beginning of the flash operation, and is cleared by the hardware when the operation is completed or an error occurs.
15	OPTVERR	RC_W1	0	Option and trimming bits loading validity error When the option and trimming bits and their complement do not match, the hardware sets this bit. Loading mismatched option bytes is forced to a safe value. Refer to section Flash option byte programming. This bit is cleared by writing 1.
14	Reserved	-	-	-
13	USRLOCK	R	0	Whether the userdata area is writable is indicated according to the value of the lower 16 bits of the first word of the power-on read userdata area 0: The value read is not 0xaa55, userdata is writable 1: The value read is 0xaa55, userdata is not writable This bit can only be reset by a power-on reset
12: 5	Reserved	-	-	-
4	WRPERR	RC_W1	0	Write protection error When the address to be programmed/erase is in the write-protected Flash area (WRP), the hardware sets this bit. Write 1 and clear the bit.
3:1	Reserved	-	-	-
0	EOP	RC_W1	0	When the program/erase operation of Flash is successfully completed, the hardware is set. This bit is set only if the EOPIE bit of the FLASH_CR register is enabled. Write 1 and clear the 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.	OBLLAUNCH	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	0	FLASH_CR Lock bit. The software can only set this bit. When set, the FLASH_CR register is locked. When the unlocking timing is successfully given, this bit is cleared by hardware, unlocking the FLASH_CR register. [This bit is set by software after the program/erase operation is completed] When an unsuccessful unlock timing is given, the bit remains set until the next system reset.
30	OPTLOCK	RS	0	Option bytes Lock bit.

				The software can only set this bit. When set, the bits associated with option bytes in the FLASH_CR register are locked. When the unlocking timing is successfully given, this bit is cleared by hardware, unlocking the FLASH_CR register. [This bit is set by software after the program/erase operation is completed] When an unsuccessful unlock timing is given, the bit remains set until the next system reset.
29: 28	Reserved	-	-	-
27	OBL_LAUNCH	RC_W1	0	Force the option bytes loading. When set, this bit forces the system to reload option bytes. This bit is cleared by hardware only when the option byte load is completed. If the OPTLOCK bit is set, the bit cannot be written. 0: Option byte loading completed 1: Generate an option byte loading request, and the system generates a reset to reload the option byte.
26	Reserved	-	-	-
25	ERRIE	RW	0	Error interrupt enable bit, when the WRPERR bit of the FLASH_SR register is set, if this bit is enabled, an interrupt request is generated. 0: No interrupt occurrence 1: An interrupt occurs
24	EOPIE	RW	0	End of operation interrupt enable When the EOP bit of the FLASH_SR register is set, if the bit is enabled, an interrupt request is generated. 0: EOP interrupt disabled 1: EOP interrupt enabled
23: 20	Reserved	-	-	-
19	PGSTRT	RW	0	The start bit of the program operation for the Flash main memory. This bit starts the program operation of the Flash main memory, is set by software, and after the BSY bit of the FLASH_SR register is cleared, the hardware clears this bit.
18	Reserved	-	-	-
17	OPTSTRT	RW	0	Start of modification of option bytes This bit initiates the modification of option bytes. Software set, after the BSY bit of the FLASH_SR register is cleared, hardware clears this bit. Note: When the Flash option bytes are modified, the hardware automatically performs an erase operation on the entire 128 Bytes page, and then performs a program operation, which also includes automatic write of complemented code.
16:12	Reserved	-	-	-
11	SER	RW	0	4 KB Sector erase operations 0: Sector erase operation for flash is not selected 1: Sector erase operation for flash selected Notes: 1. Sector erase does not work on Flash information memory. 2. Sector erase does not work for areas set to WRP.
10: 3	Reserved	-	-	-
2	MER	RW	0	Mass erase operation 0: Mass erase operation for Flash not selected 1: Mass erases operation of Flash selected Notes: Mass erase will not work on Flash information memory. Mass erase does not work when there is a WRP setting
1	PER	RW	0	Page erase operation 0: Page erase operation for flash is not selected 1: Page erase operation of flash selected
0	PG	RW	0	Program Operation 0: program operation of Flash is not selected

				1: program operation of Flash selected

4.7.6. Flash option register (FLASH_OPTR)

Address offset: 0x20

Reset value: 0x0000 90AA

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

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

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	IWDG_STOP	RW		Set the running state of IWDG timer in Stop mode 0: Freeze timer 1: Normal operation
14	NRST_MODE	RW		0: Reset pin used as NRST 1: Reset pin used as normal GPIO
13	Reserved	-	-	-
12	IWDG_SW	RW		0: Hardware watchdog 1: Software watchdog
11: 9	BOR_LEV[2:0]	RW		001: BOR rising threshold is 1.99 V and falling threshold is 1.88 V 010: BOR rising threshold is 2.19 V and falling threshold is 2.10 V 011: BOR rising threshold is 2.39 V and falling threshold is 2.30 V 100: BOR rising threshold is 2.78 V and falling threshold is 2.69 V 101: BOR rising threshold is 3.08 V and falling threshold is 2.99 V 110: BOR rising threshold is 3.68 V and falling threshold is 3.58 V 111: BOR rising threshold is 4.20 V and falling threshold is 4.08 V
8	BOR_EN	RW		BOR enable 0: BOR OFF 1: BOR enabled, BOR_LEV works
7: 0	RDP	RW		0xAA: level 0, read protection inactive Others: level 1, read protection active

4.7.7. Flash SDK address register (FLASH_SDKR)

Address offset: 0x24

Reset value: 0x0000 000F

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SA_END[3:0]				Res.	Res.	Res.	Res.	SA_STRT [3: 0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	-
11: 8	SDK_END[3:0]	RW	-	SDK area end addres, the corresponding STEP of each bit is 2 KB
7: 4	Reserved	-	-	-
3: 0	SDK_STRT[3:0]	RW	-	SDK area start address, the corresponding STEP of each bit is 2 KB

4.7.8. Flash boot control (FLASH_BTCR)

Address offset: 0x28

Reset value: 0x0000 8000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nBOOT1	BOOT0	Res.	Res.	Res.	Res.	SWD_MODE[1:0]		Res.	Res.	Res.	Res.	Res.	BOOT_SIZE [2:0]		
RW	RW	-	-	-	-	RW	RW	-	-	-	-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	-
15	NBOOT1	RW	0	Boot mode of nBOOT1 / BOOT0 X0: Boot from Main Flash 11: Boot from Load Flash 01: Boot from SRAM Note: Products with 20 KB Flash cannot boot from Load flash.
14	BOOT0			
13:10	Reserved	-	-	-
9:8	SWD_MODE[1:0]	RW	0	Select the IO port as GPIO mode or SWD mode PF3 PF4 PA13 PA14 00: SWCLK SWDIO GPIO GPIO 01: GPIO GPIO SWDIO SWCLK 10: GPIO SWDIO GPIO SWCLK 11: SWCLK GPIO SWDIO GPIO
7:3	Reserved	-	-	-
2: 0	BOOT_SIZE [2:0]	RW	0	Select Main flash block area as Load flash 000: No Load flash area 001: 1 KB（0x0800 7C00~0x0800 7FFF） 010: 2 KB（0x0800 7800~0x0800 7FFF） 011: 3 KB（0x0800 7400~0x0800 7FFF） 1xx: 4 KB（0x0800 7000~0x0800 7FFF） Note: This bit is reserved for models with 20K Flash capacity

4.7.9. Fash WRP address register (FLASH_WRPR)

Address offset: 0x2C

Reset value: 0x0000 00FF

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

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	WRP[7]	RW	1	0: sector 7, with write protection, program and erase are not allowed 1: sector 7 is unprotected
6	WRP[6]	RW	1	0: sector 6, with write protection, program and erase are not allowed 1: sector 6 is unprotected
5	WRP[5]	RW	1	0: sector 5, with write protection, program and erase are not allowed 1: sector 5 is unprotected
4	WRP[4]	RW	1	0: sector 4, with write protection, program and erase are not allowed 1: sector 4 is unprotected
3	WRP[3]	RW	1	0: sector 3, with write protection, program and erase are not allowed 1: sector 3 is unprotected
2	WRP[2]	RW	1	0: sector 2, with write protection, program and erase are not allowed 1: sector 2 is unprotected
1	WRP[1]	RW	1	0: sector 1, with write protection, program and erase are not allowed 1: sector 1 is unprotected
0	WRP[0]	RW	1	0: sector 0, with write protection, program and erase are not allowed 1: sector 0 is unprotected

4.7.10. 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								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
31:16	Reserved	-	-	Reserved
15:8	SLEEP_TIME	RW	0x50	When the system clock selects LSI or LSE, in order to obtain a more optimized operating mode power consumption, the function of using this register can be selected (this function is only recommended when LSI or LSE is the system clock).

				When this feature is enabled, the time width for which Flash is in Sleep mode for every half of the system clock low cycle is: t_{HSI_10M} * SLEEP_TIME Notes: t_{HSI_10M} is the period of HSI_10M; To ensure the function of Flash, the maximum value of this register is recommended to be set to 0x1E.
7:1	Reserved	-	-	Reserved
0	SLEEP_EN	RW	0	Flash Sleep mode enabled 1: Flash Sleep enabled 0: Flash Sleep disabled

4.7.11. 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.	TS0								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS0	RW	-	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 011C 48 MHz calibration value storage address: 0x1FFF 0130

4.7.12. 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.	TS1 [9:0]									
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:0	TS1	RW	0x1XX	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 011C 48 MHz calibration value storage address: 0x1FFF 0130

4.7.13. 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	TS2P								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS2P	RW	0xB4	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0120 48MHz calibration value storage address: 0x1FFF 0134

4.7.14. 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.	TPS3											
				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	TPS3	RW	0x6C0	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0120 48MHz calibration value storage address: 0x1FFF 0134

4.7.15. 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	TS3								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	TS3	RW	0xXX	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 011C 48 MHz calibration value storage address: 0x1FFF 0130

4.7.16. 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.	PERTPE	
														RW	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
31:18	Reserved	-	-	Reserved
17:0	PERTPE	RW	0x14820	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 0124 48 MHz calibration value storage address: 0x1FFF 0138

4.7.17. 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.	SMERTPE	
														RW	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
31:18	Reserved	-	-	Reserved
17:0	SMERTPE	RW	0x14820	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24MHz calibration value storage address: 0x1FFF 0128 48MHz calibration value storage address: 0x1FFF 013C

4.7.18. 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	-	-	Reserved
15:0	PRGTPE	RW	0x5DC0	The software reads out the data stored in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the

				erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 012C 48 MHz calibration value storage address: 0x1FFF 0140

4.7.19. 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 in the corresponding address in the information area and writes it to the corresponding register, so as to realize the configuration of the erasing and writing time required for the corresponding HSI frequency. Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 012C 48 MHz calibration value storage address: 0x1FFF 0140

5. Power control

5.1. Power supply overview

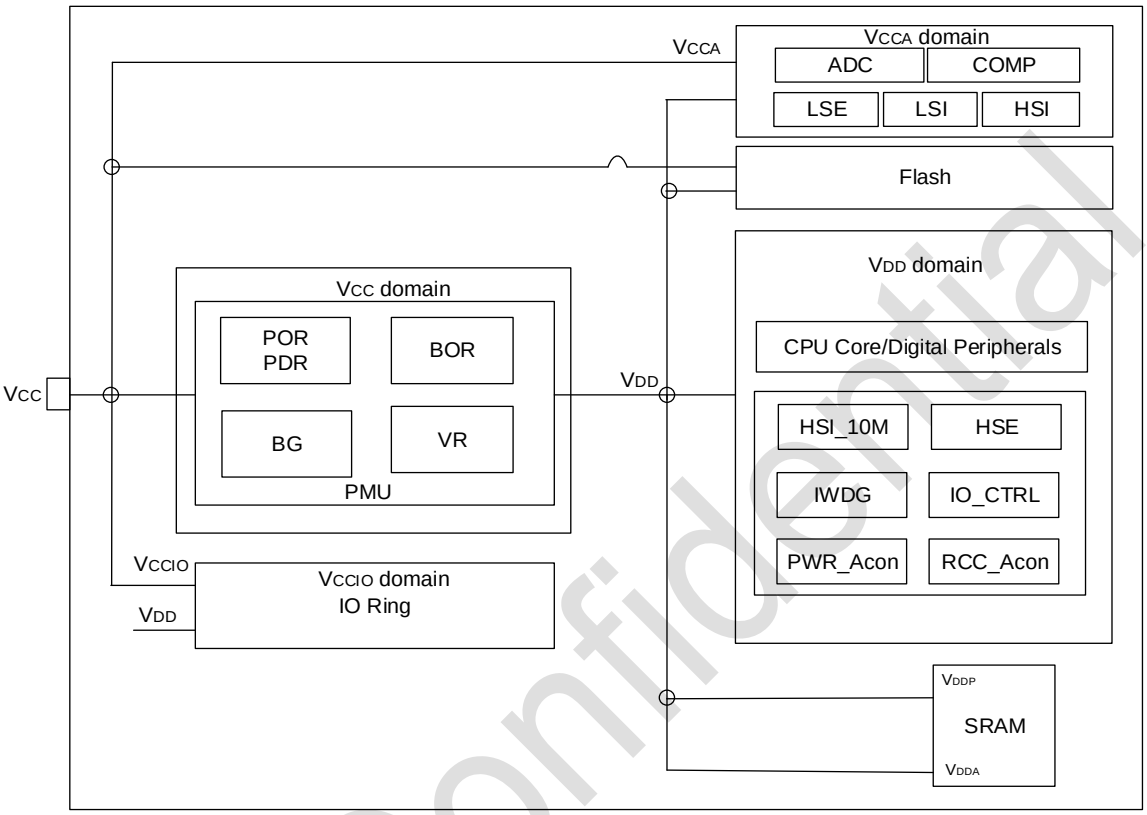


Figure 5-1 Power supply overview

Table 5-1 Power supply overview

No.	Power supply	Power value	Description
1	V_{CC}	1.8 - 5.5 V	Power supply range: 1.8 to 5.5 V. The power is supplied to the device through the power pins, with the power supply module comprising: Partial analog circuits.
2	V_{CCA}	1.8 - 5.5 V	Powers for most analog modules, sourced from the V_{CC} PAD (a dedicated power PAD can also be designed separately).
3	V_{CCIO}	1.8 - 5.5 V	Power to IO from V_{CC} PAD
4	V_{DDx} (V_{DD}/V_{DD1})	1.2 V to 0.8 V ± 10% (V1) 1.2 V ~ 1.0 V ± 10% (V1B)	VR supplies power to the main logic circuits and SRAM inside the chip. When the MR is powered, it outputs 1.2 V. When entering the Stop mode, power can be supplied by the MR or the LPR according to the software configuration.

5.2. Voltage regulator

The regulator has two operating modes:

- MR(Main regulator) is used in Run mode.
- LPR (Low power regulator) provides an option for even lower power consumption in Stop mode.

The power supply of the V_{DD} comes from MR or LPR depending on the operating mode of the device.

In Run mode (normal running state):

The MR remains working, outputting 1.2 V voltage and the LPR is off.

The software can determine whether the LDO works in MR mode or Stop/Sleep mode through configuration. In low power mode, the BOR operates or does not operate depending on the power-on load or register configuration; The Stop mode involves the wake-up function, that is, the power supply of VDD must be switched from LPR to MR, so in addition to the extremely low power consumption of LPR, the switching time is also an important design index.

The system requires that the total wake-up time of stop mode (including LPR switching to MR, HSI wake-up, Flash wake-up, CPU wake-up) should be less than 6.5 us.

5.3. Power monitoring

5.3.1. Power-on reset (POR)/power-down reset (PDR)

The device contains Power-on reset (POR) and Brown-out reset (BOR) modules to provide power-related reset. Where POR works by default in all power modes, BOR needs to work after being enabled. POR/BOR low is reset.

After the BOR is enabled, the POR will not be released immediately after reaching the VPOR threshold, and the VDDD needs to reach the threshold specified by VBORFx. (The value of VBORF is defined in the option byte).

5.3.2. Brown-out reset (BOR)

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

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

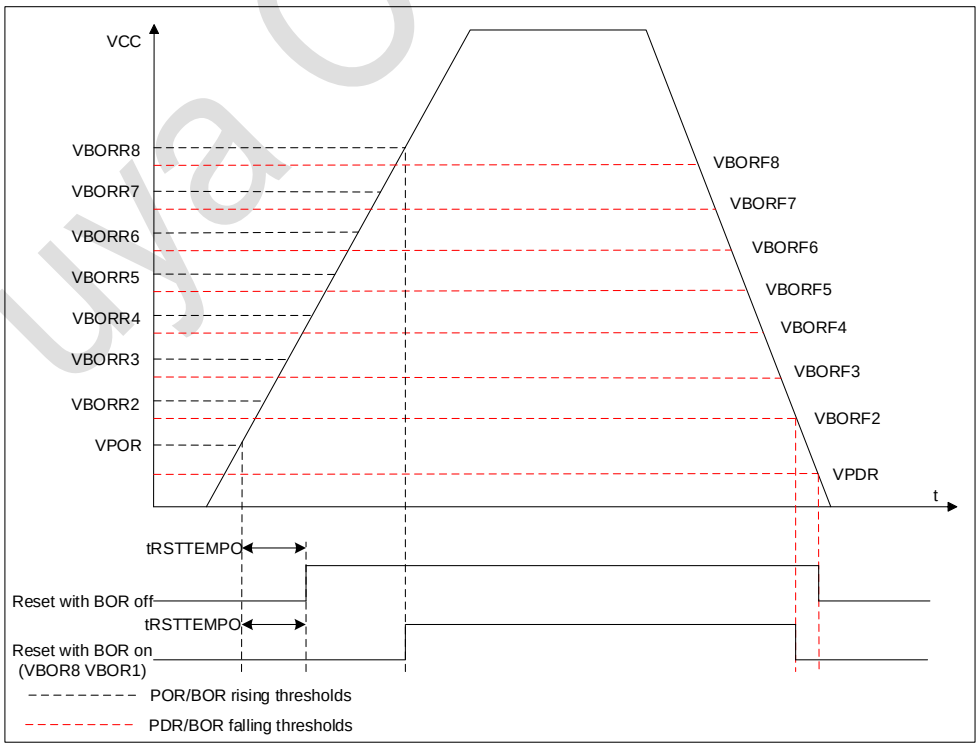


Figure 5-2 POR/PDR/BOR threshold

5.4. Low-power modes

In addition to the Run mode, the device has 2 low-power modes:

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

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

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

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

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

To sum up, the low power mode conversion diagram is as follows.

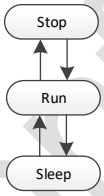


Figure 5-3 Low power mode conversion diagram

5.4.1. Run mode

In Run mode, to reduce power consumption, you can:

- Reduce system clock (SYSCLK/HCLK/PCLK) frequency
- Turn on only clocks that require peripherals.

5.4.2. CPU low-power modes

- **Entry**

The CPU enters low power mode in 3 ways:

- WFI
- WFE
- The Cortex-M0 + register SLEEPONEXIT is configured to 1 and returned from ISR

- **Wake-up**

For low power mode where V_{DDD} voltage exists, wake-up mode:

- In the low power mode entered after WFI or ISR returns, any NVIC-enabled external interrupt will wake up;
- The low power mode entered by the WFE wakes up when a wake-up event is generated.
- NVIC IRQ interrupt

When CPU control register SEVONPEND = 0: enable peripheral interrupt register + enable CPU NVIC register; After recovery from WFE, both peripheral interrupt pending and NVIC peripheral IRQ channel pending bits must be cleared;

When CPU control register SEVONPEND = 1: Enable peripheral interrupt register + CPU NVIC register (whether enabled or not); When recovering from the WFE, both the peripheral interrupt pending and the NVIC peripheral IRQ channel pending bit (if enabled) must be cleared.

- **Event**

Configure the event EXTI. When the CPU enters low power mode from WFE, there is no need to clear the EXTI peripheral interrupt pending or NVIC IRQ channel pending bit.

5.4.3. Sleep mode

In Sleep mode, the CPU clock is turned off (NVIC and SysTick work), all peripherals can be configured to work (it is recommended to enable only modules that must work), and peripheral interrupts or events can wake up the CPU.

1. The hardware only turns off the CPU Core clock, and the module clock is configured by the software whether to turn off
2. The analog PMU operates in MR mode (PWR_CR1. LPR [1: 0] = 2 'b00) with$_{\text{a digital VDD voltage}}$ of 1.2 V

In Sleep mode, the state of the IO pin is the same as in Run mode.

The selected system clock source always exists.

- **Entry**
  - SLEEPDEEP = 0, and there is no interrupt or event pending, then it enters through a WFE or WFI instruction;
  - SLEEPDEEP = 0, and SLEEPONEXIT = 1. When there is no interrupt pending, it is entered after returning from ISR (the lowest priority interrupt);
- **Wake-up**
  - WFI or ISR return: Enable interrupt;
  - WFE, and SEVONPEND = 0: wakeup event;
  - WFE, and SEVONPEND = 1: interrupt (whether enabled or not), or wake-up event;

Note: The CPU interface does not have SEVONPEND signal output, so the PWR module will not distinguish.

- **Wake up latency**

There is no extra wait time for waking up.

- **Hardware Implementation**

In Sleep mode, the hardware implementation follows the following principles:

- Controlling to close HCLK/PCLK according to the SLEEPING signal;
- Whether the peripheral IP clock is turned off is configured by the software before entering the Sleep mode, and the hardware will not actively turn off the peripheral clock;
- The clock selection is controlled by software, and the hardware will not control the system clock frequency and the selection of clock source;

5.4.4. Stop mode

The Stop mode is based on the CPU's DEEPSLEEP mode, as well as peripheral clock gating.

Both module clocks in the VDD domain are off, and HSI and HSE are off. The LSI and the LSE can be kept running.

RTC and TAMP keep working. Some peripherals with wake-up capabilities (such as TK/I²C) can be converted to HSI clocks to receive data. The HSI only responds to peripheral requests at this time.

Stop mode:

- Hardware turns off high-speed HSE, HSI clock
- The simulated PMU operates in LPR mode (PWR_CR1. LPR [1: 0] = 2 'b01).
- When PWR_CR1.SRAM_RETVCTRL [1: 0] = 2'b00, the SRAM power supply V_{DDP} is derived from V_{DDD}
- When PWR_CR1.SRAM_RETVCTRL [1: 0] = 2'b01, the SRAM power supply V_{DDA} is derived from V_{DDD}, and the SRAM power supply V_{DDP} is derived from V_{DDD}.

Stop mode, Main-VR can be configured to turn off.

After exiting the Stop mode, the system clock is HSISYS.

- **Entry**
  - SLEEPDEEP = 1, and there is no interrupt or event pending, then it is entered by WFE or WFI instruction
  - SLEEPDEEP = 1, and SLEEPONEXIT = 1. When there is no interrupt pending, enter after returning from ISR
- **Wake-up**
  - WFI or ISR returns: EXTI enabled in interrupt mode (the corresponding EXTI interrupt vector must be enabled at NVIC). The source of the interrupt may be an external interrupt or an interrupt of a peripheral with wake-up capability
  - WFE, and SEVONPEND = 0: EXTI enabled to event mode
  - WFE, and SEVONPEND = 1: EXTI enabled in interrupt mode (even if the corresponding EXTI interrupt vector is not enabled)
  - Reset (POR/PinRST/IWDG reset/LSECSS)
- **Wake up latency**

  The awakening latecny depends on the maximum duration of the following three:
  - Wake-up time of HSI clock
  - Flash wakeup time
  - Main-VR wake-up stabilization time
  - The system requires wake-up time: within 5 us

5.4.5. Functionalities depending on the working mode

Before the configuration enters stop mode, the software clears the interrupt pending registers EXTI_RPR1 and EXTI_FPR1;

To enter the stop mode, the hardware needs to turn off the HSI/HSE analog clock source;

Whether Main-VR job or LP-VR job is configured by software;

Wake-up source:

- GPIO 0 ~ 15
- COMP1
- COMP2
- RTC
- IWDG reset
- POR reset
- nRST pin reset
- LSECSS
- I^2C (I^2C wake-up enters the I^2C address matching stage. If other wake-ups come (direct wake-up sources such as RTC/GPIO), you can directly wake up Stop mode)
- TK (TK is used as the wake-up source, except that Sleep can interrupt the wake-up, other mode wake-up processes are RTC wake-up PWR, PWR wake-up TK, TK wake-up Stop mode, TK as the wake-up source HSION_CTRL must be 0)

Table 5-2 Functions in each operating mode

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wakeup ability
CPU Core	O	-	-	-
Flash memory	O	O	-	-
SRAM	O	O	-	-
Brown-out reset (BOR)	O	O	O	O
HSI	O	O	-	-
HSE	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
UART1/UART2/UART3	O	O	-	-
I2C	O	O	O	O
SPI	O	O	-	-
ADC	O	O	-	-
COMP1/COMP2	O	O	O(1)	O
OPA1	O	O	-	-
Temperature sensor	O	O	-	-
Timers(TIM1/TIM14)	O	O	-	-
IWDG	O	O	O	O
SysTick timer	O	O	-	-
CRC	O	O	-	-
TK	O	O	(1)	O (priority RTC wakes up TK, then TK wakes up Stop)
GPIOs	O	O	O	O

1. COMP1, COMP2, and TK modules, only work when VR @ MR.

5.5. Power control registers

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

5.5.1. Power control register 1 (PWR_CR1)

Address offset:0x00

Reset Value : 0x0002 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	SRAM_RETV_CTRL		Res.
												RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPR[1:0]		FLS_SLP-TIME[1:0]		Res.	Res.		DBP	Res.							
RW		RW					RW								

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	HSION_CTRL	RW	0	When waking up from Stop mode, the HSI turns on time control. 0: After waiting for MR to stabilize, enable HSI; 1: Turn on at the same time as VR, that is, HSI is enabled immediately when waking up.
18:17	SRAM_RETV_CTRL[1:0]	RW	2'b01	00/01: Normal voltage selection; 10: Voltage selection in Stop mode 11: Reserved
16	Reserved	-	-	Reserved
15:14	LPR[1:0]	RW	2'b00	Low power regulator 00: MR (Main regulator) 01: LPR (Low power regulator) 10: Reserved 11: Reserved Note: After the Stop mode wakes up, the hardware updates this register to 00.
13:12	FLS_SLPTIME	RW	2'b00	In the Stop mode wake-up timing, after the HSI is stabilized, a waiting time is required before the Flash operation. 2'b00: 5 μs 2'b01: 2 μs 2'b10: 3 μs 2'b11: 0 μs Note: When this register is set to 2'b11/2'b01, it indicates that the program is executed from SRAM after wakeup, not Flash. And the program guarantees that Flash will not be accessed within 3μs after waking up the execution program.
11:9	Reserved	-	-	Reserved
8	DBP	RW	0	RTC write protection disabled After reset, the RTC is write-protected to prevent accidental writes. To access RTC this bit must be set to 1. 0: Access to RTC disabled 1: Access to RTC abled
7:0	Reserved	-	-	Reserved

6. Reset

Two resets are designed: power reset and system reset.

6.1. Reset source

6.1.1. Power reset

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

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

Power reset includes the following:

- The POR/BOR generated by the analog circuit realizes the detection of V_{CC}. Releasing the reset when the V_{CC} voltage rises to the trigger value; A reset is generated when the V_{CC} voltage drops to a certain trigger value;
- The PORI generated by the analog circuit realizes the detection of VR output. Releasing the reset when the VDD voltage rises to the trigger value; A reset is generated when the VDD voltage drops to a certain trigger value;

6.1.2. System reset

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

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

The reset source can be identified by checking the reset identification bit of the RCC_CSR register.

Note: When entering FBIST mode, the CPU will be reset.

6.1.2.1. NRST pin (external reset)

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

- Reset input

Input: After the NRST pin is input, an external reset of the device is generated after passing through the deburring circuit (deburring can be configured to disable).

- GPIO

At this time, NRST can be used as a standard GPIO, not as a reset function. Resets are generated internally, and internally generated resets are not output through this pin.

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

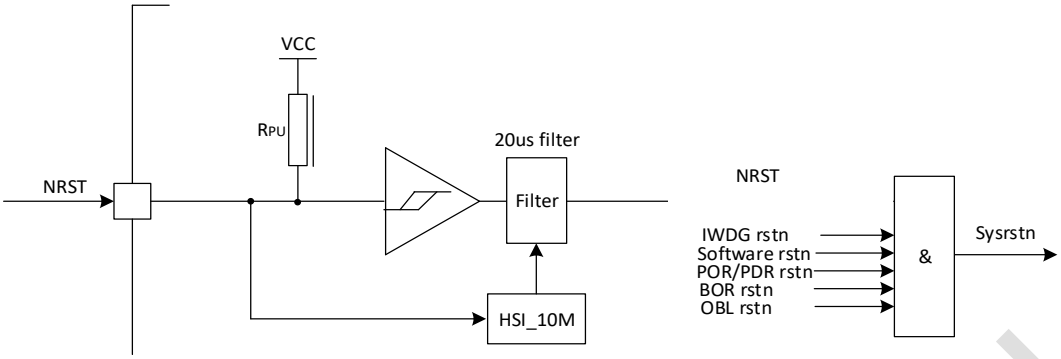


Figure 6-1 Simplified diagram of the reset circuit

6.1.2.2. Watchdog reset

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

6.1.2.3. Software reset

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

6.1.2.4. Option byte load reset

When the FLASH_CR.OBL_LAUNCH bit is set to 1, an option byte loader reset occurs. OBL_LAUNCH is used to load the FLASH option byte for software startup.

6.2. Reset range

The following is the reset range of each register by each reset source:

Table 6-1 Reset range

Reset type	Reset source	Reset register	Whether it is maskable
System reset	NRST pin low	All registers except the reset/clock register	×
	IWDG reset		√
	Software reset SYSRESETREQ		√
	Option byte loads soft reset		√
Power reset	PORE/PDR/BOR reset	All registers	×

6.3. Reset and system power mode

In-system resets may occur in any operating mode, and some resets can only occur in some specific operating modes. The following table shows the relationship between reset and run mode:

Table 6-2 Reset source and power mode

Reset type	Reset source	Run	Sleep	Stop
System reset	NRST pin low (V_{CC} domain)	√	√	√
	IWDG overflow	√	√	√
	Software reset SYSRESETREQ	√	×	×
	Option byte loads soft reset	√	×	×
Power reset	POR/PDR reset (V_{CC} domain)	√	√	√
	BOR reset (V_{CC} domain)	√	√	√

7. Clocks

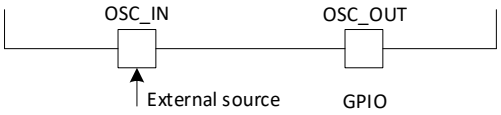
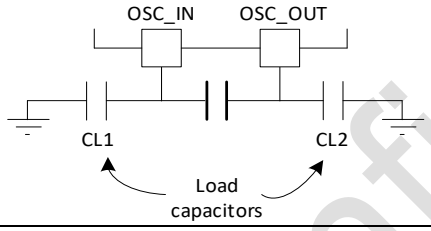
7.1. Clock sources

7.1.1. External high speed clock HSE

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

- Through external crystal oscillator and internal oscillator circuit, 4 to 8 MHz clock signal is generated.
- Input high-speed clock source directly from outside

Table 7-1 HSI clock source

Clock source	Hardware configuration
External clock	
Circumscribed crystal	

7.1.1.1. Circumscribed crystal

4 ~ 8 MHz Of the crystals have very high precision. The HSERDY flag bit of RCC_CR register indicates whether HSE is stable. The HSE may be turned on or off by the HSEON bit.

7.1.1.2. External clock source (HSE bypass)

In this mode, an external clock source is supplied. The software selects this mode via the HSEBYP and HSEON bits of RCC_CR. The external clock source will come from the OSC_IN pin, and the OSC_OUT pin is used as GPIO.

7.1.2. External low speed clock LSE

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

You can balance settling time and power consumption by configuring LSE_DRIVER.

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

- 32.768 kHz XTAL + internal vibration circuit
- External clock input via OSC_IN (LSEBYP = 1)

7.1.3. Internal high-speed clock HSI

Internal 24/48 MHz RC oscillator. Compared with XTAL OSC, RC OSC has low power consumption, shorter stabilization time, but low accuracy. The HSI analog module counts the stable time, that is, the HSI clock output to the digital is the stable clock.

After the power-on reset, the calibration and voltage/temperature trimming values of the HSI and the center frequency are loaded into the RCC_ICSCR.HSI_TRIM registerduring the loading phase.

After waking up from Stop mode, the HSI acts as the system clock source.

HSI_10M

This clock serves as a low-precision clock, used as a filtered count for nRST pins, and as low-power processing when Flash runs at low speeds.

7.1.4. Internal low speed clock LSI

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

7.2. Clock tree

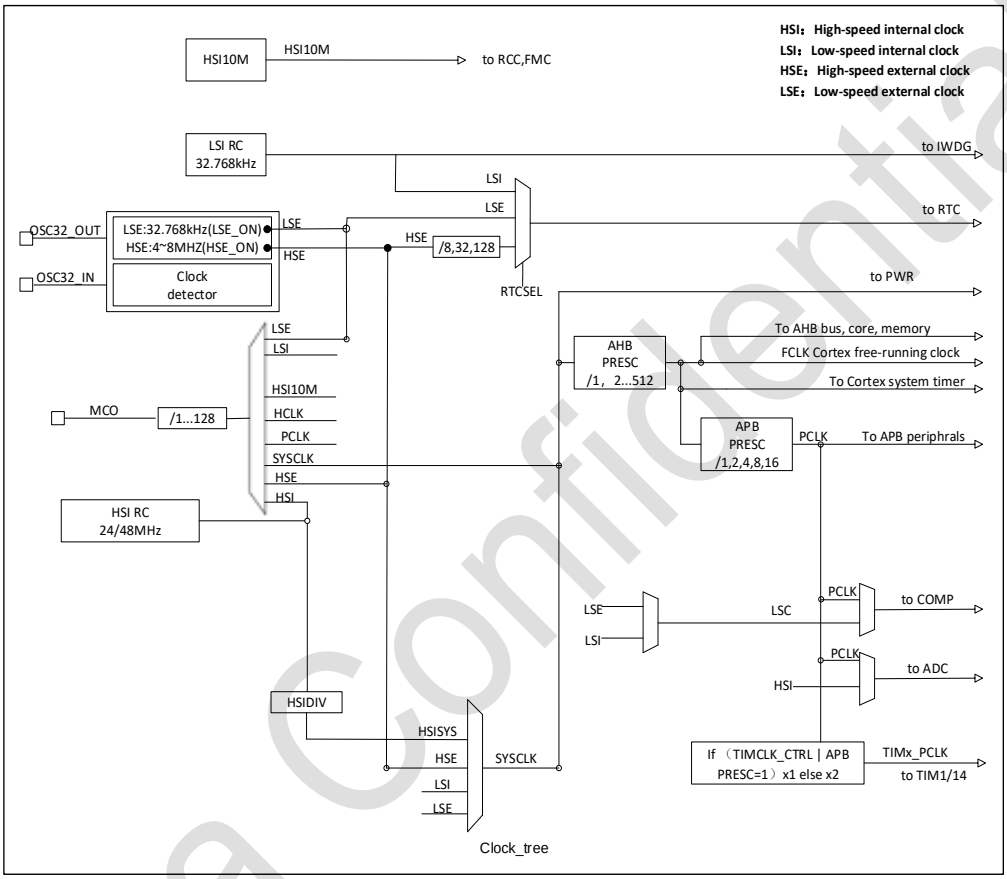


Figure 7-1 System clock structure diagram

7.3. Clock security system (CSS)

Clock security mainly includes the following aspects:

- Clock configuration and state security
- Clock source HSE security
- Clock source LSE security

7.3.1. Clock configuration and state security

The software periodically reads back the clock configuration and status registers to obtain the current clock information of the system and judge whether it is consistent with the expectation.

7.3.2. Clock source HSE monitoring

The HSE clock security system (CSS) is controlled by the RCC_CR.HSE_ CSSON register. When the HSE clock exists and stabilizes, the software can enable CSS detection. When the HSE is turned off, the CSS function is automatically turned off by the hardware.

When CSS fails:

- Turn off HSE (HSEON register is cleared by hardware). If the system clock source selects HSE at this time, switch to HSI.
- The clock failure signal is used as the brake input of TIM1 (there is a register inside the Timer to control whether this function is enabled);
- Generate an interrupt and act as an NMI interrupt. The recommended interrupt handler is as follows:
  - Write RCC_CICR.CSSC = 1, clear the HSE CSS flag register;
  - The hardware replaces the system clock source with HSI, and the software can configure the system clock source to select other clocks than HSE clocks
  - If you still select HSE as the system clock source, you need to enable HSE first (write RCC_CR.HSEON = 1)

7.3.3. Clock source LSE monitoring

The clock source LSE monitors that the reference clock is LSI. When the LSE CSS is enabled and the LSE is stable, the monitored clock counter LSECAL (cal_count) and the reference clock monitor counter LSICAL (ref_count) count simultaneously, LSICAL counts down, and LSECAL counts up. When LSICAL is decremented to 0, it is determined whether the LSECAL count overflows. If there is no overflow, it means that the LSE is operating abnormally during the monitoring time, and the LSE clock may stop or the frequency changes significantly.

When LSECSS fails:

- Turn off LSE (LSEON register cleared by hardware); Switch to LSI if the system clock selects LSE
- The clock failure signal serves as the brake input of TIM1 (there is a register inside the Timer to control whether this function is enabled)
- Generates an interrupt and acts as an NMI interrupt
  - Write RCC_CICR.LSECSSC = 1, clear LSE CSS flag register
  - The hardware replaces the system clock source with LSI, while the software can configure the system clock source to select other clocks other than LSE clocks
  - If you still select LSE as the system clock source, you need to enable LSE first (write RCC_CSR.LSEON = 1)

7.4. Reset/clock register

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

7.4.1. 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.	Res.	Res.	Res.	Res.	Res.	Res.	HSE CSSON	HSE BYP	HSE RDY	HSE ON
						R	RW					RS	RW	R	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	HSIDIV[2:0]			HSI RDY	Res.	HSION	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		RW			R		RW								

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	HSE_CSSON	RS	0	HSE clock safety system enabled. When the bit is 1, the hardware enables the clock detection module if the HSE OSC is ready; If the HSE detection fails, the clock detection module is turned off. 0: Clock safety system off (clock detection off) 1: Clock safety system is on (if HSE clock is stable, clock detection is on, otherwise clock detection is off)
18	HSEBYP	RW	0	HSE crystal oscillator masked, select pin input clock. This bit can only be written if HSEON = 0. 0: HSE crystal oscillator is not masked, external high-speed clock selects external crystal oscillator 1: HSE crystal oscillator masked, external high-speed clock select external pin input clock source
17	HSERDY	R	0	HSE crystal oscillator clock ready flag. This bit is set by hardware to indicate that the HSE crystal oscillator is stable. 0: HSE crystal oscillator is not ready 1 HSE crystal oscillator ready Note: When HSEON clears, HSERDY clears immediately
16	HSEON	RW	0	HSE clock enabled Set and cleared by software. Cleared by hardware when entering Stop mode. This bit cannot be reset if the HSE oscillator is used directly or indirectly as the system clock. 0: HSE OFF 1: HSE ON Note: HSEON and LSEON cannot be used at the same time.
15:14	Reserved	-	-	Reserved
13:11	HSIDIV[2:0]	RW	0	HSI clock division factor. The software controls these bits to set the frequency division coefficient of the HSI, generating the HSISYS clock 000: 1 001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128
10	HSIRDY	R	0	HSI clock ready flag. Set by hardware to indicate that the HSE oscillator is stable This bit is only valid if HSION = 1. 0: HSI OSC not ready 1: HSI OSC ready Once the HSEON bit is cleared, HSERDY goes low immediately.
9	Reserved	-	-	Reserved
8	HSION	RW	1	HSI clock enable Set and cleared by software. Cleared by hardware to stop the HSI oscillator when entering Stop mode. When the HSI is directly or indirectly used as the system clock (also when exiting the Stop mode, or when the HSE is used as the system clock and a failure occurs). 0: HSI OFF 1: HSI ON

Bit	Name	R/W	Reset Value	Function
7:0	Reserved	-	-	Reserved

7.4.2. Internal clock source calibration register (RCC_ICSCR)

Address offset: 0x04

Reset value: 0x0100_9100, reset by POR/PDR/BOR

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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24:16	LSI_TRIM	RW	9'h100	Internal low speed clock frequency calibration. After power-on, the hardware will write the factory information (stored in 0x1FFF 0144) into this register, so that the LSI can output an accurate frequency of 32.768 kHz. By rewriting the register value, the software increases (decreases) the output frequency of LSI by about 0.2% for every increase (decrease) of 1.
15:13	HSI_FS	RW	3'b100	HSI frequency: 100: 24 MHz 101: 48 MHz
12:0	HSI_TRIM	RW	13'h1100	Clock frequency calibration value. After power-on, the hardware uses the HSI 4 MHz register default value, and factory information will be written to this register when Trimming is waiting. The software reads out the data stored at the corresponding address in the information area and writes it to the register to realize the calibration at the specific output frequency of HSI (24/48 MHz). Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 0100 48 MHz calibration value storage address: 0x1FFF 0104 The register value can also be modified by writing the calibration value into the register, which is the center value. For every increase (decrease) of 1, the output frequency of the HSI increases (decreases) by about 0.1%.

7.4.3. Clock configuration register (RCC_CFGR)

Address offset: 0x08

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30:28	MCOPRE[2:0]	RW	0	Microcontroller clock output (MCO) frequency division coefficient. 000: 1

Bit	Name	R/W	Reset Value	Function
				001: 2 010: 4 011: 8 100: 16 101: 32 110: 64 111: 128 It is recommended to set these bits before the MCO output is enabled.
27:24	MCOSEL[3:0]	RW	0	MCO output clock selection. 0000: no clock, MCO output disabled 0001: SYSCLK 0010: HSI10M 0011: HSI 0100: HSE 0110: LSI 0111: LSE 1000: HCLK 1001: PCLK Note: The output clock may be incomplete during the clock startup or switching phase.
23:15	Reserved	-	-	Reserved
14:12	PPRE[2:0]	RW	0	APB clock division factor. PCLK is based on the division coefficient of HCLK. 0xx: 1 100: 2 101: 4 110: 8 111: 16
11:8	HPRE[3:0]	RW	0	The AHB clock HCLK is based on the frequency division coefficient of SYSCLK. 0xxx: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 64 1101: 128 1110: 256 1111: 512 To ensure the normal operation of the system, the appropriate frequency needs to be configured according to the VR power supply situation. Note: It is recommended to switch the frequency division coefficients gradually.
7:6	Reserved	-	-	Reserved
5:3	SWS[2:0]	R	0	System clock source selection. This bit is controlled by hardware and indicates the selection of the system clock source. 000: HSISYS 001: HSE 011: LSI 100: LSE Others: reserved
2:0	SW[2:0]	RW	0	System clock source selection. This bit is controlled by hardware and software and indicates the selection of the system clock source. 000: HSISYS 001: HSE 011: LSI 100: LSE Others: HSISYS Note: The system clock cannot directly switch to LSE when HSE is selected, and the system clock cannot directly switch to HSE when LSE is selected. HSISYS or LSI can be selected as the transit for switching between LSE and HSE. (For example, when the system clock selects HSE to switch

Bit	Name	R/W	Reset Value	Function
				to LSE, first SW switches HSE to HSISYS, then turns off HSEON and turns on LSEON, and then SW switches to LSE when LSE is ready; the same is true for switching LSE to HSE) Scenarios where the hardware is configured as HSISYS include: 1. The MCU exits from stop mode; 2. Software configuration is 001 (HSE) and HSE failure occurs.

7.4.4. External clock source control register (RCC_ECSCR)

Address offset: 0x10

Reset value: 0x0003_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.	LSE_STARTUP		Res.		LSE_DRIVER	
										RW				RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSE_STARTUP	Res.			HSE_DRIVER
											RW				RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:20	LSE_STARTUP	RW	0x0	LSE crystal oscillator stabilization time selection. LSEBYP=0: 00: 4096 LSE clock cycles; 01: 2048 LSE clock cycles; 10: 8192 LSE clock cycles; 11: Direct output regardless of stabilization time; LSEBYP=1: 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_DRIVER	RW	0x3	Low-speed crystal oscillator driving capability selection. 00: Weakest driving capability 01: Weak driving ability 10: Default driving capability (recommended) 11: Strongest driving capability Note: It is necessary to select the appropriate driving capability according to the crystal oscillator characteristics, load capacitance and parasitic parameters of the circuit board. The greater the driving capability, the greater the power consumption, and the weaker the driving capability, the smaller the power consumption.
15:5	Reserved	-	-	Reserved
4:3	HSE_STARTUP	RW	0x0	HSE crystal oscillator stabilization time selection. HSEBYP=0: 00: 4096 HSE clocks 01: 2048 HSE clocks 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;
2:1	Reserved	-	-	Reserved
0	HSE_DRV	RW	0x0	HSE drive capability selection, default 0

Bit	Name	R/W	Reset Value	Function
				0: Weak driving capability 1: Strong driving ability

7.4.5. 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.	Res.	HSE RDYIE	HSI RDYIE	Res.	LSE RDYIE	LSI RDYIE
											RW	RW		RW	RW

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

7.4.6. 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.	Res.	HSE RDYF	HSI RDYF	Res.	LSE RDYF	LSI RDYF
						R	R				R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSF	R	0	LSE Clock security system interrupt flag. Set by hardware when a failure is detected in the LSE oscillator. 0: No clock security interrupt caused by LSE clock failure 1: Clock security interrupt caused by LSE clock failure Write LSECSSC register 1 to clear this bit.
8	CSSF	R	0	HSE Clock security system interrupt flag. Set by hardware when a failure is detected in the HSE oscillator. 0: No clock security interrupt caused by HSE clock failure 1: Clock security interrupt caused by HSE clock failure Write CSSC register 1 to clear this bit.

Bit	Name	R/W	Reset Value	Function
7:5	Reserved	-	-	Reserved
4	HSERDYF	R	0	HSE ready interrupt enable Set by hardware when the HSE clock becomes stable and HSERDYDIE = 1. 0: HSE clock ready interrupt is not generated; 1: HSE clock ready interrupt generated; Write HSERDYC register 1 to clear this bit.
3	HSIRDYF	R	0	HSI ready interrupt enable Set by hardware when the HSI clock becomes stable and HSIRDYDIE = 1. 0: HSI clock ready interrupt not generated; 1: HSI clock ready interrupt generated; Write HSIRDYC register 1 to clear this bit.
2	Reserved	-	-	Reserved
1	LSERDYF	R	0	LSERDY ready interrupt enable Set by hardware when the LSE clock becomes stable and LSERDYDIE = 1. 0: LSERDY clock ready interrupt not generated; 1: LSERDY clock ready interrupt generated; Write LSERDYC register 1 to clear this bit.
0	LSIRDYF	R	0	LSIRDY ready interrupt enable Set by hardware when the LSI clock becomes stable and LSIRDYDIE = 1. 0: LSIRDY clock ready interrupt not generated; 1: LSIRDY clock ready interrupt generated; Write LSIRDYC register 1 to clear this bit.

7.4.7. 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.	Res.	HSE RDYC	HSI RDYC	Res.	LSE RDYC	LSI RDYC
						W	W				W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSC	W	0	LSE Clock security system interrupt flag. 0: No effect; 1: Clear LSECSSF flag
8	CSSC	W	0	LSE clock security system (CSS) interrupt flag cleared. 0: No effect 1: Clear CSSF flag
7:5	Reserved	-	-	Reserved
4	HSERDYC	W	0	HSE ready interrupt flag is cleared. 0: No effect 1: Clear HSERDYF flag
3	HSIRDYC	W	0	The HSI ready interrupt flag is cleared. 0: No effect 1: Clear HSIRDYF flag
2	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
1	LSERDYC	W	0	The LSE ready interrupt flag is cleared. 0: No effect 1: Clear LSERDYF flag
0	LSIRDYC	W	0	LSI ready interrupt flag is cleared. 0: No effect 1: Clear LSIRDYF flag

7.4.8. I/O port 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.	Res.	GPIOB RST	GPIOA RST
										RW				RW	RW

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

7.4.9. AHB peripheral reset register (RCC_AHBRSTR)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	CRCRST	RW	0	CRC reset 0: No effect 1: CRC reset
11:0	Reserved	-	-	Reserved

7.4.10. APB peripheral reset register 1 (RCC_APBRSTR1)

Address offset: 0x2C

Reset value = 0x0000_0000

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

Res.	Res.	Res.	PWR RST	DBG RST	Res.	Res.	Res.	Res.	Res	I2C RST	Res	Res.	UART2 RST	UART1 RST	Res.
			RW	RW						RW			RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	RTCAP-BRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
					RW										

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

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	Reserved
28	PWRRST	RW	0	Power interface reset 0: no effect 1: The module is reset
27	DBGRST	RW	0	DBG reset 0: no effect 1: The module is reset
26:22	Reserved	-	-	Reserved
21	I2C1RST	RW	0	I^2C1 reset. 0: no effect 1: The module is reset
20:19	Reserved	-	-	Reserved
18	UART2RST	RW	0	UART2 reset 0: no effect 1: The module is reset
17	UART1RST	RW	0	UART1 reset 0: no effect 1: The module is reset
16:11	Reserved	-	-	Reserved
10	RTCAPBRST	RW	0	RTC module APB reset. 0: no effect 1: The module is reset
9:0	Reserved	-	-	Reserved

7.4.11. APB peripheral reset register 2 (RCC_APBRSTR2)

Address offset: 0x30

Reset value = 0x0000_0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	TKRST	Res.	Res.	COMP2 RST	COMP1 RST	ADCRST	Res.	Res.	Res.	Res.
						RW			RW	RW	RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14RST	Res.	UART3RST	SPI1RST	TIM1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYS-CFGRST
RW		RW	RW	RW											RW

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

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	TKRST	RW	0	TK reset 0: no effect 1: The module is reset
24:23	Reserved	-	-	Reserved
22	COMP2RST	RW	0	COMP2 reset

				0: no effect 1: The module is reset
21	COMP1RST	RW	0	COMP1 reset 0: no effect 1: The module is reset
20	ADCRST	RW	0	ADC reset. 0: no effect 1: The module is reset
19:17	Reserved	-	-	Reserved
15	TIM14RST	RW	0	TIM14 reset. 0: no effect 1: The module is reset
14	Reserved	-	-	Reserved
13	UART3RST	RW	0	UART3 reset 0: no effect 1: The module is reset
12	SPI1RST	RW	0	SPI1 reset 0: no effect 1: The module is reset
11	TIM1RST	RW	0	TIM1 reset 0: no effect 1: The module is reset
10:1	Reserved	-	-	Reserved
0	SYSCFGRST	RW	0	SYSCFG reset 0: no effect 1: The module is reset

7.4.12. I/O Port clock enable register (RCC_IOPENR)

Address offset: 0x34,

Reset value = 0x0000_0000

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

This bit is set and cleared by software.

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	GPIOFEN	RW	0	I/O PortF clock enabled. 0: Clock disabled 1: clock enabled;
4:2	Reserved	-	-	Reserved
1	GPIOBEN	RW	0	IO port B clock enable 0: Clock disabled 1: clock enabled;
0	GPIOAEN	RW	0	IO port A clock enable 0: Clock disabled 1: clock enabled;

7.4.13. AHB peripheral clock enable register (RCC_AHBENR)

Address offset: 0x38,

Reset value = 0x0000_0300

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

This bit is set and cleared by software.

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	CRCEN	RW	0	CRC clock enable 0: Disabled 1: Enabled
11:10	Reserved	-	-	Reserved
9	SRAMEN	RW	1	SRAM memory area clock enabled for Sleep mode. 0: Disabled 1: Enabled
8	FLASHEN	RW	1	Flash interface module clock enabled, for Sleep mode. 0: Disabled 1: Enabled
7:0	Reserved	-	-	Reserved

7.4.14. APB peripheral clock enable register 1 (RCC_APBENR1)

Address offset: 0x3C,

Reset value = 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	PWREN	DBGEN	Res.	Res.	Res.	Res.	Res.	I2CEN	Res.	Res.	UART2EN	UART1EN	Res.
			RW	RW						RW			RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	RTCAPBEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
					RW										

This bit is set and cleared by software.

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	Reserved
28	PWREN	RW	0	Power interface module clock enabled. 0: Disabled 1: Enabled
27	DBGEN	RW	0	DBG module clock enable 0: Disabled 1: Enabled
26:22	Reserved	-	-	Reserved
21	I2C1EN	RW	0	I2C1 enable 0: Disabled 1: Enabled
20:19	Reserved	-	-	Reserved
18	UART2EN	RW	0	UART2 enable 0: Disabled 1: Enabled
17	UART1EN	RW	0	UART1 enable 0: Disabled 1: Enabled
16:11	Reserved	-	-	Reserved
10	RTCAPBEN	RW	0	RTC module APB clock enabled. 0: Disabled 1: Enabled
9:0	Reserved	-	-	Reserved

7.4.15. APB peripheral clock enable register 2(RCC_APBENR2)

Address offset: 0x40,

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.	TKEN	Res.	Res.	COMP2EN	COMP1EN	ADCEN	Res.	Res.	Res.	Res.
						RW			RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM14EN	Res.	UART3EN	SPI1EN	TIM1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGEN
RW		RW	RW	RW											RW

This bit is set and cleared by software.

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	TKEN	RW	0	TK clock enable 0: Disabled 1: Enabled
24:23	Reserved	-	-	Reserved
22	COMP2EN	RW	0	COMP2 enable 0: Disabled 1: Enabled
21	COMP1EN	RW	0	COMP1 enable 0: Disabled 1: Enabled
20	ADCEN	RW	0	ADC enable 0: Disabled 1: Enabled
19:16	Reserved	-	-	Reserved
15	TIM14EN	RW	0	TIM14 enable 0: Disabled 1: Enabled
14	Reserved	-	-	Reserved
13	UART3EN	RW	0	UART3 enable 0: Disabled 1: Enabled
12	SPI1EN	RW	0	SPI1 enable 0: Disabled 1: Enabled
11	TIM1EN	RW	0	TIM1 enable 0: Disabled 1: Enabled
10:1	Reserved	-	-	Reserved
0	SYSCFGEN	RW	1	SYSCFG clock enable 0: Disabled 1: Enabled

7.4.16. Peripheral clock configuration register (RCC_CCIPR)

Address offset: 0x54,

Reset value = 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADC_CKMODE[3:0]				Res.	Res.	Res.	Res.
								RW							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	IWDGSEL	Res.	Res.	COMP2SEL	COMP1SEL	Res.	Res.	Res.	Res	Res.	Res.	Res.	TIMCLK_CTRL
						RW	RW								RW

This bit is set and cleared by software.

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23: 20	ADC_CKMODE[3:0]	RW	0	ADC clock mode. These bits are set and cleared by software to define how the analog ADC is clocked. 0000: PCLK 0001: PCLK/2 0010: PCLK/4 0011: PCLK/8 0100: PCLK/16 0101: PCLK/32 0110: PCLK/64 0111: PCLK/2 1000: HSI 1001: HSI/2 1010: HSI/4 1011: HSI/8 1100: HSI/16 1101: HSI/32 1110: HSI/64 1111: HSI/2 Others: It is recommended that the software operate these bits when the ADC is off and the ADC is configured to: ADCAL = 0, ADSTART = 0, ADSTP = 0 and ADEN = 0).
19:13	Reserved	-	-	Reserved
12	IWDGSEL	RW	0	IWDG internal clock source selection. 0: LSI 1: LSE
11:10	Reserved	-	-	Reserved
9	COMP2SEL	RW	0	COMP2 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: 1. Configure selection clock before enabling COMP2_FR.FLTEN. 2. Enable the COMP2 clock before selecting LSC
8	COMP1SEL	RW	0	COMP1 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: 1. Configure selection clock before enabling COMP1_FR.FLTEN. 2. Enable COMP1 clock before selecting LSC
7:1	Reserved	-	-	Reserved
0	TIMCLK_CTRL	RW	0	TIMER PCLK frequency control. 0: TIMER PCLK is system PCLK * 2, but the frequency will not exceed HCLK 1: TIMER PCLK is system PCLK * 1

7.4.17. RTC domain control register (RCC_BDCR)

Address offset: 0x5C,

Reset value = 0x0000_0000, reset by POR/PDR/BOR

This register is only allowed to be written when PWR_CR1. DBP is 1.

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

RTC EN	Res.	Res.	Res.	RTC_HSEDIV_SEL [1: 0]	RTCSEL [1:0]	Res.	LSECSSD	LSECS-SON	Res.	Res.	LSE BYP	LSE RDY	LSE ON
RW				RW	RW		RW	RW			RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	LSCSEL	RW	0	Low speed clock output selection 0: LSI 1: LSE
24:17	Reserved	-	-	Reserved
16	BDRST	RW	0	RTC soft reset. 0: no effect 1: Reset
15	RTCEN	RW	0	RTC clock enabled. 0: Disabled 1: Enabled BDRST soft reset to 00
14:12	Reserved	-	-	Reserved
11:10	RTC_HSEDIV_SEL [1: 0]	RW	0	The RTC clock source is selected as the frequency division of the HSE clock 00: 32 division 01: 128 division 10: 8 division 11:32 division BDRST soft reset to 00
9:8	RTCSEL[1:0]	RW	0	RTC clock source selection. 00: No clock 01: LSE 10: LSI 11: HSE divided by RTC _HSEDIV_SEL [1: 0]. Once the RTC clock source is selected, it cannot be changed unless: 1. RTC reset to 00 2. Select as LSE (LSECSSD = 1) but no LSE 3. BDRST soft reset to 00
7	Reserved	-	-	Reserved
6	LSECSSD	R	0	LSE CSS (clock security system) detection failed. Set by hardware to indicate when a failure has been detected by the clock security system (CSS) on the external 32.768 kHz oscillator (LSE). 0: No failure detected on LSE 1: Failure detected on ISE
5	LSECSSON	RW	0	CSS on LSE enable 0: Interrupt is inhibited 1: enable; Note: LSECSSON must be enabled after the LSE oscillator is enabled (LSEON=1) and ready (LSERDY=1). Once enabled this bit cannot be disabled, except after an LSE failure detection (LSECSSD=1).
4:3	Reserved	-	-	Reserved
2	LSEBYP	RW	0	LSE oscillator bypass 0: Not bypassed, low-speed external clock selection crystal oscillator 1: Bypassed, low-speed external clock select external interface input clock This bit can be written only when the external 32.768 kHz oscillator is disabled (LSEON=0 and LSERDY=0).
1	LSERDY	R	0	LSE OSC ready. Hardware configuration: A value of 1 in this bit indicates the LSE clock is ready.

0	LSEON	RW	0	LSE oscillator enable 0: Disabled 1: Enabled Note: LSEON and HSEON cannot be used at the same time.

7.4.18. Control/status register (RCC_CSR)

Address offset: 0x60

Reset value = 0x0800_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IWDG RSTF	SFT RSTF	POR RSTF	PIN RSTF	OBL RSTF	Res.	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		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.	BOR RSTF	NRST_ FLT-DIS	Res.	Res.	Res.	Res.	Res.	Res.	LSI RDY	LSION
						R	RW							R	RW

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

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

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	IWDGRSTF	R	0	Independent watchdog reset flag Cleared by writing to the RMVF bit.
28	SFTRSTF	R	0	Software reset flag Cleared by writing to the RMVF bit.
27	PORRSTF	R	1	POR reset flag. Cleared by writing to the RMVF bit.
26	PINRSTF	R	0	PIN reset flag Cleared by writing to the RMVF bit.
25	OBLRSTF	R	0	Option byte loader reset flag Cleared by writing to the RMVF bit.
24	Reserved	-	-	Reserved
23	RMVF	RW	0	After the software is set, the reset flag will be cleared.
22:10	Reserved	-	-	Reserved
9	BORRSTF	R	0	BOR reset flag. Cleared by writing to the RMVF bit.
8	PINRST_FLTDIS	RW	0	NRST filtering disabled 0: HSI_10M enabled, and filtering 40 μs width function enabled 1: Filtering function is disabled and HSI_10M remains off
7:2	Reserved	-	-	Reserved
1	LSIRDY	R	0	LSI oscillator stable flag LSIRDY is generated by analog LSI.
0	LSION	RW	0	LSI oscillator enable 0: Interrupt is inhibited 1: enable; Set and cleared by software. The hardware sets this bit when IWDG is hardware-enabled (via option byte) and LSECSSON is software-enabled.

8. General-purpose I/Os (GPIO)

8.1. Introduction

Each general-purpose I/O port has:

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

8.2. GPIO main features

- Output states: 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) to allow bit write access to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configurations
- Analog function
- Alternate function selection registers (at most 8 AFs possible per I/O)
- Fast toggle capable of changing every clock cycles
- Highly flexible pin multiplexing allows the use of I/O pins as GPIOs or as one of several peripheral functions

8.3. GPIO functional description

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

- Input floating
- Input pull-up
- Input-pull-down
- Analog
- Output open-drain with pull-up or pull-down capability
- push-pull output with pull-up or pull-down
- push-pull with pull-up or pull-down multiplexing function
- 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 register 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 the read and the modify access.

Table below gives the possible port bit configurations.

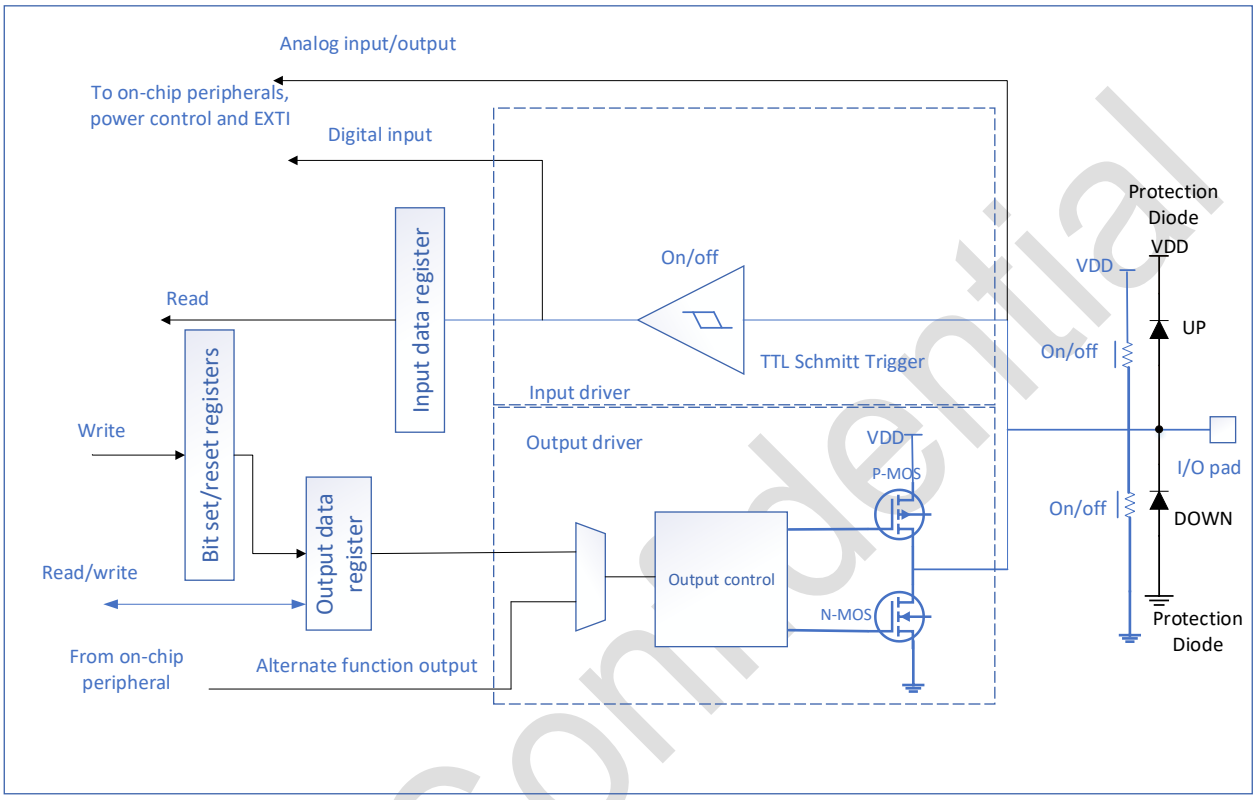


Figure 8-1 Basic structure of an I/O port bit

8.3.1. General-purpose I/Os (GPIO)

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

Table 8-1 IO port multiplexing mode

option[1:0]	PF3	PF4	PA13	PA14
0/0(default)	SWCLK	SWDIO	GPIO	GPIO
0/1	GPIO	GPIO	SWDIO	SWCLK
1/0	GPIO	SWDIO	GPIO	SWCLK
1/1	SWCLK	GPIO	SWDIO	GPIO

- PF3
  - Flash option byte defaults to normal IO when SWC is not configured
  - Flash option byte when configuring SWC, the default is SWCLK drop-down
- PA14
  - Flash option byte defaults to normal IO when SWC is not configured
  - Flash option byte when configuring SWC, the default is SWCLK drop-down
- PF4

- Flash option byte defaults to normal IO when SWD is not configured
- Flash option byte when configuring SWD, the default is SWDIO pull-up

- PA13
  - Flash option byte defaults to normal IO when SWD is not configured
  - Flash option byte when configuring SWD, the default is SWDIO pull-up

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 driver in push-pullmode 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.

8.3.2. I/O pin alternate function multiplexer and mapping

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

The multiplexer on each I/O port has up to 8 multiplexing function inputs (AF0 to AF7), 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 alternate function 0 (AF0). The I/Os are configured in alternate function mode through GPIOx_MODER register
- The multiplexing functions of each leg are distributed on the corresponding data hands as explained
- In addition to this flexible I/O multiplexing architecture, each peripheral has alternate functions mapped onto different I/O pins to optimize the number of peripherals available in smaller packages.
- To use an I/O in a given configuration, the user has to proceed as follows:
- Debug function: after each device reset these pins are assigned as alternate function pins immediately usable by the debugger host
- GPIO: configure the desired I/O as output, input or analog in the GPIOx_MODER register.
- Peripheral alternate function:
  - The I/O corresponding to the register GPIOx_AFRL or GPIOx_AFRH configuration is multiplexing function x (x = 0... 15)
  - Select the type, pull-up/pull-down and output speed via the GPIOx_OTYPER, GPIOx_PUPDR and GPIOx_OSPEEDER registers, respectively.
  - Configure the desired I/O as an alternate function in the GPIOx_MODER register.
- Additional functions:
  - ADC and COMP connection could be enabled in ADC or COMP registers regardless the configured GPIO mode. It is recommended to configure GPIO in analog mode in the

GPIOx_MODER register when ADC or COMP is used.

— For the additional functions like oscillators, configure the required function in the related PWR and RCC registers. These functions have priority over the configuration in the standard GPIO registers.

8.3.3. I/O port control registers

Each of the GPIO ports has four 32-bit memory-mapped control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) to configure up to 16 I/Os. The GPIOx_MODER register is used to select the I/O mode (input, output, AF, analog). The 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.

8.3.4. I/O port data registers

Each GPIO has two 16-bit memory-mapped data registers: input and output data registers (GPIOx_IDR and GPIOx_ODR). GPIOx_ODR stores the data to be output, it is read/write accessible. The data input through the I/O are stored into the input data register (GPIOx_IDR), a read-only register.

8.3.5. I/O data bitwise handling

The bit set reset register (GPIOx_BSRR) is a 32-bit register which allows the application to set and reset each individual bit 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 in GPIOx_BSRR: BS(i) and BR(i). When written to 1, bit BS(i) sets the corresponding ODR(i) bit. When written to 1, bit BR(i) resets the ODR(i) corresponding bit.

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

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

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

8.3.6. GPIO locking mechanism

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

To write the GPIOx_LCKR register, a specific write / read sequence has to be applied. When the right LOCK sequence is applied to bit 16 in this register, the value of LCKR [15:0] is used to lock the configuration of the I/Os (during the write sequence the LCKR [15:0] value must be the same). When the LOCK sequence has been applied to a port bit, the value of the port bit can no longer be modified 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 sequence can only be performed using a word (32-bit long) access to the GPIOx_LCKR register due to the fact that GPIOx_LCKR bit 16 has to be set at the same time as the [15:0] bits.

8.3.7. I/O alternate function input/output

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

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.

8.3.8. External interrupt/wakeup lines

All ports have external interrupt capability. To use external interrupt lines, the given pin must not be configured in analog mode or being used as oscillator pin, so the input trigger is kept enabled.

8.3.9. Input configuration

When the I/O port is programmed as input:

- Output Buffer off
- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state

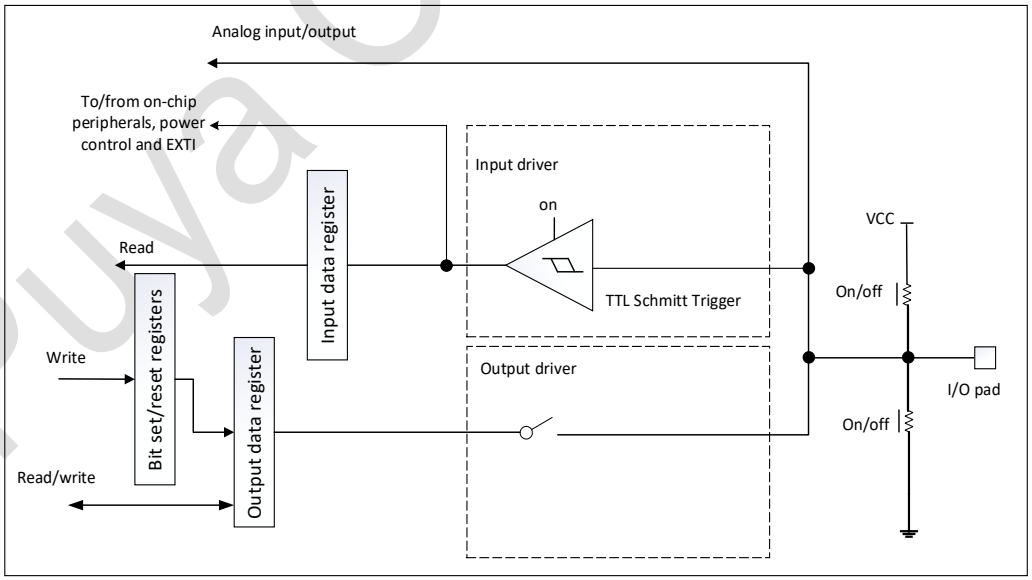


Figure 8-2 Input floating/pull-up/pull-down configuration

8.3.10. Output configuration

When the I/O port is programmed as output:

- The output buffer is disabled

— Open drain mode: A “0” in the Output register activates the N-MOS whereas a “1” in the output register leaves the port in Hi-Z (the P-MOS is never activated).

— Push-pull mode: A “0” in the Output register activates the N-MOS whereas a “1” in the Output register activates the P-MOS.

- The Schmitt trigger input is activated
- The pull-up and pull-down resistors are activated depending on the value in the GPIOx_PUPDR register
- The data present on the I/O pin are sampled into the input data register every AHB clock cycle
- A read access to the input data register provides the I/O state
- A read access to the output data register gets the last written value

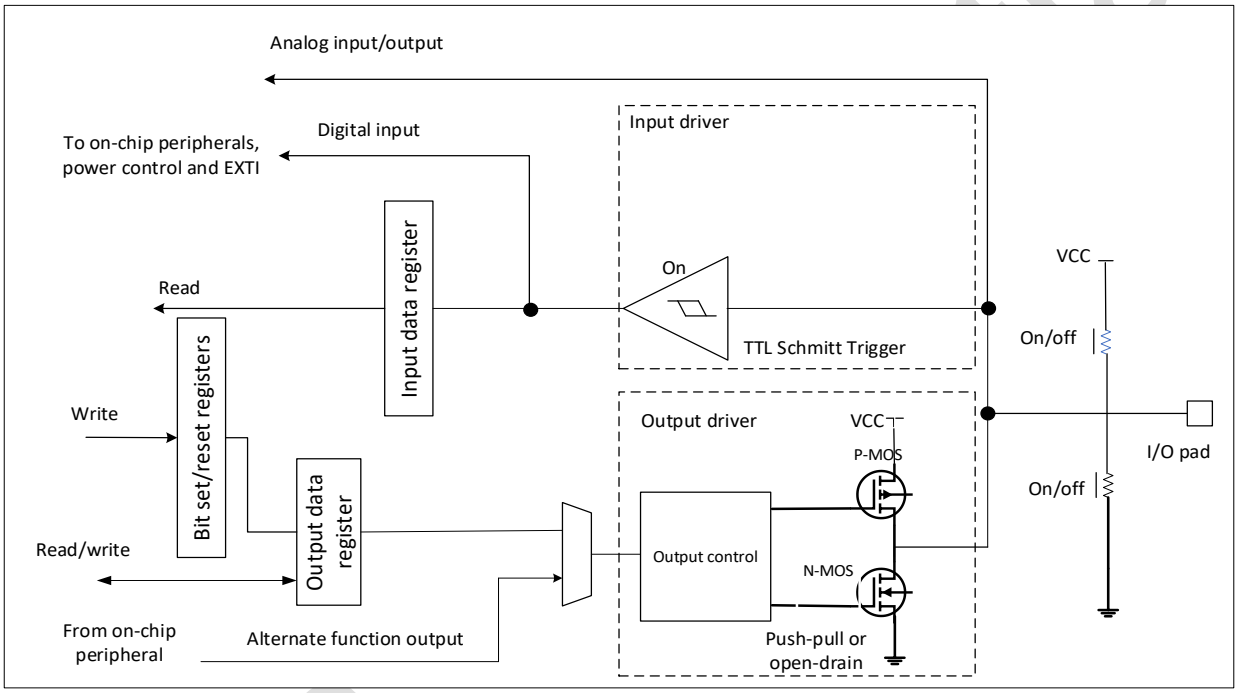


Figure 8-3 Output configuration

8.3.11. Alternate function configuration

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

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

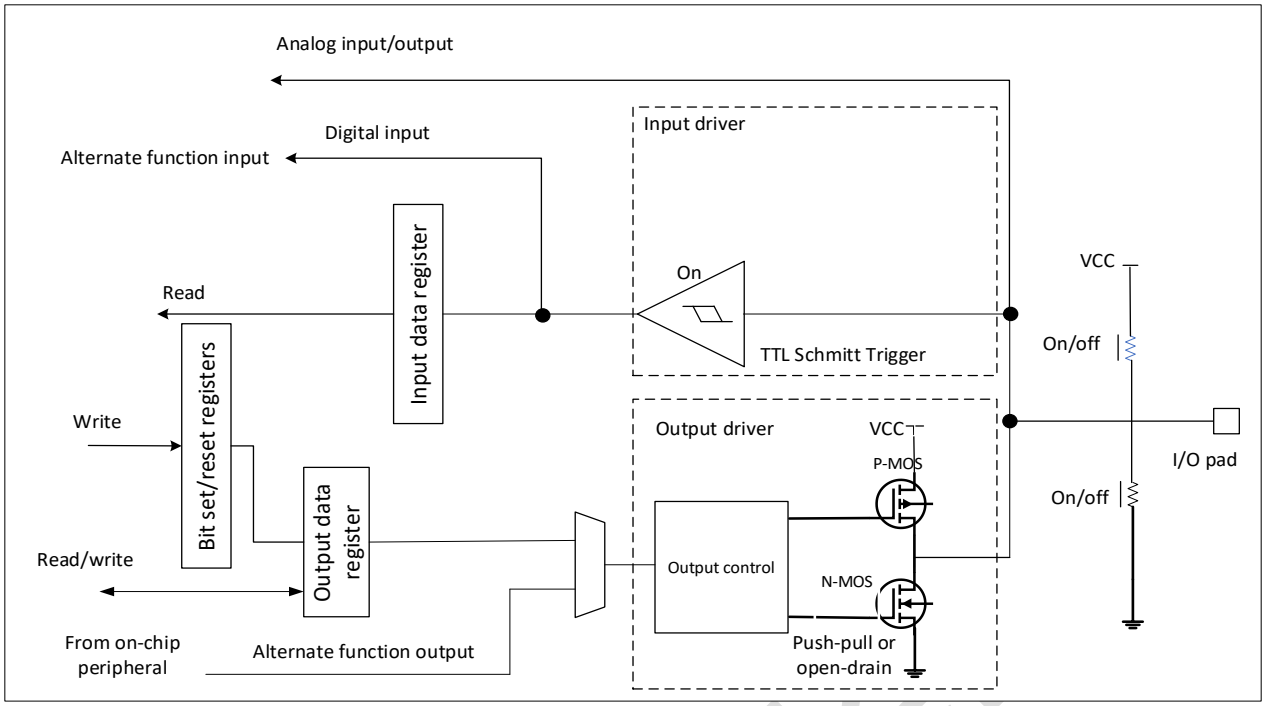


Figure 8-4 Alternate function configuration

8.3.12. Analog configuration

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

- The output buffer is disabled;
- The Schmitt trigger input is deactivated, providing zero consumption for every analog value of the I/O pin. The output of the Schmitt trigger is forced to a constant value (0);
- Weak pull-up and pull-down resistors are disabled (software setting is required);
- Read access to the input data register gets the value “0”.

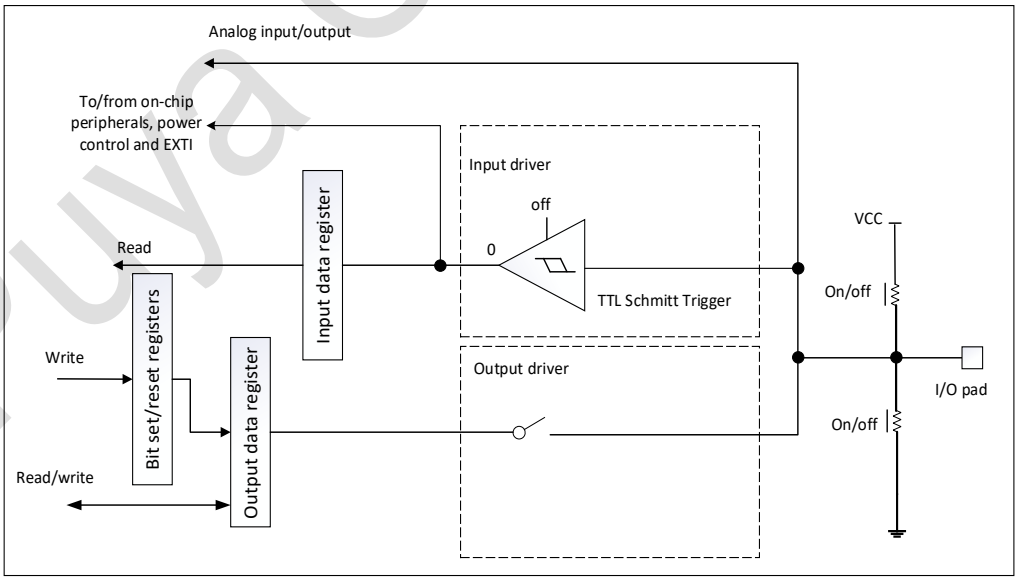


Figure 8-5 High impedance-analog configuration

8.3.13. Using the LSE oscillator pin as GPIO

When the LSE oscillator is switched OFF (default state after reset), the related oscillator pin can be used as normal GPIO.

When the HSE or LSE oscillator is switched ON (by setting the LSEON bit in the RCC_BDCR register), the corresponding port should be configured as analog port.

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

8.4. GPIO registers

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

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

Address offset: 0x00

Reset value:

- Reset value for GPIOA
  - Flash option byte when configured 11: 0xFBFF FFFF
  - Flash option byte when configured 10: 0xEFFF FFFF
  - Flash option byte when configured 01: 0xEBFF FFFF
  - Flash option byte when configured 00: 0xFFFF FFFF
- Reset value for GPIOB
  - 0x0000 FFFF
- Reset value for GPIOF
  - When Flash option byte is configured to 11: 0x0000 0FBF
  - When Flash option byte is configured to 10: 0x0000 0EFF
  - When Flash option byte is configured to 01: 0x0000 0FFF
  - When Flash option byte is configured to 00: 0x0000 0EBF

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															
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															

Bit	Name	R/W	Reset Value	Function
31:0	MODEy[1:0]	RW		y = 15..0 These bits are written by software to configure the I/O mode. 00: Input mode 01: General purpose output mode 10: Alternate function mode 11: reset state

8.4.2. 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	-	-	Reserved
15:0	OT[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

8.4.3. GPIO port output speed register (GPIOx_OSPEEDR) (x = A, B, F)

Address offset: 0x08

Reset value:

- Reset value for GPIOA
  - Flash option byte when configured 11: 0x0C00 0000
  - Flash option byte when configured 10: 0x0000 0000
  - Flash option byte when configured 01: 0x0C00 0000
  - Flash option byte when configured 00: 0x0000 0000
- Reset value for GPIOB
  - 0x0000 0000
- Reset value for GPIOF
  - Flash option byte when configured 11: 0x0000 0300
  - Flash option byte when configured 10: 0x0000 0000
  - Flash option byte when configured 01: 0x0000 0000
  - Flash option byte when configured 00: 0x0000 0300

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: Low speed 01: Medium speed 10: high speed 11: Very high speed

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

Address offset: 0x0C

Reset value:

- Reset value for GPIOA
  - Flash option byte when configured 11: 0x0400 0000
  - Flash option byte when configured 10: 0x2000 0000
  - Flash option byte when configured 01: 0x2400 0000

- Flash option byte when configured 00: 0x0000 0000

- Reset value for GPIOB
  - 0x0000 0000

- Reset value for GPIOF
  - Flash option byte when configured 11: 0x0000 0080
  - Flash option byte when configured 10: 0x0000 0100
  - Flash option byte when configured 01: 0x0000 0000
  - Flash option byte when configured 00: 0x0000 0180

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

Bit	Name	R/W	Reset Value	Function
31:0	PUPDy [1: 0]	RW		y = 15..0 These bits are written by software to configure the I/O pull-up or pull-down 00: No pull-up, pull-down 01: Pull-up 10: Pull-down 11: Pull up + pull down

8.4.5. GPIO port input data register (GPIOx_IDR) (x = A, B, F)

Address offset: 0x10

Reset value: 0x0000 XXXX

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	IDy	R	0000	y = 15..0 These bits are read-only. They contain the input value of the corresponding I/O port.

8.4.6. 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	0	y = 15..0 These bits can be read and written by software.

Bit	Name	R/W	Reset Value	Function
				Note: the ODR bits can be individually set and/or cleared by writing to the GPIOx_BSRR or GPIOx_BRR register (x = A.B.F).

8.4.7. 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	**0**	y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: No effect on the corresponding ODy bit 1: Clear the corresponding ODRy bit Note: If both BSy and BRy are set, BSy has priority.
15:0	BSy	W	**0**	y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: No effect on the corresponding ODy bit 1: Sets the corresponding ODRy bit

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

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

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

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

Address offset: 0x1C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:17	Reserved			
16	LCKK	RW	**0**	This bit can be read any time. It can only be modified using the lock key write sequence. 0: Port configuration lock key not active 1: Port configuration lock key active. The GPIOx_LCKR register is locked until the next system reset. LOCK key write sequence:

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

8.4.9. GPIO alternate function low register (GPIOx_AFRL) (x = A, B, F)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	AFSEL15 [2:0]			Res.	AFSEL14 [2:0]			Res.	AFSEL13 [2:0]			Res.	AF-SEL12 [2:0]		
	RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	AFSEL11 [3:0]			Res.	AFSEL10 [2:0]			Res.	AFSEL9[2:0]			Res.	AFSEL8[2:0]		
	RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved			
30:28	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
27	Reserved			
26:24	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
23	Reserved			
22:20	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3

Bit	Name	R/W	Reset Value	Function
				100: AF4 101: AF5 110: AF6 111: AF7
19	Reserved			
18:16	AFSELy [2: 0] (y = 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
15	Reserved			
14:12	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
11	Reserved			
10:8	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
11	Reserved			
10:8	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
7	Reserved			
6:4	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection:

Bit	Name	R/W	Reset Value	Function
				000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
3	Reserved			
2:0	AFSELy[2:0](y= 7 to 0)	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7

8.4.10. GPIO alternate function high register (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
Res.	AFSEL15 [2:0]			Res.	AFSEL14 [2:0]			Res.	AFSEL13 [2:0]			Res.	AF-SEL12 [2:0]		
	RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	AFSEL11 [3:0]			Res.	AFSEL10 [2:0]			Res.	AFSEL9[2:0]			Res.	AFSEL8[2:0]		
	RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved			
30:28	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
27	Reserved			
26:24	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5

Bit	Name	R/W	Reset Value	Function
				110: AF6 111: AF7
23	Reserved			
22:20	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
19	Reserved			
18:16	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
15	Reserved			
14:12	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
11	Reserved			
10:8	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
11	Reserved			
10:8	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1

Bit	Name	R/W	Reset Value	Function
				010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
7	Reserved			
6:4	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
3	Reserved			
2:0	AFSELy [2: 0] (y = 8 to 15)	RW		These bits are written by software to configure alternate function I/Os. AFSELy selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7

8.4.11. 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
BRy [15: 0]															
W															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	BRy	RW	16'h0	y = 15..0 These bits are write-only. A read to these bits returns the value 0. 0: No effect on the corresponding ODy bit 1: Clears the corresponding ODy bit

9. System configuration controller (SYSCFG)

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

- I2C IO related functions
- IO port filter enable control
- Mapping the initial program area according to different boot modes
- Manage TIMERS ETR and brake input

9.1. SYSCFG registers

9.1.1. SYSCFG configuration register 1 (SYSCFG_CFGR1)

This register is used for specific configurations of memory request remap and to control special I/O features.

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

Address offset: 0x00

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	ETR_SRC_TIM1 [1:0]		Res.	Res.	Res.		TIM1_IC1_SRC		MEM_MODE[1:0]	
						RW	RW					RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	ETR_SRC_TIM1 [1:0]	RW	3’b00	TIMER1 ETR input source selection. 2 'b00: ETR derived from GPIO; 2 'b01: ETR derived from COMP1 output; 2 'b10: ETR derived from COMP2 output; 2 'b11: ETR is derived from ADC analog watchdog output;
7:6	Reserved	-	0	Reserved
3:2	TIM1_IC1_SRC	RW	00	TIM1 CH1 input source: 00, 11: from TIM1_CH1 10; 01: from COMP1; 10: from COMP2.
1:0	MEM_MODE[1:0]			Memory mapped select bit Set and cleared by software. They control the mapping of 0x0000 0000 addresses of the memory. After reset these bits take the value selected by the actual boot mode configuration. X0: Main flash, mapped at 0x0000 0000 01: Load flash, mapped at 0x0000 0000 11: SRAM, mapped at 0x0000 0000

9.1.2. 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	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	COMP2_Ocref_CLR_TIM1	Res	COMP1_Ocref_CLR_TIM1	Res	Res	Res	Res	COMP2_BRK_TIM1	COMP1_BRK_TIM1	Res	Res	LOCKUP_LOCK
				RW		RW					RW	RW			RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	COMP2_Ocref_CLR_TIM1	RW	1'b0	1: COMP2 output as TIM1 ocref_clr input; 0: COMP2 output is not input as TIM1 ocref_clr
10	Reserved			
9	COMP1_Ocref_CLR_TIM1	RW	1'b0	1: COMP1 output as TIM1 ocref_clr input; 0: COMP1 output is not input as TIM1 ocref_clr
8:5	Reserved	-	-	-
4	COMP2_BRK_TIM1	RW	0	COMP2 is enabled as a TIMx break input. 0: COMP2 output is not used as TIM1 break input 1: COMP2 output as TIM1 break input
3	COMP1_BRK _TIM1	RW	0	COMP1 is enabled as a TIMx break input. 0: COMP1 output is not used as TIM1 break input 1: COMP1 output as TIM1 break input
2:1	Reserved	-	-	-
0	LOCKUP_LOCK	RW	0	Enable Cortex-M0+ LOCKUP 0: The Cortex-M0 + LOCKUP bit is not connected to the TIM1 break input; 1: Cortex-M0 + LOCKUP bit is connected to TIM1 break input

9.1.3. GPIOA filter enable register (PA_ENS)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PA_ENS [15: 0]	RW	0x0000	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter

9.1.4. GPIOB filter enable register (PB_ENS)

Address offset: 0x14

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
3:0	PB_ENS [3: 0]	RW	0	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter

9.1.5. GPIOF filter enable register (PF_ENS)

Address offset: 0x18

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5:0	PF_ENS [5: 0]	RW	0	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter

9.1.6. I2C type IO configuration register (SYSCFG_IOCFG)

Address offset: 0x1C

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23:22	PB_ENSEG [1: 0]	RW	0	PB ENSEG signal control. Used as SEG mode control. PB2:bit0 PB3:bit1 The user can modify the configuration of GPIOB_OSPEEDR and determine the driving currents of PB2 ~ 3
21:16	PA_ENSEG[5:0]	RW	0	PA ENSEG signal control. Used as SEG mode control. PA0:bit0 PA1:bit1 PA5:bit2 PA6:bit3 PA7:bit4 PA8:bit5 The user can modify the configuration of GPIOA_OSPEEDR and determine the drive currents of PA0 ~ 1, 5 ~ 8
15	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
14:13	PA_IODRVP [1: 0]	RW	0	PA IODRVP signal control. Uses as an output driver function setting for I2C_VCC. (PA2 and 7 correspond to bit0 to 1)
12:8	PA_EHS [4: 0]	RW	0	PA EHS signal control. Used as 80mA LED IO control. (PA11 to 15 correspond to bit0 to 4)
7:6	PF_PU_IIC[1:0]	RW	0	I2C_PU type IO pull-up resistor control is enabled. (PF3 to 4 correspond to bit0 to 1)
5	Reserved	-	-	Reserved
4:2	PF_EIIC[2:0]	RW	0	PF EIIC signal control. Used as_I2C type IO. (PF2 to 4 correspond to bit0 to 2)
1:0	PA_EIIC[1:0]	RW	0	PA EIIC signal control. Used as_I2C type IO. (PA8 corresponds to bit0, PA11 corresponds to bit1)

9.1.7. GPIOA analog 2 enable register (PA_ANA2EN)

Address offset: 0x20

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA_ANA2EN[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PA_ANA2EN[15:0]	RW	0	PAD_ANA2 enable, active high, default 0 0: PAD_ANA2 switch is off 1: PAD_ANA2 switch is on

9.1.8. GPIOB analog 2 enable register (PB_ANA2EN)

Address offset: 0x24

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PB_ANA2EN [3:0]			
												RW			

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:0	PB_ANA2EN [3:0]	RW	0	PAD_ANA2 enable, active high, default 0 0: PAD_ANA2 switch is off 1: PAD_ANA2 switch is on

9.1.9. GPIOF analog 2 enable register (PF_ANA2EN)

Address offset: 0x28

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF_ANA2EN [5:0]					

										RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5:0	PF_ANA2EN [5:0]	RW	0	PAD_ANA2 enable, active high, default 0 0: PAD_ANA2 switch is off 1: PAD_ANA2 switch is on

9.1.10. GPIOF analog 2 enable register (PF_ANA2EN)

Address offset: 0x28

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF_ANA2EN [5:0]					
										RW					

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5:0	PF_ANA2EN [5:0]	RW	0	PAD_ANA2 enable, active high, default 0 0: PAD_ANA2 switch is off 1: PAD_ANA2 switch is on

9.1.11. SRAM_RETVSEL register (SRAM_RETVSEL)

Address offset: 0x2C

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	SRAM_RETVSEL	RW	0	SRAM retention voltage configuration signal SRAM_RETVSEL=1b'1 Vsram provide from V_{DD} SRAM_RETVSEL=1b'0 Vsram provide from V_{DD1}

9.1.12. GPIOA pull-down resistor configuration (PA_IORP)

Address offset:0x30

Reset value: 0xFFFF_FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PA15_ IORP[1:0]		PA14_ IORP[1:0]		PA13_ IORP[1:0]		PA12_IORP[1:0]		PA11_IORP[1:0]		PA10_ IORP[1:0]		PA9_IORP[1:0]		PA8_IORP[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA7_ IORP[1:0]		PA6_ IORP[1:0]		PA5 IORP [1: 0]		PA4_IORP [1: 0]		PA3_IORP [1: 0]		PA2_IORP [1: 0]		PA1_IORP [1: 0]		PA0_IORP[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31: 30	PA15_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
29: 28	PA14_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
27: 26	PA13_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
25: 24	PA12_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
23: 22	PA11_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
21: 20	PA10_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
19: 18	PA9_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
17: 16	PA8_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
15: 14	PA7_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
13: 12	PA6_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
11: 10	PA5_ IORP[1:0]	RW	2'b11	Same as PA0_IORP [1: 0]
9: 8	PA4_IORP [1: 0]	RW	2'b11	Same as PA0_IORP [1: 0]
7: 6	PA3_IORP [1: 0]	RW	2'b11	Same as PA0_IORP [1: 0]
5: 4	PA2_IORP [1: 0]	RW	2'b11	Same as PA0_IORP [1: 0]
3: 2	PA1_IORP [1: 0]	RW	2'b11	Same as PA0_IORP [1: 0]
1: 0	PA0_ IORP[1:0]	RW	2'b11	IO pull-up/down resistor configuration, default 11 00: Up/down pull road 01: Pull-up/pull-down resistance 10 kΩ 10: Pull-up/pull-down resistor 20 kΩ 11: pull-up/pull-down resistor 40 kΩ

9.1.13. GPIOB pull-down resistor configuration (PB_IORP)

Address offset: 0x34

Reset value: 0x0000_00FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	PB3_ IORP[1:0]		PB2_ IORP[1:0]		PB1_ IORP[1:0]		PB0_ IORP[1:0]	
								RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	Reserved
7: 6	PB3_ IORP[1:0]	RW	2'b11	Same as PB0_IORP [1: 0]
5: 4	PB2_ IORP[1:0]	RW	2'b11	Same as PB0_IORP [1: 0]
3: 2	PB1_ IORP[1:0]	RW	2'b11	Same as PB0_IORP [1: 0]
1: 0	PB0_ IORP[1:0]	RW	2'b11	IO pull-up/down resistor configuration, default 11 00: Up/down pull road 01: Pull-up/pull-down resistance 10 kΩ 10: Pull-up/pull-down resistor 20 kΩ 11: pull-up/pull-down resistor 40 kΩ

9.1.14. GPIOF pull-down resistor configuration (PF_IORP)

Address offset: 0x38

Reset value: 0x0000_0FFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PF_ANA2EN [5:0]					
										RW					

Bit	Name	R/W	Reset Value	Function
31: 12	Reserved	-	-	Reserved
11: 10	PF5_ IORP[1:0]	RW	2'b11	Same as PF0_IORP [1: 0]
9: 8	PF4_ IORP[1:0]	RW	2'b11	Same as PF0_IORP [1: 0]
7: 6	PF3_ IORP[1:0]	RW	2'b11	Same as PF0_IORP [1: 0]
5: 4	PF2_ IORP[1:0]	RW	2'b11	Same as PF0_IORP [1: 0]
3: 2	PF1_ IORP[1:0]	RW	2'b11	Same as PF0_IORP [1: 0]
1: 0	PF0_ IORP[1:0]	RW	2'b11	IO pull-up/down resistor configuration, default 11 00: Up/down pull road 01: Pull-up/pull-down resistance 10 kΩ 10: Pull-up/pull-down resistor 20 kΩ 11: pull-up/pull-down resistor 40 kΩ

10. Interrupts and events

10.1. Nested vectored interrupt controller (NVIC)

10.1.1. Main features

- Support 16 maskable external interrupts (not including the sixteen CPU interrupt lines)
- 4 programmable priority levels (2 bits of interrupt priority are used)
- Low latency exception and interrupt handling
- Power management control
- Implementation of system control registers

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

10.1.2. SysTick calibration value register

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

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

10.1.3. Interrupt and exception vectors

Table 10-1 Break Exception and Vector Table

Position	Priority	Types of Priority	Acronym	Description	Address
-	-	-	-	Reserved	0x0000_0000
-	-3	fixed	Reset	Reset	0x0000_0004
-	-2	fixed	NMI_Handler	NMI The RCC clock security system (CSS) is coupled to NMI vector	0x0000_0008
-	-1	fixed	HardFualt_Handler	All classes 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	-	Reserved	Reserved	0x0000_0040
1	8	-	Reserved	Reserved	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	RCC global interrupt	0x0000_0050
5	12	settable	EXTI0_1	EXTI line [1: 0] interrupt	0x0000_0054
6	13	settable	EXTI2_3	EXTI line [3: 2] interrupt	0x0000_0058
7	14	settable	EXTI4_15	EXTI line [15: 4] interrupt	0x0000_005C
8	15	settable	TK	TK int	0x0000_0060
9	16	-	Reserved	Reserved	0x0000_0068
10	17	-	Reserved	Reserved	0x0000_0068
11	18	-	Reserved	Reserved	0x0000_006C
12	19	settable	ADC_COMP	ADC and COMP interrupts (COMP combined with EXTI 17 & 18)	0x0000_0070
13	20	settable	TIM1_BRK_UP_TRG_COM	TIM1 break, update, trigger and commutation interrupt	0x0000_0074

Position	Priority	Types of Priority	Acronym	Description	Address
14	21	settable	TIM1_CC	TIM1 capture compare interrupt	0x0000_0078
15	22	-	Reserved	Reserved	0x0000_007C
16	23	-	Reserved	Reserved	0x0000_0080
17	24	-	Reserved	Reserved	0x0000_0084
18	25	-	Reserved	Reserved	0x0000_0088
19	26	settable	TIM14	TIM14 global interrupt	0x0000_008C
20	27	-	Reserved	Reserved	0x0000_0090
21	28	-	Reserved	Reserved	0x0000_0094
22	29	-	Reserved	Reserved	0x0000_0098
23	30	settable	I^2C1	I^2C1 global interrupt	0x0000_009C
24	31	-	Reserved	Reserved	0x0000_00A0
25	32	settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	-	Reserved	Reserved	0x0000_00A8
27	34	-	Reserved	Reserved	0x0000_00AC
28	35	-	Reserved	Reserved	0x0000_00B0
29	36	settable	UART3	UART3 global interrupt	0x0000_00B4
30	37	settable	UART2	UART2 global interrupt	0x0000_00B8
31	38	settable	UART1	UART1 global interrupt	0x0000_00BC

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

10.2. Extended interrupts and events controller (EXTI)

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

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

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

EXTI allows to manage up to 21 configurable/direct event lines (including 18 configurable event lines and 3 direct event lines).

10.2.1. EXTI main features

The system can wake up via GPIO and specified module (COMP/RTC/I2C/TK) input events

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

10.2.2. EXTI block diagram

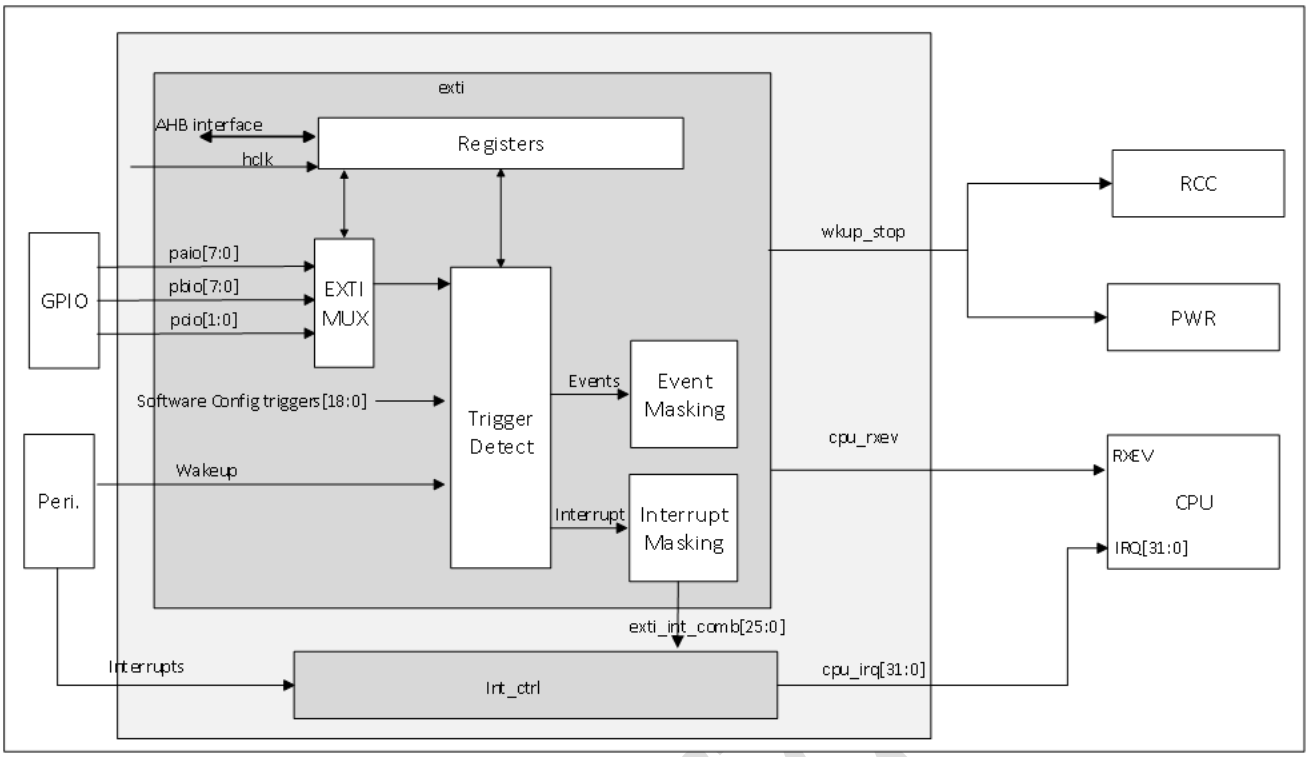


Figure 10-1 EXTI block diagram

10.2.3. EXTI connections between peripherals and CPU

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

- A wake-up signal that generates a pulse, or without an interrupt status bit inside the peripheral, is connected to the configurable line of the EXTI module. At this time, the EXTI module generates an interrupt suspend bit (this bit needs to be cleared), and the EXTI interrupt will serve as an interrupt signal for the CPU.
- The interrupt and wake-up signals of the peripheral with the associated status bit (which is cleared at the peripheral) are connected to the wake-up trigger signal line of the EXTI module.
- All GPIO ports are input to the EXTI MUX module, and through the configuration of configurable, it is allowed to be selected as a system wake-up signal.

10.2.4. EXTI configurable event trigger wake-up

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

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

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

Configurable type events have a unique interrupt suspend request register, which is shared with the CPU. The suspend register is set only if the CPU interrupt mask register (EXTI_IMR) is configured

to be unmasked. Each configurable type event will correspond to an external CPU interrupt signal (some will be multiplexed to the same external CPU interrupt signal). Configurable type event interrupts require CPU acknowledgement through the EXTI_PR register (write 1 clear).

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

10.2.5. EXTI direct event input wake-up

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

10.2.6. EXTI multiplexer

The GPIOs are connected to 16 external interrupt/event lines:

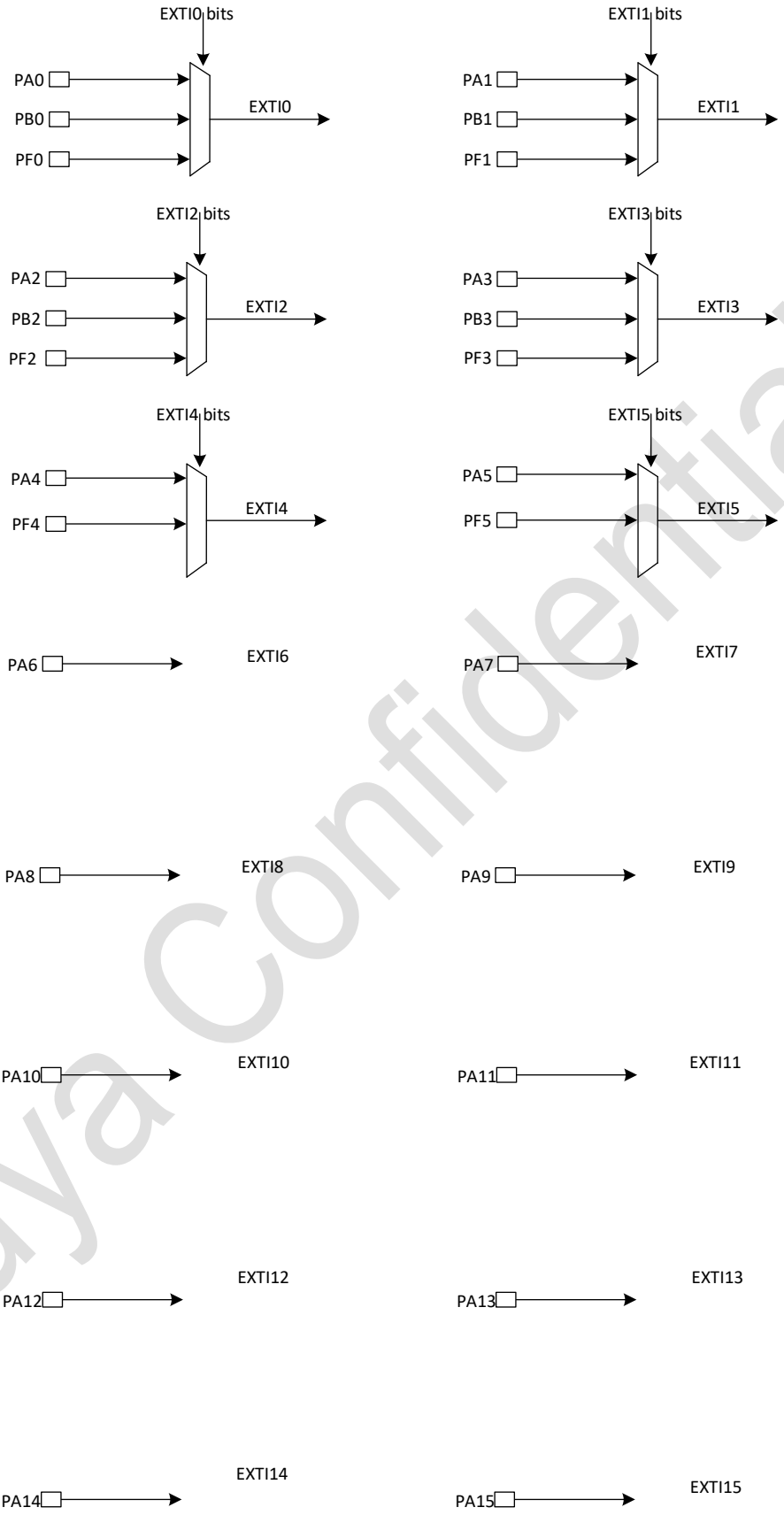


Figure 10-2 External interrupt/event GPIO mapping

The EXTI lines are connected as shown as follows:

Table 10-2 External interrupts/events line connections

EXTI line	Line source	Line type
Line 0-15	GPIO	Configurable
Line 16	Reserved	-
Line 0-7	COMP 1 output	Configurable
Line 18	COMP2 output	Configurable
Line 19	RTC	Direct
Line 20	Reserved	-
Line 21	Reserved	-
Line 22	Reserved	-
Line 23	I^2C(I2c_address match see I^2C spec)	Direct
Line 24	Reserved	-
Line 25	Reserved	-
Line 26	Reserved	-
Line 27	Reserved	-
Line 28	TK (tk touch yes see TK spec)	Direct

10.3. EXTI register

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

10.3.1. EXTI rising trigger selection register (EXTI_RTSR)

Address offset:0x00

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RT18	RT17	Res
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	RT18	RW	0	Rising trigger event configuration bit of line18 0: Disabled 1: Enabled
17	RT17	RW	0	Rising trigger event configuration bit of line17 0: Disabled 1: Enabled
16	Reserved	-	-	Reserved
15	RT15	RW	0	Rising trigger event configuration bit of line15 0: Disabled 1: Enabled
14	RT14	RW	0	Rising trigger event configuration bit of line14 0: Disabled 1: Enabled
13	RT13	RW	0	Rising trigger event configuration bit of line13 0: Disabled 1: Enabled
12	RT12	RW	0	Rising trigger event configuration bit of line12 0: Disabled

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

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

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

10.3.2. EXTI falling trigger selection register (EXTI_FTSR)

Address offset: 0x04

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT18	FT17	Res
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	Reserved
18	FT18	RW	0	Falling trigger event configuration bit of line18 0: Disabled

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

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

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

10.3.3. Software interrupt event register (EXTI_SWIER)

Address offset: 0x08

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SW18	SW17	Res.
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SW15	SW14	SW13	SW12	SW11	SW10	SW9	SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

Bit	Name	R/W	Reset Value	Function
8	SWI8	RW	0	Rising trigger event configuration bit of line8 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
7	SWI7	RW	0	Rising trigger event configuration bit of line7 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
6	SWI6	RW	0	Rising trigger event configuration bit of line6 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
5	SWI5	RW	0	Rising trigger event configuration bit of line5 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
4	SWI4	RW	0	Rising trigger event configuration bit of line4 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
3	SWI3	RW	0	Rising trigger event configuration bit of line3 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
2	SWI2	RW	0	Rising trigger event configuration bit of line2 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
1	SWI1	RW	0	Rising trigger event configuration bit of line1 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
0	SWI0	RW	0	Rising trigger event configuration bit of line0 0: No effect 1: Generate rising trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)

10.3.4. Pending register (EXTI_PR)

Address offset: 0x0C

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR18	PR17	Res.
													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
31:19	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
18	PR18	RC_W1	0	Rising trigger event configuration bit of line18 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
17	PR17	RC_W1	0	Rising trigger event configuration bit of line17 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
16	Reserved	-	-	Reserved
15	PR15	RC_W1	0	Rising trigger event configuration bit of line15 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
14	PR14	RC_W1	0	Rising trigger event configuration bit of line14 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
13	PR13	RC_W1	0	Rising trigger event configuration bit of line13 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
12	PR12	RC_W1	0	Rising trigger event configuration bit of line12 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
11	PR11	RC_W1	0	Rising trigger event configuration bit of line11 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
10	PR10	RC_W1	0	Rising trigger event configuration bit of line10 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
9	PR9	RC_W1	0	Rising trigger event configuration bit of line9 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred

Bit	Name	R/W	Reset Value	Function
				1: Generate rising edge/falling edge/software trigger event request;
8	PR8	RC_W1	0	Rising trigger event configuration bit of line8 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
7	PR7	RC_W1	0	Rising trigger event configuration bit of line7 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
6	PR6	RC_W1	0	Rising trigger event configuration bit of line6 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
5	PR5	RC_W1	0	Rising trigger event configuration bit of line5 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
4	PR4	RC_W1	0	Rising trigger event configuration bit of line4 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
3	PR3	RC_W1	0	Rising trigger event configuration bit of line3 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
2	RPIF2	RC_W1	0	Rising trigger event configuration bit of line2 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
1	PR1	RC_W1	0	Rising trigger event configuration bit of line1 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;
0	PR0	RC_W1	0	Rising trigger event configuration bit of line0 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1

Bit	Name	R/W	Reset Value	Function
				0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request;

10.3.5. EXTI external interrupt selection register 1 (EXTI_EXTICR1)

Address offset: 0x60

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	EXTI3 [1:0]		Res	Res.	Res.	Res.	Res.	Res.	EXTI2 [1:0]	
						RW	RW							RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	EXTI1 [1:0]		Res	Res.	Res	Res.	Res.	Res.	EXTI0 [1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI3 [1:0]	RW	0	EXIT3 GPIO port selection 2'b00: PA[3] pin 2'b01: PB[3] pin 2'b10: PF[3] pin 2'b11: Reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI2 [1:0]	RW	0	EXTI2 corresponds to GPIO port selection. 2'b00: PA[2] pin 2'b01: PB[2] pin 2'b10: PF[2] pin 2'b11: Reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI1 [1:0]	RW	0	EXTI1 corresponds to GPIO port selection. 2'b00: PA[1] pin 2'b01: PB[1] pin 2'b10: PF[1] pin 2'b11: Reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI0 [1:0]	RW	0	EXTI0 corresponds to GPIO port selection. 2'b00: PA[0] pin 2'b01: PB[0] pin 2'b10: PF[0] pin 2'b11: Reserved

10.3.6. EXTI external interrupt selection register 2 (EXTI_EXTICR2)

Address offset: 0x64

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	EXTI5		Res	Res.	Res	Res.	Res.	Res.	EXTI4 [1:0]	
						RW								RW	

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	EXTI5 [1:0]	RW	0	EXTI5 corresponds to GPIO port selection. 2'b00: PA[5] pin

Bit	Name	R/W	Reset Value	Function
				2'b01: PB[5] pin 2'b10: PF[5] pin 2'b11: Reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI4 [1:0]	RW	0	EXTI4 corresponds to GPIO port selection. 2'b00: PA[4] pin 2'b01: PB[4] pin 2'b10: PF[4] pin 2'b11: Reserved

10.3.7. Interrupt mask register (EXTI_IMR)

Address offset: 0x80

Reset value: 0x1088 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	IM28	Res.	Res.	Res.	Res.	IM23	Res.	Res.	Res.	IM19	IM18	IM17	Res.
			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
31:29	Reserved			
28	IM28	RW	1	EXTI line28 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
27:24	Reserved			
23	IM23	RW	1	EXTI line23 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
22:20	Reserved			
19	IM19	RW	1	EXTI line19 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
18	IM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	IM17	RW	0	Interrupt mask on line 17. 0: Interrupt request is masked 1: Interrupt request is not masked
16	Reserved			
15	IM15	RW	0	EXTI line15 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
14	IM14	RW	0	EXTI line14 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked

Bit	Name	R/W	Reset Value	Function
13	IM13	RW	0	EXTI line13 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
12	IM12	RW	0	EXTI line12 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
11	IM11	RW	0	EXTI line11 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
10	IM10	RW	0	EXTI line10 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
9	IM9	RW	0	EXTI line9 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
8	IM8	RW	0	EXTI line8 wakes up CPU mask control as an interrupt. 0: Interrupt request is masked 1: Interrupt request is not masked
7	IM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	IM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	IM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked 1: Interrupt request is not masked
4	IM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked
3	IM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	IM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	IM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	IM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

10.3.8. 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.	Res.	EM28	Res.	Res.	Res.	Res.	EM23	Res.	Res.	Res.	EM19	EM18	EM17	Res

			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
31:29	Reserved			
28	EM28	RW	0	EXTI line 28 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
27:24	Reserved			
23	EM23	RW	0	EXTI line 23 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
22:20	Reserved			
19	EM19	RW	0	EXTI line 19 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
18	EM18	RW	0	Interrupt mask on line 18. 0: Interrupt request is masked 1: Interrupt request is not masked
17	EM17	RW	0	Interrupt mask on line 17. 0: Interrupt request is masked 1: Interrupt request is not masked
16	Reserved			
15	EM15	RW	0	EXTI line15 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
14	EM14	RW	0	EXTI line14 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
13	EM13	RW	0	EXTI line13 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
12	EM12	RW	0	EXTI line12 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
11	EM11	RW	0	EXTI line11 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
10	EM10	RW	0	EXTI line10 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
9	EM9	RW	0	EXTI line9 wakes up CPU mask control as an event. 0: Interrupt request is masked 1: Interrupt request is not masked
8	EM8	RW	0	EXTI line8 wakes up CPU mask control as an event.

Bit	Name	R/W	Reset Value	Function
				0: Interrupt request is masked 1: Interrupt request is not masked
7	EM7	RW	0	Interrupt mask on line 7. 0: Interrupt request is masked 1: Interrupt request is not masked
6	EM6	RW	0	Interrupt mask on line 6. 0: Interrupt request is masked 1: Interrupt request is not masked
5	EM5	RW	0	Interrupt mask on line 5. 0: Interrupt request is masked 1: Interrupt request is not masked
4	EM4	RW	0	Interrupt mask on line 4. 0: Interrupt request is masked 1: Interrupt request is not masked
3	EM3	RW	0	Interrupt mask on line 3. 0: Interrupt request is masked 1: Interrupt request is not masked
2	EM2	RW	0	Interrupt mask on line 2. 0: Interrupt request is masked 1: Interrupt request is not masked
1	EM1	RW	0	Interrupt mask on line 1. 0: Interrupt request is masked 1: Interrupt request is not masked
0	EM0	RW	0	Interrupt mask on line 0. 0: Interrupt request is masked 1: Interrupt request is not masked

11. Cyclic redundancy check calculation unit (CRC)

11.1. Introduction

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

In other applications, CRC technology is mainly used to verify the correctness and integrity of data transmission or data storage.

11.2. CRC main features

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

  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

- Supports 32-bit data input
- Single input/output 32-bit data register
- General-purpose 8-bit register (can be used for temporary storage)
- CRC computation done in 4 AHB clock cycles (HCLK) for the 32-bit data size

11.3. CRC functional description

11.3.1. CRC block diagram

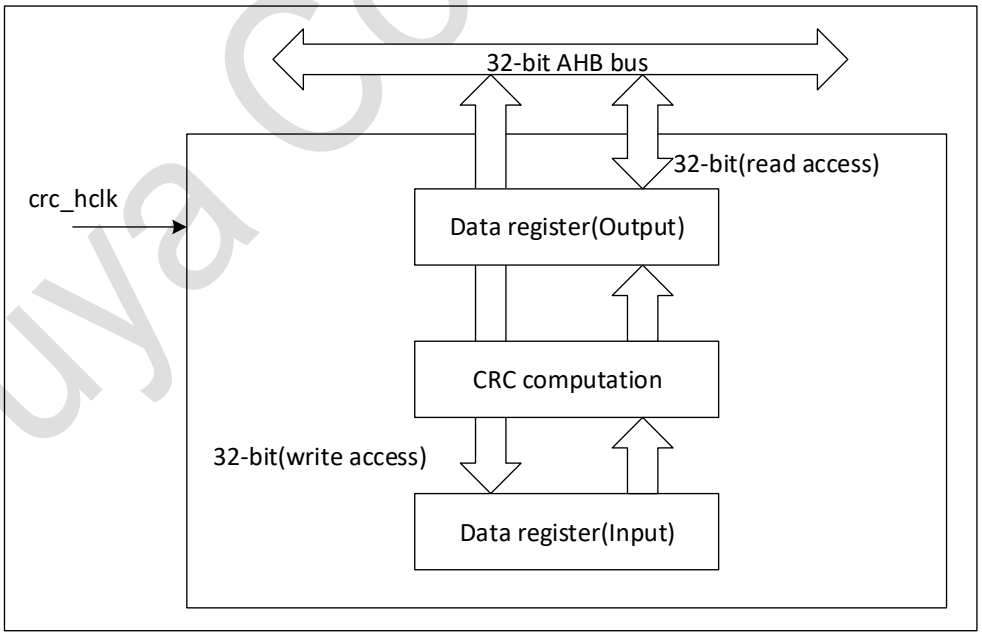


Figure 11-1 CRC calculation unit block diagram

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

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

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

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

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

11.4. CRC registers

11.4.1. CRC data register (CRC_DR)

Address offset: 0x00

Reset value: 0xFFFF FFFF

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

Bit	Name	R/W	Reset Value	Function
31:0	DR	RW	32'hFFFFFFFF	Data register. This register is used to write new data to the CRC calculator It holds the previous CRC calculation result when it is read.

11.4.2. CRC independent data register (CRC_IDR)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

11.4.3. CRC control register (CRC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

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

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

12. Analog-to-digital converters (ADC)

12.1. Introduction

The device has a 12-bit SAR-ADC. The module has a total of up to 13 channels to be measured, including 10 external and 3 internal channels. The ADC internal voltage reference: V_{REFBUF} (1.5 V, 2.048 V, 2.5 V) or the power supply voltage V_{CC}.

The internal channels are: T_{S_VIN}, V_{REFINT}, and $V_{CC}/3$.

A/D conversion of the various channels can be performed in single, continuous, scan or discontinuous mode. The result of the ADC is stored in a left-aligned or right-aligned 16-bit data register.

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

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

Interrupt requests are triggered by the following events: end of sampling, conversion, continuous conversion and Analog watchdog threshold violation (converted voltage exceeds preset limits)

12.2. ADC main features

- High performance
  - 12 bits, 10 bits, 8 bits, and 6 bits resolutions are configurable
  - ADC conversion time: 1.33 μs @ 12-bit (12 MHz)
  - Support self - calibration (initiated by software)
  - Programmable sampling time
  - Programmable data alignment mode (left or right alignment)
- Low-power
  - Reduce PCLK frequency for low power operation while still maintaining proper ADC performance
  - Auto off mode: prevents ADC overrun in applications with low frequency PCLK
- Analog input channels
  - 10 external analog input channels: PA [7: 0] and PB [1: 0]
  - 1 internal temperature sensor channel
  - 1 channel for internal reference voltage (V_{REFINT})
  - 1 channel for internal $V_{CC}/3$
- Start-of-conversion can be initiated
  - By software
  - By hardware triggers with configurable polarity (TIM1)
- Conversion modes
  - Single mode: converts a single channel or can scan a sequence of channels

- Continuous mode: converts selected inputs continuously
- Discontinuous mode: converts selected inputs once per trigger

■ Interrupt generation

- At the end of sampling
- End of conversion
- End of sequence conversion
- Analog watchdog
- Overflow event

■ Analog watchdog

12.3. ADC functional description

12.3.1. ADC block diagram

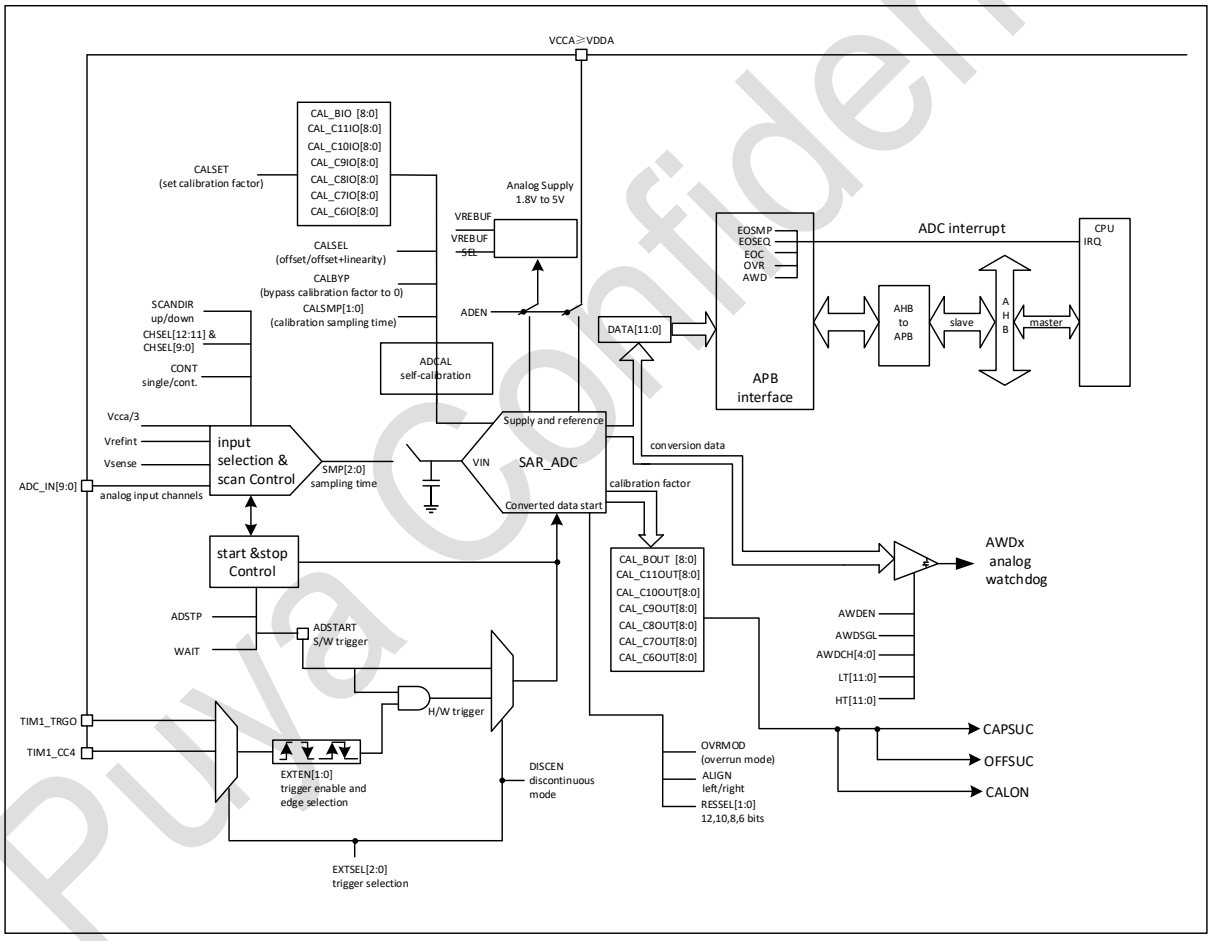


Figure 12-1 ADC channel with analog switch

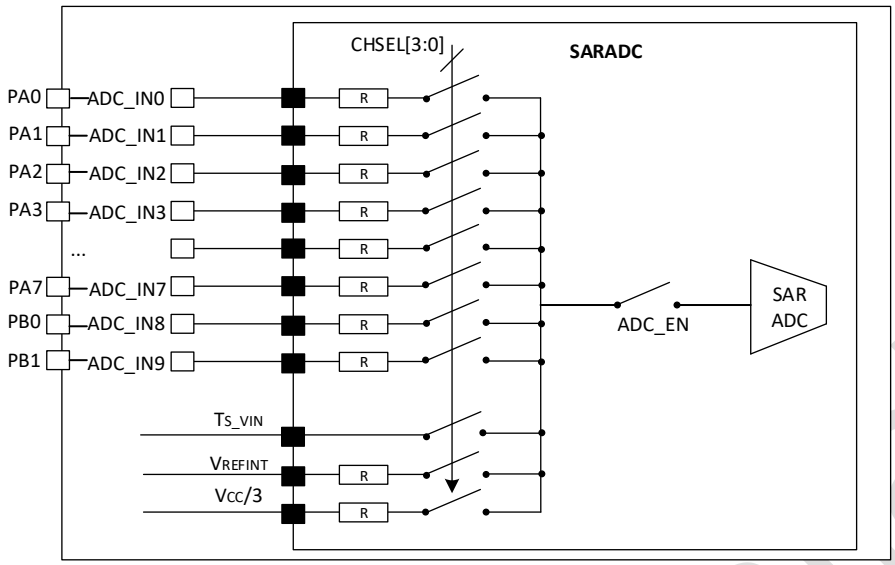


Figure 12-2 ADC channel with analog switch

12.3.2. Calibration (ADCAL)

The ADC has a calibration feature. During the procedure, the ADC calculates a calibration factor which is internally applied to the ADC until the next ADC power-off. The application cannot use the ADC module during the ADC calibration and until the calibration is complete.

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

The accuracy operation includes software accuracy.

ADC software calibration

The software sets ADCAL = 1 to start the calibration. The calibration can only be started when the ADC is not enabled (ADEN = 0), and only the system clock is supported as the clock of the ADC.

When the calibration is complete, the ADCAL is cleared to 0 by the hardware.

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

Software procedure to calibrate the ADC:

- Confirm ADEN = 0, CKMODE Select system clock
- Set ADCAL=1
- Wait until ADCAL=0

12.3.3. ADC on-off control (ADEN)

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

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

Follow this procedure to enable the ADC:

1. Set ADEN=1 in the ADC_CR register.

The ADC conversion is also initiated by setting ADSRART or (if initiation is triggered) by an external trigger event that triggers the initiation on.

Follow this procedure to disable the ADC:

Check that ADSTART=0 in the ADC_CR register to ensure that no conversion is ongoing. If ADSTART = 0 and ADEN = 1, the ADC can be disabled by setting 1 for ADDIS in ADC_CR. If desired, set the ADSTP in the ADC_CR register to 1 to stop the ongoing ADC conversion and wait for the ADSTP to be cleared to 0 by hardware (cleared to 0 indicates that the conversion stop is complete). The ADSTART, ADSTP, ADEN and ADDIS registers are configured in accordance with the following principles:

- When the software is not configured with ADEN = 1, any bit of ADSTART/ADSTP/ADDIS cannot be configured to be 1 (hardware masking), that is, when ADEN = 1 is configured, ADSTART/ADSTP/ADDIS must be 0, or when ADSTART/ADSTP/ADDIS is configured to be 1, ADEN = 1;
- When ADSTART is 0, the software cannot set ADSTP (hardware masking);
- When ADEN = 1 and ADSTART is 0, the software can configure ADDIS = 1 (hardware shielding);
- If ADEN = 1, setting ADDIS (ADSTART = 0 is the premise) will clear ADEN; After ADEN is cleared, ADDIS is also cleared;
- The software clears ADEN and then re-enables ADEN and ADSTART at 8 ADC_CLK cycles.

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

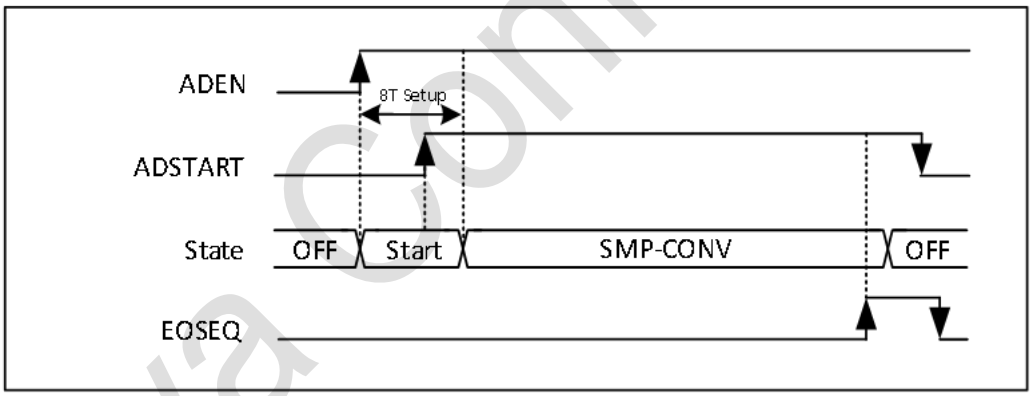


Figure 12-3 Enable/Disable ADC

12.3.4. ADC clock

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

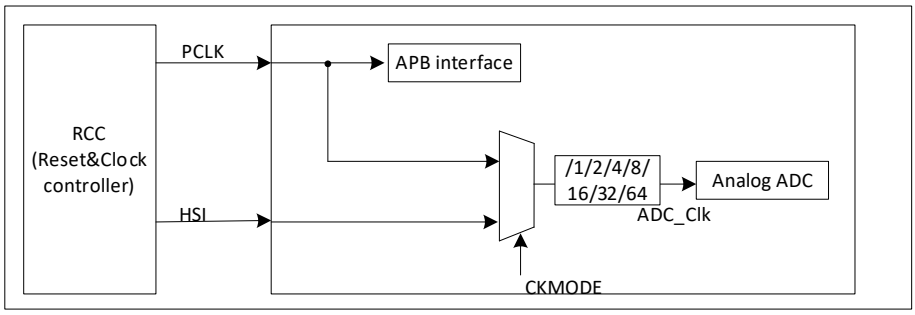


Figure 12-4 ADC clock scheme

Table 12-1 ADC clock source and frequency division coefficient table

ADC clock source	CKMODE[3:0]	dividing factor
PCLK	0000	1
	0001	2
	0010	4
	0011	8
	0100	16
	0101	32
	0110	64
	0111	/
HSI	1000	1
	1001	2
	1010	4
	1011	8
	1100	16
	1101	32
	1110	64
	1111	/

Notes:

1. When ADC_CLK > PCLK, it is recommended to enable WAIT mode;

2. ADC_CLK > PCLK clock under different conversion accuracy conditionsRESSEL [1: 0]= 00,ADC_CLK < 4 * PCLK; RESSEL [1: 0]= 01,ADC_CLK < 3PCLK; RESSEL [1: 0]= 10,ADC_CLK < 3PCLK; RESSEL [1: 0]= 11,ADC_CLK < 2PCLK.

12.3.5. Configuring the ADC

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

For the ADC_IER, ADC_CFGRi, ADC_SMPR, ADC_TR, and ADC_CCR registers, the software must be rewritten with the ADC on (ADEN = 1) and no transition period (ADSTART = 0). ADC_CHSELR is written with ADEN = 0 and ADSTART = 0.

ADC_CHSELR is written with ADEN = 0 and ADSTART = 0.

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

12.3.6. Channel selection (CHSEL, SCANDIR)

There are up to 13 multiplexed channels:

- 10 analog inputs from GPIO pins (ADC_IN0...ADC_IN9)

- 3 internal analog inputs (Temperature Sensor, Internal Reference Voltage, V_{BAT} channel)

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

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

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

- SCANDIR = 0: forward scan Channel 0 to Channel 12
- SCANDIR = 1: backward scan Channel 10 to Channel 0

The temperature sensor is connected to channel ADC_IN10 (T_{S_VIN}). The internal voltage reference V_{REFINT} is connected to channel ADC_IN11. VCC/3 is connected to ADC1_IN12.

12.3.7. Programmable sampling time (SMP)

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

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

The number of ADC clocks used by the ADC to sample the input voltage can be modified with the SMP [2: 0] bit in the ADC_SMPR register. This programmable sampling time is common to all channels. If required by the application, the software can change and adapt this sampling time between each conversions.

The total conversion time is calculated as follows:

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

For example:

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

t_{CONV} = (3.5 + 12.5) x ADC clock cycles = 16 x ADC clock cycles = 1.33 μ s

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

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

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

Conversion is started by either:

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

Inside the sequence, after each conversion is complete:

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

After the sequence of conversions is complete:

- The EOSEQ (end of sequence) flag is set

- An interrupt is generated if the EOSIE bit is set.

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

Note: To convert a single channel, program a sequence with a length of 1. To switch the conversion mode, you must first turn off ADEN (use ADDIS to turn off ADEN).

12.3.9. Continuous conversion mode (CONT=1)

In continuous conversion mode, when a software or hardware trigger event occurs, the ADC performs a sequence conversion. Converts all channels once and automatically restarts performing the same sequence conversion. When CONT = 1 in the register ADC_CFGR1, the ADC is selected to the continuous conversion mode. Conversion is started by either:

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

Inside the sequence, after each conversion is complete:

- The converted data are stored in the 16-bit ADC_DR register.
- The EOC (end of conversion) flag is set
- An interrupt is generated if the EOCIE bit is set.
- After the sequence of conversions is complete:
- The EOSEQ (end of sequence) flag is set
- An interrupt is generated if the EOSEQIE bit is set

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

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

The ADC cannot be in the discontinuous conversion mode and the continuous conversion mode at the same time, in which case (DISCEN = 1, CONT = 1), it behaves as a single conversion mode. To switch the mode, you must first turn off ADEN (turn off ADEN with ADDIS).

12.3.10. Discontinuous conversion mode (DISCEN = 1)

This mode is turned on by setting the DISCEN bit in the ADC_CFGR1 register.

In this mode (DISCEN = 1), hardware-triggered events or software are required to initiate each channel transition defined in a sequence.

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

For example:

DISCEN = 1, the channels to be converted are: 0, 3, 7, 10:

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

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

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

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

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

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

- ...

DISCEN = 0, the channels to be converted are: 0, 3, 7, 10:

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

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

- Any trigger event restarts the full sequence transition.

Note: It is not possible to have the ADC in both continuous conversion mode and non-continuous conversion mode. In this configuration (DISCEN = 1, CONT = 1), it behaves as a single conversion mode. To switch the mode, you must first turn off ADEN (turn off ADEN with ADDIS).

12.3.11. Starting conversions (ADSTART)

Software starts ADC conversions by setting ADSTART=1.

When ADSTART is set, the conversion:

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

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

The ADSTART bit is cleared by hardware.

- Single transition mode is triggered by software (CONT = 0, EXTSEL = 0x0)
  - End of conversion sequence (EOSEQ = 1)
- Discontinuous conversion mode is triggered by software (CONT = 0, DISCEN = 1, EXTSEL = 0x0)
  - At end of conversion (EOC=1)
- In all cases (CONT = X, EXTSEL = X)
  - After the software invokes and executes the ADSTP procedure

Note: In continuous mode (CONT=1), the ADSTART bit is not cleared by hardware when the EOSEQ flag is set because the sequence is automatically relaunched. When hardware trigger is selected in single mode (CONT=0 and EXTEN = 0x01), ADSTART is not cleared by hardware when the EOSEQ flag is set. This avoids the need for software to reset the ADSTART bit and ensures that no hardware trigger events are missed. Software trigger mode, the START control bit still keeps the 1 state, and software trigger again is invalid.

12.3.12. ADC timing

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

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

$t_{ADC} = t_{SMPL} + t_{SAR}$ = 291.7ns|min + 1041.625 ns|12bit = 1.33 µs|min (for fADC_CLK = 12 MHz)

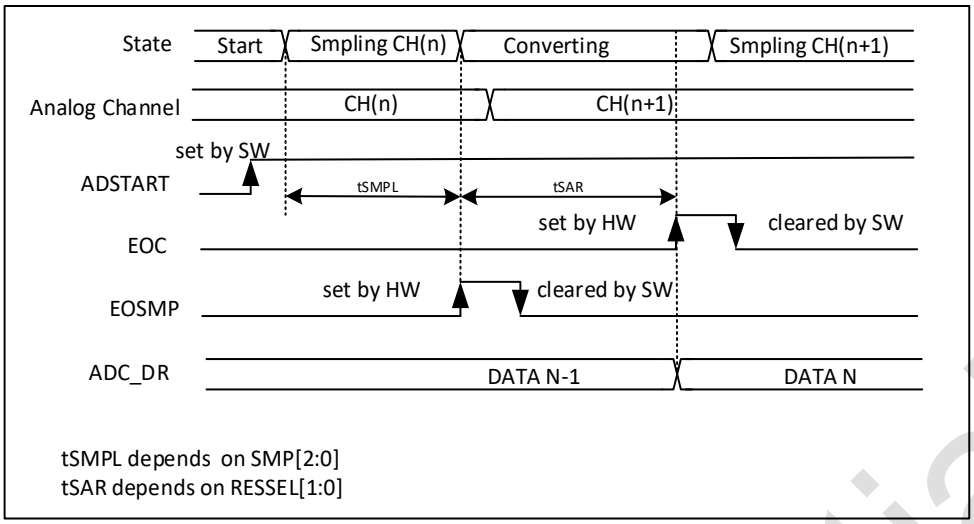


Figure 12-5 analog-to-digital conversion timing

12.3.13. 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.

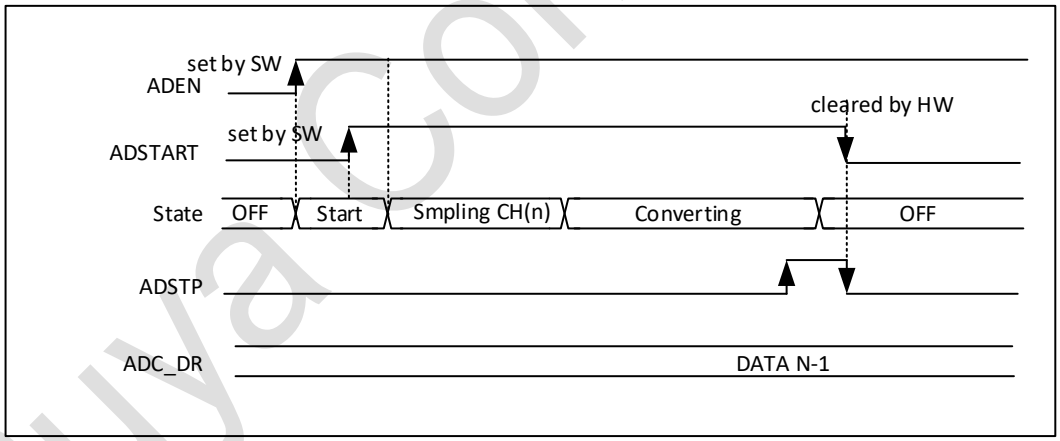


Figure 12-6 Stop timing

12.3.14. STOP ADEN (ADDIS)

The software can decide to clear ADEN bit by setting ADDIS=1 in the ADC_CR register. This will reset the ADC operation and the ADC will be idle, ready for a new operation.

When ADDIS is set to 1 by software (ADSTART = 0), ADEN is cleared

Once this procedure is complete, the ADDIS and ADEN bits are both cleared by hardware.

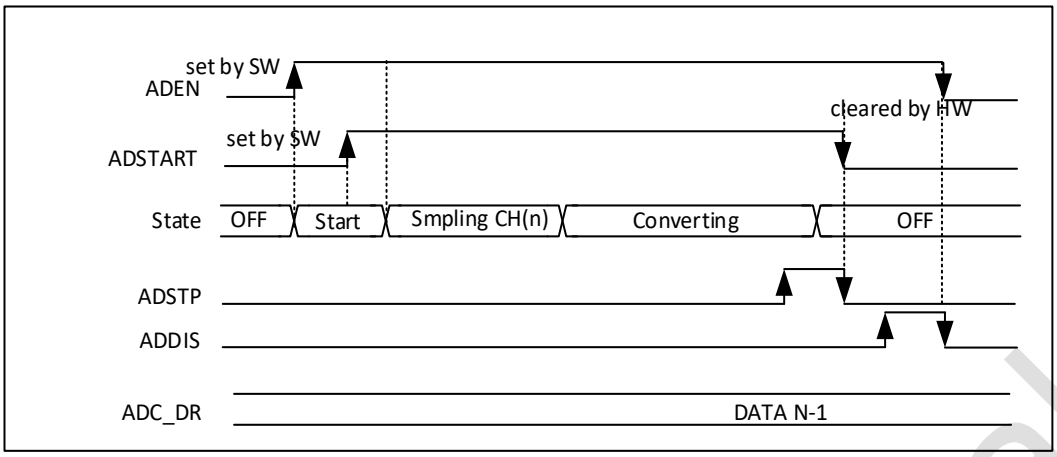


Figure 12-7 ADDSI timing

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

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

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

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

Table 12-2 External triggers

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

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

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

Table 12-3 External triggers

Name	Source	EXTSEL[2:0]
EXT0	TIM1_TRGO	000
EXT1	TIM1_CC4	001
EXT2	TIM14_CC1	010
EXT3	Reserved	011
EXT4	Reserved	100
EXT5	Reserved	101
EXT6	Reserved	110
EXT7	Reserved	111

Note: External trigger polarity cannot be changed during transition.

12.3.15.1. Quick transition mode

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

speed up conversion times. The conversion result is also 12 bits wide, and the lower bits are supplemented by 0.

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

Table 12-4 Conversion resolution vs. Conversion time

RESSEL [1:0]	t_{SAR} (ADC Clock Cycle)	t_{SAR}(ns) @ fADC = 12 MHz	t_{SMP} (ADC Clock Cycle)	t_{ADC}(t_{SMP} = 3.5) (ADC Clock Cycle)	t_{SAR}(ns) @ fADC = 12 MHz
12	12.5	1041.6 ns	3.5	16	1333.3 ns
10	10.5	833.3 ns	3.5	14	1166.2 ns
8	8.5	708.3 ns	3.5	12	1000 ns
6	6.5	541.65 ns	3.5	10	833.3 ns

12.3.15.2. End of conversion / end of sampling phase

The ADC indicates each end of conversion (EOC) event.

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

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

12.3.15.3. End of conversion sequence (EOSEQ flag)

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

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

12.3.15.4. Sampling time diagram

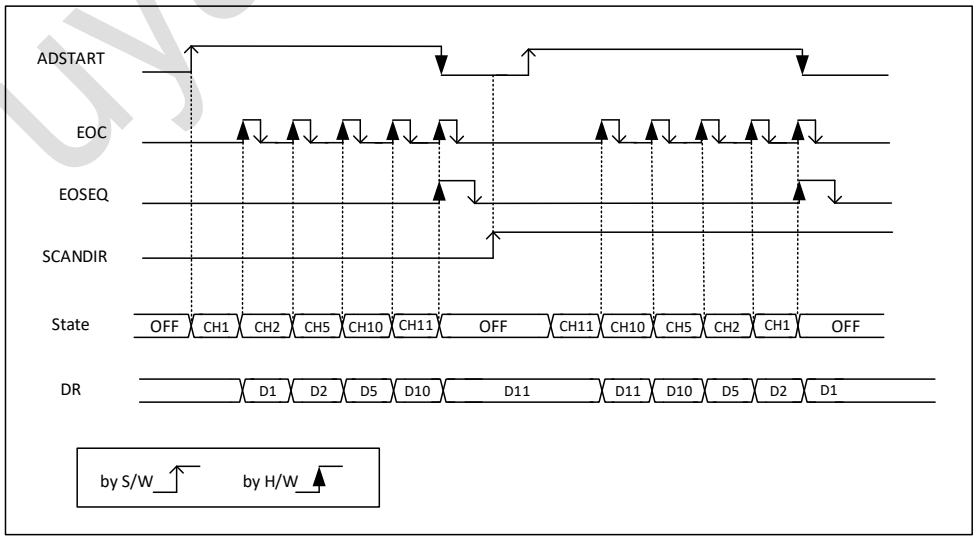


Figure 12-8 Single conversions of a sequence, software trigger

1. EXTEN = 0x0, CONT = 0
2. CHSEL = 0x20601, WAIT = 0
3. Under the condition of high frequency division of ADC_CLK, the START software triggers the next conversion. You need to check the status of START (START: 1-> 0, the current conversion is over), and the next software trigger can be effective

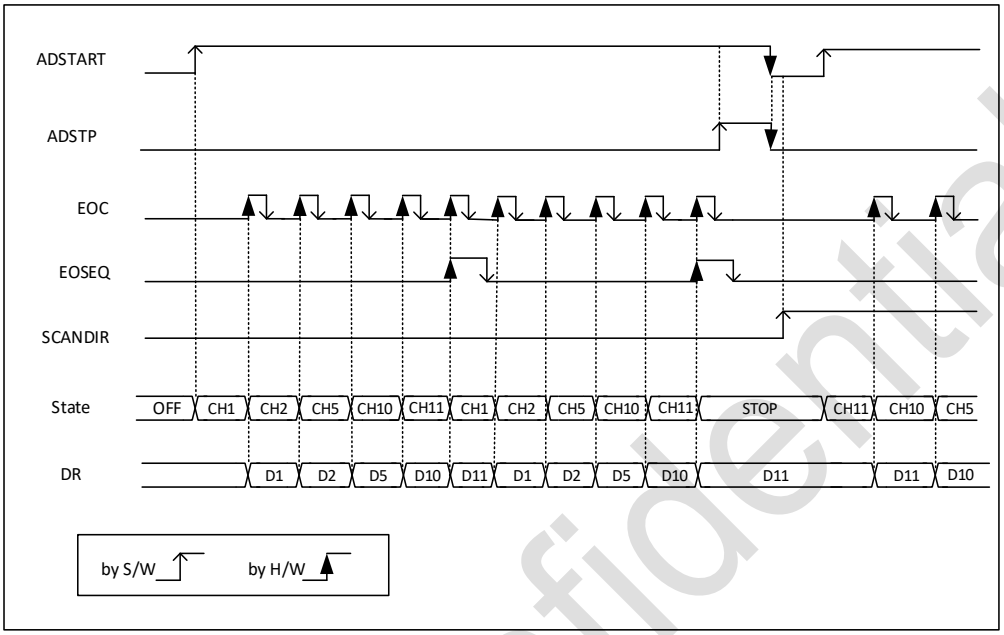


Figure 12-9 Continuous conversion of a sequence, software trigger

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

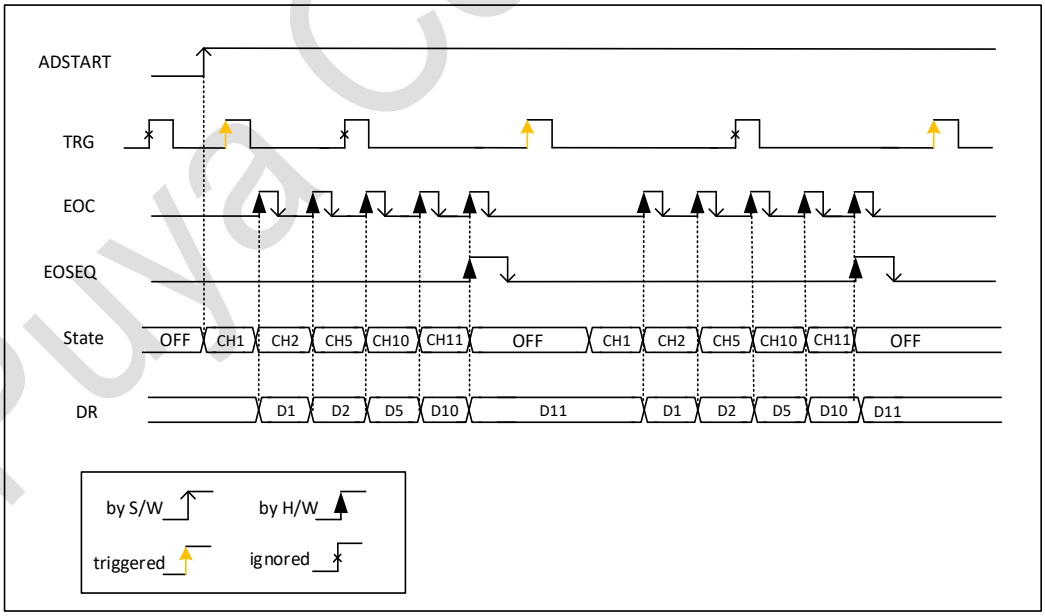


Figure 12-10 Single conversions of a sequence, hardware trigger

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

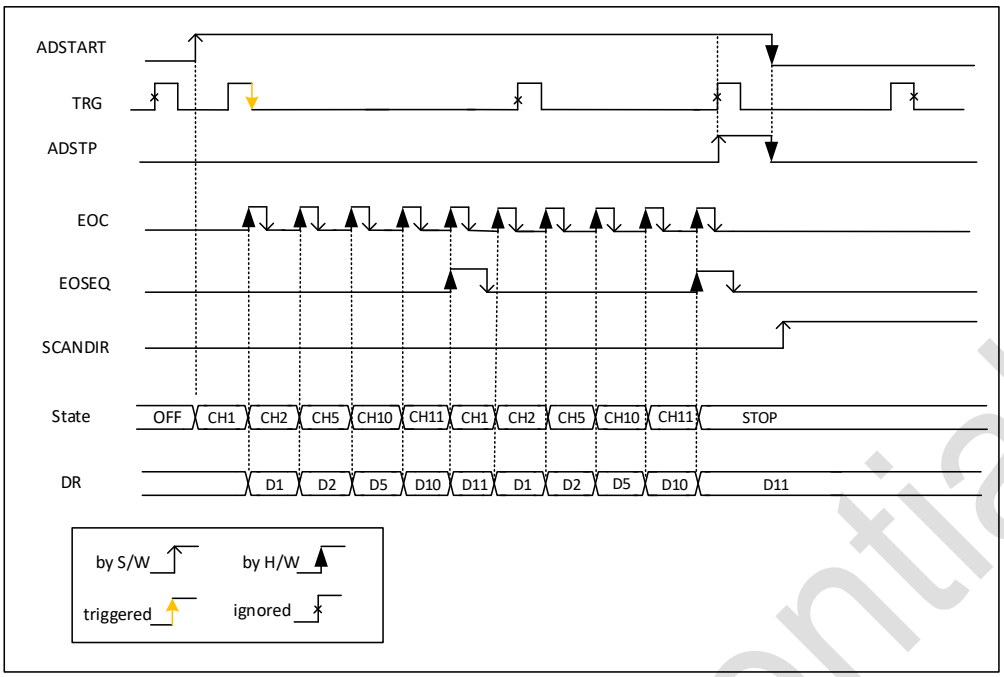


Figure 12-11 Continuous conversions of a sequence, hardware trigger

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

12.3.16. Data management

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

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

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

Table 12-5 Data registers and data alignment

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

12.3.16.2. ADC overrun (OVR, OVRMOD)

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

The OVR flag is set in the ADC_ISR register if the EOC flag is still at ‘1’ at the time when anew conversion completes. An interrupt can be generated if the OVRIE bit is set in the ADC_IER register.

When an overshoot event occurs, the ADC will continue to operate and continue to convert unless the software decides to stop and reset this sequence conversion. The ADC conversion can be stopped by setting ADSTP in the ADC_CR register to 1 by software, and the OVR flag can be cleared by software writing 1.

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

- OVRMOD=0
  - An overrun event preserves the data register from being overwritten: the old data is maintained, and the new conversion is discarded. If the OVR remains 1, subsequent transformations will be executed but the results will be discarded (after the OVR occurs, Read DR cannot clear the OVR).
- OVRMOD=1
  - The data register is overwritten with the last conversion result and the previous unread data is lost. If the OVR remains 1, the subsequent conversion is performed and the ADC_DR register stores the result value of the latest conversion (after the OVR occurs, the Read DR cannot clear the OVR).

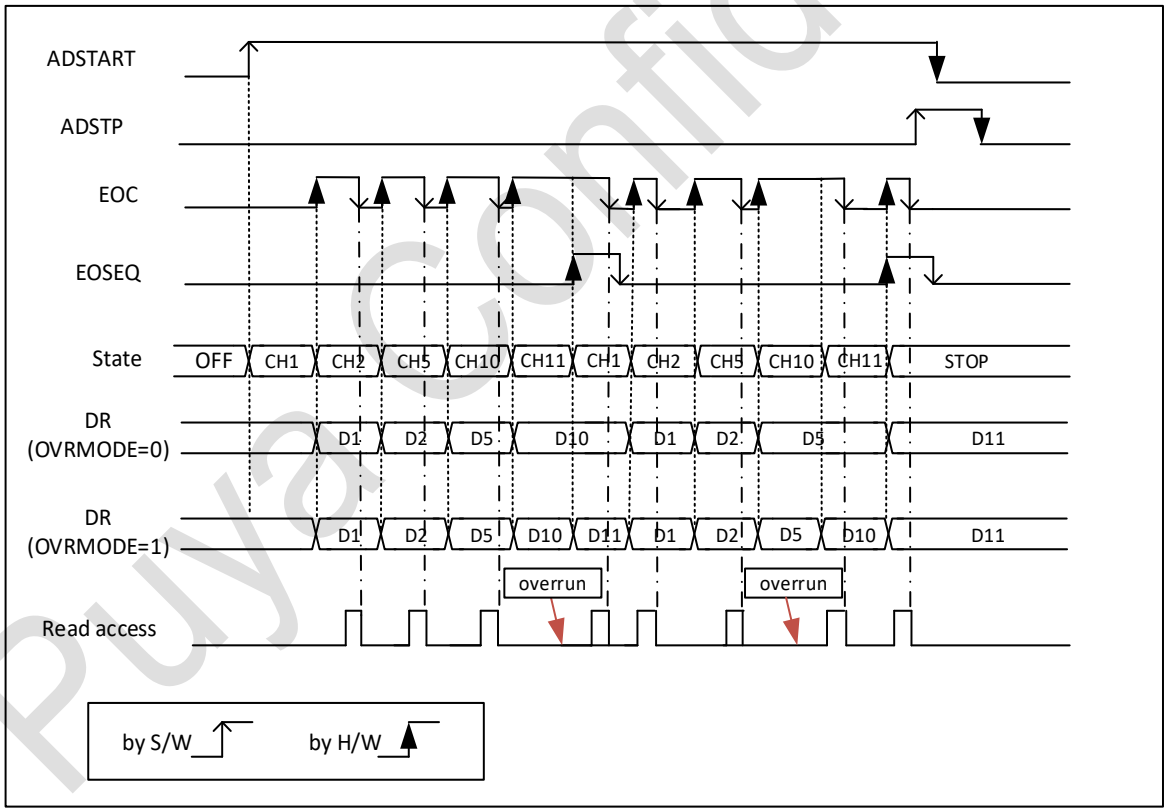


Figure 12-12 Overload

12.3.17. Low-power features

12.3.17.1. Wait mode conversion

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

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

Note: Any hardware triggers which occur while a conversion is ongoing or during the wait time preceding the read access are ignored. When ADC_CLK > PCLK, it is recommended to enable WAIT mode;

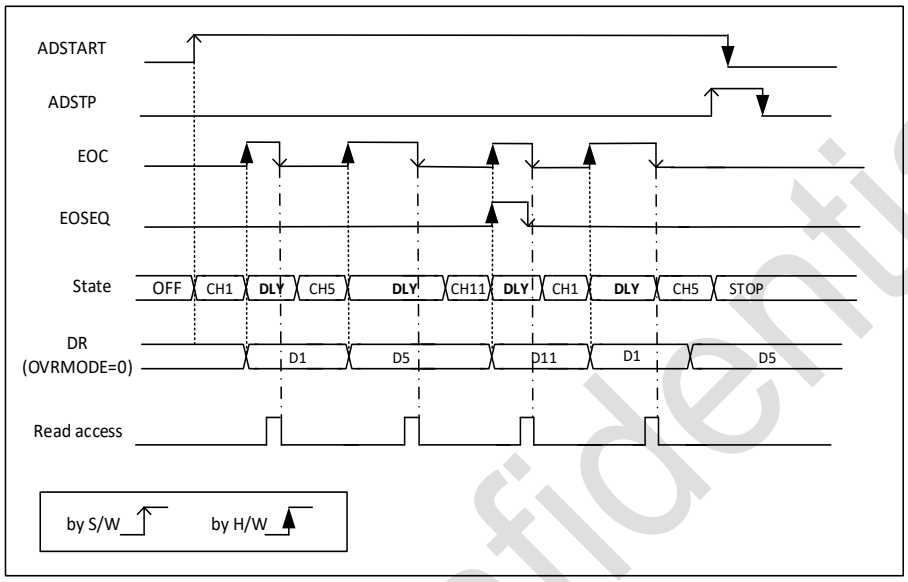


Figure 12-13 Wait mode conversion

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

12.3.18. Analog watchdog

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

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

Table 12-6 Simulated watchdog comparison

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

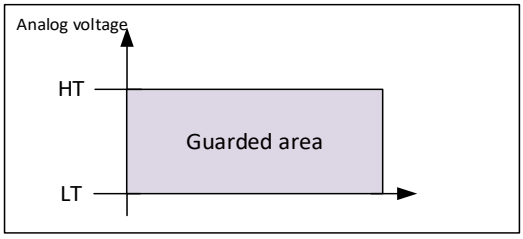


Figure 12-14 Analog watchdog guarded area

Table 12-7 Analog watchdog channel selection

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

12.3.18.1. ADC_AWD_OUT signal output generation

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

ADC_AWD_OUT is activated when the associated analog watchdog is enabled:

- ADC_AWD_OUT is set when a guarded conversion is outside the programmed thresholds.
- After the transition of the next channel selected by AWDCH is completed, ADC_AWD_OUT is reset within the programmed threshold. It remains at 1 if the next guarded conversions are still outside the programmed thresholds.
- ADC_AWD_OUT is also reset when disabling the ADC (when setting ADDIS to 1). Note that stopping conversions (ADSTP set to 1), might clear the ADC_AWDx_OUT state.

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

The ADC_AWD_OUT signal is generated by the PCLK domain.

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

12.3.19. Temperature sensor and internal reference voltage

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

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

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

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

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

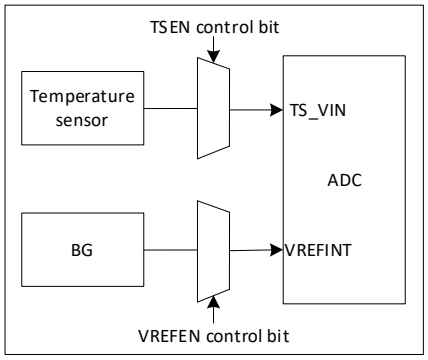


Figure 12-15 Temperature sensor and reference voltage channel

Reading the temperature:

1. Select the ADC_IN10 input channel
2. Select an appropriate sampling time specified in the device datasheet (T_{S_temp}).
3. Set the TSEN bit in the ADC_CCR register to wake up the temperature sensor from power down mode and wait for its stabilization time.
4. Start the ADC conversion by setting the ADSTART bit in the ADC_CR register (or by external trigger).
5. Read VSENSE conversion data from ADC_DR register
6. Calculate the actual temperature using the following formula:

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

TSCAL2 represents the calibration value of the 105 ℃ temperature sensor

TSCAL1 represents the calibration value of the 30 ℃ temperature sensor

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

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

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

The following formula can give the voltage value of the true V_{CC}:

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

Use the Vcc voltage to calculate Vchannnel:

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

V_{REFINT} is fixed at 1.2 V;

VCHANNEL is the channel voltage;

ADC_DATA is the conversion data in ADC_DR;

4096 is represented as 12 bits.

The micro-controlled V_{CC} power supply is easily affected or the magnitude of this value is not very clear. The internal voltage reference (V_{REFINT}) and the calibration data obtained by the V_{CC} = 3.3 V ADC during production can be used to evaluate the true V_{CC} voltage level.

The following formula can give the voltage value of the true V_{CC}:

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

V_{REFINT} is fixed at 1.2 V;

VCHANNEL is the channel voltage;

ADC_DATA is the conversion data in ADC_DR;

4096 is represented as 12 bits.

12.3.20. ADC interrupts

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

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

Separate interrupt enable bits are available for flexibility.

Table 12-8 ADC Interrupts

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

12.4. ADC registers

12.4.1. ADC interrupt and status register (ADC_ISR)

Address offset:0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	AWD	Res	Res	OVR	EOSEQ	EOC	EOSMP	Res.
								RC_W1			RC_W1	RC_W1	RC_W1	RC_W1	

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

Bit	Name	R/W	Reset Value	Function
				0: No analog watchdog event occurred (or the flag event was already cleared by software) 1: Analog watchdog event occurred
6:5	Reserved	-	-	Reserved
4	OVR	RC_W1	0	ADC overrun This bit is set by hardware when an overrun occurs. When the EOC flag is set, a new transition has been completed. It is cleared by software writing 1 to it. 0: No overrun occurred (or the flag event was already acknowledged and cleared by software) 1: Overrun has occurred
3	EOSEQ	RC_W1	0	End of sequence flag This bit is set by hardware at the end of the conversion of a sequence of channels selected by the CHSEL bits. This bit is cleared by writing 1 0: Conversion sequence not complete (or the flag event was already acknowledged and cleared by software) 1: Conversion sequence complete
2	EOC	RC_W1	0	End of conversion flag This bit is set by hardware at the end of each conversion of a channel when a new data result is available in the ADC_DR register. It is cleared by software writing 1 to it or by reading the ADC_DR register. 0: Conversion sequence not complete (or the flag event was already acknowledged and cleared by software) 1: Channel conversion completed
1	EOSMP	RC_W1	0	This bit is set by hardware during the conversion, at the end of the sampling phase. It is cleared by software by programming it to '1'. 0: Not at the end of the sampling phase (or the flag event was already acknowledged and cleared by software) 1: End of sampling phase reached
0	Reserved	-	-	Reserved

12.4.2. ADC interrupt enable register (ADC_IER)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	AWDIE	Res.	Res.	OVRIE	EOSE-QIE	EO-CIE	EOSMPIE	Res.
								RW			RW	RW	RW	RW	

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

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

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

12.4.3. ADC control register (ADC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AD-CAL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res.	Res	Res
RS															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res.	Res.	Res.	AD-STP	Res.	AD-START	AD-DIS	ADEN
											RS		RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RS	0	This bit is set by software to start the calibration of the ADC. It is cleared by hardware after calibration is complete. 0: Calibration complete 1: Write 1 to calibrate the ADC. Read at 1 means that a calibration is in progress. Note: The software cannot write 0 when writing 1, and the hardware is cleared.
30:5	Reserved	-	-	Reserved
4	ADSTP	RS	0	ADC stop conversion command This bit is set by software to stop and discard an ongoing conversion (ADSTP Command). It is cleared by hardware when the conversion is effectively discarded and the ADC is ready to accept a new start conversion command. 0: No ADC stop conversion command ongoing 1: Write 1 to stop the ADC. Read 1 means that an ADSTP command is in progress. Software writing that the bit is 0 is an invalid operation.
3	Reserved	-	-	Reserved
2	ADSTART	RS	0	ADC start conversion command This bit is set by software to start ADC conversion. Depending on the EXTEN [1:0] configuration bits, a conversion either starts immediately (software trigger configuration) or once a hardware trigger event occurs (hardware trigger configuration). If this bit is cleared by hardware:

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

12.4.4. ADC configuration register 1 (ADC_CFGR1)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	AWDCH				Res	Res.	AWDEN	AWDSGL	Res.	Res	Res.	Res.	Res	DIS-CEN
		RW	RW	RW	RW			RW	RW						RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	WAIT	CONT	OV-RMOD	Res.	Res.	Res.	EXTSEL			ALIGN	RES_SEL		SCAN-DIR	Res.	Res.
	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:26	AWDCH[3:0]	RW	0	Analog watchdog channel selection. These bits are set and cleared by software. They select the input channel to be guarded by the analog watchdog. 0000: ADC analog input Channel 0 0001: ADC analog input Channel 1 … 1011: ADC analog input Channel 11 1100: ADC analog input Channel 12 Others: Reserved Note: The channel selected by the AWDCH[3:0] bits must be also set into the CHSELR register. Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
25:24	Reserved	-	-	Reserved
23	AWDEN	RW	0	Analog watchdog enable This bit is set and cleared by software.

				0: Analog watchdog disabled 1: Analog watchdog enabled Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
22	AWDSGL	RW	0	Enable the watchdog on a single channel or on all channels By setting and clearing this bit, the software can enable or disable the analog watchdog on the channel set by the AWDCH [3: 0] bit or on all channels. 0: Analog watchdog enabled on all channels 1: Enable analog watchdog on one channel (AWDCH [3: 0] configure which channel) Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
21:17	Reserved	-	-	Reserved
16	DISCEN	RW	0	Discontinuous mode enabled. This bit is set and cleared by software to enable/disable discontinuous mode. 0: Discontinuous mode disabled 1: Discontinuous mode enabled It is not possible to have both discontinuous mode and continuous mode enabled: it is forbidden to set both bits DISCEN=1 and CONT=1. Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
15	Reserved	-	-	Reserved
14	WAIT	RW	0	Waiting for conversion mode. This bit can be set and cleared by software, enabling/disabling waiting transition mode. 0: Wait conversion mode off 1: Wait for conversion mode to turn on Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
13	CONT	RW	0	Single / continuous conversion mode This bit is set and cleared by software. If it is set, conversion takes place continuously until it is cleared. It is not possible to have both discontinuous mode and continuous mode enabled: it is forbidden to set both bits DISCEN=1 and CONT=1. Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
12	OVRMOD	RW	0	Overload management mode. This bit is set and cleared by software and configure the way data overruns are managed. 0: ADC_DR register is preserved with the old data when an overrun is detected. 1: ADC_DR register is overwritten with the last conversion result when an overrun is detected. Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
11:10	EXTEN[1:0]	RW	00	External trigger enable and polarity selection. The software can set and clear this bit, select the drive polarity and enable the drive. 00: Hardware trigger detection disabled (conversions can be started by software) 01: Hardware trigger detection on the rising edge 10: Hardware trigger detection on the falling edge 11: Hardware trigger detection on both the rising and falling edges Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
9	Reserved	-	-	Reserved
8:6	EXTSEL[2:0]	RW	000	External driver selection (only allow software to write these bits if ADSART = 0 (make sure there is no conversion in progress)) This bit selects the external event that triggers the start of the transition 000: TRG0(TIM1_TRGO)

				001: TRG1(TIM1_CC4) 010: TRG2(TIM14_CC1) 011: TRG3 (reserved) 100: TRG4 (reserved) 101: TRG5 (reserved) 110: TRG6 (reserved) 111: TRG (reserved)
5	ALIGN	RW	0	Data alignment. This bit is set and cleared by software to select right or left alignment. 0: Right alignment 1: Left alignment Note: Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
4:3	RESSEL[1:0]	RW	00	Data resolution. These bits are written by software to select the resolution of the conversion. 00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit Note: Software is allowed to write these bits only when ADEN=0.
2	SCANDIR	RW	0	Scan sequence direction This bit is set and cleared by software to select the direction in which the channels will be scanned in the sequence. 0: Upward scan (from channel 0 to 12) 1: Backward scan (from channel 12 to 0) Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
1:0	Reserved	-	-	Reserved

12.4.5. ADC sampling time register (ADC_SMPR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.													SMP0		
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:0	SMP[2:0]	RW	0	Sampling time selection The software can configure the sample time for this bit to select channel x. 000: 3.5 ADC clock cycles 001: 5.5 ADC clock cycles 010: 7.5 ADC clock cycles 011: 13.5 ADC clock cycles 100: 28.5 ADC clock cycles 101: 41.5 ADC clock cycles 110: 134.5 ADC clock cycles 111: 239.5 ADC clock cycles Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).

12.4.6. ADC watchdog threshold register (ADC_TR)

Address offset: 0x20

Reset value: 0x0FFF 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:16	HT[11:0]	RW	0xFFF	Analog watchdog higher threshold These bits are written by software to define the higher threshold for the analog watchdog. Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).
15:12	Reserved	-	-	Reserved
11:0	LT[11:0]	RW	0x000	Analog watchdog lower threshold These bits are written by software to define the lower threshold for the analog watchdog. Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).

12.4.7. ADC channel selection register (ADC_CHSELR)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	CHSEL12	RW	0	Channel 12 ($V_{CC/3}$) selection 0: Channel is not selected 1: Channel is selected Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).
11	CHSEL11	RW	0	Channel 11 (V_{REFINT}) selection 0: Channel is not selected 1: Channel is selected Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).
10	CHSEL10	RW	0	Channel 10 (TS_VIN) selection 0: Channel is not selected 1: Channel is selected Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).
9: 0	CHSELx	RW	0x0000	Channel selection These bits are written by software and define which channels are part of the sequence of channels to be converted. 0: Do not select input channel-x 1: Select input channel-x

				Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).

12.4.8. ADC digital register(ADC_DR)

Address offset: 0x40

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	DATA[15:0]	R	0	Converted data This bit is read-only. The conversion result of the last converted channel is placed in this register. The data are left- or right-aligned.

12.4.9. ADC calibration configuration and status register (ADC_CCSR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CALON.	CAPSUC	OFFSUC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R	RC_W1	RC_W1													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALSET	CALBYP	CALSMP		CALSEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW		RW											

Bit	Name	R/W	Reset Value	Function
31	CALON	R	0	Calibration in progress flag bit indicates that ADC calibration is in progress. 1: ADC calibration is in progress 0: ADC calibration has ended or has not started
30	CAPSUC	RC_W1	0	Capacitance calibration status bit. Indicates whether the ADC capacitance calibration is successful. Hardware set 1; The software writes 1 and sets 0; CALON = 0, CALSEL = 0, CAPSUC = 1: Invalid status CALON = 0, CALSEL = 0, CAPSUC = 0: CAPs calibration was not performed CALON = 0, CALSEL = 1, CAPSUC = 1: ADC CAPs Calibration successful CALON = 0, CALSEL = 1, CAPSUC = 0: ADC CAPs Calibration failed Note: The capacitance calibration is only calibrated once, regardless of whether the C11 ~ C6 calibration is successful or not.
29	OFFSUC	RC_W1	1'b0	Offset calibration status bit. Indicates whether the ADC offset calibration was successful. Hardware set 1; The software writes 1 and sets 0; CALON = 0, CALSEL = 0, OFFSUC = 0: ADC OFFSET Calibration failed CALON = 0, CALSEL = 0, OFFSUC = 1: ADC OFFSET calibration successful

Bit	Name	R/W	Reset Value	Function
				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	RW	0	Calibration factor selection Software set (before ADCAL = 0, i.e. before calibration), hardware cleared. 1: Set CAL_CXIN data as final calibration data 0: Close the path from CAL_CXIN to CAL_CXOUT, and select the result generated internally by the calibration circuit.
14	CALBYP	RW	0	Calibration factor bypass. Software set (before ADCAL = 0, i.e. before calibration), hardware cleared. 1: Mask automatic calibration and set the output result of CALSET calibration to CAL_CXOUT 0: Select automatic calibration or set the output result of CALSET calibration to CAL_CXOUT
13:12	CALSMP	RW	0	Calibration sampling time selection Configure the number of clock cycles of the sampling phase of the calibration based on the following information: 00: 1 ADC clock cycles 01: 2 ADC clock cycles 10: 4 ADC clock cycles 11: 8 ADC clock cycles The longer the SMP is configured during calibration, the more accurate the calibration result is, but this configuration will bring the problem of extended calibration period
11	CALSEL	RW	0	Calibration content selection bit for selecting what needs to be calibrated 1: Calibrate OFFSET and Linearity 0: Calibrate OFFSET only
10:0	Reserved	-	-	Reserved

12.4.10. ADC common configuration register (ADC_CCR)

Address offset: 0x308

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	-
23	TSEN	RW	0	Temperature sensor enable. This bit is set and cleared by software to enable/disable the temperature sensor. 0: Disabled 1: Enabled Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
22	VREFEN	RW	0	Reference Vrefint enable bit that can be set and cleared by software to enable/turn off the reference V_{REFINT}

				0: Disabled 1: Enabled Software is allowed to write these bits only when AD-START=0 (which ensures that no conversion is ongoing).
21:8	Reserved	-	-	-
7:6	Vrefbuff_sel	RW	2'b0	V_{REFBUF} output voltage selection 00: 0.6 V 01:1.5 V 10: 2.048 V 11: 2.5 V
5	Vref_buffere	RW	1'b0	V_{REFBUF} enable The software writes 0 and sets 0, writes 1 and sets 1, 0: V_{REFBUF} disabled 1: V_{REFBUF} enabled
4: 0	Reserved	-	-	-

13. Comparator (COMP)

13.1. Introduction

The device integrates two general-purpose comparators (COMP), namely COMP1 and COMP2. The COMP1/2 module can be used as a separate module or in combination with timer.

The comparator may be used as follows:

- Triggered by analog signal to wake-up function from low-power mode
- Analog signal conditioning
- Cycle by cycle current control loop when comparators are connected with PWM output from timer.

13.2. COMP main features

- Each comparator has configurable positive or negative input for flexible voltage selection:
  - Multiple I/O pins
  - Power supply V_{CC} and 64 fractional values provided by voltage division (1/64, 2/64... 64/64)
  - Internal reference voltage is 0.6V, 1.5 V, 2.048 V or 2.5 V, and 64 submultiple values (1/64、 2/64 … 64/64) provided by voltage divider
- The output can be triggered by a connection to the I/O or timer input
  - OCREF_CLR event (cycle by cycle current control)
  - Brakes for fast PWM shutdown
- COMP1 and COMP2 comparators can be combined in a window comparator.
- Each COMP has interrupt generation capability and is used to wake up the device from low power mode (Sleep/Stop) (via EXTI)

13.3. COMP functional description

13.3.1. COMP block diagram

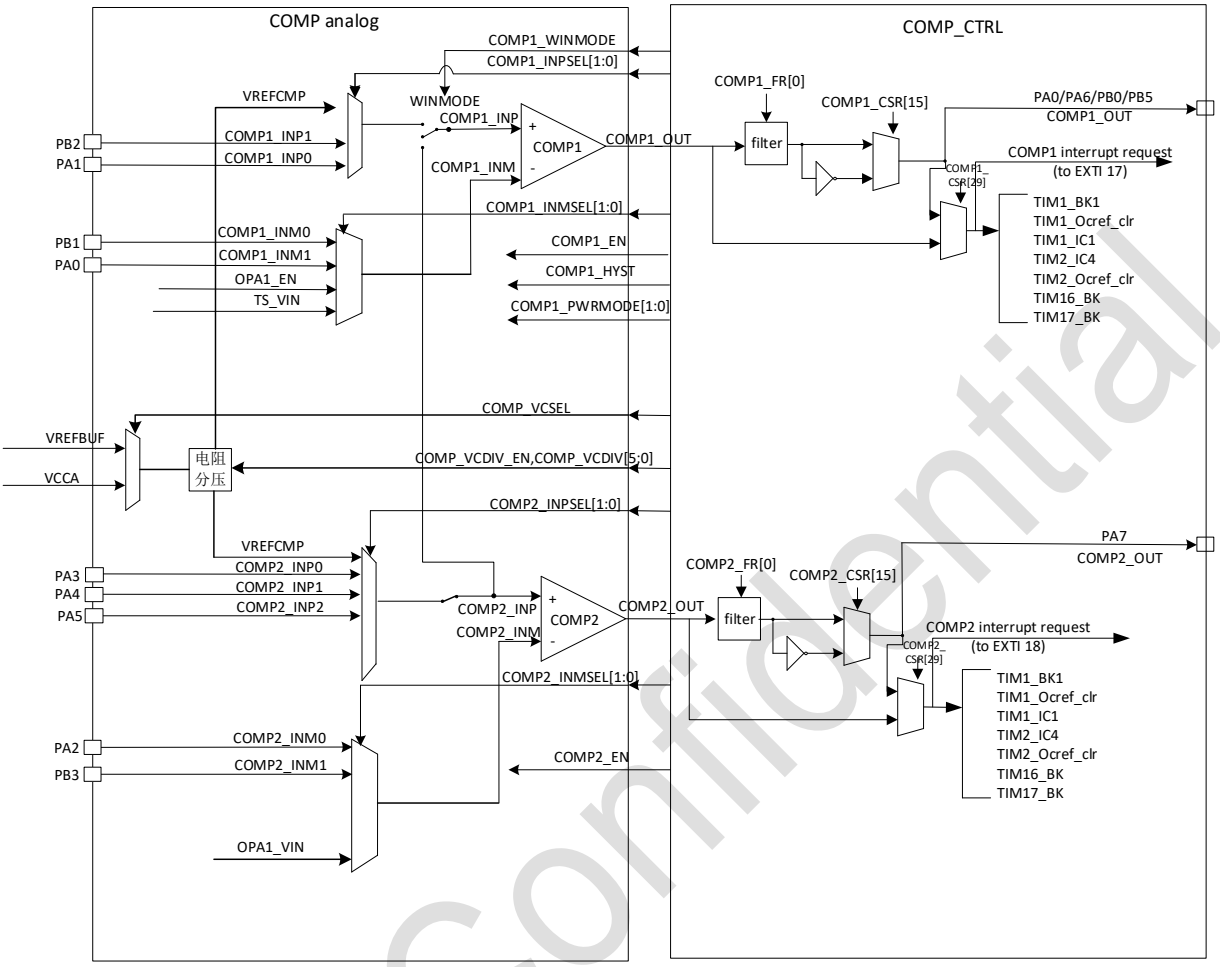


Figure 13-1 Comparator architecture block

13.3.2. COMP pins and internal signals

The I/Os used as comparators inputs must be configured in analog mode in the GPIOs registers.

- The comparator output can be connected to the I/Os using the alternate function channel.

The output can also be internally redirected to a variety of timer input for the following purposes:

- Emergency shut-down of PWM signal when brake input is connected
- OCREF_CLR event (cycle by cycle current control)
- Input capture for timing measures

The comparator output can be redirected externally or internally. The following table shows the relationship of the comparator output to other IPs.

Table 13-1 COMP module digital output

COMP1 out	COMP2 out
TIM1_BRK brake enabled	TIM1_BRK brake enabled
TIM1_OREF_CLR enabled	TIM1_Ocref_clr enabled
TIM1_IC1 channel enabled	TIM1_IC1 channel enabled
TIM1_ETR enabled	TIM1_ETR channel enabled
Wake up as EXTI17 input event	Wake up as EXTI18 input event

COMP1 out	COMP2 out
Output to PB3	Output to PA2

13.3.3. COMP reset and clocks

The COMP module has two clock sources:

- PCLK (APB clock), system bus clock
- COMP clock, used to simulate the clock of the circuit after the output of the comparator (analog output latch circuit, glitch filter circuit, etc.), can be selected as PCLK, LSE or LSI. When you need to work in Stop mode, select LSE or LSI.

Note:

The reset signal of the COMP module includes APB reset source and COMP module software reset source:

1) APB reset, for resetting COMP registers

2) COMP software reset, used to reset the circuit after analog comparator output (latch circuit of analog output, filter circuit, etc.)

When the reset signal in RCC_APBRSTR2 is enabled, both the COMP module PRESETn and COMP_RSETn signals are reset.

13.3.4. Comparator lock device

The comparator can be used for safety purposes, such as overcurrent and temperature protection. For applications with specific functional security requirements, it is necessary to ensure that the programming of the comparator cannot be overwritten in the event of false register access and PC (program counter) confusion.

Thus, the comparator control and status registers can be write-protected (read-only).

If the writing of the register is complete, the COMPx Lock bit is set to 1, which makes the entire register read-only, including the COMPx Lock bit.

The write protection can only be reset by the reset signal of the device.

13.3.5. Window comparator

The role of the Window comparator is to monitor whether the analog voltage is within the low and high threshold ranges.

You can create a window comparator using two comparators. The monitored analog voltage is simultaneously connected to the non-inverting (+ terminal) inputs of both comparators, and the high threshold and the low threshold are connected to the inverting inputs (-terminal) of both comparators, respectively.

By enabling the WINMODE bit, the non-inverting (+ inputs) of the two comparators can be connected together to save an I/O pin.

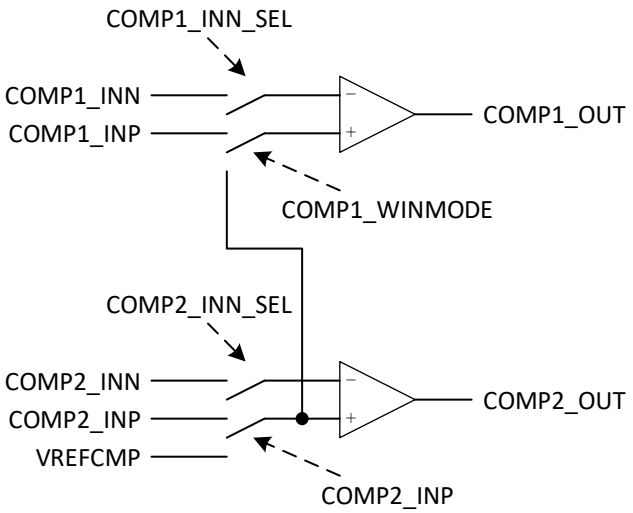


Figure 13-2 window comparator

13.3.6. Low-power modes

- **Sleep Mode**: No effect on COMP, comparator interrupt can cause the device to exit Sleep Mode.
- **Stop Mode**: No effect on COMP, comparator interrupt can cause the device to exit Stop Mode.

13.3.7. Comparator filtering

If the working environment of the device is harsh, the output of the hysteresis comparator will have a noise signal. When the digital filter module is enabled, all noise signals whose pulse width is less than the set time of FRx.FLTCNT [15: 0] in the output waveform of the hysteresis comparator can be filtered out. If the digital filter module is disabled, the input and output signals of the digital filter module are the same.

Note: Set the COMP filtering time, and enable the filtering should be completed before COMP_EN enables.

The filtering schematic diagram is as follows:

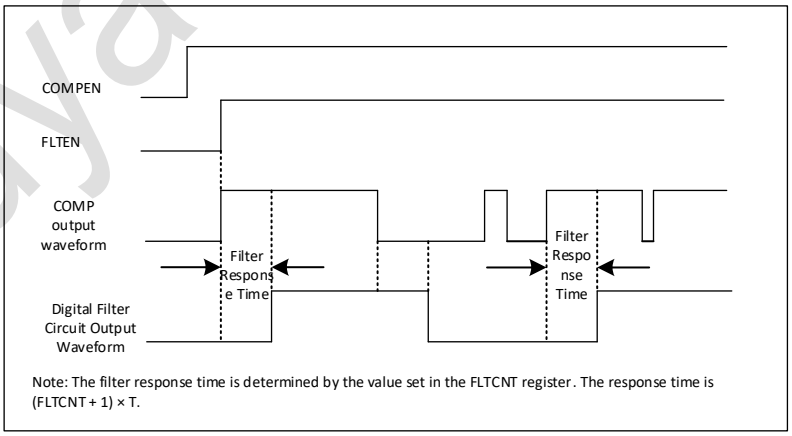


Figure 13-3 Comparator filtering

13.3.8. COMP interrupts

The comparator outputs are internally connected to the Extended interrupts and events controller. Each comparator has its own EXTI line and can generate either interrupts or events. The same mechanism is used to exit from low-power modes. Refer to the NVIC section for details.

13.4. COMP registers

13.4.1. Comparator 1 control and status register (COMP1_CSR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	COMP_OUT	TIM_EXTI_OUT_SEL	Res.	COMP_VCSEL	COMP VCDIV_EN	COMP_VCDIV[5:0]						Res.	PWR-MODE	Res.	Res.
	R	RW		RW	RW	RW							RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PO-LARITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL [1; 0]		INMSEL[1: 0]		Res.	Res.	Res.	Res.	Res.	COMP 1 _EN
RW		-	-	RW	-	RW		RW							RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	COMP_OUT	R	-	COMP1 output This read-only flag reflects the level of the comparator 1 output before the polarity selector.
29	TIM_EXTI_OUT_SEL	RW	0	COMP1 output to TIM/EXTI select. 0: Without filtering, ana_out is directly output to TIM/EXTI 1: Select comp_out output to PAD output to TIM/EXTI
28	Reserved	-	-	-
27	COMP_VCSEL	RW	0	V_{REFCMP} reference voltage source selection 0: V_{REFBUF} 1: V_{CC}
26	COMP_VCDIV_EN	RW	0	V_{REFCMP} enabled, high active.
25:20	COMP_VCDIV[5:0]	RW	0	COMP1, COMP2 partial pressure selection 000000: 1/64 Vref 000001: 2/64 Vref 000010: 3/64 Vref … 111110: 63/64 Vref 111111: Vref
19	Reserved	-	-	Reserved
18	PWRMODE	RW	0	COMP1 power consumption mode selection The power consumption and thus the speed of COMP1 were chosen 0: High speed 1: Medium speed
17:16	Reserved	-	-	Reserved
15	POLARITY	RW	0	COMP1 polarity selection These bits can be read and written by software. 0: Non-inverted 1: Inverted
14:12	Reserved	-	-	Reserved
11	WINMODE	RW	0	COMP1 window mode enabled 0: Turn off window mode, the non-inverting input of COMP1 is PB1 1: Turn on windows mode, COMP1 non-inverting (+ end) input is COMP2 non-inverting (+ end)
10	Reserved			
9:8	INPSEL[1:0]	RW	00	Comparator 1 non-inverting input selection 00: COMP1_INP from PA1 01: COMP1_INP from PB2 10: COMP1_INP from VREFCMP 11: Reserved
7:6	INMSEL[1:0]	RW	00	Comparator 1 inverting input selection 00: COMP1_INM from PA0

Bit	Name	R/W	Reset Value	Function
				01: COMP1_INM from PB1 10: COMP1_INM from TS_VIN 11: Reserved
5:1	Reserved	-	-	Reserved
0	COMP1_EN	RW	0	Comparator 1 enable Software can be read and written (if not locked) 0: Disabled 1: Enabled

13.4.2. Comparator 1 filtering register (COMP1_FR)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1 [15:0]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1	RW	0x0	Comparator 1 Sampling Filter Counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count cycle = FLTCNT [15: 0]
15:1	Reserved	-	-	Reserved
0	FLTEN1	RW	0x0	Comparator 1 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when COMP1_EN is 0

13.4.3. Comparator 2 control and status register (COMP2_CSR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	COMP_OUT	TIM_EXTI_OUT_SEL	Res.	Res.	Res.	Res.				Res.	Res.	Res.	PWR-MODE	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 [1; 0]		INMSEL[1:0]		Res.	Res.	Res.	Res.	Res.	COMP2_EN
RW		-	-		-	RW		RW							RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	COMP_OUT	R		COMP2 output This read-only flag reflects the level of the comparator 2 output before the polarity selector.
29	TIM_EXTI_OUT_SEL	RW	0	COMP2 output to TIM/EXTI select. 0: Without filtering, ana_out is directly output to TIM/EXTI 1: Select comp_out output to PAD output to TIM/EXTI
28:19	Reserved	-	-	Reserved
18	PWRMODE	RW	0	COMP2 power consumption mode selection The power consumption mode and the consequent speed of COMP2 are selected 0: High speed 1: Medium speed
17:16	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
15	POLARITY	RW	0	COMP2 polarity selection Software can be read and written (if not locked) 0: Non-inverted 1: Inverted
14:10	Reserved	-	-	Reserved
9:8	INPSEL[1:0]	RW	0	Comparator 2 non-inverting input selection 00: COMP2_INP from PA3 01: COMP2_INP from PA4 10: COMP2_INP from PA5 11: Reserved
7:6	INMSEL[1:0]	RW	00	COMP2 inverting (-end) input selection 00: COMP2_INM from PA2 01: COMP2_INM from PB3 10: Reserved 11: Reserved
5:1	Reserved	-	-	Reserved
0	COMP2_EN	RW	0	Comparator 2 enable Software can be read and written (if not locked) 0: Disabled 1: Enabled

13.4.4. Comparator 2 filtering register (COMP2_FR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT2 [15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2	RW	0	Comparator 2 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count cycle = FLTCNT [15: 0]
15:1	Reserved	-	-	Reserved
0	FLTEN2	RW	0	Comparator 2 digital filter function configuration 0: Digital filtering function disabled 1: Digital filtering function enabled Note: This bit must be set when COMP2_EN is 0

14. Advanced-control timers (TIM1)

14.1. Introduction

The advanced-control timer (TIM1) is consist of a 16-bit auto-reload counter driven by a programmable prescaler. It can be used in various scenarios, including pulse length measurement of input signals (input capture) or generating output waveforms (output compare, output PWM, complementary PWM with dead-time insertion).

The pulse length and waveform period can be modulated from microseconds to milliseconds using a timer divider and an RCC clock control divider. The Advanced Timer (TIM1) and General Purpose (TIMx) timers are completely independent and do not share any resources. They can sync up.

14.2. TIM1 main features

- 16-bit up, down, up/down auto-reload counter
- 16-bit programmable prescaler that allows the clock frequency of the counter to be divided from 1 to 65,536 (can be modified in real time)
- Up to 4 independent channels
  - Input capture
  - Output compare
  - PWM generation (edge or center alignment 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 generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event
  - Input capture
  - Output compare
  - Break input
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

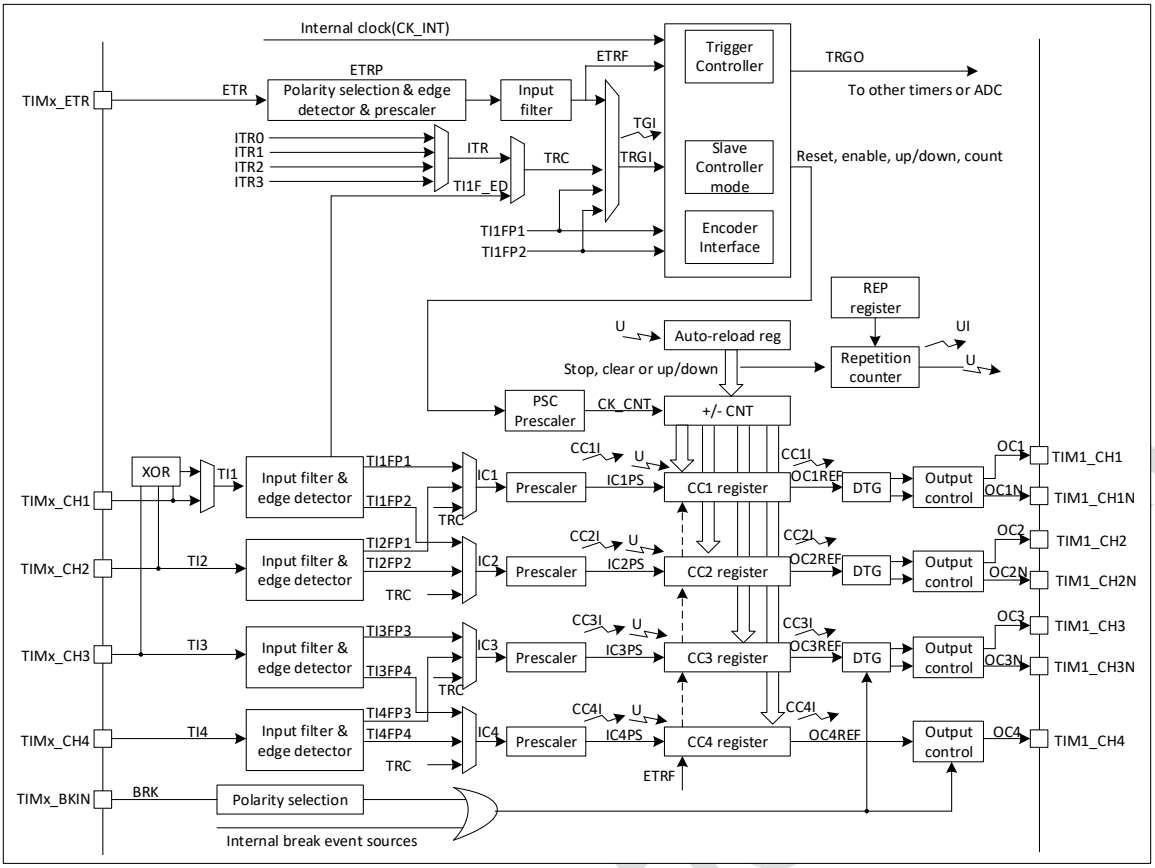


Figure 14-1 Advanced-control timer block diagram

14.3. TIM1 functional description

14.3.1. Time-base unit

The main block of the programmable advanced-control timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM1_CNT)
- Prescaler register (TIM1_PSC)
- Auto-reload register (TIM1_ARR)
- Repetition counter register (TIM1_RCR)

The auto-load register is preloaded. Writing to or reading from the auto-load register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 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.

Figures below give some examples of the counter behavior when the prescaler ratio is changed on the fly:

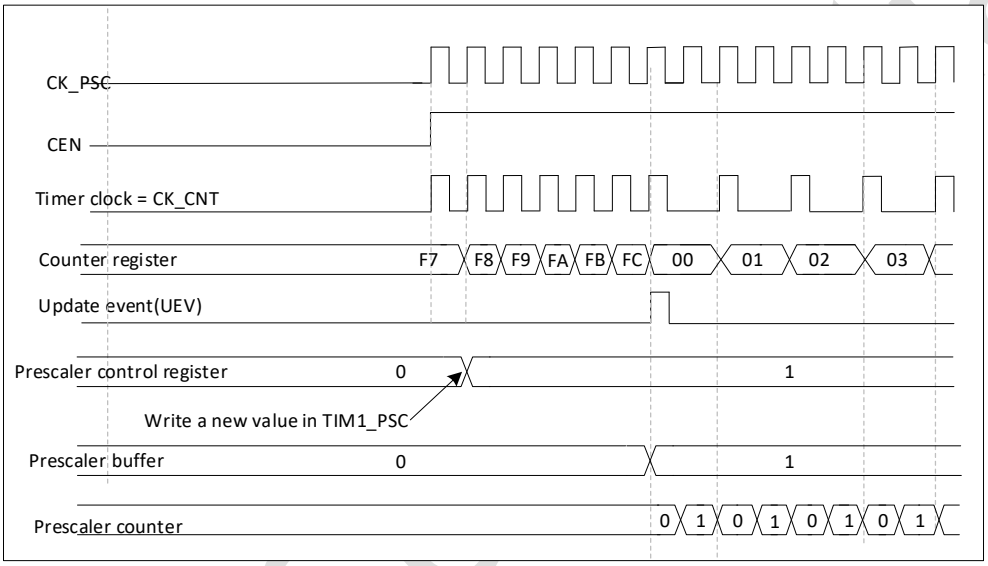


Figure 14-2 Counter timing diagram with prescaler division change from 1 to 2

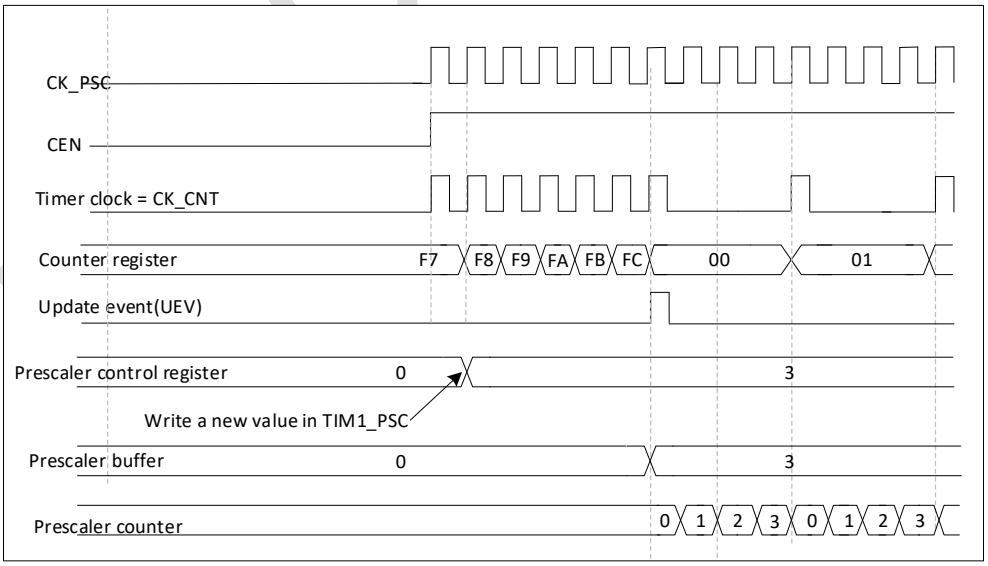


Figure 14-3 Counter timing diagram with prescaler division change from 1 to 4

14.3.2. Timer enable

14.3.2.1. Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value, then restarts from 0 and generates a counter overflow event.

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register. Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. Even then, when an update event should occur, the counter will still be cleared '0' and the count of the prescaler will be called 0 (but the value of the prescaler will not change). In addition, if the URS bit (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 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-load shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

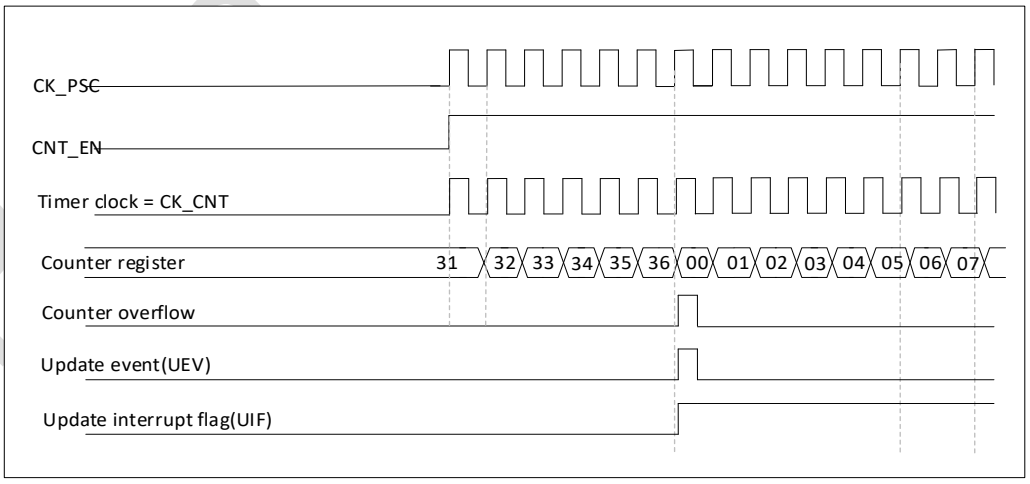


Figure 14-4 Counter timing diagram, internal clock divided by 1

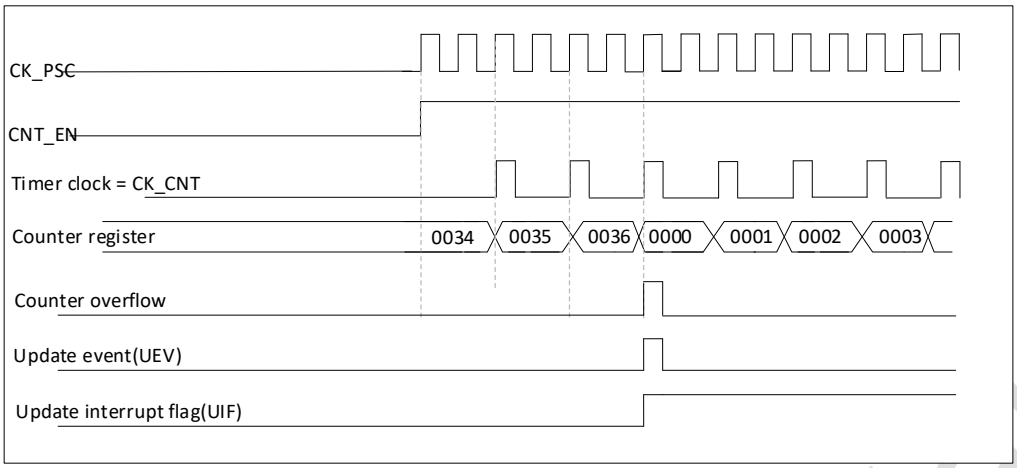


Figure 14-5 Counter timing diagram, internal clock divided by 2

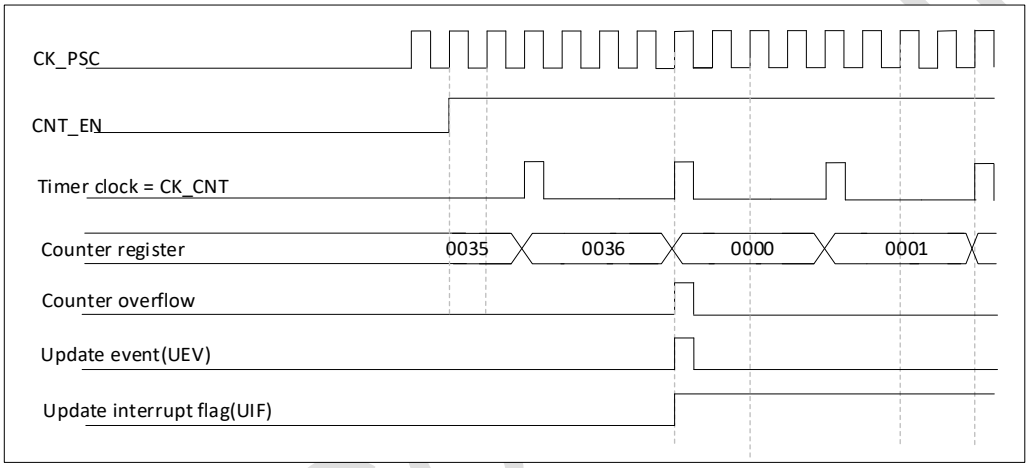


Figure 14-6 Counter timing diagram, internal clock divided by 4

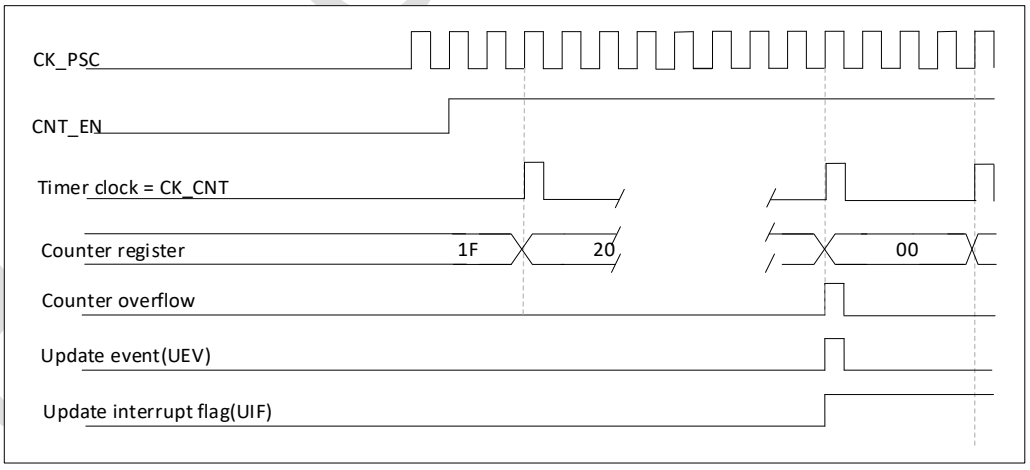


Figure 14-7 Counter timing diagram, internal clock divided by N

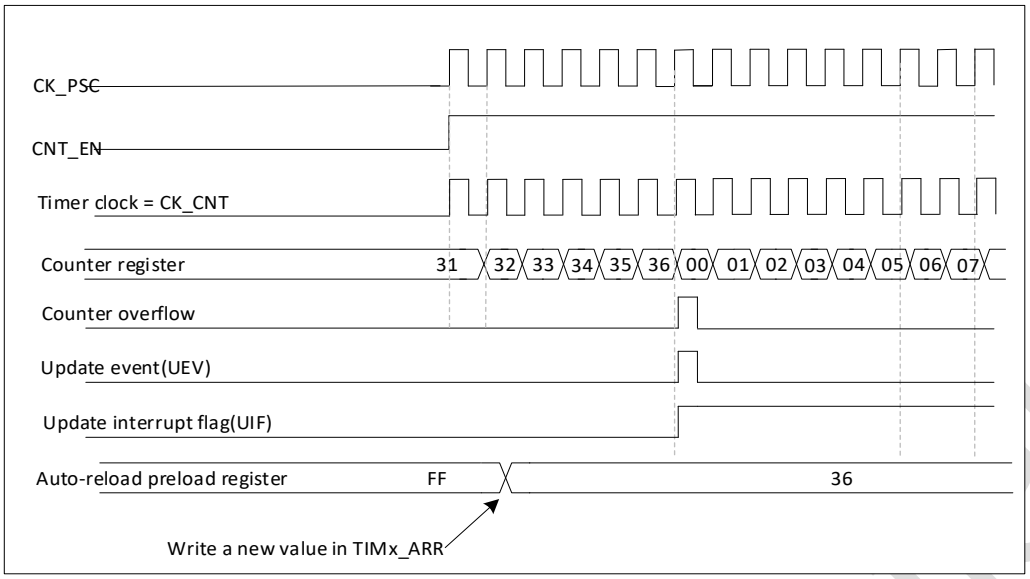


Figure 14-8 Counter timing diagram, update event when ARPE=0 (TIM1_ARR not preloaded)

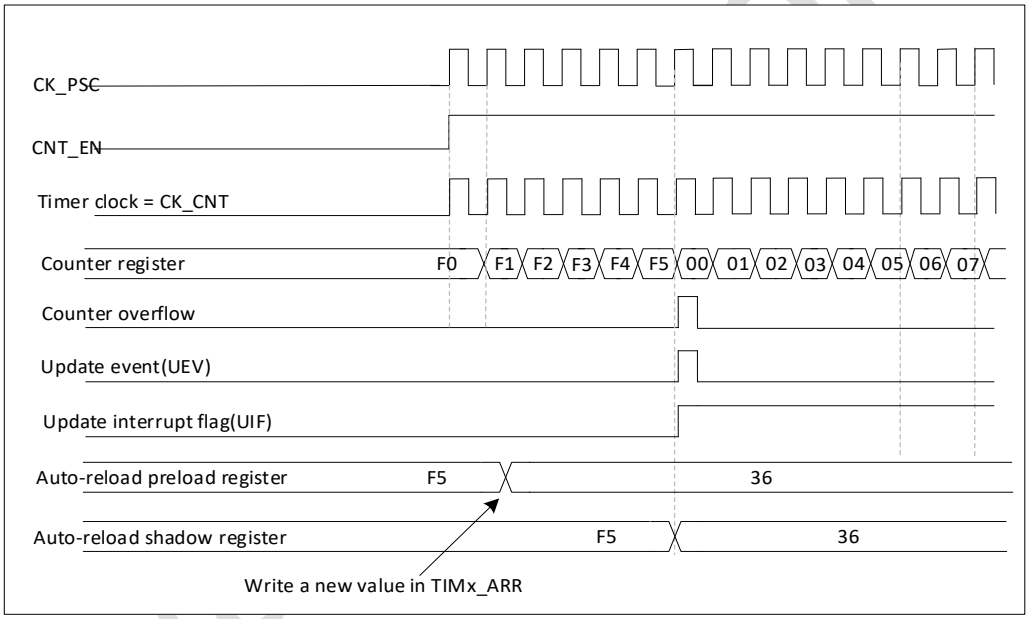


Figure 14-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)

14.3.2.2. Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (the contents of TIMx_ARR) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If a repeat counter is used, in downcounting mode, the counter counts from the auto-reload value (the contents of TIMx_ARR) down to 0, then restarts from the auto-reload value and generates a counter underflow event.

If a repetition counter is used, the update event (UEV) will not be generated until the number of underflows reaches the value of the configured repetition count register plus one (i.e. TIMx_RCR+1); If a repetition counter is not used (i.e., TIMx_RCR = 0), an update event will be generated every time the count underflows. Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. Even so, when an update event should occur, the counter will still restart counting from the current autoload value, while the counter inside the prescaler is cleared '0' (but the prescaling factor remains unchanged). In addition, if the URS bit (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 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). An update event (UEV) needs to be generated when the number of underflows reaches the value of the configured repeat count register plus one (i.e. TIMx_RCR+1); If a repetition counter is not used (i.e., TIMx_RCR = 0), an update event will be generated every time the count underflows. Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. Even so, when an update event should occur, the counter will still restart counting from the current autoload value, while the counter inside the prescaler is cleared '0' (but the prescaling factor remains unchanged). In addition, if the URS bit (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 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 that the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR=0x36.

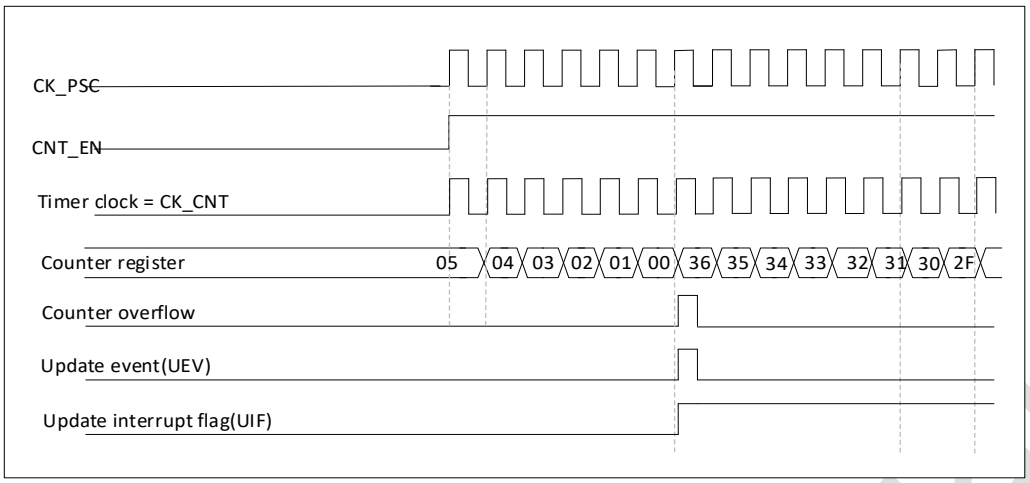


Figure 14-10 Counter timing diagram, internal clock divided by 1

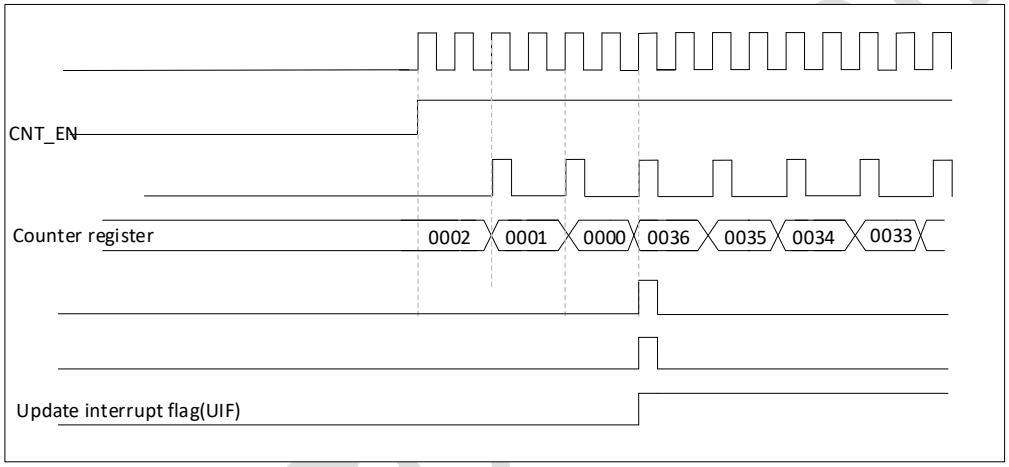


Figure 14-11 Counter timing diagram, internal clock divided by 2

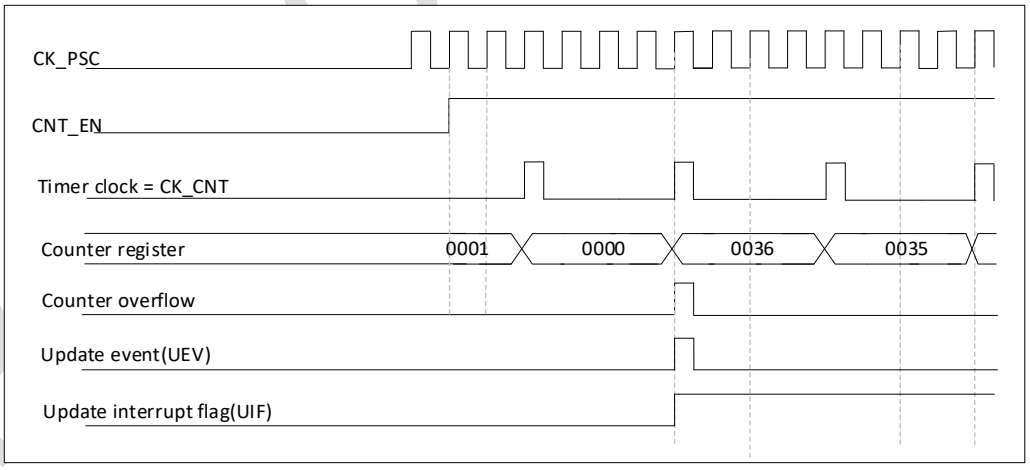


Figure 14-12 Counter timing diagram, internal clock divided by 4

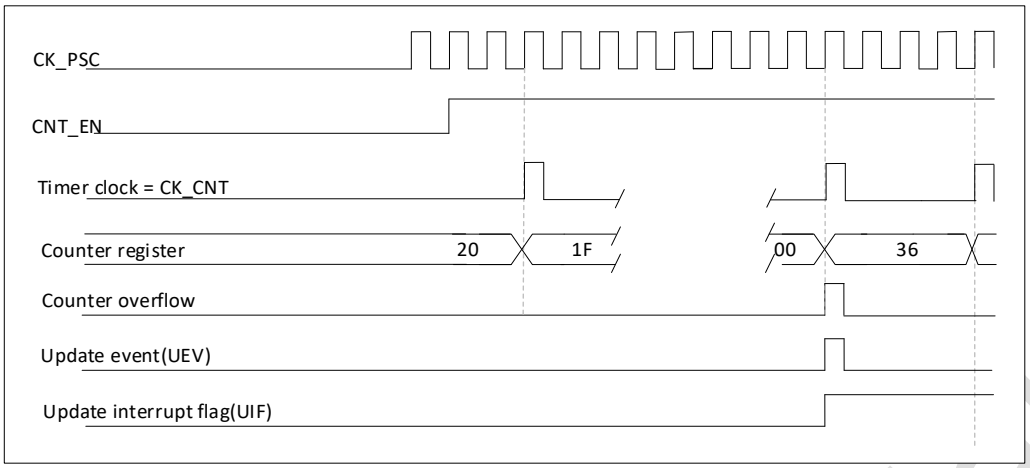


Figure 14-13 Counter timing diagram, internal clock divided by N

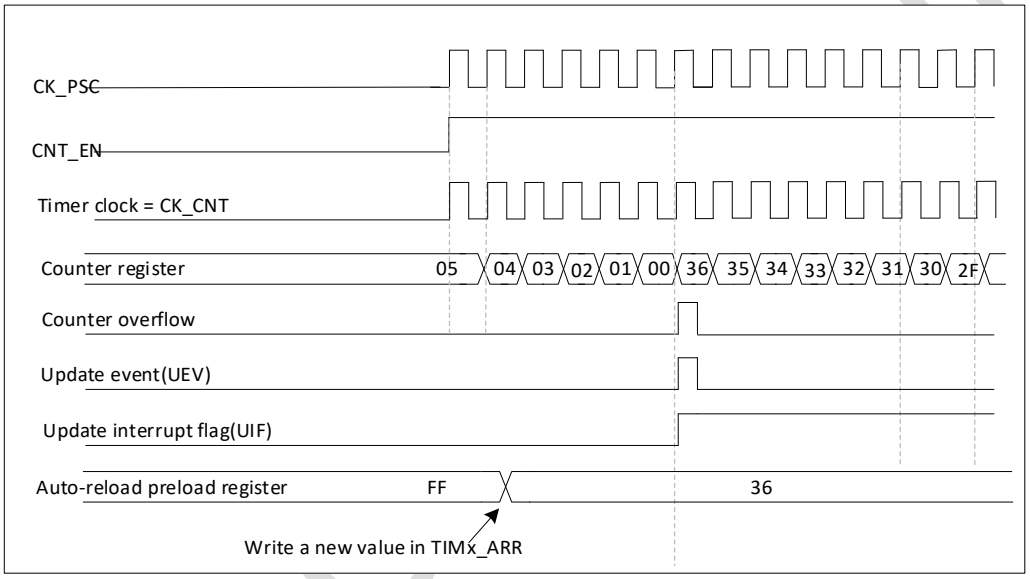


Figure 14-14 Counter timing diagram, update event when repetition counter is not used

14.3.2.3. Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register) – 1, generates a counter overflow event, then counts from the auto-reload value down to 1 and generates a counter underflow event. Then it restarts counting from 0.

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are not equal to '0'. The Output compare interrupt flag of channels configured in output is set when: the counter counts down (Center aligned mode 1, CMS = “01”), the counter counts up (Center aligned mode 2, CMS = “10”) the counter counts up and down (Center aligned mode 3,CMS = “11”).

In this mode, the DIR direction bit in the TIMx_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also

generates an update event. In this case, the counter restarts counting from0, as well as the counter of the prescaler.

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt 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 that if the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value).

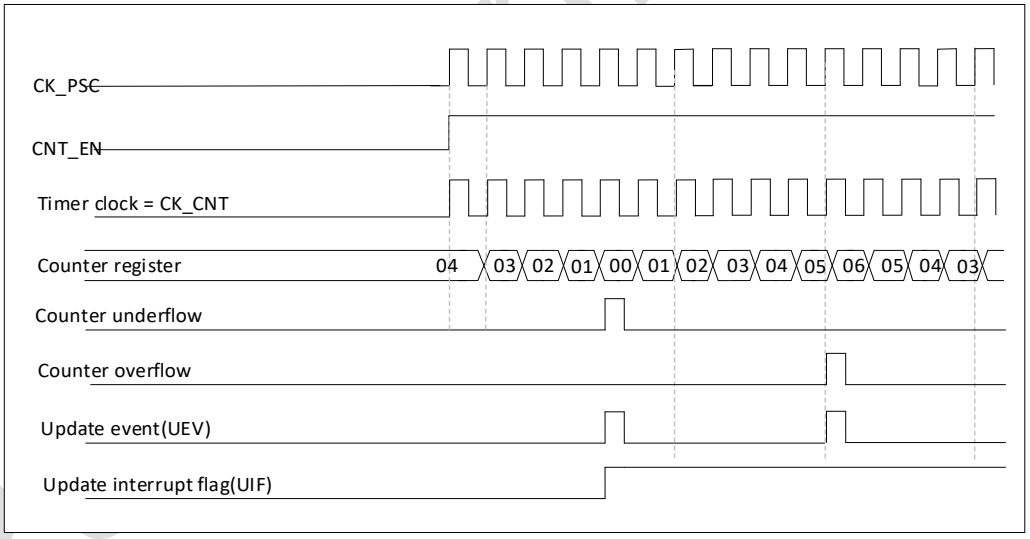


Figure 14-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

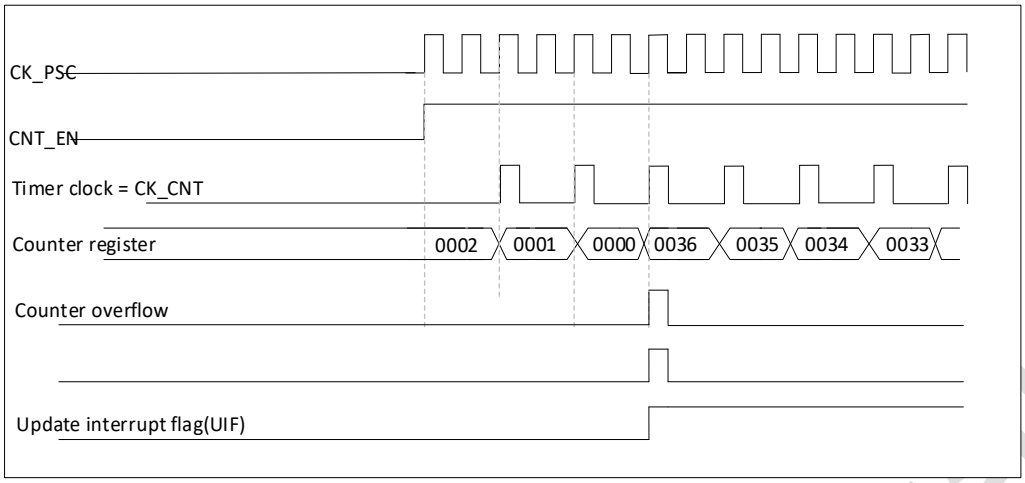


Figure 14-16 Counter timing diagram, internal clock divided by 2, TIMx_ARR=0x36

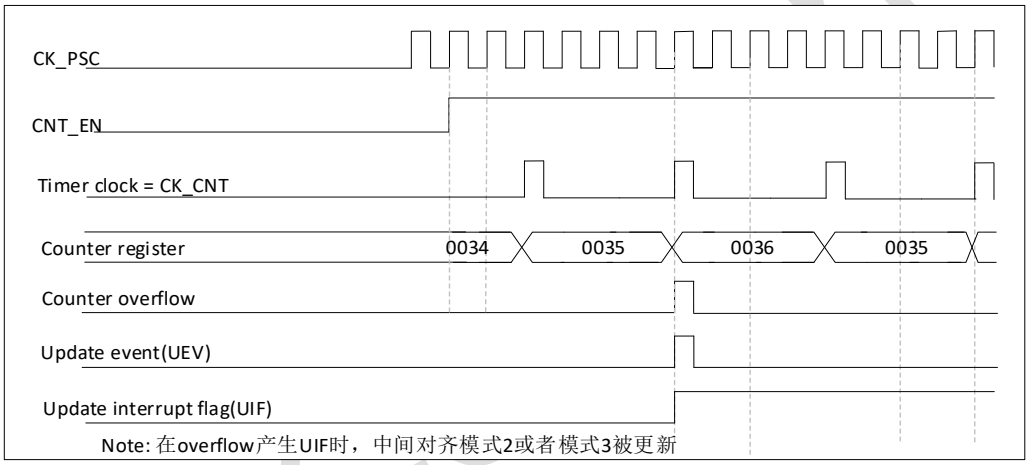


Figure 14-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x36

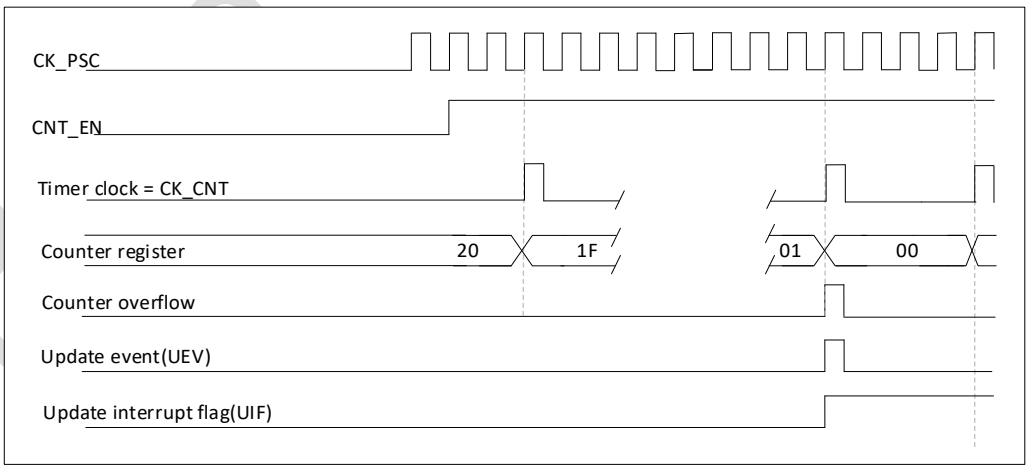


Figure 14-18 Counter timing diagram, internal clock divided by N, TIMx_ARR=0x36

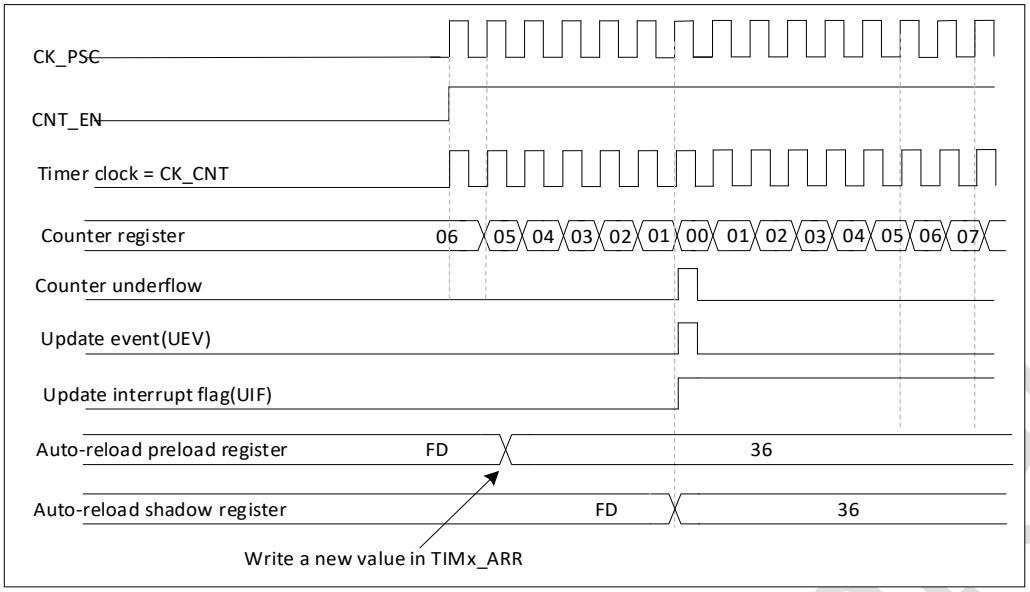


Figure 14-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

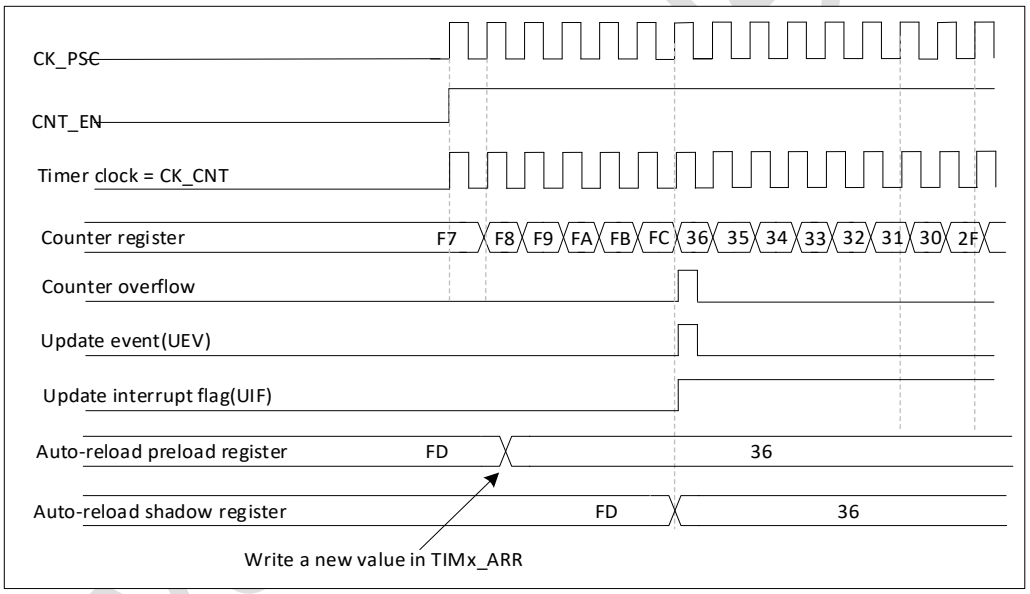


Figure 14-20 Counter timing diagram, Update event with ARPE=1 (counter overflow)

14.3.3. Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals.

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N + 1 counter overflows or underflows, where N is the value in the TIMx_RCR repetition counter register.

The repetition counter is decremented:

- At each counter overflow in upcounting mode

- At each counter underflow in downcounting mode
- At each counter overflow and at each counter underflow in center-aligned mode.

Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. In center alignment mode, because the waveform is symmetrical, the maximum resolution is 2xTck if the comparison register is flushed only once per PWM cycle.

The repetition counter is an auto-reload type; the repetition rate is maintained as defined by the TIMx_RCR register value. When the update event is generated by software (by setting the UG bit in TIMx_EGR register) or by hardware through the slave mode controller, it occurs immediately whatever the value of the repetition counter is, and the repetition counter is reloaded with the content of the TIMx_RCR register.

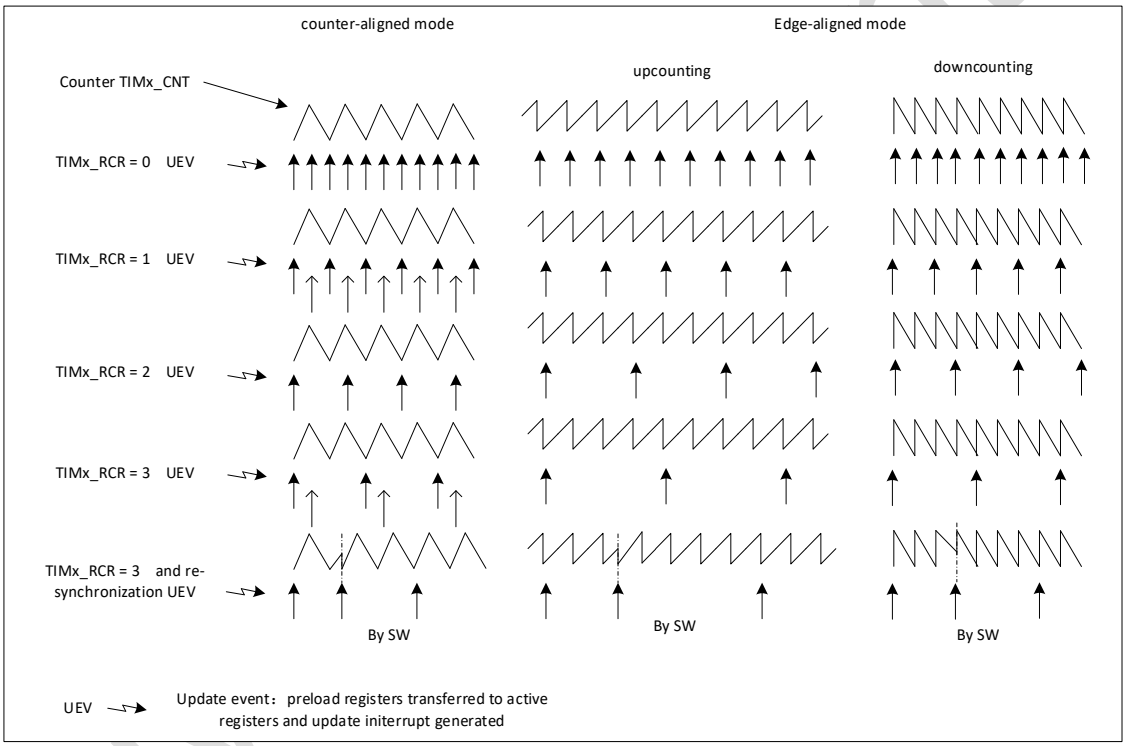


Figure 14-21 Examples of update rates in different modes, and register settings for TIMx_RCR

14.3.4. Clock sources

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode1: external input pin
- External clock mode2: external trigger input ETR
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer. For example, one timer Timer1 may be configured as a prescaler for another timer Timer3.

14.3.4.1. Internal clock source (CK_INT)

If the slave mode controller is disabled, then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software. As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

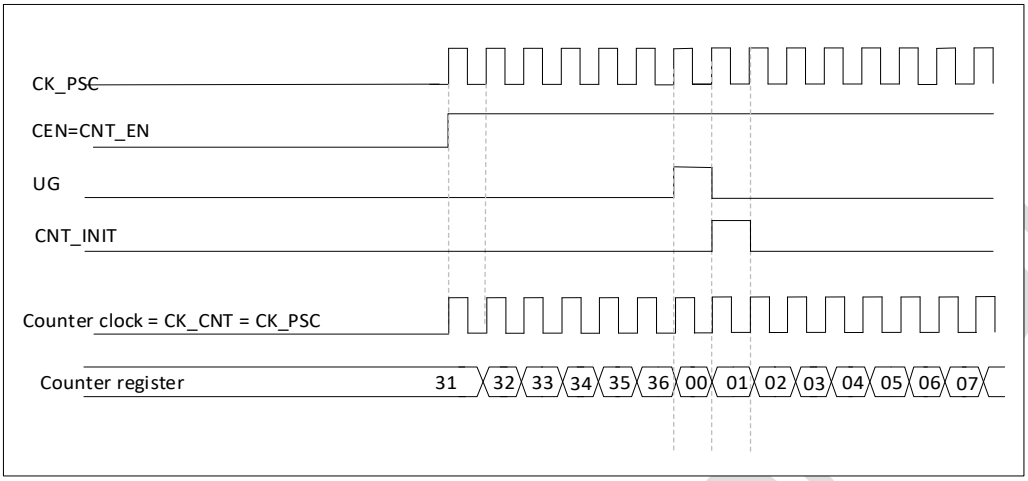


Figure 14-22 Control circuit in normal mode, internal clock divided by 1

14.3.4.2. External clock source mode 1

This mode is selected when SMS [2:0]=111 in the TIMx_SMCR register. The counter can count at each rising or falling edge on a selected input.

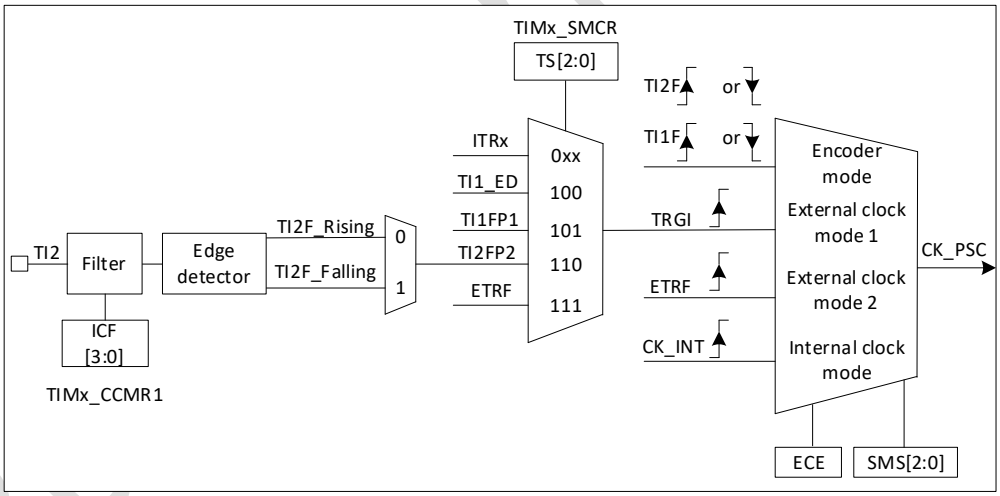


Figure 14-23 TI2 external clock connection example

For example, to configure the counter to count up on the rising edge of the T12 input, use the following steps:

1. Configure the TIMx_CCMR1 register CC2S = 01 such that channel 2 detects a rising edge at the input of TI2;

2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F=0000).

3. Configure CC2P = 0 of the TIMx_CCER register, and select the rising edge polarity;

4. Configure the SMS [2: 0] = 111 of the TIMx_SMCR register, and select the timer to external clock mode 1;

5. Configure TS [2: 0] = 110 in the TIMx_SMCR register, and select TI2 as the trigger input source;

6. Enable the counter by writing CEN=1 in the TIMx_CR1 register.

Note: The capture prescaler is not used as a trigger, so it does not need to be configured

When the rising edge occurs at TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization circuit at the input of TI2.

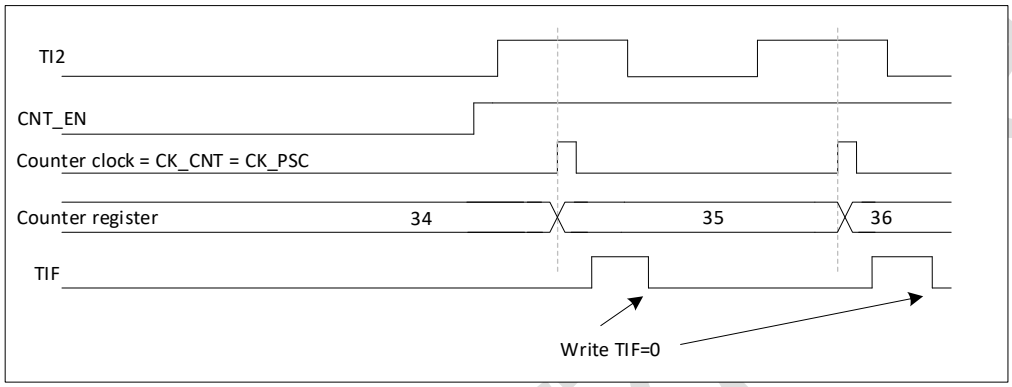


Figure 14-24 Control circuit in external clock mode 1

14.3.4.3. 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.

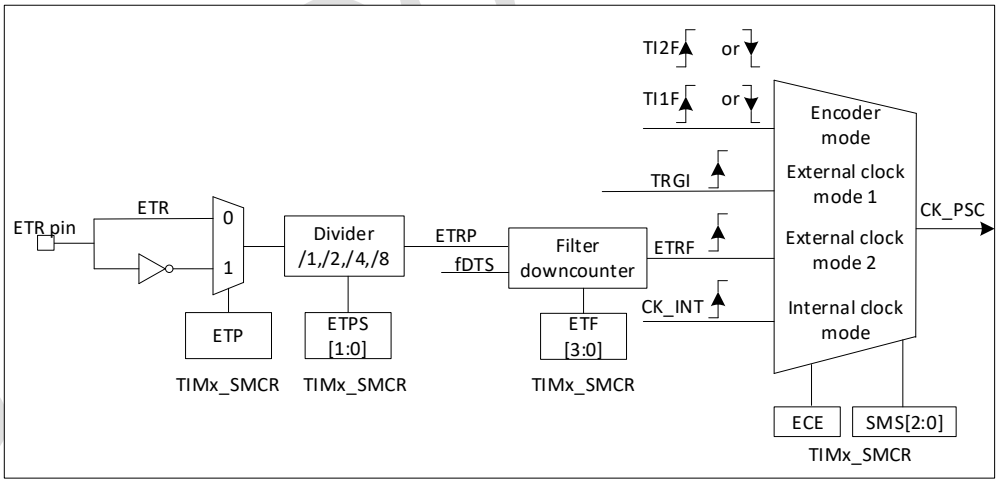


Figure 14-25 TI2 external trigger input block

For example, to configure an up-counter that counts every 2 rising edges under ETR, use the following steps:

1. As no filter is needed in this example, write ETF[3:0]=0000 in the TIMx_SMCR register.

2. Set the prescaler and set ETPS [1: 0] = 01 in the TIMx_SMCR register;

3. Select the rising edge of the ETR input terminal and set ETP = 0 in the TIMx_SMCR register;

4. Turn on external clock mode 2 and write ECE = 1 in the TIMx_SMCR register;

5. Start counter, write CEN = 1 in TIMx_CR1 register;

The counter counts at every 2 ETR rising edges.

The delay between the rising edge of the ETR and the actual clock of the counter depends on the resynchronization circuit of the ETRP signal.

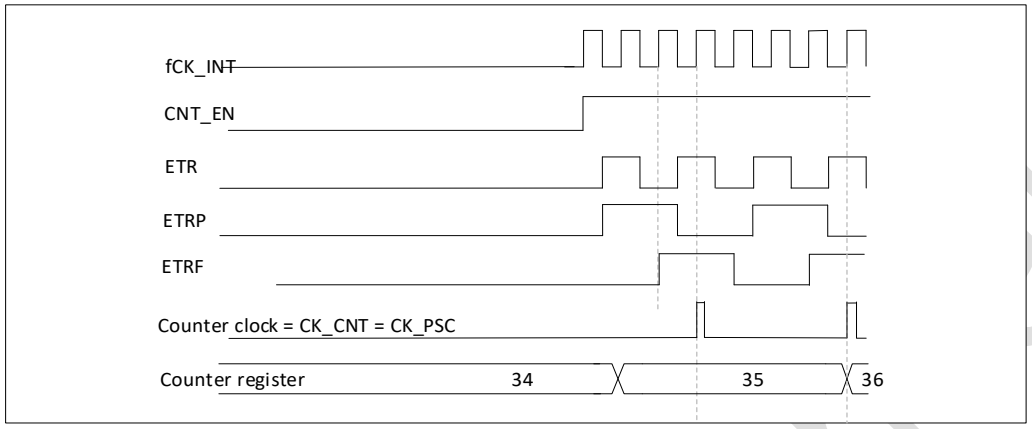


Figure 14-26 Control circuit in external clock mode 2

14.3.5. 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).

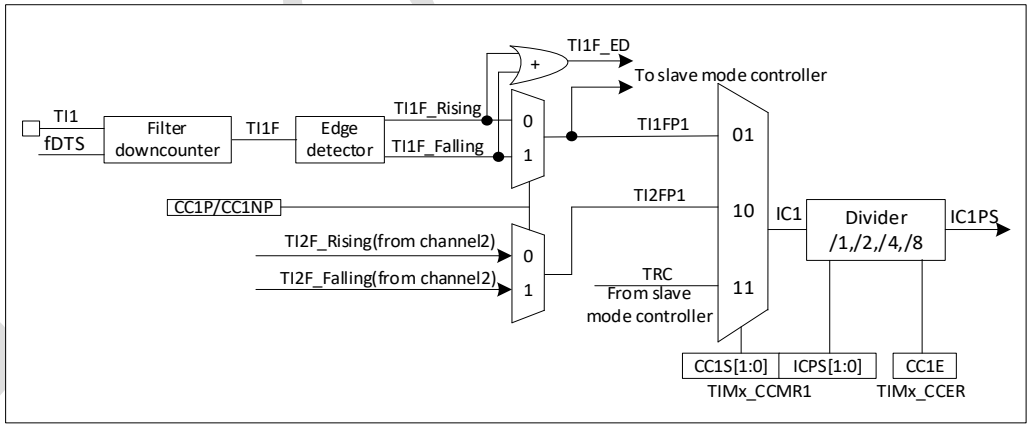


Figure 14-27 Capture/compare channel (example: channel 1 input stage)

The output stage generates an intermediate waveform which is then used for reference: OCxRef (active high). The polarity acts at the end of the chain.

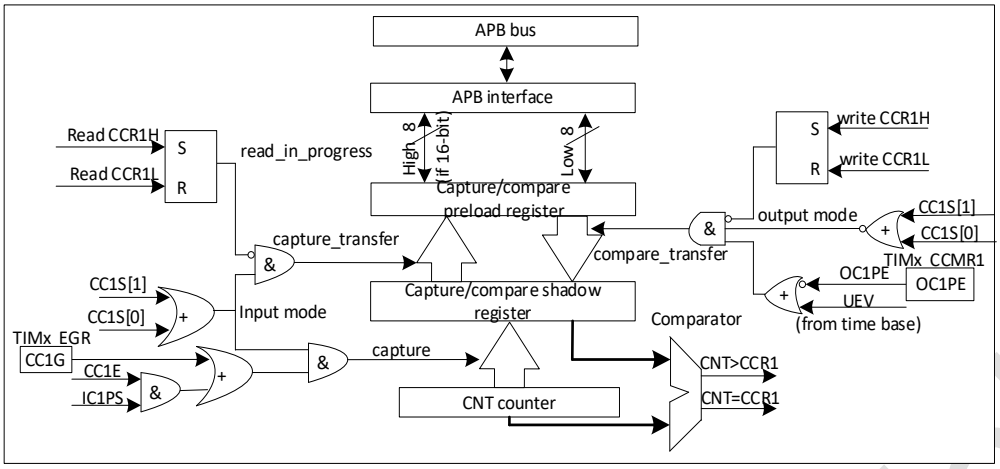


Figure 14-28 Capture/compare channel 1 main circuit

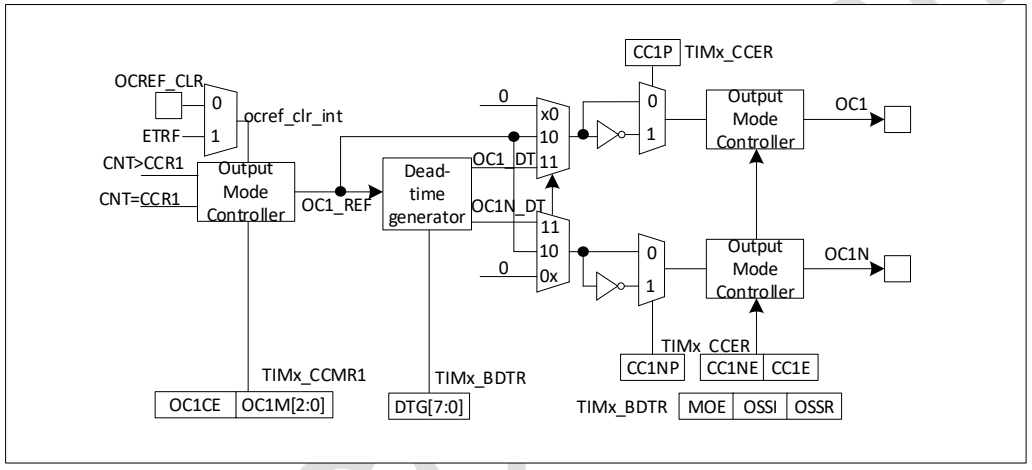


Figure 14-29 Output stage of capture/compare channel (channel 1 to 3)

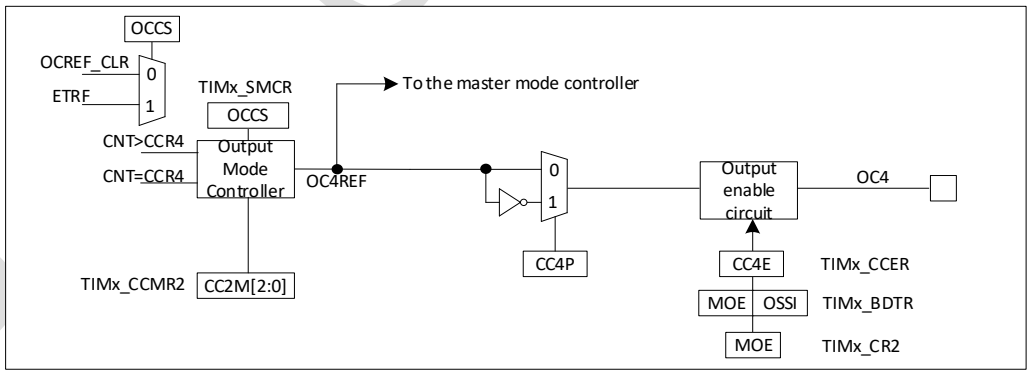


Figure 14-30 Output stage of capture/compare channel (channel 4)

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.

14.3.6. Input capture mode:

In Input capture mode, the capture/compare registers are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIMx_SR register) is set and an interrupt 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 you write it to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1input rises. To do this, use the following procedure:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at Fck_int frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register.
- When an input capture occurs:
- The TIMx_CCR1 register gets the value of the counter on the active transition.
- 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 can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

14.3.7. Input capture mode (PWM input mode)

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.
- For example, you can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):
- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to '1' (active on falling edge).
- Select the valid trigger input: Set TS[2: 0] = 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller to reset mode: Set SMS [2: 0] = 100 in TIMx_SMCR.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

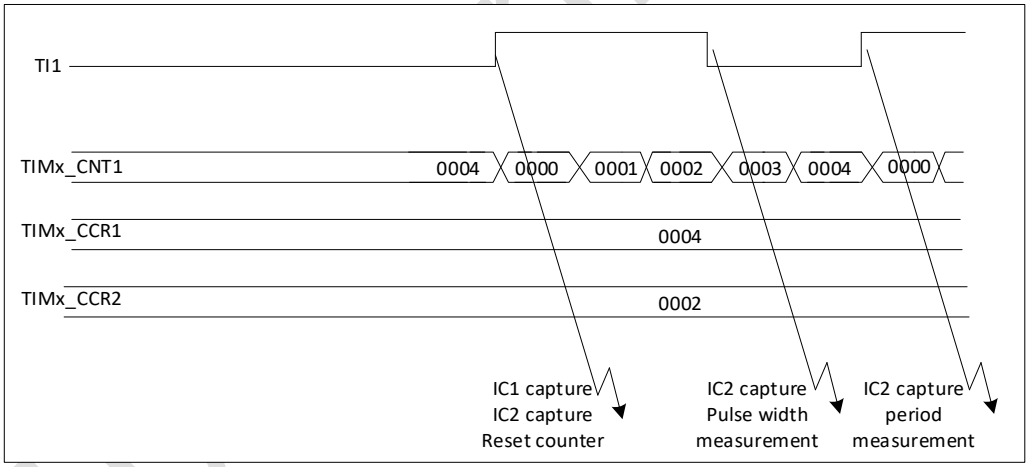


Figure 14-31 PWM input mode timing

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use the TIMx_CH1/TIMx_CH2 signal.

14.3.8. Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compares signal(OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, you just need to write101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

14.3.9. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example:

   Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx

   Write OCxPE = 0 to disable preload register

   Write CCxP = 0 to select active high polarity

   Write CCxE = 1 to enable the output
5. Enable the counter by setting the CEN bit in the TIMx_CR1 register.

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE= ’0’, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the figure below.

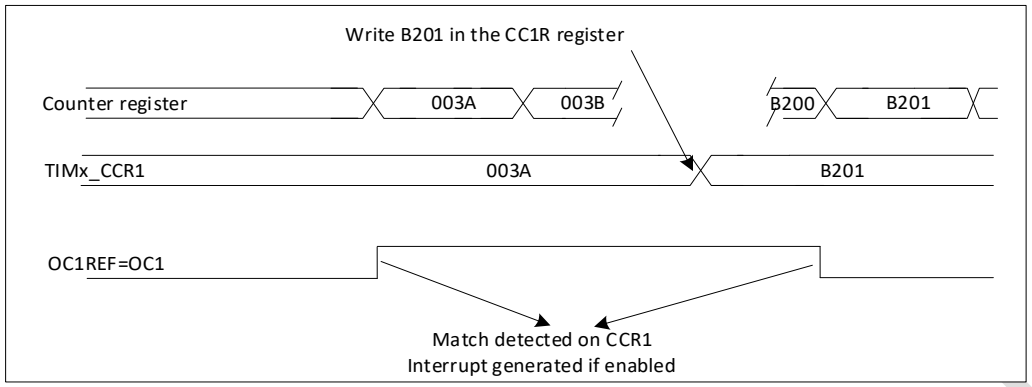


Figure 14-32 Output compare mode, toggle on OC1

14.3.10. PWM mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

14.3.10.1. PWM edge-aligned mode

- **Upcounting configuration**

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

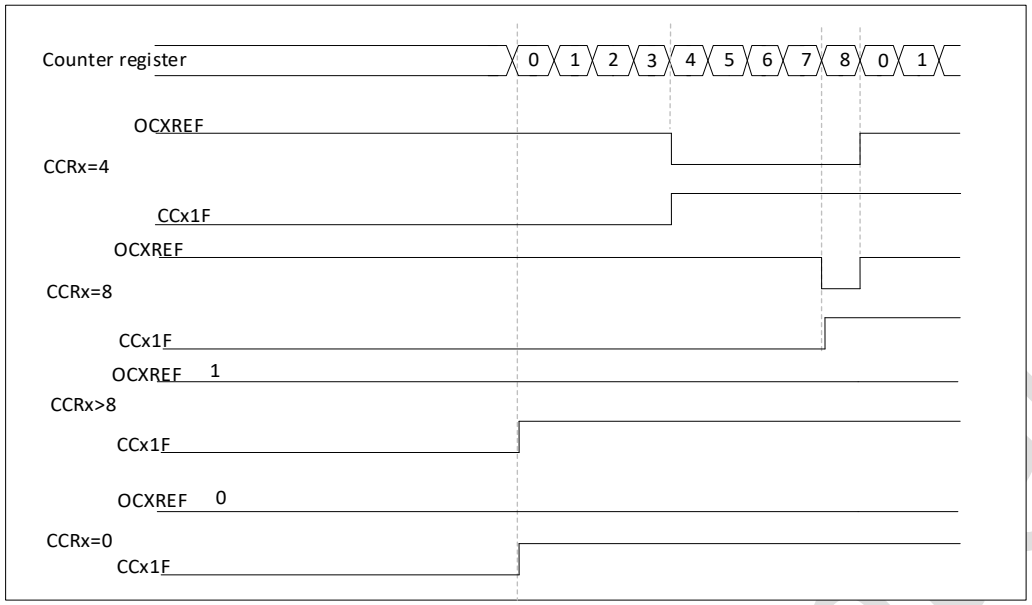


Figure 14-33 Edge-aligned PWM waveforms (ARR=8)

- **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 asTIMx_CNT > TIMx_CCRx else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’. 0% PWM is not possible in this mode.

14.3.10.2. PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software.

Figure below shows some center-aligned PWM waveforms in an example where:

- TIMx_ARR = 8
- PWM mode is the PWM mode 1
- The flag is set when the counter counts down corresponding to the center-aligned mode 1 selected for CMS=01 in TIMx_CR1 register.

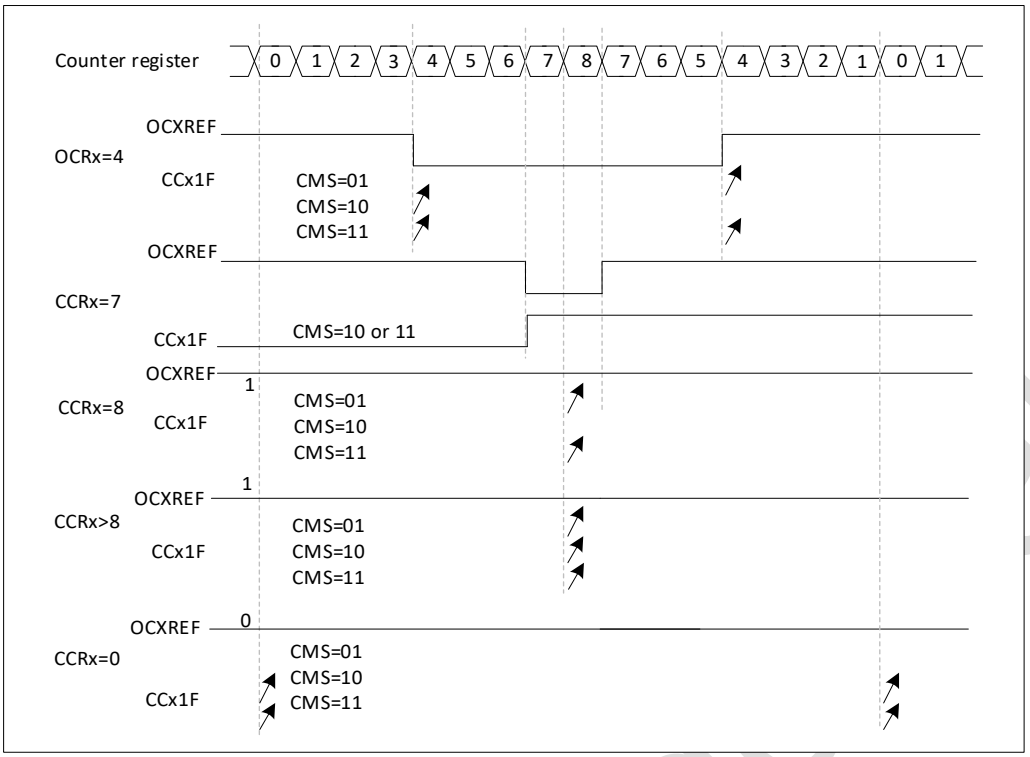


Figure 14-34 Center-aligned PWM waveforms (ARR=8)

Hints on using center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular: The direction is not updated if you write a value in the counter that is greater than the auto-reload value (TIMx_CNT>TIMx_ARR). For example, if the counter was counting up, it continues to count up. The direction is updated if you write 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated.
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

14.3.11. Complementary outputs and dead-time insertion

The advanced-control timers (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs. This time is generally known as dead-time, and you have to adjust it depending on the devices you have connected to the outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches...)

You can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. Each channel has an 8-bit dead time generator DTG [7: 0], configured by configuring the TIMx_BDTR register. 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)

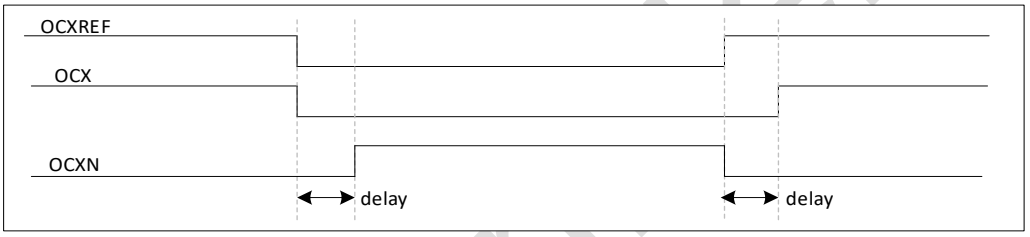


Figure 14-35 Complementary output with dead-time insertion

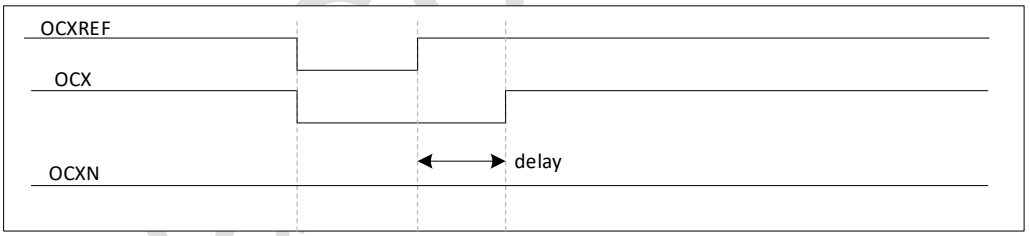


Figure 14-36 Dead-time waveforms with delay greater than the negative pulse

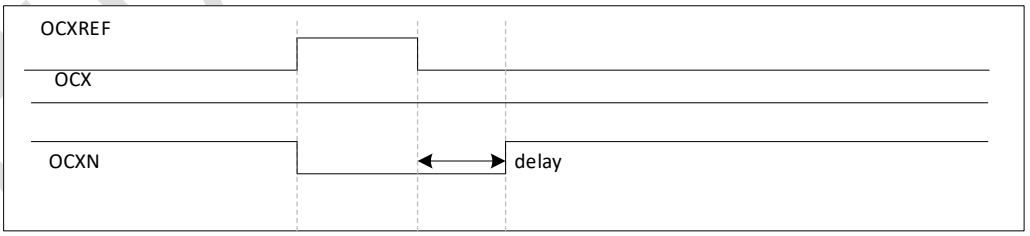


Figure 14-37 Dead-time waveforms with delay greater than the positive pulse.

The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register.

Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register. This allows you to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other alternative possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.

Note: When only OCxN is enabled (CCxE=0, CCxNE=1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP=0 then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1) OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

14.3.12. Using the break function

When the brake function is used, both the output enable signal and the invalid level are modified according to the corresponding 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. Output control bits for complementary OCx and OCxN channels with break feature. The brake source can be either the brake input pin or the following events:

- the CPU LOCKUP output
- SRAM parity error signal
- Clock failure event generated by CSS monitoring
- the output from a comparator

When exiting from reset, the break circuit is disabled and the MOE bit is low. You can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time. When the BKE and BKP bits are written, a delay of 1 APB clock cycle is applied before the writing is effective. Consequently, it is necessary to wait 1APB clock period to correctly read back the bit after the write operation.

MOE falling edge can be asynchronous, thus a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if you write MOE to 1 whereas it was low, you must insert a delay (dummy instruction) before reading it correctly. This is because the write acts on the asynchronous signal whereas the read reflects the synchronous signal.

When a break occurs (selected level on the break input):

- The MOE bit is cleared asynchronously, putting the 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: due to the resynchronization of the MOE, the dead time is longer than usual (approximately 2 clock cycles of ck_tim).
  - If OSSI=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set at the next update event UEV; For example, this can be used for shaping. Otherwise, the MOE remains low until it is set '1' again; At this time, this feature can be used in safety. You can connect the brake input to the power-driven alarm output, thermal sensor or other safety devices.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF can not be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR register. The brake can also be generated by software setting the BG bit in the TIMx_EGR register.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The application can choose from 3 levels of protection selected by the LOCK bits in the TIMx_BDTR register. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

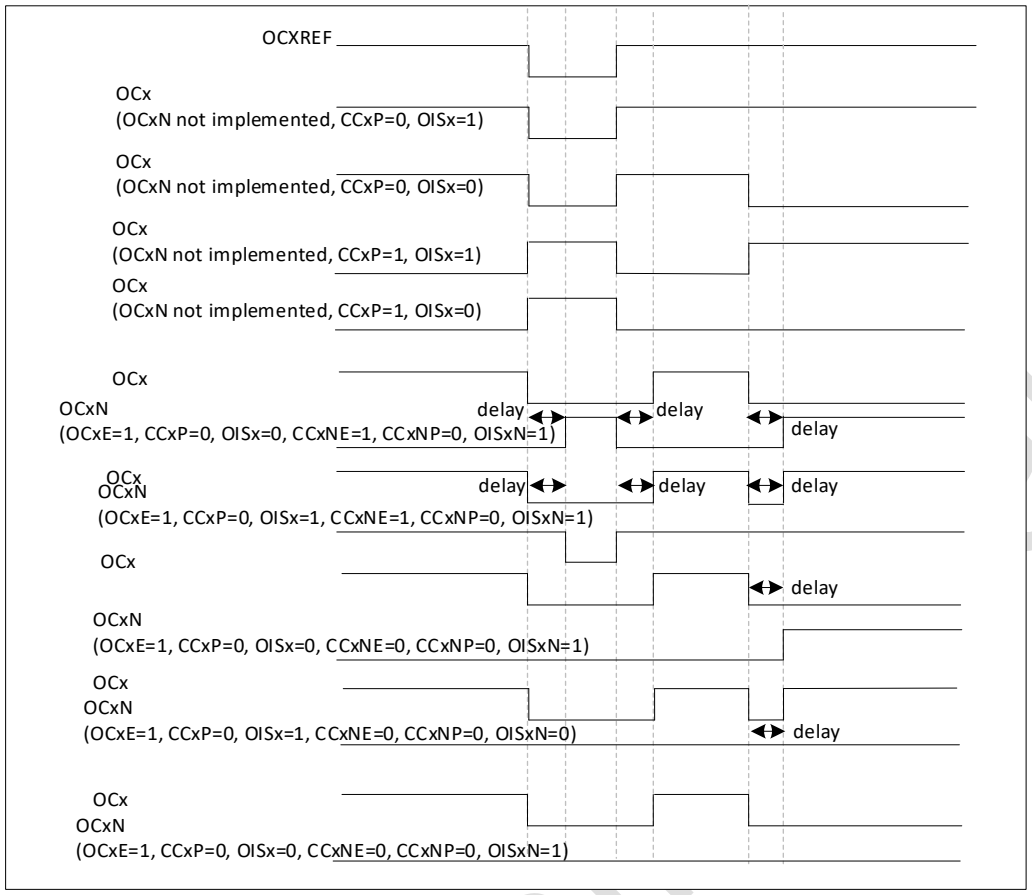


Figure 14-38 Output behavior in response to a break

14.3.13. Clearing the OCxREF signal on an external event

The OCxREF signal of a given channel can be cleared when a high level is applied on the OC-REF_CLR_INPUT (OCxCE enable bit in the corresponding TIMx_CCMRx register set to1). OCxREF remains low until the next update event (UEV) occurs. This function can only be used in output compare and PWM modes. It does not work in Forced mode.

While OCREF_CLR_INPUT can be selected between OCREF_CLR and ETRF (after ETR filtering) by configuring the OCCS bit in the TIMx_SMCR register.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, ETR must be configured as follows:

1. The external trigger prescaler should be kept off: bits ETPS[1:0] in the TIMx_SMCR register are cleared to 00.
2. The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
3. The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is programmed in PWM mode.

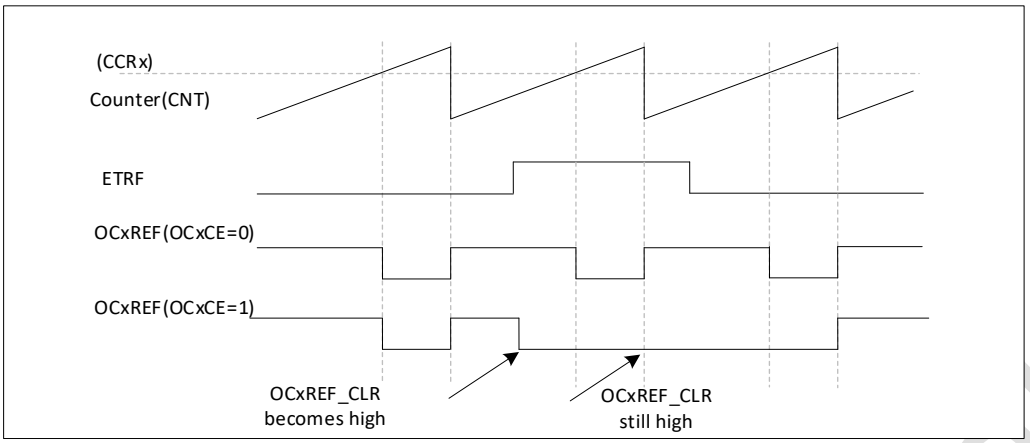


Figure 14-39 OCxREF for clearing TIM1

Note: In case of a PWM with a 100% duty cycle (if CCRx>ARR), then OCxREF is enabled again at the next counter overflow.

14.3.14. 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. Thus you can program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

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).

The figure below describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

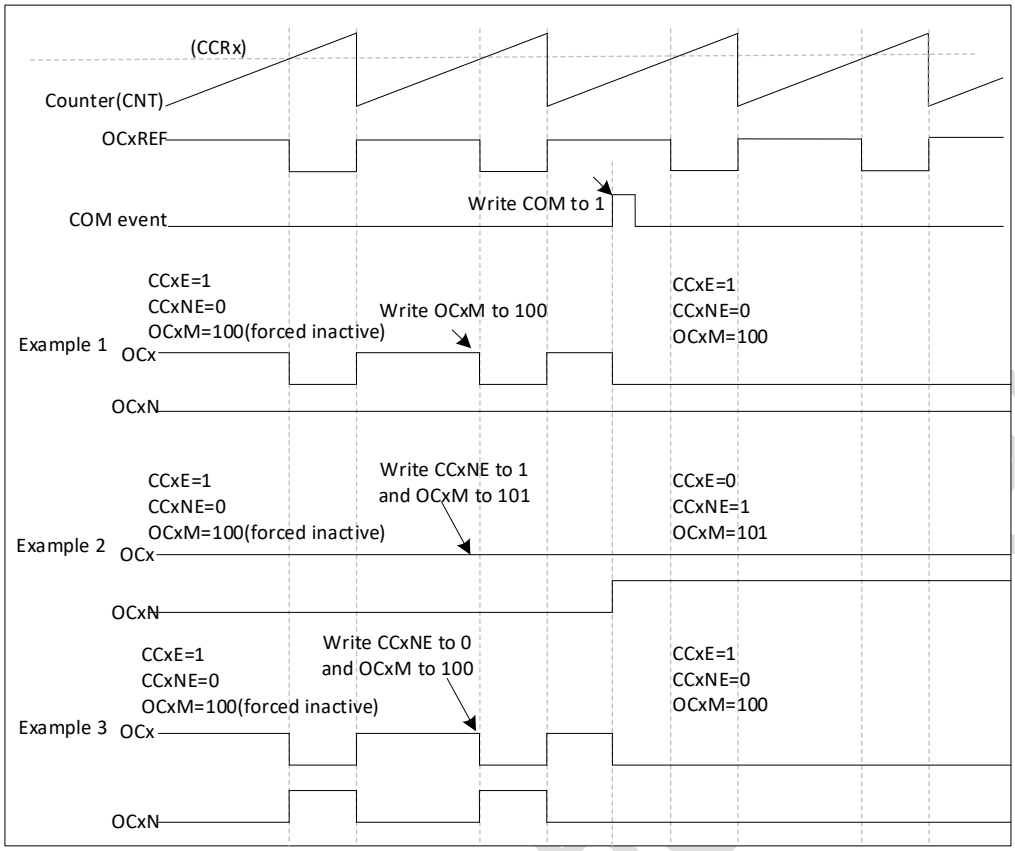


Figure 14-40 6-step generation, COM example (OSSR=1)

14.3.15. One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. One-pulse mode is selected by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

- In upcounting: CNT < CCRx ≤ ARR (in particular, 0 > CCRx)
- In downcounting: CNT > CCRx

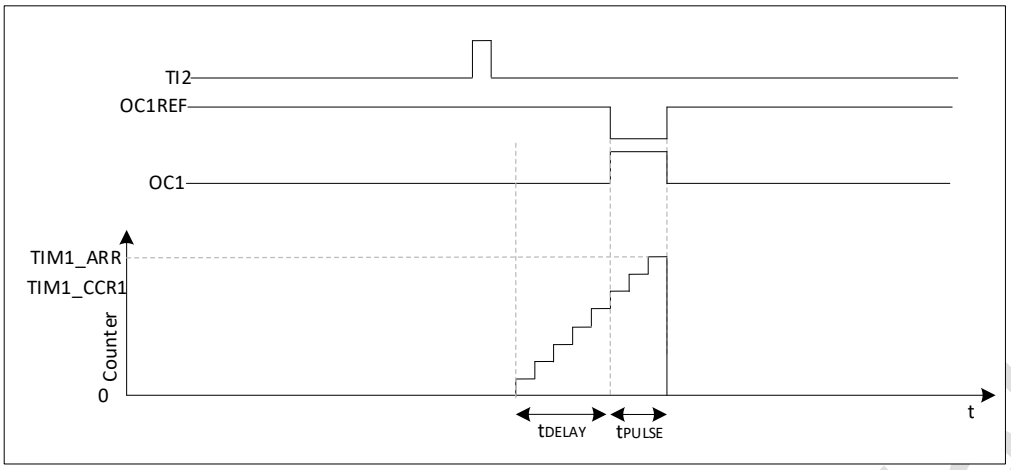


Figure 14-41 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS[2:0] to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let's say one want to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

Particular case: OCx fast enable

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay t_{DELAY} min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF (and OCx) is forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2mode.

14.3.16. Retriggerable one pulse mode (OPM)

This mode allows the counter to start in response to excitation and generate pulses of programmable length, which differs from the non-retriggerable single pulse mode described in the previous section as follows:

- The pulse starts as soon as the trigger occurs (no programmable delay).
- If a new trigger occurs before the pulse generated by the previous trigger is completed, the pulse is extended.

To use the retriggerable monopulse mode, the timer must be in slave mode, the bit SMS [3: 0]= "1000" in the TIMx_SMCR register (combined mode-reset + trigger), and the OCxM [3: 0] bit set to "1000" or "1001" (retriggerable OPM mode 1 or 2).

If the timer is configured in Up-counting mode, the corresponding CCRx must be set to 0 (the ARR register sets the pulse length). If the timer is configured in Down-counting mode, CCRx must be above or equal to ARR.

Note: The OCxM[3:0] and SMS[3:0] bit fields are split into two parts for compatibility reasons, the most significant bit are not contiguous with the 3 least significant ones.

This mode must not be used with center-aligned PWM modes. It is mandatory to have CMS[1:0] = 00 in TIMx_CR1.

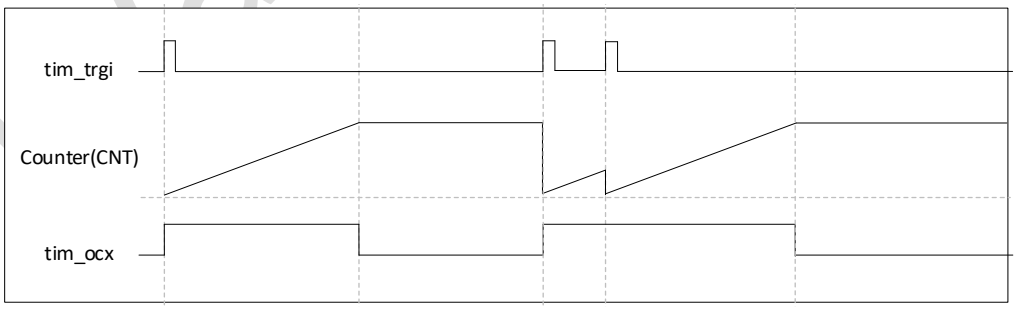


Figure 14-42 Example of a retriggerable one-pulse mode

14.3.17. Encoder interface mode

To select Encoder Interface mode: write SMS='001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS=010 if it is counting on TI1 edges only and SMS=011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, you can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Referring to Table 14-1, assuming that the counter has been started (CEN = 1 in the TIMx_CR1 register), the counter is driven by each valid jump on TI1FP1 or TI2FP2. TI1FP1 and TI2FP2 are the signals of TI1 and TI2 after passing through the input filter and polarity control; If there is no filtering and disguised phase, then TI1FP1 = TI1 and TI2FP2 = TI2. The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So the TIMx_ARR must be configured before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together. In this mode, the counter is modified automatically following the speed and the direction of the quadrature encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor. The table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 14-1 Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Counting on TI2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicate the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points. For this example we assume that the configuration is the following:

- CC1S='01' (TIMx_CCMR1 register, TI1FP1 mapped on TI1).

- CC2S= 01 (TIMx_CCMR2 register, TI2FP2 mapped on TI2)
- CC1P and CC2NP = ‘0’ (TIMx_CCER register, TI1FP1 non-inverted, TI1FP1=TI1)
- CC2P and CC2NP = ‘0’ (TIMx_CCER register, TI2FP2 non-inverted, TI2FP2=TI2)
- SMS= 011 (TIMx_SMCR register, both inputs are active on both rising and falling edges)
- CEN='1' (TIMx_CR1 register, Counter enabled)

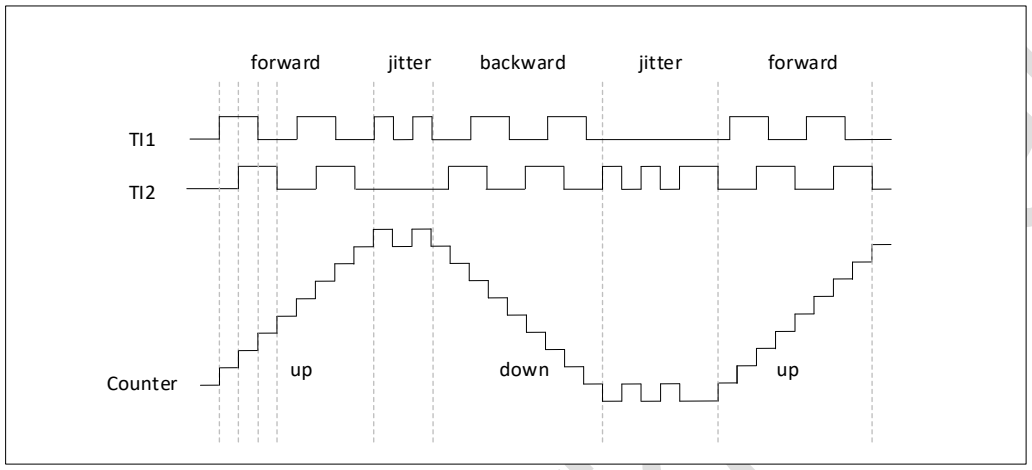


Figure 14-43 Example of counter operation in encoder interface mode

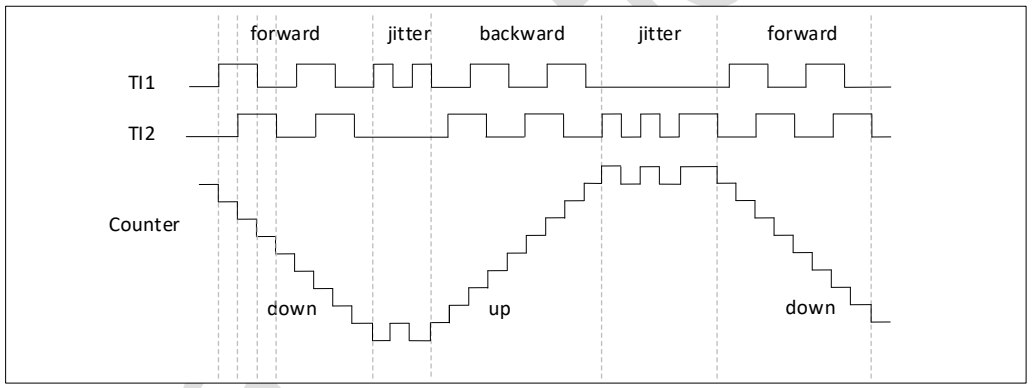


Figure 14-44 Example of encoder interface mode with TI1P1 polarity inverted.

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).

14.3.18. Timer input XOR function

The TI1S bit in the TIM_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

14.3.19. Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIMx referred to as interfacing timer. The “interfacing timer” captures the 3 timer input pins (CC1, CC2, CC3) connected through an XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

The slave mode controller is configured in reset mode; the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

On the interfacing timer, capture/compare channel 1 is configured in capture mode, capture signal is TR. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1) through the TRGO output.

Example: one wants to change the PWM configuration of the advanced-control timer after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIMx_CR2 register to ‘1’.
- Program the time base: write the TIMx_ARR to the max value (the counter must be cleared by the TI1 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors.
- Program the channel 1 in capture mode (TRC selected): write the CC1S bits in theTIMx_CCMR1 register to ‘01’. The digital filter can also be programmed if needed,
- Program channel 2 in PWM 2 mode with the desired delay: write the OC2M[2:0] bits to ‘111’ and the CC2S[1:0] bits to ‘00’ in the TIMx_CCMR1 register.
- Select OC2REF as trigger output on TRGO: write the MMS[2:0] bits in the TIMx_CR2 register to ‘101’.

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the TIMx_CR2 register) and the COM event is controlled by the trigger input (CCUS=1 in the TIMx_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step, which can be done in an interrupt subroutine generated by the rising edge of OC2REF).

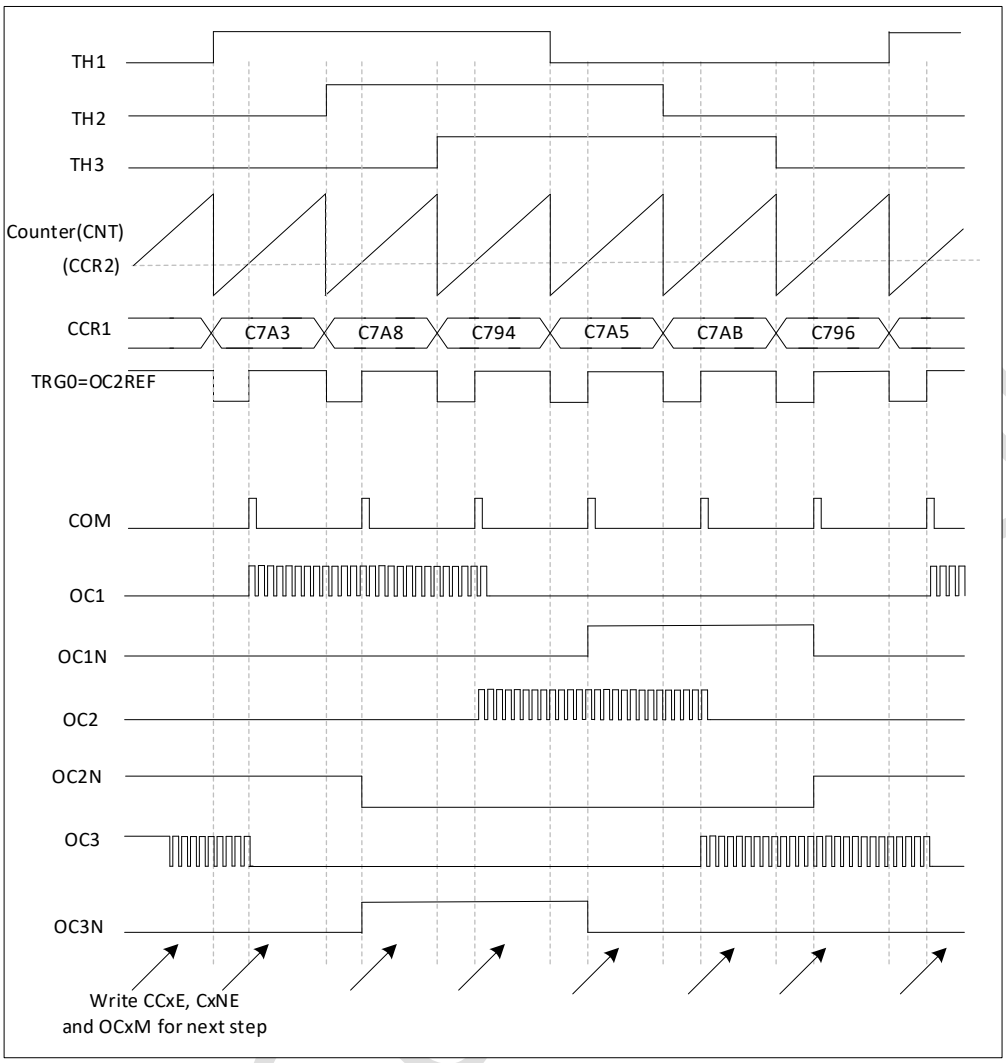


Figure 14-45 Example of Hall sensor interface

14.3.20. TIM and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F[3:0]=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[1:0]=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS[2:0]=100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.

- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request can be sent if enabled (depending on the TIE bit 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.

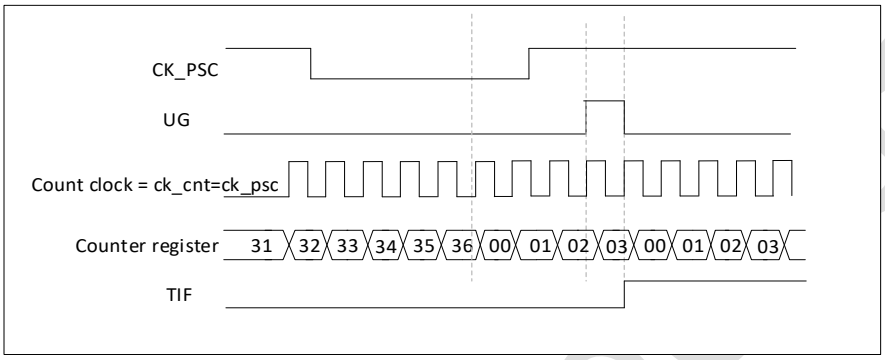


Figure 14-46 Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. SelectTI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

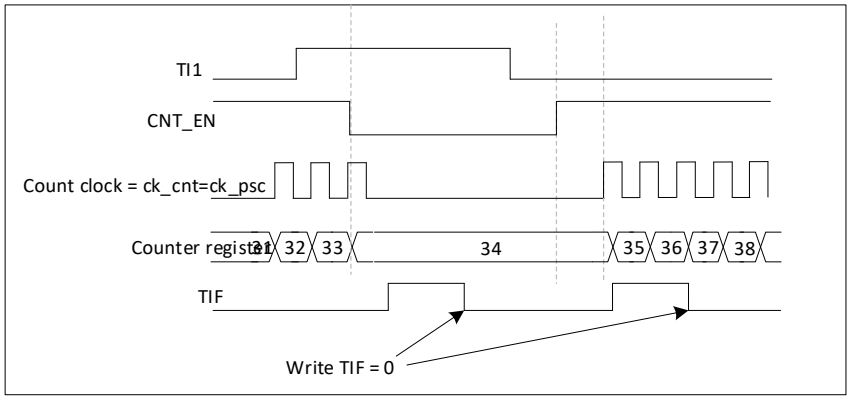


Figure 14-47 Control circuit in Gated mode

Slave mode: Trigger mode

The selected event on the input enables the counter.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we do not need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMx_CCMR1 register. Write CC2P=1 in TIMx_CCER register to validate the polarity (and detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI2 as the input source by writing TS=110 in TIMx_SMCR register.

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set. The delay between the rising edge on TI2 and the actual start of the counter is due to the resynchronization circuit on TI2 input.

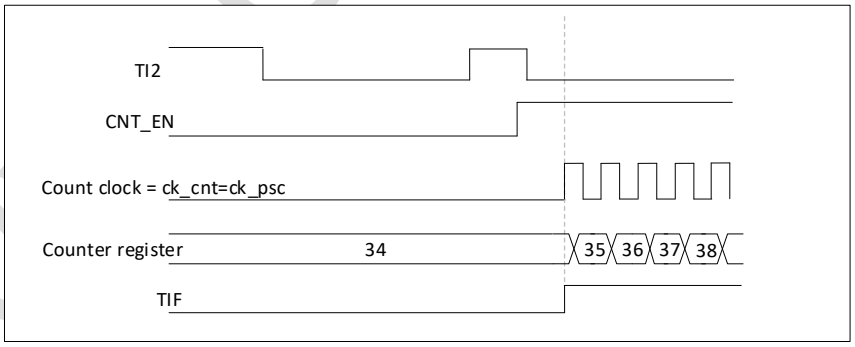


Figure 14-48 Control circuit in Gated mode

Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when operating in reset mode, gated mode or trigger mode. It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

1. Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
   - ETF = 0000: no filter
   - ETPS = 00: prescaler disabled
   - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
2. Configure the channel 1 as follows, to detect rising edges on TI:
   - IC1F = 0000: no filter
   - The capture prescaler is not used for triggering, so it does not need to be configured.
   - CC1S=01 in TIMx_CCMR1 register to select only the input capture source
   - Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
3. Configure the timer in trigger mode by writing SMS[2:0]=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.

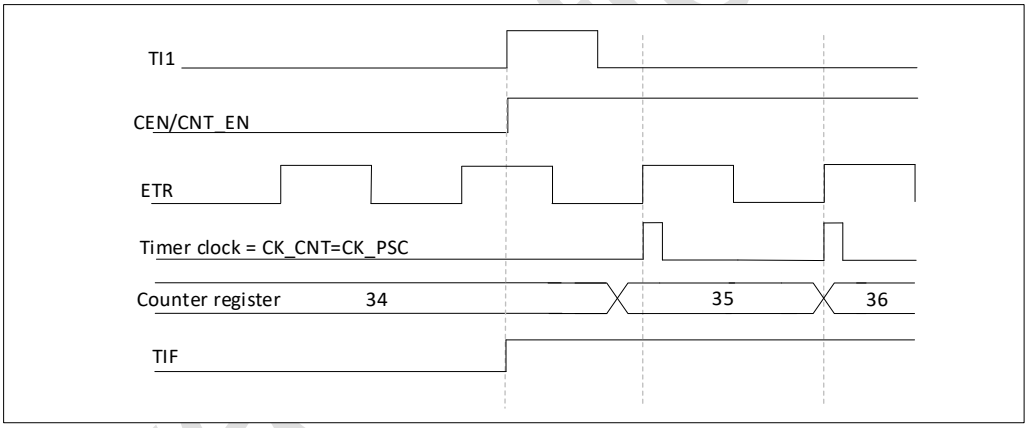


Figure 14-49 Control circuit in external clock mode 2 + trigger mode

14.3.21. Debug mode

When the microcontroller enters debug mode, the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

14.4. TIM1 registers

14.4.1. TIM1 control register 1 (TIM1_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9:8	CKD[1:0]	RW	00	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock used by the dead-time generators and the digital filters (ETR, TIx) 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 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
6:5	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. Configured as output channel Output compare interrupt flags of channels configured in output (CCxS=00 in TIM1_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. The counter counts up and down alternatively. Output comparison interrupt mark of channel configured as output (CCxS = 00 in TIM1_CCMRx register) 11: Center-aligned mode 3. The counter counts up and down alternatively. The counter counts up and down alternatively. Output comparison interrupt mark of channel configured as output (CCxS = 00 in TIM1_CCMRx register) Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
4	DIR	RW	0	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	RW	0	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN).
2	URS	RW	0	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt 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 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:

Bit	Name	R/W	Reset Value	Function
				– 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.

14.4.2. TIM1 control register 2 (TIM1_CR2)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			Res	CCUS	Res	CCPC
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	-	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	OIS4	RW		Output idle state 4 (OC4 output) Refer to OIS1 bit.
13	OIS3N	RW	0	Output idle state 3 (OC3N output) Refer to OIS1N bit
12	OIS3	RW	0	Output idle state 3 (OC3 output) Refer to OIS1 bit.
11	OIS2N	RW	0	Output idle state 2 (OC2N output) Refer to OIS1N bit
10	OIS2	RW	0	Output idle state 2 (OC2 output) Refer to OIS1 bit
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BKR register).
8	OIS1	RW	0	Output idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 Note: this bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BKR register).
7	TI1S	RW	0	TI1 selection 0: The TIM1_CH1 pin is connected to TI1 input 1: The TIM1_CH1, CH2 and CH3 pins are connected to the TI1 input.
6:4	MMS[2:0]	RW	000	Master mode selection These two bits are used to select 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 TIM1_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 enabled. The Counter Enable signal is generated by a logic AND between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM1_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 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) Note: 1. The clock of the slave timer or ADC must be enabled prior to receive events from the master timer, and must not be changed. 2. If the master and slave timers are not on the same bus, the master mode should be configured to the width that can be picked by the slave timer.
3	Reserved	-	-	Reserved
2	CCUS	RW	0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit only. 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	-	-	Reserved
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded. 1: CCxE, CCxNE, and OCxM bits are preloaded; Once this bit is set, they are only set when COM Note: This bit acts only on channels that have a complementary output.

14.4.3. TIM1 slave mode control register (TIM1_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SMS[3]
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	Reserved
16	SMS[3]	RW	0	See SMS description for details
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge.

Bit	Name	R/W	Reset Value	Function
				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 drived by any active edge on the ETRF signal. Note: 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gate mode and trigger mode; However, the TRGI cannot be connected to the ETRF at this time (the TS bit cannot be '111'). Note 3: When External Clock Mode 1 and External Clock Mode 2 are enabled simultaneously, the input of the external clock is ETRF
13:12	ETPS[1:0]	RW	00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIM1 CLK frequency. When a faster external clock is input, the frequency of the ETRP can be reduced using prescalation. 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. These bits define the frequency at which the ETRP signal is sampled and the bandwidth at which the ETRP is digitally filtered. In fact, the digital filter is an event counter, which will produce an output jump after recording N events. 0000: No filter, sampling is done at f_{DTS} 0001: $f_{SAMPLING}=f_{CK\_INT}$, N=2 0010: $f_{SAMPLING}=f_{CK\_INT}$, N=4 0011: $f_{SAMPLING}=f_{CK\_INT}$, N=8 0010: $f_{SAMPLING}=f_{CK\_INT}/2$, N=6 0101: $f_{SAMPLING}=f_{CK\_INT}/2$, N=8 0110: $f_{SAMPLING}= fCK\_INT/4$, N = 6 0111: $f_{SAMPLING}= fCK\_INT/4$, N = 8 1000: $f_{SAMPLING}= fCK\_INT/8$, N = 6 1001: $f_{SAMPLING}= fCK\_INT/8$, N = 8 1010: $f_{SAMPLING}= fCK\_INT/16$, N = 5 1011: $f_{SAMPLING}= fCK\_INT/16$, N = 6 1100: $f_{SAMPLING}= fCK\_INT/16$, N = 8 1101: $f_{SAMPLING}= fCK\_INT/32$, N = 5 1110: $f_{SAMPLING}= fCK\_INT/32$, N = 6 1111: $f_{SAMPLING}= fCK\_INT/32$, N = 8 It must be noted that when ETF [3: 0] = 1 or 2 or 3, f_{DTS} is replaced by CK_INT in the equation
7	MSM	RW	0	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
6:4	TS[2:0]	RW	000	Trigger selection. This bit-field selects the trigger input to be used to synchronize the counter. 000: Reserved (ITR0) Note: These bits can only be changed when not used (e.g. SMS = 000) to avoid erroneous edge detection when changing
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF_CLR_INT clear source is connected to OCREF_CLR input 1: OCREF_CLR_INT is connected to ETRF
2:0	SMS[2:0]	RW	000	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 000: Slave mode disabled If CEN = ‘1’ then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1

Bit	Name	R/W	Reset Value	Function
				Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. If SMS [3] = 0: 010: Encoder mode 2 Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 011: Encoder mode 3 The counter counts up/down on the edges of TI1FP1 and TI2FP2 depending on the level of the other inputs. 100: Reset mode Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated mode The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal. If SMS [3] = 1, SMS [2: 0] must be configured to 0. 000: Combined "reset + trigger" mode-the rising edge of the selected trigger input (tim_trgi) reinitializes the counter, generates a register update and starts the counter. Note: In encoder mode, do not use uev as trgo output signal, (i.e. mms cannot be configured to 010)

Table 14-2 TIM1 internal trigger connection

Slave TIM	ITR0 (TS=000)
TIM1	TIM14

14.4.4. TIM1 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	Res	Res	Res	Res	Res	Res	Res	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	Reserved
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	CC4IE: Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	RW	0	CC3IE: Capture/Compare 3 interrupt enable

Bit	Name	R/W	Reset Value	Function
				0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	RW	0	CC2IE: Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

14.4.5. TIM1 status register (TIM1_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	IC2IR	IC1IR
-	-	-	-	-	-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	-	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31: 24	Reserved	-	-	Reserved
23	IC4IF	RC_W0	0	Falling edge capture 4 flag Refer to IC1IF description
22	IC3IF	RC_W0	0	Falling edge capture 3 flag Refer to IC1IF description
21	IC2IF	RC_W0	0	Falling edge capture 2 flag Refer to IC1IF description
20	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by falling edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A falling edge capture event occurs.
19	IC4IR	RC_W0	0	Rising edge capture 4 flag Refer to IC1IR description
18	IC3IR	RC_W0	0	Rising edge capture 3 flag Refer to IC1IR description
17	IC2IR	RC_W0	0	Rising edge capture 2 flag Refer to IC1IR description
16	IC1IR	RC_W0	0	Rising edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by rising edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A rising edge capture event occurs.
15:13	Reserved	-	-	Reserved
12	CC4OF	RC_W0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description
11	CC3OF	RC_W0	0	Capture/Compare 3 overcapture flag Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 overcapture flag Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’.

Bit	Name	R/W	Reset Value	Function
				0: No overcapture has been detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1OF flag was already set.
8	Reserved	-	-	Reserved
7	BIF	RC_W0	0	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.
6	TIF	RC_W0	0	Trigger interrupt flag When a trigger event occurs (when the slave mode controller is in a mode other than the gated mode, it is detected at the TRGI input that Effect edge, or either edge in gated mode) by hardware to the position 1. It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	RC_W0	0	COM interrupt flag Once a COM event is generated (when CCxE, CCxNE, OCxM have been updated) this bit is set to 1 by the hardware. It is cleared by software. 0: No update occurred. 1: COM interrupt pending.
4	CC4IF	RC_W0	0	Capture/Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	RC_W0	0	Capture/Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	RC_W0	0	Capture/Compare 2 interrupt flag Refer to CC1IF description
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set to 1 by hardware when the counter value matches the comparison value, except in centrosymmetric mode (refer to TIM1_CR1 register It is cleared by software. 0: No match; 1: The content of the counter TIM1_CNT matches the content of the TIM1_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM1_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM1_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	RC_W0	0	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – At overflow or underflow regarding the repetition counter value (update if REP_CNT = 0) and if the UDIS=0 in the TIM1_CR1 register. – When CNT is reinitialized by software using the UG bit in TIM1_EGR register, if URS=0 and UDIS=0 in the TIM1_CR1 register. –When CNT is reinitialized by a trigger event, if URS=0 and UDIS=0 in the TIM1_CR1 register. (Refer to TIM1 slave mode control register (TIM1_SMCR))

14.4.6. TIM1 event generation register (TIM1_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	-	-	-	-	-	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	Reserved
7	BG	W	0	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt can occur if enabled.
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 TIM1_SR register. Related interrupt 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 that have a complementary output.
4	CC4G	W	0	Capture/Compare 4 generation Refer to CC1G description
3	CC3G	W	0	Capture/Compare 3 generation Refer to CC1G description
2	CC2G	W	0	Capture/Compare 2 generation Refer to CC1G description
1	CC1G	W	0	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as input: CC1IF flag is set, Corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM1_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reloadvalue (TIM1_ARR) if DIR=1 (downcounting).

14.4.7. TIM1 capture/compare mode register 1 (TIM1_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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
15	OC2CE	RW	0	Output Compare 2 clear enable
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	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 (TIM1_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER).
7	OC1CE	RW	0	Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF signal; 1: OC1REF is cleared as soon as a High level is detected on ETRF signal.
6:4	OC1M [2:0]	RW	0	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 andOC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output comparison register TIM1_CCR1 and the counter TIM1_CNT does not work for OC1REF; 001: Set channel 1 to active level on match. 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 TIM1_CNT<TIM1_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0') as long as TIM1_CNT>TIM1_CCR1 else active (OC1REF='1'). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIM1_CNT<TIM1_CCR1 else active. In downcounting, channel 1 is active as long as TIM1x_CNT>TIM1_CCR1 else inactive. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIM1_CCR1 disabled. TIM1_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM1_CCR1 enabled. Read/Write operations access the preload register. TIM1_CCR1 preload value is loaded in the active register at each update event. Note: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note: The PWM mode can be used without validating the preload register only in one pulse mode. Else the behavior is not guaranteed.

Bit	Name	R/W	Reset Value	Function
2	OC1FE	RW	0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on OC1 output. Then, OC1 is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	0	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: 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 the TS bit (TIM1_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER).

Input capture mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IC2F [3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F [3:0]				IC1PSC [1:0]		CC1S [1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	IC2F[3:0]	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 the TS bit (TIM1_SMCR register). Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIM1_CCER).
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 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, sample 1000 at f_{DTS}: sampling frequency $f_{SAMPLING} = f_{DTS/8}$, N = 6 0001: Sampling frequency $f_{SAMPLING}$= fCK_INT, N = 2 1001: Sampling frequency $f_{SAMPLING}$ = fDTS/8, N = 8 0010: Sampling frequency $f_{SAMPLING}$= fCK_INT, N = 4 1010: Sampling frequency $f_{SAMPLING}$ = fDTS/16, N = 5 0011: Sampling frequency $f_{SAMPLING}$= fCK_INT, N = 8 1011: Sampling frequency $f_{SAMPLING}$= $f_{DTS/16}$, N = 6

Bit	Name	R/W	Reset Value	Function
				0100: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/2}$, N = 6 1100: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/16}$, N = 8 0101: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/2}$, N = 8 1101: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/32}$, N = 5 0110: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/4}$, N = 6 1110: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/32}$, N = 6 0111: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/4}$, N = 8 1111: Sampling frequency $f_{SAMPLING}$ = $f_{DTS/32}$, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: 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 the TS bit (TIM1_SMCR register). Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM1_CCER).

14.4.8. 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
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. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).
7	OC3CE	RW	0	Output Compare 3 clear enable
6:4	OC3M [2:0]	RW	00	Output compare 3 mode
3	OC3PE	RW	0	Output Compare 3 preload enable

2	OC3FE	RW	0	Output Compare 3 fast enable
1:0	CC3S [1:0]	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (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 the TS bit (TIM1_SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIM1_CCER).

Input capture mode:

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15:12	IC4F	RW	0000	Input capture 4 filter
11:10	IC4PSC	RW	00	Input/capture 4 prescaler
9:8	CC4S	RW	00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4. 10: CC4 channel is configured as input, IC4 is mapped on TI3. 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register). Note: CC4S bits are writable only when the channel is OFF (CC4E = '0' in TIM1_CCER).
7:4	IC3F	RW	0000	Input capture 3 filter
3:2	IC3PSC	RW	00	Input/capture 3 prescaler
1:0	OC3S	RW	00	Capture/Compare 3 selection This bit-field defines the direction of the channel (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, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIM1_SMCR register). Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIM1_CCER).

14.4.9. 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	-	-	Reserved
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 output enable Refer to CC1E description.
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.

Bit	Name	R/W	Reset Value	Function
10	CC3NE	RW	0	Input/Capture 3 complementary output enable. Refer to CC1NE description.
9	CC3P	RW	0	Input/Capture 3 output polarity Refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable Refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	CC2NE	RW	0	Input/Capture 2 complementary output enable. Refer to CC1NE description.
5	CC2P	RW	0	Input/Capture 2 output polarity Refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable Refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 0: OC1N active high. 1: OC1N active low. Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register) and CC1S='00' (channel configured as output).
2	CC1NE	RW	0	Input/Capture 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1Nand CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: CC1NP/CC1P bits select the active polarity of TI1FP1and TI2FP1 for trigger or capture operations. 00: non-inverted/rising edge. The circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode or encoder mode). 01: inverted/falling edge. The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is inverted (trigger operation in gated mode or encoder mode). 10: reserved, do not use this configuration. 11: inverted/falling edge. The circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). TIxFP1 is not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. 2. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).
0	CC1E	RW	0	Input/Capture 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 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 (TIM1_CCR1) or not. 0: Capture disabled. 0: Capture enabled. Notes: For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.

Table 14-3 Output control bits for complementary OCx and OCxN channels with break feature

Control bit					Output status	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx output state**	**OCxN output state**
1	X	0	0	0	Output disabled (not driven by the timer), OCx=0, OCx_EN=0	Output disabled (not driven by the timer), OCxN=0, OCxN_EN=0
		0	0	1	Output disabled (not driven by the timer), OCx=0, OCx_EN=0	OCxREF + Polarity OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + Polarity OCx=OCREF xor CCxP, OCx_EN=1	Output disabled (not driven by the timer), OCxN=0, OCxN_EN=0
		0	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
		1	0	0	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
		1	0	1	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=1	OCxREF+Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+Polarity OCx=OCxREF xor CCxP, OCx_EN=1	Off-State (output enabled with inactive state), OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCN_EN=1
0	0	X	0	0	Output disabled (not driven by the timer), OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
	0		0	1	Output disabled (not driven by the timer anymore). Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 If the clock ispresent: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	0		1	0		
	0		1	1		
	1		0	0	Output disabled (not driven by the timer anymore). OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer), OCxN=CCxNP, OCxN_EN=0
	1		0	1	Off-State (output enabled with inactive state) Asynchronously: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 If the clock ispresent: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCX and OCxN both in active state.	
	1		1	0		
	1		1	1		

In particular, when the dead band is active, the output is always output according to the rules of dead band output, although moe may have been set to 1 by software or hardware at this time. If neither of the two outputs of a channel is used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP, and CCxNP must all be cleared.

Note: The state of the external I/O pins connected to complementary OCx and OCxN channels depends on the OCx and OCxN channel status and the GPIO and AFIO registers.

14.4.10. TIM1 counter register (TIM1_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

14.4.11. TIM1 prescaler register (TIM1_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler 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).

14.4.12. 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
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.

14.4.13. TIM1 repetition counter register (TIM1_RCR)

Address offset: 0x30

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	REP[7:0]	RW	0	Repetition counter value 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 enabled, as well as the update interrupt generation rate, if this interrupt is enabled. 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 TIM1_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.

14.4.14. TIM1 capture/compare register 1 (TIM1_CCR1)

Address offset: 0x34

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

14.4.15. TIM1 capture/compare register 2 (TIM1_CCR2)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR2 [15:0]	RW	0	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signaled on OC output. CC2 channel configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).

14.4.16. TIM1 capture/compare register 3 (TIM1_CCR3)

Address offset: 0x3C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15:0	CCR3 [15:0]	RW	0	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signaled on OC output. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).

14.4.17. TIM1 capture/compare register 4 (TIM1_CCR4)

Address offset: 0x40

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4 [15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15:0	CCR4 [15:0]	RW	0	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR4 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM1_CNT and signaled on OC output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4).

14.4.18. TIM1 break and dead-time register (TIM1_BDTR)

Address offset: 0x44

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	**AOE**	**BKP**	**BKE**	**OSSR**	**OSSI**	**LOCK[1:0]**		**DTG[7:0]**							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

Bit	Name	R/W	Reset Value	Function
15	MOE	RW	0	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active. It is cleared by software or automatically setting depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIM1_CCER register).
14	AOE	RW	0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if none of the break inputs is active). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
13	BKP	RW	0	Break polarity 0: Break input BRK is active low; 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
12	BKE	RW	0	Break enable 0: Brake input disabled (BRK and BRK_ACTH); 1: Brake input enabled (BRK and BRK_ACTH). Note: This bit can not be modified as long as LOCK level 1 has been set (LOCK bits in TIM1_BDTR register).
11	OSSR	RW	0	Off-state selection for Run mode This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR bit is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details (Section 14.4.9: TIM1 capture/compare enable register (TIM1_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
10	OSSI	RW	0	Off-state selection for Idle mode This bit is used when MOE=0 and on channels configured as outputs. See OC/OCN enable description for more details (Section 14.4.9: TIM1 capture/compare enable register (TIM1_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal = 0). 1: When inactive, OC/OCN outputs are first forced to their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been set (LOCK bits in TIM1_BDTR register).
9:8	LOCK[1:0]	RW	00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: LOCK Level 1, DTG, BKE, BKP, AOE bits in TIM1_BDTR register and OISx and OISxN bits in TIM1_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIM1_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIM1_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written.

Bit	Name	R/W	Reset Value	Function
				Note: The LOCK bits can be written only once after the reset. Once the TIM1_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	0000 0000	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG[7:5]=0xx => DT=DTG[7:0] × T_{dtg}, T_{dtg} = T_{DTS}; DTG[7:5]=10x => DT=(64+DTG[5:0]) × T_{dtg}, T_{dtg} = 2 × T_{DTS}; DTG[7:5]=110 => DT=(32+DTG[4:0]) × T_{dtg}, T_{dtg} = 8 × T_{DTS}; DTG[7:5]=111 => DT=(32+DTG[4:0]) × T_{dtg}, T_{dtg} = 16 × T_{DTS}; Example if T_{DTS}=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps; 16us to 31750ns, if the step time is 250ns; 32us to 63us, if the step time is 1us; 64 us to 126 us by 2 us steps Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIM1_BDTR register).

15. General-purpose timer (TIM14)

15.1. TIM14 introduction

The universal timer TIM14 consists of a 16-bit auto-load counter driven by a programmable prescaler. It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM).

Using a timer prescaler, the pulse length and waveform period can be adjusted from a few microseconds to a few milliseconds.

The TIM14 timers are completely independent and do not share any resources with each other.

15.2. 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

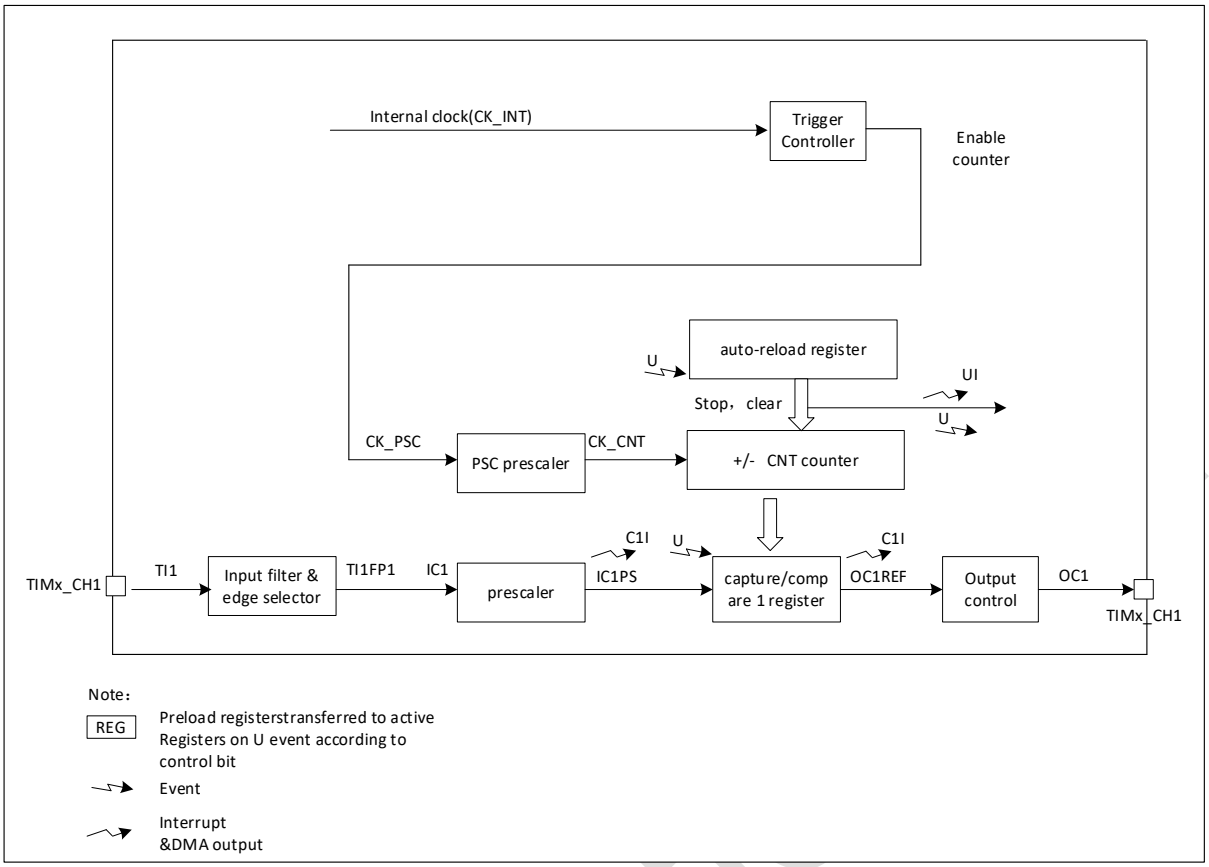


Figure 15-1 General-purpose timer block diagram (TIM14)

15.3. TIM14 functional description

15.3.1. Time-base unit

The main block of the timer is a 16-bit up-counter with its related auto-reload register. The counter clock can be divided by a prescaler.

The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIM14_CNT)
- Prescaler register (TIM14_PSC)
- Auto-reload register (TIM14_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM14_CR1 register. The update event is sent when the counter reaches the overflow value and if the UDIS bit equals 0 in the TIM14_CR1 register. It can also be generated by software.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM14_CR1 register is set.

Note that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM14_CR1register.

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.

Tables below give some examples of the counter behavior when the prescaler ratio is changed on the fly.

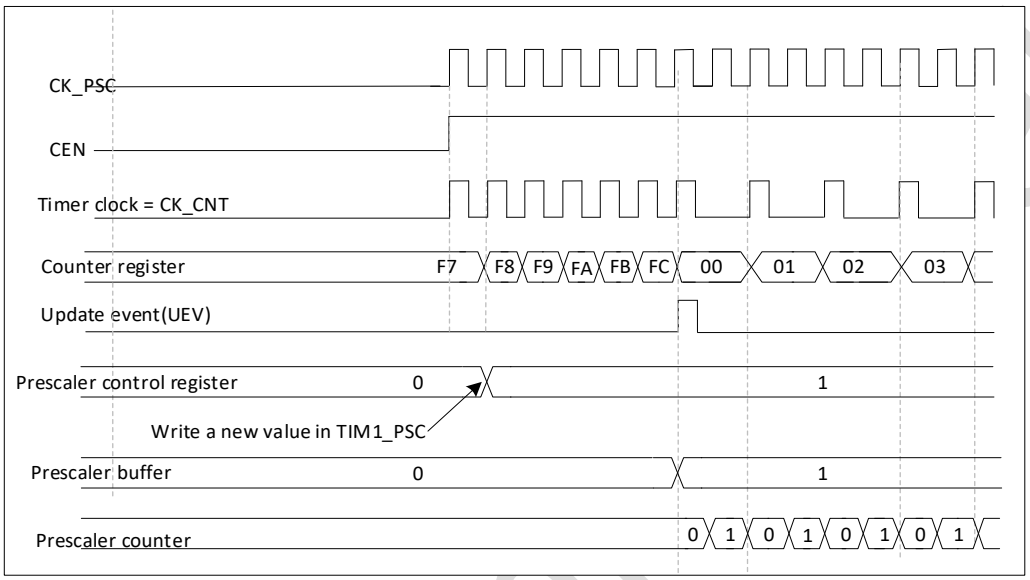


Figure 15-2 Counter timing diagram with prescaler division change from 1 to 2

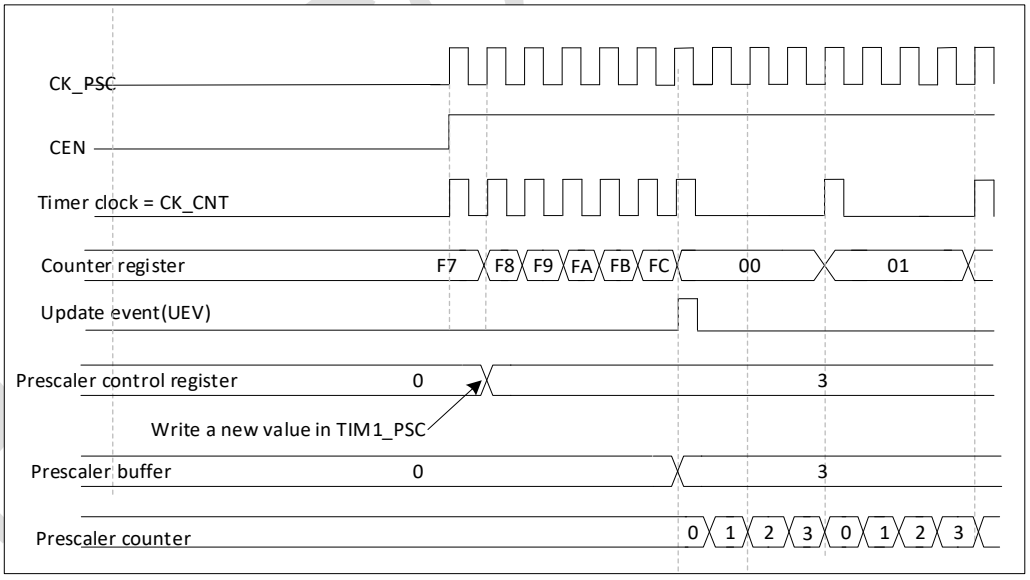


Figure 15-3 Counter timing diagram with prescaler division change from 1 to 4

15.3.2. Counting mode

Upcounting mode

The counter counts from 0 to the auto-reload value (contents of the TIM14_ARR register), then re-starts from 0 and generates a counter overflow event.

Else the update event is generated at each counter overflow. Setting the UG bit in the TIM14_EGR register (by software) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter 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 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.

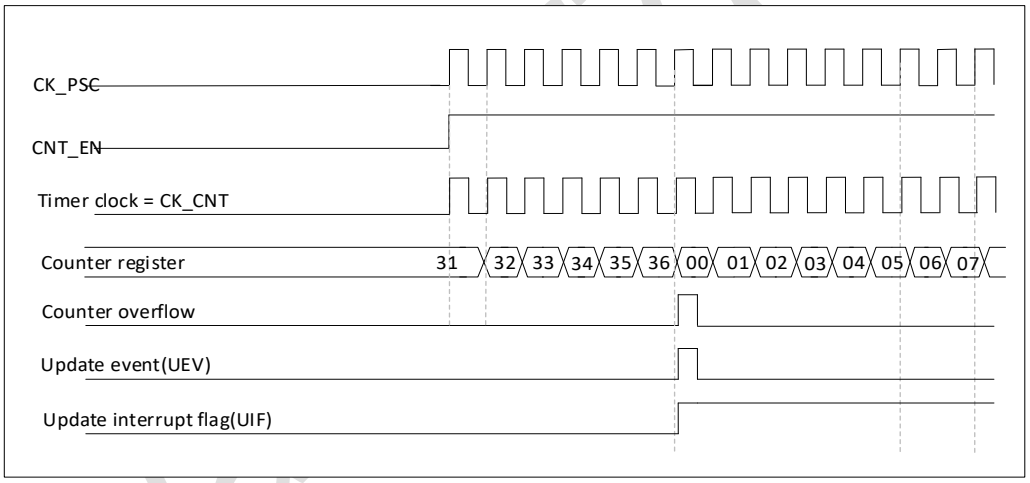


Figure 15-4 Counter timing diagram, internal clock divided by 1

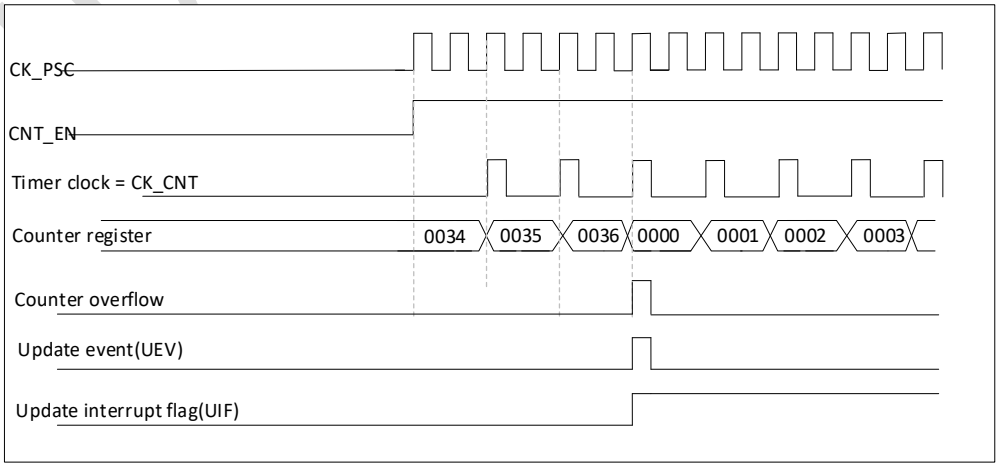


Figure 15-5 Counter timing diagram, internal clock divided by 2

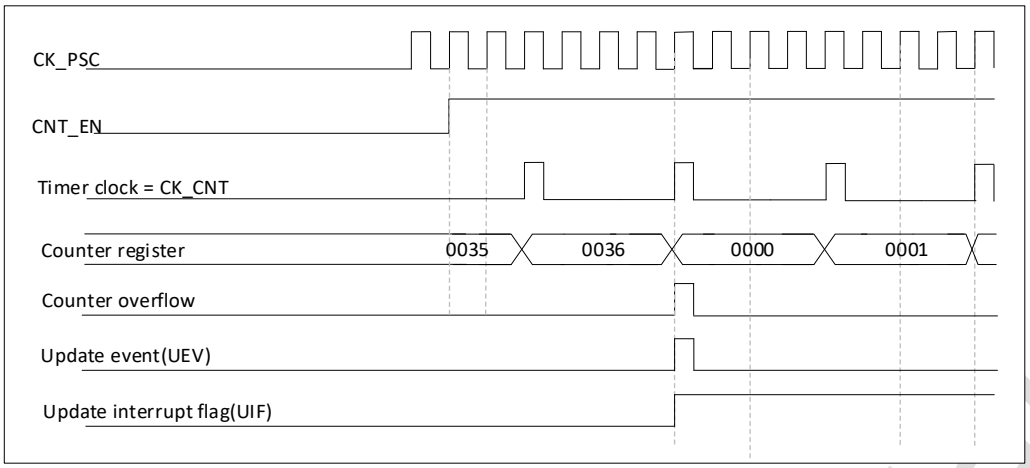


Figure 15-6 Counter timing diagram, internal clock divided by 4

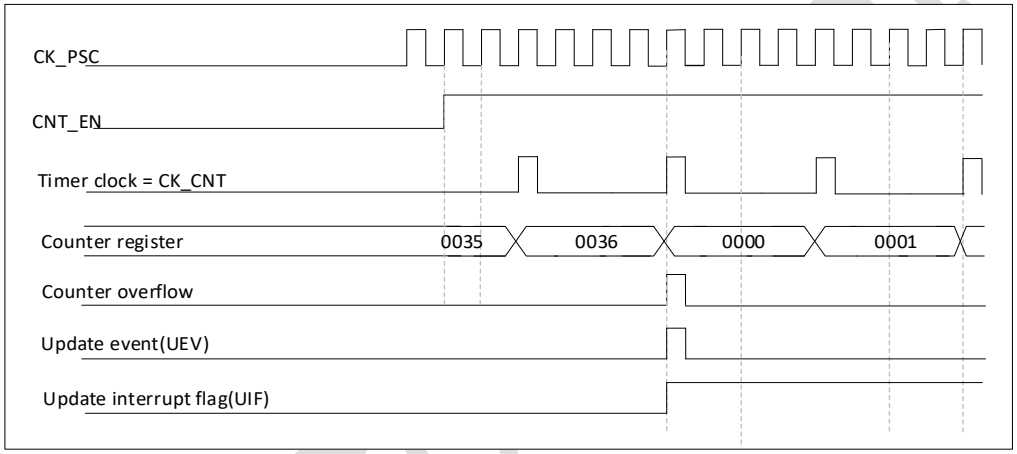


Figure 15-7 Counter timing diagram, internal clock divided by N

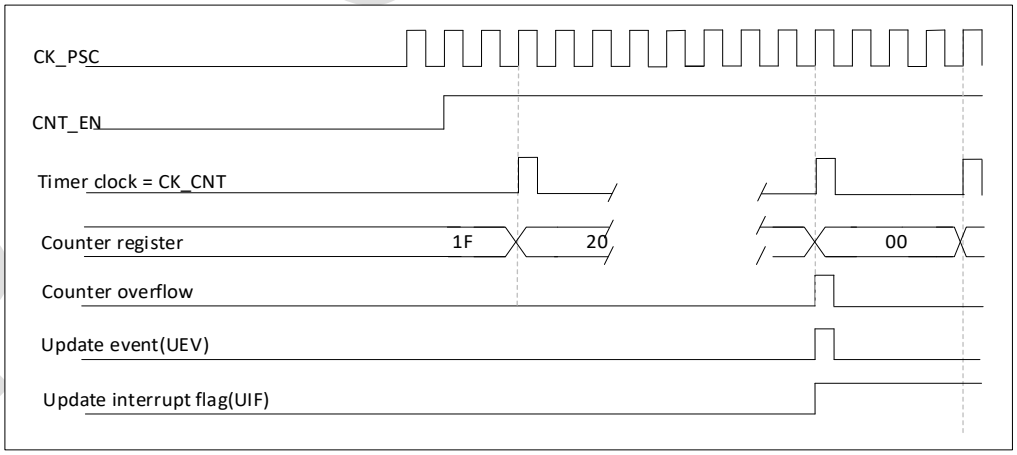


Figure 15-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

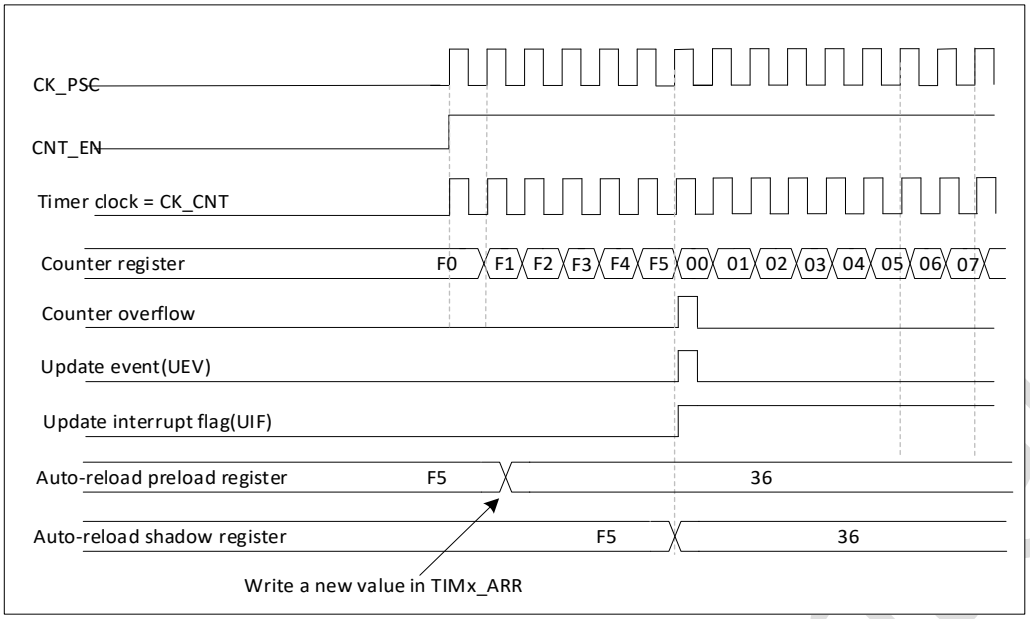


Figure 15-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

15.3.3. Clock sources

The counter clock is provided by the Internal clock (CK_INT) source. The CEN bit of the TIMx_CR1 register and the UG bit of the TIM14_EGR register are the actual control bits (except that the UG bit is automatically cleared) and can only be changed by software. Once the CEN bit is set to 1, the internal clock provides clock to the frequency divider.

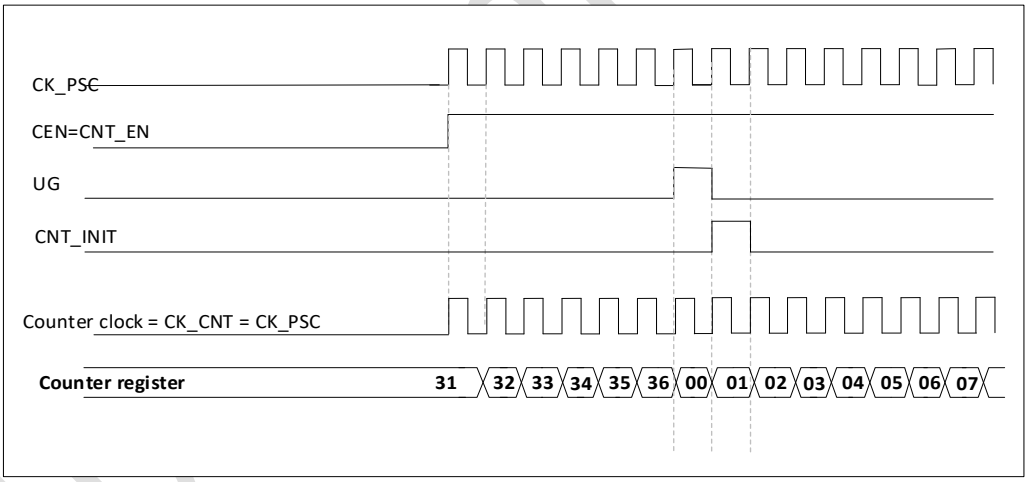


Figure 15-10 Control circuit in normal mode, internal clock divided by 1

15.3.4. Capture/Compare channels

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

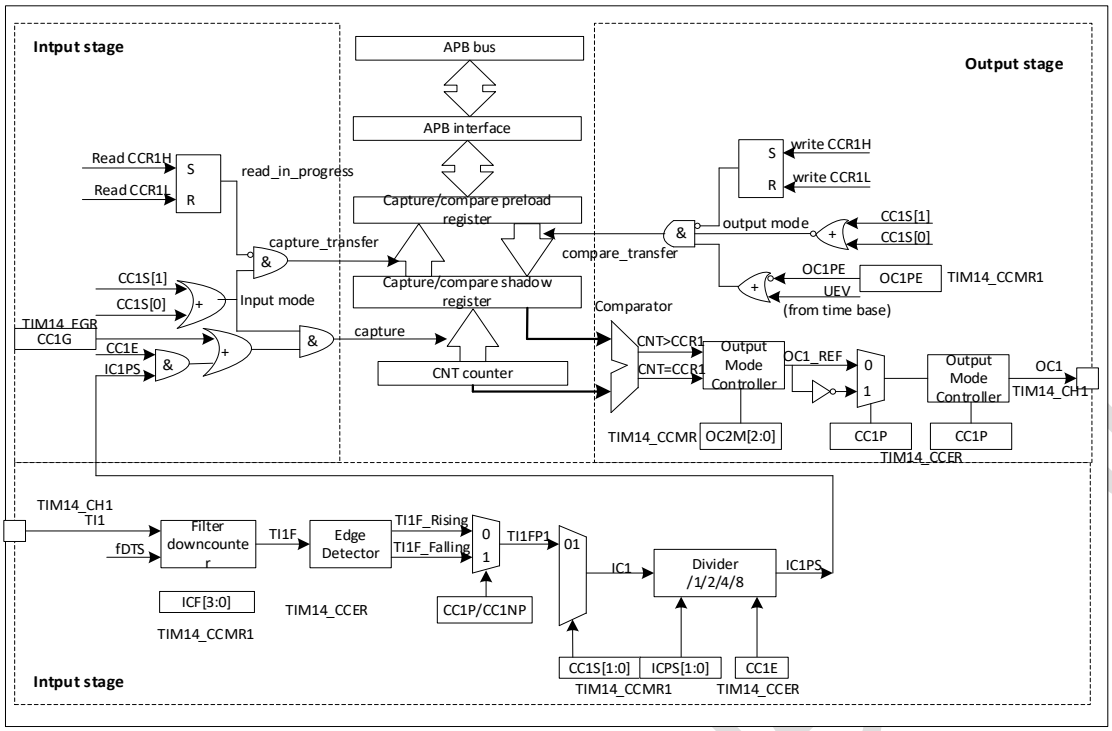


Figure 15-11 TIM14 block diagram

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 (active high) which is then used for reference. The polarity acts at the end of the chain.

The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

15.3.5. 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 you write it to ‘0’.

The following example shows how to capture the counter value in TIM14_CCR1 when TI1input rises. To do this, use the following procedure:

- Select the active output: 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 and the TIM14_CCR1 register becomes read-only.
- Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIM14_CCMRx register)). Let's imagine that, when toggling, the input signal is not stable during at least 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to'0011' in the TIM14_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by programming CC1P and CC1NP bits to '00' in the TIMx_CCER register (rising edge in this case).
- Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
- If needed, enable the related interrupt request by setting the CC1E 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 inthe TIMx_EGR register.

15.3.6. Forced output mode

In output mode (CCxS bits = 00 in the TIM14_CCMRx register), each output compares 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, you just need to write101 in the OCxM bits in the corresponding TIM14_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM14_CCMRx register.

Anyway, the comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt can be sent accordingly. This is described in the output compare mode section below.

15.3.7. Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM14_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets 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.

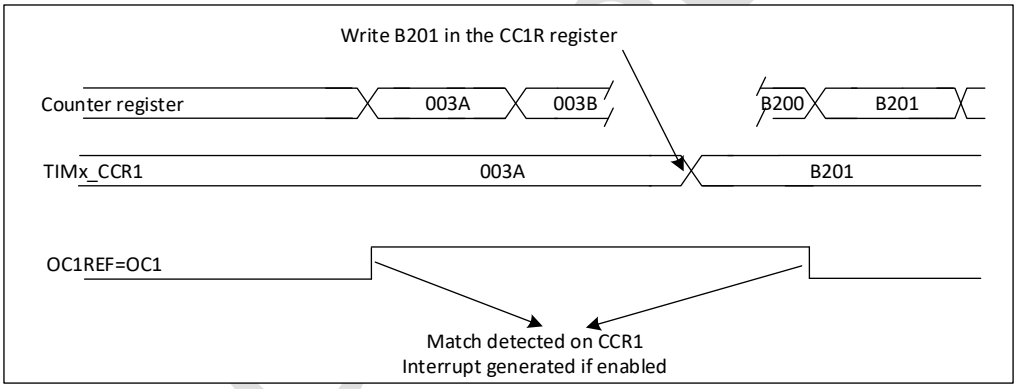


Figure 15-12 Output compare mode, toggle on OC1

15.3.8. Pulse width adjustment (PWM) mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing ‘110’ (PWM mode 1) or ‘111’ (PWM mode 2) in the OCxM bits in the TIM14_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIM14_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIM14_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIM14_CCER register. It can be programmed as active high or active low. The OCx output is enabled by the CCxE bit in the TIM14_CCER register.

In PWM mode (1 or 2), TIM14x_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT ≤ TIM14_CCRx.

The timer is able to generate PWM in edge-aligned mode only since the counter is upcounting.

PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIM14_CNT < TIMx_CCRx else it becomes low. If the compare value in TIM14_CCRx is greater than the auto-reload value (in TIM14_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'.

Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

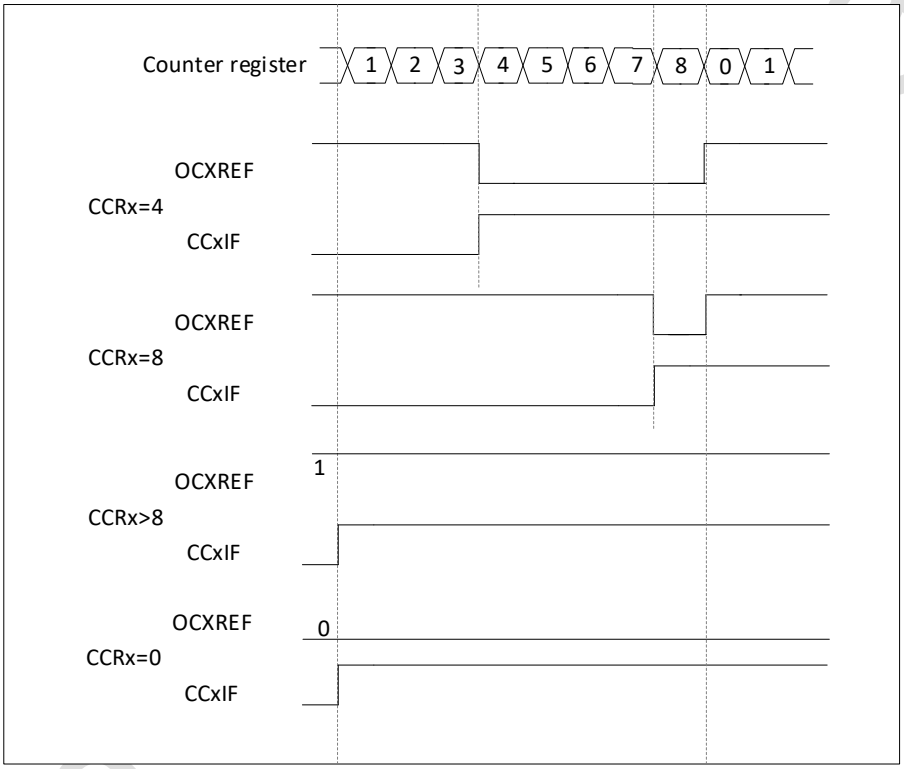


Figure 15-13 Edge-aligned PWM waveforms (ARR=8)

15.3.9. Synchronous mode

All TIMx timers are connected internally for timer synchronization or linking. When one timer is in master mode, it can reset, start, stop, or provide a clock on the counter of another timer in slave mode.

15.3.10. Debug mode

When the microcontroller enters debug mode (Cortex®-M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

15.4. TIM14 registers

15.4.1. 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 00: $t_{DTS} = t_{CK}\_INT$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: reserved, do not use this configuration.
7	ARPE	RW	0	Auto-reload preload enable 0: TIM14_ARR register is not buffered 1: TIM14_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 and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt if enabled. These events can be: – Counter overflow/underflow − Setting the UG bit 1: Only counter overflow/underflow generates an update interrupt 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 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.

15.4.2. TIM14 Interrupt enable register (TIM14_DIER)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.														CC1IE	UIE
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	CC1IE	RW	0	CC1IE: Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled

Bit	Name	R/W	Reset Value	Function
				1: Capture/Compare 1 interrupt enabled
0	UIE	RW	0	UIE: Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled

15.4.3. TIM14 status register (TIM14_SR)

Address offset: 0x010

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC1IF	Res	Res	Res	IC1IR
-	-	-	-	-	-	-	-	-	-	-	RC_W0	-	-	-	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						CC1OF	Res							CC1F	UIF
-						RC_W0	-							RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	Reserved
20	IC1IF	RC_W0	0	Falling edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by falling edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A falling edge capture event occurs.
19:17	Res.	-	0	Reserved, must be kept at reset value.
16	IC1IR	RC_W0	0	Rising edge capture 1 flag This flag is set by hardware only when the corresponding channel is configured in input capture mode and triggered by rising edge. It is cleared by software or reading TIMx_CCR1. 0: No overcapture has been generated. 1: A rising edge capture event occurs.
15:10	Reserved	-	-	Reserved
9	CC1OF	Rc_w0	0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1IF flag was already set.
8:2	Reserved	-	-	Reserved
1	CC1IF	Rc_w0	0	Capture/Compare 1 interrupt flag **If channel CC1 is configured as output:** When the counter value matches the comparison value, this bit is set to 1 by the hardware and it is cleared to 0 by the software. 0: No match; 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. **If channel CC1 is configured as input:** This bit is set by hardware on a capture. It is cleared by software or by reading the TIM14_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM14_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Rc_w0	0	Update interrupt flag. This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: – At overflow and if UDIS '0' in the TIMx_CR1 register. If UDIS of the TIMx_CR1 register = 0 and URS = 0, an update occurs when UG of the TIMx_EGR register = 1

15.4.4. TIM14 event generation register (TIM14_EGR)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.														CC1G	UG
-														W	W

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	CC1G	W	0	Capture/compare 1 generation. This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: **If channel CC1 is configured as input:** CC1IF flag is set, corresponding interrupt is sent if enabled. **If channel CC1 is configured as input:** The current value of the counter is captured in TIM14_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if theCC1IF flag was already set.
0	UG	W	0	Update generation. This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected).

15.4.5. 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	Res	CC1S [1:0]	
-								-	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	Reserved
6:4	OC1M [2:0]	RW	00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output comparison register TIM1_CCR1 and the counter TIMx_CNT does not work for OC1REF 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-

Bit	Name	R/W	Reset Value	Function
				In upcounting, channel 1 is active as long as TIM1_CNT<TIM1_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0') as long as TIM1_CNT>TIM1_CCR1 else active (OC1REF='1'). 111: PWM mode 2 Channel 1 is inactive as long as TIMx_CNT < TIMx_CCR1 else active. Note: In PWM mode 1 or 2, the OCREF level changes when the result of the comparison changes or when the output compare mode switches from frozen to PWM mode.
3	OC1PE	RW	0	Output Compare 1 preload enable 0: Preload register on TIM14_CCR1 disabled. TIM14_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM14_CCR1 enabled. Read/Write operations access the preload register. TIM14_CCR1 preload value is loaded in the active register at each update event.
2	Reserved	-	-	Reserved
1:0	CC1S [1:0]	RW	00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved; 11: Reserved. Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM14_CCER).

Input capture mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								IC1F [3:0]				IC1PSC [1:0]		CC1S [1:0]	
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:4	IC1F [3:0]	RW	0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 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, sample 1000 at fDTS: sampling frequency fSAMPLING = fDTS/8, N = 6 0001: Sampling frequency fSAMPLING= fCK_INT, N = 2 1001: Sampling frequency fSAMPLING= fDTS/8, N = 8 0010: Sampling frequency fSAMPLING= fCK_INT, N = 4 1010: Sampling frequency fSAMPLING= fDTS/16, N = 5 0011: Sampling frequency fSAMPLING= fCK_INT, N = 8 1011: Sampling frequency fSAMPLING= fDTS/16, N = 6 0100: Sampling frequency fSAMPLING = fDTS/2, N = 6 1100: Sampling frequency fSAMPLING = fDTS/16, N = 8 0101: Sampling frequency fSAMPLING = fDTS/2, N = 8 1101: Sampling frequency fSAMPLING = fDTS/32, N = 5 0110: Sampling frequency fSAMPLING = fDTS/4, N = 6 1110: Sampling frequency fSAMPLING = fDTS/32, N = 6 0111: Sampling frequency fSAMPLING = fDTS/4, N = 8 1111: Sampling frequency fSAMPLING = fDTS/32, N = 8
3:2	IC1PSC [1:0]	RW	00	Input/capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0' (TIM1_CCER register).

Bit	Name	R/W	Reset Value	Function
				00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S [1:0]	RW	00	CC1S[1:0]: Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1. 10: Reserved 11: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIM14_CCER).

15.4.6. TIM14 capture/compare enable register (TIM14_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. CC1 channel configured as output: CC1NP must be kept cleared. CC1 channel configured as input: CC1NP bit is used in conjunction with CC1P to defineTI1FP1 polarity (refer to CC1P description).
2	Reserved	-	-	Reserved
1	CC1P	RW	0	Input/Capture 1 output polarity. CC1 channel configured as output: 0: OC1 active high. 1: OC1 active low. CC1 channel configured as input: Both CC1NP/CC1P bits select the active polarity of TI1FP1 and TI2FP1 for capture operations. 00: non-inverted/rising edge. Circuit is sensitive to TIxFP1 rising edge (capture or trigger operations in reset, external clock or trigger mode); 01: inverted/falling edge. Circuit is sensitive to TIxFP1 falling edge (capture or trigger operations in reset, external clock or trigger mode). 10: Reserved, invalid configuration. 11: non-inverted/both edges
0	CC1E	RW	0	Input/Capture 1 output enable **CC1 channel configured as output:** 0: Off - OC1 is not active 1: ON - OC1 signal output to corresponding output pin **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.

CcxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)

1	OCx= OCxREF+Polarity, OCx_EN=1

15.4.7. TIM14 counter (TIM14_CNT)

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CNT[15:0]	RW	0	Counter value

15.4.8. TIM14 prescaler (TIM14_PSC)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	PSC[15:0]	RW	0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler 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).

15.4.9. TIM14 auto-reload register (TIM14_ARR)

Address offset: 0x2C

Reset value: 0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	ARR[15:0]	RW	0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. For details, refer to 12.4.1: Time base unit updates and actions relating to ARR. The counter is blocked while the auto-reload value is null.

15.4.10. 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															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	CCR1 [15:0]	RW	0	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM1_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).

15.4.11. 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	-	-	Reserved
1:0	TI1_RMP	RW	0	Timer input 1 remap Set and cleared by software. 00: TIM14 channel 1 is connected to the GPIO. Refer to the multiplexing function in the datasheet for details. 01: TIM14 channel 1 is connected to RTCCLK. 10: TIM14 channel 1 is connected to HSE/32 clock 11: TIM14 channel 1 is connected to the MCU clock output (MCO). This configuration is determined by setting the MCOSEL[2: 0] of the RCC_CFG register.

16. Real-time clock (RTC)

16.1. Introduction

The real time clock is an independent timer. It has a set of continuous counting counters, which can provide a clock calendar function under the corresponding software configuration. Modifying the value of the counter can reset the current time and date of the system.

The RTC module and the clock configuration system (RCC_BDCR register) are in the protection zone, that is, the setting and time of the RTC remain unchanged after the system is reset.

Access to the RTC is disabled after the system is reset, this is to prevent accidental write operations to the RTC. Doing the following will enable access to the RTC:

- Set PWREN of register RCC_APBENR1 to enable power and clock.
- The DBP bit of the register PWR_CR1 is set to enable access to the BDTR register and RTC.

16.2. Main features

- Programmable pre-division factor: division factor up to 220
- 32-bit programmable counter for measurements over longer periods
- 2 separate clocks: PCLK1 and RTC clock for APB1 interface (the frequency of the RTC clock must be less than more than a quarter of the PCLK1 clock frequency)
- The following three RTC clock sources can be selected:
  - HSE clock division (HSE/128, HSE/32 or HSE/8)
  - LSI oscillator clock
  - LSE oscillator clock
- 2 independent reset types:
  - The APB1 interface is reset by the system;
  - RTC cores (prescalers, alarms, counters, and dividers) can only be reset by backup domains
- 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 turns back to 0 state

16.3. RTC functional description

16.3.1. Register reset

The RTC_PRL, RTC_ALR, RTC_CNT and RTC_DIV and BKP_RTCCR registers can only be reset by a backup domain reset signal (power-on reset or RCC _BDCR.BDRST). Other system registers

(CRH, CRL) are reset asynchronously by system reset or RCC_APBRSTR1. RTCAPBRSTor power reset.

16.3.2. Read RTC register

The RTC core is completely independent of the RTC APB1 interface.

The software accesses the prescalation values, counter values and alarm clock values of RTC through APB1 interface. However, the associated readable register is updated only when the signal after synchronization with the rising edge of the RTC clock to the RTC APB1 clock is valid. The same goes for the RTC logo.

This means that if the APB1 interface was once closed and the read operation was just after APB1 was re-opened, the RTC register value read from APB1 may have been corrupted (typically read to 0) before the first internal register update. This can happen in the following situations:

System reset or power reset occurs

The system just woke up from standby mode

The system just woke up from shutdown mode (in this case, the count value just won't update because the CPU is not working, the system clock stops, but the RTC counts normally, and the count value won't sync to the VDD area)

In all of the above cases, the RTC core remains running when the APB1 interface is disabled (reset, no clock, or power down).

Therefore, if the APB1 interface of the RTC was once in a disabled state when reading the RTC register, the software must first wait for the RSF bit (register synchronization flag) in the RTC_CRL register to be set '1' by the hardware.

Note: The APB1 interface of the RTC is not affected by low power modes such as WFI and WFE.

Description:

1. CPU-readable registers include RTC_CR, RTC_CNT, and RTC_DIV;
2. The RTC_CR register is the RTC_PCLK field, and the CPU can read a stable value at any time;
3. RTC_CNT and RTC_DIV are derived from the RTC_CLK domain. After RTC operation, the RTC_DIV register is updated at the rising edge of each RTC_CLK; The RTC_CNT and the flag bits originating from the RTC_CLK clock domain also use the same update signal as the RTC_DIV register, although the value of RTC_CNT does not change every time it is updated;
4. In the RTC_PCLK domain, the RSF is set when the pulse signal after the RTC_CLK is synchronized to the RTC_PCLK is valid;
5. RSF only controls the read timing of RTC_CNT and RTC_DIV (hardware does not control it)

16.3.3. Configure the RTC register

The CNF bit in the RTC_CRL register must be set so that RTC enters configuration mode before writing to the RTC_PRL, RTC_CNT, RTC_ALR, BKP_RTCCR registers.

In addition, the write operation to any register of RTC must be performed after the previous write operation is completed. It is possible to determine whether the RTC register is being updated by querying the RTOFF status bit in the RTC_CR register. The RTC register may be written only if the RTOFF status bit is' 1 '.

Configuration process:

1. Querying the RTOFF bit until the value of RTOFF becomes' 1 '; (indicates that the previous configuration has been completed)
2. Set the CNF value to 1 to enter the configuration mode; (At this time, the RTOFF bit is still 1, ensuring that RTOFF = 1 when the CPU writes to the register)
3. Performing a write operation to one or more RTC registers; (RTOFF is still 1 in this process, and the RTC_CLK field register is written to the buffer register in this step)
4. Clear the CNF flag bit and exit the configuration mode; (After the hardware detects that CNF is cleared, it starts to perform the register write operation in the previous step and starts to write to the registers in the RTC_CLK field; At the same time, RTOFF is cleared)
5. The RTOFF is queried until the RTOFF bit becomes' 1 'to confirm that the write operation has completed. (The buffer register is written to the RTC_CLK field after setting RTOFF)

Note: The write operation can only be performed when the CNF flag bit is cleared, and this process requires at least 3 RTC_CLK cycles. (3 RTC_CLK cannot restart the configuration after the CNF flag is cleared, otherwise a configuration error will occur (in this case, it is controlled by RTOFF = 0)

Notes:

1. In this process, RTOFF = 1 when the CPU writes to the register;
2. The write cycle of the CPU is from CNF = 1 to CNF = 0, and other registers are configured between these two operations;
3. First write CNF to 1 and then clear CNF. This operation clears RTOFF; RTOFF will not be cleared after only writing CNF = 0 or not clearing after writing CNF = 1;
4. First write the CNF to 1 and then clear the CNF. This operation starts the process of writing the buffer register to the RTC_CLK field register;
5. RTOFF is implemented in the RTC_PCLK domain;

16.3.4. RTC flag settings

The RTC Second Flag (SECF) is set before changing the RTC counter during each RTC core clock cycle.

In the last RTC clock cycle before the counter reaches 0x0000, the RTC overflow flag (OWF) is set.

The RTC_Alarm and the RTC alarm flag (ALRF) are set in the RTC clock cycle before the value of the counter reaches the value of the alarm register plus 1 (RTC_ALR+1).

Writes to the RTC alarm register (RTC_ALR) must be synchronized with the RTC second flag using one of the following procedures:

1. Use the RTC alarm interrupt and modify the RTC alarm register (RTC_ALR) and/or the RTC counter register (RTC_CNT) in the interrupt handler.
2. Wait for the SECF bit in the RTC control register to be set before changing the RTC alarm register (RTC_ALR) and/or the RTC counter register (RTC_CNT).

16.3.5. RTC timing

RTC seconds and alarm clock timing are shown in the following figure:

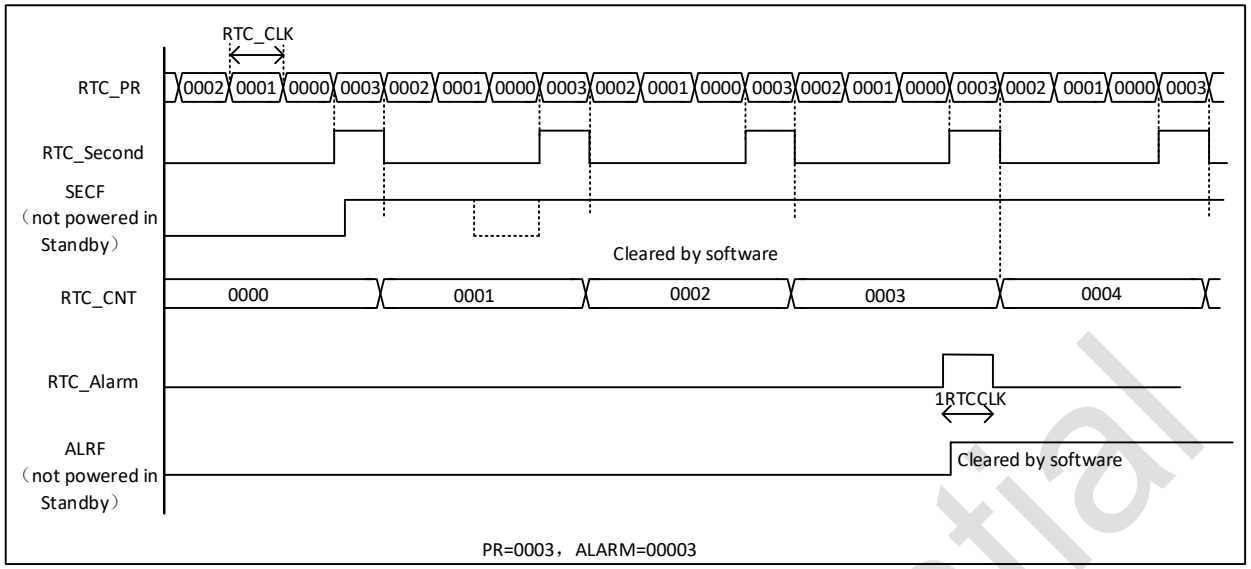


Figure 16-1 RTC seconds and alarm clock timing diagram

SECF and ALRF are RTC_PCLK domain signals, and are generated by sampling RTC_Sectond and RTC_Alarm signals in RTC_CLK domain, respectively.

The RTC overflow timing is shown in the following figure:

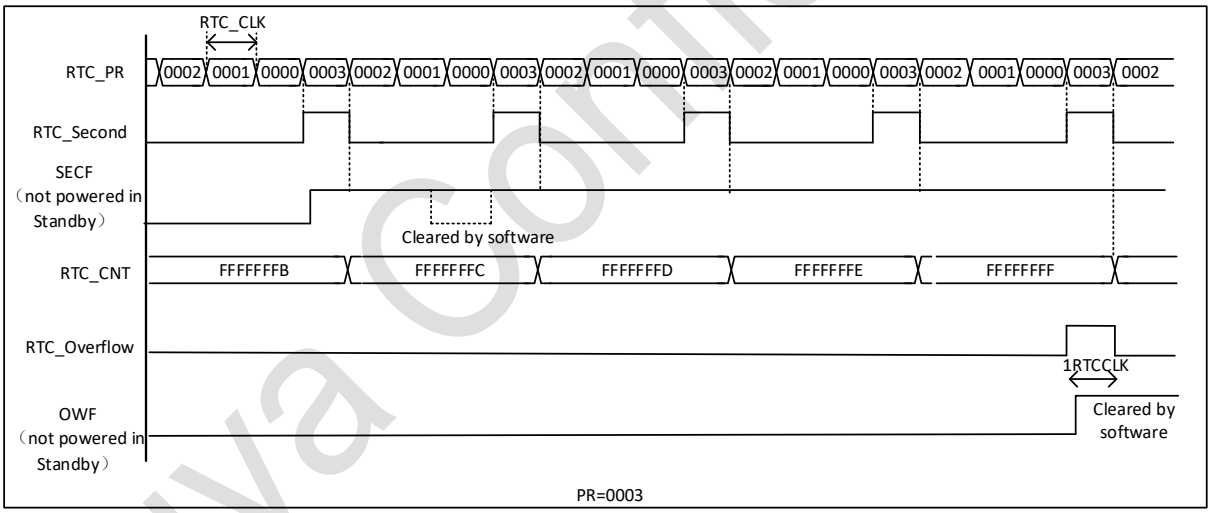


Figure 16-2 RTC overflow timing diagram

SECF and OWF are RTC_PCLK domain signals, which are generated by sampling RTC_CLK domain RTC_Sectond and RTC_Overflow numbers, respectively.

16.3.6. RTC calibration

For measurement purposes, the 64 division of the RTC clock can be output to the IO pin (PF5). This function is achieved by setting the CCO bit (RTCCR register).

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

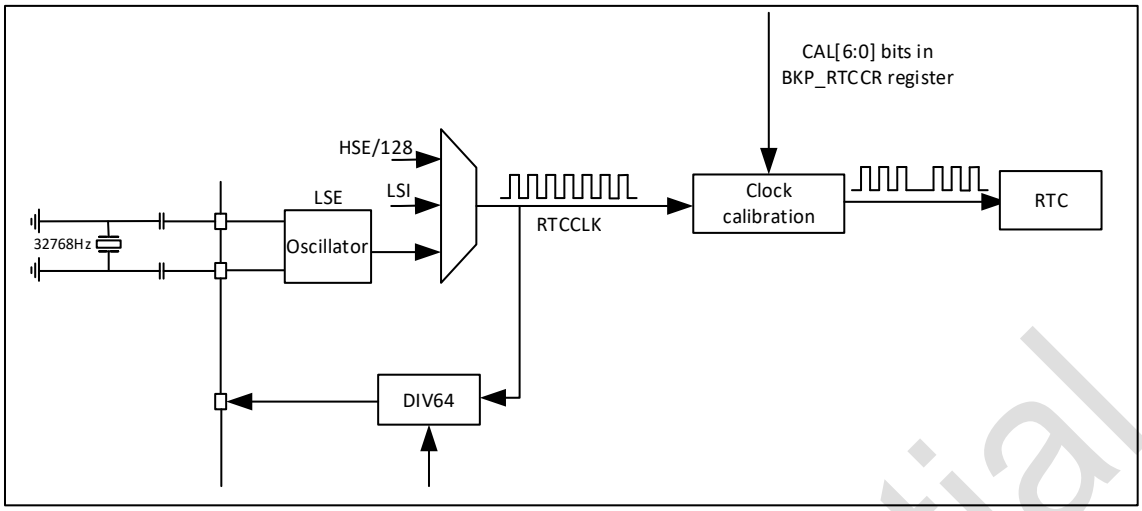


Figure 16-3 RTC calibration chart

16.4. RTC registers

16.4.1. RTC control register (RTC_CRH)

Address offset: 0x00
Reset value: 0x0000 0000

This register is reset by a system reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TK_OWIE	TK_AL-RIE	TK_SECIE	OWIE	ALR IE	SEC IE
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
5	TK_OWIE	RW	0	TK overflow interrupt allow bit 0: TK overflow interrupt is not allowed 1: Allow TK overflow interrupt
4	TK_ALRIE	RW	0	TK alarm interrupt allow bit 0: TK alarm interrupt is not allowed 1: Allow TK alarm interrupt
3	TK_SECIE	RW	0	TK second interrupt allowance bit 0: TK second interrupt is not allowed 1: Allow TK seconds interrupt
2	OWIE	RW	0	Overflow interrupt allowance bit 0: No overflow interrupt allowed 1: Allow overflow interrupt
1	ALRIE	RW	0	Alarm clock interrupt allowance bit 0: Alarm clock interrupt not allowed 1: Allow alarm interruption
0	SECIE	RW	0	Second interrupt allowance bit 0: No second interrupt allowed 1: Allow second interrupt

These bits are used to mask interrupt requests. Note: After reset, all interrupts are not enabled, so after initialization, make sure that there are no pending interrupt requests when writing to the RTC register. However, when the peripheral is completing the previous write operation (RTOFF = 0), it cannot write the RTC_CRH register.

This control register controls the function of the RTC. Some bits must use a dedicated configuration process to be written.

16.4.2. RTC control register (RTC_CRL)

Address offset: 0x04

Reset value: 0x0020 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	RTOFF	CNF	RSF	OWF	ALRF	SECF
										R	RW	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	RTOFF	R	1	RTC operation OFF, this bit is read-only. The RTC module utilizes this bit to indicate the status of the last operation performed on its registers (indicating whether the operation is completed or not). If this bit is' 0 ', it means that no write operation can be performed to any RTC register. 0: The last write to the RTC register is still in progress 1: The last write operation to RTC register has been completed
4	CNF	RW	0	Configuration flag This bit must be set '1' by the software to enter the configuration mode to allow new values to be written to the RTC_CNT, RTC_ALR or RTC_PRL registers. The write operation will only be performed when this bit is set to '1' and cleared again by '0' by the software. 0: Exit configuration mode (start updating RTC registers) 1: Enter configuration mode
3	RSF	RC_W0	0	Register sync flag. When the RTC_CNT register and the RTC_DIV register are updated, the hardware sets the bit '1' and the software clears the bit. After the APB1 is reset, or after the APB1 clock stops, this bit must be cleared '0' by the software. Before any read operation is to be performed, the user program must wait for the bit to be set '1' by the hardware to ensure that RTC_CNT, RTC_ALR, or RTC_PRL have been synchronized. 0: Register has not been synchronized 1: Registers have been synchronized
2	OWF	RC_W0	0	Overflow flag When the 32-bit programmable counter overflows, this bit is set '1' by the hardware. An interrupt is generated if OWIE = 1 in the RTC_CRH register. This bit can only be cleared by the software '0', and writing '1' is invalid. 0: no overflow; 1: 32-bit programmable counter overflow
1	ALRF	RC_W0	0	Alarm flag When the 32-bit programmable counter reaches a predetermined value set by the RTC_ALR register, this bit is set '1' by the hardware. An interrupt is generated if ALRIE = 1 in the RTC_CRH register. This bit can only be cleared by the software '0', and writing '1' is invalid. 0: No alarm; 1: There is an alarm clock.
0	SECF	RC_W0	0	Second flag When the 32-bit programmable prescaler overflows, this bit is set '1' by the hardware and the RTC counter is incremented by 1. Therefore, this flag provides a periodic signal (typically 1 second) to the resolution programmable RTC counter. An

Bit	Name	R/W	Reset Value	Function
				interrupt is generated if SECIE = 1 in the RTC_CRH register. This bit can only be cleared by software, write '1' is invalid. 0: Second flag condition does not hold 1: The second flag condition is established

The function of the RTC is controlled by the control register. When the peripheral is continuing the last write operation (RTOFF = 0), the RTC_CR register cannot be written.

Notes:

1. Any flag bit will remain suspended until the appropriate RTC_CR request bit is reset by the software, indicating that the requested interrupt has been accepted.
2. All interrupts are prohibited during reset, and there are no pending interrupt requests. Write operations can be performed to the RTC register.
3. When the APB1 clock is not running, the OWF, ALRF, SECF, and RSF bits are not updated.
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, an RTC global interrupt is allowed to be generated. If an EXTI line 17 interrupt is allowed to be generated in the EXTI controller, an RTC global interrupt and an RTC alarm interrupt are allowed to be generated.
6. If ALRF = 1, an RTC alarm interrupt is allowed if the interrupt mode of the EXTI line 17 is set in the EXTI controller; If the event mode of EXTI line 17 is set in the EXTI controller, a pulse is generated on this line (no RTC alarm interrupt is generated).

16.4.3. RTC prescaler load register high (RTC_PRLH)

The PRL register holds the count value of the RTC prescaler periodicity. This register is write-protected by the RTOFF bit of the RTC_CR register. Only when RTOFF = 1 is allowed to write the CPU.

Address offset: 0x08

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	Reserved
3: 0	PRL[19:16]	W	0	RTC prescaler reload value high (RTC prescaler reload value high) These bits define the clock frequency of the counter according to the following formula: FTR_CLK = fRTCCLK/(PRL [19: 0] +1) Note: 0 value is not recommended, otherwise RTC interrupt and flag bits cannot be generated correctly

16.4.4. RTC reload register low bit (RTC_PRLL)

Address offset: 0x0C

Reset value: 0x8000 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**
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[15:0]	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 RTC interrupt and flag bits cannot be generated correctly

16.4.5. RTC prescaler divider register high (RTC_DIVH)

At each TR_CLK cycle, the value of the RTC_PRL register is reloaded into the RTC prescalation counter. The user can obtain an accurate time measurement by reading the RTC_DIV register to obtain the current value of the prescalation counter without stopping the operation of the frequency division counter.

This register is read-only, and when there is any change in the value of the RTC_PRL or RTC_CNT register, the register value will be reloaded by the hardware.

Address offset: 0x10

Reset value: 0x0000 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	RTC_DIV [19:16]			
												R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	Reserved
3: 0	RTC_DIV [19:16]	R	0	RTC clock divider remainder high bit.

16.4.6. RTC prescaler divider register low (RTC_DIVL)

Address offset: 0x14

Reset value: 0x8000 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[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15: 0	DIV[15:0]	R	0x8000	RTC clock divider

16.4.7. RTC counter register high (RTC_CNTH)

The RTC module has a 32-bit programmable counter, which is accessed through two 16-bit registers. The count is based on the TR_CLK time reference generated by the prescaler.

The RTC_CNT register holds the count value of this counter. The register is write-protected and can only be written if RTOFF = 1. Write to the upper 16-bit RTC_CNTH or lower 16-bit RTC_CNTL register, load it directly into the corresponding programmable counter, and reload the RTC frequency division. When a read operation occurs, the current value of the counter (system time) is returned.

Address offset: 0x18

Reset value: 0x0000 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
RTC_CNT [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15: 0	RTC_CNT [31: 16]	RW	0x0000	The upper 16 bits of the RTC core counter When the RTC_CNTH register is read, the upper 16 bits of the current value of the RTC counter register are returned. Write operations to this register can only be performed when you enter configuration mode.

16.4.8. RTC counter register low (RTC_CNTL)

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
RTC_CNT [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15: 0	RTC_CNT [15: 0]	RW	0x0000	RTC kernel counter 16 bits lower When the RTC_CNTL register is read, the lower 16 bits of the current value of the RTC counter register are returned. Write operations to this register can only be performed when you enter configuration mode.

16.4.9. RTC alarm register high (RTC_ALRH)

When the programmable counter (count) reaches the 32-bit value stored in the RTC_ALR register, an alarm interrupt request is generated. This register is write-protected by the RTOFF bit, and write access is only allowed if RTOFF = 1.

Address offset: 0x20

Reset value: 0xFFFF 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
RTC_ALR [31: 16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15: 0	ALR[31:16]	RW	0xFFFF	RTC alarm Higher 16 bits The software can write the upper 16 bits of the Alarm time. Writing to this register must enter configuration mode.

16.4.10. RTC alarm register high (RTC_ALRL)

Address offset: 0x24

Reset value: 0xFFFF 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**
RTC_ALR [15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved	-	-	Reserved
15: 0	ALR[15:0]	RW	0xFFFF	RTC alarm Lower 16 bits The software can write the lower 16 bits of the Alarm time. Writing to this register must enter configuration mode.

16.4.11. RTC clock calibration and output configuration register (BKP_RTCCR)

Address offset: 0x2C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	Reserved
9	ASOS	RW	0	Second/alarm pulse output select bit When that ASOE bit is set, the ASOS bit can be used to select whether the output on the Pin is an RTC second pulse signal or an Alarm pulse signal 0: RTC alarm pulse signal 1: RTC second pulse signal
8	ASOE	RW	0	Second/alarm pulse output enable bit When this bit is set, the ASOS bit determines whether an RTC second pulse signal or an Alarm pulse signal is output on the pin.
7	CCO	RW	0	Calibration clock output When the ASOE is not set, the division of 64 of the CCO output RTC clock can be set 0: No action 1: When this bit is set, the 64-division of 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} clocks, which allows the RTC to calibrate to slow down the clock in steps of 1000000/220 PPM. The RTC clock can be slowed down from 0 to 121 PPM.

Writing the RTCCR register is a read and write of its cache register. The value of the cache register needs to be loaded into the actually functional register through the cnf-related configuration process to affect the corresponding result.

17. Independent watchdog (IWDG)

17.1. Introduction

Independent watchdog (IWDG) is integrated in the device, and this module has the characteristics of high-security level, accurate timing and flexible use. IWDG can be used to detect and resolve faults caused by software errors and trigger a system reset when the counter reaches a specified TIMEOUT value (TIMEOUT).

The independent watchdog (IWDG) is clocked by its own dedicated low-speed clock (LSI) and thus stays active even if the main clock fails.

The IWDG is best suited for applications that require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.

17.2. IWDG main features

- 12-bit down counter
- The clock is provided by an independent RC oscillator (can operate in Sleep/Stop mode)
- Support CPU configuration After the IWDG module is started, the RCC module automatically enables LSI as the IWDG clock.
- When the watchdog is activated, a reset occurs when the counter counts to 0x000

17.3. IWDG functional description

17.3.1. IWDG block diagram

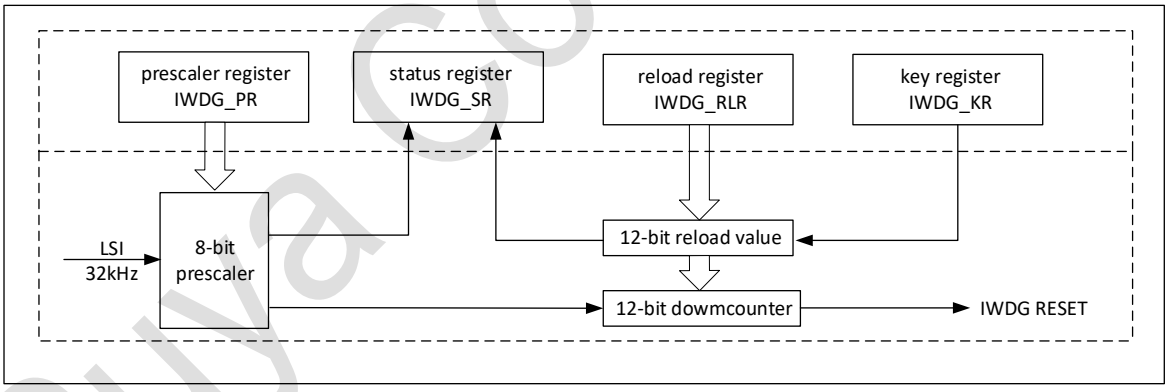


Figure 17-1 IWDG block diagram

Note: The watchdog function is in the VDD power supply zone, that is, it can still work normally in shutdown and standby modes.

Write 0xCCCC in the key register (IWDG_KR) to start enabling the independent watchdog; At this point the counter starts to count down from its reset value of 0xFFF. When the counter counts to the end of 0x000, a reset signal (IWDG_RESET) is generated.

Whenever0xAAAA is written in the key register IWDG_KR, the value in IWDG_RLR is reloaded into the counter to avoid an IWDG reset.

Table 17-1 Input clock (LSI) for IWDG timeout 32.768 kHz

Prescaler	PR[2:0]	Min	Max	Unit
/4	0	0.122	499.712	ms
/8	1	0.244	999.424	
/16	2	0.488	1998.848	
/32	3	0.976	3997.696	
/64	4	1.952	7995.392	
/128	5	3.904	15990.784	
/256	6 or 7	7.808	31981.568	

Note: These times are given as per the 32.768 kHz clock. Actually, the RC frequency inside the MCU will vary between 30 kHz and 60kHz. Furthermore, even if the frequency of the RC oscillator is accurate, the exact timing still depends on the phase difference between the APB interface clock and the RC oscillator clock, so there will always be a complete RC cycle that is uncertain. A relatively accurate watchdog timeout can be obtained by calibrating the LSI.

17.3.2. 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 IWDG key register (IWDG_KR) is written by the software before the counter reaches end of count.

17.3.3. Register access protection

Write access to the registers IWDG_PR, IWDG_RLR is protected. In order to modify them, the user must first write 0x0000 5555 to IWDG_KR. Writing other numbers to these registers will destroy the timing, such as writing 0x0000AAAA load, and the registers will be protected again.

If the values of IWDG_PR and IWDG_RLR are being updated, the status register will be reflected.

17.3.4. Debug mode and Stop mode

This function only exists when the system supports DBG_MCU. When the device enters Debug mode, the IWDG counter either continues to work normally or stops, depending on the configuration of the DBG_IWDG_STOP bit in DBG module.

The Stop mode means that there is an IWDG_Stop bit in the option byte, which can control the CPU when entering DEEPSLEEP mode. Whether IWDG continues counting normally or pauses counting depends on the configuration of IWDG_STOP in the option byte in Flash.

The configuration of IWDG_STOP in the Option byte is as follows:

Set the running state of IWDG timer in Stop mode

0: Freeze timer

1: Normal operation

The default IWDG_STOP in option byte is '1'

17.4. IWDG registers

17.4.1. IWDG key register (IWDG_KR)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	KEY[15:0]	W	0x00	Key value These bits must be written by software at regular intervals with the key value 0xAAAA, otherwise the watchdog generates a reset when the counter reaches 0. Writing the key value 0x5555 to enable access to the IWDG_PR and IWDG_RLR registers; 0xCCCC: indicates that IWDG is started (not restricted by this command word if hardware watchdog is selected).

17.4.2. IWDG prescaler register (IWDG_PR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:0	PR[2:0]	RW	0	Prescaler divider They are written by software to select the prescaler divider feeding the counter clock. PVU bit of the IWDG status register (IWDG_SR) must be reset in order to be able to change the prescaler divider. 000: 4 division 001: 8 division 010: 16 division 011: 32 division 100: 64 division 101: 128 division 110: 256 division 111: 256 division

17.4.3. IWDG reload register (IWDG_RLR)

Address offset: 0x08

Reset value: 0x0000 0FFF

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	RL[11:0]	RW	0	IWDG counter reload value They are written by software to define the value to be loaded in the watchdog counter each time the value0xAAAA is written in the IWDG key register (IWDG_KR). The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. The RVU bit in the IWDG status register (IWDG_SR) must be reset to be able to change the reload value.

				In addition, it should be noted that when the reload value is written to the RLR register, it needs to wait for three LSI clocks to read it out.

17.4.4. Status register (WWDG_SR)

Address offset: 0x0C

Reset value: 0x0000 0000

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

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 window 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: Wait for IWDG_PVU and IWDG_SR.RVU to be 0 before updating IWDG_PR and IWDG_SR.RLR, respectively. However, after updating the prescaler and/or the reload value, it is not necessary to wait until RVU or PVU is reset before continuing code execution.

18. Inter-integrated circuit (I^2C) interface

18.1. Introduction

The I^2C (inter-integrated circuit) bus interface handles communications between the microcontroller and the serial I^2C bus. It provides multimaster capability, and controls all I^2C bus-specific sequencing, protocol, arbitration and timing. It supports Standard-mode (Sm), Fast-mode (Fm) and Fast-mode plus (Fm+).

18.2. Main features

- Slave and master modes
- Multi-master function: can be used as a master or a slave
- Support different communication speeds
  - Standard mode (Sm): up to 100 kHz
  - Fast mode (Fm): up to 400 kHz
  - Fast mode plus (Fm +): Up to 1 MHz
- As Master
  - Clock generation
  - Start and Stop generation
- As Slave
  - Programmable I^2C address detection
  - Discovery of the Stop bit
- 7-bit addressing mode
- General call
- Status flag
  - Transmit/receive mode flags
  - Byte transfer complete flag
  - I^2C busy flag bit
- Error flag
  - Master arbitration loss
  - ACK failure after address/data transfer
  - Detection of misplaced start or stop condition
  - Overrun/Underrun (clock stretching function disable)
- Optional clock stretching
- Software reset
- Analog noise filter function
- Supports Stop wake-up when addresses match

— Supports configurable wake-up timeout time

— Support configurable timeout wake-up

18.3. I²C functional description

18.3.1. I²C block diagram

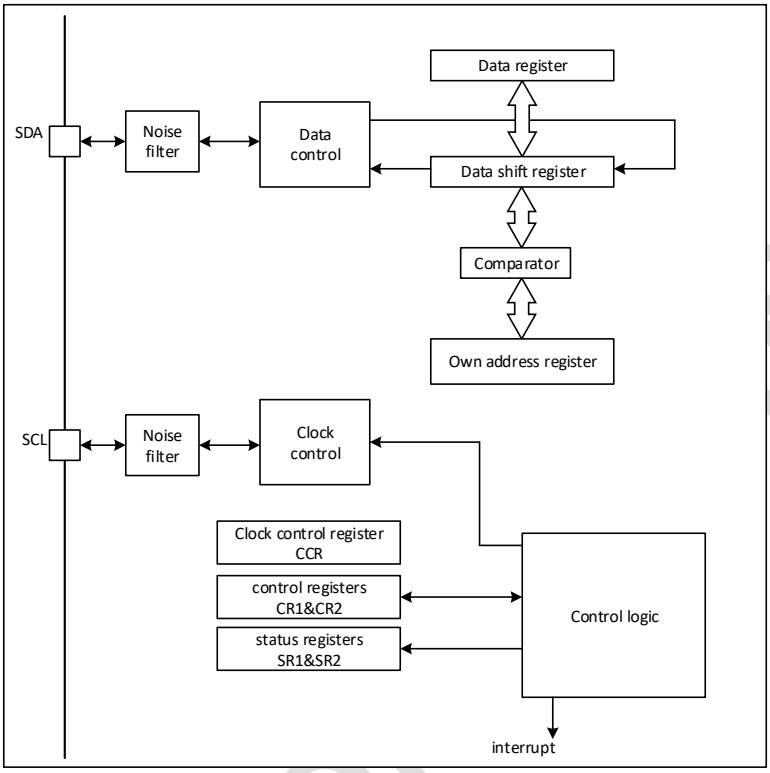


Figure 18-1 I²C block diagram

18.3.2. Mode selection

The interface can operate in one of the four following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

By default, it operates in slave mode. The interface automatically switches from slave to master, after it generates a START condition and from master to slave, if an arbitration loss or a Stop generation occurs, allowing multimaster capability.

18.3.2.1. 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-bit), and the General Call address. The General Call address detection may be enabled or disabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the start condition contain the address. 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 acknowledged bit to the transmitter. Refer to the figure below.

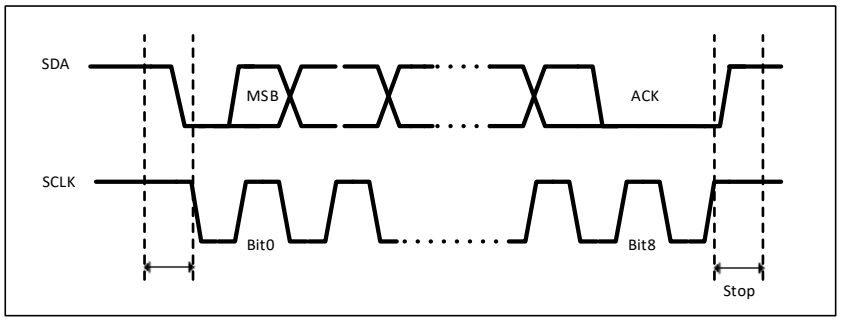


Figure 18-2 I²C bus protocol

Ack may be enabled or disabled by software. The I²C interface addresses (7-bit addressing and/or general call address) can be selected by software.

18.3.3. I²C initialization

18.3.3.1. Enable/turn off I²C module

The I²C peripheral clock must be configured and enabled by the I2C1EN bit in the RCC_APBENR1 register. Then the I²C can be enabled by setting the PE bit in the I2C_CR1 register.

18.3.3.2. I2C timings

The holding and setup time of data signal (SDA) is required to meet the I²C standard protocol, and the I²C timing needs to be set. This is achieved by writing to the I2C_CCR and I2C_TRISE registers.

18.3.4. I²C slave mode

By default, the I²C interface operates in Slave mode. To switch from default Slave mode to Master mode a Start condition generation is needed.

The peripheral input clock must be programmed in the I2C_CR2 register in order to generate correct timings. The peripheral input clock frequency must be at least:

- 2 MHz in Sm mode
- 4 MHz in Fm mode
- In fast mode plus: 16 MHz

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register. Then it is compared with the address of the interface (OAR1) or the General Call address (if ENGC = 1).

Address not matched:

The interface ignores it and waits for another Start condition.

Address matched:

The interface generates in sequence:

- An acknowledged pulse if the ACK bit is set
- The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set

The TRA bit indicates whether the slave is in Receiver or Transmitter mode.

18.3.4.1. Slave transmitter

Following the address reception and after clearing ADDR, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave stretches SCL low until ADDR is cleared and DR filled with the data to be sent (refer to EV1, EV3).

When the acknowledge pulse is received: The TxE bit is set by hardware with an interrupt if the ITEVFEN and the ITBUFEN bits are set.

If TxE is set and some data were not written in the I2C_DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read to I2C_SR1 followed by a write to the I2C_DR register, stretching SCL low.

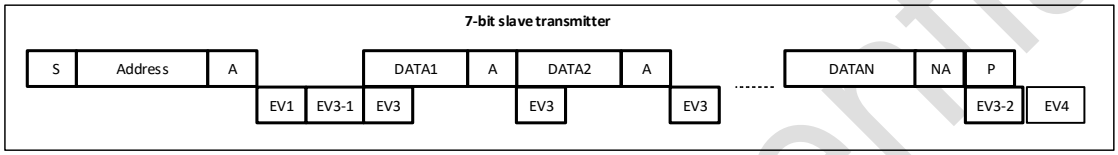


Figure 18-3 Transfer sequence diagram for slave transmitter

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, NA= Non-acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by reading SR2

EV3-1: TxE=1, shift register empty, data register empty, write Data1 in DR.

EV3: TxE=1, shift register not empty, data register empty, cleared by writing DR

EV3-2: AF=1; AF is cleared by writing '0' in AF bit of SR1 register.

EV4: STOPF=1, cleared by reading SR1 register followed by writing to the CR1 register.

Notes:

The 1.EV1 event stretches SCL low until the end of the corresponding software sequence.

The 2.EV3 software sequence must be completed before the end of the current byte transfer.

18.3.4.2. Slave receiver

Following the address reception and after clearing ADDR, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- An acknowledge pulse if the ACK bit is set
- The RxNE bit is set by hardware. An interrupt is generated if the ITEVTEN and ITBUFEN bit is set.

If RxNE is set and the data in the DR register is not read before the end of the next data reception, the BTF bit is set, and the interface waits until BTF is cleared by a read fromI2C_SR1 followed by a read from the I2C_DR register, stretching SCL low. (See figure below).

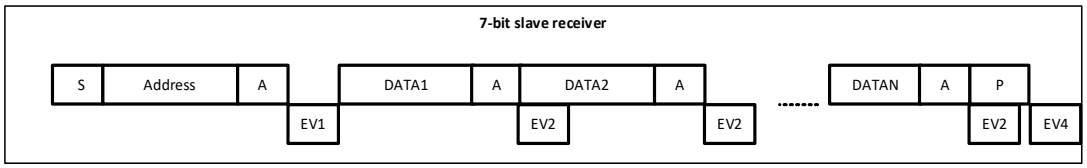


Figure 18-4 Transfer sequence diagram for slave receiver

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by reading SR2

EV2: RxNE=1 cleared by reading DR register

EV4: STOPF=1, cleared by reading SR1 register followed by writing to the CR1 register.

Notes:

1. The EV1 event stretches SCL low until the end of the corresponding software sequence.
2. The EV2 software sequence must be completed before the end of the current byte transfer.
3. After checking the SR1 register content, the user should perform the complete clearing sequence for each flag found set. Thus, for ADDR and STOPF flags, the following sequence is required inside the I²C interrupt routine:

If ADDR = 1, reading SR1 followed by reading SR2; if STOPF =1, reading SR1 followed by reading CR1.

The purpose is to make sure that both ADDR and STOPF flags are cleared if both are found set.

18.3.4.3. Closing slave communication

After the last data byte is transferred, a Stop Condition is generated by the master. The interface detects this condition and sets:

- The STOPF bit is set and generates an interrupt if the ITEVFEN bit is set.

The STOPF bit is cleared by a read of the SR1 register followed by a write to the CR1 register. (See EV4 in figure above)

18.3.4.4. Low power wake-up

When the device is in the stop state, the address can be received through I2C. If the addresses match, the chip can be awakened, otherwise, it can return to the previous state.

1) To implement the above functions, it is necessary to enable the PE and WUPEN of I2C_CR1 before entering stop, and enable the clock of I2C in RCC.

2) The Stop wake-up process also supports configurable timeout time, which can be configured by setting I2C_WT. If ITERREN is enabled, the chip will wake up after the timeout is generated. If ITERREN is turned off, the chip will wake up after the timeout is generated. Return to the previous state.

18.3.5. I²C master mode

In Master mode, the I²C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition.

Master mode is selected as soon as the Start condition is generated on the bus with a START bit.

- The following is the required sequence in master mode:
- Program the peripheral input clock in I2C_CR2 register in order to generate correct timings

- Configure the clock control registers
- Configure the rise time register
- Program the PE bit in I2C_CR1 register to enable the peripheral
- Set the START bit in the I2C_CR1 register to generate a Start condition

The peripheral input clock frequency must be at least:

- 2 MHz in Sm mode
- 4 MHz in Fm mode
- In fast mode plus: 16 MHz

18.3.5.1. SCL master clock generation

The CCR bits are used to generate the high and low level of the SCL clock, starting from the generation of the rising edge. As a slave may stretch the SCL line, the peripheral checks the SCL input from the bus at the end of the time programmed in TRISE register after rising edge generation.

If the SCL line is low, it means that a slave is stretching the bus, and the high level counter stops until the SCL line is detected high. This allows to guarantee the minimum HIGH period of the SCL clock parameter.

If the SCL line is high, the high level counter keeps on counting.

Indeed, the feedback loop from the SCL rising edge generation by the peripheral to the SCL rising edge detection by the peripheral takes time even if no slave stretches the clock. This loopback duration is linked to the SCL rising time, plus delay due to the noise filter present on the SCL input path, plus delay due to internal SCL input synchronization with APB clock. The maximum time used by the feedback loop is programmed in the TRISE bits, so that the SCL frequency remains stable whatever the SCL rising time.

18.3.5.2. Start condition

Setting the START bit causes the interface to generate a Start condition and to switch to Master mode (MSL bit set) when the BUSY bit is cleared.

Note: In master mode, setting the START bit causes the interface to generate a ReStart condition at the end of the current byte transfer.

Once the Start condition is sent:

- The SB bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register with the Slave address.

18.3.5.3. Slave address transmission

Then the slave address is sent to the SDA line via the internal shift register.

- In 7-bit addressing mode, one address byte is sent.

As soon as the address byte is sent,

  - The ADDR bit is set by hardware and an interrupt is generated if the ITEVFEN bit is set.

Then the master waits for a read of the SR1 register followed by a read of the SR2 register.

The master can decide to enter Transmitter or Receiver mode depending on the LSB of the slave address sent.

- In 7-bit addressing mode,
  - To enter Transmitter mode, a master sends the slave address with LSB reset.
  - To enter Receiver mode, a master sends the slave address with LSB set.

The TRA bit indicates whether the master is in Receiver or Transmitter mode.

18.3.5.4. Master transmitter

Following the address transmission and after clearing ADDR, the master sends bytes from the DR register to the SDA line via the internal shift register.

The host waits until the first data byte is written to the DR register.

When the acknowledge pulse is received, the TxE bit is set by hardware and an interrupt isgenerated if the ITEVFEN and ITBUFEN bits are set.

If TxE is set and some data were not written in the DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read from I2C_SR1 followed by a write to I2C_DR, stretching SCL low.

Closing slave communication

After the last byte is written to the DR register, the STOP bit is set by software to generate a Stop condition. The interface automatically goes back to slave mode (MSL bit cleared).

Note: When the TxE or BTF position is bit, the stop condition shall be scheduled for the occurrence of the EV8_2 event.

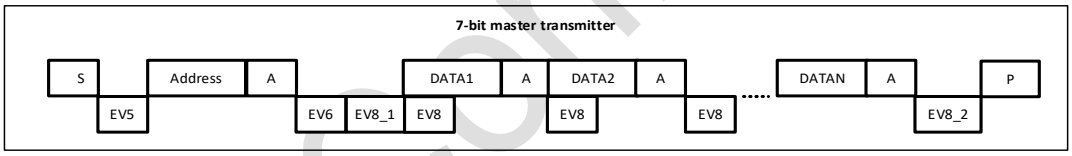


Figure 18-5 Transfer sequence diagram for master transmitter

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register with Address.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV8_1: TxE=1, shift register empty, data register empty, write Data1 in DR.

EV8: TxE=1, shift register not empty, data register empty, cleared by writing DR

EV8_2: TxE=1, BTF = 1, Program Stop request. TxE and BTF are cleared by hardware by the Stop condition

Notes:

1) The EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

2) The EV8 software sequence must complete before the end of the current byte transfer. In case EV8 software sequence can not be managed before the current byte end of transfer, it is recommended to use BTF instead of TXE with the drawback of slowing the communication.

18.3.5.5. Master receiver

Following the address transmission and after clearing ADDR, the I2C interface enters Master Receiver mode. In this mode the interface receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- An acknowledge pulse if the ACK bit is set
- The RxNE bit is set and an interrupt is generated if the ITEVFEN and ITBUFEN bits are set.

If the RxNE bit is set and the data in the DR register is not read before the end of the last data reception, the BTF bit is set by hardware and the interface waits until BTF is cleared by a read in the I2C_SR1 register followed by a read in the I2C_DR register, stretching SCL low.

Closing slave communication

Method 1: This method is for the case when the I2C is used with interrupts that have the highest priority in the application.

The master sends a NACK for the last byte received from the slave. After receiving this NACK, the slave releases the control of the SCL and SDA lines. Then the master can send a Stop/Restart condition.

1. To generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just after reading the second last data byte (after second last RxNE event).
2. To generate the Stop/Restart condition, software must set the STOP/START bit just after reading the second last data byte (after the second last RxNE event).
3. In case a single byte has to be received, the Acknowledge disable and the Stop condition generation are made just after EV6 (in EV6_1, just after ADDR is cleared).
4. After the Stop condition generation, the interface goes automatically back to slave mode (MSL bit cleared).

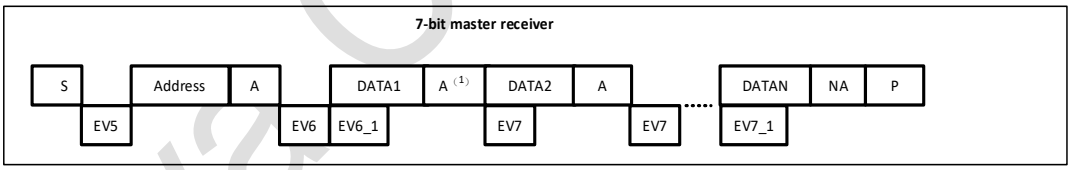


Figure 18- 6 Method 1: transfer sequence diagram for master receiver

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV6_1: no associated flag event, used for 1 byte reception only.

EV7: RxNE=1 cleared by reading DR register.

EV7_1: RxNE=1 cleared by reading DR register, program ACK=0 and STOP request

1. If a single byte is received, the place marked in the above figure will be NA
2. The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
3. The EV7 software sequence must complete before the end of the current byte transfer. In

case EV7 software sequence can not be managed before the current byte end of transfer. It is recommended to use BTF instead of RXNE with the drawback of slowing the communication.

4. The EV6_1 or EV7_1 software sequence must complete before the ACK pulse of the current byte transfer.

Method 2: This method is for the case when the I²C is used with interrupts that do not have the highest priority in the application or when the I²C is used with polling.

With this method, DataN_2 is not read, so that after DataN_1, the communication is stretched (both RxNE and BTF are set). Then, clear the ACK bit before reading DataN-2 in DR register to ensure it is cleared before the DataN Acknowledge pulse. After that, just after reading DataN_2, set the STOP/ START bit and read DataN_1. After RxNE is set, read DataN.

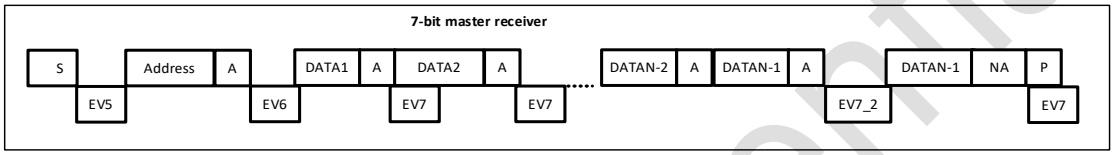


Figure 18-7 Method 2: transfer sequence diagram for master receiver when N>2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing the DR register.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV7: RxNE=1 cleared by reading DR register

EV7_2: BTF = 1, DataN-2 in DR register and DataN-1 in shift register, program ACK = 0, Read DataN-2 in DR. Program STOP = 1, read DataN-1.

Notes:

1. The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
2. The EV7 software sequence must complete before the end of the current byte transfer. In case EV7 software sequence can not be managed before the current byte end of transfer. It is recommended to use BTF instead of RXNE with the drawback of slowing the communication.

When 3 bytes remain to be read:

- RxNE = 1 => Nothing (DataN-2 not read).
- DataN-1 received
- BTF = 1 because both shift and data registers are full: DataN-2 in DR and DataN-1 in the shift register => SCL tied low: no other data will be received on the bus.
- Clear ACK bit
- Read DataN-2 in DR => This will launch the DataN reception in the shift register
- DataN received (with a NACK)
- Program START/STOP bit
- Read DataN-1

- RxNE=1
- Read DataN
- The procedure described above is valid for N>2. The cases where a single byte or two bytes are to be received should be handled differently, as described below:

Case of two bytes to be received:

- Set POS and ACK bit
- Wait for the ADDR flag to be set
- Clear ADDR bit
- Clear ACK bit
- Wait for BTF to be set
- Program STOP bit
- Read DR twice

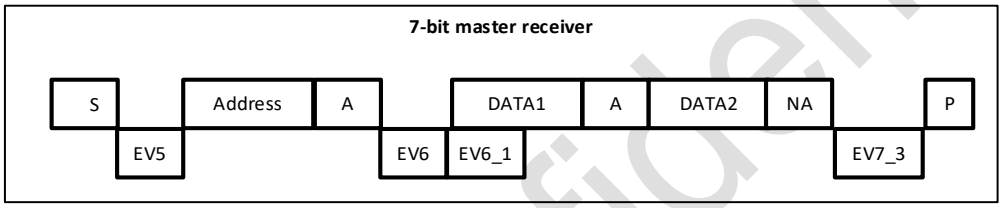


Figure 18-8 Method 2: transfer sequence diagram for master receiver when N=2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing the DR register.

EV6: ADDR=1, cleared by reading SR1 followed by reading SR2

EV6_1: No associated flag event. The acknowledge disable should be done just after EV6, that is after ADDR is cleared.

EV7_3: BTF = 1, program STOP = 1, read DR twice (Read Data1 and Data2) just after programming the STOP.

Notes:

1. The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
2. The EV6_1 software sequence must complete before the ACK pulse of the current byte transfer.

- Case of a single byte to be received:
  - In the ADDR event, clear the ACK bit
  - Clear ADDR
  - Program STOP/START bit
  - Read the data after the RxNE flag is set.

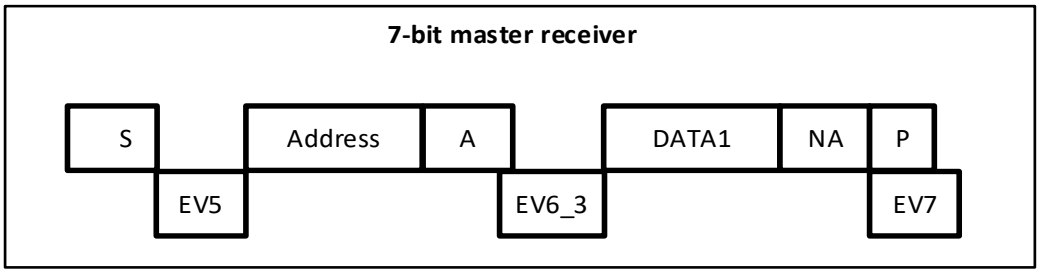


Figure 18-9 Method 2: transfer sequence diagram for master receiver when N=1

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing the DR register.

EV6_3: ADDR = 1, program ACK = 0. Clear ADDR by reading SR1 register followed by reading SR2 register. Program STOP =1 just after ADDR is cleared.

EV7: RxNE=1 cleared by reading DR register

Notes:

The EV5, EV6, EV8_1 and EV8_2 events stretch SCL low until the end of the corresponding software sequence.

18.3.6. Error conditions

18.3.6.1. Bus error (BERR)

This error occurs when the I²C interface detects an external Stop or Start condition during an address or a data transfer. In this case:

The BERR bit is set, and an interrupt is generated if the ITERREN bit is set;

In Slave mode: data are discarded, and the lines are released by hardware:

- In case of a misplaced Start, the slave considers it is a restart and waits for an address, or a Stop condition
- In case of a misplaced Stop, the slave behaves like for a Stop condition and the lines are released by hardware

In Master mode: the lines are not released and the state of the current transmission is not affected. It is up to the software to abort or not the current transmission.

18.3.6.2. Acknowledge failure (AF)

This error occurs when the interface detects a non-acknowledge bit. In this case:

The AF bit is set, and an interrupt is generated if the ITERREN bit is set

A transmitter which receives a NACK must reset the communication:

- If Slave: lines are released by hardware.
- If Master: a Stop or repeated Start condition must be generated by software

18.3.6.3. Arbitration lost (ARLO)

This error occurs when the I²C interface detects an arbitration lost condition. In this case:

The ARLO bit is set by hardware (and an interrupt is generated if the ITERREN bit is set)

The I2C Interface goes automatically back to slave mode (the MSL bit is cleared). When the I²C loses the arbitration, it is not able to acknowledge its slave address in the same transfer, but it can acknowledge it after a repeated Start from the winning master.

18.3.6.4. Overrun/underrun error (OVR)

An overrun error can occur in slave mode when clock stretching is disabled and the I²C interface is receiving data. The interface has received a byte (RxNE=1) and the data in DR register has not been read, before the next byte is received by the interface.

In this case:

In case of Overrun error, software should clear the RxNE bit and the transmitter should re-transmit the last received byte.

Underrun error can occur in slave mode when clock stretching is disabled and the I²C interface is transmitting data. The interface has not updated the DR with the next byte (TxE=1), before the clock comes for the next byte. In this case:

The same byte in the DR register will be sent again

The user should make sure that data received on the receiver side during an underrun error are discarded. The next bytes are written within the clock low time specified in the I²C bus standard.

For the first byte to be transmitted, the DR must be written after ADDR is cleared and before the first SCL rising edge. If not possible, the receiver must discard the first data.

18.3.7. SDA/SCL line control

If clock stretching is enabled:

- Transmitter mode: If TxE=1 and BTF=1: the interface holds the clock line low before transmission to wait for the microcontroller to read SR1 and then write the byte in the Data register (both DR and shift register are empty).
- Receiver mode: If RxNE=1 and BTF=1: the interface holds the clock line low after reception to wait for the microcontroller to read SR1 and then read the byte in the Data register (both DR and shift register are full).
- If clock stretching is disabled in Slave mode:
- Overrun Error in case of RxNE=1 and no read of DR has been done before the next byte is received. The last received byte is lost.
- Underrun Error in case TxE=1 and no write into DR register has been done before the next byte must be transmitted. The same byte will be sent again.
- Write Collision not managed.

18.4. I²C interrupts

Table 18-1 I²C interrupt requests

Interrupt event	Event flag	Enable control bit
Start bit sent (Master)	SB	ITEVTEN
Address sent (Master) or Address matched (Slave)	ADDR	
Stop received (Slave)	STOPF	
Data byte transfer finished	BTF	

Interrupt event	Event flag	Enable control bit
Receive buffer not empty	RxNE	ITEVTEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error (BERR)	BERR	ITERREN
Arbitration loss (Master)	ARLO	
Acknowledge failure	AF	
Overrun/Underrun	OVR	
Timeout error	TIMEOUT	

18.5. I²C registers

Registers can be accessed half-word or word.

18.5.1. 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
SWRST	Res.	Res.	Res.	POS	ACK	STOP	START	NOSTRETCH	ENGC	Res.	Res.	Res.	WUP.	Res.	PE
RW				RW	RW	RW	RW	RW	RW				RW		RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	SWRST	RW	0	Software reset When set, the I²C is under reset state. Before resetting this bit, make sure the I²C lines are released and the bus is free. 0: I²C Peripheral not under reset 1: I²C Peripheral under reset state Note: This bit can be used to reinitialize the peripheral after an error or a locked state. As an example, if the BUSY bit is set and remains locked due to a glitch on the bus, the SWRST bit can be used to exit from this state.
14:12	Reserved	-	-	Reserved
11	POS	RW	0	Acknowledge position (for data reception). This bit is set and cleared by software and cleared by hardware when PE=0. 0: ACK bit controls the (N)ACK of the current byte being received in the shift register. 1: ACK bit controls the (N)ACK of the next byte which will be received in the shift register. Note: The POS bit is used when the procedure for reception of 2 bytes is followed. It must be configured before data reception starts. To NACK the 2nd byte, the ACK bit must be cleared just after ADDR is cleared.
10	ACK	RW	0	Acknowledge enable This bit is set and cleared by software and cleared by hardware when PE=0. 0: No acknowledge returned 1: Acknowledge returned after a byte is received. (matched address or data)
9	STOP	RW	0	Stop generation. The bit is set and cleared by software, cleared by hardware when a Stop condition is detected, set by hardware when a timeout error is detected. In Master Mode: 0: No Stop generation. 1: Stop generation after the current byte transfer or after the current Start condition is sent. In slave mode: 0: No Stop generation. 1: Release the SCL and SDA lines after the current byte transfer.

Bit	Name	R/W	Reset Value	Function
8	START	RW	0	Start generation This bit is set and cleared by software and cleared by hardware when start is sent or PE=0. In Master mode: 0: No Start generation 1: Repeated start generation In Slave mode: 0: No Start generation 1: Start generation when the bus is free (automatically switching to Master mode by hardware)
7	NOSTRETCH	RW	0	Clock stretching disable (Slave mode) This bit is used to disable clock stretching in slave mode when ADDR or BTF flag is set, until it is reset by software. 0: Clock stretching enabled 1: Clock stretching disabled
6	ENGC	RW	0	General call enable 0: General call disabled. . Address 00h is NACKed. 1: General call enabled. Address 00h is ACKed.
5:3	Reserved	-	-	Reserved
2	WUPEN	RW	0	Slave mode wake up low power mode enable 0: Slave mode wake up low power mode enabled 1: Slave mode wake up low power mode disabled Note: If the Wakeup from Slave mode feature is not supported, this bit is reserved and forced by hardware to '0'.
1	Reserved	-	-	Reserved
0	PE	RW	0	I^2C enable 0: Disabled 1: Enabled Note: If this bit is reset while a communication is on going, the peripheral is disabled at the end of the current communication, when back to IDLE state. All bit resets due to PE=0 occur at the end of the communication. In master mode, this bit must not be reset before the end of the communication.

Note: When the STOP or START bit is set, the software must not perform any write access to I2C_CR1 before this bit is cleared by hardware. Otherwise, there is a risk of setting a second STOP or START request.

18.5.2. I2C control register 2 (I2C_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.	Res.	Res.	Res.	Res.	IT-BUFEN	ITEV-TEN	ITER-REN	Res.	FREQ[6:0]						
					RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	ITBUFEN	RW	0	Buffer interrupt enable 0: TxE = 1 or RxNE = 1 does not generate any interrupt. 1: TxE = 1 or RxNE = 1 generates event interrupt
9	ITEVTEN	RW	0	Event interrupt enable 0: Disabled 1: Event interrupt enabled

Bit	Name	R/W	Reset Value	Function
				This interrupt is generated when: 1. SB = 1 (Master mode) 2. ADDR = 1 (Master/Slave) 3. STOPF = 1 (Slave) 4. BTF = 1 with no TxE or RxNE event TxE event to 1 if ITBUFEN = 1 RxNE event to 1 if ITBUFEN = 1
8	ITERREN	RW	0	Error interrupt enabled. 0: Error interrupt disabled; 1: Error interrupt enabled; This interrupt is generated when: 1.BERR=1 2.ARLO=1 3.AF=1 4.OVR=1 5. TIMEOUT = 1;
7	Reserved	-	-	Reserved
6:0	FREQ	RW	0	I^2C clock frequency The FREQ bits must be configured with the APB clock frequency value (I2C peripheral connected to APB). The FREQ field is used by the peripheral to generate data setup and hold times compliant with the I2C specifications. The minimum allowable settable frequency is 4 MHz, and the maximum frequency is the highest APB clock frequency of the device. 0000000: Disabled 0000001: Disabled 0000010: Disabled 0000011: Disabled 0000100: 4 MHz ….. 0100100: 36 MHz ….. 0110000: 48 MHz Greater than 1001000: Disabled.

18.5.3. I2C own address register 1 (I2C_OAR1)

Address offset: 0x08

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.			ADD[7:1]							Res.
						RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:1	ADD[7:1]	RW	0	7 ~ 1 bits of the interface address.
0	Reserved	-	-	Reserved

18.5.4. I2C Data register (I2C_DR)

Address offset: 0x10

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	DR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
7:0	DR[7:0]	RW	0	The 8-bit data register, inside the device, actually two independent buffers share an address, which are used to store the received data (RX_DR) and place the data to be sent to the bus (TX_DR). **Transmitter mode:** Byte transmission starts automatically when a byte is written in the DR (TX_DR actually) register. A continuous transmit stream can be maintained if the next data to be transmitted is put in DR once the transmission is started (TxE=1). **Receiver mode:** Received byte is copied into DR (RX_DR actually) register (RxNE=1). A continuous transmit stream can be maintained if DR is read before the next data byte is received (RxNE=1). Notes: 1) 1. In slave mode, the address is not copied into DR. 2) Write collision is not managed (DR can be written if TxE=0). 3) If an ARLO event occurs on ACK pulse, the received byte is not copied into DR and so cannot be read.

18.5.5. I2C status register (I2C_SR1)

Address offset: 0x14

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TIMEOUT	Res.	Res	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	Res	BTF	ADDR	SB
	RC_W0			RC_W0	RC_W0	RC_W0	RC_W0	R	R		R		R	R	R

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	TIMEOUT	RC_W0	0	Timeout or Tlow error. 0: No timeout error; 1: The timeout time can be set by setting I2C_WT when waking up with low power consumption inI2C mode. Cleared by software writing 0, or by hardware when PE=0.
13:12	Reserved	-	-	Reserved
11	OVR	RC_W0	0	Overrun/Underrun 0: No overrun/underrun 1: Overrun or underrun Set by hardware in slave mode when NOSTRETCH=1 and: In reception when a new byte is received (including ACK pulse) and the DR register has not been read yet. New received byte is lost. In transmission when a new byte should be sent and the DR register has not been written yet. The same byte is sent twice. Cleared by software writing 0, or by hardware when PE=0. Note: If the DR write occurs very close to SCL rising edge, the sent data is unspecified and a hold timing error occurs.
10	AF	RC_W0	0	Acknowledge failure 0: No acknowledge failure 1: Acknowledge failure Set by hardware when no acknowledge is returned. Cleared by software writing 0, or by hardware when PE=0.
9	ARLO	RC_W0	0	Arbitration lost (master mode) 0: No Arbitration Lost detected 1: Arbitration Lost detected

Bit	Name	R/W	Reset Value	Function
				Set by hardware when the interface loses the arbitration of the bus to another master. Cleared by software writing 0, or by hardware when PE=0. After an ARLO event the interface switches back automatically to Slave mode (MSL=0).
8	BERR	RC_W0	0	Bus error 0: No misplaced Start or Stop condition 1: Misplaced Start or Stop condition Set by hardware when the interface detects wrong start or end conditions during a byte transfer. Cleared by software writing 0, or by hardware when PE=0.
7	TxE	R	0	Data register empty (transmitters) 0: Data register not empty 1: Data register empty Set when DR is empty in transmission. TxE is not set during address phase. Cleared by software writing to the DR register or by hardware after a start or a stop condition or when PE=0. TxE is not set if a NACK is received. Note: TxE is not cleared by writing the first data being transmitted, or by writing data when BTF is set, as in both cases the data register is still empty.
6	RxNE	R	0	Data register not empty (receivers) 0: Data register empty 1: Data register not empty Set when data register is not empty in receiver mode. RxNE is not set during address phase. Cleared by software reading or writing the DR register or by hardware when PE=0. Note: RxNE is not cleared by reading data when BTF is set, as the data register is still full.
5	Reserved	-	-	Reserved
4	STOPF	R	0	Stop detection (slave mode) 0: No Stop condition detected 1: Stop condition detected Set by hardware when a Stop condition is detected on the bus by the slave after an acknowledge (if ACK=1). Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0.
3	Reserved	-	-	Reserved
2	BTF	R	0	Byte transfer complete flag 0: Data byte transfer not done 1: Byte transfer transmission ended successfully Set by hardware when NOSTRETCH=0 and: In reception when a new byte is received (including ACK pulse) and DR has not been read yet (RxNE=1). In transmission when a new byte should be sent and DR has not been written yet (TxE=1). Cleared by software reading I2C_SR1 followed by either a read or write in the DR register or by hardware after a start or a stop condition in transmission or when PE=0. Notes: The BTF bit is not set after a NACK reception.
1	ADDR	R	0	Address sent (master mode)/matched (slave mode) Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0. **Address match (slave):**

Bit	Name	R/W	Reset Value	Function
				0: Address mismatched or not received. 1: Received address matched. Set by hardware as soon as the received slave address matched with the OAR registers content or a general call. Note: In slave mode, it is recommended to perform the complete clearing sequence after ADDR is set. **Address sent (master):** 0: No end of address transmission 1: End of address transmission For 7-bit addressing, the bit is set after the ACK of the byte. Note: ADDR is not set after a NACK reception.
0	SB	R	0	Start bit (Master mode) 0: No Start condition 1: Start condition generated. —Set when a Start condition generated. —Cleared by software reading the I2C_SR1 register followed by a write in the I2C_CR1 register, or by hardware when PE=0.

18.5.6. I2C status register 2 (I2C_SR2)

Address offset: 0x18

Reset value: 0x0000 0000

Note: Reading I2C_SR2 after reading I2C_SR1 clears the ADDR flag, even if the ADDR flag was set after reading I2C_SR1. Consequently, I2C_SR2 must be read only when ADDR is found set in I2C_SR1 or when the STOPF bit is cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.											GENCALL	Res.	TRA	BUSY	MSL
											R		R	R	R

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	GENCALL	R	0	General call address (Slave mode) 0: No General Call; 1: General Call Address received when ENGC=1. Cleared by hardware after a Stop condition or repeated Start condition, or when PE=0.
3	Reserved	-	-	Reserved
2	TRA	R	0	Transmit/receive flags 0: Data bytes received 1: Data bytes transmitted This bit is set depending on the R/W bit of the address byte, at the end of total address phase. It is also cleared by hardware after detection of Stop condition (STOPF=1), repeated Start condition, loss of bus arbitration (ARLO=1), or when PE=0.
1	BUSY	R	0	Bus busy 0: No communication on the bus 1: Polar data communication on the bus Set by hardware on detection of SDA or SCL low Cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).
0	MSL	R	0	Master/slave 0: Slave Mode 1: Master Mode

Bit	Name	R/W	Reset Value	Function
				- Set by hardware as soon as the interface is in Master mode (SB=1). - Cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1), or by hardware when PE=0.

18.5.7. I2C Clock control register (I2C_CCR)

Address offset: 0x1C

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	F +	Res.	CCR[11:0]											
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15	F/S	RW	0	I^2C master mode selection 0: Sm mode I^2C 1: Fm mode I^2C
14	DUTY	RW	0	Fm mode duty cycle 0: In fast mode: tlow/high = 2 1: In fast mode: tlow/high = 16/9
13	F +	RW	0	I^2C F+ master mode selection 0: Standard mode or fast mode, which one to choose is determined by bit15; 1: Fast mode plus; Note: When configuring this register to 1, the value of bit15 is not concerned;
12	Reserved	-	-	Reserved
11:0	CCR[11:0]	RW	0	Clock control register in Fm/Sm mode (Master mode) Controls the SCL clock in master mode. ■ Standard mode: —t_{high}=CCR x t_{pclk} —tl_{ow} =CCR x t_{pclk} ■ Fast mode: —DUTY=0: t_{high}=CCR x t_{pclk} tl_{ow} =2 x CCR x t_{pclk} -DUTY = 1 (up to 400 kHz): $_{thigh}$ = 9 x CCR x tpclk tlow = 16 x CCR x tpclk ■ In fast boost mode: —DUTY=0: thigh = 3 * CCR x tpclk tlow = 5 x CCR x tpclk -DUTY = 1 (up to 1 MHz): $_{thigh}$ = 2 x CCR x tpclk tlow = 3 x CCR x tpclk Notes: ■ The minimum allowed value is 0x04, and the minimum allowed value in fast DUTY mode is 0x01 $_{thigh}$ = $_{tr\ (SCL)}$ + tw (SCLH) tlow = $_{tr\ (SCL)}$ + tw $_{(SCLL)}$ ■ These delays do not have filters ■ This register can only be configured if PE = 0; ■ The fCK should be an integer multiple of 10 MHz to correctly produce the 400 kHz fast time at which

Note: The combination of bit15 and bit13 extends the pattern as shown in the following table:

F +	F/S	Mode
0	0	Standard mode
0	1	Fast mode

1	0	Fast enhancement mode
1	1	Fast enhancement mode

18.5.8. I2C TRISE register (I2C_TRISE)

Address offset: 0x20

Reset value: 0x0082 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	THOLDDATA_SEL	THOLDDATA					TRISE[6:0]						
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	THOLDDATA_SEL	RW	0	Data retention time selection 0: Default hardware calculation data retention time 1: Configure data retention time through THLDDATA
11:7	THLDDATA	RW	1	Maximum data retention time in fast/standard/fast enhanced mode (transmission mode) These bits are the minimum holding time used to ensure the data holding time in the data transmission mode. For example, the maximum SDA drop time bit allowed in standard mode is 300ns. If the value in the I2C_CR2 register species FREQ [6: 0] is equal to 0x08 and t_{pclk}= 125ns, configured to 0x03 in TRISE (300ns/125ns = 2.4 +1) If the result is not an integer, the integer part is written to THLDDATA to ensure configuration.
6:0	TRISE	RW	0x2	Maximum rise time in Fm/Sm mode (Master mode) These bits should provide the maximum duration of the SCL feedback loop in master mode. The purpose is to keep a stable SCL frequency whatever the SCL rising edge duration. These bits must be programmed with the maximum SCL rise time given in the I^2C bus specification, incremented by 1. For instance: in Sm mode, the maximum allowed SCL rise time is 1000 ns. If the value in FREQ [6: 0] in the I2C_CR2 register is equal to 0x08 and t_{pclk}= 125 ns, configured to 0x09 in TRISE (1000 ns/125 ns = 8 + 1 = 9). If the result is not an integer, TRISE[5:0] must be programmed with the integer part, in order to respect the t_{HIGH} parameter. Note: TRISE[5:0] must be configured only when the I^2C is disabled (PE = 0).

18.5.9. I2C wakeup time register (I2C_WT)

Address offset: 0x24

Reset value: 0x0008

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	CNT		DIV	
												RW		RW	

Bit	Name	R/W	Reset Value	Function
15:4	Reserved	-	-	Reserved

3:2	CNT	RW	0x2	Number of counts of timeout 00: 2 01: 8 10: 32 11: 128 Note: timout time is calculated as (DIV/FREQ) * CNT us Example: DIV selects 11 as 1024, FREQ is 8 M, CNT selects 00 as 2, then the timeout time is (1024/8) * 2 = 256 us
1:0	DIV	RW	00	The division coefficient of PCLK, the divided clock is used to countthe timeout time when Stop wakes up 00: 128 01: 256 10: 512 11: 1024

19. Serial peripheral interface (SPI)

19.1. Introduction

The serial peripheral interface (SPI) protocol supports half-duplex, full-duplex and simplex synchronous, serial communication with external devices. The interface can be configured as master and in this case it provides the serial clock (SCK) to the external slave device. The interface is also capable of operating in multimaster configuration. It can be used for a variety of purposes, including two-wire simplex synchronous transmission using one bi-directional data line, and reliable communication using CRC verification.

19.2. Main features

- Master or Slave mode
- 3-wire full-duplex simultaneous transmission
- 2-wire half-duplex synchronous transmission (with bidirectional data line)
- 2-wire simplex synchronous transmission (no bidirectional data line)
- 8-bit or 16-bit transmission frame selection
- Support multi-master mode
- 8 master mode baud rate prescale factors (max 24 M)
- Slave mode frequency (max 24 M)
- Both Master and Slave modes can be managed by software or hardware NSS: dynamic change of Master/Slave operating mode
- Programmable clock polarity and phase
- Programmable data order, MSB first or LSB first
- Dedicated transmit and receive flags that can trigger interrupts
- SPI bus busy status flag
- Interrupt-causing Master mode faults or overloads
- 2 x 32-bit Rx and Tx FIFOs

19.3. SPI functional description

19.3.1. Overview

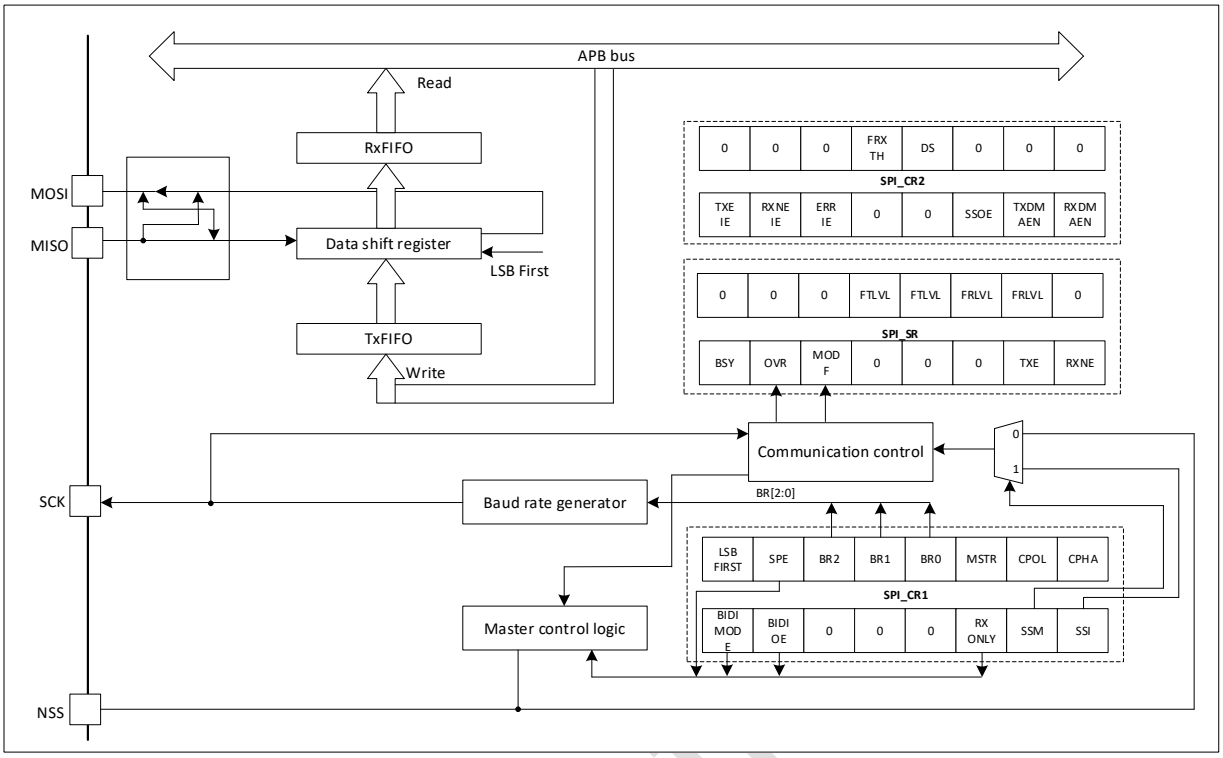


Figure 19- 1 SPI block diagram

Four I/O pins are dedicated to SPI communication with external devices.

MISO: Master In / Slave Out data. In the general case, this pin is used to transmit data in slave mode and receive data in master mode.

MOSI: Master Out / Slave In data. In the general case, this pin is used to transmit data in master mode and receive data in slave mode.

SCK: Serial Clock output pin for SPI masters and input pin for SPI slaves.

NSS: Slave select pin. Depending on the SPI and NSS settings, this pin can be used to either:

- Select an individual slave device for communication
- Synchronize the data frame
- Detect a conflict between multiple masters

The SPI bus allows the communication between one master device and one or more slave devices. The bus consists of at least two wires: one for the clock signal and the other for synchronous data transfer. Other signals can be added depending on the data exchange between SPI nodes and their slave select signal management.

19.3.2. Communications between one master and one slave

The SPI allows the MCU to communicate using different configurations, depending on the device targeted and the application requirements. These configurations use 2-wire, 3-wire (software NSS management), or 4-wire (hardware NSS management). Communication is usually initiated by the host computer.

19.3.2.1. Full-duplex communication

By default, the SPI is configured for full duplex communication. In this configuration, the shift registers of the master and slave are connected together using two unidirectional lines between the MOSI and the MISO. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master transmits the data to be sent to the slave via the MOSI line and receives data from the slave via the MISO line. When the data frame transmission is completed (all bits are shifted), the information between the master and slave is exchanged.

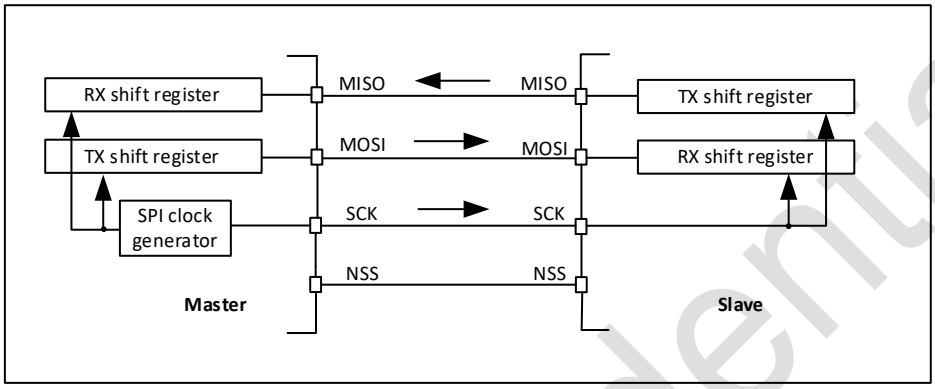


Figure 19- 2 Full-duplex single master/ single slave application

19.3.2.2. Half-duplex communication

The SPI can communicate in half-duplex mode by setting the BIDIMODE bit in the SPI_CR1 register. In this configuration, one single cross connection line is used to link the shift registers of the master and slave together. During this communication, the data is synchronously shifted between the shift registers on the SCK clock edge in the transfer direction selected reciprocally by both master and slave with the BDIOE bit in their SPI_CR1 registers. In this configuration, the MISO pin of master and the MOSI pin of slave are released as GPIO for other applications.

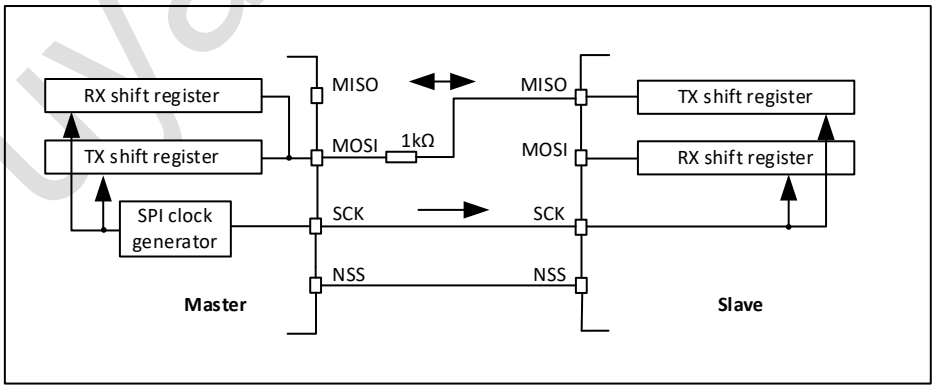


Figure 19-3 Half-duplex single master/ single slave application

The NSS pin may be used for hardware control flow between master and slave. Optionally, NSS may not be used. Then the flow has to be handled internally.

In this configuration, the MISO pin of master and the MOSI pin of slave can be used as GPIO.

19.3.2.3. Simplex communications

By using RXONLY (SPI_CR2 register), the SPI is set to transmit-only or receive-only, so that the SPI works in simplex mode. With this configuration, only 1 wire is used between the shift registers of master and slave. The remaining MISO and MOSI pins pair is not used for communication and can be used as standard GPIOs.

- Transmit-only mode (RXONLY=0): The configuration settings are the same as for full- duplex. The application has to ignore the information captured on the unused input pin. This pin can be used as a standard GPIO.
- Receive-only mode (RXONLY=1): The application can disable the SPI output functionby setting the RXONLY bit.
  - In the Slave mode configuration, the MISO output is disabled and this pin is used as the GPIO. When Slave select signal is active, Slave continues to receive data from the MOSI pin. Received data events appear depending on the data buffer configuration.
  - In the Master mode configuration, the MOSI output is disabled and the pin can be used as the GPIO. The clock signal is generated continuously as long as the SPI is enabled. The only way to stop the clock is to clear the RXONLY bit or the SPE bit and wait until the input pattern from the MISO pin is complete and the data buffer is filled.

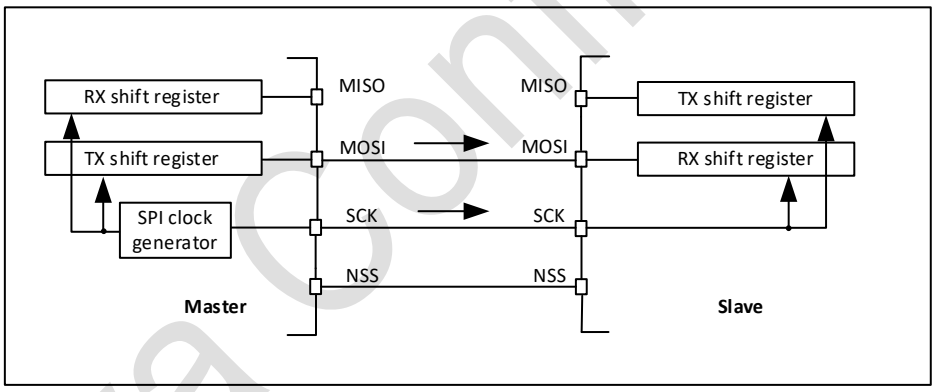


Figure 19-4 Single master/ single slave application

(master in transmit-only/slave in receive-only mode)

(1) The NSS pins can be used to provide a hardware control flow between master and slave. Optionally, NSS may not be used. Then the flow has to be handled internally.

(2) An accidental input information is captured at the input of transmitter Rx shift register. In the standard transmit-only mode, all events related to transmission reception must be ignored.

(3) In this configuration, both the MISO pins can be used as GPIOs.

Any simplex communication can be alternatively replaced by a variant of the half-duplex communication with a constant setting of the transaction direction (bidirectional mode is enabled while BIDIOE bit is not changed).

19.3.3. Multi-slave communication

In a configuration with two or more independent slaves, the master uses GPIO to manage NSS for each slave. The master must select one of the slaves individually by pulling low the GPIO connected to the slave NSS input. When this is done, a standard master and dedicated slave communication is established.

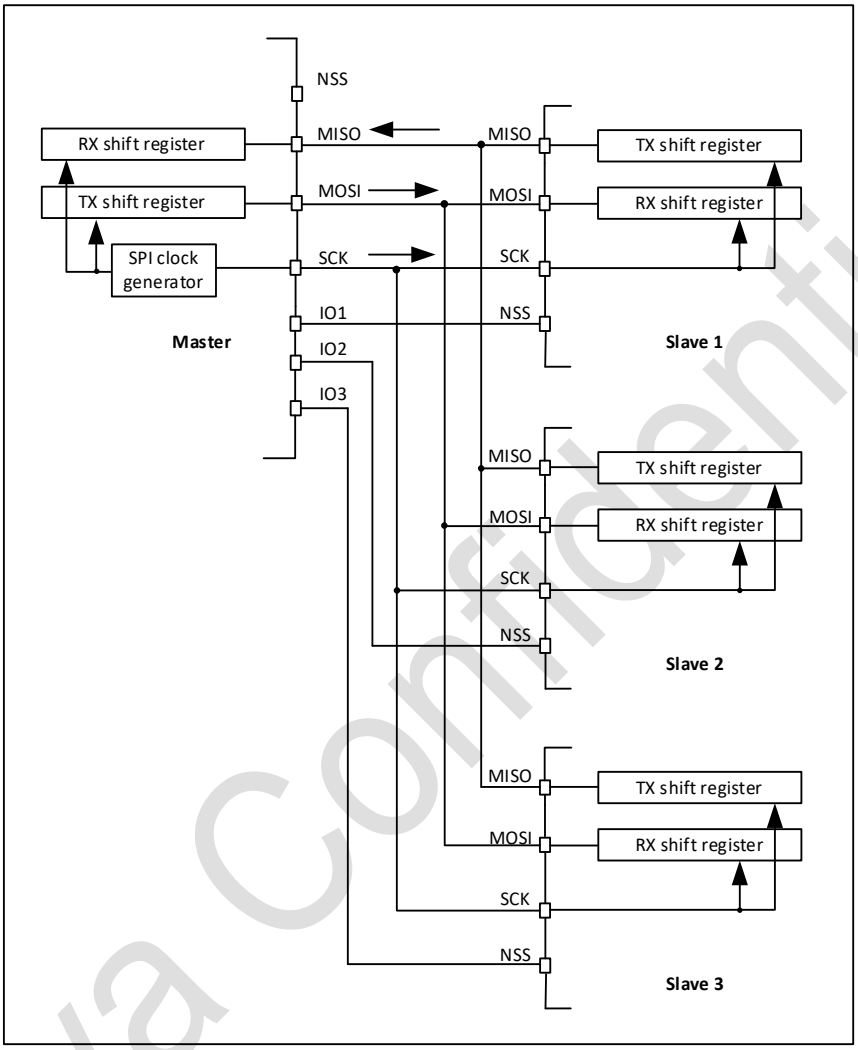


Figure 19-5 Master and three independent slaves

NSS pin is not used on master side at this configuration. It has to be managed internally (SSM=1, SSI=1) to prevent any MODF error.

As MISO pins of the slaves are connected together, all slaves must have the GPIO configuration of their MISO pin set as alternate function open-drain.

19.3.4. Multi-master communication

Unless SPI bus is not designed for a multi-master capability primarily, the user can use build in feature which detects a potential conflict between two nodes trying to master the bus at the same time. For this detection, NSS pin is used configured at hardware input mode.

The connection of more than two SPI nodes working at this mode is impossible as only one node can apply its output on a common data line at time.

When nodes are non active, both stay at slave mode by default. Once one node wants to overtake control on the bus, it switches itself into master mode and applies active level on the slave select input of the other node via dedicated GPIO pin. After the process is completed, the valid slave select signal is released, and the node controlling the bus temporarily returns to passive mode and waits for a new process to start.

If potentially both nodes raised their mastering request at the same time a bus conflict event appears (see mode fault MODF event). The user can then apply some simple arbitration procedure (e.g. postpone the next attempt by different timeouts defined in advance for the two nodes)

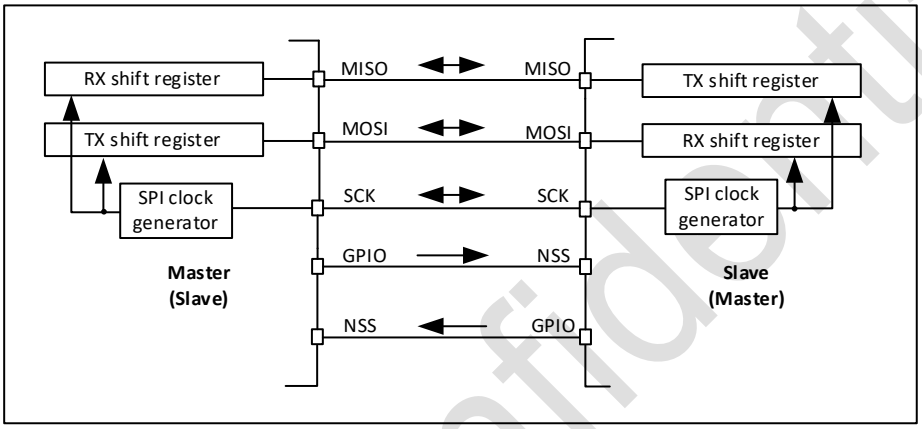


Figure 19-6 Multi-master application

The NSS is configured in a hardware input mode at both nodes. The active level enables MISO output control, while the passive node is configured as a slave.

19.3.5. Slave Select (NSS) pin management

In slave mode, the NSS works as a standard “chip select” input and lets the slave communicate with the master. In master mode, NSS can be used either as output or input. As an input it can prevent multimaster bus collision, and as an output it can drive a slave select signal of a single slave.

Hardware or software slave select management can be set using the SSM bit in the SPI_CR1 register:

- Software NSS management (SSM = 1): in this configuration, slave select information is driven internally by the SSI bit value in register SPI_CR1. The external NSS pin is free for other application uses.
- Hardware NSS management (SSM = 0): in this case, there are two possible configurations.
  - NSS output enabled (SSM = 0, SSOE = 1): This configuration is used only when the device operates in master mode. The NSS pin is managed by the hardware. The NSS signal is driven low as soon as the SPI is enabled in master mode (SPE=1), and is kept low until the SPI is disabled (SPE =0). The SPI cannot work in multimaster configuration with this NSS setting.

— 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.

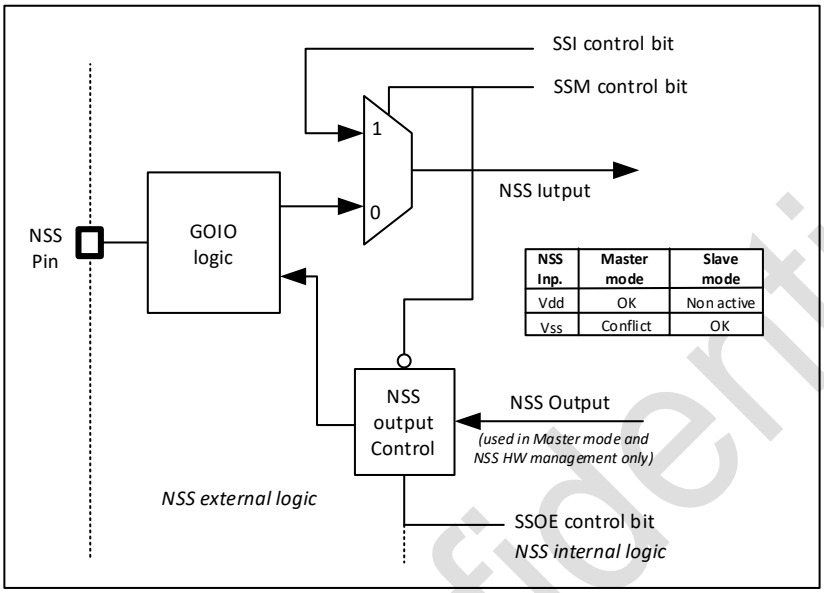


Figure 19-7 Hardware/software slave select management

19.3.6. Communication formats

During SPI communication, receive and transmit operations are performed simultaneously SCK (serial clock) synchronizes the information shift and sampling operation on the data line. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slaves devices must follow the same communication format.

19.3.6.1. Clock phase and polarity controls

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits in the SPI_CR1 register. CPOL (clock polarity) controls the IDLE state of the clock when there is no data transmission. This bit affects both master and slave modes. If CPOL is reset, the SCK pin has a low-level idle state. If CPOL is set, the SCK pin has a high-level idle state.

If the CPHA bit is set, the second edge on the SCK pin captures the first data bit transacted (falling edge if the CPOL bit is reset, rising edge if the CPOL bit is set). Data are latched on each occurrence of this clock transition type. If the CPHA bit is reset, the first edge on the SCK pin captures the first data bit transacted (falling edge if the CPOL bit is set, rising edge if the CPOL bit is reset). Data are latched on each occurrence of this clock transition type.

The combination of CPOL (clock polarity) and CPHA (clock phase) bits selects the data capture clock edge.

Prior to changing the CPOL/CPHA bits the SPI must be disabled by resetting the SPE bit.

The idle state of SCK must correspond to the polarity selected in the SPI_CR1 register.

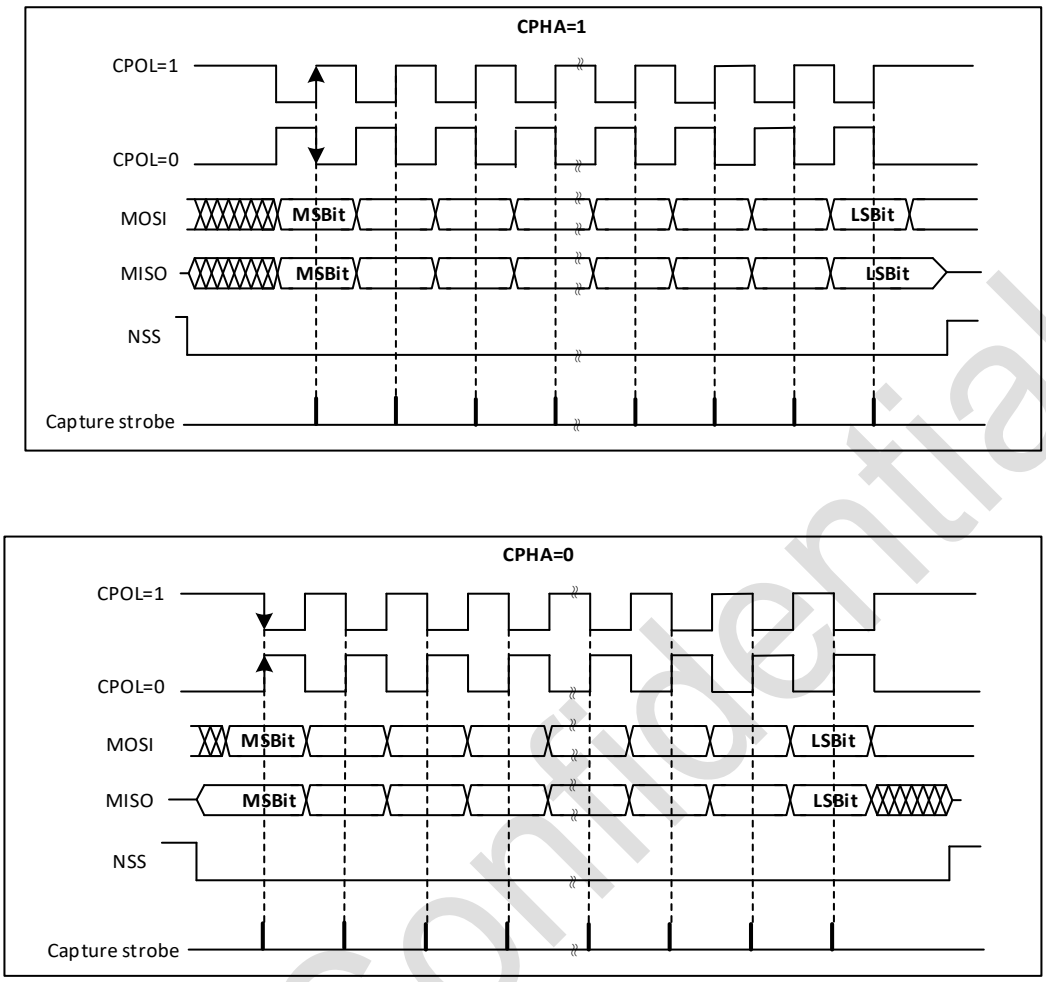


Figure 19-8 Data clock timing diagram

The order of data bits depends on LSBFIRST bit setting.

19.3.6.2. Data frame format

The SPI shift register can be set up to shift out MSB-first or LSB-first, depending on the value of the LSBFIRST bit in SPI_CR1 register. The data frame size is chosen by using the DFF bit in SPI_CR1 register. It can be set 8-bit or 16-bit length and the setting applies for both transmission and reception.

19.3.7. Configuration of SPI

The configuration procedure is almost the same for master and slave. For specific pattern establishment, follow the special chapter introduction. When a standard communication is to be initialized, perform these steps:

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:
    1) Configure the serial clock baud rate using the BR[2:0] bits (not required in slave mode)
    2) Configure the CPOL and CPHA bits
    3) Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE (RXONLY and BIDIMODE cannot be valid at the same time).

4) Configure the LSBFIRST bit
5) Configure DFF bits, select data frame bits
6) Configure SSM and SSI
7) Configure the MSTR bit (in multimaster NSS configuration, avoid conflict state on NSS if master is configured to prevent MODF error).

3. Write to SPI_CR2 register:
   1) Configure SSOE (not required for slave mode)

19.3.8. Procedure for enabling SPI

It is recommended to enable the SPI slave before the master sends the clock. If not, undesired data transmission might occur. The slave's data register must already contain the data to be sent before starting communication with the master (either at the first edge of the communication clock or, if the clock signal is continuous, before the end of the ongoing communication). The SCK signal must be fixed at the IDLE state level (corresponding selected polarity) before the SPI slave is enabled.

Full-duplex mode (or transmit-only), when SPI is enabled and TXFIFO is not empty, or the next write is made to TXFIFO, the master starts communication.

In any master receive only mode (RXONLY=1 or BIDIMODE=1 & BIDIOE=0), master starts to communicate, and the clock starts running immediately after SPI is enabled.

19.3.9. Data transmission and reception procedures

19.3.9.1. RXFIFO and TXFIFO

SPI All data traffic passes through a FIFO with a depth of 2 and a width of 16 bits (8 bits when the data frame is set to 8 bits). This enables the SPI to work in a continuous flow, and prevents overruns when the data frame size is short. Each direction has its own FIFO called TXFIFO and RXFIFO. These FIFOs are used in all SPI modes.

FIFO processing depends on a variety of parameters, including: data exchange mode (full duplex, half duplex), data frame format, size (8-bit or 16-bit) to access the FIFO data register.

A read access to the SPI_SR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPI_DR stores the written data in the TXFIFO at the end of a send queue. The read access must generally be aligned with the RXFIFO threshold, and the FTLVL and FRLVL bits show the current occupancy level of both FIFOs.

A read access to the SPI_DR register must be managed by the RXNE event. This event is triggered when data is stored in RXFIFO. When RXNE is cleared, RXFIFO is considered to be empty.

In a similar way, write access of a data frame to be transmitted is managed by the TXE event. This event is fired when TXFIFO Level is greater than 1. Otherwise, TXE is cleared and the TXFIFO is considered as full.

In this way, RXFIFO can store up to two data frames.

Both TXE and RXNE events can be polled or handled by interrupts.

If the next data is received when the RXFIFO is full, an overrun event occurs. An overrun event can be polled or handled by an interrupt.

The BSY bit being set indicates ongoing transaction of a current data frame. When the clock signal runs continuously, the BSY flag stays set between data frames at master. But becomes low for a minimum duration of one SPI clock at slave between each data frame transfer.

In some application scenarios, when writing data to TXFIFO, you can clear the TXFIFO data by setting the CLRTXFIFO bit, so as to write new data to TXFIFO again and communicate again.

19.3.9.2. Sequence handling

Some data frames can be passed through single sequence to complete a piece of information. When sending is enabled and there is any data in the TXFIFO of the maser, the sequence starts and continues. The clock signal is provided continuously by the master until TXFIFO becomes empty, then it stops waiting for additional data.

In receive-only mode, i.e. half-duplex (BIDIMODE = 1, BIDIOE = 0) or simplex mode (BIDIMODE = 0, RXONLY = 1), the master starts transmitting immediately when SPI is enabled and receive-only mode is activated. The master will always provide the clock until the master stops SPI or receive-only mode. Master receives data continuously.

While the master can provide all the transactions in continuous mode (SCK signal is continuous) it has to respect slave capability to handle data flow and its content at anytime. When necessary, the master must slow down the communication and provide either a slower clock or separate frames or data sessions with sufficient delays. It should be noted that there is no underflow error signal for master or slave, and the data from slave is usually interacted with and processed by master (even if slave cannot prepare the data in time).

Each sequence must be enclosed by an NSS pulse, and one of the slaves to communicate with is selected in a multi-slave system. In a single slave system, there is no need to use NSS to control the slave.

When the BSY bit is set it signifies an ongoing data frame transaction. When the dedicated frame transaction is finished, the RXNE flag is raised. The last bit is just sampled, and the complete data frame is stored in the RXFIFO.

19.3.9.3. Procedure for disabling the SPI

When SPI is disabled, a specific disabling process must be followed. It is important to do this before the system enters a low-power mode when the peripheral clock is stopped. Ongoing transactions can be corrupted in this case. In some modes the disable procedure is the only way to stop continuous communication running.

In full duplex or transmit-only mode, the master can complete the interaction when it stops providing the data to be sent. In this case, the clock stops after the last data transaction. Pay extra attention to the packing mode (when interacting with an odd number of data frames to place some dummy bytes interacting). In these modes, the user must use the standard disable process before the SPI can be disabled. When the SPI is disabled at the master transmitter while a frame transaction is ongoing or next data frame is stored in TXFIFO, the SPI behavior is not guaranteed.

When the master is in any receive only mode, the only way to stop the continuous clock is to disable the peripheral by SPE=0. Specific procedure must be followed when disabling SPI in this mode.

Data received but not read remains stored in RXFIFO when the SPI is disabled, and must be processed the next time the SPI is enabled, before starting a new sequence. To prevent unread data, make sure that the RXFIFO is empty when the SPI is disabled (using the correct disable process, or by resetting it with software to initialize all SPI registers).

Standard disable procedure is based on pulling BSY status together with FTLVL[1:0] to check if a transmission session is fully completed. This check can be done in specific cases, too, when it is necessary to identify the end of ongoing transactions, for example:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave. Or
- When transactions' streams from FIFO are completed while the last data frame is still ongoing in the peripheral bus.

The correct disable process is (except in receive-only mode):

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 SPI (SPE = 0)
4. Read data until FRLVL[1:0] = 00 (read all the received data)

The correct disable process for a specific receive-only mode is:

1. Interrupt the receive flow by disabling SPI (SPE=0) in the specific time window while the last data frame is ongoing.
2. Wait until BSY=0 (the last data frame is processed).
3. Read data until FRLVL[1:0] = 00 (read all the received data)

19.3.10. Status flag

The application can fully monitor the status of the SPI bus through 3 status flags.

19.3.10.1. 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 SPI_CR2 register is set. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

19.3.10.2. Rx buffer not empty flag (RXNE)

Depending on the value of the FRXTH bit (SPI_CR2), the RXNE flag bit is set:

- If FRXTH is 1, RXNE becomes high and remains high until RXFIFO level is greater than or equal to 1/4 (8-bit).
- If FRXTH is 0, RXNE becomes high and remains high until RXFIFO level is greater than or equal to 1/2 (16-bit).

The RXNE flag is set depending on the FRXTH bit value in the SPI_CR2 register.

The RXNE is cleared by hardware automatically when the above conditions are no longer true.

19.3.10.3. Busy flag (BSY)

The BSY flag is set and cleared by hardware (writing to this flag has no effect).

When it is set to '1', it indicates that the SPI is busy communicating with one exception: in the bidirectional reception mode of the main mode (MSTR = 1, BDM = 1, and BDOE = 0), the BSY flag remains low during reception.

The BSY flag can be used in certain modes to detect the end of a transfer so that the software can disable the SPI or its peripheral clock before entering a low-power mode which does not provide a clock for the peripheral. This avoids corrupting the last transfer.

The BSY flag is also useful for preventing write collisions in a multimaster system.

Except for the bidirectional reception mode of the master mode (MSTR = 1, BDM = 1, and BDOE = 0), the BSY flag is set to '1' when the transmission starts.

The BSY flag is cleared under any one of the following conditions:

- When the SPI is correctly disable
- When a fault is detected in Master mode (MODF bit set to 1)
- In Master mode, when it finishes a data transmission and no new data is ready to be sent
- In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: Do not use the BSY flag to handle each data send and receive. It is more appropriate to use TXE and RXNE.

19.3.11. Error flags

19.3.11.1. Mode fault (MODF)

Mode fault occurs when the master device has its internal NSS signal (NSS pin in NSS hardware mode) pulled low. Or when the SSI bit is set to '0' under software mode management of the NSS pin. This automatically sets the MODF bit. Master mode fault affects the SPI interface in the following ways:

- The MODF bit is set and an SPI interrupt is generated if the ERRIE bit is set.
- The SPE bit is cleared. This blocks all output from the device and disables the SPI interface.
- The MSTR bit is cleared, thus forcing the device into slave mode.

Use the following software sequence to clear the MODF bit:

1. Make a read or write access to the SPI_SR register while the MODF bit is set.
2. Then write to the SPI_CR1 register.

To avoid any multiple slave conflicts in a system comprising several MCUs, the NSS pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits can be restored to their original state after this clearing sequence.

As a security, hardware does not allow the SPE and MSTR bits to be set while the MODF bit is set. Normally, the MODF bit of the slave device cannot be set to '1'. However, in a multi-master configuration, a device may be in slave mode with the MODF bit set; At this point, the MODF bit indicates that a multi-master conflict may have occurred. The interrupt program may perform a reset or return to the default state to recover from the error state.

19.3.11.2. Overload mode

An overrun condition occurs when data is received by a master or slave and the RXFIFO has not enough space to store this received data. This can happen if the software did not have enough time to read the previously received data (stored in the RXFIFO).

When over normal operation occurs, the newly received data will not overwrite the data previously stored in RXFIFO. The newly received value is discarded, and all data transmitted subsequently is lost.

Clearing the OVR bit is done by a read access to the SPI_DR register followed by a read access to the SPI_SR register.

19.3.12. SPI interrupts

Table 19-1 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

19.3.13. SPI register

The peripheral registers have to be accessed by half-words (16 bits) or words (32 bits), while the DR register supports 32-bit, 16-bit and 8-bit access.

19.3.13.1. SPI control register 1 (SPI_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BI-DIMODE	BIDIOE	Res	Res	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW			RW	RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	Bidirectional data mode enable 0: 2-line unidirectional data mode selected 1: 1-line bidirectional data mode selected
14	BIDIOE	RW	0	Output enable in bidirectional mode This bit combined with the BIDIMODE bit selects the direction of transfer in bidirectional mode. 0: Output disabled (receive-only mode) 1: Output enabled (transmit-only mode) In master mode, the MOSI pin is used while the MISO pin is used in slave mode.
13:12	Reserved	-	-	Reserved
11	DFF	RW	0	Data frame format 0: 8-bit data frame format is selected for transmis-sion/reception 1: 16-bit data frame format is selected for trans-mission/reception Note: This bit should be written only when SPI is disabled (SPE = ‘0’) for correct operation.

Bit	Name	R/W	Reset Value	Function
10	RXONLY	RW	0	Receive only This bit combined with the BIDIMODE bit selects the direction of transfer in 2-line unidirectional mode. This bit is also useful in a multislave system in which this particular slave is not accessed, the output from the accessed slave is not corrupted. 0: Full-duplex (Transmit and receive) 1: Output disabled (receive-only mode)
9	SSM	RW	0	Software slave management When the SSM bit is set, the NSS pin input is replaced with the value from the SSI bit. 0: Software slave management disabled 1: Software slave management enabled
8	SSI	RW	0	Internal slave select This bit has an effect only when the SSM bit is set. The value of this bit is forced onto the NSS pin and the IO value of the NSS pin is ignored.
7	LSBFIRST	RW	0	Frame format 0: MSB transmitted first 1: LSB transmitted first Note: This bit should not be changed when communication is ongoing. Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur.
6	SPE	RW	0	SPI enable 0: Peripheral disabled 1: Peripheral enabled
5:3	BR[2:0]	RW	0	Baud rate control 000: $f_{PCLK}/2$ 001: $f_{PCLK}/4$ 010: $f_{PCLK}/8$ 011: $f_{PCLK}/16$ 100: $f_{PCLK}/32$ 101: $f_{PCLK}/64$ 110: $f_{PCLK}/128$ 111: $f_{PCLK}/256$ Note: The value of this bit cannot be changed when the communication is in progress. If the software modifies the communication while the communication is in progress, an error will occur.
2	MSTR	RW	0	Master selection 0: Slave configuration 1: Master configuration Note: This bit should not be changed when communication is ongoing.
1	CPOL	RW	0	Clock polarity 0: In idle state, SCK remains low 1: SCK maintains high level in idle state Note: This bit should not be changed when communication is ongoing.
0	CPHA	RW	0	Clock phase 0: Data sampling starts from the first clock edge 1: Data sampling starts from the second clock edge Note: This bit should not be changed when communication is ongoing.

19.3.13.2. SPI control register 2 (SPI_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	Res	Res	Res	Res	Res	Res	Res	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	Res	Res
								RW	RW	RW	RW		RW		

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	TXEIE	RW	0	Tx buffer empty interrupt enable 0: TXE interrupt masked 1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set.
6	RXNEIE	RW	0	Rx buffer not empty interrupt enable 0: RXNE interrupt masked 1: RXNE interrupt not masked. Used to generate an interrupt request when the RXNE flag is set.
5	ERRIE	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled When CRCERR, OVR, or MODF is 1, an interrupt request is generated.
4	CLRTXFIFO	RW	0	Clear TXFIFO Set by software and reset by hardware 0: No action 1: Clear TXFIFO Note: This bit should be written only when SPI is disabled (SPE = '0') for valid operation.
3	Reserved	-	-	Reserved
2	SSOE	RW	0	SS output enable 0: SS output is disabled in master mode and the SPI interface can work in multimaster configuration. 1: SS output is enabled in master mode and when the SPI interface is enabled. The SPI interface cannot work in a multimaster environment.
1:0	Reserved	-	-	Reserved

19.3.13.3. SPI status register (SPI_SR)

Address offset: 0x08

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	FTLVL	Res	FRLVL	Res	BSY	OVR	MODF	Res	UDR	Res	TXE	RXNE
				R		R		R	R	R		R		R	R

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	FTLVL	R	0	FIFO transmission level Set and cleared by hardware 0: FIFO empty 1: FIFO full (considered as FULL when the FIFO threshold is greater than 1/2)
10	Reserved	-	-	Reserved
9	FRLVL	R	0	FIFO reception level. Set and cleared by hardware 0: FIFO empty 1: FIFO full

Bit	Name	R/W	Reset Value	Function
8	Reserved	-	-	Reserved
7	BSY	R	0	Busy flag 0: SPI not busy 1: SPI is busy in communication or Tx buffer is not empty This bit is set or reset by hardware.
6	OVR	R	0	Overrun flag 0: No overflow error 1: Overrun occurred This flag is set by hardware and reset by a software sequence.
5	MODF	R	0	Mode fault 0: No mode fault occurred 1: Mode fault occurred This flag is set by hardware and reset by a software sequence.
4	Reserved	-	-	Reserved
3	UDR	R	0	Underflow flag bit. 0: No underrun occurred 1: An underflow error occurred. Hardware set, software sequence cleared.
2	Reserved	-	-	Reserved
1	TXE	R	1	Transmit buffer empty 0: Tx buffer not empty 1: Tx buffer empty
0	RXNE	R	0	Receive buffer not empty 0: Rx buffer not empty 1: Rx buffer empty

19.3.13.4. SPI data register (SPI_DR)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
15:0	DR[15:0]	RW	0	Data register. Data received or to be transmitted. The data register serves as an interface between the Rx and Tx FIFOs. When the data register is read, RxFIFO is accessed while the write to data register accesses TxFIFO. Note: Depending on the data frame format selection bit (DFF in SPI_CR1 register), the data sent or received is either 8-bit or 16-bit. For 8-bit data frames, the data registers send and receive based on the right-aligned 8-bit data. When in reception mode, the MSB of the register (SPI_DR[15:8]) is forced to 0. For a 16-bit data frame, the buffers are 16-bit and the entire register, SPI_DR[15:0] is used for transmission/reception.

20. Universal asynchronous receivers / transmitter (UART)

UART is a programmable universal asynchronous transceiver. The component is an AMBA 2.0 compliant advanced peripheral bus (APB) slave device.

20.1. UART main features

- AMBA APB interface
- Support 5/6/7/8/9-bit serial data
- Support 1/2 STOP bits (1/1.5 STOP bits for 5-bit data)
- Support sending address/data
- Support fixed parity check
- Support break frames
- Detect start bit errors
- Support programmable fractional baud rate: The serial data baud rate is programmable and derived from the formula: Baud Rate = Serial Clock Frequency / (16 × Divisor).
- Support SWAP function
- Support MSB FIRST endianness switching

20.2. Functional overview

20.2.1. UART (RS232) serial protocol

Because the serial communication between the UART and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the start and end. These bits can be used to synchronize the two devices. This serial data structure consisting of start bits and stop bits is called a character, as shown in the following figure.

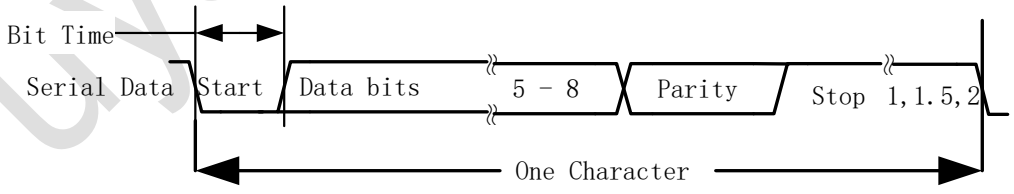


Figure 20-1 Serial data format

An additional parity bit can be added to the serial character. This bit occurs after the last data bit in the character structure and before the stop bit in order to provide the UART with the ability to perform simple error checking on the received data.

The UART control register (CR1 section in Registers) is used to control the serial character characteristics. The individual bits of the data word are transmitted after the start bit, starting with the least

significant bit (LSB). They are followed by an optional parity bit followed by a stop bit, which can be 1, 1.5, or 2.

Note: The STOP bit duration of UART implementations may be longer for the following reasons:

- Idle time inserted between some configured characters

All bits in the transmission are transmitted for exactly the same duration; The exception to this is the half stop bit when 1.5 stop bits are used. This duration is referred to as a bit period or bit time; One bit of time is equal to sixteen baud clocks.

To ensure line stability, once the start bit is detected, the receiver samples the serial input data at approximately the midpoint of the bit time. Because the exact number of baud clocks transmitted per bit is known, it is not difficult to calculate the midpoint of the sample; That is, every sixteen baud clocks after the midpoint sampling of the start bit.

Together with serial input dejittering, this sampling helps avoid the detection of erroneous start bits. Short faults are filtered out by de-dithering and no transitions are detected on the line. A falling edge is detected if the fault is wide enough that filtering can be avoided by de-dithering. However, that start bit is detect only when the line is sampled low again after half a sampling period

Figure 20-2 receiver serial data sampling points

As part of the 16550 standard, an optional baud clock reference output signal (baudout_n) provides timing information for receiving devices that need it. The baud rate of the UART is controlled by a serial clock sclk or pclk in a single-clock implementation and a frequency division latch register (BRR).

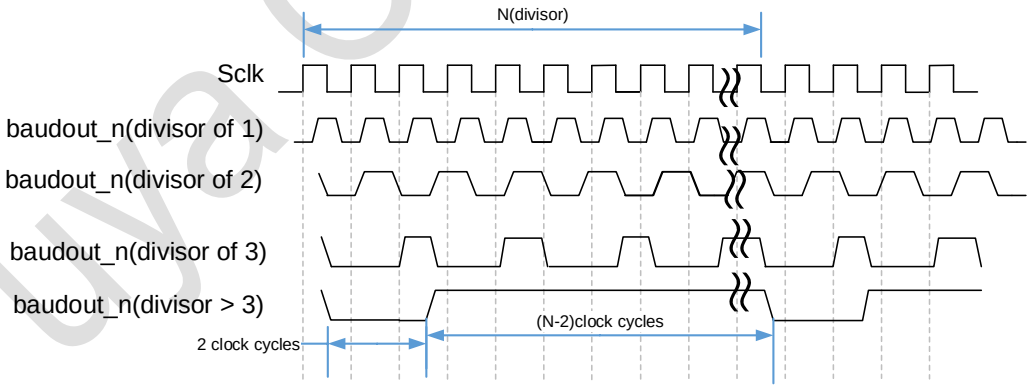


Figure 20-3 Timing diagram of baudout_n output with different divisor values.

20.2.2. 9-bit data transfer

The UART may be configured to have 9-bit data transmission in both transmit and receive modes. The 9th bit in the character occurs after the 8th bit of the character and before the parity bit. The figure below shows the serial transfer of characters, where D8 denotes the 9th bit, and also shows the general serial transfer in 9-bit mode. (This is the normal little-endian mode. If it is the big-endian mode, the order of 9 data bits will be switched)

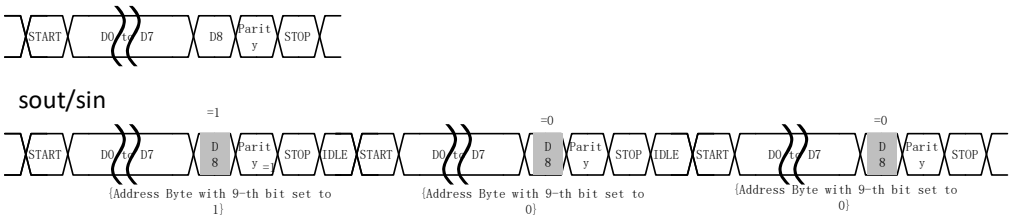


Figure 20-4 9-bit data transmission characteristics

By enabling 9-bit data transfer mode, UART can be used in multipoint systems where one master device is connected to multiple slave devices in the system. The master communicates with one of the slave. When the master device wants to transfer a block of data to a slave device, it first sends an address byte to identify the target slave device.

The distinction between address/data bytes is done based on the 9th bit in the input character. If the 9th bit is set to 0, the character represents a data byte. If the 9th bit is set to 1, the character represents the address byte. All slave systems compare the address bytes to their own addresses, and only the target slave system (where the addresses match) is able to receive data from the master system. The master starts sending data bytes to the target slave. The unaddressed slave system ignores incoming data until a new address byte is received.

Note an address followed by 2 data bytes. The address byte is output with the 9th bit (D8) set to 1, while the data byte is emitted with the 9th bit (D8) set to 0. The parity bit is an optional field.

The configuration of the UART for 9-bit data transmission does the following:

1. The CR3 [0] bit is used to enable or disable 9-bit data transfer.
2. In the case of reception, the CR1 [1] bit is used to select between hardware and software based address matching.
3. The CR3 [2] bit is used to enable the transmit address in the case of transmission.
4. The CR3 [3] bit is used to select between hardware and software based address transfer.
5. The TAR and RAR registers are used to send the address and match the received address, respectively.
6. The TDR and RDR registers are 9-bit registers for data transfer in 9-bit mode.
7. The SR [8] bit is used to indicate an address reception interrupt.

20.2.2.1. Send mode

UART supports two types of transmission modes:

- Transmission mode 0 (when (CR3 [3]) is set to 0)
- Transmission mode 1 (when (CR3 [3]) is set to 1)

20.2.2.2. Transmission mode 0

In transfer mode 0, addresses are programmed in the transfer address register (TAR) and data is written to the transfer hold register (TDR). The 9th bit of the TDR register is not applicable in this mode.

The following figure illustrates address and data transmission based on SEND_ADDR (CR3 [2]), TDR NULL conditions.

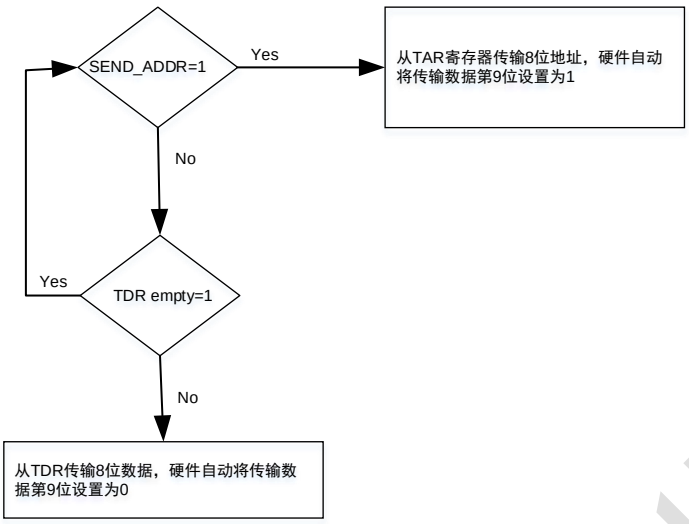


Figure 20-5 Automatic address transfer flow chart

The address of the destination slave to which data is to be transferred is programmed in the TAR register. The SEND_ADDR (CR3 [2]) bit must be enabled to transfer the destination slave address in the TAR register on the serial UART line, where the 9th data bit is set to 1, indicating that the address is being sent to the slave. UART clears the SEND_ADDR bit after the address character begins to be transmitted on the UART line.

The data needed to be transferred to the target slave is programmed through a transfer hold register (TDR). Data is transmitted on the UART line, and the 9th data bit is set to 0, indicating that data is being sent to the slave.

20.2.2.3. Transmission mode 1

In transfer mode 1, the TDR register is 9 bits wide and both addresses and data are programmed through the TDR register. UART does not distinguish between address and data. The SEND_ADDR (CR3 [2]) bit and the transfer address register (TAR) are not applicable for this mode. Depending on whether address/data must be sent, the software must write the 9th bit with 1/0.

Table 20-1 Transfer Relationship Table

M_E	TX_MODE	SEND_ADDR	Usage scenarios
0	X	X	Send 8-bit TDR data
1	0	0	Send "0 +8-bit TDR data"
1	0	1	Send "1 +8-bit TAR address"
1	1	X	Send "9-bit TDR data"

Note: This table describes several sending scenarios and corresponding configurations that can be used as a sender.

20.2.2.4. Receiving mode

UART supports two reception modes:

- Hardware address match receive mode (when ADDR_Match (CR3 [1]) is set to 1)
- Software address match receive mode (when ADDR_Match (CR3 [1]) is set to 0)

20.2.2.5. Hardware address matching receiving mode

In hardware address matching receive mode, if the 9th bit of the receive character is set to 1, the UART matches the receive character to the address programmed in the receive address register (RAR):

If the received address successfully matches the programmed address in the RAR register, the data byte is then received.

If the address match fails, the UART controller discards the data characters until a matching address is received.

The following figure shows the flow chart of data byte receiving based on the address matching function.

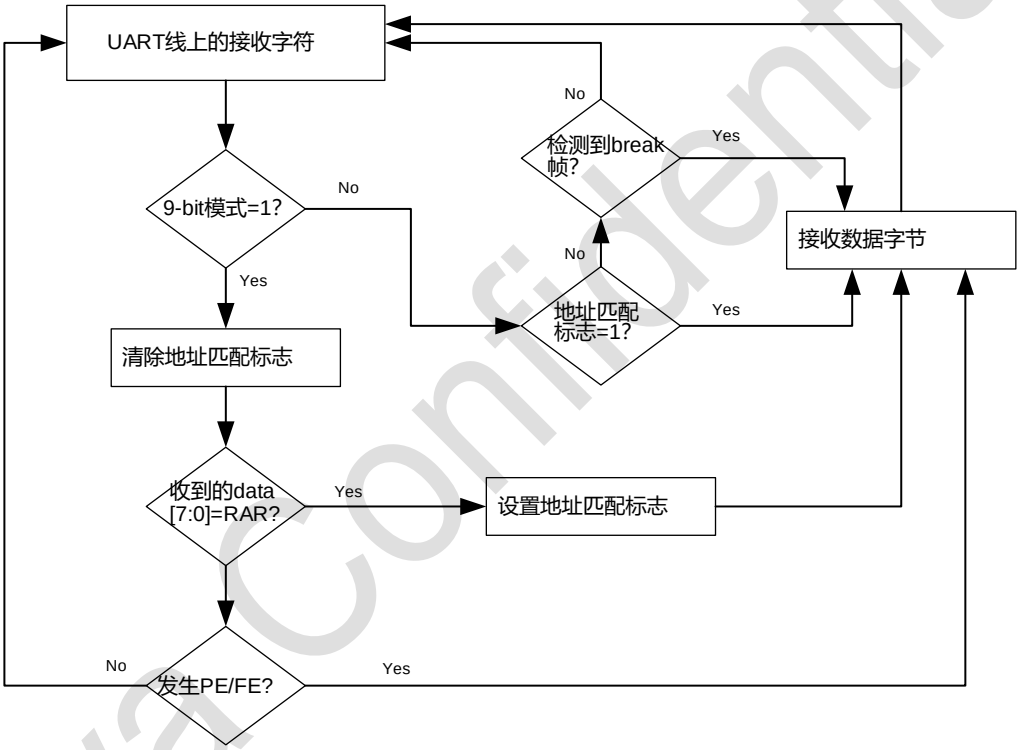


Figure 20-6 Hardware address matching receive mode

The UART receives characters regardless of whether the 9th bit of data is set to 1. If the 9th bit of the received character is set to 1, it clears the internal address match flag and then compares the received 8-bit character information with the address programmed in the RAR register.

If the received address character successfully matches the address programmed in the RAR register, the address match flag is set to 1 and the received character is pushed to the RDR register, and the ADDR_RCVD bit in the SR register is set to indicate that the address has been received, and the subsequent data byte (the 9th bit of the received character is set to 0) is pushed to or RDR.

If the address fails to match the RAR register and (parity or a frame error is found in the received address character), the received address string is still pushed to the RDR register, and the ADDR_RCVD and PE/FE error bits are set to 1.

If any interrupt character is received, UART treats it as a special character and pushes it to the RDR register regardless of the address match flag.

20.2.2.6. Software address matching receiving mode

In this mode of operation, the UART does not perform address matching to the received address character of the RAR register (the 9th bit of data is set to 1). The UART always receives 9 bits of data, advancing the RDR register. Whenever an address byte is received and indicated by the ADDR_RCVD bit in the line status register, the address must be compared by the user himself.

20.2.3. Baud rate

The baud rate of the UART is controlled by pclk and a frequency division latch register (BRR).

The baud rate is determined by:

- Serial clock operating frequency (pclk)
- Required baud rate
- Baud rate generator divisor (consists of BRR register)
- Acceptable baud rate error%

The equation for calculating the baud rate is as follows:

*Baud rate = serial clock operating frequency/(16 * DIVISOR)-equation (1)*

Where, DIVISOR-the number (hex) used to program the BRR.

Serial clock frequency-The frequency at the sclk or pclk pin of the UART.

According to equation (1), DIVISOR can be calculated as: DIVISOR = serial clock frequency/(16 * baud rate)

Also from equation (1), it can also be shown that:

Serial clock frequency = baud rate * 16 * divisor

The error between the baud rate and the selected baud rate calibration is as follows:

Percentage error = (baud rate-selected baud rate)/baud rate * 100%

Table 20-2 Error calculation when setting baud rate

Baud rate		**fpclk = 4 MHZ**			**fpclk = 24 MHZ**			**fpclk = 48 MHZ**		
No.	**kbps**	**Actual**	**Value placed in the baud rate register**	**Er-ror%**	**Actual**	**Value placed in the baud rate register**	**Er-ror%**	**Actual**	**Value placed in the baud rate register**	**Error%**
1	2.4	2.404	108	0.16 %	2.4	625	0.00 %	2.4	1250	0.00 %
2	9.6	9.615	26	0.16 %	9.615	156	0.16 %	9.615	326	0.16 %
3	19.2	19.231	13	0.16 %	19.231	78	0.16 %	19.231	156	0.16 %
4	57.6	62.5	4	8.51 %	57.692	26	0.16 %	57.692	52	0.16 %
5	115.2	125	2	8.51 %	115.385	13	0.16 %	115.385	26	0.16 %
6	230.4	250	1	8.51 %	250	6	8.51 %	230.769	13	0.16 %
7	460.8	Impos-sible	Impossible	Impos-sible	500	3	8.51 %	500	6	8.51 %
8	921.6	Impos-sible	Impossible	Impos-sible	1500	1	62.76 %	1000	3	8.51 %
9	2250	Impos-sible	Impossible	Impos-sible	Impossi-ble	Impossible	Impos-sible	3000	1	33.33 %
10	4500	Impos-sible	Impossible	Impos-sible	Impossi-ble	Impossible	Impos-sible	Impossi-ble	Impossible	Impossi-ble

Notes:

1. The lower the clock frequency of the CPU, the lower the error at a particular baud rate. The upper limit of the achievable baud rate can be obtained from this set of data.

2. When configuring the clock, because the baud rate clock is divided on the system clock, there will be an error between the obtained clock frequency and the actual clock frequency. The configured clock error needs to be within 2% to work properly.

20.2.4. The UART receiver tolerates changes in the clock

The UART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the UART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter's local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator
- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

DTRA + DQUANT + DREC + DTCL < UART receiver tolerance

For normally received data, the tolerance of the UART receiver is equal to the maximum tolerable change of 96% to 105%.

20.2.5. Interrupts and events

Assertion of UART interrupt output signal (intr)-An interrupt occurs as soon as one of several preferential interrupt types is enabled and activated.

When an interrupt occurs, the master may access the SR register to determine the type of interrupt. The following interrupt types can be enabled via the CR2 register:

- Receiver error
- Receiver data available
- Send hold register empty (in programmable TDR interrupt mode)
- Busy detection indication

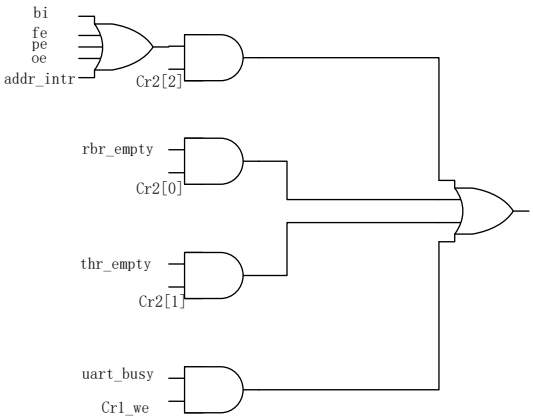


Figure20-7 UART interrupt map

These interrupt types are detailed in the table below.

Table 20-3interrupt control function

Interrupt type	Interrupt source	Clear interrupt
Receiver line status	Overflow/parity/frame error, interrupt or address receive interrupt	Write 1 and clear 0 for the corresponding bit
Receiver data available	Receiver data available	Read receiver buffer register
Send hold register empty	Send hold register empty	Write TDR
Busy monitoring	When UART is BUSY (BUSY [0] is set to 1), master tries to write to the CR1 register.	Write 1 and clear 0 for the corresponding bit

20.3. UART registers

20.3.1. Status register (UART_DR)

Address offset: 0x00

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
8:0	DR	RW	9'b0	Data value The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR). The RDR register is the data byte received on the serial input port (sin) in UART mode. The data in the status register (SR) is only valid if the Receive Non-Null (RXNE) bit in this register is set. The data in the RDR must be read before the next data arrives, otherwise it will be overwritten resulting in an over-run error. The TDR register is the data transferred on the serial output port (sout) in UART mode. The software guarantees that data can be written to the TDR again only if the TDR NULL (TDRE) bit (SR [5]) is set. If the TDRE is set, writing a single character to the TDR will clear the TDRE. Any additional writes to the TDR before the TDRE is set again will cause the TDR data to be overwritten. The 9th bit applies only if CR3 [3] = 1.

20.3.2. Baud rate register (UART_BRR)

Address offset: 0x04

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	BRR	RW	16'b0	This register constitutes a 16-bit divisor, which contains the baud rate divisor of UART. The output baud rate is equal to the serial clock (pclk) frequency divided by sixteen times the baud rate divisor, as follows: baud rate = (serial clock frequency)/(16 * divisor). Note: When the divisor latch register (BRR) is set to zero, the Baud clock is disabled and no serial communication occurs. Furthermore, once the BRR is set, you should wait for at least 8 clock cycles to pass before sending or receiving data.

20.3.3. Status register (UART_SR)

Address offset: 0x10

Reset value: 0x0000 0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	BUSY_ERR	BUSY	ADDR_RCVD	Res.	TXE	TDRE	BRI	FE	PE	ORE	RXNE
					R_W1	R	R_W1		R	R	R_W1	R_W1	R_W1	R_W1	R

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	Reserved
10	BUSY_ERR	R_W1	1’b0	Busy detect error: A misoperation when busy is detected. Error flag: When UART is busy (SR [9] is set to 1), master attempts to write to the CR1 register. Value: ■ 0x0: No busy misoperation error. ■ 0x1: When UART is busy (SR [9] is set to 1), master tries to write to the CR1 register. Write 1 to clear this bit
9	BUSY	R	1'b0	UART Busy. UART Busy This indicates that a serial transfer is in progress, and when cleared it indicates that the UART is idle or inactive. This bit will be set to 1 (busy) in either of the following cases: -Transfer ongoing on serial interface -Receiving on serial interface -Transmission of data present in TDR with non-zero baud divisor (BRR not equal to 0) -Receive data presence in RDR Note: The UART busy bit may be cleared even if a new character may have been sent from another device. That is, if the UART has no data in the TDR and RDR, and there is no ongoing transmission, and the start bit of the new character has just arrived at the UART. This is because a valid start is not seen until the middle of the bit period, and this duration depends on the programmed baud divisor. Value: 0x0 (IDLE): UART is IDLE or inactive Status: 0x1 (busy): UART busy (active data transfer)

Bit	Name	R/W	Reset Value	Function
8	ADDR_RCVD	R_W1	1'b0	Address receive. If 9-bit data mode is enabled (CR3 [0] = 1), this bit is used to indicate that the 9th bit of received data is set to 1. This bit can also be used to indicate whether the input character is an address or data. ■ 1 = means that the character is an address. ■ 0 = indicates that the character is data. Read SR Clear 9BIT. Note: The user needs to ensure that the interrupt is cleared before the next address byte arrives (this bit writes 1 clear 0). If there is a delay in clearing an interrupt, the software will not be able to distinguish between multiple address-dependent interrupts. 0x1 (RCVD_1): The incoming character is the address 0x0 (RCVD_0): Incoming character is data
7	Reserved	-	-	Reserved
6	TXE	R	1'b1	Send is empty. This bit is set whenever no data is currently being sent and the TDR is empty. 0x0 (DISABLED): Send is not empty 0x1 (ENABLED): Send is empty
5	TDRE	R	1'b1	Send keeps register empty. This bit indicates that the TDR is null. This bit is set whenever data is transferred from the TDR to the transmitter shift register and no new data is written to the TDR. This also causes a TDR interrupt to occur if it is enabled. 0x0 (DISABLED): TDR is not empty 0x1 (ENABLED): TDR is empty
4	BRI	R_W1	1'b0	break interrupt bit This is used to indicate that an interrupt sequence is detected on the serial input data. If in UART mode, it is set whenever the serial input sin remains in a logic "0" state for longer than the sum of the start time + data bits + parity bits + stop bits. Writing 1 to this bit will clear the BRI bit. 0x0 (NO_break): No break sequence detected 0x1 (break): break sequence detected
3	FE	R_W1	1'b0	Frame error bit. This is used to indicate that a frame error has occurred in the receiver. A frame error occurs when the receiver does not detect a valid STOP bit in the received data. It should be noted that if a break occurs, the frame error (FE) bit (SR [3]) will also be set, as shown by the break interrupt (BRI) bit (SR [4]). This happens because the break character implicitly generates a frame error by holding the sin input to a logical 0 longer than the duration of the character. Writing 1 to this bit will clear the FE bit. 0x0 (NO_FRAMING_ERROR): No frame error 0x1 (FRAMING_ERROR): Frame error
2	PE	R_W1	1'b0	Parity error bit. If the parity enable (PEN) bit (CR1 [3]) is set, this is used to indicate the occurrence of a parity error in the receiver. It should be noted that if a break occurs, in which case PE is also set if parity generation and detection is enabled (CR1 [3] = 1) and parity is set to odd (CR1 [4] = 0). Writing 1 to this bit will clear the PE bit. 0x0 (NO_PARITY_ERROR): No parity error

Bit	Name	R/W	Reset Value	Function
				0x1 (PARITY_ERROR): Parity error
1	ORE	R_W1	1'b0	Overflow error bit. This is used to indicate the occurrence of an overflow error. This happens if a new data character is received before the previous data is read. The ORE bit is set when a new character reaches the receiver before the previous character is read from the RDR. When this happens, the data in the RDR is overwritten. Writing 1 to this bit will clear the ORE bit. 0x0 (NO_OVER_RUN_ERROR): No overflow error 0x1 (OVER_RUN_ERROR): Overflow error
0	RXNE	R	1'b0	Data ready bit. This is used to indicate that the receiver includes at least one character in the RDR. This bit is cleared when the RDR is read. 0x0 (NOT_READY): Data not ready 0x1 (READY): Data READY

20.3.4. USART control register 1 (UART_CR1)

Address offset: 0x14

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	MSBFIRST	SWAP	Res	SBK	SP	PS	PCE	STOP	M	
						RW	RW		RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31: 10	Reserved	-	-	Reserved
9	MSBFIRST	RW	1'b0	MSB first. 0: After the start bit, the 0th bit data is transmitted and received; 1: After the start bit, send and receive the 5/6/7/8/9th bits of data; This bit cannot be modified during data transfer.
8	SWAP	RW	1'b0	Interchangeable Tx/Rx pins 0: TX/RX pins are defined according to standard pinout; 1: Interchangeable Tx/Rx pins When used as cross-wired to connect to other UARTs.
7	Reserved	-	-	Reserved
6	SBK	RW	1'b0	The break control bit. This is used to send the break to the receiving device. If set to 1, the serial output will be forced into an interval (logic 0) state. The sout line is forced low until the SBK bit is cleared. 0x0 (DISABLED): Release serial output for data transfer 0x1 (ENABLED): Serial output is forced into interval state
5	SP	RW	1'b0	Fixed parity. Writable only when UART is not BUSY (BUSY [0] is 0). This bit is used to force a fixed parity value. When PCE, PS, and Stick Parity are set to 1, a parity bit is sent and checked to be a logical 0. If PCE and Stick Parity are set to 1, and PS is a logical 0, a parity bit is sent and checked

Bit	Name	R/W	Reset Value	Function
				to be a logical 1. If this bit is set to 0, Stick Parity is disabled. 0x0 (DISABLED): Fixed parity DISABLED 0x1 (ENABLED): Fixed parity ENABLED
4	PS	RW	1'b0	Parity selection. Writable only when UART is not BUSY (BUSY [0] is zero). This is used to select between even and odd parity when parity is enabled (PCE set to 1). If set to 1, an even number of logical 1s is transmitted or checked. If set to zero, an odd number of logical ones is transmitted or checked. 0x0 (ODD_PARITY): odd check 0x1 (EVEN_PRITY): Even check
3	PCE	RW	1'b0	Parity enabled. Writable only when UART is not BUSY (BUSY [0] is zero). This bit is used to enable and disable parity generation and detection in transmitted and received serial characters, respectively. 0x0 (DISABLED): Parity disabled 0x1 (ENABLED): Parity enabled
2	STOP	RW	1'b0	Number of Stop bits. Writable only when UART is not BUSY (BUSY [0] is zero). This is used to select the number of stop bits for each character that the peripheral device will send and receive. If set to zero, a stop bit is transferred in the serial data. If it is set to 1 and the data bit is set to 5 (CR1 [1: 0] is set to 0), a half stop bit is sent. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver will only check the first Stop bit. Note: The STOP bit duration of UART implementations may be longer due to the idle time inserted between characters and the baud clock divisor value in the transmission direction for some configurations; 0x0 (STOP_1BIT): 1 Stop bit 0x1 (STOP_1_5BIT_OR_2BIT): 1.5/2 Stop bits
1:0	M	RW	2'b0	Data length selection. Writable only when UART is not BUSY (BUSY [0] is zero). When M_E in CR3 is set to 0, this register is used to select the number of data bits for each character that the peripheral device will send and receive. 0x0 (CHAR_5BITS): 5 data bits per character 0x1 (CHAR_6BITS): 6 data bits per character 0x2 (CHAR_7BITS): 7 data bits per character 0x3 (CHAR_8BITS): 8 data bits per character

20.3.5. USART control register 2 (UART_CR2)

Address offset: 0x18

Reset value: 0x0000_0008

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

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	Reserved
3	BUSYERRIE	RW	1’b1	Enable the BUSYERR state interrupt. This is used to enable/disable BUSYERR state interrupt generation. When UART is busy (SR [9] is set to 1), master tries to write to the CR1 register. Lowest priority = 4. 0x0: Disable BUSYERR status interrupt 0x1: Enable BUSYERR Status Interrupt
2	LSIE	RW	1'b0	Enable receiver line status interrupt. This is used to enable/disable the generation of receiver line state interrupts. This is the highest priority interrupt. The interrupt flag is a combination of PE, FE, OE, BI, ADDR_RCV interrupt flags. 0x0: Disable receiver line state interrupt 0x1: Enable Receiver Line State Interrupt
1	TDREIE	RW	1'b0	Transfer hold register null interrupt enabled This is used to enable/disable the generation of a transfer hold register null interrupt. This is the third highest priority interruption. 0x0: Transmission null interrupt disabled 0x1: Transmission null interrupt enabled
0	RXNEIE	RW	1'b0	Receive data availability interrupt enabled This is used to enable/disable the receive data available interrupt. These are the second highest priority interruptions. 0x0: Reception data interrupt disabled 0x1: Receive data interrupt enabled

20.3.6. USART control register 3 (UART_CR3)

Address offset: 0x1c

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	TX_MODE	SEND_ADDR	ADDR_MATCH	M_E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 4	Reserved	-	-	Reserved
3	TX_MODE	RW	1'b0	Transfer mode control bit. This bit is used to control the type of transmission mode during the 9-bit data transmission. Value: ■ 0x1 (TX_MODE_1): In this mode of operation, the transfer hold register (TDR) is 9 bits wide. The user needs to ensure that the address/data of the TDR register is written correctly. Address: 9th digit set to 1 Data: 9th digit set to 0 Note: The transfer address register (TAR) is not suitable for this mode of operation. ■ 0x0 (TX_MODE_0): In this mode of operation, the width of the transfer hold register (TDR) is 8 bits. The user needs to program the address into the transfer address register (TAR) and the data into the TDR register.

Bit	Name	R/W	Reset Value	Function
				■ The SEND_ADDR bit serves as a control knob indicating when the UART transmits an address.
2	SEND_ADDR	RW	1'b0	Send the address control bit. This bit serves as a control knob for the user to determine when to send an address in transmission mode. Note: 1. After the address character is issued, this bit is automatically cleared by the hardware. Users should not program this bit to 0. 2. This field is only applicable if the M_E bit is set to 1 and TX_MODE is set to 0. Value: ■ 0x1 (SEND_ADDR_1): 9-bit character content: The 9th bit is set to 1, and the remaining 8 bits will match what is being programmed in the transfer address register. ■ 0x0 (SEND_ADDR_0): 9-bit character content from TDR register
1	ADDR_MATCH	RW	1'b0	Address matching. This bit is used to enable address matching during reception. ■ In address match mode, UART will wait for an incoming character with the 9th bit set to 1. And further check whether the address matches the address programmed in the receive address match register. If there is a match, the subsequent characters are considered valid data and UART starts receiving data. ■ In normal mode, the UART will start receiving data and 9-bit characters will form. The user is responsible for reading data and distinguishing between b/n addresses and data. Note: This field applies only if M_E is set to 1. Value: ■ 0x0 (DISABLE): Normal mode ■ 0x1 (enabled): Address matching mode
0	M_E	RW	1'b0	Enable_9_bit This bit is used to enable the 9-bit data used to send and receive transmissions

20.3.7. Receive address register (UART_RAR)

Address offset: 0x20

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	RAR	RW	8'b0	This is the address match register in receive mode. If the 9th bit is set to 1 in the input character, the remaining 8 bits will be checked against the register value. If the match is successful, the subsequent characters with bit 9 set to 0 are treated as data bytes until the next address byte is received. Note: 1. This register is only applicable if the "ADDR_MATCH" (CR3 [1]) and "M_E" (CR3 [0]) bits are set to 1. 2. The RAR can be programmed at any point in time. However, the user must not change this register value while any reception is in progress.

20.3.8. Transmit address register (UART_TAR)

Address offset: 0x24

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	TAR	RW	8'b0	This is the address match register in transfer mode. If the M_E (CR3 [0]) bit is enabled and the SEND_ADDR (CR3 [2]) bit is set to 1, then UART will send a 9-bit character with the 9th bit set to 1, and the remaining 8-bit address will be sent from this register. Note: 1. This register is only used to send addresses. Normal data should be sent through the programmed TDR register. 2. After starting sending addresses on the UART serial channel, the hardware will automatically clear the "send_ADDR" bit

21. Touch key

21.1. Introduction

This module is a 26-channel high-sensitivity touch button sensor, which can realize touch applications such as air touch and proximity sensing. Its anti-jamming performance is superior and can pass the dynamic CS 10 V test. The unique low-power design allows the overall power consumption of the multi-button wake-up chip to be less than 8μA when touching in power-saving mode. The company provides a complete intelligent touch development kit, which can quickly complete the development and debugging of various touch applications in combination with the touch library and supporting tools provided by the company.

21.2. Main features

- Supports up to 26 touch keys
- Cmod capacitors can be built-in or external
- Supports two TK_SW modes: PWM mode and PRS mode, and the PRS mode polynomial can be flexibly set
- Support touch compensation function, compensation mode and voltage are optional
- Support multi-channel parallel connection
- Supports software mode and hardware mode
- Supports low power consumption normal and abnormal wake-up, and the overall power consumption of the multi-button wake-up chip can be less than 8μA

22. Debug support (DBG)

22.1. Overview

The device is built around a Cortex®-M0+ core which contains hardware extensions for advanced debugging features. The Debug extension allows the CPU to stop at a given instruction (breakpoint) or at data access (watchpoint). When stopped, the internal state of the CPU core and the external state of the system may be checked. Once the check is completed, the CPU Core and the system may be restored and the execution of the program continues.

The debug features are used by the debugger host when connecting to and debugging the MCUs. The debugging interface is serial wire. The debug features embedded in the Cortex®-M0+ core are a subset of the Arm® CoreSight Design Kit.

M0 + provides integrated on-chip debug support. It is comprised of:

- SW-DP: Serial wire
- BPU: Break point unit
- DWT: Data watchpoint trigger

It also includes debug features dedicated to the device:

- Flexible debug pinout assignment, SWIO @ PA13, SWCLK @ PA14
- MCU debug box (support for low-power modes, control over peripheral clocks, etc.)

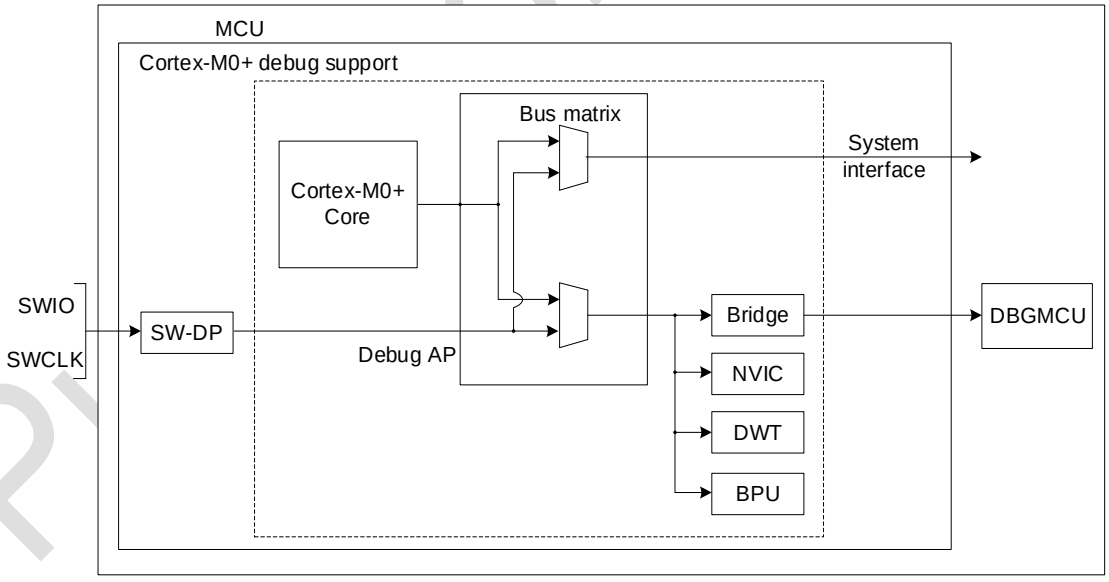


Figure 22-1 DBG block diagram

22.2. Pinout and debug port pins

22.2.1. SWD port

There are two pins related to debugging functions, which can be used as alternate functions of GPIO. These two pins are visible in all packaging forms.

Table 22-1 SWD port

SW-DP pin name	SWD debug port pins		Pin assignment
	Type	Debug assignment	
SWDIO	I/O	Serial Wire Data Input/Output	PA13
SWCLK	I	Serial Wire Clock	PA14

22.2.2. Flexible SWJ-DP pin assignment

After RESET (SYSRESETn or PORESETn), all pins used for the SWJ-DP are assigned as dedicated pins immediately usable by the debugger host.

In addition, the MCU offers the possibility to disable the SWD port and can then release the associated pins for general-purpose I/O (GPIO) usage

22.2.3. Internal pull-up & pull-down on SWD pins

Once the SW I/O is released by the user software, the GPIO controller takes control of these pins. The reset states of the GPIO control registers put the I/Os in the equivalent states:

- SWDIO: input pull-up
- SWCLK: input pull-down

On-chip pull-up and pull-down resistors save the need for additional resistance for the periphery.

22.3. ID codes and locking mechanism

There are several ID codes inside the MCU. It is strongly recommended the tool manufacturers (for example Keil and IAR) to lock their debugger using the MCU device ID located at address 0x40 015 800.

After the device is powered on, the hardware reads the factory config of the Flash. byte's 0x1FFF 01F8 address, loaded into the DBG_IDCODE register.

22.4. SWD port

22.4.1. 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

22.4.2. SWD protocol introduction

Each sequence consist of three phases:

- Packet requests sent by the master (8 bits)
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 22-2 Request packet (8-bits)

Bit	Acronym	Description
0	Start	Must be “1”
1	ApnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request 1: Read request
4:3	A[3:2]	Address field of the DP or AP registers
5	Parity	The check bit of the previous bit
6	Stop	0
7	Park	Not driven by the host. 1 is read out by the device due to the pull-up attribute.

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drive the line.

Table 22-3 ACK response (3 bits)

Bit	Acronym	Description
[2:0]	ACK	001: FAULT 010: WAIT 100: OK

The ACK Response must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 22-4 DATA transfer (33 bits)

Bit	Acronym	Description
[31:0]	WDATA or RDATA	Write or Read data
32	Parity	Single parity of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

22.4.3. 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+).

22.4.4. DP and AP read/write accesses

- The operation of reading DP will not be posted: the device response can be immediate (ACK = OK), or can be delayed (ACK = WAIT)
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. If the next access to be made is not an AP access, the DP-RDBUFF register must be called out to obtain this result.
- The READOK flag of the DP-CTRL/STAT register is updated at each AP read access or RDBUFF read request (knowing whether the AP read access was successful).
- SW-DP implements a write buffer (for DP and AP writes), which receives a write operation even when other operations are still incomplete. If the write buffer is full, the answer response is WAIT. IDCODE read, CTRL/STAT read or ABORT write are exceptions (even if the write buffer is full)
- Since SWCLK and HCLK are asynchronous clocks, two additional SWCLK cycles are required after the write operation (after the check bit) to ensure the internal validity of the write. These cycles should be applied when the drive signal line is low.

- The above is particularly important when writing CTRL/STAT for power-on requests. fail if the next action (which requires power-up) occurs immediately.

22.4.5. SW-DP registers

Access to these registers are initiated when APnDP=0

Table 22-5 SW_DP register

A[3:2]	R/W	CTRLSEL bit or SELECT registers	Register	Description
00	R		IDCODE	Fixed to 0x0BC11477
00	W		ABORT	-
01	RW	0	DP_CTRL/STAT	- Request a system or debug power-on; - Configure the transfer operation for AP accesses; - Control comparison, verification operation; - Read some status flags
01	RW	1	WIRE CONTROL	Configure the physical serialport protocol
10	R	-	READ RESEND	Enables recovery of the read data from a corrupted debugger transfer, without repeating the original AP transfer.
10	W	-	SELECT	Select the current access port and the active 4-words register window
11	RW	-	READ BUFFER	This read buffer is useful because AP accesses are posted (the result of a read AP request is available on the next AP transaction). This read buffer captures data from the AP, presented as the result of a previous read, without initiating a new transaction.

22.4.6. SW-AP registers

Table 22-6 SW-AP registers

Address	A [3: 2] value	Description
0x0	00	Reserved
0x4	01	DP CTRL/STAT register, used as -Request a system or debug power-up -Configure transport operations for AP access -controlling pushed comparison and pushed verification operations -Read some status flags (overflow, power-up response)
0x8	10	DP SELECTRION register: Used as a register to select the current access port and active 4 words in the window. -Bit 31: 24: APSEL: Select current AP -Bit 23: 8: Reserved -Bit 7: 4: APBANKSEL: In the current AP, select active 4 word registers window -Bit 3: 0: Reserved
0xC	11	DP RDBUFF register: Used to provide the debugger with the final result after a sequence of operations (without requesting a new JTAG-DP operation)

22.4.7. 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 the following four registers

Table 22- 7 Core debug registers

Register	Description
DHCSR	The 32-bit Debug halting control and status register
DCRSR	The 17-bit Debug core register selector register
DHCDR	The 32-bit Debug core register data register
DEMCR	The 32-bit Debug Exception and Monitor Control Register

These registers are not reset by a system reset. They will be reset by powered on reset. In order to Hart on reset, you need:

- bit0 (VC_CORRESET) of the debug and exception monitoring control register, enabled
- Debug stop control and status register (VC_DEBUGEN), enabled

22.5. BPU (Break Point Unit)

The Cortex®-M0+ BPU implementation provides four breakpoint registers. BPU is a set of ARMv7-M Flash Patch and Breakpoint (FPB) Blocks.

22.5.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.

Refer to the ARMv6-M ARM and ARM Coresight Components Technical Reference Manual for more information on BPU Coresight's identity registers and their addresses and access kinds.

22.6. DWT (Data Watchpoint)

The Cortex-M0 DWT provides 2 watchpoint registers.

22.6.1. DWT functionality

The processor watchpoints implement PC based watchpoint functionality.

22.6.2. DWT program counter sample register

A processor that implements the data watchpoint unit also implements the ARMv6-M optional DWT program counter sample register (DWT_PCSR). This register permits a debugger to periodically sample the PC without halting the processor. This mechanism provides coarse-grained analysis.

The Cortex®-M0+ DWT_PCSR records both instructions that pass their condition codes and those that fail.

22.7. MCU debug component (DBG)

The MCU debug component helps debuggers provide the following support:

- Low-power modes: Sleep, Stop
- Clock control of timer, watchdog and RTC during breakpoint

22.7.1. 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. Since these are the needs of the debugger connection, they must be kept on during a debugging period. The MCU integrates special means to allow the user to debug software in low-power modes.

Therefore, the debugger host must first set the contents of some debug configuration registers to change the low-power behavior:

- In Sleep mode: FCLK and HCLK are still valid. This mode does not impose any restrictions on standard debugging features.
- In Stop mode: The debugger host must set the DBG_Stop bit in advance. In this way, the system maintains FCLK and HCLK present in STOP mode.

22.7.2. Supports timers, watchdogs and RTC

During a breakpoint, it is necessary to choose how the timer's counter and watchdog want to behave:

- They can continue to count in breakpoint. This is typically required, for example, when a PWM is controlling a motor.
- They can stop and count inside the breakpoint. This is determined by the characteristics of watchdog.

22.8. DBG register

22.8.1. DBG device ID code register (DBG_IDCODE)

Address offset:0x00

Only 32-bit access supported. Read-only

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

Bit	Name	R/W	Reset Value	Function
31: 0	DBG_IDCODE[31:0]	R	0x0044 0020	This field indicates the device ID.

22.8.2. DBG configuration register (DBG_CR)

This register configures the low-power modes of the MCU under debug.

It is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support this feature, it is still possible for the user software to write to this register.

Address offset: 0x04

Reset value: 0x0000 0000 (not reset by system reset)

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

Bit	Name	R/W	Reset Value	Function
31: 2	Reserved	-	-	Reserved
1	DBG_STOP	RW	0	Debug stop mode 0: (FCLK = off, HCLK = off). In Stop mode, both HCLK and FCLK are off. When exiting from Stop mode, the clock configuration is identical to the one after RESET (CPU clocked by the internal RC oscillator (HSI)). Consequently, the software must reprogram the clock controller. 1: (FCLK = on, HCLK = on). In this case, when entering Stop mode, FCLK and HCLK are provided by the internal RC oscillator (HSI). When exiting Stop mode, the software must reprogram the clock controller if the clock control needs to be changed.
0	DBG_SLEEP	RW	0	Debugging sleep mode. 0: (FCLK ON, HCLK OFF). In Sleep mode, FCLK is provided by the previously configured system clock, and HCLK is turned off. Since sleep mode does not reset the configured clock system, the software does not need to reconfigure the clock after exiting sleep mode. 1: (FCLK=On, HCLK=On). In sleep mode, both the FCLK and HCLK clocks are provided by the previously configured system clock.

22.8.3. DBG APB freeze register 1 (DBG_APB_FZ1)

This register configures the clocking of timers, RTC and IWDG of the MCU under debug. The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Address offset: 0x08

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	DBG_ IWDG_STOP	Res	DBG_RTC_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
			RW		RW										

Bit	Name	R/W	Reset Value	Function
31: 13	Reserved	-	-	Reserved
12	DBG_IWDG_STOP	RW	0	When the CPU core is in the halt state, it controls the count clock of the IWDG. 0: Clock enabled; 1: Clock disabled;
11	Reserved	-	-	Reserved
10	DBG_RTC_STOP	RW	0	When the CPU core is in the halt state, it controls the counting clock of the RTC. 0: Clock enabled; 1: Clock disabled;
9:0	Reserved	-	-	Reserved

22.8.4. DBG APB freeze register 2 (DBG_APB_FZ2)

This register configures the clocking of timer counters when the MCU is under debug. The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Address offset: 0x0C

Reset value: 0x0000 0000

Only 32-bit access supported. Read-only

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res			Res
													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
31:16	Reserved	-	-	Reserved
15	DBG_TIM14_STOP	RW	0	When the CPU core is in the halt state, the counting clock of the TIM 14 is controlled. 0: Clock enabled; 1: Clock disabled;
14:12	Reserved	-	-	Reserved
11	DBG_TIM1_STOP	RW	0	When the CPU core is in the halt state, it controls the count clock of TIM1. 0: Clock enabled; 1: Clock disabled;
10:0	Reserved	-	-	Reserved

23. Revision history

Version	Date	Descriptions
V0.7	2025.10.24	1. Initial version 2. Consistent with Chinese version number

Puya Semiconductor Co., Ltd.