PY32L020 Series

32-bit ARM® Cortex®-M0+ Microcontroller

Reference Manual

Puya Semiconductor (Shanghai) Co., Ltd.

Contents

1. List of abbreviations for registers

Abbreviation	Description
Read/Write (RW)	Software can read and write to this bit.
Read-only (R)	Software can only read this bit.
Write-only (W)	Software can only write to this bit.
Read/Clear Write0 (RC_W0)	Software can read as well as clear this bit by writing 0. Writing 1 has no effect on this bit.
Read/Clear Write1 (RC_W1)	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on this bit.
Read/Clear 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

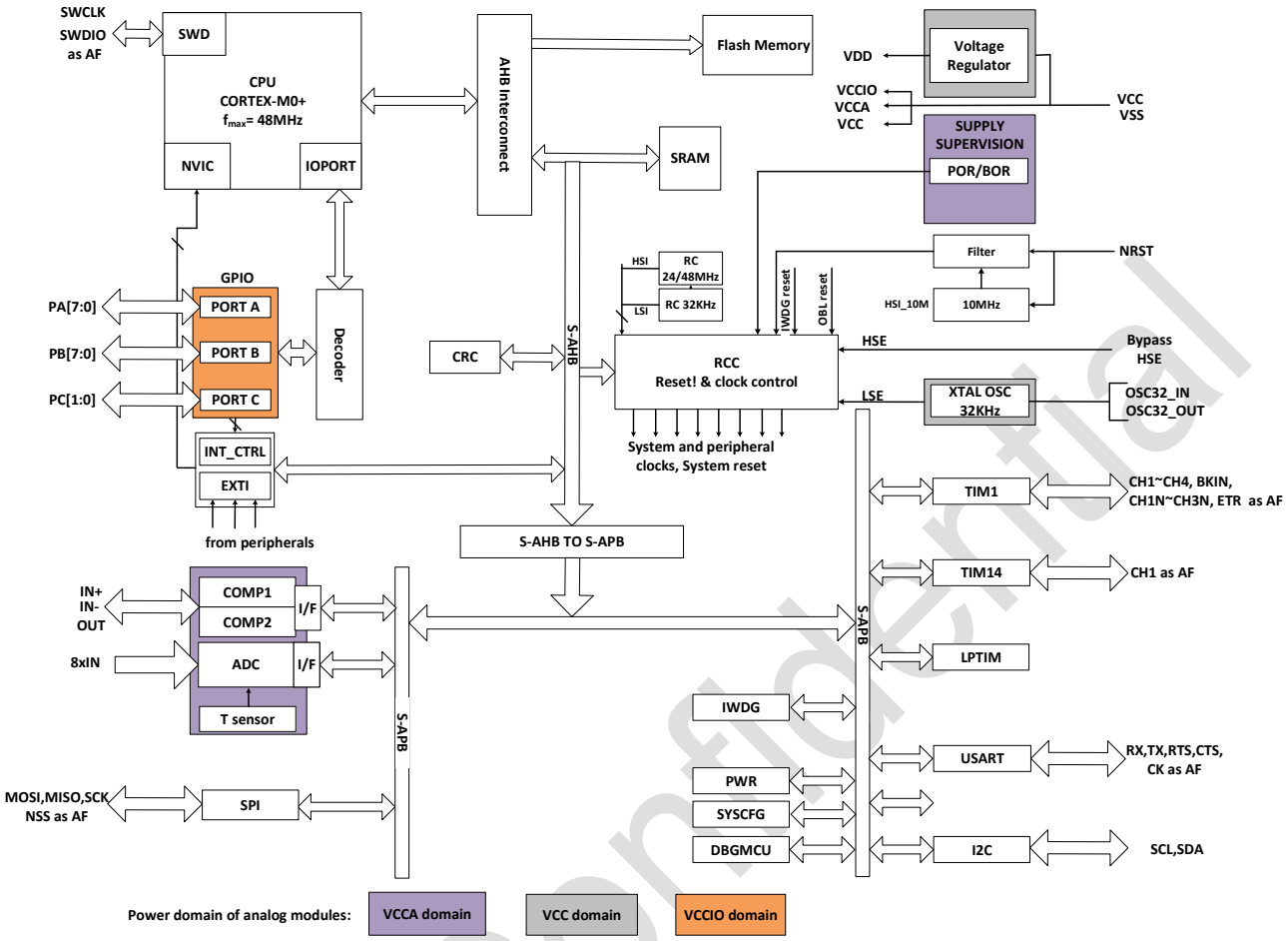

Figure 2-1 System block diagram

3. Memory and bus architecture

3.1. System architecture

The system consists of:

- A Master
  - Cortex-M0+
- Three Slaves
  - Internal SRAM
  - Internal Flash memory
  - AHB with AHB-APB bus bridge

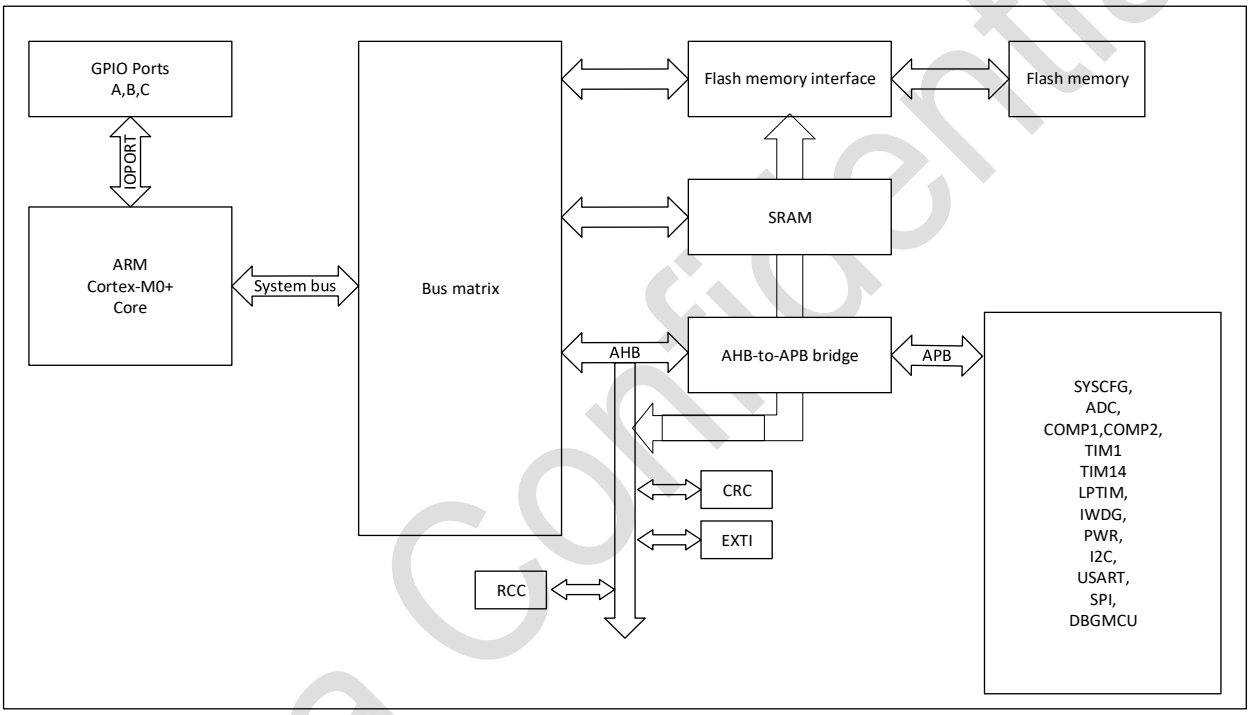

Figure 3-1 System architecture

- System bus

This bus connects the system bus of the Cortex-M0+ core to a Bus matrix.

- Bus matrix

The bus matrix consists of Master (CPU) and slaves (Flash memory, SRAM, and AHB-to-APB bridge).

- AHB-to-APB bus bridge

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, each of 512 MB.

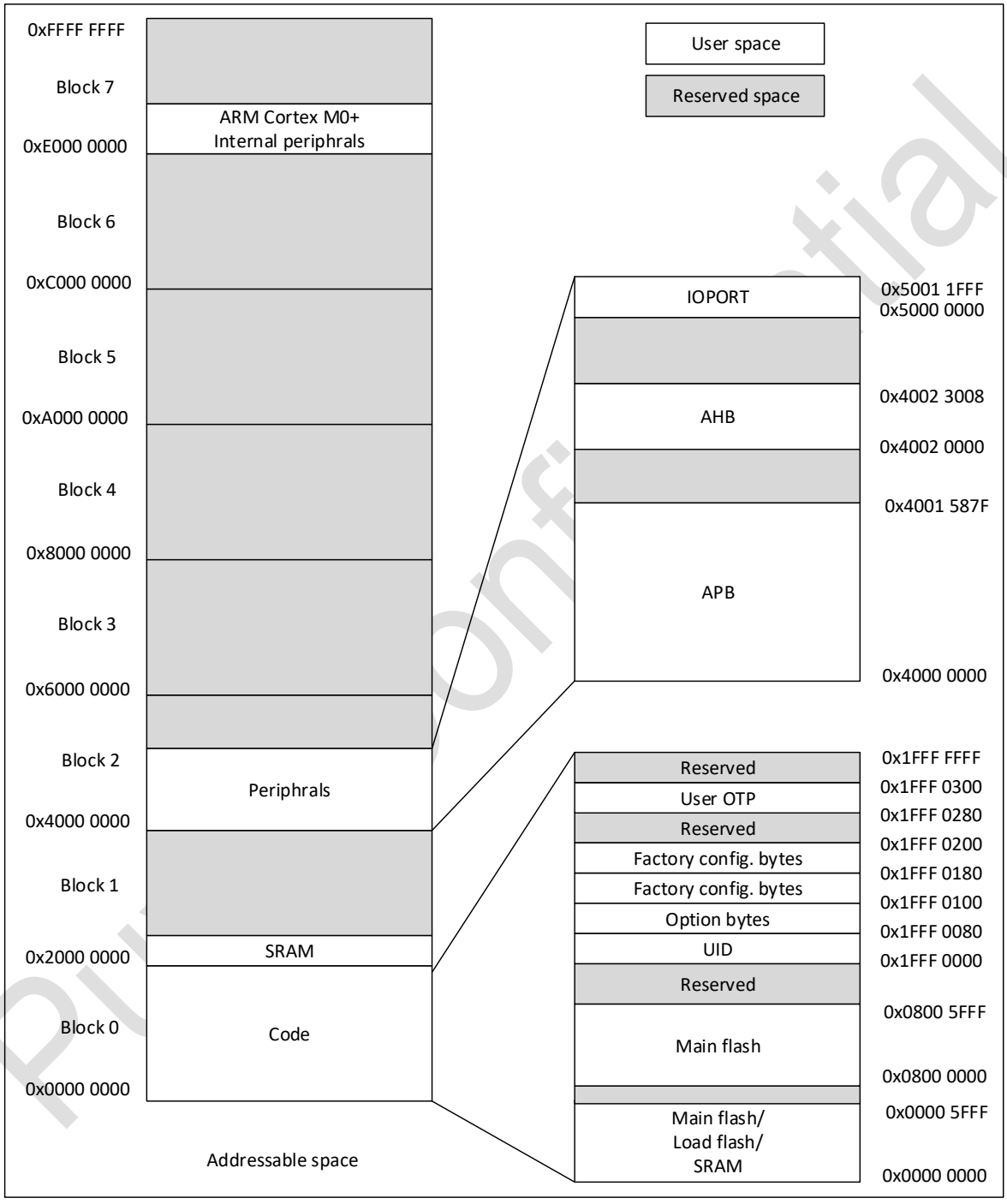

Figure 3-2 Memory map

Table 3-1 Memory boundary address

Type	Boundary address	Size	Memory area	Description
SRAM	0x2000 C000-0x3FFF FFFF	-	Reserved	-
	0x2000 0000-0x2000 0BFF	3 KB	SRAM	-
Code	0x1FFF 0300-0x1FFF FFFF	-	Reserved	-
	0x1FFF 0280-0x1FFF 02FF	128 Bytes	USER OTP memory	Store user data
	0x1FFF 0180-0x1FFF 01FF	128 Bytes	Factory configuration bytes 1	Store trimming data (including HSI trimming) and power-on verification code reading
	0x1FFF 0100-0x1FFF 017F	128 Bytes	Factory configuration bytes 0	HSI trimming data, Flash erase/write time configuration parameters
	0x1FFF 0080-0x1FFF 00FF	128 Bytes	Option bytes	Software and hardware option bytes information
	0x1FFF 0000-0x1FFF 007F	128 Bytes	UID	Unique ID
	0x0800 6000-0x1FFE FFFF	-	Reserved	-
	0x0800 0000-0x0800 5FFF	24 KB	Main flash	-
	0x0000 6000-0x07FF FFFF	-	Reserved	-
	0x0000 0000-0x0000 5FFF	24 KB	Depending on the Boot configuration: 1. Main flash 2. Load flash 3. SRAM	-

The address is marked as **Reserved**, which cannot be written, read as 0, and a response error is generated.

Table 3-2 Peripheral register address

Bus	Boundary address	Size	Peripheral
	0xE000 0000-0xE00F FFFF	-	M0+
IOPORT	0x5000 0C00-0x5FFF FFFF	-	Reserved
	0x5000 0800-0x5000 0BFF	1 KB	GPIOC
	0x5000 0400-0x5000 07FF	1 KB	GPIOB
	0x5000 0000-0x5000 03FF	1 KB	GPIOA
AHB	0x4002 3400-0x4FFF FFFF	-	Reserved
	0x4002 300C-0x4002 33FF	1 KB	Reserved
	0x4002 3000-0x4002 3008		CRC
	0x4002 2400-0x4002 2FFF	-	Reserved
	0x4002 2000-0x4002 23FF	1 KB	Flash
	0x4002 1C00-0x4002 1FFF	-	Reserved
	0x4002 1900-0x4002 1BFF	1 KB	Reserved
	0x4002 1800-0x4002 18FF		EXTI
	0x4002 1400-0x4002 17FF	-	Reserved
	0x4002 1080-0x4002 13FF	1 KB	Reserved
	0x4002 1000-0x4002 107F		RCC
	0x4002 0000-0x4002 0FFF	-	Reserved
APB	0x4001 5C00-0x4001 FFFF	-	Reserved
	0x4001 5880-0x4001 5BFF	1 KB	Reserved
	0x4001 5800-0x4001 587F		DBG
	0x4001 3C00-0x4001 57FF	-	Reserved
	0x4001 381C-0x4001 3BFF	1 KB	Reserved
	0x4001 3800-0x4001 3018		USART1

Bus	Boundary address	Size	Peripheral
	0x4001 3400-0x4001 37FF	-	Reserved
	0x4001 3010-0x4001 33FF	1 KB	Reserved
	0x4001 3000-0x4001 300C		SPI1
	0x4001 2C50-0x4001 2FFF	1 KB	Reserved
	0x4001 2C00-0x4001 2C4C		TIM1
	0x4001 2800-0x4001 2BFF	-	Reserved
	0x4001 270C-0x4001 27FF	1 KB	Reserved
	0x4001 2400-0x4001 2708		ADC
	0x4001 0400-0x4001 23FF	-	Reserved
	0x4001 0220-0x4001 03FF	1 KB	Reserved
	0x4001 0200-0x4001 021F		COMP1/2
	0x4001 0000-0x4001 01FF		SYSCFG
	0x4000 8000-0x4000 FFFF	-	Reserved
	0x4000 7C28-0x4000 7FFF	1 KB	Reserved
	0x4000 7C00-0x4000 7C24		LPTIM
	0x4000 7400-0x4000 7BFF	-	Reserved
	0x4000 7018-0x4000 73FF	1 KB	Reserved
	0x4000 7000-0x4000 7014		PWR
	0x4000 5800-0x4000 6FFF	-	Reserved
	0x4000 5434-0x4000 57FF	1 KB	Reserved
	0x4000 5400-0x4000 5430		I2C
	0x4000 3400-0x4000 53FF	-	Reserved
	0x4000 3014-0x4000 33FF	1 KB	Reserved
	0x4000 3000-0x4000 0010		IWDG
	0x4000 2400-0x4000 2FFF	-	Reserved
	0x4000 2054-0x4000 23FF	1 KB	Reserved
	0x4000 2000-0x4000 0050		TIM14
	0x4000 0000-0x4000 1FFF	-	Reserved

3.3. Embedded SRAM

PY32L020 series feature 3 KB SRAM. It is accessed by byte (8 bits), half-word (16 bits) or word (32 bits). Accessing memory outside the defined address range generates a HardFault interrupt.

3.4. Flash memory

The Flash memory is composed of two distinct physical areas:

- The Main flash area: 24 KB, consisting of application and user data, with up to 4 KB configurable as a User Bootloader through customer settings.
- 0.75 KB of Information area:
  - Factory config. bytes 0, 128 Bytes, used to store:
    - HSI frequency selection value and corresponding trimming value

- Configuration parameter values of erase and write time corresponding to different frequencies of HSI

— Factory config. bytes 1, 128 Bytes, used to store:
- Store trimming data (including HSI trimming)
- Power-on verification code reading

— UID: 128 Bytes, used to store the UID

— Option byte: 128 Bytes, used to store configuration values for hardware and storage protection

— User OTP memory: 128 Bytes, used to store user data

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

3.5. Boot modes

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

Table 3-3 Boot configuration

Boot modes		Mode	
nBOOT1 bit	**nBOOT0 bit**	**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

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 Load flash: the Load flash memory is aliased in the boot memory space (0x0000 0000), but still accessible from the following address depending on Load flash memory space.

User Bootloader	Access address
None	None
1 KB	0x0800 5C00 - 0x0800 5FFF
2 KB	0x0800 5800 - 0x0800 5FFF
3 KB	0x0800 5400 - 0x0800 5FFF
4 KB	0x0800 5000 - 0x0800 5FFF

- 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

Once the boot mode is selected, the application software can modify the memory accessible in the code area. This modification is performed by programming the MEM_MODE bits in the SYSCFG_CFGR1 (see the SYSCFG chapter for details).System configuration controller (SYSCFG)

4. Embedded Flash memory

4.1. Flash main features

- Main flash block: maximum 24 KB (6 K x 32 bits)
- Information block: 0.75 KB (192 x 32 bits)
- Page size: 128 Bytes
- Sector size: 4 KB

Flash memory interface features:

- Flash program / erase operation
- Write protection
- Read protection

4.2. Flash memory functional description

4.2.1. Flash memory organization

The Flash memory is organized as 64-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, it can be devided into Main flash (max capacity 24 KB) and Information flash (max capacity 0.75 KB).

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

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

Table 4-1 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	Page 128-159	0x0800 4000-0x0800 4FFF	4 KB
	Sector 5	Page 160-191	0x0800 5000-0x0800 5FFF	4 KB
UID	Sector 6	Page 0	0x1FFF 0000-0x1FFF 007F	128 Bytes
Option byte		Page 1	0x1FFF 0080-0x1FFF 00FF	128 Bytes
Factory config 0		Page 2	0x1FFF 0100-0x1FFF 017F	128 Bytes
Factory config 1		Page 3	0x1FFF 0180-0x1FFF 01FF	128 Bytes
Reserved		Page 4	0x1FFF 0200-0x1FFF 027F	128 Bytes
USER OTP memory		Page 5	0x1FFF 0280-0x1FFF 02FF	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.

The instruction fetch and data access are both done 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. When Latency is set as zero, wait state of the Flash read operation is not added. When set as one, one wait state is added. 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 offers quick and efficient design iterations.

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 during Flash program and erase operations, the contents of the Flash memory are not protected.

During a program/erase operation to the Flash memory, any attempt to read the Flash memory will stall 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 program and erase operations on the Flash memory (write/erase), the HSI must be ON.

4.2.3.1. Unlocking the Flash memory

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

This is done in the following steps:

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

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

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

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

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

4.2.3.2. Flash memory programming

Flash memory writes the entire page in word (32-bit) units at a time.

Attention:It must be in word units. Performing half-word or byte operations will produce HardFault!

The program operation is started when the CPU writes a half-word into a Main flash memory address with the PG bit of the FLASH_CR register set. Any attempt to write data that are not full word long will result in a bus error generating a HardFault interrupt.

If the addressed Main flash memory location is write-protected by the FLASH_WRPR register, the program operation is skipped and a warning is issued by the WRPRTERR bit in the FLASH_CR 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 WRPRTERR bit of the FLASH_CR 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
11. When the step 7 is carried out, the program operation is automatically started and the BSY bit is set simultaneously.
12. Flash memory erase

Flash memory can be erased according to page, or can be erased by sector erase and mass erase (sector erase and mass erase do not work on information memory).

4.2.3.3. 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 WRPRTERR bit will also be set. A page erase operation needs to perform the following steps:

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

4.2.3.4. Mass erase

Mass erase is used to erase the entire Main flash, but it does not work on the Information area. Further, 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:

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

4.2.3.5. Sector erase

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

The steps for sector erase are as follows:

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

Information memory except that the user programs memory once is read-only and will never be programmed/erased.

4.2.3.6. Write and erase time configuration

The time of Flash program and erase needs to be strictly controlled, otherwise the operation will fail. The software needs to read out the data from the corresponding info area address and then write it to the corresponding register to realize the configuration of the required erasing and writing time. When the HSI output frequency is changed, the Flash program and erase time control registers need to be properly configured according to the following table.

Table 4-2 Program and erase time configuration

Register	HSI 24 MHz corresponding info area address	HSI 48 MHz corresponding info area address
TS0	0x1FFF 011C	0x1FFF 0130
TS1	0x1FFF 011C	0x1FFF 0130
TS2P	0x1FFF 0120	0x1FFF 0134
TPS3	0x1FFF 0120	0x1FFF 0134
TS3	0x1FFF 011C	0x1FFF 0130
PERTPE	0x1FFF 0124	0x1FFF 0138
SMERTPE	0x1FFF 0128	0x1FFF 013C
PRGTPE	0x1FFF 012C	0x1FFF 0140
PRETPE	0x1FFF 012C	0x1FFF 0140

4.3. Unique device ID (UID)

Typical application scenarios of unique identification code:

- Used as a serial number
- When programming internal flash memory, use it as a key or encryption primitive to improve code security
- To activate secure boot processes, etc.

The UID provides a reference number that is unique to any device.

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

4.4. Flash option bytes

4.4.1. Flash option byte

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 its real and complement code form.

Table 4-3 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 4-4 Option byte organization

Word Address	Description
0x1FFF 0080	User option byte and complemented code
0x1FFF 0084	SDK area address option byte and complemented code
0x1FFF 0088	Boot control option byte and complemented code
0x1FFF 008C	WRP address option byte and complemented code
0x1FFF 0090	Reserved
0x1FFF 0094	Reserved
…	Reserved
…	Reserved
…	Reserved
0x1FFF 00FC	Reserved

- Option bytes for Flash user options

Flash memory address: 0x1FFF 0080

Production value: 0x4F55 B0AA

After the power-on reset (POR/BOR/OBL_LAUNCH) is released, the corresponding value is read from the selection 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	~ SWD_MODE	~ IWDG _SW	~BOR_LEV[2:0]			~BOR_ EN	Res							
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	SWD_MODE	IWDG _SW	BOR_LEV[2:0]			BOR_ EN	Res							
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	~SWD_MODE	R	Complemented code of SWD_MODE
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
23:16	Reserved	-	-
15	IWDG_STOP	R	Set the running state of IWDG timer in Stop mode 0: Freeze timer 1: Normal operation
14	NRST_MODE	R	NRST_MODE SWD_MODE 0 X:PC0:NRST PB6:SWD 1 0:PC0:GPIO PB6:SWD 1 1:PC0:SWD PB6:GPIO
13	SWD_MODE	R	
12	IWDG_SW	R	0: Hardware window watchdog 1: Software window watchdog
11:9	BOR_LEV[2:0]	R	000:BOR rising threshold is 1.8V and falling threshold is 1.7V 001:BOR rising threshold is 2.0V and falling threshold is 1.9V 010:BOR rising threshold is 2.2V and falling threshold is 2.1V 011:BOR rising threshold is 2.4V and falling threshold is 2.3V 100:BOR rising threshold is 2.6V and falling threshold is 2.5V 101:BOR rising threshold is 2.8V and falling threshold is 2.7V 110:BOR rising threshold is 3.0V and falling threshold is 2.9V 111:BOR rising threshold is 3.2V and falling threshold is 3.1V
8	BOR_EN	R	BOR enabled 0:BOR disabled 1:BOR enabled, BOR_LEV works
7:0	Reserved	-	-

■ Flash SDK area address option byte

Flash memory address: 0x1FFF 0084

Production value: 0xFFF8 0007

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	Stop address of SDK area, STEP corresponding to each bit is 2 KB
7:4	Reserved	-	-
3:0	SDK_STRT[3:0]	R	Start address of SDK area, STEP corresponding to each bit is 2 KB

■ Option byte for FLASH boot control

Flash memory address: 0x1FFF 0088

Production value: 0xFFFF 0000

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ nBOOT1	~ BOOT0	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	~BOOT_SIZE [2:0]		
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	Res	Res	Res	Res	Res	Res	Res	BOOT_SIZE [2:0]		
R	R	-	-	-	-	-	-	-	-	-	-	-	R	R	R

Bit	Name	R/W	Function
31	~ nBOOT1	R	Complemented code of nBOOT1
30	~ BOOT0	R	Complemented code of BOOT0
29:19	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	
13:3	Reserved	-	-
2:0	BOOT_SIZE [2:0]	R	Select Main flash area as Load flash 000: No Load flash area 001:1 KB (0x0800 5C00-0x0800 5FFF) 010:2 KB (0x0800 5800-0x0800 5FFF) 011:3 KB (0x0800 5400-0x0800 5FFF) 1xx:4 KB (0x0800 5000-0x0800 5FFF)

- Flash write-protected address

Flash memory address: 0x1FFF 008C

Production value:0xFFC0 003F

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	~WRP[5:0]					
-	-	-	-	-	-	-	-	-	-	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	Res	Res	WRP[5:0]					
-	-	-	-	-	-	-	-	-	-	R	R	R	R	R	R

Bit	Name	R/W	Function
31:22	Reserved	-	-
21:16	~ WRP	R	Complemented code of WRP
15:6	Reserved	-	-
5:0	WRP	R	0: sector [y] is protected 1:sector[y] is unprotected y = 0 to 5

4.4.2. Flash option bytes

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

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. In the case of a wrong key sequence, a bus error is detected, 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_BTCR / FLASH_WRP
4. Set OPTSTRT bit
5. Write any 32-bit data to the Main flash 0x4002 2080 address (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, FLASH_BTCR or FLASH_WRPR registers. The hardware automatically calculates the corresponding complement 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, 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 complemented code at its same double-word address (next half-word). During the option byte load, the option bit and its complemented code are verified, which ensures that the load is carried out correctly.

If the word and its complement are matching, the option word/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 not enabled)

- The NRST_MODE bit is written as 0 (reset input only)
- The rest of the mismatched values are written as 1

- For the SDK address option, SDKR_STRT [3: 0] = 0x0, SDKR_END [3: 0] = 0xB, that is, all Flash space is set to SDK
- For Flash boot startup selection
  - 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 WRP option, the mismatched value is the default value Unprotected

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

- 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 options.

4.5. Flash config bytes

A partial section (2 pages in total) of the Flash information area in the chip is used as the Factory config. bytes use.

Page 2 stores information for software to read (only the code, not its complement is stored):

- 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

Page 3 stores hardware factory information (the word and its complement storage):

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

For data security, Page 3's Factory config. bytes are stored separately in the form of the word and its complement.

Table 4-5 Factory config. bytes configuration

Page	Word	Address	Contents
1	0-3	0x1FFF 0000-0x1FFF 000F	UID
	4-7	0x1FFF 0010-0x1FFF 001F	Reserved
	8	0x1FFF 0020	1.2 V VREFINT(decimal with positive upper 16 bits)
	9-10	0x1FFF 0024-0x1FFF 002B	Reserved
	11	0x1FFF 002C	1.5 V VREFBUF trimming value (upper 16 positive decimal)
	12-31	0x1FFF 0030-0x1FFF 007F	Reserved
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

Page	Word	Address	Contents
	2	0x1FFF 0108	Reserved
	3	0x1FFF 010C	Reserved
	4	0x1FFF 0110	Reserved
	5	0x1FFF 0114	Normal temperature temperature sensor data
	6	0x1FFF 0118	High Temperature Sensor 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
	17-31	0x1FFF 0144- 0x1FFF 017F	Reserved
3	0	0x1FFF 0180	Read check code 0x55AA AA55
	1	0x1FFF 0184	Power on read check code 0xAA55 55AA
	2	0x1FFF 0188	Read check code 0x55AA AA55
	3	0x1FFF 018C	Power on read check code 0xAA55 55AA
	4	0x1FFF 0190	PMU trimming bit and complemented code
	5	0x1FFF 0194	PMU trimming bit and complemented code
	6	0x1FFF 0198	PMU trimming bit and complemented code
	7	0x1FFF 019C	Reserved
	8	0x1FFF 01A0	Stores HSI 24 MHz frequency selection control and corresponding trimming values and its complemented code
	9	0x1FFF 01A4	LSI 32.768 kHz frequency selection control and corresponding trimming value and its complemented code
	10	0x1FFF 01A8	Reserved
	11	0x1FFF 01AC	Reserved
	12	0x1FFF 01B0	Reserved
	13	0x1FFF 01B4	Reserved
	14	0x1FFF 01B8	Flash trimming and complemented code
	15	0x1FFF 01BC	Flash trimming and complemented code
	16	0x1FFF 01C0	Flash trimming and complemented code
	17	0x1FFF 01C4	Flash trimming and complemented code
	18	0x1FFF 01C8	Flash trimming and complemented code

Page	Word	Address	Contents
	19	0x1FFF 01CC	Flash trimming and complemented code
	20	0x1FFF 01D0	TS trimming and complemented code
	21	0x1FFF 01D4	Reserved
	22	0x1FFF 01D8	Reserved
	23	0x1FFF 01DC	Reserved
	24	0x1FFF 01E0	Reserved
	25	0x1FFF 01E4	Reserved
	26	0x1FFF 01E8	Reserved
	27	0x1FFF 01EC	Reserved
	28	0x1FFF 01F0	Reserved
	29	0x1FFF 01F4	Reserved
	30	0x1FFF 01F8	Device ID code
	31	0x1FFF 01FC	Reserved

4.5.1. 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.

4.5.2. 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.

4.5.3. 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.

4.5.4. 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
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

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

4.5.5. HSI_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	SMER TPE[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.

4.5.6. 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.

4.5.7. LSI_32.768K

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.

4.5.8. 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 4-6 USER OTP memory bytes configuration

Page	Word	Address	Contents
5	0	00x1FFF 0280	Bit[31:16]: store user data Bit[15:0]: USER OTP MEMORY_LOCK
	1	00x1FFF 0284	Store user data
	2	00x1FFF 0288	Store user data
	…	…	Store user data
	…	…	Store user data
	…	…	Store user data
	31	00x1FFF 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 4-7 Write protection status of Flash USER OTP memory bytes

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

4.6. Flash protection

The protection of Main flash area includes the following mechanisms:

- SDK (software design kit) protection, used to protect the access of specific program areas, the size is 2 KB.
- 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.

4.6.1. SDK area protection

The protection area is defined by SDKR_STRT [3:0], SDKR_END [3:0] of the FLASH_SDKR register, and each bit corresponds to 2 KB.

Start address:

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

End address:

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

When SDK_STRT [3: 0] is greater than SDK_END [3: 0], SDK protection is invalid. SDK protection is valid when SDK_STRT [3: 0] is less than or equal to SDK_END [3: 0].

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 a mass erase (the program whose SDK area is protected has been written before, and full erase plays a role in protecting the SDK area program), and then update the value of the SDK option in the Flash option byte (the updated value at this time is that the

SDK protection is invalid). The 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.

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

region	SDK region protection level	Boot from Main flash (CPU)						Commissioning/ Execute from SRAM		
		Users actions (From Non SDK area)			Users actions (From SDK area)					
		Read	Write	Erase	Read	Write	Erase	Read	Write	Erase
Non SDK area	Disabled	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes
	Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SDK area	Disabled	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
System memory	-	Yes	No	No	Yes	No	No	Yes	No	No
Option byte	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Factory bytes	-	Yes	No	No	Yes	No	No	Yes	No	No
UID	-	Yes	No	No	Yes	No	No	Yes	No	No

Notes:

1. A mass erase command issued by any area will erase the SDK area.
2. There are two situations for executing a program from SRAM: one is to start by setting Boot, and the other is to start from another memory, and the program jumps to SRAM.
3. N/A means that when the SDK area is disabled, since there is no SDK area, there is no read program in the SDK area in the above table, and there is no access to the SDK area by read programs from other areas.

4.6.2. Flash write protection

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

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

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

Note: Write protection only works on Main flash.

4.6.3. 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.

4.6.4. Option byte write protection

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

4.7. Flash interrupt

Table 4-9 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 HardFault:

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

4.8. Flash registers

4.8.1. Flash access control register (FLASH_ACR)

Address offset:0x00

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	-
0	LATENCY	RW	0	Waiting state corresponding to Flash read operation: 0: No waiting state for Flash read operation (system clock at 24 MHz and below) 1: The Flash read operation has a waiting state, that is, each read of Flash requires two system clock cycles Note: The system clock can only be set to 1 above 24 MHz.

4.8.2. Flash key register (Flash_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

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

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

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

4.8.3. Flash option key register (Flash_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

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

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

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

4.8.4. Flash status register (Flash_SR)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	-
16	BSY	R	0	Busy bit 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 trim bit load validity error When the option and trimming bits and their complement do not match, the hardware sets this bit. Loads mismatched option bytes, forced to safe values. This bit is cleared by writing 1.
14:5	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.

4.8.5. Flash control register (FLASH_CR)

Address offset: 0x14

Reset value: 0xC000 0000

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

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	FLASH_CR Lock bit. 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	1	Option byte Lock bit. The software can only set this bit. After reset, the bits associated with the option bytes in the FLASH_CR register are locked. When the 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	Forces option byte loading. When set, this bit forces the system to reload the option byte. 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	RW	-	-
19	PGSTRT	RW	0	The start bit of the program operation of the Main flash. This bit starts the program operation of the Main flash, 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	Flash option byte modified start bit This bit initiates the modification of the option byte. Software set, after the BSY bit of the FLASH_SR register is cleared, hardware clears this bit. Note: When the flash option bytes are modified, the hardware automatically performs an erase operation on the entire 128 Bytes page, and then performs a program operation, which also includes automatic write of complemented code.
16:12	Reserved	-	-	-

Bit	Name	R/W	Reset Value	Function
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

4.8.6. Flash option register (FLASH_OPTR)

Address offset: 0x20

Reset value: 0x0000 B0AA

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 -															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IWDG_STOP	NRST_MODE	SWD_MODE	IWDG_SW	BOR_LEV[2:0]			BOR_EN	Res							
RW	RW	RW	RW	RW	RW	RW	RW	-							

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15	IWDG_STOP	RW	1	Set the running state of IWDG timer in Stop mode 0: Freeze timer 1: Normal operation
14	NRST_MODE	RW	0	NRST_MODE SWD_MODE 0 X:PC0:NRST PB6:SWD 1 0:PC0:GPIO PB6:SWD 1 1:PC0:SWD PB6:GPIO
13	SWD_MODE	RW	1	
12	IWDG_SW	RW	1	0: Hardware window watchdog 1: Software window watchdog
11:9	BOR_LEV[2:0]	RW	3'h0	000:BOR rising threshold is 1.8V and falling threshold is 1.7V 001:BOR rising threshold is 2.0V and falling threshold is 1.9V 010:BOR rising threshold is 2.2V and falling threshold is 2.1V 011:BOR rising threshold is 2.4V and falling threshold is 2.3V 100:BOR rising threshold is 2.6V and falling threshold is 2.5V 101:BOR rising threshold is 2.8V and falling threshold is 2.7V 110:BOR rising threshold is 3.0V and falling threshold is 2.9V 111:BOR rising threshold is 3.2V and falling threshold is 3.1V
8	BOR_EN	RW	0	BOR enabled 0:BOR disabled 1:BOR enabled, BOR_LEV works
7:0	Reserved	-	-	-

4.8.7. Flash SDK address register (FLASH_SDKR)

Address offset: 0x24

Reset value: 0x0000 0007

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															
-															
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]			
-	-	-	-	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	4'h0	Stop address of SDK area, STEP corresponding to each bit is 2 KB
7:4	Reserved	-	-	-
3:0	SDK_STRT[3:0]	RW	4'h7	Start address of SDK area, STEP corresponding to each bit is 2 KB

4.8.8. Flash boot control (FLASH_BTCR)

Address offset: 0x28

Reset value: 0x0000 0000

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

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15	nBOOT1	RW	2'h0	Boot mode of nBOOT1 / BOOT0 X0:Boot from Main flash 11:Boot from Load flash 01:Boot from SRAM
14	BOOT0			
13:3	Reserved	-	-	-
2:0	BOOT_SIZE [2:0]	RW	3'h0	Select Main flash area as Load flash 000: No Load flash area 001:1 KB (0x0800 5C00-0x0800 5FFF) 010:2 KB (0x0800 5800-0x0800 5FFF) 011:3 KB (0x0800 5400-0x0800 5FFF) 1xx:4 KB (0x0800 5000-0x0800 5FFF)

4.8.9. Flash WRP address register (FLASH_WRPR)

Address offset: 0x2C

Reset value: 0x0000 003F

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

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	-
5:0	WRP	RW	6'h3F	0: sector [y] is protected 1:sector[y] is unprotected y=0-5

4.8.10. 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[7:0]								Res	Res	Res	Res	Res	Res	Res	SLEEP_EN
RW	RW	RW	RW	RW	RW	RW	RW	-							RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:8	SLEEP_TIME	RW	9'h64	FLASH sleep time count (counter based on HSI_10M clock) 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 sleeping 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 0x28.
7:1	Reserved	-	-	-
0	SLEEP_EN	RW	0	Flash Sleep mode enabled 1: Flash Sleep enabled 0: Flash Sleep disabled

4.8.11. 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	-	-	-
8:0	TS0	RW	9'hB4	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

4.8.12. 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									
-						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	-
9:0	TS1	RW	10'h1B0	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

4.8.13. 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	-	-	-
8:0	TS2P	RW	9'hB4	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 0120 48 MHz calibration value storage address: 0x1FFF 0134

4.8.14. 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	-	-	-
11/0	TPS3	RW	12'h6C0	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.

Bit	Name	R/W	Reset Value	Function
				Saved in the following address in Flash: 24 MHz calibration value storage address: 0x1FFF 0120 48 MHz calibration value storage address: 0x1FFF 0134

4.8.15. 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	-	-	-
8:0	TS3	RW	9'hB4	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

4.8.16. 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	-	-	-
17:0	PERTPE	RW	18′ h1 4820	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

4.8.17. 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	-	-	-

Bit	Name	R/W	Reset Value	Function
17:0	SMERTPE	RW	18'h14820	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 0128 48 MHz calibration value storage address: 0x1FFF 013C

4.8.18. 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	-	-	-
15:0	PRGTPE	RW	16'h5DC0	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

4.8.19. 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	14'h12C0	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

5.1.1. Power supply overview

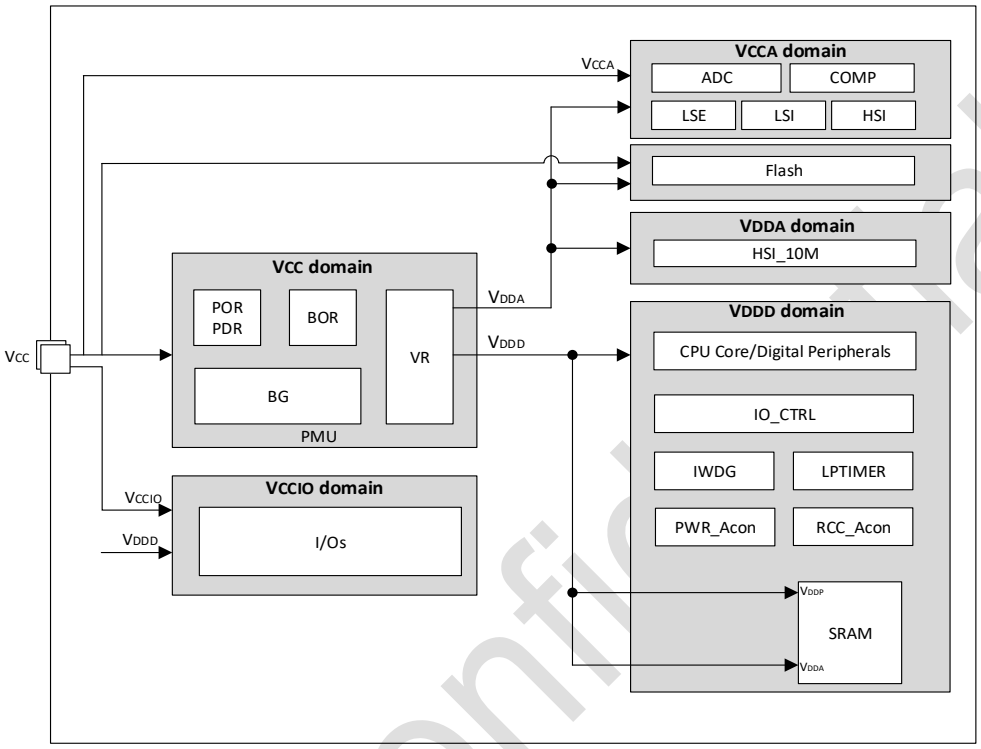

Figure 5-1 Power supply overview

Table 5-1 Power supply overview

No.	Power supply	Power value	Description
1	V_{CC}	1.7 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}	1.7 to 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}	1.7 to 5.5 V	Power to IO from V_{CC} PAD

5.2. Voltage regulator

The device has three voltage regulators:

- MR (Main regulator) is used in Run mode.
- LPR (Low power regulator) provides lower power consumption options in Stop mode.
- DLPR (Deep low power regulator) ensures the lowest power consumption in low power mode.

In Run mode, the MR remains working, outputting 1.2 V voltage and the LPR is off.

In Stop mode, power can be supplied from MR or LPR at the discretion of the software.

5.3. Power monitoring

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

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

In addition to POR/ PDR, BOR (Brown-out reset) is also implemented.

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.

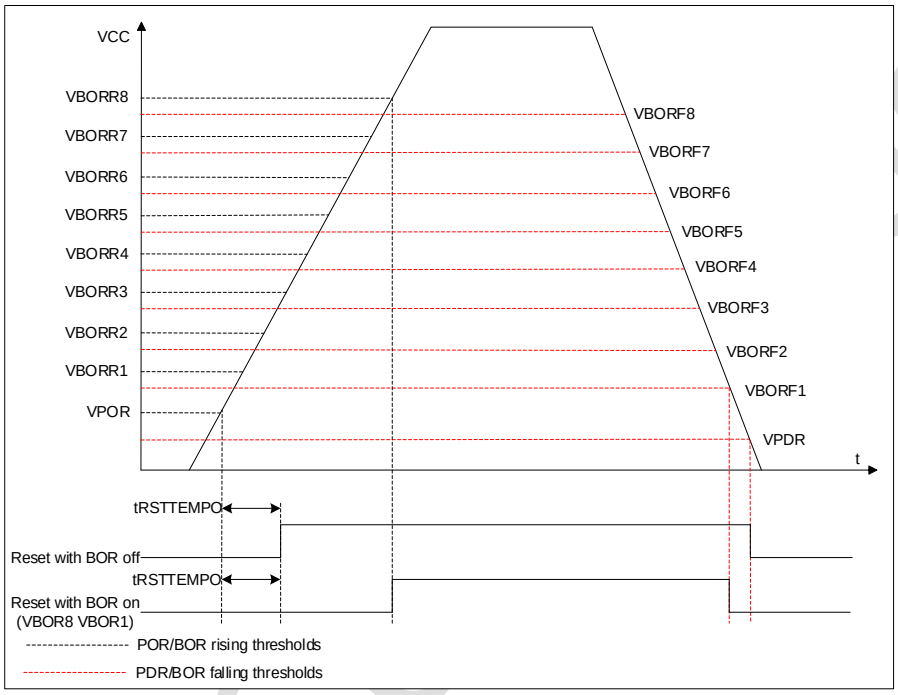

Figure 5-2 POR,PDR,and BOR thresholds

6. Low power control

By default, the microcontroller is in Run mode after a system or a power Reset. Low-power modes are available to save power when the CPU does not need to be kept running.

6.1. Low-power modes

6.1.1. Introduction

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

- **Sleep mode**: Peripherals can be configured to keep working when the CPU clock is off (NVIC, SysTick, etc.). It is recommended only to enable the modules that must work, and close the module after the module works.
- **Stop mode:** the contents of SRAM and registers are maintained and HSI is turned off.
- **Deep_stop mode**: This mode is the same as the Stop mode, but requires a longer wake-up time. GPIO, IWDG (clock source is LSE), NRST, LPTIM (clock source is LSE) may wake up the Deep_stop mode.

In Stop mode, the LSI, LSE and LPTIM can be kept running. For details of the working conditions of each module in this mode, refer to the table below.

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

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

- Reduce the system clock frequency
- Turn off the idle peripheral clock

Table 6-1 Low power modes

Mode	Entry	Wake-up source	Wake-up clock	Effect	LDO	
					MR	LPR
Sleep (sleep-now or sleep-on-exit)	WFI or return from ISR	Any interrupt	Same as before entering Sleep mode	CPU clock OFF. no effect on other clocks or analog clock sources	ON(1)	OFF
	WFE	Wake up event				
Stop/ Deep_stop	SLEEPDEEP bit 1.WFI 2. Return from ISR 3.WFE Note: LSI and LSE cannot be selected as the system clock	Any wake up EXTI line (configured in the EXTI registers), IWDG and NRST	HSI maintains the frequency configuration before entering Stop mode, without frequency division.	HSI OFF. LSI and LSE can be selected as ON/OFF. LPTIM, IWDG: configured by the software Low power wake-up and some modules such as RCC keep working. The clock is off for the remaining modules.	OFF/ON	ON

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

6.1.2. Functionalities depending on the working mode

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

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wake up capability
CPU	Y	-	-	-
Flash	Y	Y	-(2)	-
SRAM	Y	O(3)	-(4)	-

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wake up capability
Brown-out reset (BOR)	Y	Y	O	O
HSI	O	O	-	-
LSI	O	O	O	-
LSE	O	O	O	-
LSE Clock Security System (CSS)	O	O	O	O
USART1	O	O	-	-
I2C	O	O	-	-
SPI1	O	O	-	-
ADC	O	O	-	-
COMP1/COMP2	O	O	O	-
Temperature sensor	O	O	-	-
Timers (TIM1 / TIM14)	O	O	-	-
LPTIM	O	O	O	O
IWDG	O	O	O	O
SysTick timer	O	O	-	-
CRC	O	O	-	-
GPIOs	O	O	O	O

Note 1: Legend: Y = Yes (Enable). O = Optional (Disable by default. Can be enabled by software). - = Not available.

Note 2: Flash does not power down, but no clock is provided, and enters the lowest power consumption state.

Note 3: The SRAM clock can be gated on or off.

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

6.2. Sleep mode

6.2.1. Entering Sleep mode

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

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

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

6.2.2. Exiting Sleep mode

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

If the WFE instruction is used to enter Sleep mode, the MCU exits Sleep mode as soon as an event occurs. The wake up event can be generated either by:

- enabling an interrupt in the peripheral control register but not in the NVIC, and enabling the SEVON-PEND bit in the Cortex®-M0+ System Control register. When the MCU resumes from WFE, the periph-eral interrupt pending bit has to be cleared.
- configuring an external or internal EXTI line in event mode. When the CPU resumes from WFE, it is not necessary to clear the peripheral interrupt pending bit. This mode offers the lowest wake up time as no time is wasted in interrupt entry/exit.

Table 6-3 Sleep-now

Sleep-now	Description
Mode entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 and - SLEEPONEXIT = 0
Mode exit	If WFI was used for entry: Interrupt If WFE was used for entry: Wake up event
Wakeup latency	None

Table6-4 Sleep-on-exit

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

6.3. Stop mode

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

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

6.3.1. Entering Stop mode

To further reduce power consumption in Stop mode, you can enter different Stop modes by configuring PWR_CR1. LPR.

If Flash memory programming is ongoing, the Stop mode entry is delayed until the memory access is finished (the software reads the BSY bit of the Flash_SR register to determine whether the erase and write operation has finished).

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

6.3.2. Exiting Stop mode

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

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

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

Table 6-5 Stop mode

Stop mode	Description
Mode entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Settings: 1） Select the voltage regulator mode by configuring LPR bit in PWR_CR1 2） Set the wake-up time by configuring FLS_SLPTIME bit in PWR_CR1 - Set SLEEPDEEP bit in Cortex®-M0+ System Control register Notes:

Stop mode	Description
	To enter Stop mode, all EXTI Line pending bits (in Pending register EXTI_PR), all peripherals interrupt pending bits and RTC Alarm flag must be reset. Otherwise, the Stop mode entry procedure is ignored and program execution continues. In order to make the change of chip power consumption as balanced as possible, the software needs to follow the principle of step-by-step shutdown: step-by-step shutdown of the clock of each module and select HSI as the system clock. In order to shorten the wake-up time, before entering the stop mode, the system clock should be configured to select HSI high-frequency clock, and the HPRE of RCC_CFGR register is set to 0, otherwise the hardware switching clock will consume additional clock cycles after waking up.
Mode exit	If WFI was used for entry: - Any EXTI Line configured in Interrupt mode - If WFE was used for entry: - Any EXTI Line configured in Interrupt mode - Interrupt flag bit in case of CPU SEVONPEND position bit
Wakeup latency	LPR to MR wakeup time HSI wakeup time Flash wakeup time

6.4. Reduce the system clock frequency

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

6.5. Peripheral clock gating

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

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

6.6. Power control registers

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

6.6.1. Power control register 1 (PWR_CR1)

Address offset:0x00

Reset value: 0x0002 0000 (reset by POR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												HSION_CTRL	SRAM_RETV_DLP	SRAM_RET	Res
-												RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPR		FLS_SLPTIME[1:0]		Res											
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	SRAM_RETV_DLP	RW	1	SRAM retention voltage control in Stop mode 1: SRAM voltage is consistent with digital LDO output. 0: SRAM voltage is low voltage
17	SRAM_RETV	RW	1	SRAM retention voltage control in Stop mode 1: SRAM voltage is consistent with digital LDO output. 0: SRAM voltage is low voltage
16	Reserved	-	-	-

Bit	Name	R/W	Reset Value	Function
15:14	LPR	-	0	Low power regulator LPR=2'b00, Stop mode powered by MR LPR =2'b01, Stop mode powered by LPR LPR =2'b01, Deep_stop mode powered by DLPR LPR =2'b11，reserved
13:12	FLS_SLPTIME	RW	2′ b00	In the Stop mode wake-up timing, after the HSI is stabilized, a waiting time is required before the Flash operation. 2'b00: 5 us 2'b01: 2 us 2'b10: 3 us 2'b11: 0 us In the Stop mode wake-up timing, after the HSI is stabilized, a waiting time is required before the Flash operation. 2'b00: 5 us 2'b01: 2 us 2'b10: 3 us 2'b11: 0 us Note: When this register is set to 2'b11/2'b01, it indicates that the program is executed from SRAM after wakeup, not Flash. And the program guarantees that Flash will not be accessed within the specified time above after waking up the execution program.
11:0	Reserved	-	-	Reserved

7. Reset

Two resets are designed: power reset and system reset.

7.1. Reset source

7.1.1. Power reset

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

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

7.1.2. System reset

A system reset sets all registers to their reset values except the reset flags in the clock control/status register and the registers in the RTC domain.

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

- NRST pin (external reset)
- Independent watchdog reset (IWDG)
- SYSRESETREQ software reset
- Option byte loader reset
- Power reset (POR/PDR, BOR)

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

7.1.3. NRST pin (external reset)

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

- Reset input

In this mode, any valid reset signal on the NRST pin is propagated to device internal logic, but resets generated internally by the device are not visible on the pin.

In this mode, the GPIO functionality (PC0) is not available.

The pulse generator guarantees a minimum reset pulse duration of 40 μs for each internal reset source to be output on the NRST pin.

- GPIO

In this mode, the pin can be used as PC10 standard GPIO. The reset function of the pin is not available. Reset is only possible from device internal reset sources, and it is not propagated to the pin.

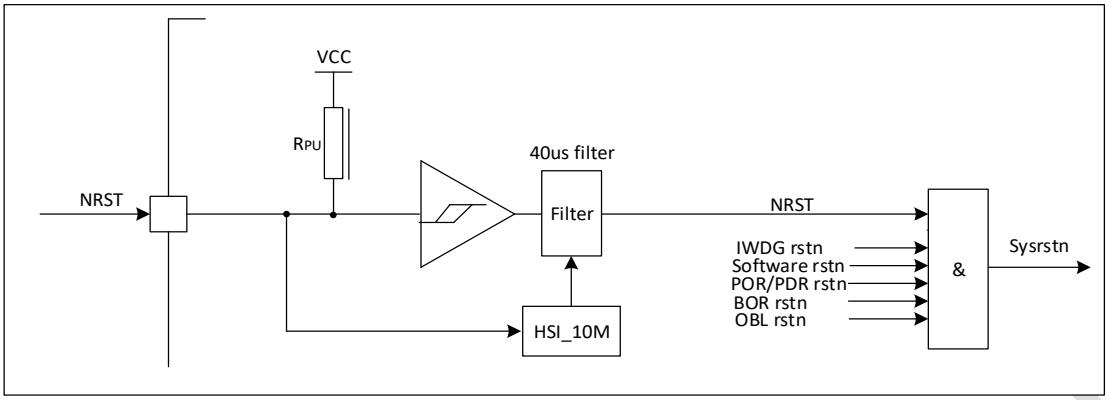

Figure 7-1 Simplified diagram of the reset circuit

7.1.4. Watchdog reset

Refer to Independent watchdog (IWDG).

7.1.5. Software reset

Software reset can be achieved by setting the SYSRESETREQ bit of the ARM M0+ interrupt and reset control register.

7.1.6. Option byte loader reset

The option byte loader reset is generated when the OBL_LAUNCH bit is set in the FLASH_CR register.

8. Clocks

8.1. Clock sources

8.1.1. External high speed clock (HSE bypass)

- The external clock is input by PA6.
- PA6 automatically enables the input of the IO when the external clock signal is enabled. And PA6 is prohibited from being used as GPIO.

8.1.2. External low speed clock LSE

The LSE crystal is a low speed external crystal or ceramic resonator at 32.768 kHz. It provides a real-time clock or other timing function

A low power and accurate clock source.

The LSE crystal is turned on and off by the LSEON bit in the backup domain control register (RCC_BDCR), and the driving capability can be adjusted by LSE_DRIVER [1: 0].

LSERDY in the backup domain control register (RCC_BDCR) indicates whether the LSE crystal oscillation is stable. During the startup phase, the LSE clock signal is not released until this bit is set to 1 by hardware. If allowed in the clock interrupt register, an interrupt request may be generated.

8.1.3. External clock source (LSE bypass)

In this mode, an external clock source with a frequency of 32.768 kHz must be provided. You can select this mode by setting the LSEBYP and LSEON bits in the backup domain control register (RCC_BDCR). An external clock signal with a 50% duty cycle must be connected to the OSC32_IN pin while keeping the OSC32_OUT pin floating.

8.1.4. Internal high-speed clock HSI

The internal high-speed clock serves as the most critical source of the chip’s system clock. After power-up, the default system clock frequency is 24 MHz, which can be switched to 48 MHz by software configuration.

8.1.5. Internal low speed clock LSI

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

8.2. Clock tree

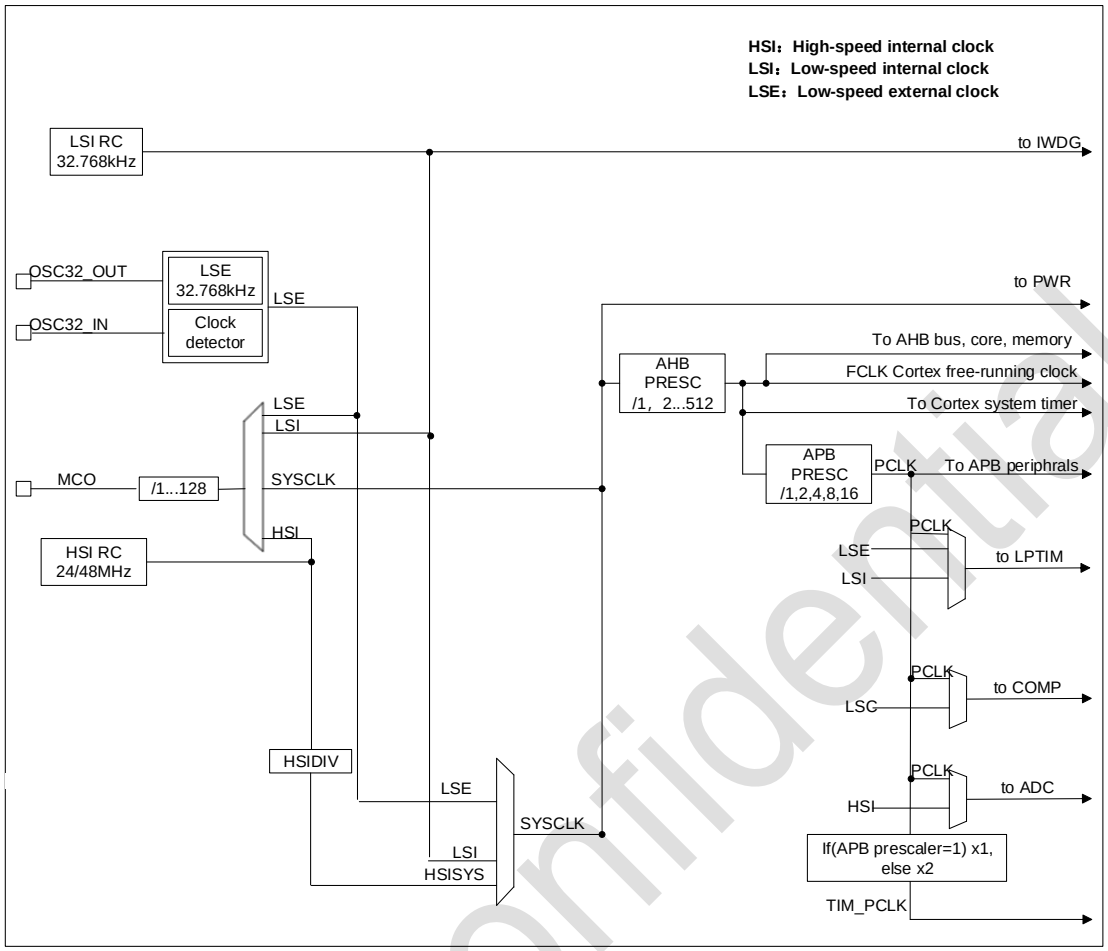

Figure 8-1 System clock structure diagram

8.3. Clock security system (CSS)

Clock security system can be activated by software. In this case, after the start-up delay of the LSE, the clock detection function is turned on. When this LSE is turned off, the clock detection function is turned off.

If a failure is detected on the LSE clock, the LSE oscillator is automatically disabled, a clock failure event is sent to the break input of the advanced-control timers (TIM1) and an interrupt is generated to inform the software about the failure (clock security system interrupt, CSSI), allowing the MCU to perform rescue operations. The CSSI is linked to the NMI (Non-maskable interrupt) of Cortex-M0 +.

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

If the LSE oscillator is used directly or indirectly as the system clock, a detected failure causes a switch of the system clock to the LSI oscillator and the disabling of the LSE oscillator.

8.4. Output clock

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

Table 8-1 Output clock selection

Clock sources	MCO output clock source
HSI	√
SYSCLK	√
HSE (bypass)	√
LSE	√
LSI	√

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

8.5. Clock calibration

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

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

8.5.1. HSI Calibration

8.5.1.1. Clock measurement

The basic principle is based on the ratio of clock sources, and the accuracy is closely related to the ratio of two clock sources. The larger the ratio, the better the measurement.

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

The HSI oscillator has dedicated user-accessible calibration bits for this purpose.

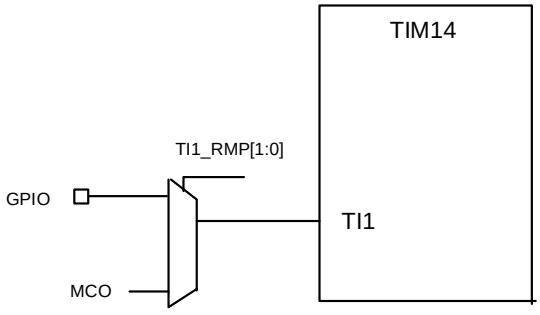

Figure 8-2 Frequency measurement vs. TIM14 capture mode

The input capture channel of the TIM14 may be a GPIO or an in-chip clock. The selection of these clocks is achieved through the TI1_RMP [1: 0] register of TIM14_OR. The 2 options are shown as follows:

- TIM14 channel 1 is connected to GPIO
- TIM14 channel 1 is connected to MCO (Microcontroller clock output)

8.5.1.2. Clock calibration

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

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

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

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

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

8.5.2. LSI calibration

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

The calibration of the LSI is to connect the output of the LSI and the input capture of the TIM 14

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

8.6. Reset/clock register

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

8.6.1. Clock control register (RCC_CR)

Address offset: 0x00

Reset value: 0x0000 0100

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

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	-
18	HSEEN	RW	0	HSE clock enable

Bit	Name	R/W	Reset Value	Function
				1.HSE bypass enabled 2.HSE bypass disabled
17:14	Reserved	-	-	-
13:11	HSIDIV[2:0]	RW	3'h0	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	HSE 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	-	-	-
8	HSION	RW	1	HSI clock enable Set and cleared by software. Cleared by hardware to stop the HSI oscillator when entering Stop mode. This bit is set by hardware if the HSI is used directly or indirectly as system clock. 0: HSI OFF 1:HSI ON
7:0	Reserved	-	-	-

8.6.2. Internal clock sources calibration register (RCC_ICSCR)

Address offset: 0x04

Reset value: 0x00FF 10FF, by POR/BOR reset

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	LSI_STARTUP		Res	LSI_TRIM[8:0]								
-	-	-	-	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
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
31:28	Reserved	-	-	-
27:26	LSI_STARTUP	RW	2'h0	Internal low speed clock LSI stabilization time selection: 11: 256 LSI clock cycles 10: 64 LSI clock cycles 01: 16 LSI clock cycles 00: 4 LSI clock cycles
25	Reserved	-	-	-
24:16	LSI_TRIM	RW	9'hFF	Internal low-speed clock frequency adjustment, through calibration, the internal low-speed clock can output 32.768 kHz. After power-on, the device will write the factory information (stored in 0x1FFF 0144) into this register to achieve calibration at the specific output frequency of the LSI. The calibration values are saved in the following address in Flash: 32.768 kHz calibration address: 0x1FFF 0144 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'h000	HSI frequency: 000: reserved 001: reserved 100:24 MHz

Bit	Name	R/W	Reset Value	Function
				101:48 MHz Others: Reserved When powered on, 24 MHz is selected by default.
12:0	HSI_TRIM	RW	13'h10FF	Clock frequency calibration value. After power-on, the hardware uses the default calibration value of HSI 24MHz. When calibrated, the factory information (stored in 0x1FFF 0100) will be written to this register. The software reads out the data stored at the corresponding address in the information area and writes it to the register to realize the calibration at the specific output frequency of HSI. Saved in the following address in Flash: 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%.

8.6.3. Clock configuration register (RCC_CFGR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	MCOPRE[2:0]			Res	MCOSEL[2:0]			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	-	-	-
30:28	MCOPRE[2:0]	RW	3'h0	MCO (Microcontroller clock output) frequency division factor. The software controls these bits and sets the frequency division factor of the MCO output: 000:1 001:2 010:4 011:8 100:16 101:32 110:64 111:128 It is recommended to set these bits before the MCO output is enabled.
27	Reserved	-	-	-
26:24	MCOSEL[2:0]	RW	3'h0	MCO Selection 000: No clock, MCO output is not enabled 001: SYSCLK 010: Reserved 011: HSI 100: HSE 101: Reserved 110: LSI 111: LSE Note: The output clock may be incomplete during the clock startup or switching phase.
23:15	Reserved	-	-	-
14:12	PPRE[2:0]	RW	3'h0	This bit is controlled by software. To generate the PCLK clock, it sets the division coefficients of HCLK as follows: 0xx:1 100:2 101:4 110:8 111:16
11:8	HPRE[3:0]	RW	3'h0	AHB clock division factor.

Bit	Name	R/W	Reset Value	Function
				The software controls this bit. To generate the HCLK clock, it sets the division coefficients of SYSCLK as follows: 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	-	-	-
5:3	SWS[2:0]	R	3'h0	System clock switching status bit These bits are controlled by hardware and indicate which clock source is currently being used as the system clock: 000: HSISYS 001: HSE 010: Reserved 011: LSI 100: LSE Others: Reserved
2:0	SW[2:0]	RW	3'h0	System clock source select bit. These bits are controlled by software and hardware and are used to select the system clock: 000: HSISYS 001: HSE 010: Reserved 011: LSI 100: LSE Others: Reserved Scenarios where the hardware is configured as HSISYS include: 1) The system exits from Stop mode

8.6.4. External clock sources control register (RCC_ECSCR)

Address offset: 0x10

Reset value: 0x0001 0000

31	30	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	Res	Res		Res	
-															

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	-
21:20	LSE_STARTUP	RW	2'h0	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	-	-	-
17:16	LSE_DRIVER	RW	2'h1	Low-speed crystal oscillator driving capability selection.

				00: Weakest driving capability 01: Weak driving ability 10: Default driving capability (Recommended). 11: Strongest driving ability. 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.
15:0	Reserved	-	-	-

8.6.5. 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	Res	HSI RDYIE	Res	LSE RDYIE	LSI RDYIE
-												RW	-	RW	RW

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

8.6.6. 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	Res	Res	Res	Res	Res	HSI RDYF	Res	LSE RDYF	LSI RDYF
-						R	-					R	-	R	R

Bit	Name	R/W	Reset Value	Function
31:10	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.
8:4	Reserved	-	-	-
3	HSIRDYF	R	0	HSI ready interrupt flag Set by hardware when the HSI clock becomes stable and HSIRDYIE is enabled. Cleared by software setting the HSIRDYC bit. 0: No clock ready interrupt caused by the HSI oscillator 1: Clock ready interrupt caused by the HSI oscillator
2	Reserved	-	-	-
1	LSERDYF	R	0	LSE ready interrupt flag

Bit	Name	R/W	Reset Value	Function
				Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSERDYC bit. 0: No clock ready interrupt caused by the LSE oscillator 1: Clock ready interrupt caused by the LSE oscillator
0	LSIRDYF	R	0	LSI ready interrupt flag Set by hardware when the LSE clock becomes stable and LSERDYIE is enabled. Cleared by software setting the LSERDYC bit. 0: No clock ready interrupt caused by the LSI oscillator 1: Clock ready interrupt caused by the LSI oscillator

8.6.7. 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	Res	Res	Res	Res	Res	HSI RDYC	Res	LSE RDYC	LSI RDYC
-						W	-					W	-	W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	-
9	LSECSSC	W	0	LSE clock security system interrupt flag. 0: No effect. 1: Clear LSECSSF flag
8:4	Reserved	-	-	-
3	HSIRDYC	W	0	HSE ready interrupt clear 0: No effect. 1: Clear HSIRDYE bit.
2	Reserved	-	-	-
1	LSERDYC	W	0	LSE ready interrupt clear. 0: No effect. 1: Clear LSERDYE bit.
0	LSIRDYC	W	0	LSI ready interrupt clear. 0: No effect. 1: Clear LSIRDYE bit.

8.6.8. 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	Res	Res	Res	GPIOC RST	GPIOB RST	GPIOA RST
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	-
2	GPIOCRST	RW	0	I/O PortC reset. 0: No effect. 1: PortC I/O reset
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.

Bit	Name	R/W	Reset Value	Function
				1: I/O PortA reset

8.6.9. 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	FLASH RST	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	RW	-	-	-	RW	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	-
12	CRCRST	RW	0	CRC reset 0: No effect. 1: CRC reset
11:9	Reserved	-	-	-
8	FLASHRST	RW	0	Flash memory interface reset 0: No effect. 1: Reset Flash memory interface.
7:0	Reserved	-	-	-

8.6.10. 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
LPTIM RST	Res	Res	PWR RST	DBG RST	Res	Res	Res	Res	Res	I2C RST	Res	Res	Res	Res	Res
RW	-	-	RW	RW	-	-	-	-	-	RW	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31	LPTIMRST	RW	0	Low power timer reset 0: No effect. 1: Reset
30:29	Reserved	-	-	-
28	PWRRST	RW	0	Power interface reset 0: No effect. 1: Reset
27	DBGRST	RW	0	DBG reset 0: No effect. 1: Reset
26:22	Reserved	-	-	-
21	I2CRST	RW	0	I^2C1 reset. 0: No effect. 1: Reset
20:0	Reserved	-	-	-

8.6.11. 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	Res	Res	Res	COMP2 RST	COMP1 RST	ADC RST	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	RW	RW	RW	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

TIM14 RST	USART1 RST	Res	SPI1 RST	TIM1 RST	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SYS CFG RST
RW	RW	-	RW	RW	-	-	-	-	-	-	-	-	-	-	RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	-
22	COMP2RST	RW	0	COMP2 reset 0: No effect. 1: Reset
21	COMP1RST	RW	0	COMP1 reset 0: No effect. 1: Reset
20	ADCRST	RW	0	ADC reset. 0: No effect. 1: Reset
19:16	Reserved	-	-	-
15	TIM14RST	RW	0	TIM14 reset. 0: No effect. 1: Reset
14	USART1RST	RW	0	USART1 reset 0: No effect. 1: Reset
13	Reserved	-	-	-
12	SPI1RST	RW	0	SPI1 reset 0: No effect. 1: Reset
11	TIM1RST	RW	0	TIM1 reset 0: No effect. 1: Reset
10:1	Reserved	-	-	-
0	SYSCFGRST	RW	0	SYSCFG reset 0: No effect. 1: Reset

8.6.12. 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															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res													GPIOCEN	GPIOBEN	GPIOAEN
-													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	-
2	GPIOCEN	RW	0	IO port C clock enable 0: clock disabled. 1: clock enabled.
1	GPIOBEN	RW	0	IO port B clock enable 0: clock disabled. 1: clock enabled.
0	GPIOAEN	RW	0	IO port A clock enable 0: clock disabled. 1: clock enabled.

8.6.13. 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	CRC EN	Res	Res	SRAMEN	FLASH EN	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	RW	-	-	RW	RW	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	-
12	CRCEN	RW	0	CRC clock enable 0: Disabled 1: Enabled
11:10	Reserved	-	-	-
9	SRAMEN	RW	1	In Sleep mode, the clock enable control of the SRAM 0: The clock is disabled in Sleep mode 1: The clock is enabled in Sleep mode Note: This bit only affects the clock enable of this module in Sleep mode. In Run mode, the clock of this module will not be turned off
8	FLASHEN	RW	1	In Sleep mode, Flash clock enables control 0: The clock is disabled in Sleep mode 1: The clock is enabled in Sleep mode Note: This bit only affects the clock enable of this module in Sleep mode. In Run mode, the clock of this module will not be turned off
7:0	Reserved	-	-	-

8.6.14. 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
LPTIM EN	Res	Res	PWR EN	DBG EN	Res	Res	Res	Res	Res	I2C EN	Res	Res	Res	Res	Res
RW	-	-	RW	RW	-	-	-	-	-	RW	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Bit	Name	R/W	Reset Value	Function
31	LPTIMEN	RW	0	LPTIM1 enable 0: Disabled 1: Enabled
30:29	Reserved	-	-	-
28	PWREN	RW	0	Power interface clock enable 0: Disabled 1: Enabled
27	DBGEN	RW	0	DBG enable 0: Disabled 1: Enabled
26:22	Reserved	-	-	-
21	I2CEN	RW	0	I2C1 enable 0: Disabled 1: Enabled
20:0	Reserved	-	-	-

8.6.15. APB peripheral clock enable register 2(RCC_APBENR2)

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	COMP2 EN	COMP1 EN	ADC EN	Res	Res	Res	Res
-									RW	RW	RW	-			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14 EN	USART1 EN	Res	SPI1 EN	TIM1 EN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SYS CFG EN

RW	RW	-	RW	RW	-	RW

Bit	Name	R/W	Reset Value	Function
31:23	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	-	-	-
15	TIM14EN	RW	0	TIM14 enable 0: Disabled 1: Enabled
14	USART1EN	RW	0	USART1 enable 0: Disabled 1: Enabled
13	Reserved	-	-	-
12	SPI1EN	RW	0	SPI1 enable 0: Disabled 1: Enabled
11	TIM1EN	RW	0	TIM1 enable 0: Disabled 1: Enabled
10:1	Reserved	-	-	-
0	SYSCFGEN	RW	0	SYSCFG clock enable 0: Disabled 1: Enabled

8.6.16. Peripherals independent clock configuration register (RCC_CCIPR)

Address offset: 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	LPTIM1SEL [1:0]		Res	Res
-												RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		Res		COMP2 SEL	CO MP1 SEL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-				RW	RW	-									

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	-
19:18	LPTIMSEL[1:0]	RW	2'h0	LPTIM1 clock source selection 00: PCLK 01: LSI 10: No clock 11: LSE Note: When PCLK is selected, the prescaler of the LPTIM (Low-Power Timer) must be configured to 2 or higher (set by LPTIM_CFGR.PRESC).
17:12	Reserved	-	-	-
11	COMP2SEL	RW	0	COMP2 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: Configure selection clock before enabling COMP2_FR.FLTEN.
10	COMP1SEL	RW	0	COMP1 clock source selection 0: PCLK 1: LSC (clock selected by RCC_BDCR.LSCOSEL) Note: Configure selection clock before enabling COMP1_FR.FLTEN.
9:0	Reserved	-	-	-

8.6.17. RCC domain control register (RCC_BDCR)

Address offset: 0x5C

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	LSCO SEL	Res	Res	Res	Res	Res	Res	Res	Res	Res
-						RW	-								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res		Res	LSECSSD	LSECS-SON	Res		LSE BYP	LSE RDY	LSE ON
-									R	RW	-		RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	-
25	LSCOSEL	RW	0	Low speed clock output selection 0: LSI 1: LSE
24:7	Reserved	-	-	-
6	LSECSSD	R	0	CSS on LSE failure Detection 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 LSE
5	LSECSSON	RW	0	CSS on LSE enable 0: Disabled 1: Enabled LSECSSON must be enabled after the LSE oscillator is enabled (LSEON=1) and ready (LSERDY=1). Once enabled this bit cannot be disabled, except after an LSE failure detection (LSECSSD=1).
4:3	Reserved	-	-	-
2	LSEBYP	RW	0	LSE oscillator bypass 0: LSE oscillator not bypassed 1: LSE oscillator bypassed 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 oscillator ready Set and cleared by hardware to indicate that the LSE oscillator is stable. 0: Not ready 1: Ready
0	LSEON	RW	0	LSE oscillator enable 0: Disabled 1: Enabled

8.6.18. Control/status register (RCC_CSR)

Address offset: 0x60

Reset value: 0x0000 0000

Reset by:1) [29:25]: POR reset: 2) LSION: system reset. 3) NRST_FLTIDS will not be reset by the system

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	IWDG RSTF	SFT RSTF	PWR 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	Res	NRST_FLT-IDIS	Res	Res	Res	Res	Res	Res	LSI RDY	LSION
-							RW	-						R	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29	IWDGRSTF	R	0	Independent watchdog reset flag Cleared by writing to the RMVF bit.

Bit	Name	R/W	Reset Value	Function
28	SFTRSTF	R	0	Software reset flag Cleared by writing to the RMVF bit.
27	PWRRSTF	R	0	BOR/POR/PDR reset flag Cleared by writing to the RMVF bit.
26	PINRSTF	R	0	PIN reset flag Cleared by writing to the RMVF bit.
25	OBLRSTF	R	0	Option byte loader reset flag Cleared by writing to the RMVF bit.
24	Reserved	-	-	-
23	RMVF	RW	0	Set by software to clear the reset flags of [29: 25].
8	NRST_FLTDIS	RW	0	NRST filtering disabled 0: HSI_10M enabled, and filtering 40 us width function enabled 1: Filtering function is disabled and HSI_10M remains off
7:2	Reserved	-	-	-
1	LSIRDY	R	0	LSI oscillator stable flag 0: LSI oscillator unstable 1: LSI oscillator stable
0	LSION	RW	0	LSI oscillator enable 0: Disabled 1: Enabled Set and cleared by software. The hardware sets this bit when IWDG is hardware-enabled (via option byte) and LSECSSON is software-enabled.

9. General-purpose I/Os (GPIO)

9.1. Introduction

Each general-purpose I/O port has:

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

9.2. 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 states: floating, pull-up/down, analog
- Input data to input data register (GPIOx_IDR) or peripheral (alternate function input)
- Bit set and reset register (GPIOx_ BSRR) for bitwise write access to GPIOx_ODR
- Locking mechanism (GPIOx_LCKR) provided to freeze the I/O port configurations
- Analog function
- Alternate function selection registers (at most 8 AFs possible per I/O)
- Ability to flip quickly in a single cycle
- Highly flexible I/O multiplexing function, allowing the I/O port to function as GPIO or as various peripheral interface functions

9.3. GPIO functional description

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

- Input floating
- Input pull-up
- Input-pull-down
- Analog
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Alternate function push-pull with pull-up or pull-down capability
- Alternate function open-drain with pull-up or pull-down capability

Each I/O port bit is freely programmable, however the I/O port registers have to be accessed as 32-bit words, half-words or bytes. The purpose of the GPIOx_BSRR 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.

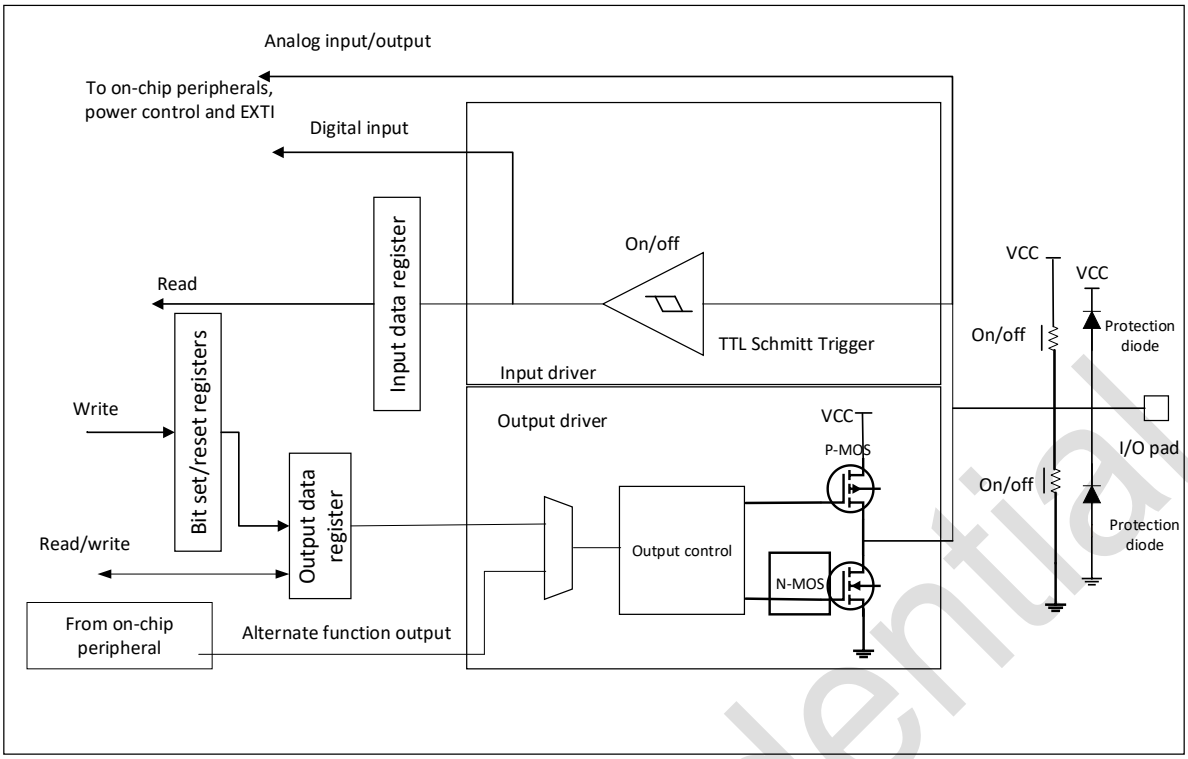

Figure 9-1 Basic structure of an I/O port bit

9.3.1. General-purpose I/Os (GPIO)

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in analog mode.

The debug pins are in AF pull-up/pull-down after reset:

1. PA2: SWCLK in pull-down
2. PB6-SWDIO: in pull-up mode. When the pin is configured as output, the value written to the output data register (GPIOx_ODR) is output on the I/O pin. It is possible to use the output 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 ornot depending on the value in the GPIOx_PUPDR register.

9.3.2. I/O pin alternate function multiplexer and mapping

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

Each I/O pin has a multiplexer with up to eight alternate function inputs (AF0 to AF7) that can be configured through the GPIOx_AFRL

(for pin 0 to 7) register.

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

- 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:
  - Connect the I/O to the desired AFx in the GPIOx_AFRL register.
  - 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 DAC connection could be enabled in ADC or DAC 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.

9.3.3. I/O port control registers

Each of the GPIO ports has four 32-bit memory-mapped control registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR) to configure up to 8 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 whatever the I/O direction.

9.3.4. I/O port data registers

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

9.3.5. 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.

9.3.6. GPIO locking mechanism

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

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 [7:0] is used to lock the configuration of the I/Os (during the write sequence the LCKR [7:0] 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, and GPIOx_AFRL.

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 [7:0] bits.

9.3.7. I/O alternate function input/output

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

This means that a number of possible peripheral functions are multiplexed on each GPIO using the GPIOx_AF 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.

9.3.8. 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.

9.3.9. Input configuration

When the I/O port is programmed as input:

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

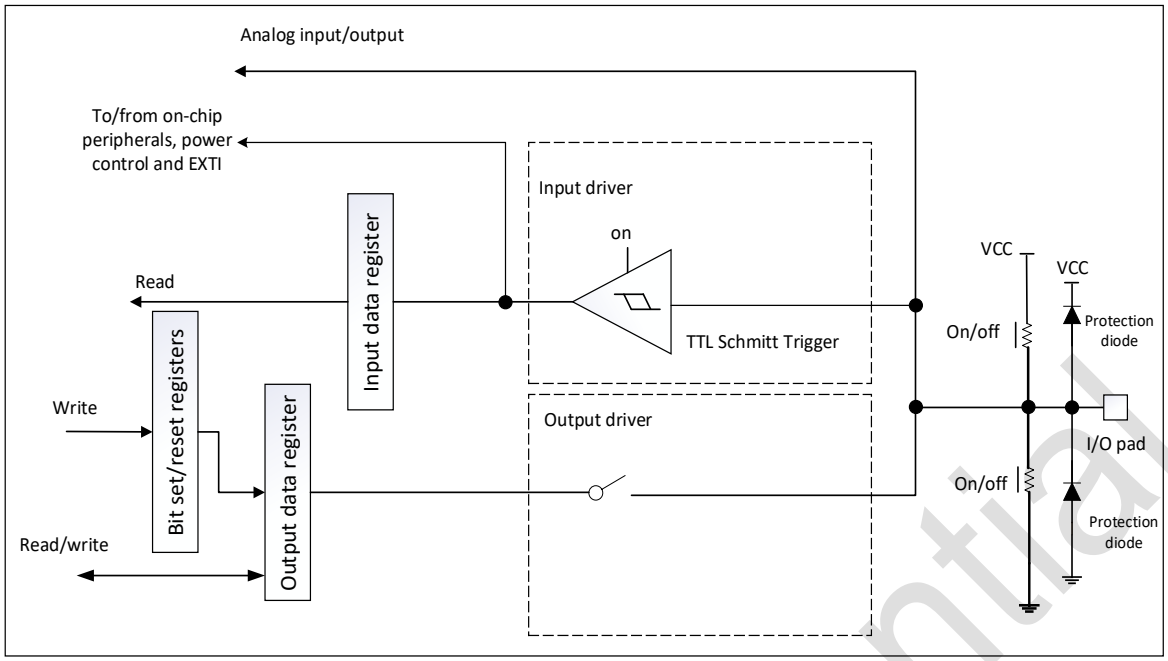

Figure 9-2 Input floating/pull up/pull down configurations

9.3.10. 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

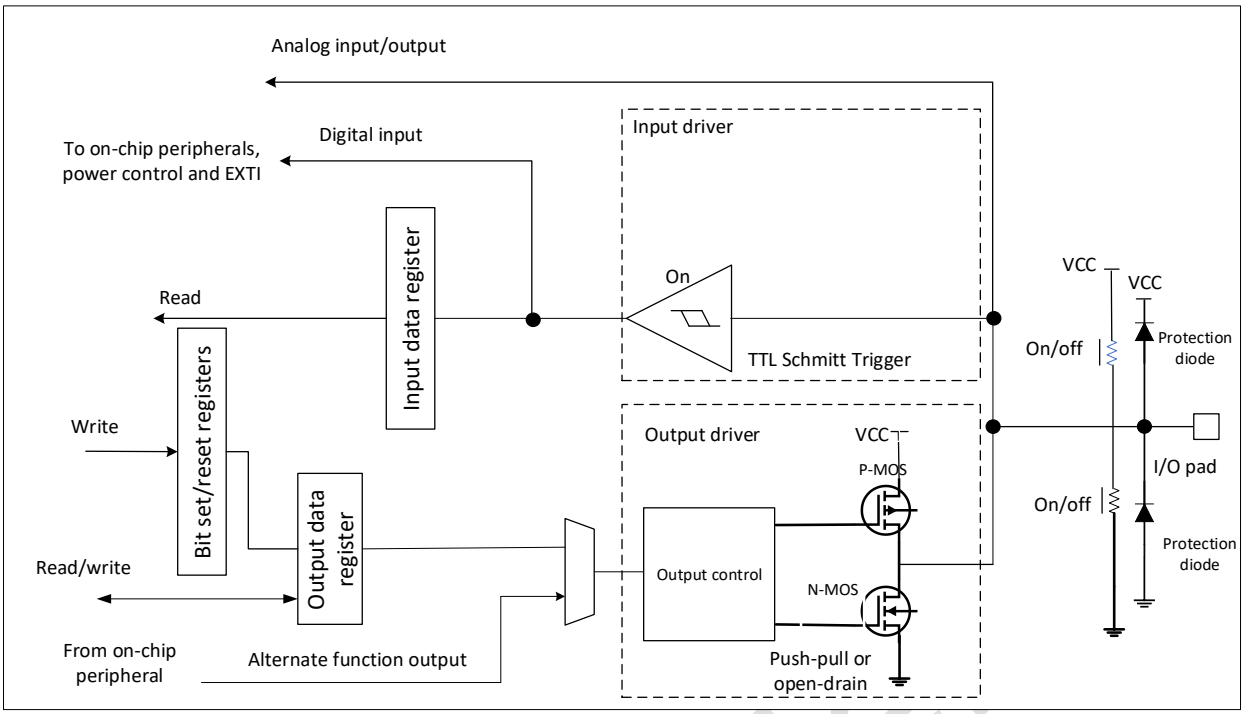

Figure 9-3 Output configuration

9.3.11. Alternate function configuration

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

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

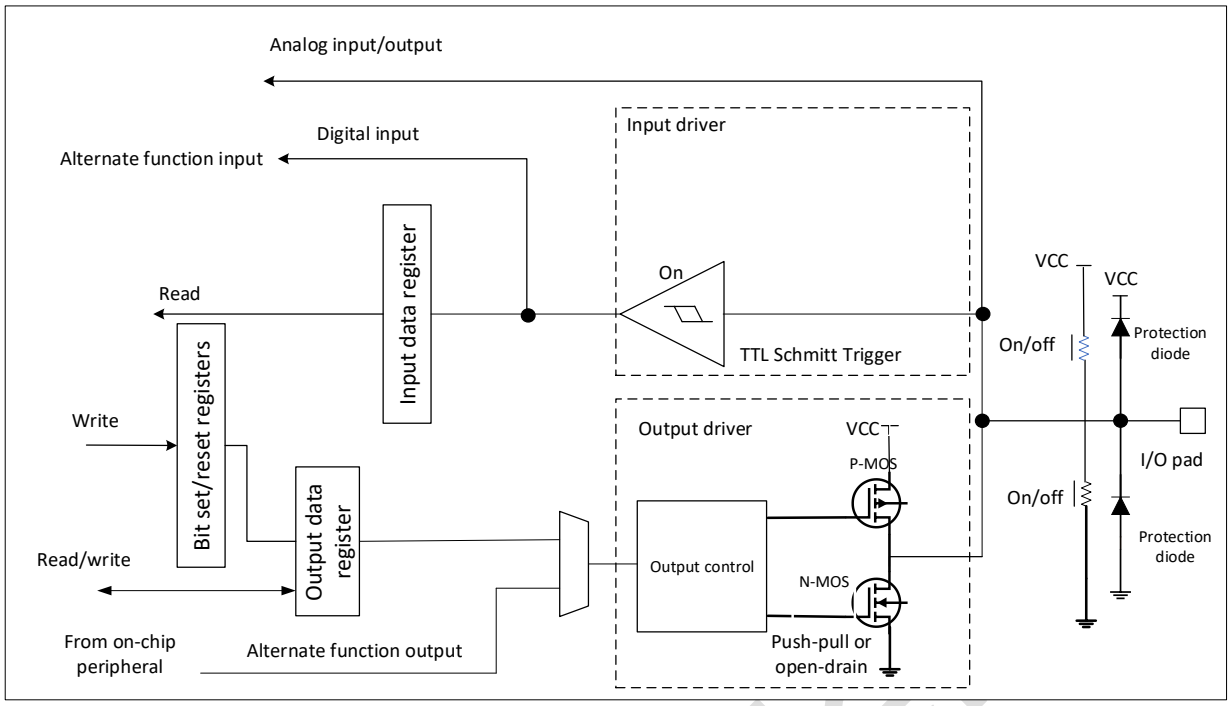

Figure 9-4 Alternate function configuration

9.3.12. Analog configuration

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

- The output buffer is disabled.
- The Schmitt trigger input is deactivated, providing zero consumption for every analog value of the I/O pin. The output of the Schmitt trigger is forced to a constant value (0).
- The weak pull-up and pull-down resistors are disabled by hardware (set GPIOx_PUPDR to 00).
- Read access to the input data register gets the value “1”.

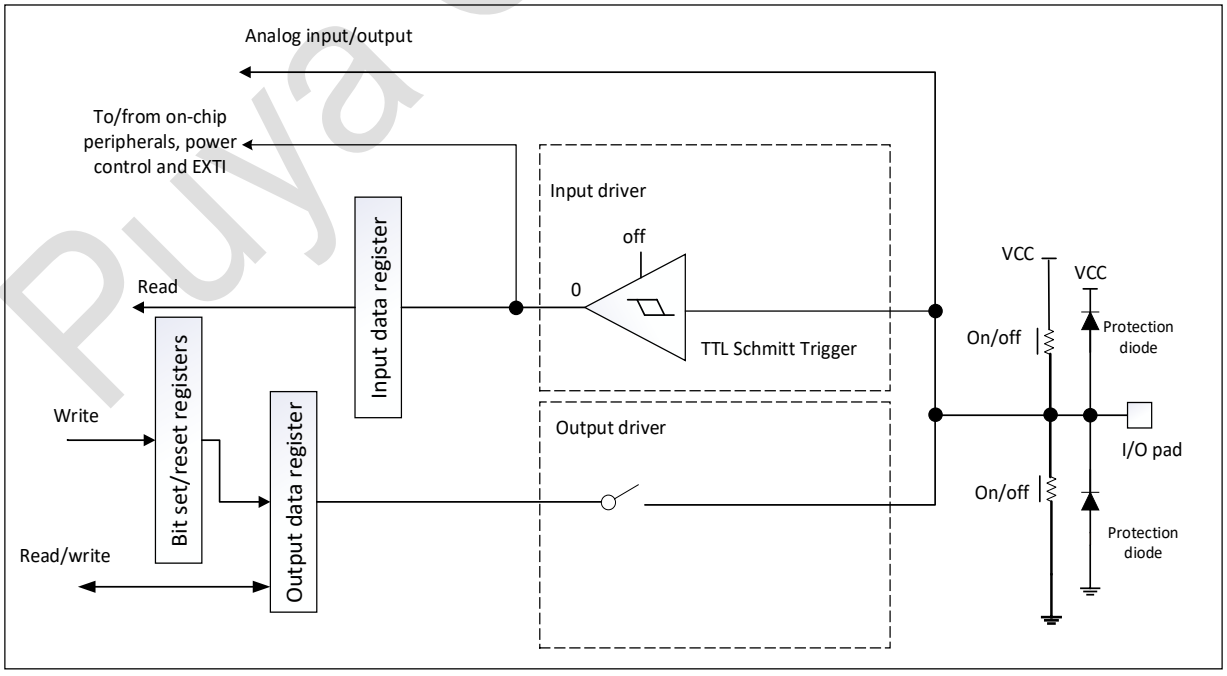

Figure 9-5 High impedance-analog configuration

9.3.13. 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_CSR 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.

9.4. GPIO registers

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

9.4.1. GPIO port mode register (GPIOx_MODER)(x =A, B, C)

Address offset:0x00

Reset value:

0x0000 FFEF for GPIOA

Reset value for GPIOB

1. Flash option byte When configuring SWD: 0x0000 FFFF
2. Flash option byte When SWD is not configured: 0x0000 EFFF

Reset value for GPIOC

1. Flash option byte When configuring SWD: 0x0000 FFFF
2. Flash option byte When SWD is not configured: 0x0000 EFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:0	MODEy[1:0]	RW	16'hFFEF(PA). 16 'hFFFF (PB, SWD configured), 16 'hEFFF (PB, SWD not configured), 16'h000E (PC, SWD configured), 16'h000F (PC, SWD not configured)	y = 7..0 These bits are written by software to configure the I/O mode. 00: Input mode 01: General purpose output mode 10: Alternate function mode 11: Analog mode (reset state)

9.4.2. GPIO port output type register (GPIOx_OTYPER) (x = A, B, C)

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	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	-
7:0	OTy	RW	8′ h0	y = 7..0 These bits are written by software to configure the I/O output type. 0: Output push-pull (reset state) 1: Output open-drain

9.4.3. GPIO port output speed register (GPIOx_OSPEEDR) (x = A, B, C)

Address offset: 0x08

Reset value:

0x0000 0000 for GPIOA

Reset value for GPIOB

1. Flash option byte When configuring SWD: 0x0000 0000
2. Flash option byte When SWD is not configured: 0x0000 3000

Reset value: 0x0000 0000 (for other ports)

Reset value for GPIOC

1. Flash option byte When configuring SWD: 0x0000 0003
2. Flash option byte When SWD is not configured: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
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
31:16	Reserved	-	-	-
15:0	OSPEEDy[1:0]	RW	16'h0(PA), 16'h0 (PB, SWD configured), 16'h3000 (PB, SWD not configured), 16'h3 (PC, SWD configured), 16'h0 (PC, SWD not configured)	y = 7..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

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

Address offset: 0x0C

Reset value:

0x0000 0020(for port A)

Reset value for GPIOB

- Flash option byte When configuring SWD: 0x0000 0000
- Flash option byte When SWD is not configured: 0x0000 1000

Reset value for GPIOC

- Flash option byte When configuring SWD: 0x0000 0001
- Flash option byte When SWD is not configured: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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
31:16	Reserved	-	-	-
15:0	PUPDy[1:0]	RW	16'h20(PA), 16'h0 (PB, SWD configured), 16'h1000 (PB, SWD not configured), 16'h1 (PC, SWD configured), 16'h0 (PC, SWD not configured)	y = 7..0 These bits are written by software to configure the I/O pull-up or pull-down 00: No pull-up, pull-down 01: Pull-up 10: Pull-down 11: Reserved11: Reserved

9.4.5. GPIO port input data register (GPIOx_IDR) (x = A, B, C)

Address offset: 0x10

Reset value: 0x0000 XXXX

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	0	-
7:0	IDy	R	-	y = 7..0 These bits are read-only. They contain the input value of the corresponding I/O port.

9.4.6. GPIO port output data register (GPIOx_ODR) (x = A, B, C)

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	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31: 8	Reserved	-	-	-
7: 0	ODy	RW	8'h0	y = 7..0 These bits can be read and written by software. Note: the ODR bits can be individually set and/or cleared by writing to the GPIOx_BSRR or GPIOx_BRR register (x = A..F).

9.4.7. GPIO port bit set/reset register (GPIOx_BSRR) (x = A, B, C)

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	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
-								W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
-								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	0	-
23:16	BRy	W	8'h0	y = 7..0

				These bits are write-only. A read to these bits returns the value 0. 0: No action on the corresponding ODy bit 1: Clears the corresponding ODy bit Note: If both BSy and BRy are set, BSy has priority.
15:8	Reserved	-	-	-
7: 0	BSy	W	8'h0	y = 7..0 These bits are write-only. A read to these bits returns the value 0. 0: No action on the corresponding ODy bit 1: Sets the corresponding ODy bit

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

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 [7:0] is used to lock the configuration of the GPIO. During the write sequence, the value of LCKR [7:0] 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**
Res	Res	Res	Res	Res	Res	Res	Res	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
-								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. 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[7:0] 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.
15:8	Reserved	-	-	-
7: 0	LCKy	RW	8'h0	y = 7..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

9.4.9. GPIO alternate function low register (GPIOx_AFRL) (x = A, B, C)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	AFSEL7 [2:0]			Res	AFSEL6 [2:0]			Res	AFSEL5 [2:0]			Res	AFSEL4 [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	AFSEL3 [2:0]			Res	AFSEL2 [2:0]			Res	AFSEL1 [2:0]			Res	AFSEL0 [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	AFSEL7 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL7 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
27	Reserved	-	-	-
26:24	AFSEL6 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL6 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
23	Reserved	-	-	-
22:20	AFSEL5 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL5 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
19	Reserved	-	-	-
18:16	AFSEL4 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL4 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
15	Reserved	-	-	-
14:12	AFSEL3 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL3 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
11	Reserved	-	-	-
10:8	AFSEL2 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL2 Selection:

Bit	Name	R/W	Reset Value	Function
				000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
7	Reserved	-	-	-
6:4	AFSEL1 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL1 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7
3	Reserved	-	0	-
2:0	AFSEL0 [2:0]	RW	3'h0	These bits are written by software to configure alternate function I/Os. AFSEL0 Selection: 000: AF0 001: AF1 010: AF2 011: AF3 100: AF4 101: AF5 110: AF6 111: AF7

9.4.10. GPIO port bit set/reset register (GPIOx_BSRR) (x = A, B, C)

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	2	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
-								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	-
7:0	BRy	W	8′ h0	y = 7..0 These bits are write-only. A read to these bits returns the value 0. 0: No action on the corresponding ODy bit 1: Clears the corresponding ODy bit

10. System configuration controller (SYSCFG)

The PY32L020 series devices feature a set of configuration registers. The main purposes of the system configuration controller are the following:

- Enabling/disabling I²C analog filtering on some IO ports
- Remapping the memory located at the beginning of the code area
- Managing the external interrupt line connection to the GPIOs
- Managing robustness feature

10.1. SYSCFG registers

10.1.1. SYSCFG configuration register 1 (SYSCFG_CFGR1)

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

Two bits are used to configure the type of memory accessible at address 0x0000 0000. These bits are used to select the physical remap by software and so, 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												I2C_PB6_EIIC	I2C_PB4_EIIC	I2C_PB3_EIIC	I2C_PA2_EIIC
-												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res										COMP2_OC-REF_CLR TIM1	COMPI_OC-REF_CLR TIM1	TIM1_IC1_SRC		MEM_MODE[1:0]	
-										RW	RW	RW		RW	

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	-
19	I2C_PB6_EIIC	RW	0	I²C Analog Filter Enable for Control PB6 0: Analog filter disabled 1: Analog filter enabled
18	I2C_PB4_EIIC	RW	0	I²C Analog Filter Enable for Control PB4 0: Analog filter disabled 1: Analog filter enabled
17	I2C_PB3_EIIC	RW	0	I²C analog filtering enable for controlling PB3 0: Analog filter disabled 1: Analog filter enabled
16	I2C_PA2_EIIC	RW	0	I²C Analog Filter Enable for Control PA2 0: Analog filter disabled 1: Analog filter enabled
15:6	Reserved	-	-	-
5	COMP2_OCREF_CLR TIM1	RW	0	1: COMP2 output as TIM1 ocref chr input 0: COMP2 output is not as TIM1 ocref chr input
4	COMPI_OCREF_CLR TIM1	RW	0	1: COMP1 output as TIM1 ocref chr input 0: COMP1 output is not as TIM1 ocref chr input
3:2	TIM1_IC1_SRC	RW	0	TIM1 CH1 input source: 00, 11: TIM1 CH1 IO 01: COMP1 10: COMP2
1:0	MEM_MODE[1:0]	RW	-	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.

Bit	Name	R/W	Reset Value	Function
				X0: Main flash, mapped at 0x0000 0000 01: System flash, mapped at 0x0000 0000 11: SRAM at 0x0000 0000

10.1.2. SYSCFG configuration register 2 (SYSCFG_CFGR2)

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					ETR_SRC_TIM1		Res				COMP2_BRK_TIM1	COMP1_BRK_TIM1	Res		LOCKUP_LOCK
-					RW		-				RW	RW	-		RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	-
10:9	ETR_SRC_TIM1[1:0]	RW	2’h0	TIMER1 ETR input source selection. 2 'b00: ETR derived from GPIO 2 'b01: ETR derived from COMP1 2 'b10: ETR derived from COMP2 2 'b11: ETR derived from ADC
8:5	Reserved	-	-	-
4	COMP2_BRK_TIM1	RW	0	COMP2 is enabled as a TIMx break input. 0: COMP2 output is not input as TIM1 break 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 input as TIM1 break 1: COMP1 output as TIM1 break input
2:1	Reserved	-	-	-
0	LOCKUP_LOCK	RW	0	Enable Cortex-M0+ LOCKUP Software set, system reset and clear. It enables and locks the LOCKUP (HardFault) output of Cortex-M0 + to the brake input of TIM1. 0: The LOCKUP output of Cortex-M0 + is not connected to the brake input of TIM1 1: The LOCKUP output of Cortex-M0 + is connected to the brake input of TIM1

10.1.3. GPIO filter enable (GPIO_ENS)

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PC_ENS	
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB_ENS								PA_ENS							
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	-	-	-
17:16	PC_ENS[x]	RW	2’h0	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter
15:8	PB_ENS[x]	RW	8’h0	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter
7:0	PA_ENS[x]	RW	8’h	Noise filter enabled, set 1 to take effect 0: Noise filter is not activated 1: Start the noise filter

11. Interrupts and events

11.1. Nested vectored interrupt controller (NVIC)

11.1.1. Main features

- Support 13 maskable external interrupts (not including the six 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. All interrupts including the core exceptions are managed by the NVIC.

11.1.2. SysTick calibration value register

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

11.1.3. Interrupt and exception vectors

Table 11-1Vector table

Position	Priority	Type 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	HardFault_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	settable	SysTick	System tick timer	0x0000_003C
0	7	-	Reserved	Reserved	0x0000_0040
1	8	-	Reserved	Reserved	0x0000_0044
2	9	-	Reserved	Reserved	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_7	EXTI Line [7:4] interrupt	0x0000_005C
8	15	-	Reserved	Reserved	0x0000_0060
9	16	-	Reserved	Reserved	0x0000_0064
10	17	-	Reserved	Reserved	0x0000_0068
11	18	-	Reserved	Reserved	0x0000_006C
12	19	settable	ADC_COMP	ADC and COMP interrupt (COMP through EXTI lines 17/18)	0x0000_0070
13	20	settable	TIM1_BRK_UP_TRG_COM	TIM1 break, update, trigger and commutation interrupt	0x0000_0074
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	settable	LPTIM1	LPTIM interrupts	0x0000_0084
18	25	-	Reserved	Reserved	0x0000_0088
19	26	settable	TIM14	TIM14 global interrupt	0x0000_008C

Position	Priority	Type of priority	Acronym	Description	Address
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	settable	USART1	USART1 global interrupt	0x0000_00AC
28	35	-	Reserved	Reserved	0x0000_00B0
29	36	-	Reserved	Reserved	0x0000_00B4
30	37	-	Reserved	Reserved	0x0000_00B8
31	38	-	Reserved	Reserved	0x0000_00BC

11.2. 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, generating IRQ of CPU
- 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 11 configurable/direct event inputs (including 10 configurable event inputs and 1 direct event inputs).

11.2.1. EXTI main features

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

11.2.2. EXTI block diagram

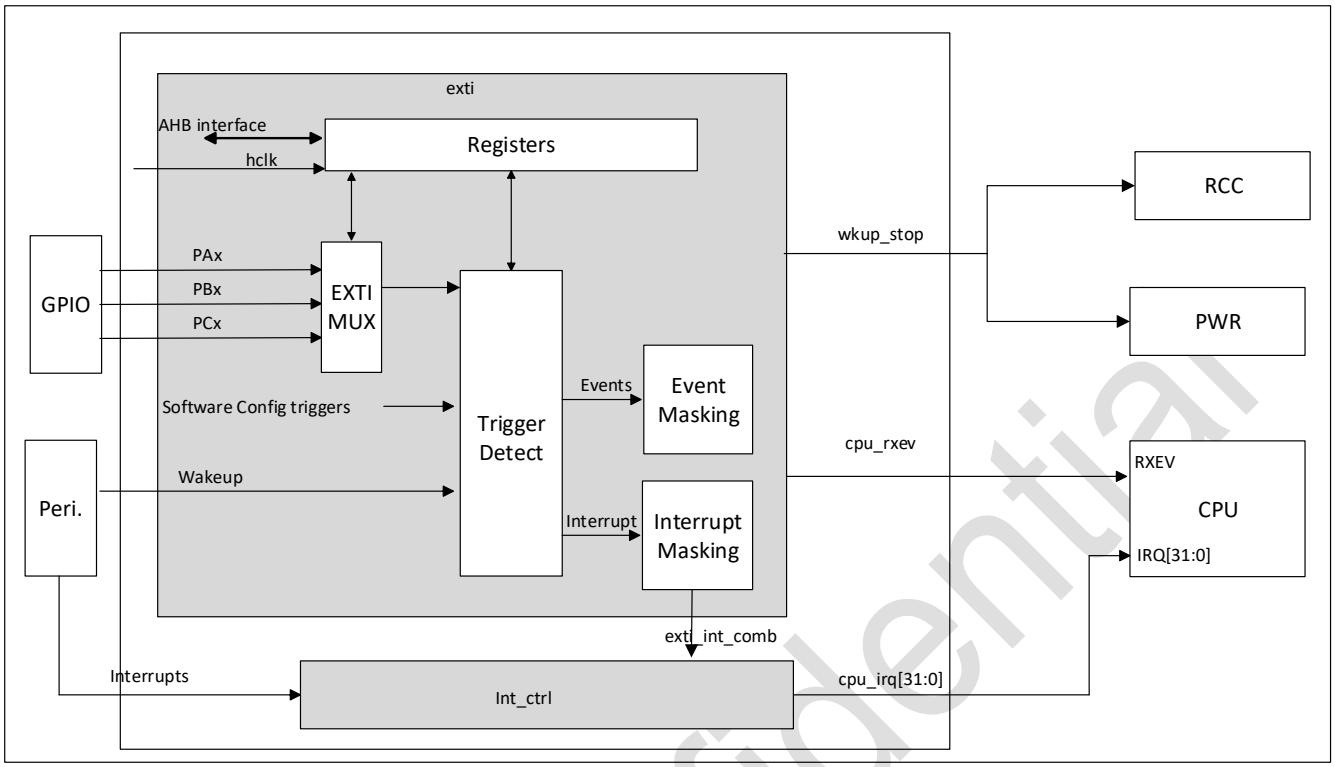

Figure 11-1 EXTI block diagram

11.2.3. 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. The 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). A Configurable type event interrupt requires the CPU to clear it by writing 1 to the EXTI_PR register.

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

11.2.4. 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. When the CPU processes the interrupt generated by this type of trigger event, the software needs to clear the interrupt status bit of the peripheral module.

11.2.5. EXTI multiplexer

The GPIOs are connected to 8 external interrupt/event lines:

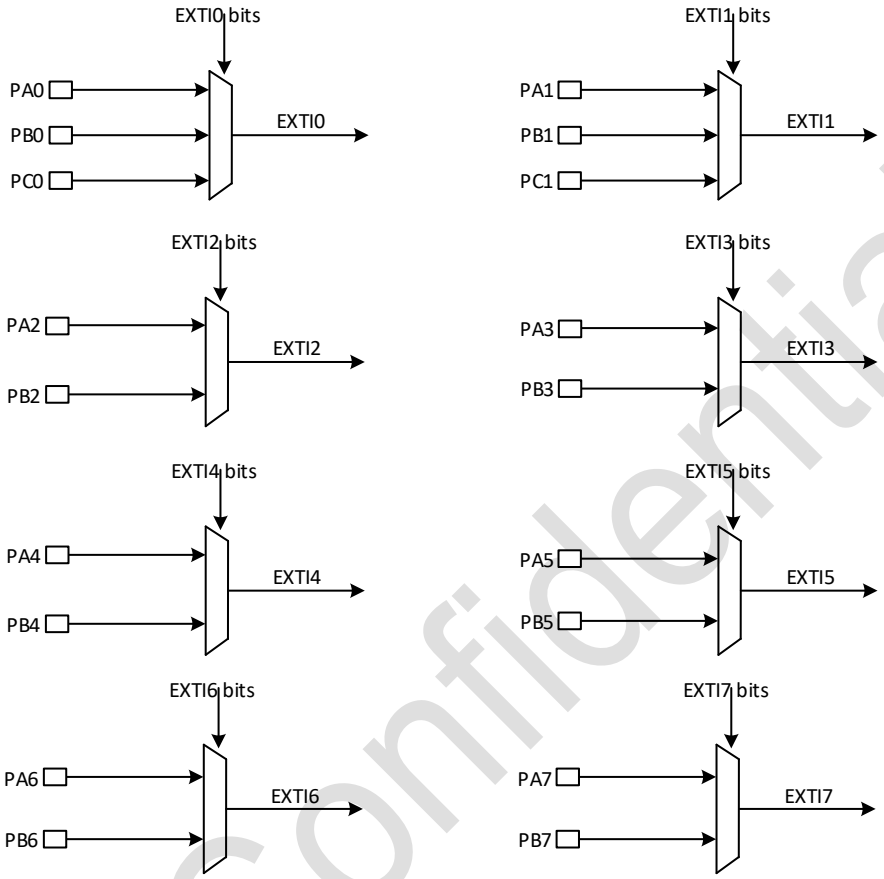

Figure 11-2 External interrupt/event GPIO mapping

The EXTI lines are connected as shown as follows:

Table 11-2 EXTI lines connections

EXTI line	Line source	Line type
Line 0-7	GPIO	Configurable
Line 8-16	Reserved	-
Line 0-7	COMP 1 output	Configurable
Line 18	COMP 2 output	Configurable
Line 19	Reserved	-
Line 20	Reserved	-
Line 21	Reserved	-
Line 22	Reserved	-
Line 23	Reserved	-
Line 24	Reserved	-
Line 25	Reserved	-
Line 26	Reserved	-
Line 27	Reserved	-
Line 28	Reserved	-
Line 29	LPTIM	direct

11.3. EXTI register

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

11.3.1. 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**
Res	Res	Res	Res	Res	Res	Res	Res	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	-
18	RT18	RW	0	Rising trigger event configuration bit of line18 0: Disabled 1: Enabled
17	RT17	RW	0	Rising trigger event configuration bit of line17 0: Disabled 1: Enabled
16:8	Reserved	-	-	-
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 external wakeup lines are edge triggered. No glitches must be generated on these lines. If a rising edge on an external 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.

11.3.2. 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	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	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	-
18	FT18	RW	0	Falling trigger event configuration bit of line18 0: Disabled 1: Enabled
17	FT17	RW	0	Falling trigger event configuration bit of line17 0: Disabled 1: Enabled
16:8	Reserved	-	-	-
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.

11.3.3. 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**
Res	Res	Res	Res	Res	Res	Res	Res	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	-
18	SWI18	RW	0	Rising trigger event configuration bit of line18 0: No effect.

Bit	Name	R/W	Reset Value	Function
				1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
17	SWI17	RW	0	Rising trigger event configuration bit of line17 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is self-cleared by the hardware. Read returns 0.
16:8	Reserved	-	-	-
7	SWI7	RW	0	Rising trigger event configuration bit of line7 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
6	SWI6	RW	0	Rising trigger event configuration bit of line6 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
5	SWI5	RW	0	Rising trigger event configuration bit of line5 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
4	SWI4	RW	0	Rising trigger event configuration bit of line4 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
3	SWI3	RW	0	Rising trigger event configuration bit of line3 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
2	SWI2	RW	0	Rising trigger event configuration bit of line2 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
1	SWI1	RW	0	Rising trigger event configuration bit of line1 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)
0	SWI0	RW	0	Rising trigger event configuration bit of line0 0: No effect. 1: Generate rising edge trigger event, which in turn generates interrupt This bit is cleared by hardware, and read returns 0 (after hardware clearing) or configuration value (before hardware clearing)

11.3.4. Pending register (EXTI_PR)

Address offset: 0x0C

Reset value: 0x0000 0000

Contains only register bits for configurable events.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res													PR18	PR17	Res
-													RC_W1	RC_W1	-

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
-								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	-	-	-
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.
16:8	Reserved	-	-	-
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	PR2	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

Bit	Name	R/W	Reset Value	Function
				1: Generate rising edge/falling edge/software trigger event request.
0	PR0	RC_W1	0	Rising trigger event configuration bit of line0 When software or hardware generates rising/falling edge trigger events, this position is bit. This bit is cleared by writing 1 0: No trigger request occurred 1: Generate rising edge/falling edge/software trigger event request.

11.3.5. EXTI external interrupt selection register 1 (EXTI_EXTICR1)

Address offset: 0x60

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	EXTI3	Res	Res	Res	Res	Res	Res	Res	EXTI2
-							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
31:25	Reserved	-	-	Reserved
24	EXTI3	RW	0	EXIT3 GPIO port selection 0: PA[3] pin 1: PB[3] pin
23:17	Reserved	-	-	Reserved
16	EXTI2	RW	0	EXIT2 GPIO port selection 0: PA[2] pin 1: PB[2] pin
15:10	Reserved	-	-	Reserved
9:8	EXTI1 [1:0]	RW	2'h0	EXIT1 GPIO port selection 2'b00: PA[1] pin 2'b01: PB[1] pin 2'b10: PC[1] pin 2'b11: Reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI0 [1:0]	RW	2'h0	EXIT0 GPIO port selection 2'b00: PA[0] pin 2'b01: PB[0] pin 2'b10: PC[0] pin 2'b11: Reserved

11.3.6. EXTI external interrupt selection register 2 (EXTI_EXTICR2)

Address offset: 0x64

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	EXTI7	RW	0	EXIT7 GPIO port selection 0: PA[7] pin 1: PB[7] pin
23:17	Reserved	-	-	Reserved
16	EXTI6	RW	0	EXIT6 GPIO port selection 0: PA[6] pin 1: PB[6] pin
15:9	Reserved	-	-	Reserved
8	EXTI5	RW	0	EXIT5 GPIO port selection

Bit	Name	R/W	Reset Value	Function
				0: PA[5] pin 1: PB[5] pin
7:1	Reserved	-	-	Reserved
0	EXTI4	RW	0	EXIT4 GPIO port selection 0: PA[4] pin 11: PB[4] pin

11.3.7. Interrupt mask register (EXTI_IMR)

Address offset: 0x80

Reset value: 0x2000 0000

Attention:The reset value is set such as to, by default, enable interrupt from direct lines, and disableinterrupt from configurable lines.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	IM29	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IM18	IM17	Res
-		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	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29	IM29	RW	1	Interrupt mask on line 29. 0: Interrupt request is masked 1: Interrupt request is not masked
28:19	Reserved	-	-	-
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
16:8	Reserved	-	-	-
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

11.3.8. 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	EM29	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	EM18	EM17	Res
-		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	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29	EM29	RW	0	Interrupt mask on line 29. 0: Interrupt request is masked 1: Interrupt request is not masked
28:19	Reserved	-	-	-
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
16:8	Reserved	-	-	-
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

12. Cyclic redundancy check (CRC)

12.1. Introduction

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

12.2. 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

12.3. CRC functional description

12.3.1. CRC block diagram

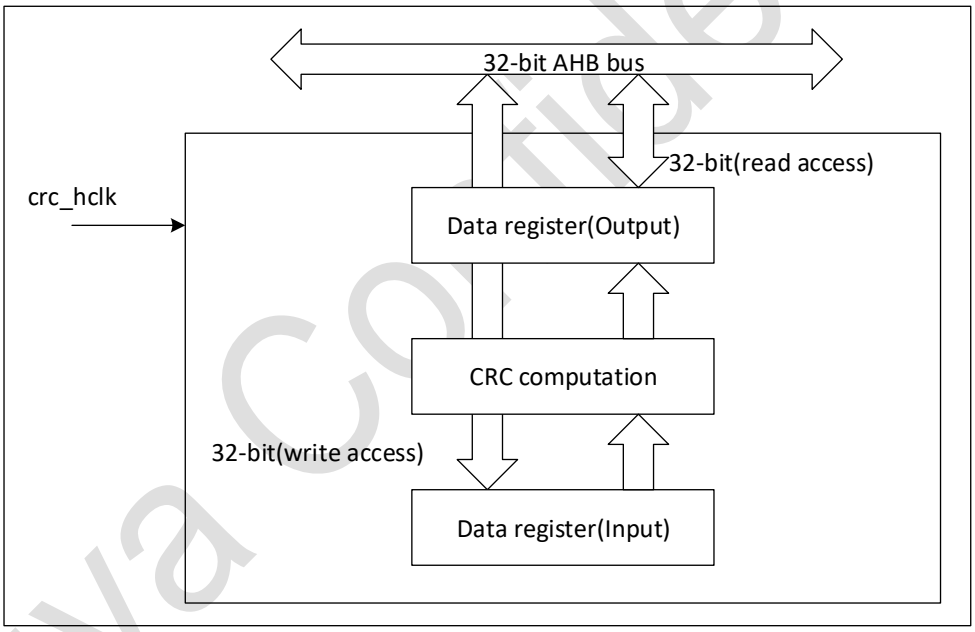

Figure 12-1 CRC calculation unit block diagram

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

12.4. CRC registers

12.4.1. 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.

12.4.2. 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	8'h0	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.

12.4.3. CRC control register (CRC_CR)

Address offset: 0x08

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	-
0	RESET	W	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.

13. Analog-to-digital converters (ADC)

13.1. Introduction

The device has a 12-bit SAR-ADC. The module has a total of up to 10 channels to be measured, including 8 external and 2 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} and V_{REFINT}。

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

The analog watchdog features allow the application to detect if the input voltage goes outside the user-defined high or low thresholds.

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

Interrupt generation at the end of sampling, end of conversion, end of sequence conversion, and in case of analog watchdog or overrun events.

13.2. ADC main features

- High performance
  - 12, 10, 8 or 6-bit configurable resolution
  - ADC conversion time: 1.3 μs @12 bit (0.75 Msps)
  - Self-calibration
  - Programmable sampling time
  - Programmable data alignment mode
- Low-power
  - Application can reduce PCLK frequency for low-power operation while still keeping optimum ADC performance.
  - Auto off mode: prevents ADC overrun in applications with low frequency PCLK
- Analog input channels
  - 8 external analog inputs
  - 1 channel for internal temperature sensor (T_{S_VIN})
  - 1 channel for internal reference voltage (V_{REFINT})
- Start-of-conversion can be initiated
  - By software
  - By hardware triggers with configurable polarity (e.g. 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
- Overrun events

■ Analog watchdog

13.3. ADC functional description

13.3.1. ADC block diagram

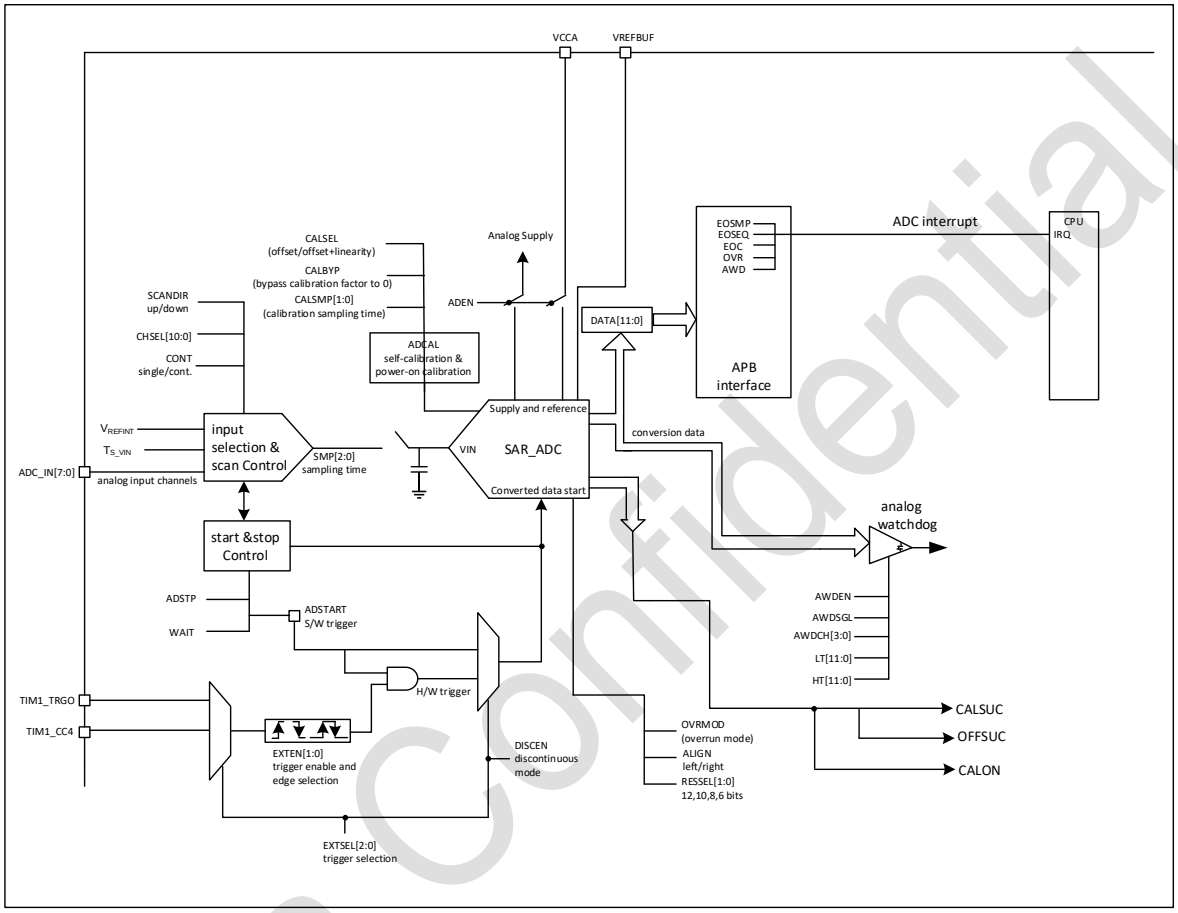

Figure 13-1 ADC block diagram

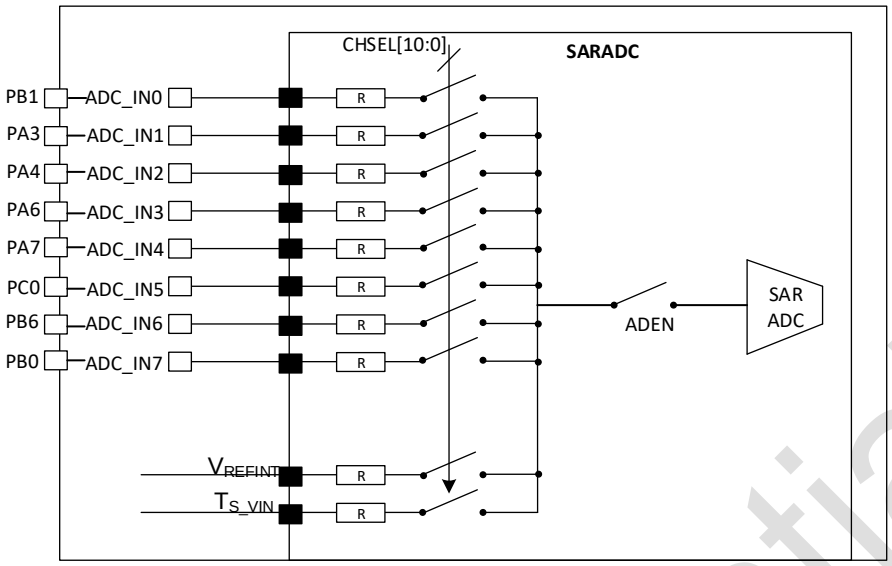

Figure 13-2 ADC connectivity

13.3.2. 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 must not use the ADC during calibration and must wait until it is complete.

Calibration is preliminary to any ADC operation. It removes the offset error which may vary from chip to chip due to process.

ADC software calibration

The software sets ADCAL = 1 to start the calibration and only supports selecting the system clock as the clock of the ADC. When the calibration is complete, the ADCAL is cleared by the hardware.

When the operating conditions of the ADC change (V_{CC} change is the main factor of ADC offset offset and mismatch offset, followed by temperature change), it is recommended to perform a recalibration operation.

Software procedure to calibrate the ADC:

- CKMODE select system clock
- Set ADCAL=1
- Wait until ADCAL=0

13.3.3. 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.

2. ADC conversion can be initiated by setting ADSTART or by an external trigger event.

Follow this procedure to disable the ADC:

1. Check that ADSTART=0 in the ADC_CR register to ensure that no conversion is ongoing.

2. If ADSTART = 0 and ADEN = 1, ADC can be disabled in two ways

— Disable the ADC by setting 1 for ADDIS in ADC_CR.

— Setting 1 to ADSTP in the ADC_CR register disables the ADC and wait for the ADSTP to be cleared by hardware (clearing indicates that the conversion stops complete).

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

13.3.4. 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 can be generated by two possible clock sources, the clock structure of ADC is shown in the figure below and the ADC clock division coefficient is shown in the table below.

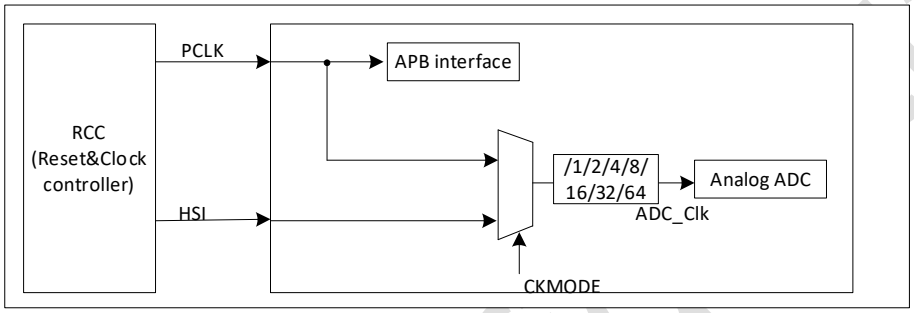

Figure 13-3 ADC block diagram

Table 13-1 ADC clock source and frequency division coefficient table

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

13.3.5. Configuring the ADC

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

For all the other control bits in the ADC_IER, ADC_CFGRi (i=1,2), ADC_SMPR, ADC_TR, and ADC_CCR registers, software must only write to the configuration control bits if the ADC is enabled (ADEN = 1) and if there is no conversion ongoing (ADSTART = 0). ADC_CHSELR is written with ADEN = 0 and ADSTART = 0.

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

13.3.6. Channel selection (CHSEL, SCANDIR)

There are up to 10 multiplexed channels:

- 8 analog inputs from GPIO pins (ADC_IN0...ADC_IN7)
- 2 internal analog inputs (temperature sensor and internal reference voltage)

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

The temperature sensor is connected to channel ADC_IN8 (T_{S_VIN}). The internal voltage reference V_{REFINT} is connected to channel ADC_IN9.

13.3.7. 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.

The ADC samples the input voltage for a number of ADC clock cycles that can be modified using the SMP[2:0] bits 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+0.5)) x ADC clock cycles = 16 x ADC clock cycles = 1.33 µs

13.3.8. 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 EOSEQIE 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.

13.3.9. Continuous conversion mode (CONT=1)

In continuous conversion mode, when the software sets ADSTART or the hardware trigger event occurs, the ADC continuously performs a 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:

1. The converted data are stored in the 16-bit ADC_DR register.
2. 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. 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.

13.3.10. 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.

13.3.11. 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. Itis possible to re-configure the ADC while ADSTART=0, indicating that the ADC is idle.

The ADSTART bit is cleared by hardware:

1. The single mode is triggered by software (CONT=0, EXTEN=00). At any end of conversion sequence (EOSEQ=1), the hardware automatically clears ADSTART.
2. The discontinuous mode is triggered by software (CONT=0， DISCEN=1, EXTSEL=0x0). At any end of conversion sequence (EOC=1), the hardware automatically clears ADSTART.
3. In all cases (CONT=x, EXTEN=XX), after the ADSTP procedure invoked by software, the hardware automatically clears ADSTART.

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 having to set the ADSTART bit again.

13.3.12. 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: Refer to Figure 13-4 for the conversion timing.

$t_{ADC} = t_{SMPL} + t_{SAR} = [\ 3.5_{|min} + 12.5_{|12bit}] * t_{ADC\_CLK}$

$t_{ADC} = t_{SMPL} + t_{SAR} = 291.66ns_{|min} + 1041.66\ ns_{|12bit} = 1.33\ \mu s_{|min}$ (for f_{ADC_CLK} = 12 MHz)

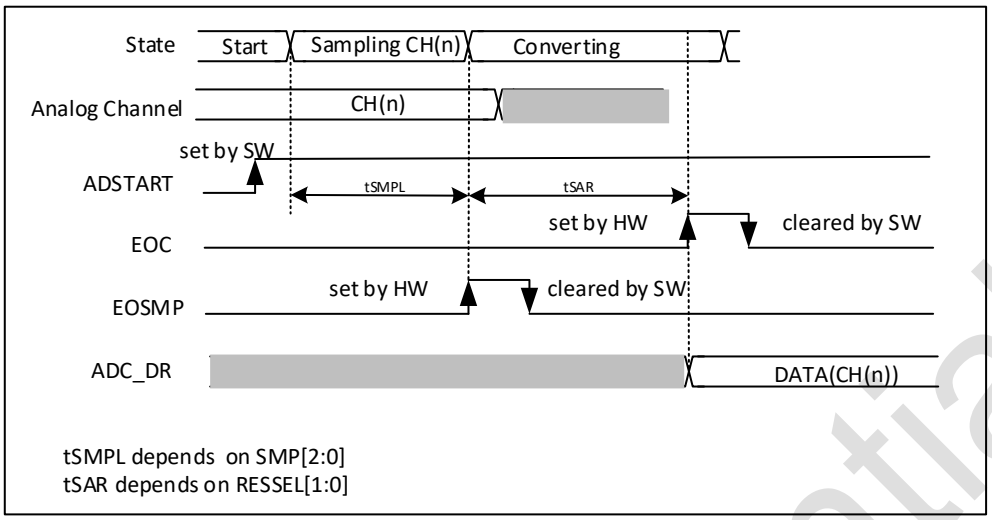

Figure 13-4 ADC conversion timings

13.3.13. 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. The ADSTP stop conversion timing can be shown in figure below.

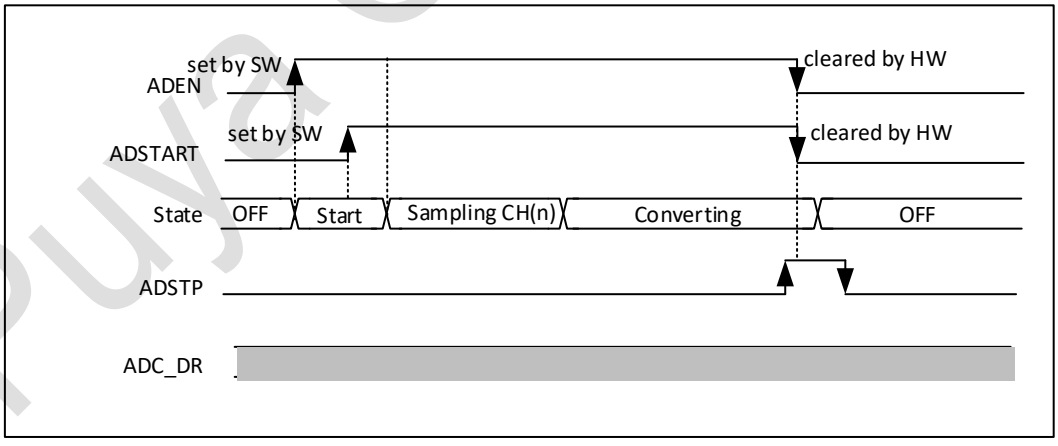

Figure 13-5 ADSTP stopping an ongoing conversion

13.3.14. 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 capture). 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 13-2 Configuring the trigger polarity

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 hardware trigger sources are shown in table below Software source trigger events can be generated by setting the ADSTART bit in the ADC_CR register.

Table 13-3 External triggers

Source	EXTSEL[2:0]
TIM1_TRGO	000
TIM1_CC4	001

Note: The trigger selection can be changed only when the ADC is not converting

13.3.15. 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 RESSEL [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 13-4 Resolution and total transition time

RESSEL [1:0]	t_{SAR} (ADC Clock Cycle)	t_{SAR}(ns) f_{ADC} = 12 MHz	t_{SMP} (ADC Clock Cycle)	t_{ADC}(t_{SMP} = 3.5) (ADC Clock Cycle)	t_{ADC}(ns) f_{ADC} = 12 MHz
00 (12 bits)	12.5	1042 ns	3.5	16	1334 ns
01 (10 bits)	10.5	876ns	3.5	14	1166ns
10 (8 bits)	8.5	792 ns	3.5	12	1000 ns
11 (6 bits)	6.5	542 ns	3.5	10	834 ns

13.3.16. 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.

13.3.17. End of conversion sequence (EOSEQ flag)

The ADC notifies the application of each end of sequence (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.

13.3.18. Example timing diagrams

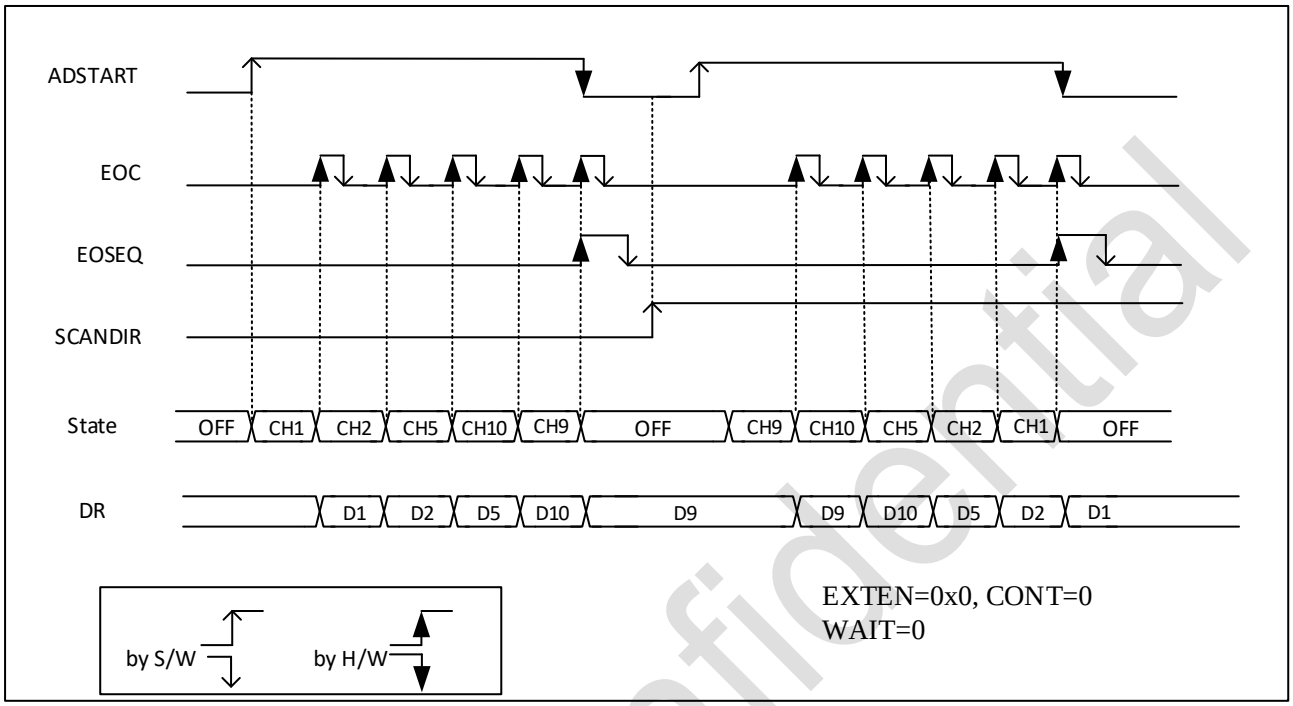

Figure 13-6 Single conversions of a sequence, software trigger

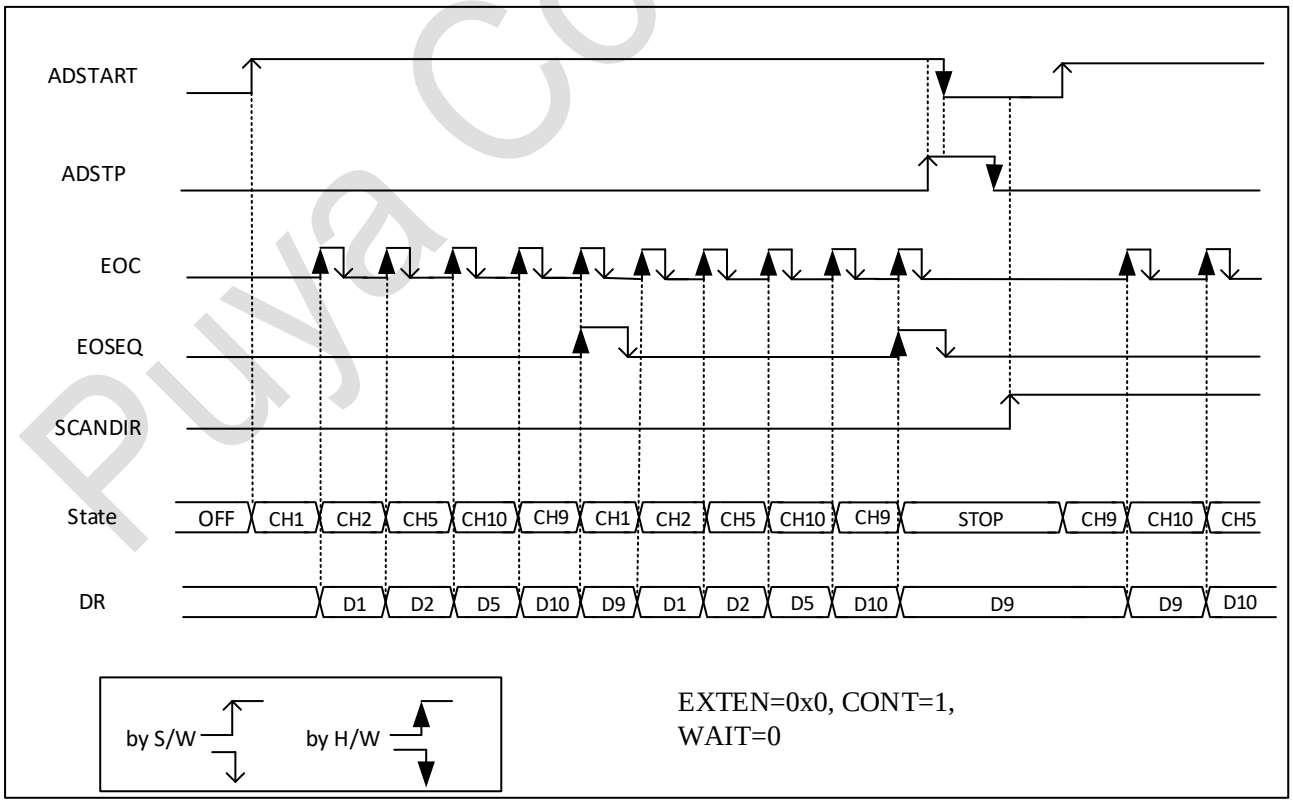

Figure13-7 Continuous conversion of a sequence, software trigger

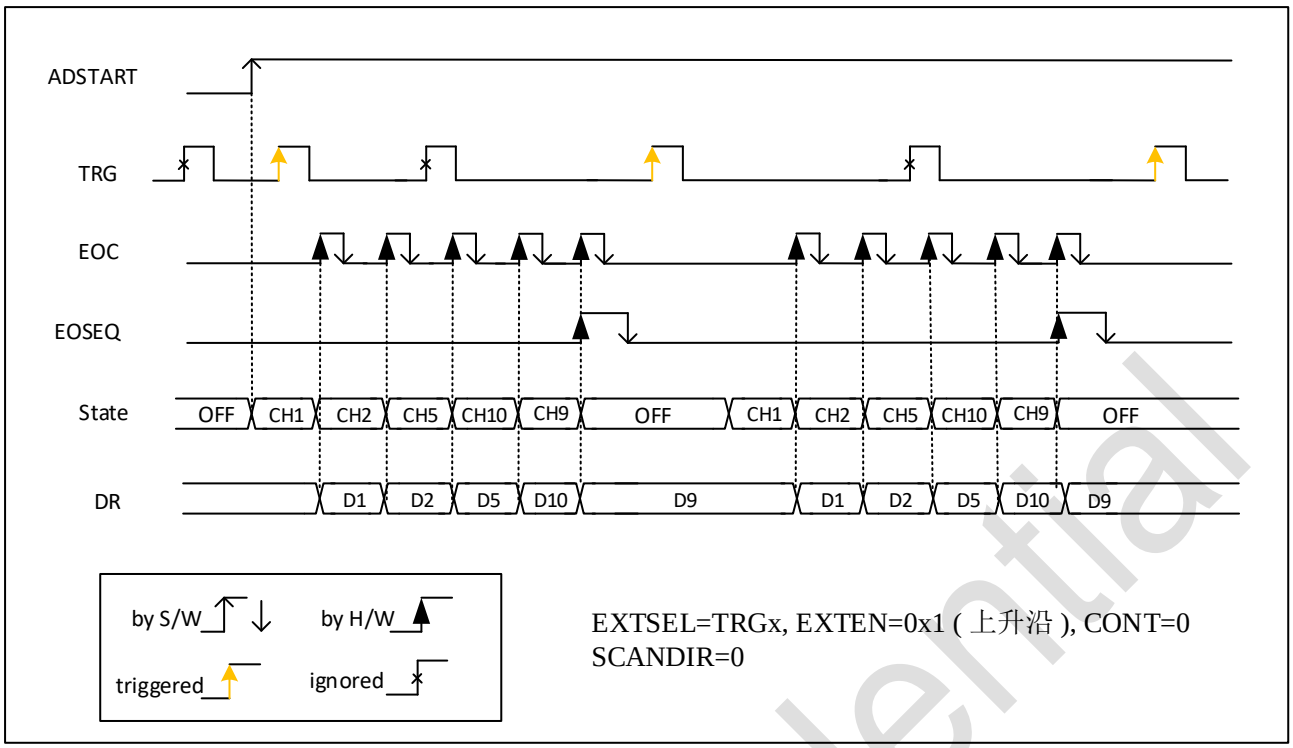

Figure13-8 Single conversions of a sequence, hardware trigger

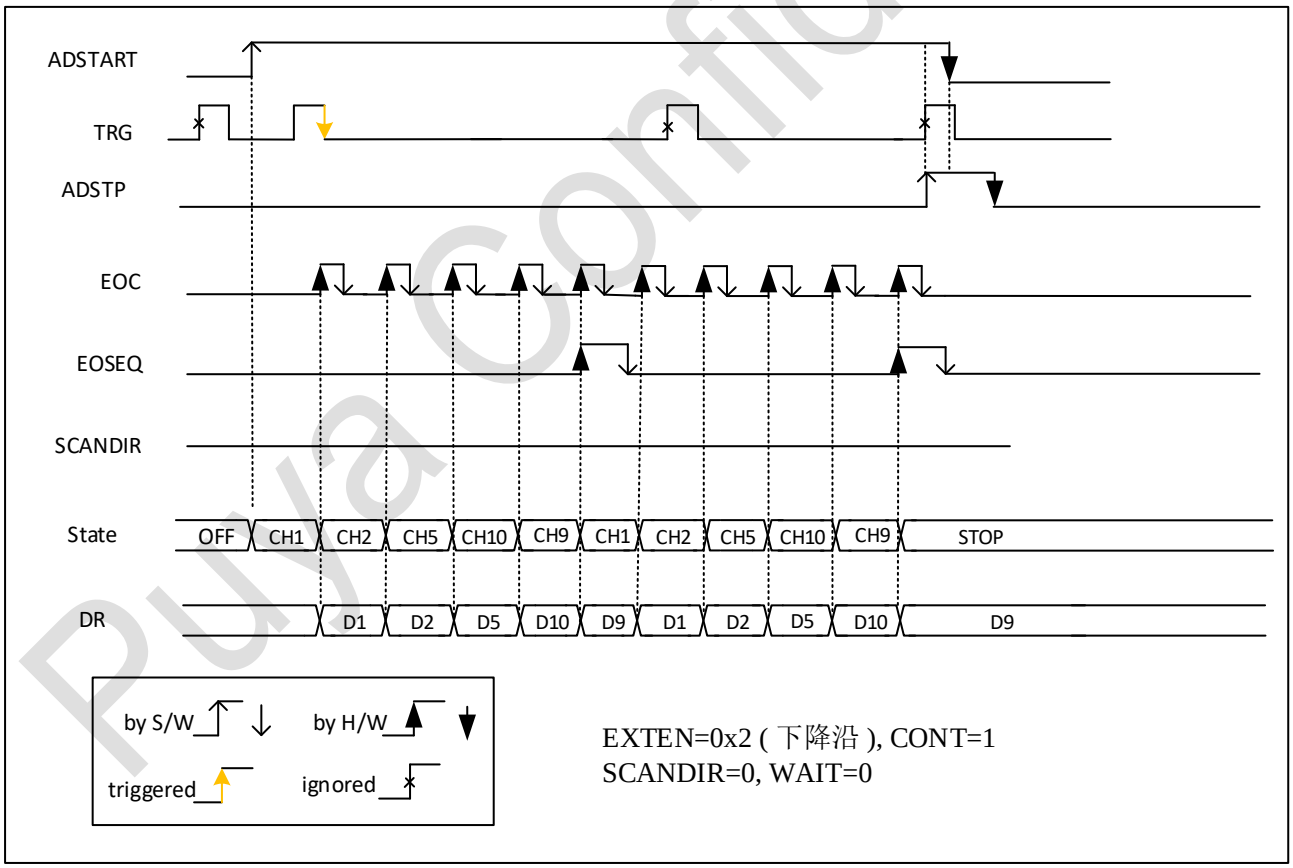

Figure13-9 Continuous conversions of a sequence, hardware trigger

13.3.19. Data management

13.3.19.1. 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) as shown in table below.

Table 13-5 Data alignment and resolution

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			

13.3.19.2. ADC overrun (OVR, OVRMOD)

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

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 overrun condition occurs, the ADC keeps operating and can continue to convert unless the software decides to stop and reset the sequence by setting the ADSTP bit in the ADC_CR register The OVR flag is cleared by software by writing 1 to it.

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:Figure 13-10 Example of overrun (OVR)

- OVRMOD=0

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

- OVRMOD=1

The data register is overwritten with the last conversion result and the previous unread data is lost. If OVR remains at 1, further conversions can be performed, and the ADC_DR register always contains the data from the latest conversion.

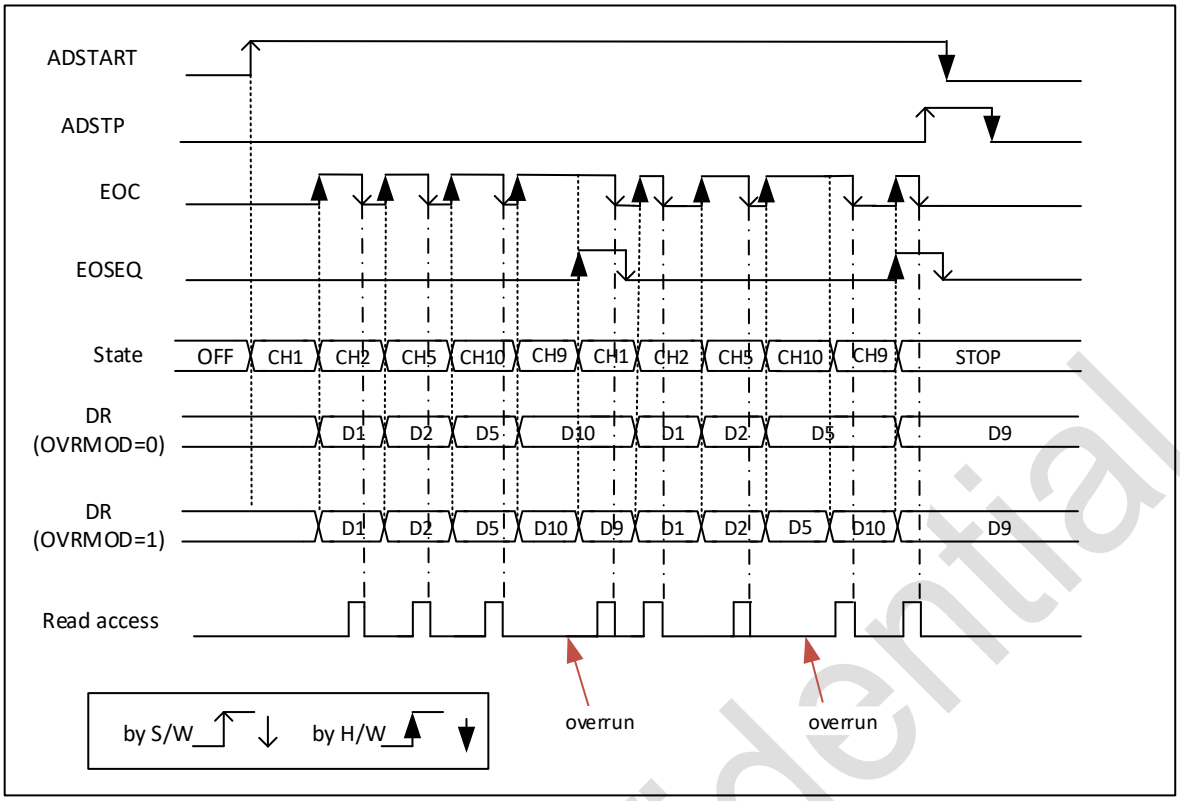

Figure 13-10 Example of overrun (OVR)

13.3.19.3. Managing conversions

If the conversion of the ADC is slow enough, the conversion sequence can be controlled by software. In this case, the software processes each conversion data through the EOC flag and its associated interrupt. At the end of each conversion, at the EOC position bit in the ADC_ISR register, the converted value of the ADC_DR register can be read at this time. The OVRMOD bit in the ADC_CFGR1 register may be configured to 0 to manage overload events.

13.3.19.4. Conversion in case of overload detection

When there is an application that converts one or more channels and does not have to read the result every time the conversion is made. In this case, the OVRMOD bit must be set to 1 and the software should ignore the OVR flag. When OVRMOD = 1, the overload event cannot prevent the ADC from continuing conversion and the data in the ADC_DR register is always the latest converted data.

13.3.20. Low-power features

13.3.20.1. 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.

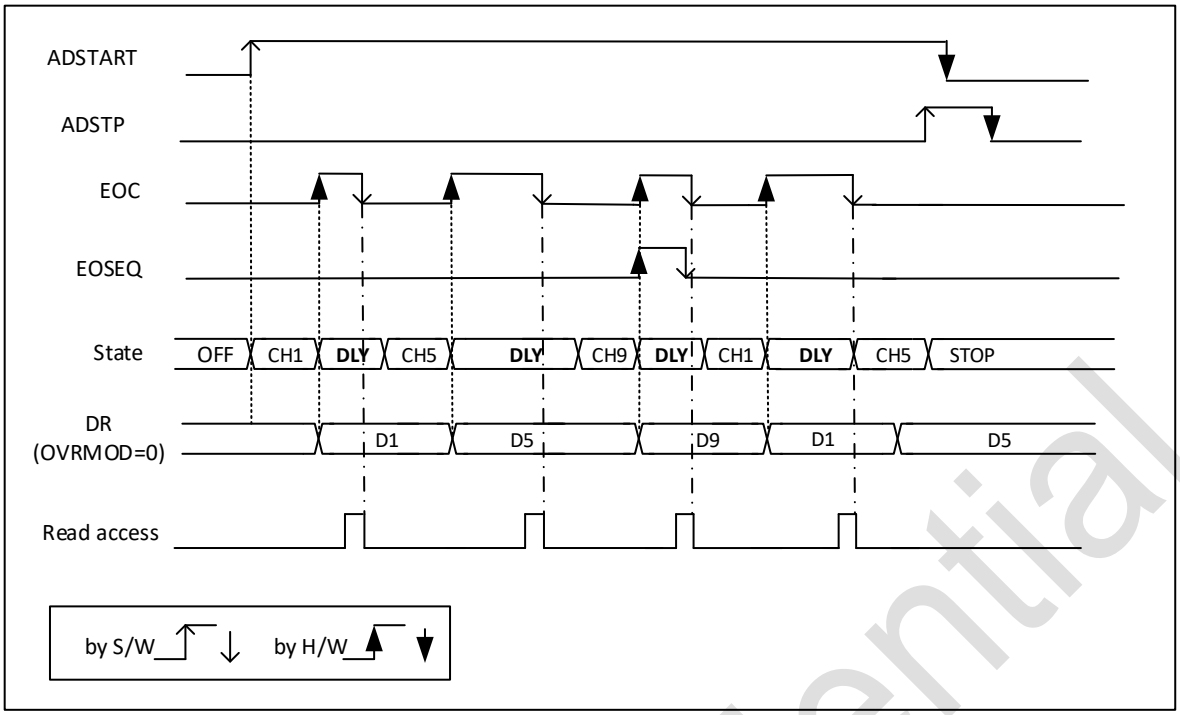

Figure 13-11 Wait mode conversion

13.3.21. 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. Is the analog watchdog configured for single channel monitoring or all channel monitoring.

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. These thresholds are programmed in the ADC_TR register. 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.

Note: ADC analog input channel 0 does not support single channel analog watchdogs.

Table 13-6 Analog watchdog comparison

Resolution (bits)	Analog watchdog comparison between:		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"

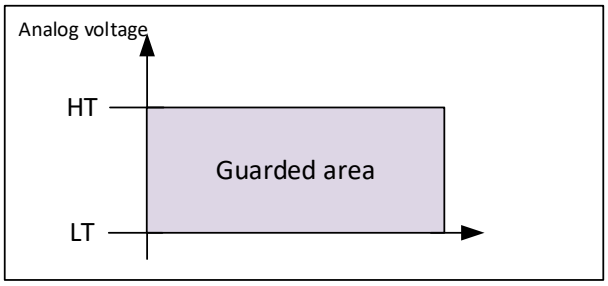

Figure 13-12 Analog watchdog guarded area

Table13-7 Analog watchdog channel selection

Analog watchdog monitoring channel	AWDSGL bit	AWDEN bit
None	x	0
All channels	0	1
Single channel	1	1

13.3.21.1. 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.
- ADC_AWD_OUT is reset after the end of the next guarded conversion which is inside the programmed thresholds. 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_AWD_OUT state.
- A channel not selected as an analog watchdog does not affect the ADC_AWD_OUT status bit.

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

The ADC_AWD_OUT signal is generated by the PCLK domain.

The analog watchdog threshold comparison is performed at the end of each ADC transition.

13.3.22. Temperature sensor and internal reference voltage

The temperature sensor can be used to measure the junction temperature ($_{TJ}$) of the device.

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

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

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

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

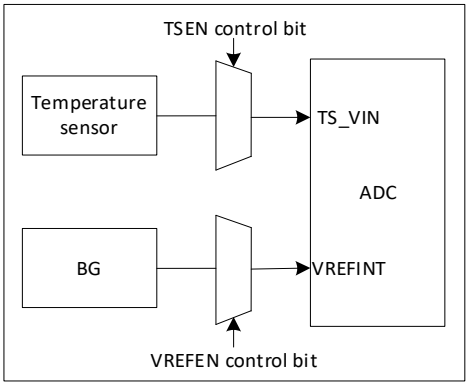

Figure 13-13 Temperature sensor and V_{REFINT} channel block diagram

Reading the temperature:

1. Select the ADC_IN18 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 the resulting data in the ADC_DR register
6. Calculate the actual temperature using the following formula (x7 version):

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

T_{S_CAL2} is the temperature sensor calibration value acquired at 105°C, calibration value storage address: 0x1FFF 0118

T_{S_CAL1} is the temperature sensor calibration value acquired at 30°C, calibration value storage address: 0x1FFF 0114

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

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

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

T_{S_CAL2} is the temperature sensor calibration value acquired at 85°C, calibration value storage address: 0x1FFF 0118

T_{S_CAL}1 is the temperature sensor calibration value acquired at 30°C, calibration value storage address: 0x1FFF 0114

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

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

Use V_{CC} voltage to calculate Vchannel

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

V_{REFINT} is fixed at 1.2 V.

$V_{CHANNEL}$ is the channel voltage.

ADC_DATAx is the actual conversion result of the V_{REFINT} channel

ADC_DATA is the ADC_DR conversion data of the channel.

4095 is represented as 12 bits.

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

13.3.23. 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 13-8 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
Overrun	OVR	OVRIE

13.4. ADC registers

13.4.1. 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															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								AWD	Res		OVR	EOSEQ	EOC	EOSMP	Res
-								RC_W1	-		RC_W1	RC_W1	RC_W1	RC_W1	-

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	-
7	AWD	RC_W1	0	Analog watchdog This bit is set by hardware when the converted voltage crosses the values programmed in the ADC_TR register. This bit is cleared by writing 1 0: No analog watchdog event occurred (or the flag event was already cleared by software) 1: Analog watchdog event occurred

Bit	Name	R/W	Reset Value	Function
6:5	Reserved	-	0	-
4	OVR	RC_W1	0	ADC overrun This bit is set by hardware when an overrun occurs. 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	-	0	-

13.4.2. 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															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								AWDIE	Res		OVRIE	EOSEQIE	EOCIE	EOSMPIE	Res
-								RW	-		RW	RW	RW	RW	

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

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

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

13.4.3. 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
ADCAL	Res														
RS	-														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								Vrefbuff_sel		Vrefbuff_en	ADSTP	Res	ADSTART	ADDIS	ADEN
-								RW		RW	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.
30:8	Reserved	-	0	-
7:6	Vrefbuff_sel	RW	2'h0	V_{REFBUF} output voltage selection 00: 1.5 V 01: 2.048 V 10: 2.5 V Others: Reserved Remarks: (1.5 V V_{REFBUF} trimming value storage address: 0x1FFF002C. 2.048 V V_{REFBUF} trimming value storage address: 0x1FFF0030 2.5 V V_{REFBUF} trimming value storage address: 0x1FFF0034. For example: the 16-bit value read from address 0x1FFF002C is 0x1501, indicating that the accurate value of V_{REFBUF} is 1.501 V)
5	Vrefbuff_en	RW	0	V_{REFBUF} enable The software writes 0 and sets 0, writes 1 and sets 1, 0: V_{REFBUF} disabled 1: V_{REFBUF} enabled
4	ADSTP	RS	0	ADC stop conversion command This bit is set by software to stop and discard an ongoing conversion (ADSTP Command). It is cleared by hardware when the conversion is effectively discarded and the ADC is ready to accept a new start conversion command. 0: No ADC stop conversion command ongoing 1: Write 1 to stop the ADC. Read 1 means that an ADSTP command is in progress.
3	Reserved	-	0	-
2	ADSTART	RS	0	ADC start conversion command

Bit	Name	R/W	Reset Value	Function
				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). It is cleared by hardware: 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: Software is allowed to set ADSTART only when ADEN=1 and ADDIS=0. (ADC is enabled and there is no pending request to disable the ADC)
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 command ongoing 1: Write 1 to disable the ADC. Read 1 means that an ADDIS command is in progress. Note: Setting ADDIS to ’1’ is only effective when ADEN=1 and ADSTART=0 (which ensures that no conversion is ongoing)
0	ADEN	RS	0	ADC enable command This bit is set by software to enable the ADC. 0: ADC is disabled 1: ADC is enabled

13.4.4. 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	DISCEN
-	-	RW				-	-	RW	RW	-	-	-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	WAIT	CONT	OVRMOD	EXTEN[1:0]		Res	EXTSEL			ALIGN	RESSEL		SCANDIR	Res	Res
-	RW	RW	RW	RW		-	RW			RW	-	-	RW	-	-

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	-
29:26	AWDCH[3:0]	RW	4’h0	Analog watchdog channel selection. These bits are set and cleared by software. They select the input channel to be guarded by the analog watchdog. 00000: ADC analog input Channel 1 00001: ADC analog input Channel 2 0010: ADC analog input Channel 3 … 1001: ADC analog input Channel 10 Note: ADC analog input channel 0 does not support single channel analog watchdogs. Others: Reserved Note: The channel selected by the AWDCH[3:0] bits must be also set into the CHSELR register.

Bit	Name	R/W	Reset Value	Function
				Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
25:24	Reserved	-	-	-
23	AWDEN	RW	0	Analog watchdog enable This bit is set and cleared by software. 0: Analog watchdog disabled 1: Analog watchdog enabled 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 This bit is set and cleared by software to enable the analog watchdog on the channel identified by the AWDCH[3:0] bits or on all the channels 0: Analog watchdog enabled on all channels 1: Analog watchdog enabled on a single channel Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
21:17	Reserved	-	-	-
16	DISCEN	RW	0	Discontinuous mode 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. Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
15	Reserved	-	-	-
14	WAIT	RW	0	Wait mode conversion This bit is set and cleared by software to enable/disable wait conversion mode. 0: Wait conversion mode off 1: Wait conversion mode on 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. Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
12	OVRMOD	RW	0	Overrun 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. Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
11:10	EXTEN[1:0]	RW	2'h0	External trigger enable and polarity selection These bits are set and cleared by software to select the external trigger polarity and enable the trigger 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 Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
9	Reserved	-	-	-
8:6	EXTSEL[2:0]	RW	3'h0	External trigger selection These bits select the external event used to trigger the start of conversion 000:TRG0(TIM1_TRG0)

Bit	Name	R/W	Reset Value	Function
				001:TRG1(TIM1_CC4) 010: Reserved 011: Reserved 100: Reserved 101: Reserved 110: Reserved 111: 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 Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
4:3	RESSEL[1:0]	RW	2'h0	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 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 10) 1: Backward scan (from channel 10 to 0) Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
1:0	Reserved	-	-	-

13.4.5. ADC configuration register 2 (ADC_CFGR2)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	CKMODE [3:0]:	RW	4'h0	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 1000: HSI 1001:HSI/2 1010:HSI/4 1011:HSI/8 1100:HSI/16 1101:HSI/32 1110:HSI/64 Others: Software is allowed to write these bits only when the ADC is disabled (ADCAL=0, ADSTART=0, ADSTP=0, ADDIS=0 and ADEN=0).
27:0	Reserved	-	-	-

13.4.6. 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	Res	Res	Res	Res	Res	Res	Res	Res	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	SMP		
-													RW		

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	-
2:0	SMP[2:0]	RW	3'h0	Sampling time selection These bits are written by software to select the sampling time that applies to all channels. 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 AD-START=0 (which ensures that no conversion is ongoing).

13.4.7. 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	-	-	-
27:16	HT[11:0]	RW	12'hFFF	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	-	-	-
11:0	LT[11:0]	RW	12'h0	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).

13.4.8. 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															
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	-
9	CHSEL9	RW	0	Channel 9 (V_{REFINT}) selection 0: Channel is not selected 1: Channel is selected Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
8	CHSEL8	RW	0	Channel 8 (T_{S_VIN}) selection 0: Channel is not selected 1: Channel is selected Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
7:0	CHSELx	RW	8'h0	Channel selection These bits are written by software and define which channels are part of the sequence of channels to be converted. 0: Input Channel-x is not selected for conversion (x=0-7) 1: Input Channel-x is selected for conversion (x=0-7) Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).

13.4.9. ADC data register (ADC_DR)

Address offset: 0x40

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:0	DATA[15:0]	R	16'h0	Converted data These bits are read-only. They contain the conversion result from the last converted channel. The data are left- or right-aligned.

13.4.10. 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.	CALSUC	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
Res	CALBYP	CALSMP[1:0]		CALSEL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
	RC_W1	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	CALSUC	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, CALSUC = 1: Invalid status CALON = 0, CALSEL = 0, CALSUC = 0: Capacitance calibration is not performed CALON = 0, CALSEL = 1, CALSUC = 1: ADC capacitance calibration successful

Bit	Name	R/W	Reset Value	Function
				CALON = 0, CALSEL = 1, CALSUC = 0: ADC capacitor calibration failed
29	OFFSUC	RC_W1	0	OFFSET calibration status bit. Indicates whether the ADC OFFSET calibration is successful. Hardware set 1. The software writes 1 and sets 0. CALON = 0, CALSEL = 0, OFFSUC = 0: ADC OFFSET Calibration failed CALON = 0, CALSEL = 0, OFFSUC = 1: ADC OFFSET calibration successful CALON = 0, CALSEL = 1, OFFSUC = 1: ADC OFFSET Calibration successful CALON = 0, CALSEL = 1, OFFSUC = 0: ADC OFFSET Calibration failed
28:15	Reserved	-	-	-
14	CALBYP	RC_W1	0	Calibration factor bypass. When ADCAL is 0, the software writes 1 to 1. When ADCAL is active or ADSTART is active, the hardware is set to 0. 1: The calibration result is the reset value 0: The calibration result is self-calibration factor or factory calibration factor
13:12	CALSMP[1:0]	RW	2'h0	Calibration sampling time selection Configure the number of clock cycles of the sampling phase of the calibration based on the following information: 00: 2 ADC clock cycles 01:4 ADC clock cycles 10: 8 ADC clock cycles 11:1 ADC clock cycle The longer the SMP is configured during calibration, the more accurate the calibration result is, but this configuration will bring the problem of longer calibration cycle
11	CALSEL	RW	0	Calibration content selection bit for selecting what needs to be calibrated 1: Calibrate OFFSET and capacitance 0: Calibrate OFFSET only
10:0	Reserved	-	-	-

13.4.11. 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	Res	Res	Res	Res	Res	Res	Res
-	-	-		-	-	-	-	-	-	-	-	-	-	-	-

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	V_{REFINT} enable 0: Disabled 1: Enabled Software is allowed to write these bits only when ADSTART=0 (which ensures that no conversion is ongoing).
21:0	Reserved	-	-	-

14. Comparator (COMP)

14.1. 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:

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

14.2. COMP main features

- Each comparator has configurable positive or negative inputs for flexible voltage selection
  - Multiple I/O pins
  - V_{REFCMP} (V_{REFBUF}/supply voltage 16-step voltage division)
- The output can be connected to the input of I/O or timer as a trigger
  - OCREF_CLR event (cycle by cycle current control)
  - Break events for fast PWM shutdowns
- COMP1 and COMP2 comparators can be combined in a window comparator.
- Each comparator has interrupt generation capability with wake-up from Sleep and Stop modes (through the EXTI controller)
- Provides software to configure the digital filtering time to enhance the anti-interference capability of the device

14.3. COMP functional description

14.3.1. COMP block diagram

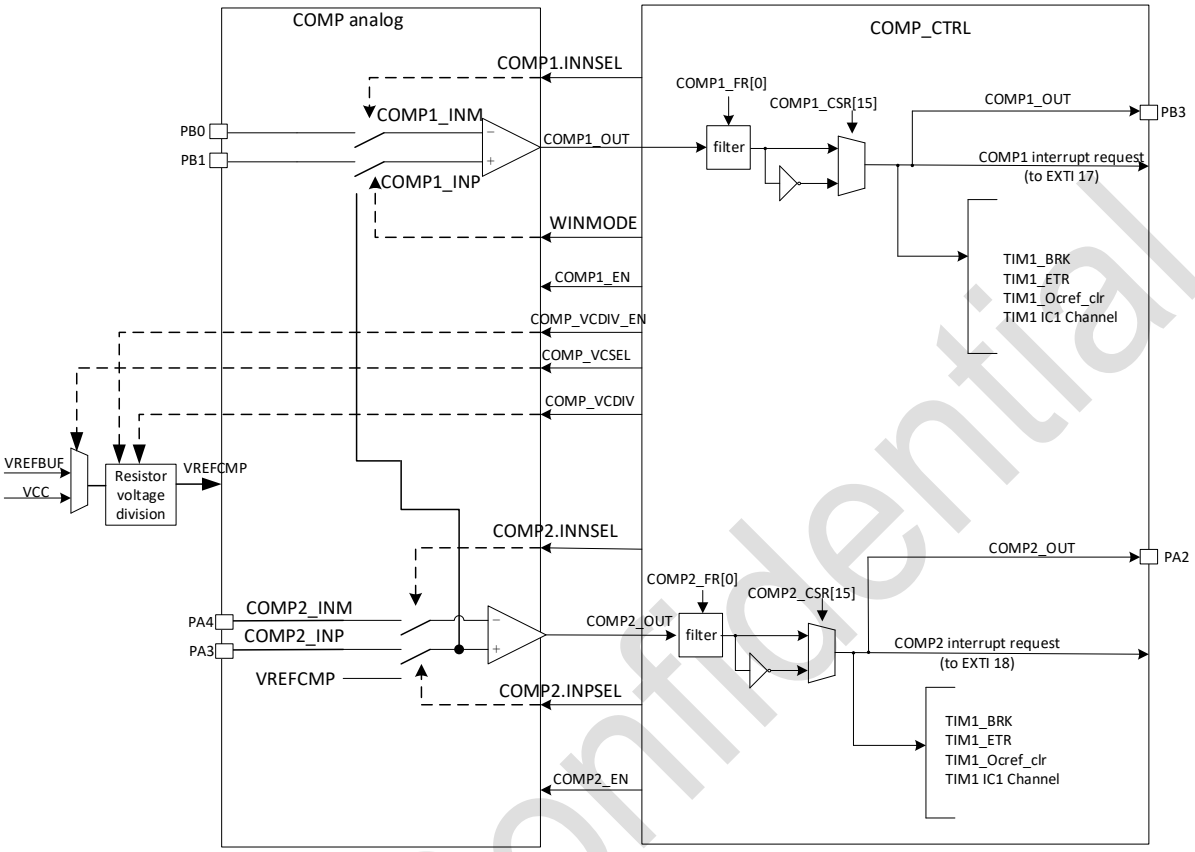

Figure 14-1 Comparator 1 and 2 block diagrams

14.3.2. 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 signals, using BKIN
- OCREF_CLR event (cycle by cycle current control)
- Input capture for timing measures

14.3.3. COMP reset and clocks

The COMP module has two clock sources:

PCLK (APB clock), used to provide clock to the configuration register

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.

The reset signal sources of COMP module are: APB reset source and COMP module software reset source

- APB reset, for resetting COMP registers
- COMP software reset, used to reset the circuit after analog comparator output (latch circuit of analog

output, filter circuit, etc.)

14.3.4. Window comparator

The function of the Window comparator is to monitor whether the analog voltage is within the threshold range.

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.

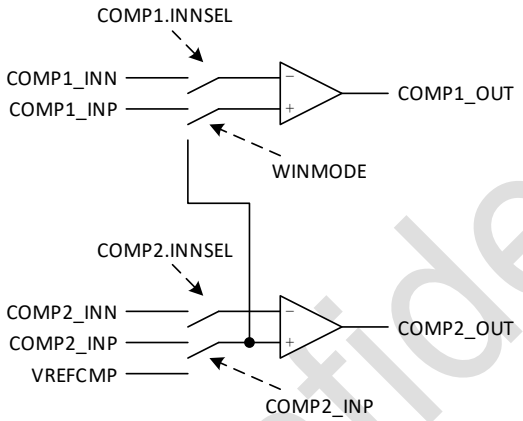

Figure 14-2 Window comparator

14.3.5. Low-power modes

Mode	Description
Sleep	There was no effect on COMP. The comparator interrupt may cause the device to exit the Sleep mode

14.3.6. Comparator filtering

If the working environment of the chip 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.FLTCNTx [15: 0] (x = 1, 2) 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 shown in figure below:

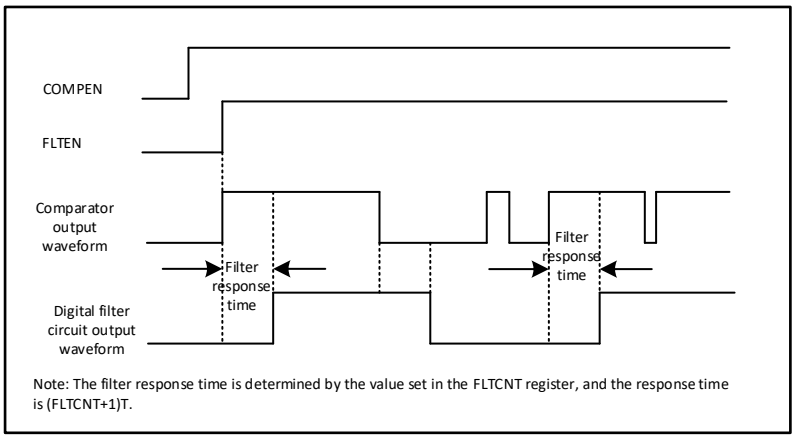

Figure 14-3 COMP filtering

14.3.7. COMP interrupts

The comparator outputs are internally connected to the Extended interrupts and events controller. Each comparator has its own EXTI line (17 and 18) and can generate either interrupts or events. The same mechanism is used to exit from low-power modes.

14.3.8. COMP select V_{REFCMP} configuration

When the comparator selects V_{REFCMP} as the non-inverting input of the comparator, the software configures the V_{REFCMP} flow

When V_{REFCMP} selects V_{REFBUF} as the reference voltage source:

1. Configure the VREFEN bit in the ADC_CCR of the ADC module and turn on the V_{REFBUF} reference voltage
2. Configure Vrefbuff_sel [1: 0] in ADC_CR of ADC module, and select V_{REFBUF} gear
3. Configure Vrefbuff_en in ADC_CR of ADC module to enable V_{REFBUF} voltage
4. Configure COMP_VCSEL in COMP1_CSR to 0, and select V_{REFBUF}
5. Configure COMP_VCDIV_EN in COMP1_CSR to enable V_{REFCMP}
6. Configure COMP_VCDIV [3: 0] in COMP1_CSR and select the voltage dividing gear of V_{REFCMP}

When V_{REFCMP} selects V_{CC} as the reference voltage source:

1. Configure COMP_VCSEL in COMP1_CSR to 1, and select V_{CC}
2. Configure COMP_VCDIV_EN in COMP1_CSR to enable VREFCMP
3. Configure COMP_VCDIV [3: 0] in COMP1_CSR and select the voltage dividing gear of V_{REFCMP}

14.4. COMP registers

14.4.1. 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	Res		COMP_VCSEL	COMP_VCDIV_EN	COMP_VCDIV[3:0]				Res					
-	R	-		RW	RW	RW				-					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	Res			WINMODE	Res	Res				INNSEL	Res				COMP1_EN
RW	-			RW	-	-				RW	-				RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30	COMP_OUT	R	0	COMP1 output This read-only flag reflects the level of the comparator 1 output before the polarity selector.
29:28	Reserved	-	-	-
27	COMP_VCSEL	RW	0	V_{REFCMP} reference voltage source selection. 0: V_{REFBUF}, when selecting V_{REFBUF}, you need to make VREFINT_EN. 1: V_{CC}, V_{REFINT}, and V_{REFBUF} are closed at COMP_VCSEL = 1.
26	COMP_VCDIV_EN	RW	0	V_{REFCMP} enabled, high active.
25:22	COMP_VCDIV[3:0]	RW	4'h7	V_{REFCMP} partial pressure selection 0: 1/16 1: 2/16 2: 3/16 3: 4/16 4: 5/16 5: 6/16 6: 7/16 7: 8/16 8: 9/16 9: 10/16 10: 11/16 11: 12/16 12: 13/16 13: 14/16 14: 15/16 15: 16/16
21:16	Reserved	-	-	-
15	POLARITY	RW	0	COMP1 output polarity selection These bits can be read and written by software. 0: Non-inverted 1: Inverted
14:12	Reserved	-	0	-
11	WINMODE	RW	0	COMP1 window mode enable 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)
10:6	Reserved	-	-	-
5	INNSEL	RW	0	Comparator 1 inverting input selection 0: COMP1_INM from PB0 1: COMP1_INM from PB1
4:1	Reserved	-	-	-
0	COMP1_EN	RW	0	Comparator 1 enable These bits can be read and written by software. 0: Disabled 1: Enabled

14.4.2. Comparator 1 filtering register (COMP1_FR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1	RW	16'h0	Comparator 1 Sampling Filter Counter

Bit	Name	R/W	Reset Value	Function
				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	-	-	-
0	FLTEN1	RW	0	Comparator 1 digital filter function configuration 0: Disable digital filtering function 1: Enable digital filtering function Note: This bit must be set when COMP1_EN is 0

14.4.3. 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	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	R	-													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	Res	Res	Res	Res	Res	INPSEL	Res	Res	Res	INNSEL	Res	Res	Res	Res	COMP2_EN
RW	-	-	-	-	-	RW	-	-	-	RW	-	-	-	-	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	-
30	COMP_OUT	R	0	COMP2 output This read-only flag reflects the level of the comparator 2 output before the polarity selector.
29:16	Reserved	-	-	-
15	POLARITY	RW	0	COMP2 polarity selection 0: Non-inverted 1: Inverted
14:10	Reserved	-	-	-
9	INPSEL	RW	0	Comparator 2 non-inverting input selection 0: PA3 1:V_{REFCMP}
5	INNSEL	RW	0	Comparator 2 inverting input selection 0: PA4 1: PA3
4:1	Reserved	-	-	-
0	COMP2_EN	RW	0	Comparator 2 enable Software readable and writable (if not locked) 0: Disabled 1: Enabled

14.4.4. 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															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FLTEN2
-															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2 [15:0]	RW	16'h0	Comparator 2 sampling filter counter The sampling clock is APB or LSI or LSE. The filter count value is configurable. When the number of samples reaches the filter count value, the results are output consistently. Sample count cycle = FLTCNT [15: 0]
15:1	Reserved	-	-	-

0	FLTEN2	RW	0	Comparator 2 digital filter function configuration 0: Disable digital filtering function 1: Enable digital filtering function Note: This bit must be set when COMP2_EN is 0

15. Advanced-control timers (TIM1)

15.1. TIM1 introduction

The advanced-control timer (TIM1) consists of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare, PWM, complementary PWM with dead-time insertion).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

The advanced-control (TIM1) and general-purpose (TIMx) timers are completely independent, and do not share any resources. They can be synchronized together.

15.2. TIM1 main features

- 16-bit up, down, up/down auto-reload counter
- 16 programmable prescaler that allows the clock frequency of the counter to be divided from 1 to 65,536
- 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

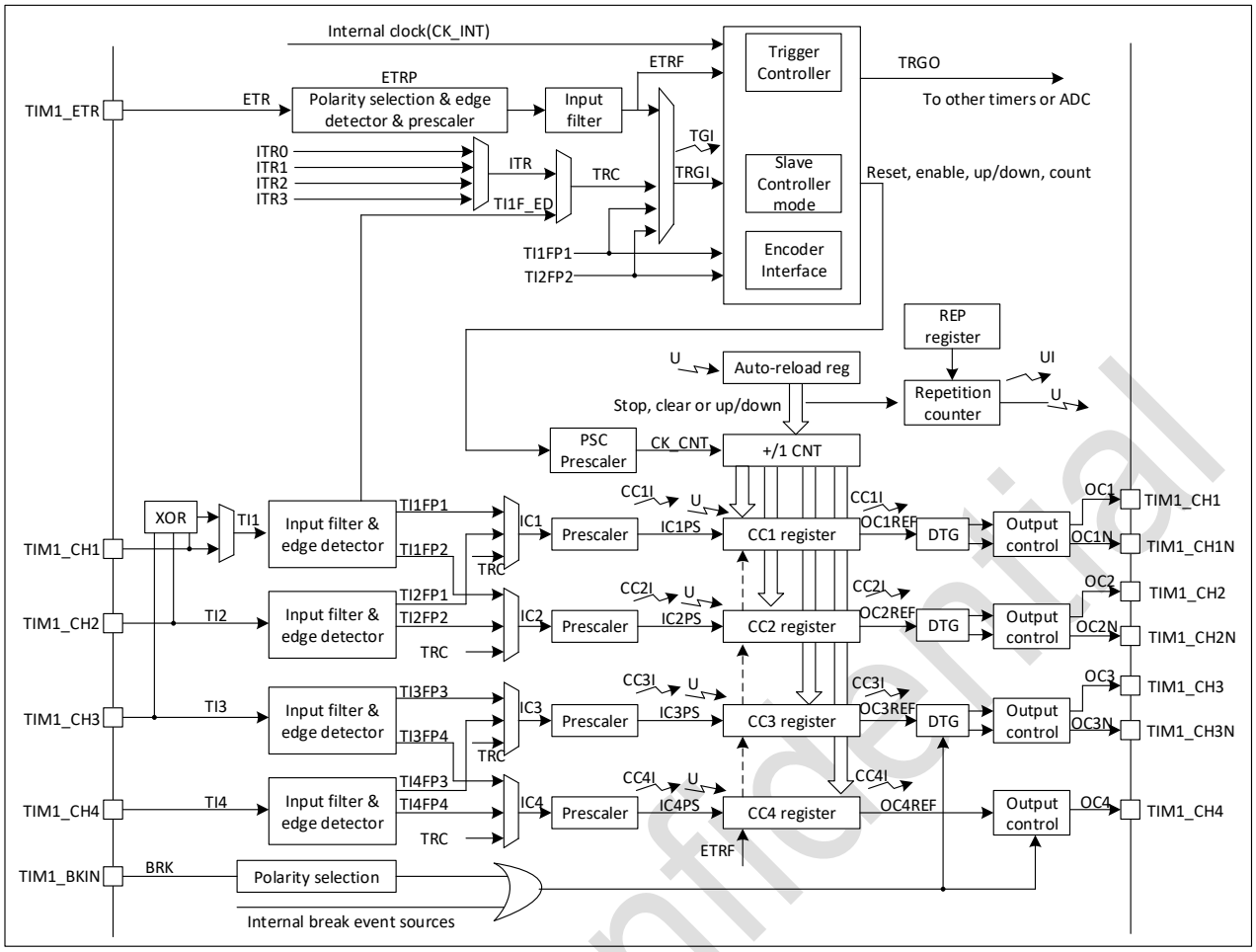

Figure 15-1 Advanced-control timer block diagram

15.3. TIM1 functional description

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

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM1_CR1 register is set.

Attention that the counter starts counting 1 clock cycle after setting the CEN bit in the TIM1_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 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:

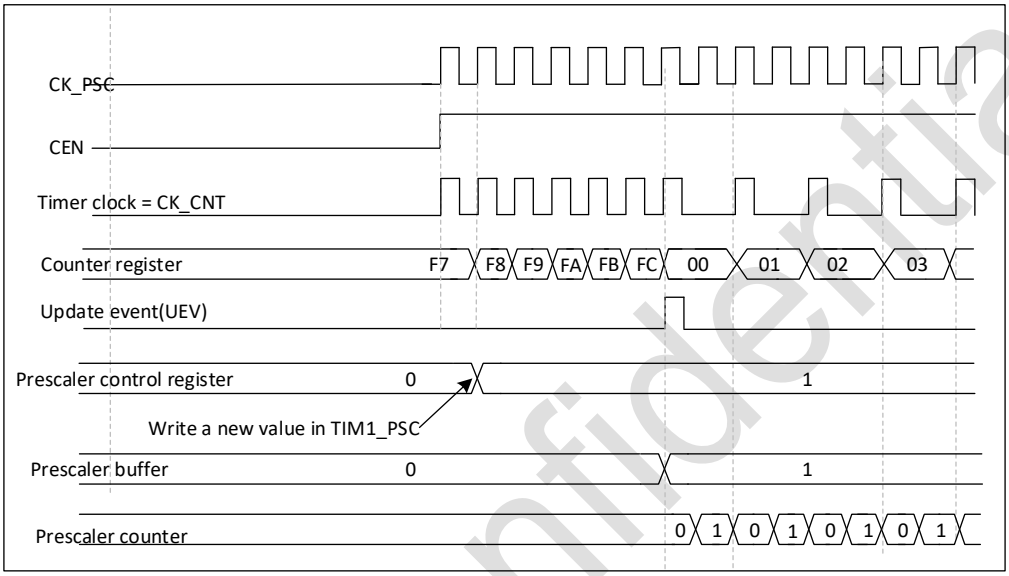

Figure 15-2 Counter timing diagram with prescaler division change from 1 to 2

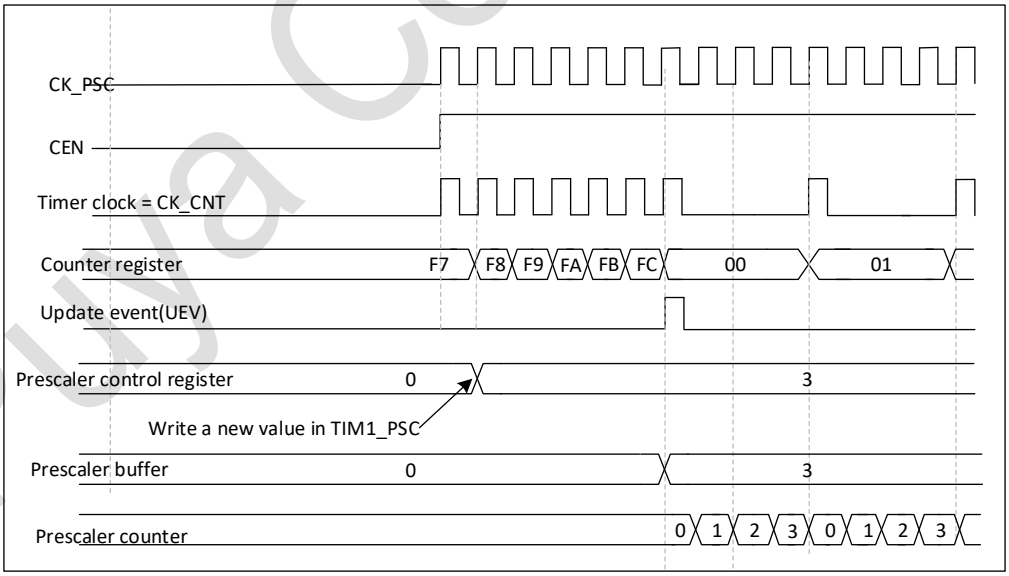

Figure 15-3 Counter timing diagram with prescaler division change from 1 to 4

15.3.2. Timer enable

Upcounting mode

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

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

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

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescaler rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMx_RCR register.
- The auto-reload shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

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

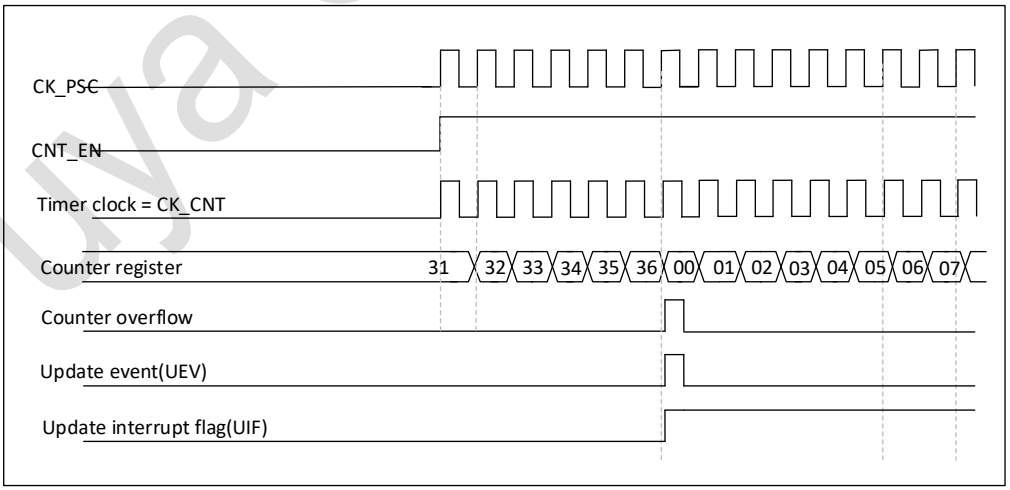

Figure 15-4 Counter timing diagram, internal clock divided by 1

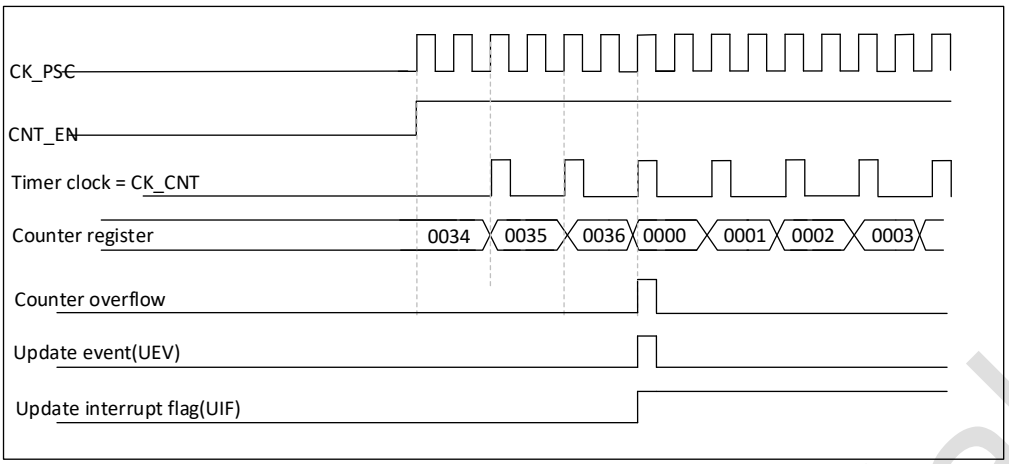

Figure 15-5 Counter timing diagram, internal clock divided by 2

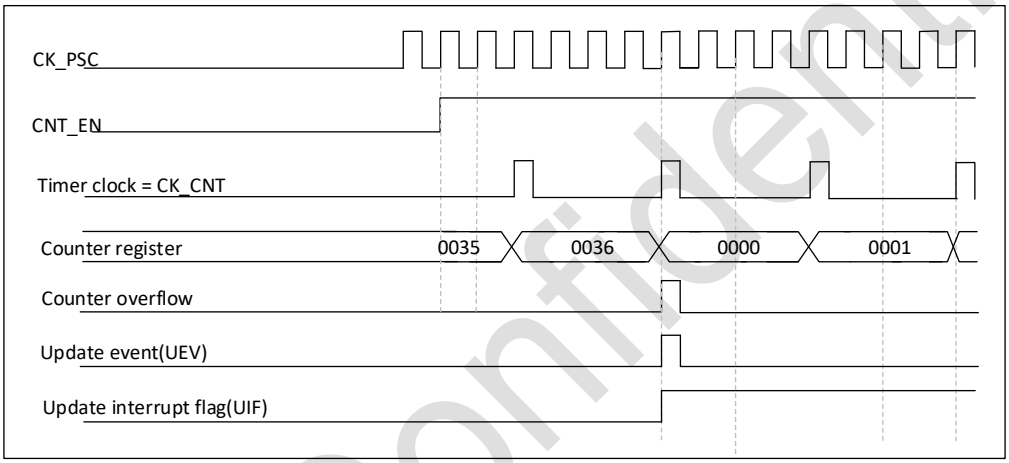

Figure 15-6 Counter timing diagram, internal clock divided by 4

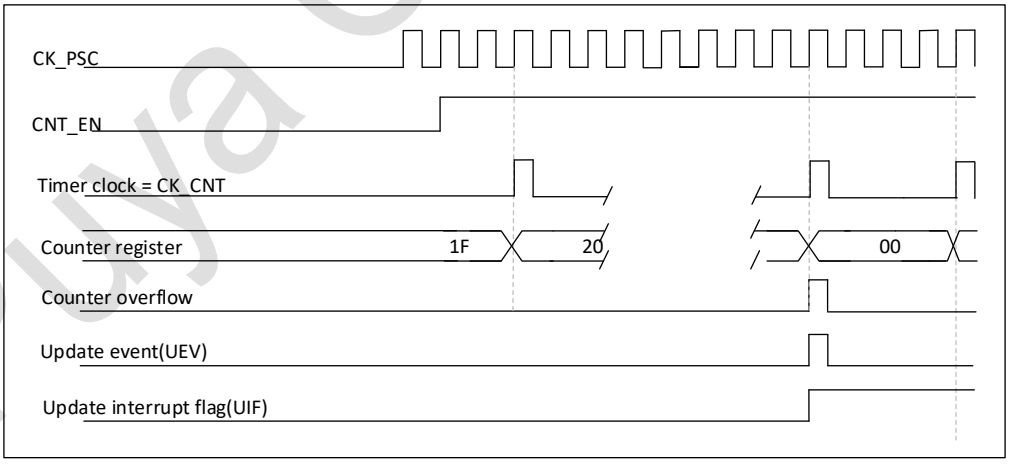

Figure 15-7 Counter timing diagram, internal clock divided by N

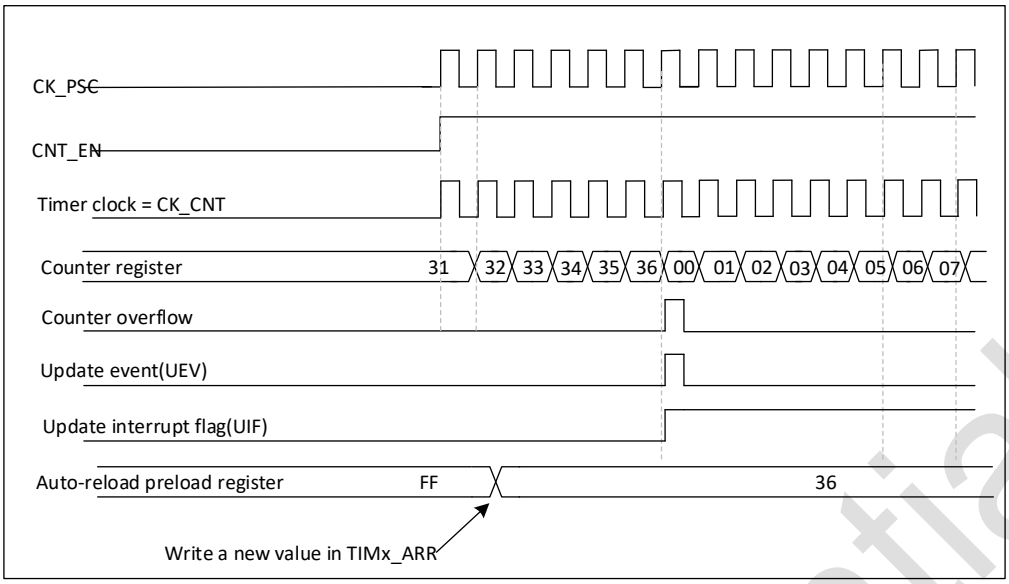

Figure 15-8 Counter timing diagram, update event when ARPE=0 (TIM1_ARR not preloaded)

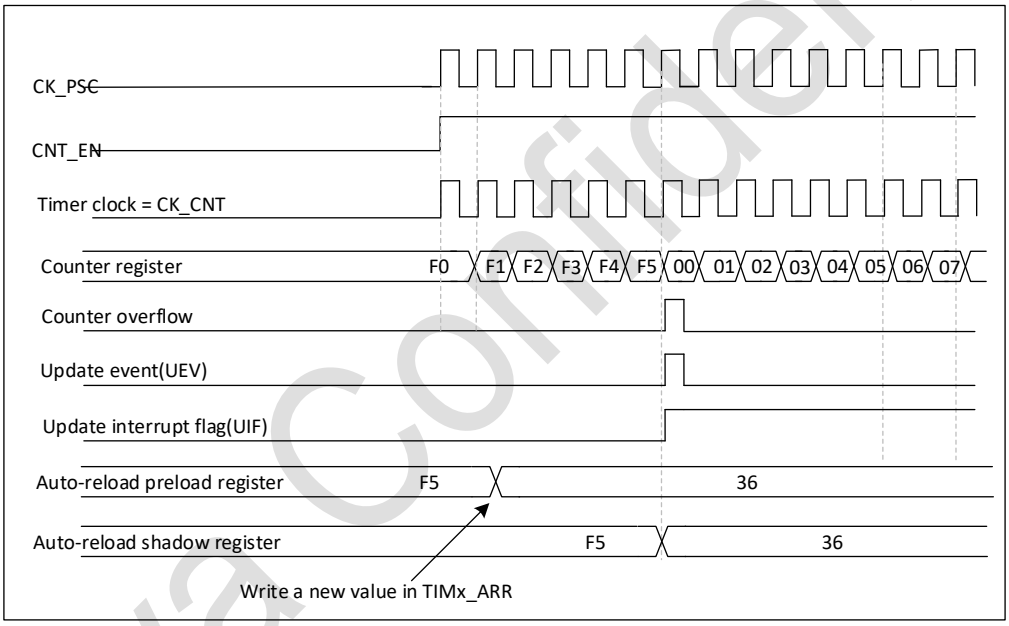

Figure 15-9 Counter timing diagram, update event when ARPE=1 (TIM1_ARR preloaded)

Downcounting mode

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

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

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

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescaler rate doesn’t change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an 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.

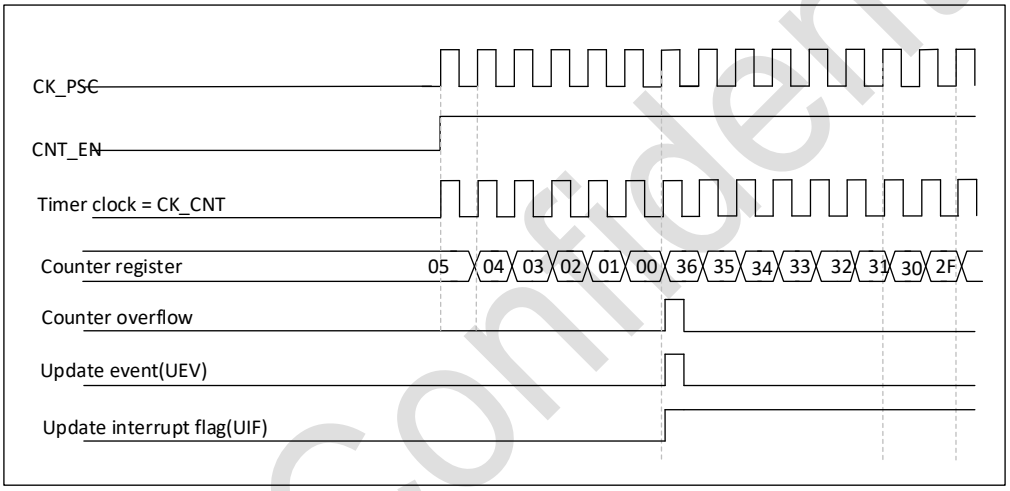

Figure 15-10 Counter timing diagram, internal clock divided by 1

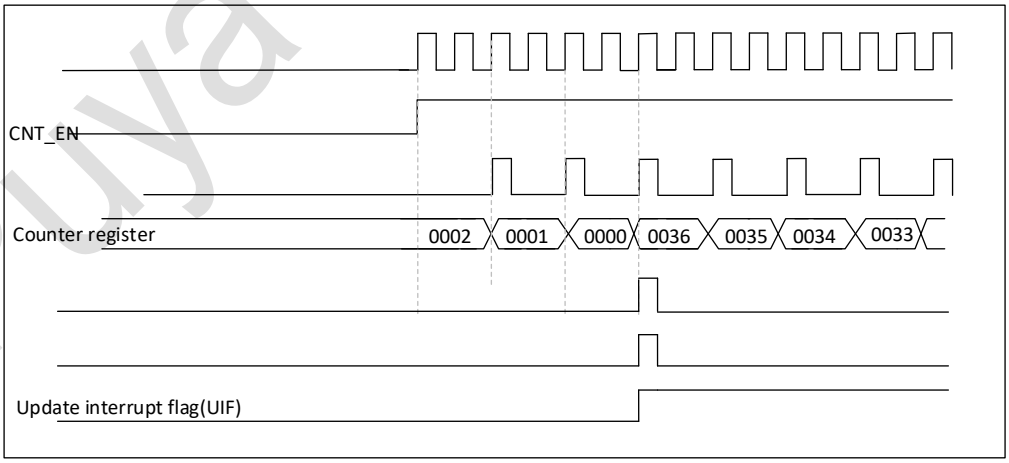

Figure 15-11 Counter timing diagram, internal clock divided by 2

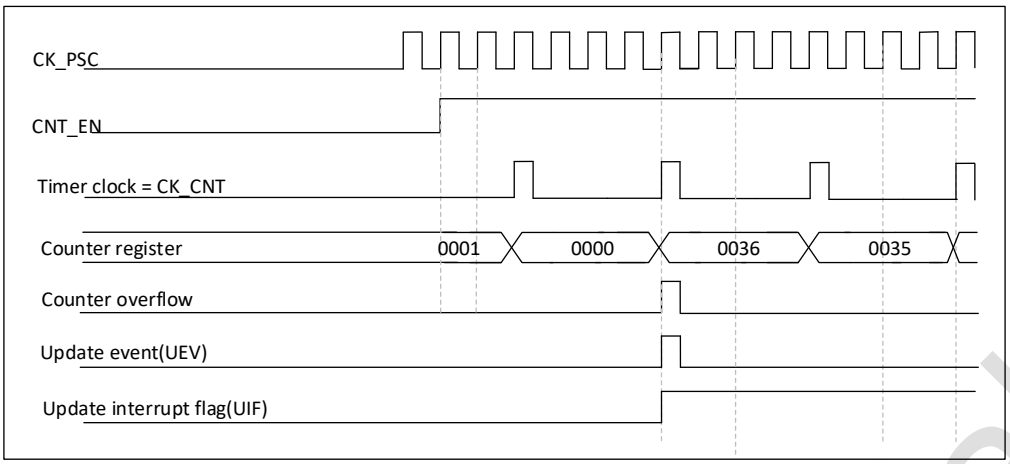

Figure 15-12 Counter timing diagram, internal clock divided by 4

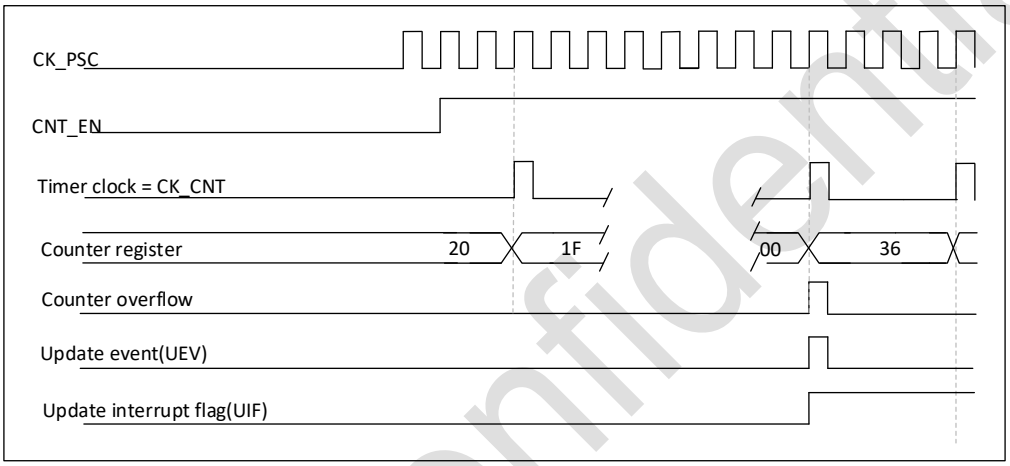

Figure 15-13 Counter timing diagram, internal clock divided by N

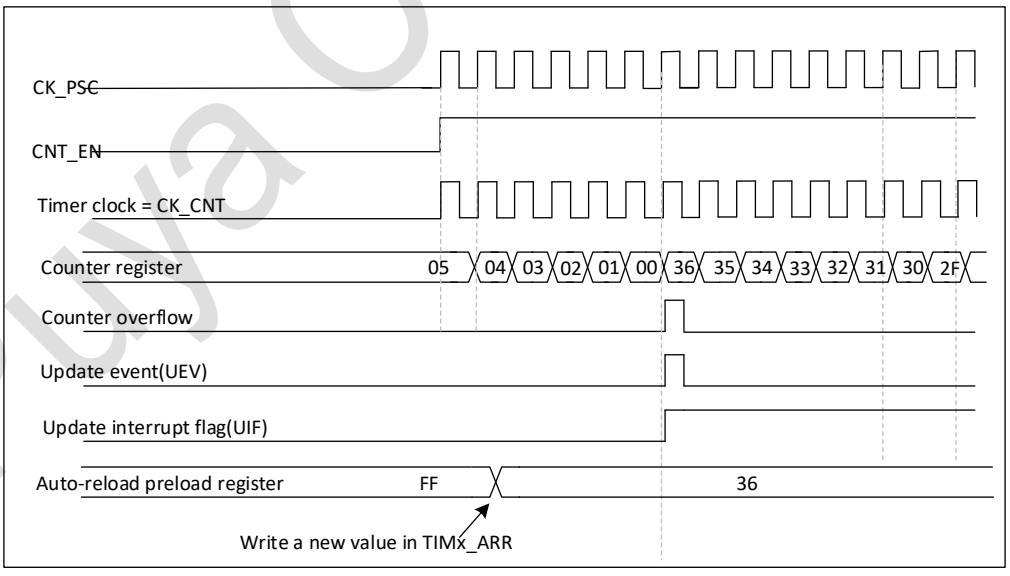

Figure 15-14 Counter timing diagram, update event when repetition counter is not used

Center-aligned mode (up/down counting)

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

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

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

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event. In this case, the counter restarts counting from0, as well as the counter of the prescaler.

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt 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).

The following figures show some examples of the counter behavior for different clock frequencies.

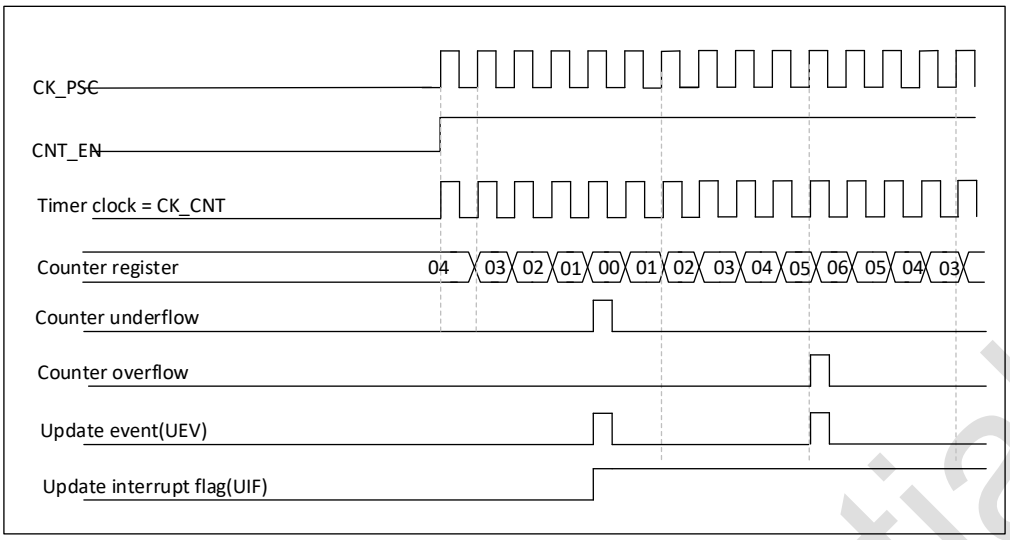

Figure 15-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x6

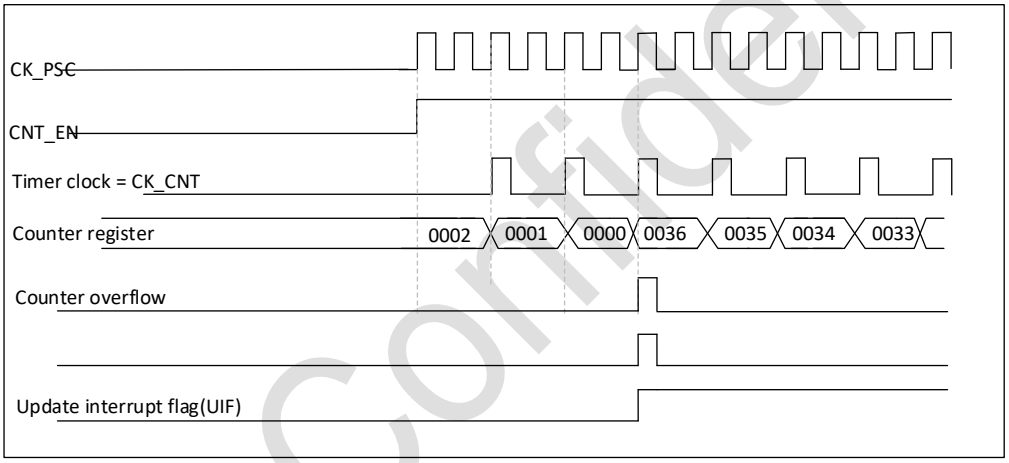

Figure 15-16 Counter timing diagram, internal clock divided by 2, TIMx_ARR=0x36

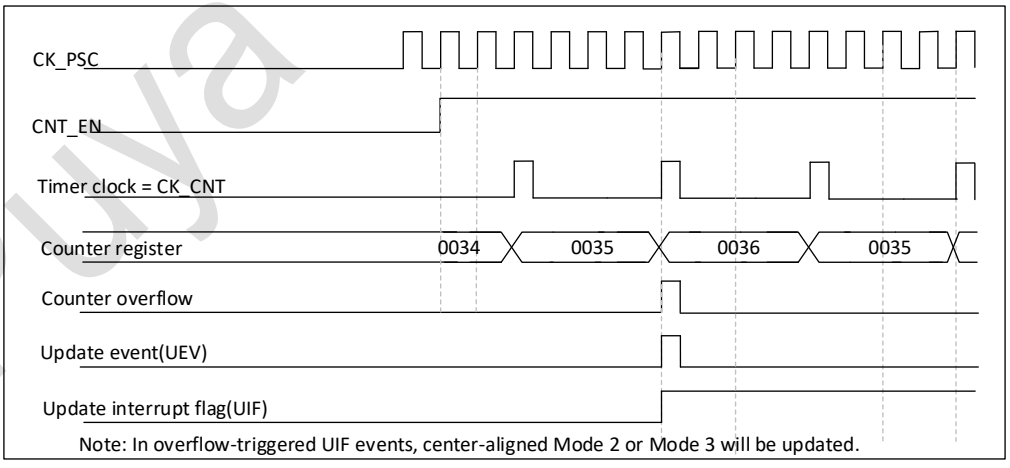

Figure 15-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x36

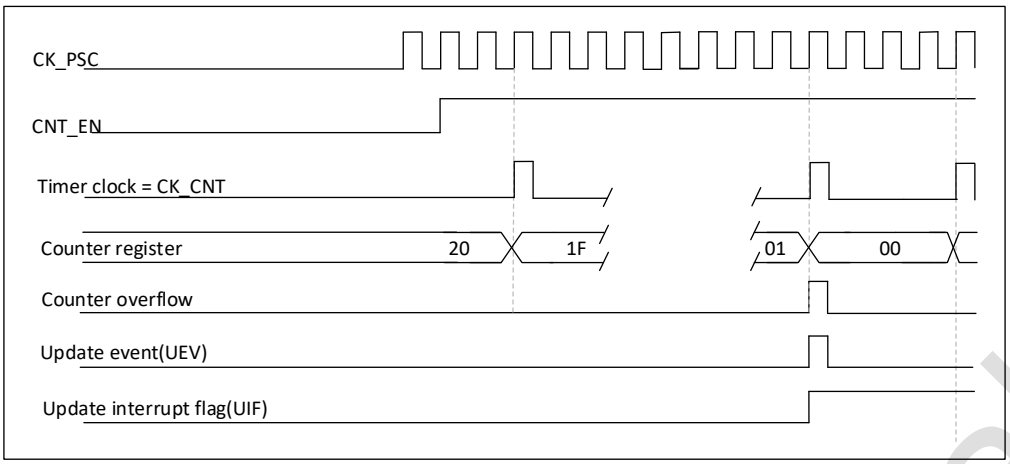

Figure 15-18 Counter timing diagram, internal clock divided by N

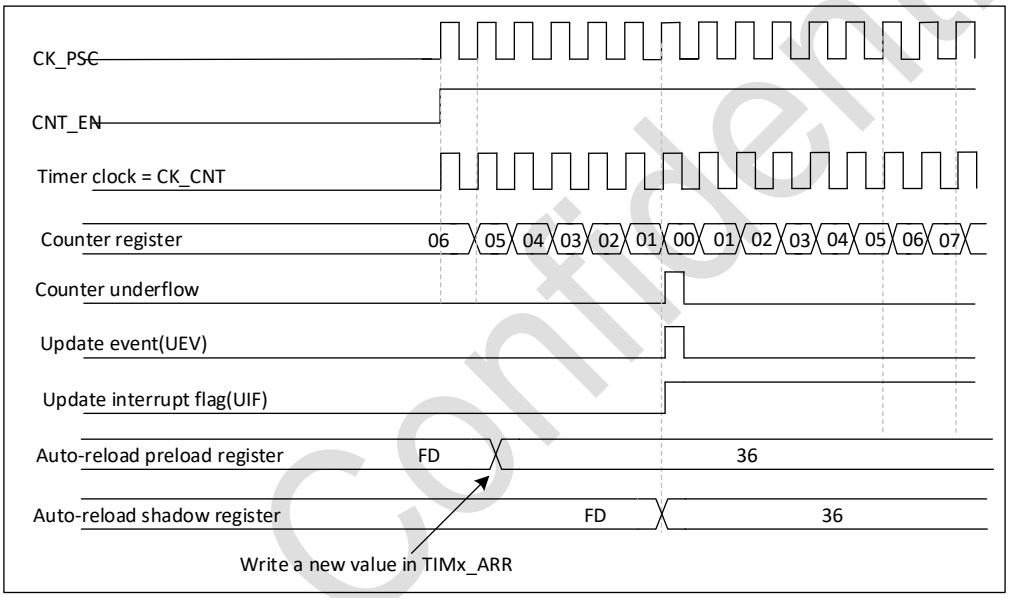

Figure 15-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

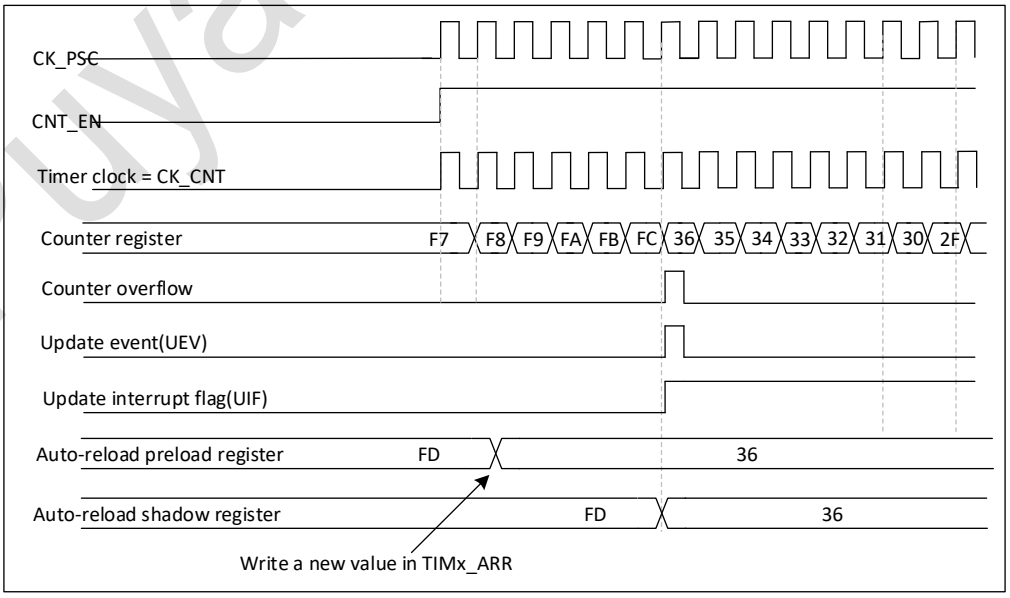

Figure 15-20 Counter timing diagram, Update event with ARPE=1 (counter overflow)

15.3.3. Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter over-flows/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. .

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

In center-aligned mode, for odd values of RCR, the update event occurs either on the overflow or on the underflow depending on when the RCR register was written and when the counter was started. If the RCR was written before starting the counter, the UEV occurs on the overflow. For example for RCR = 3, the UEV is generated on each 4th overflow or underflow event depending on when RCR was written.

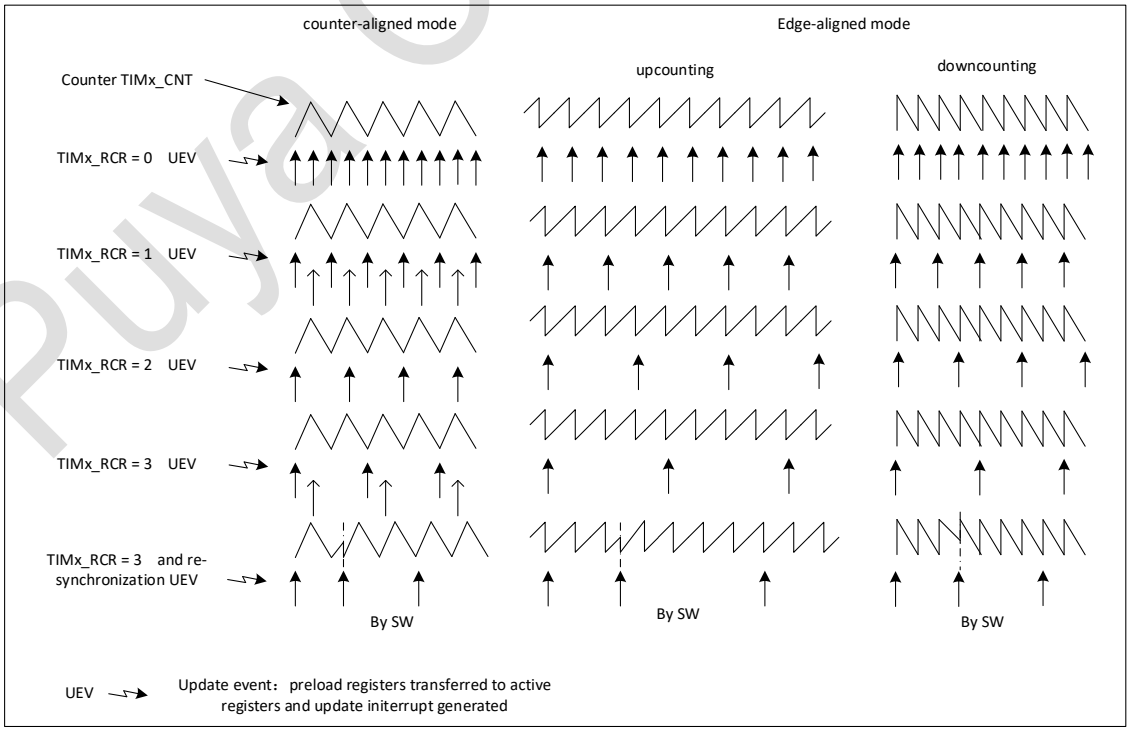

Figure 15-21 Update rate examples depending on mode and TIM1_RCR register settings

15.3.4. 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. .

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_EGR register) are actual control bits and can be changed only by software. As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

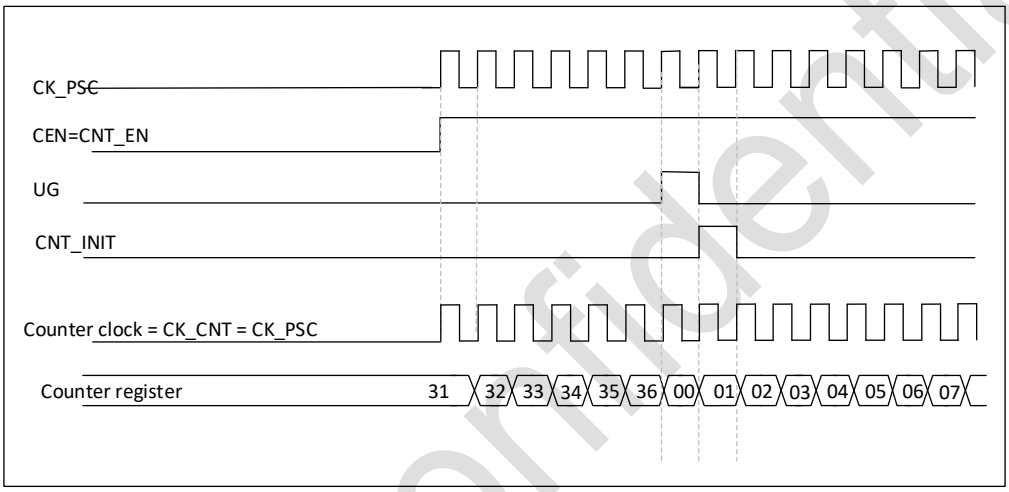

Figure 15-22 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

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

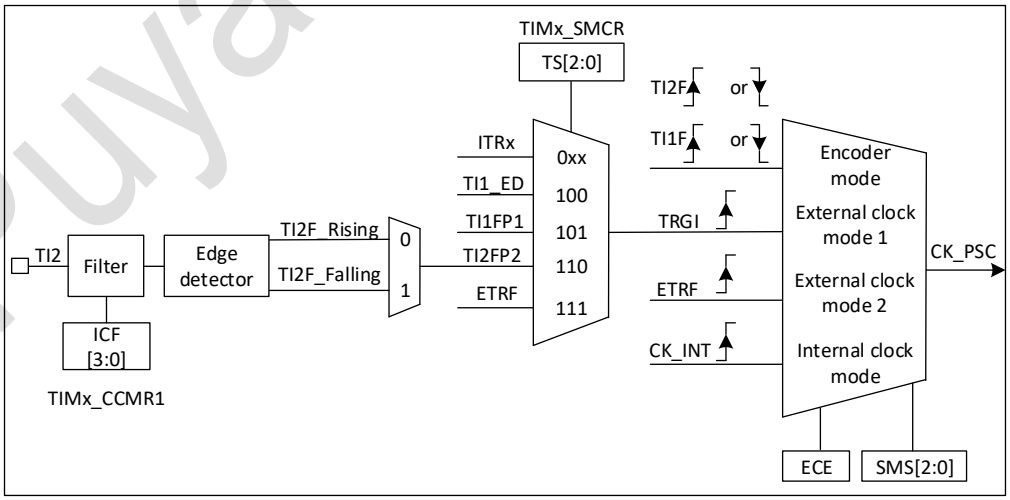

Figure 15-23 TI2 external clock connection example

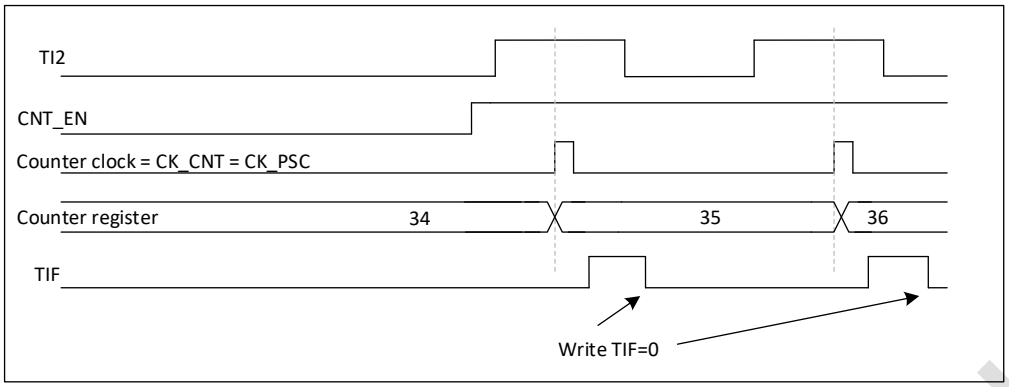

Figure 15-24 Control circuit in external clock mode 1

External clock source mode 2

This mode is selected by writing ECE=1 in the TIMx_SMCR register. The counter can count at each rising or falling edge on the external trigger input ETR.

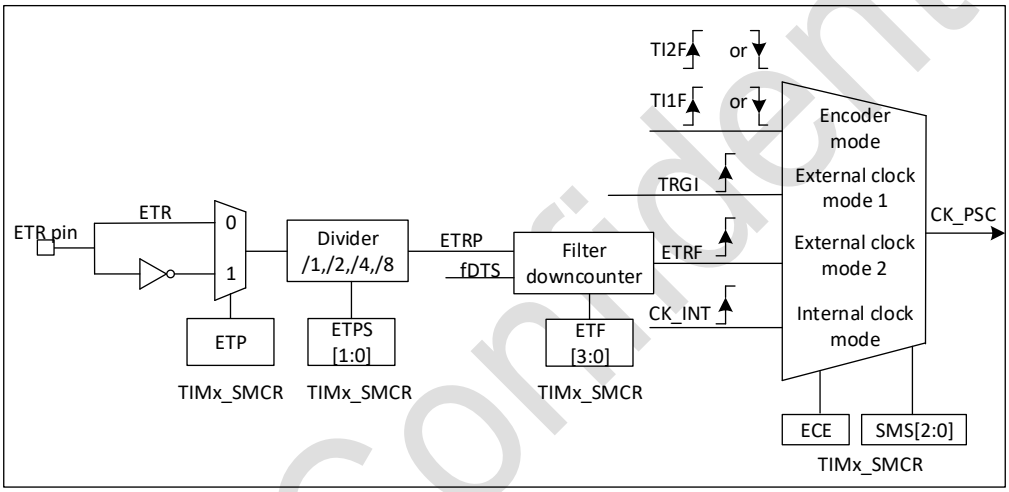

Figure 15-25 TI2 external trigger input block

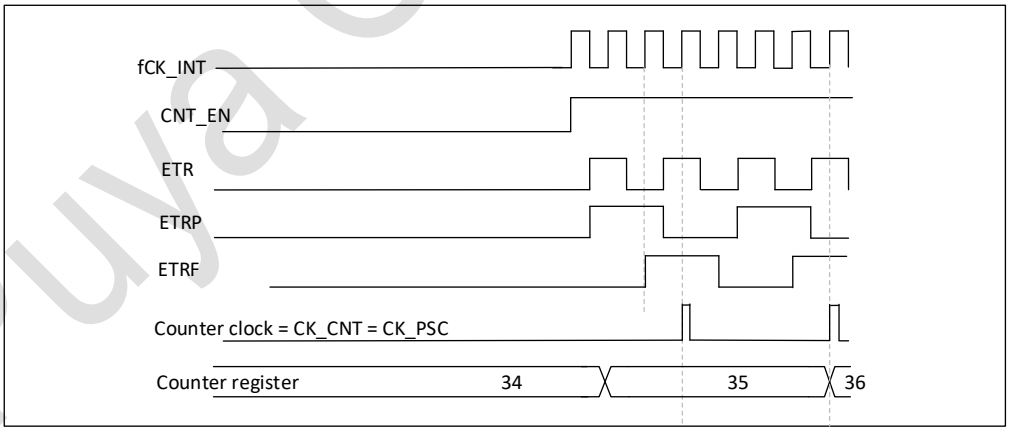

Figure 15-26 Control circuit in external clock mode 2

15.3.5. 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).

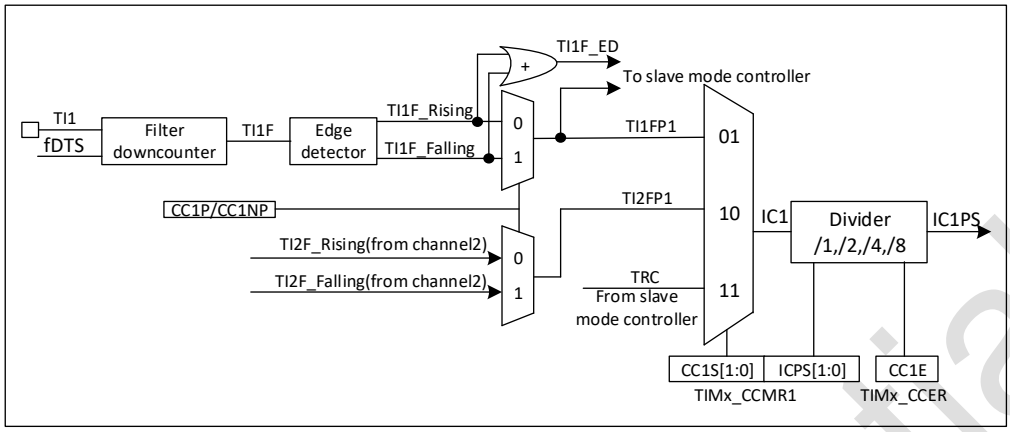

Figure 15-27 Capture/compare channel (example: channel 1 input stage)

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.

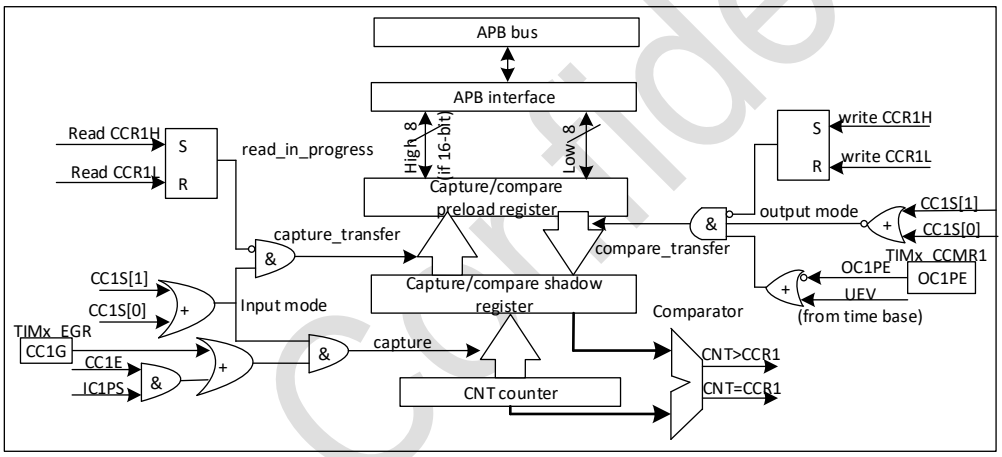

Figure 15-28 Capture/compare channel 1 main circuit

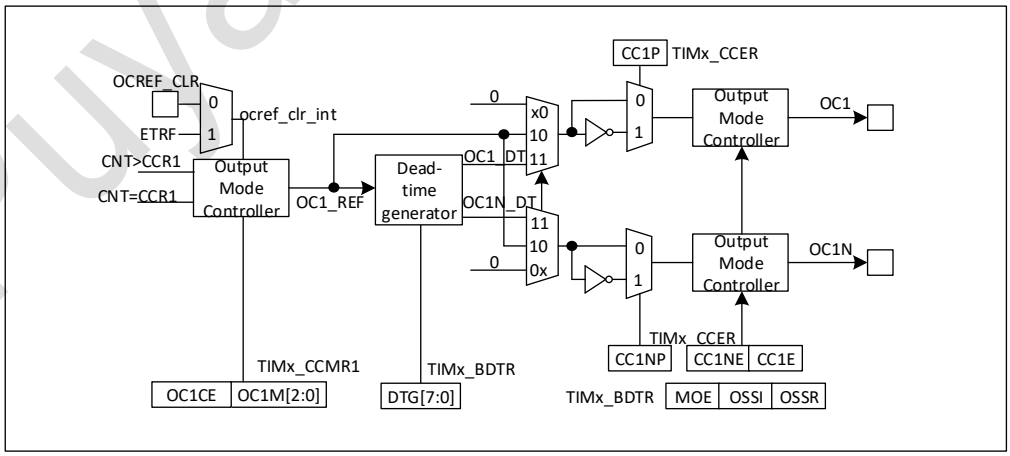

Figure 15-29 Output stage of capture/compare channel (channel 1 to 3)

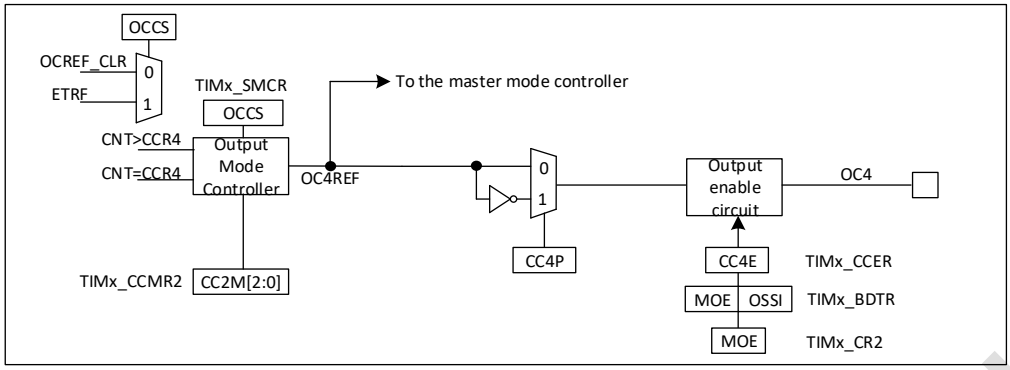

Figure 15-30 Output stage of capture/compare channel (channel 4)

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.

15.3.6. 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 $_{f}$DTS 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.

15.3.7. PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode.

For example, you can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to ‘0’ (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2Pbit to ‘1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

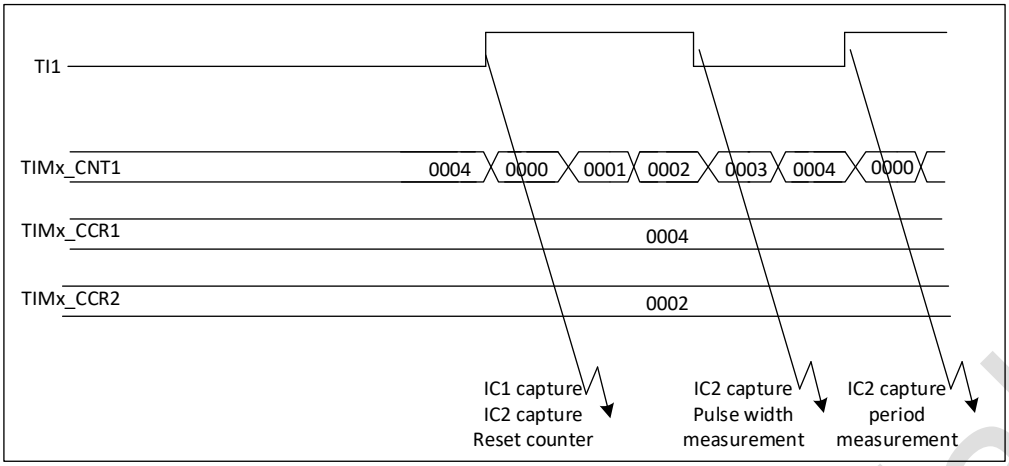

Figure 15-31 PWM input mode timing

15.3.8. 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.

15.3.9. 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
   - 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.

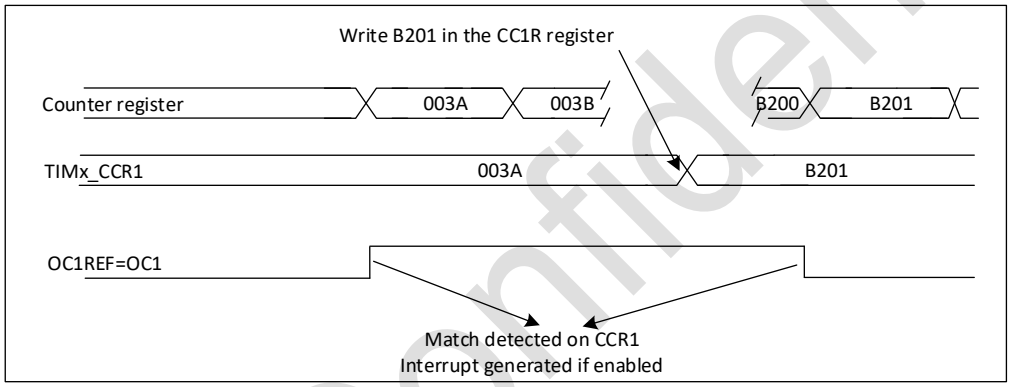

Figure 15-32 Output compare mode, toggle on OC1

15.3.10. PWM mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIMx_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, all registers must be initialized by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by a combination of the CCxE, CCxNE, MOE, OSSI and OSSR bits (TIMx_CCER and TIMx_BDTR registers). Refer to the TIMx_CCER register description for more details.

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤ TIMx_CNT or TIMx_CNT ≤ TIMx_CCRx (depending on the direction of the counter).

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

- Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at '1'. If the compare value is 0 then OCxRef is held at '0'. Figure below shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

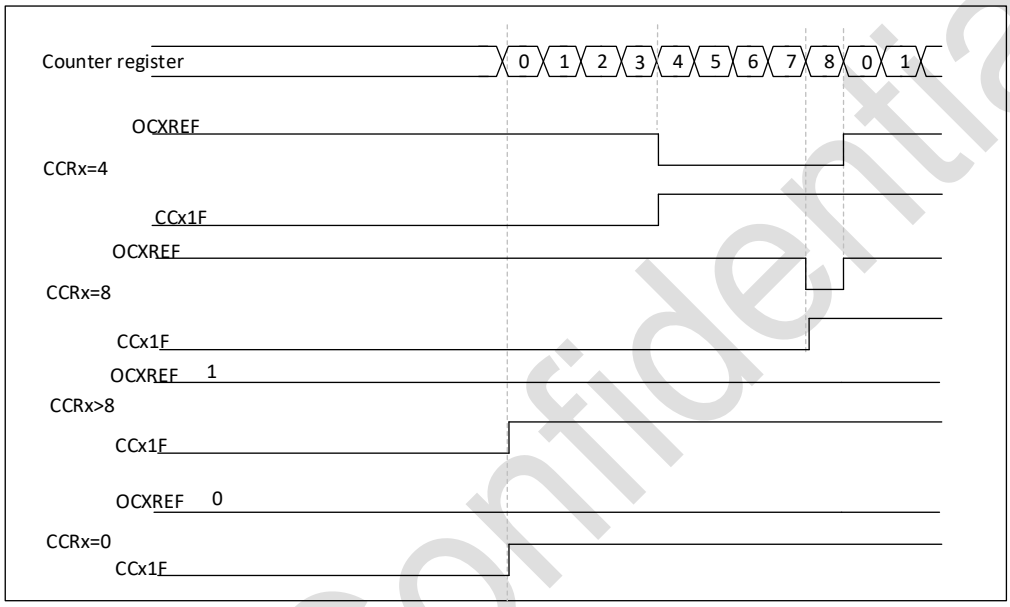

Figure 15-33 Edge-aligned PWM waveforms (ARR=8)

- 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.

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.

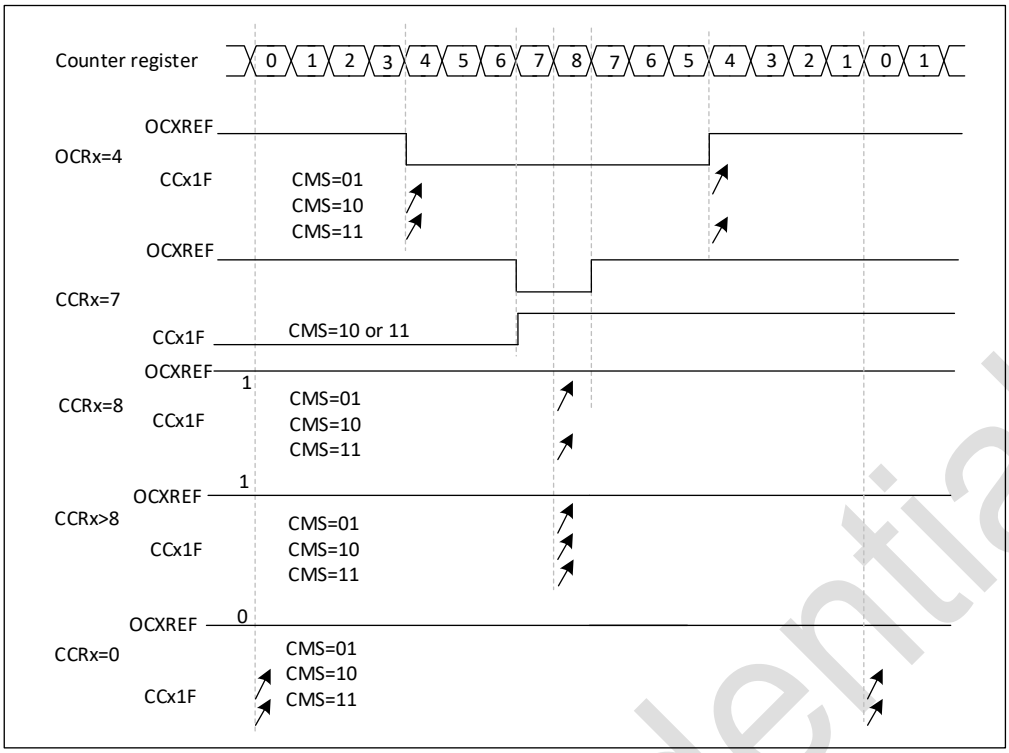

Figure 15-34 Center-aligned PWM waveforms (ARR=8)

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.

15.3.11. 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. Output control bits for complementary OCx and OCxN channels with

break feature. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. There is one 8-bit dead-time generator for each channel. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal except for the rising edge, which is delayed relative to the reference rising edge.
- The OCxN output signal is the opposite of the reference signal except for the rising edge, which is delayed relative to the reference falling edge.

If the delay is greater than the width of the active output (OCx or OCxN) then the corresponding pulse is not generated.

The following figures show the relationships between the output signals of the dead-time generator and the reference signal OCxREF. (we suppose CCxP=0, CCxNP=0, MOE=1, CCxE=1 and CCxNE=1 in these examples)

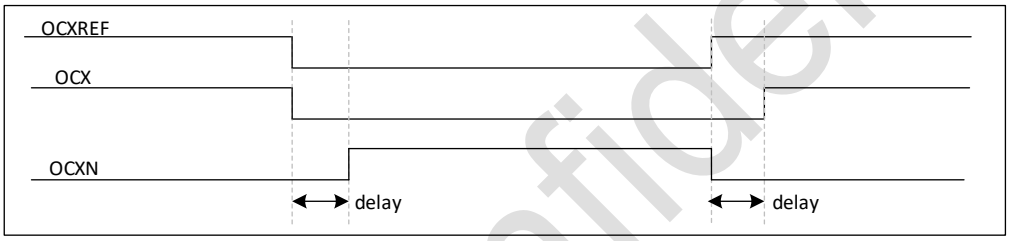

Figure 15-35 Complementary output with dead-time insertion

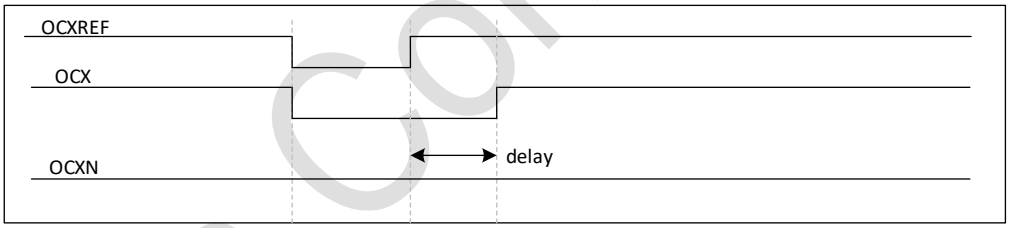

Figure 15-36 Dead-time waveforms with delay greater than the negative pulse

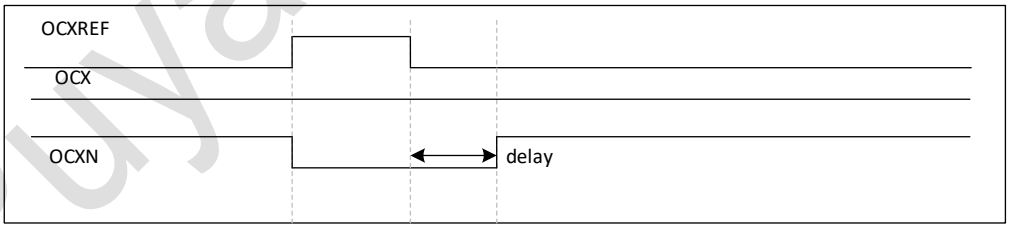

Figure 15-37 Dead-time waveforms with delay greater than the positive pulse.

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.

15.3.12. Using the break function

The purpose of the break function is to protect power switches driven by PWM signals from the timers. The two break inputs are usually connected to fault outputs of power stages and 3-phase inverters. When activated, the break circuitry shuts down the PWM outputs and forces them to a predefined safe state. A number of internal MCU events can also be selected to trigger an output shut-down.

When using the break function, the output enable signals and inactive levels are modified according to additional control bits. In any case, the OCx and OCxN outputs cannot be set both to active level at a given time.

The source for break (BRK) channel can be an external source connected to the BKIN pin or one of the following internal sources:

- the CPU LOCKUP output
- a clock failure event generated by the CSS detector
- 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 that because of the

resynchronization on MOE, the dead-time duration is a bit longer than usual.

— If OSSI=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.

- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until you write it to '1' again. In this case, it can be used for security and you can connect the break input to an alarm from power drivers, thermal sensors or any security components.

Note: The break inputs are active on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF can not be cleared.

The break can be generated by the BRK input which has a programmable polarity and an enable bit BKE in the TIMx_BDTR Register.

In addition to the break input and the output management, a write protection has been implemented inside the break circuit to safeguard the application. It allows to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The application can choose from 3 levels of protection selected by the LOCK bits in the TIMx_BDTR register. The LOCK bits can be written only once after an MCU reset.

The figure below shows an example of behavior of the outputs in response to a break.

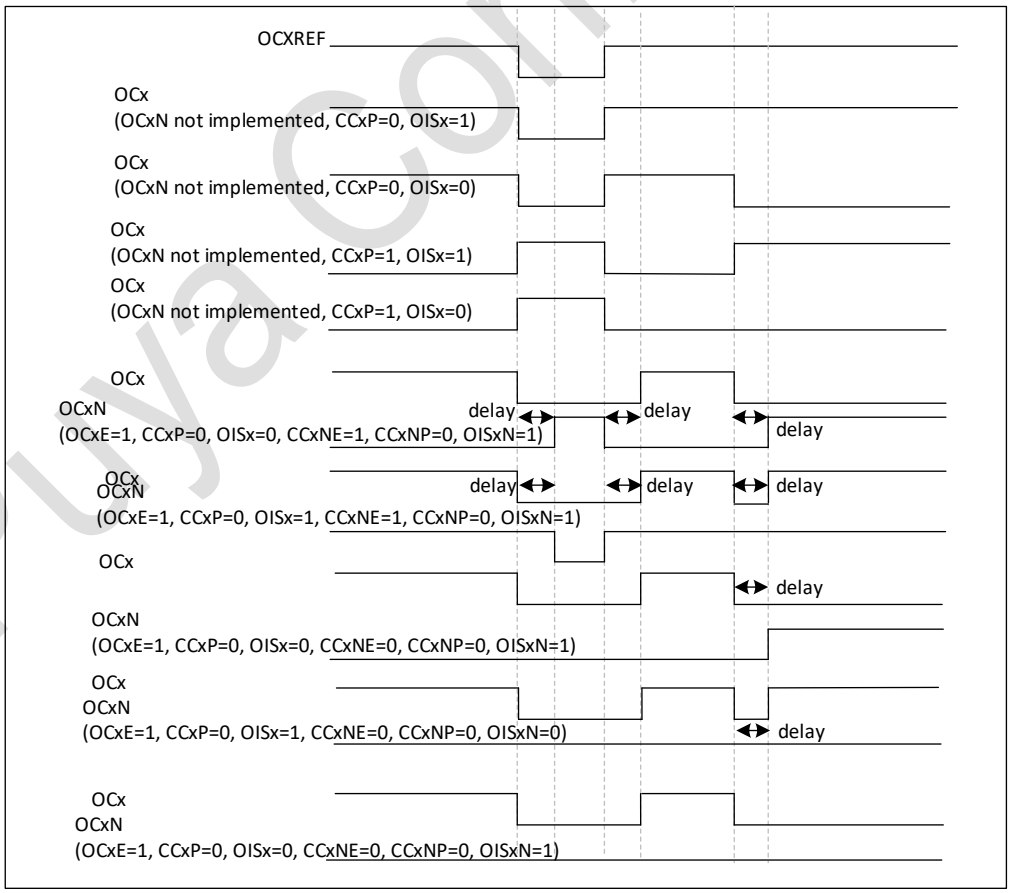

Figure 15-38 Output behavior in response to a break

15.3.13. 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 OCREF_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.

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:

The external trigger prescaler should be kept off: bits ETPS[1:0] in the TIMx_SMCR register are cleared to 00.

The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.

The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured according to the user needs.

The figure below shows the behavior of the OCxREF signal when the ETRF Input becomes High, for both values of the enable bit OCxCE. In this example, the timer TIMx is programmed in PWM mode.

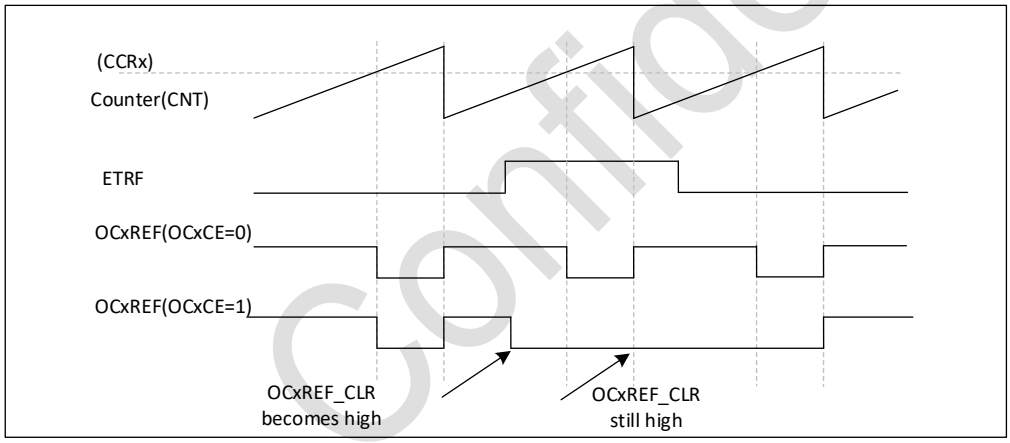

Figure 15-39 OCxREF for clearing TIM1

15.3.14. 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.

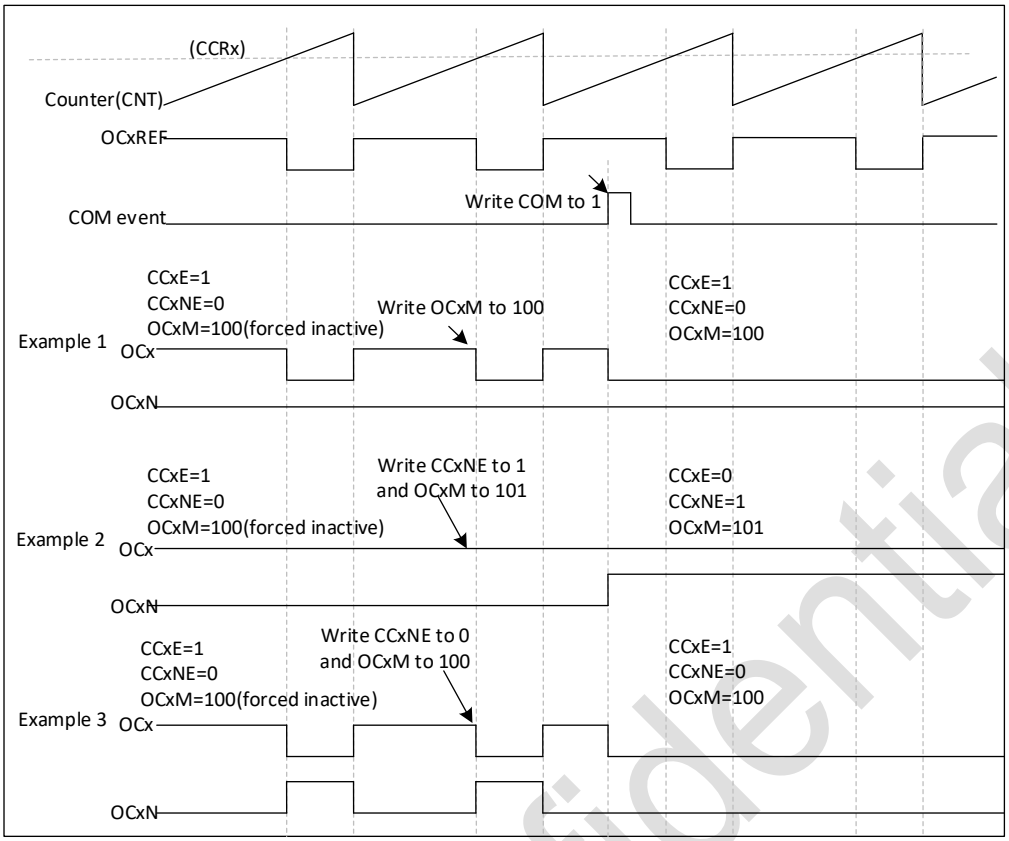

Figure 15-40 6-step generation, COM example (OSSR=1)

15.3.15. 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

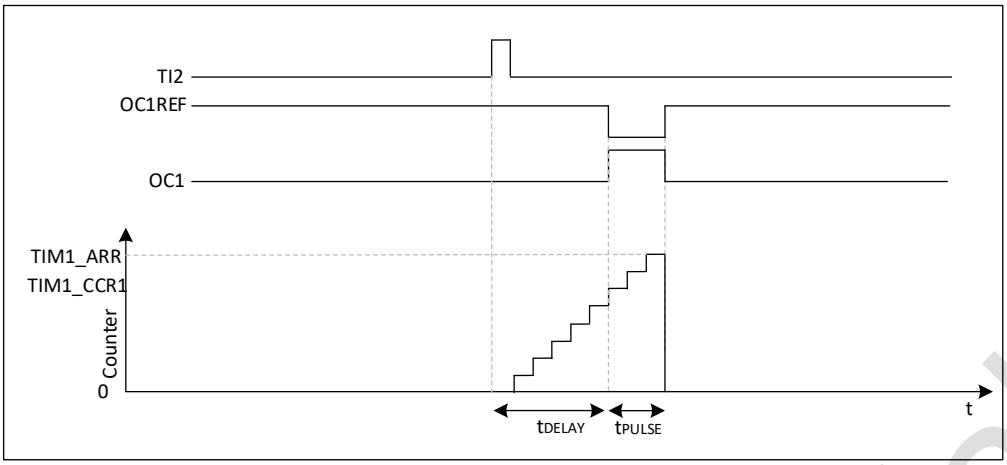

Figure 15-41 Example of one pulse mode

For example one may want to generate a positive pulse on OC1 with a length of t_{PULSE} and after a delay of t_{DELAY} as soon as a positive edge is detected on the TI2 input pin.

Let's use TI2FP2 as trigger 1:

- Map TI2FP2 on TI2 by writing CC2S=01 in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=0 in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=110 in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMx_SMCR register (trigger mode).

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler).

- The $t_{DELAY\ is}$ defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let's say one want to build a waveform with a transition from '0' to '1' when a compare match occurs and a transition from '1' to '0' when the counter reaches the auto-reload value. To do this PWM mode 2 must be enabled by writing OC1M=111 in the TIMx_CCMR1 register. Optionally the preload registers can be enabled by writing OC1PE='1' in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case one has to write the compare value in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '0' in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.

Since only 1 pulse (Single mode) is needed, a 1 must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0)

Particular case: OCx fast enable:

In One-pulse mode, the edge detection on TIx input set the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But several clock cycles are needed for these operations, and it limits the minimum delay t_{DELAY} min we can get.

If one wants to output a waveform with the minimum delay, the OCxFE bit can be set in the TIMx_CCMRx register. Then OCxREF 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.

15.3.16. Encoder interface mode

To select Encoder Interface mode: write SMS='001' in the TIMx_SMCR register if the counter is counting on TI2 edges only, SMS=010 if it is counting on TI1 edges only and SMS=011 if it is counting on both TI1 and TI2 edges.

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, you can program the input filter as well.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. 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 15-1 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 CC1NP = ‘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)

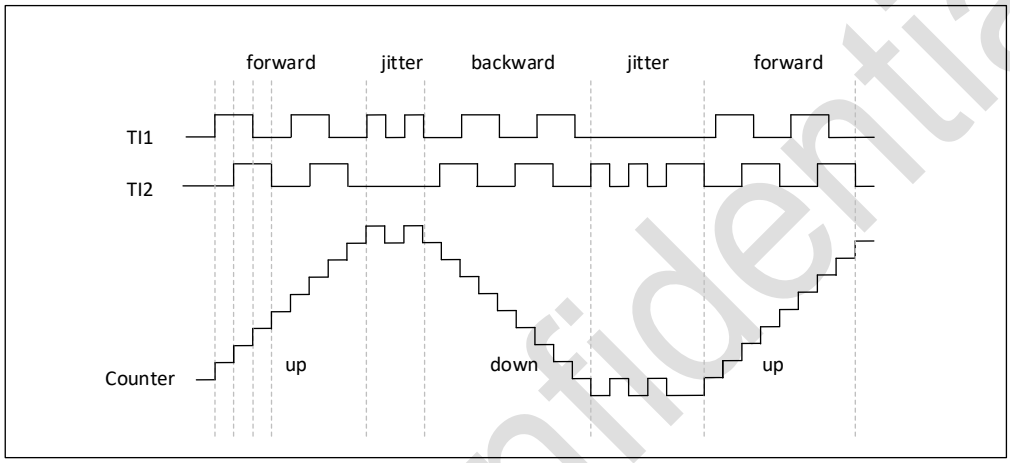

Figure15-42 Example of counter operation in encoder interface mode

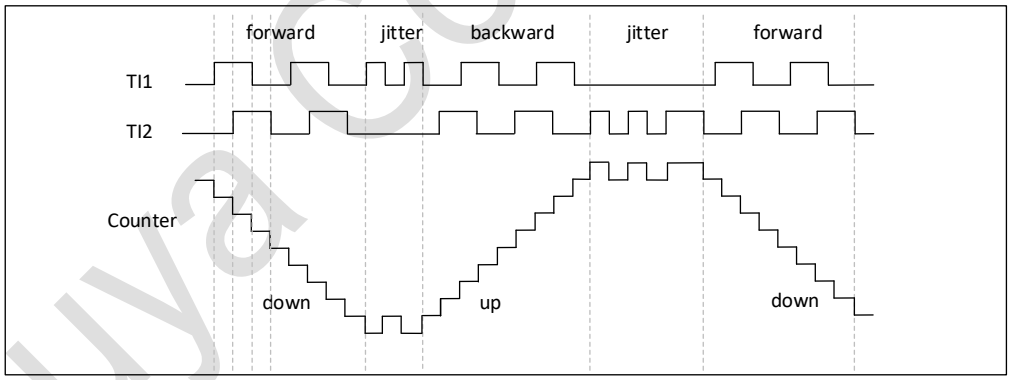

Figure15-43 Example of encoder interface mode with IC1FP1 polarity inverted

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).

15.3.17. Timer input XOR function

The TI1S bit in the TIMx_CR2 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMx_CH1, TIMx_CH2 and TIMx_CH3.

The XOR output can be used with all the timer input functions such as trigger or input capture.

15.3.18. Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIMx referred to as “interfacing timer”. The “interfacing timer” captures the 3 timer input pins (CH1, CH2, CH3) connected through an XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

The slave mode controller is configured in reset mode. the slave input is TI1F_ED. Thus, each time one of the 3 inputs toggles, the counter restarts counting from 0. This creates a time base triggered by any change on the Hall inputs.

On the “interfacing timer”, capture/compare channel 1 is configured in capture mode, capture signal is TRC. The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

The “interfacing timer” can be used in output mode to generate a pulse which changes the configuration of the channels of the advanced-control timer (TIM1) (by triggering a COM event). The TIM1 timer is used to generate PWM signals to drive the motor. To do this, the interfacing timer channel must be programmed so that a positive pulse is generated after a programmed delay (in output compare or PWM mode). This pulse is sent to the advanced-control timer (TIM1) through the TRGO output.

Example: one wants to change the PWM configuration of the advanced-control timer after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMx timers.

- Configure 3 timer inputs ORed to the TI1 input channel by writing the TI1S bit in the TIMx_CR2 register to ‘1’.
- Program the time base: write the TIMx_ARR to the max value (the counter must be cleared by the TI1 change. Set the prescaler to get a maximum counter period longer than the time between 2 changes on the sensors.
- Program the channel 1 in capture mode (TRC selected): write the CC1S bits in theTIMx_CCMR1 register to ‘01’. The digital filter can also be programmed if needed,
- Program channel 2 in PWM 2 mode with the desired delay: write the OC2M bits to ‘111’ and the CC2S bits to ‘00’ in the TIMx_CCMR1 register.
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIMx_CR2 register to ‘101’.

In the advanced-control timer TIM1, the right ITR input must be selected as trigger input, the timer is programmed to generate PWM signals, the capture/compare control signals are preloaded (CCPC=1 in the TIMx_CR2 register) and the COM event is controlled by the trigger input (CCUS=1 in the TIMx_CR2 register). The PWM control bits (CCxE, OCxM) are written after a COM event for the next step, which can be done in an interrupt subroutine generated by the rising edge of OC2REF).

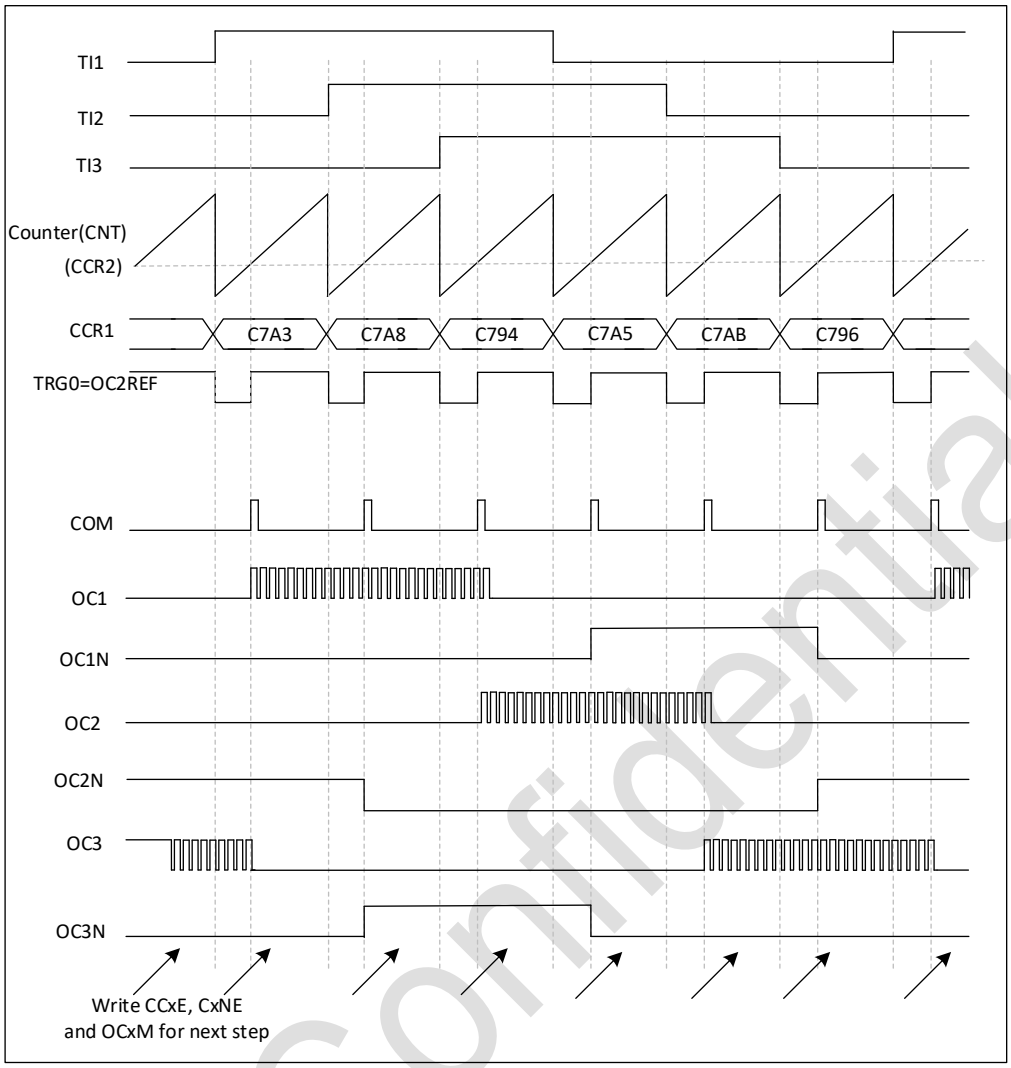

Figure 15-44 Example of Hall sensor interface

15.3.19. TIM and external trigger synchronization

The TIMx timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the URS bit from the TIMx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMx_ARR, TIMx_CCRx) are updated.

In the following example, the upcounter is cleared in response to a rising edge on TI1 input:

- Configure the channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we do not need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so it does not need to be configured. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.

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

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. 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.

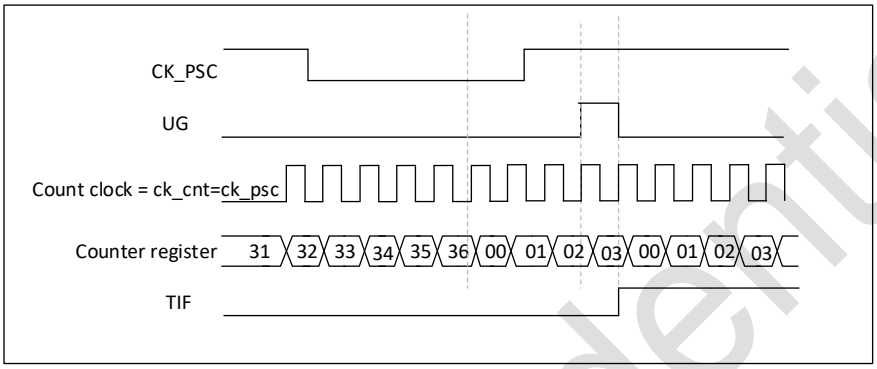

Figure 15-45 Control circuit in reset mode

Slave mode: Gated mode

The counter can be enabled depending on the level of a selected input.

In the following example, the upcounter counts only when TI1 input is low:

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we 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.

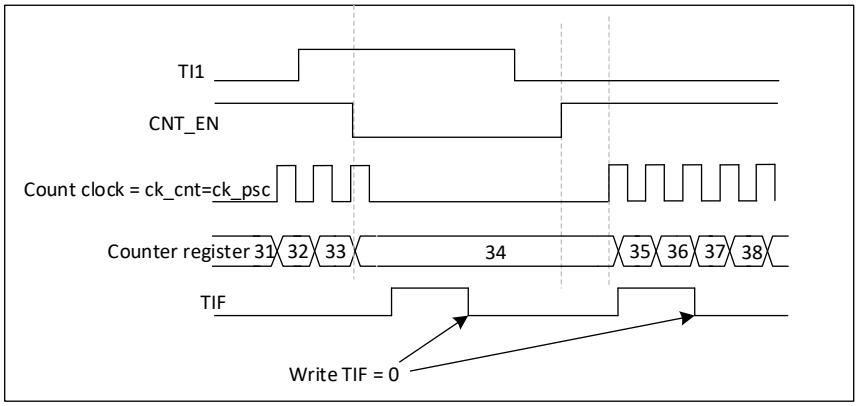

Figure 15-46 Control circuit in Gated mode

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.

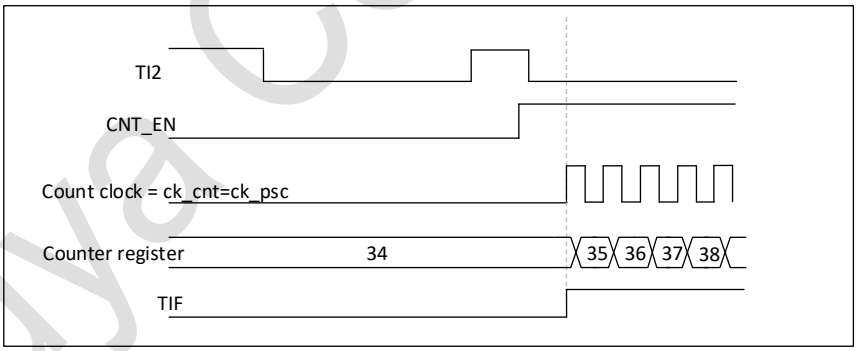

Figure 15-47 Control circuit in Gated mode 2

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:

- Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
  - ETF = 0000: no filter

- ETPS = 00: prescaler disabled
- ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.

- Configure the channel 1 as follows, to detect rising edges on TI:
  - IC1F = 0000: no filter
  - The capture prescaler is not used for triggering, 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).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges. The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

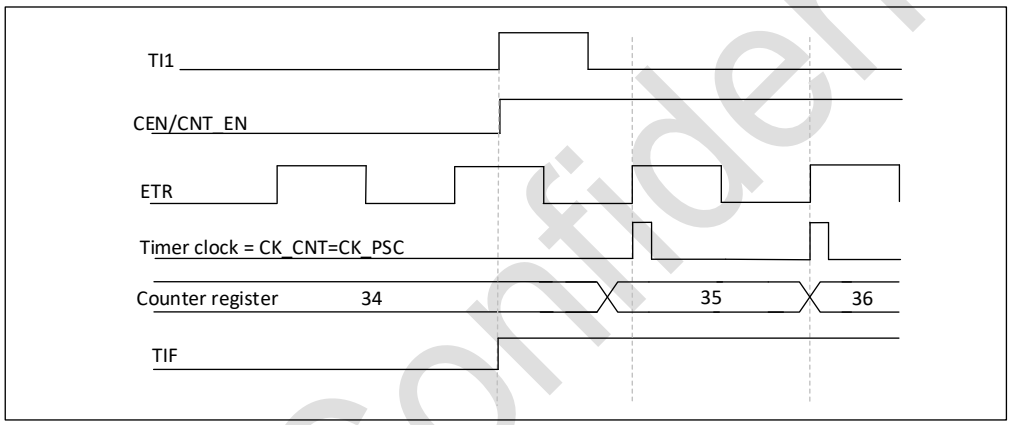

Figure 15-48 Control circuit in external clock mode 2 + trigger mode

15.3.20. 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.

15.4. TIM1 registers

15.4.1. 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	-	-	-
9:8	CKD[1:0]	RW	2'b0	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: $t_{DTS} = t_{CK\_INT}$

Bit	Name	R/W	Reset Value	Function
				01: t_{DTS} = 2 x t_{CK_INT} 10: t_{DTS} = 4 x 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	2'b0	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 compare interrupt flags of channels configured in output (CCxS=00 in TIM1_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. The counter counts up and down 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: – 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.

15.4.2. 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, must be kept at reset value.
14	OIS4	RW	0	Output idle state 4 (OC4 output) Refer to OIS1 bit.
13	OIS3N	RW	0	Output idle state 3 (OC3N output) Refer to OIS1N bit
12	OIS3	RW	0	Output idle state 3 (OC3 output) Refer to OIS1 bit.
11	OIS2N	RW	0	Output idle state 2 (OC2N output) Refer to OIS1N bit
10	OIS2	RW	0	Output idle state 2 (OC2 output) Refer to OIS1 bit
9	OIS1N	RW	0	Output idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 Note: this bit 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	3'b0	Master mode selection These 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. 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	-	-	-

Bit	Name	R/W	Reset Value	Function
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	-	-	-
0	CCPC	RW	0	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded. 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs. Note: This bit acts only on channels that have a complementary output.

15.4.3. TIM1 slave mode control register (TIM1_SMCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15	ETP	RW	0	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
14	ECE	RW	0	External clock enable. This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.
13:12	ETPS[1:0]	RW	2'b0	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIM1 CLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
11:8	ETF[3:0]	RW	4'b0	External trigger filter. This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output. 0000: No filter, sampling is done at 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 0100:$f_{SAMPLING}=f_{DTS}/2$, N=6 0101:$f_{SAMPLING}=f_{DTS}/2$, N=8 0110:$f_{SAMPLING}=f_{DTS}/4$, N=6 0111:$f_{SAMPLING}=f_{DTS}/4$, N=8 1000:$f_{SAMPLING}=f_{DTS}/8$, N=6 1001:$f_{SAMPLING}=f_{DTS}/8$, N=8 1010:$f_{SAMPLING}=f_{DTS}/16$, N=5 1011:$f_{SAMPLING}=f_{DTS}/16$, N=6 1100:$f_{SAMPLING}=f_{DTS}/16$, N=8 1101:$f_{SAMPLING}=f_{DTS}/32$, N=5 1110:$f_{SAMPLING}=f_{DTS}/32$, N=6 1111:$f_{SAMPLING}=f_{DTS}/32$, N=8

Bit	Name	R/W	Reset Value	Function
				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	3'b0	Trigger selection. This bit-field selects the trigger input to be used to synchronize the counter. 000: TIM14 (ITR0) 001: Reserved (ITR1) 010: Reserved (ITR2) 011: Reserved (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF) Note: These bits must be changed only when they are not used to avoid wrong edge detections at the transition.
3	OCCS	RW	0	OCREF clear selection This bit is used to select the OCREF clear source. 0: OCREF clear source is connected to OCREF_CLR input 1: OCREF clear source is connected to ETRF
2:0	SMS[2:0]	RW	3'b0	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 000: Slave mode disabled If CEN = '1' then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 010: Encoder mode 2 Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 011: Encoder mode 3 Synthesis of Modes 1 and 2 100: Reset mode Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated mode The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

TIM1 internal trigger connection

Slave TIM	ITR0 (TS=000)	ITR1 (TS=001)	ITR2 (TS=010)	ITR3 (TS=011)
TIM1	TIM14	Reserved	Reserved	Reserved

15.4.4. 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	-	-	-
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 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

15.4.5. 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								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			CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			RC_W0	RC_W0	RC_W0	RC_W0	-	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:24	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.
12	CC4OF	RC_W0	0	Capture/Compare 4 overcapture flag Refer to CC1OF description

Bit	Name	R/W	Reset Value	Function
11	CC3OF	RC_W0	0	Capture/Compare 3 overcapture flag Refer to CC1OF description
10	CC2OF	RC_W0	0	Capture/Compare 2 overcapture flag Refer to CC1OF description
9	CC1OF	RC_W0	0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in TIM1_CCR1 register while CC1OF flag was already set.
8	Reserved	-	-	Reserved, must be kept at reset value.
7	BIF	RC_W0	0	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input.
6	TIF	RC_W0	0	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode). It is set when the counter starts or stops when gated mode is selected. It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	RC_W0	0	COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 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 flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIM1_CR1 register description). 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))

15.4.6. 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	-	-	-
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).

15.4.7. TIM1 capture/compare mode register 1 (TIM1_CCMR1)

Address offset: 0x18

Reset value: 0x0000 0000

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

Input compare mode:

Bit	Name	R/W	Reset value	Function
31:16	Reserved	-	-	-
15	OC2CE	RW	0	Output Compare 2 clear enable
14:12	OC2M [2:0]	RW	3'b0	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	2'b0	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	3'b0	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 compare register TIM1_CCR1 and the counter TIM1_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle OC1REF toggles when TIMx_CNT = TIMx_CCR1. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as 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'). 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 OC output. 0: OC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate OC1 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 OC1 output is reduced to 3 clock cycles. OC1FE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	RW	2'b0	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	

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:12	IC2F	RW	4'b0	Input capture 2 filter
11:10	IC2PSC [1:0]	RW	2'b0	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	4'b0	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at 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 0100:$f_{SAMPLING}=f_{DTS}/2$, N=6 0101:$f_{SAMPLING}=f_{DTS}/2$, N=8 0110:$f_{SAMPLING}=f_{DTS}/4$, N=6 0111:$f_{SAMPLING}=f_{DTS}/4$, N=8 1000:$f_{SAMPLING}=f_{DTS}/8$, N=6 1001:$f_{SAMPLING}=f_{DTS}/8$, N=8 1010:$f_{SAMPLING}=f_{DTS}/16$, N=5

Bit	Name	R/W	Reset Value	Function
				1011:$f_{SAMPLING}=f_{DTS}/16$, N=6 1100:$f_{SAMPLING}=f_{DTS}/16$, N=8 1101:$f_{SAMPLING}=f_{DTS}/32$, N=5 1110:$f_{SAMPLING}=f_{DTS}/32$, N=6 1111:$f_{SAMPLING}=f_{DTS}/32$, N=8
3:2	IC1PSC [1:0]	RW	2'b0	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	2'b0	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).

15.4.8. TIM1 capture/compare mode register 2 (TIM1_CCMR2)

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**
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

Input compare mode:

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15	OC4CE	RW	0	Output Compare 4 clear enable
14:12	OC4M [2:0]	RW	3'b0	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	2'b0	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	2'b0	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	2'b0	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.

Bit	Name	R/W	Reset Value	Function
				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	-	-	-
15:12	IC4F	RW	4'b0	Input capture 4 filter
11:10	IC4PSC	RW	2'b0	Input/capture 4 prescaler
9:8	CC4S	RW	2′ b0	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	4'b0	Input capture 3 filter
3:2	IC3PSC	RW	2'b0	Input/capture 3 prescaler
1:0	OC3S	RW	2'b0	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).

15.4.9. TIM1 capture/compare enable register (TIM1_CCER)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res		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	-	-	-
13	CC4P	RW	0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	RW	0	Input/Capture 4 output enable Refer to CC1E description.
11	CC3NP	RW	0	Input/Capture 3 complementary output polarity. Refer to CC1NP description.
10	CC3NE	RW	0	Input/Capture 3 complementary output enable. Refer to CC1NE description.
9	CC3P	RW	0	Input/Capture 3 output polarity refer to CC1P description.
8	CC3E	RW	0	Input/Capture 3 output enable refer to CC1E description.
7	CC2NP	RW	0	Input/Capture 2 complementary output polarity. Refer to CC1NP description.
6	CC2NE	RW	0	Input/Capture 2 complementary output enable. Refer to CC1NE description.
5	CC2P	RW	0	Input/Capture 2 output polarity refer to CC1P description.
4	CC2E	RW	0	Input/Capture 2 output enable refer to CC1E description.
3	CC1NP	RW	0	Input/Capture 1 complementary output polarity. 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

Bit	Name	R/W	Reset value	Function
				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. 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 not inverted (trigger operation in gated mode). This configuration must not be used in encoder mode. Notes: 1. For complementary output channels, this bit is preloaded. If the CCPC bit is set in the TIMx_CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register). 3. CH4 is effective without double edges. 4. Input capture If you need to pay attention to ICxIR or ICxIF, CCxS only supports 01 cases.
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. 1: Capture enabled.

Table 15-2 Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output state	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
1	X	0	0	0	Output disabled (not driven by the timer), OCx=0, OCx_EN=0	Output disabled (not driven by the timer), OCxN=0, OCxN_EN=0
		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		

15.4.10. 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	-	-	-
15:0	CNT[15:0]	RW	16′ h0	Counter value

15.4.11. 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	-	-	-
15:0	PSC[15:0]	RW	16′ h0	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC[15:0] + 1). The PSC contains the value loaded into the current prescaler register when an update event occurs. Update event includes counter Clear 0 by the UG bit of TIM_EGR or by the slave controller operating in reset mode.

15.4.12. 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	-	-	-
15:0	ARR[15:0]	RW	16'hFFFF	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.

15.4.13. 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	-	-	-
7:0	REP[7:0]	RW	8'h0	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.

15.4.14. 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	-	-	-
15:0	CCR1 [15:0]	RW	16'h0	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).

15.4.15. 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	-	-	-
15:0	CCR2 [15:0]	RW	16'h0	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).

15.4.16. 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	-	-	-
15:0	CCR3 [15:0]	RW	16'h0	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).

15.4.17. 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	-	-	-
15:0	CCR4 [15:0]	RW	16'h0	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).

15.4.18. 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	-	-	-
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: Break function disabled 1: Break function enabled 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 (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.

Bit	Name	R/W	Reset Value	Function
				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 (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	2'b0	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF, no bit is write protected. 01: LOCK Level 1, DTG, BKE, BKP, AOE bits in TIM1_BDTR register and OISx and OISxN bits in TIM1_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIM1_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIM1_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIM1_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	RW	8'h0	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG[7:5]=0xx => DT=DTG[7:0] × T_{dtg}, T_{dtg} = T_{DTS} DTG[7:5]=10x => DT=(64+DTG[5:0]) ×T_{dtg}, T_{dtg} = 2 ×T_{DTS} DTG[7:5]=110 => DT=(32+DTG[4:0]) ×T_{dtg}, T_{dtg} = 8 ×T_{DTS} DTG[7:5]=111 => DT=(32+DTG[4:0]) ×T_{dtg}, T_{dtg} = 16 ×T_{DTS} Example if TDTS=125ns (8MHz), dead-time possible values are: 0 to 15875 ns by 125 ns steps. 16 us to 31750 ns by 250 ns steps. 32 us to 63us by 1 us steps. 64 us to 126 us by 2 us steps Note: 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).

16. General-purpose timer (TIM14)

16.1. 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).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.

16.2. 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

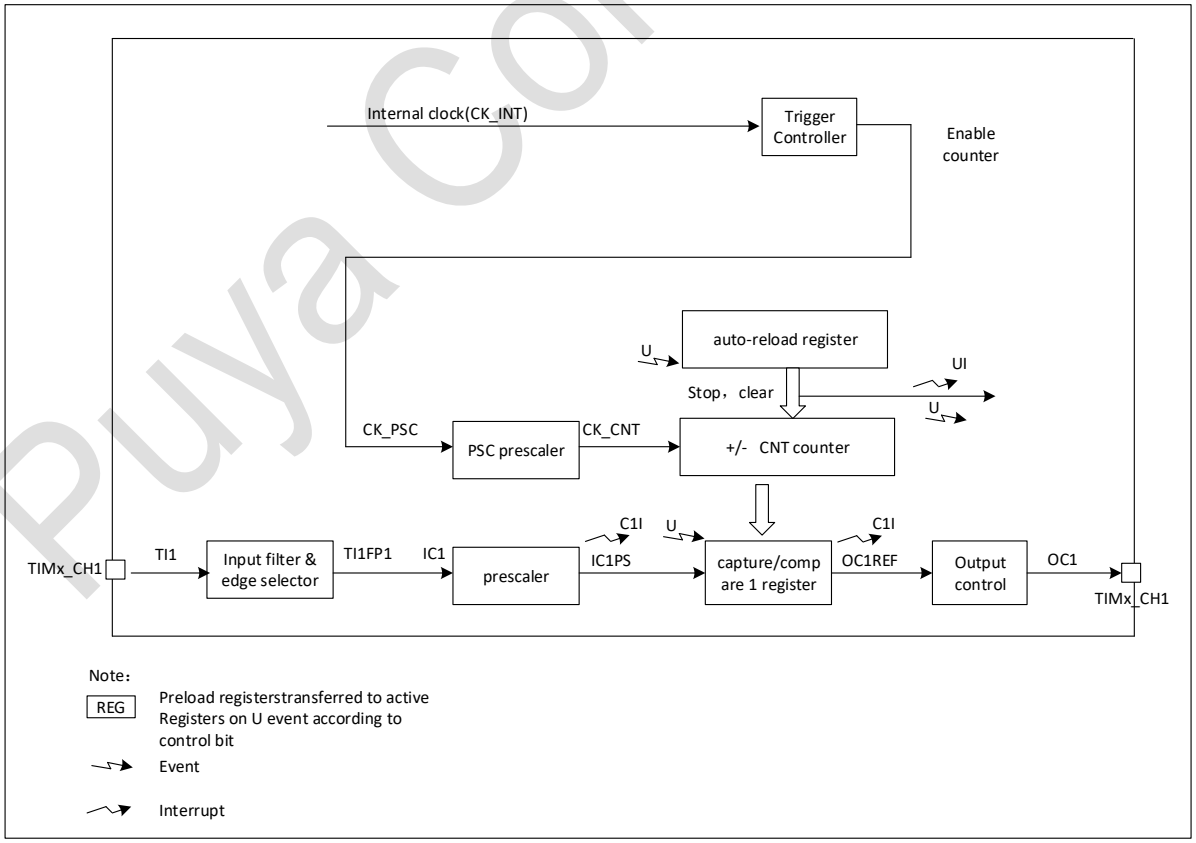

Figure 16-1 General-purpose timer block diagram (TIM14)

16.3. TIM14 functional description

16.3.1. 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 pre-load 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.

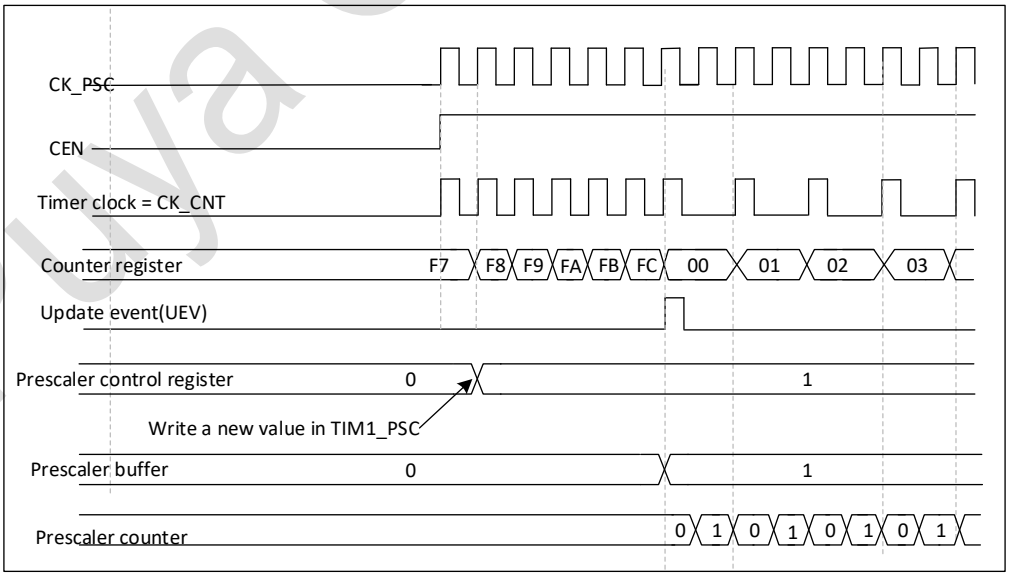

Figure 16-2 Counter timing diagram with prescaler division change from 1 to 2

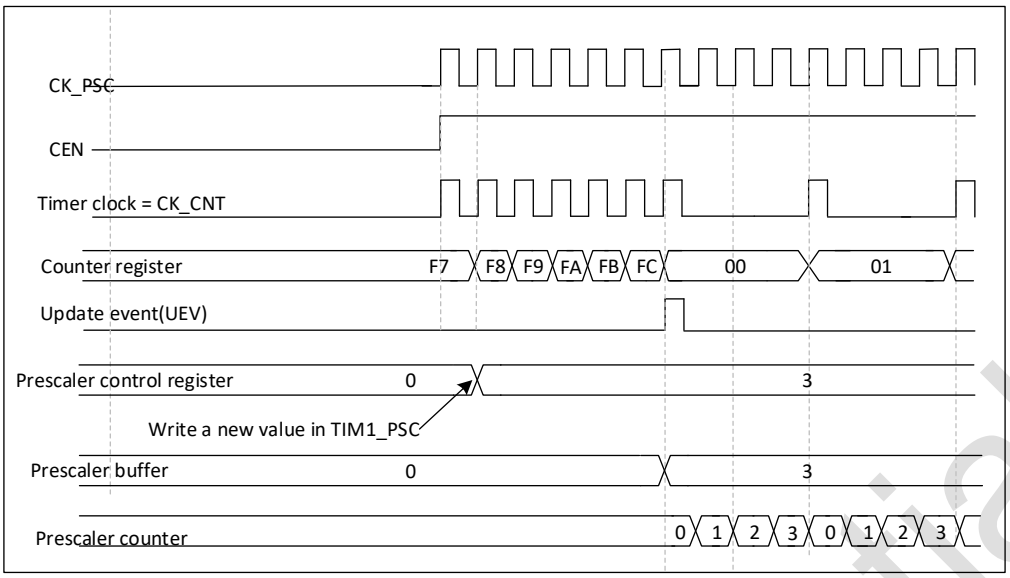

Figure 16-3 Counter timing diagram with prescaler division change from 1 to 4

Upcounting mode

The counter counts from 0 to the auto-reload value (contents of the TIM14_ARR register), then restarts from 0 and generates a counter overflow event.

Else the update event is generated at each counter overflow. Setting the UG bit in the TIM14_EGR register (by software) also generates an update event.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter 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.

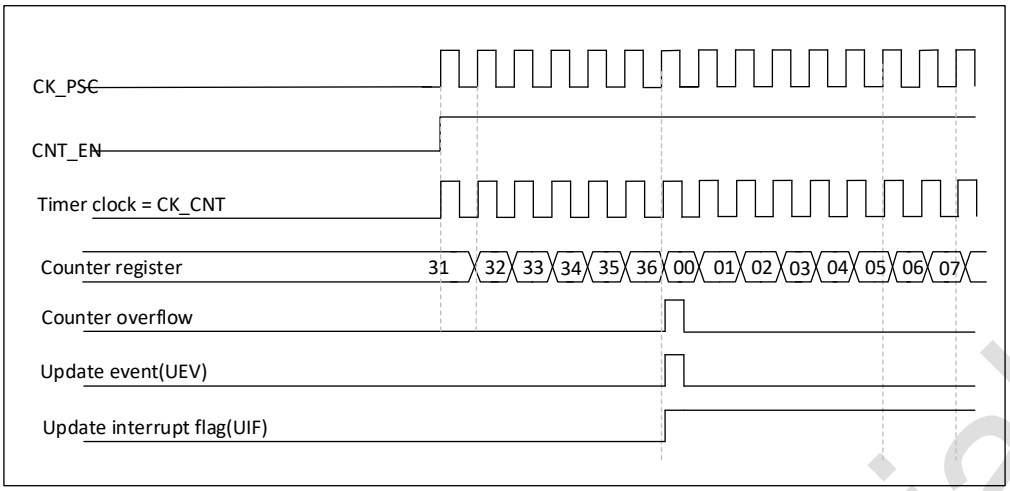

Figure 16-4 Counter timing diagram, internal clock divided by 1

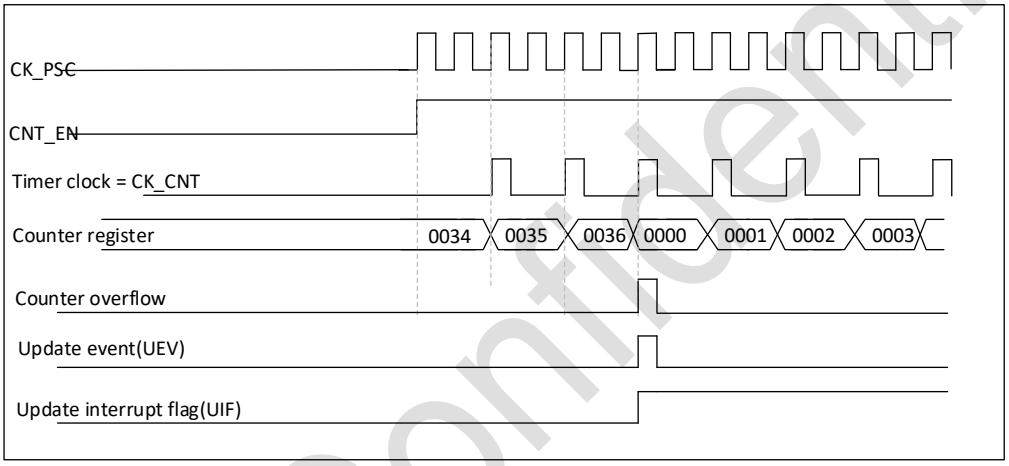

Figure 16-5 Counter timing diagram, internal clock divided by 2

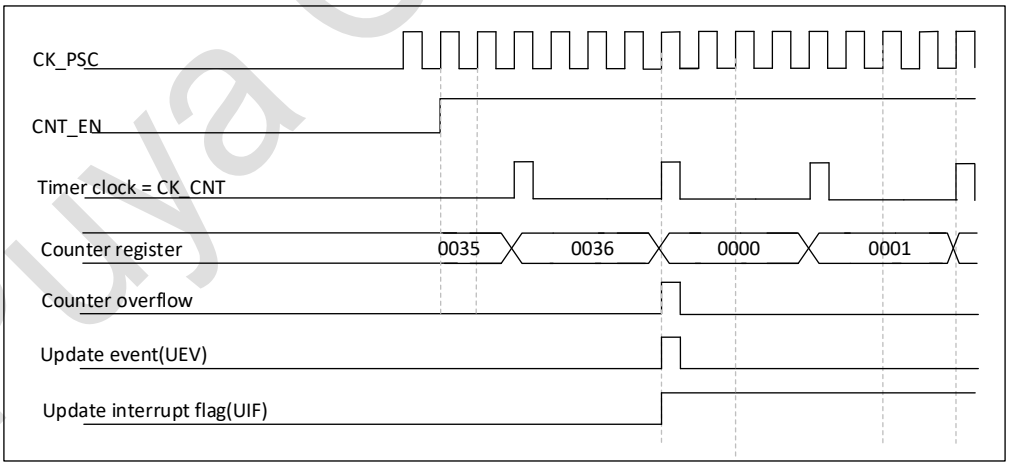

Figure 16-6 Counter timing diagram, internal clock divided by 4

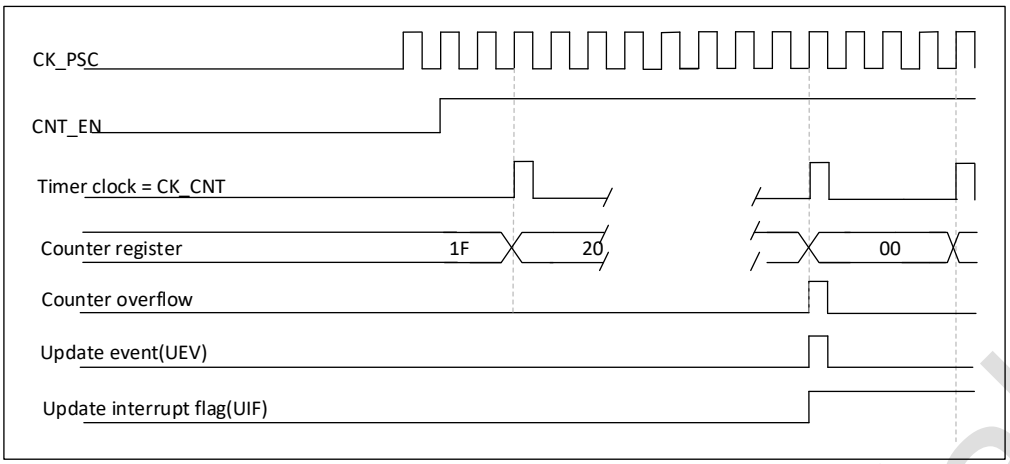

Figure 16-7 Counter timing diagram, internal clock divided by N

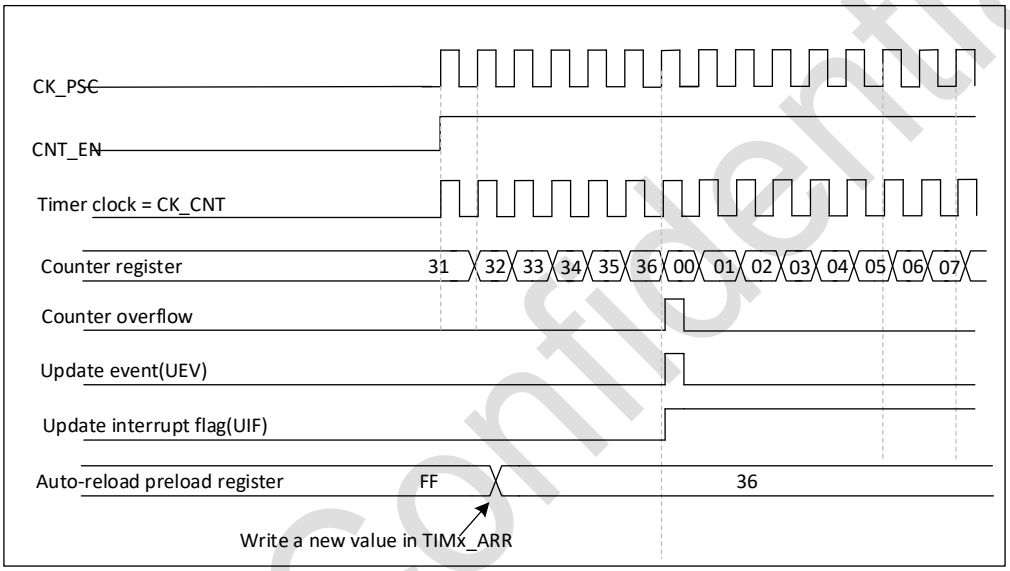

Figure 16-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

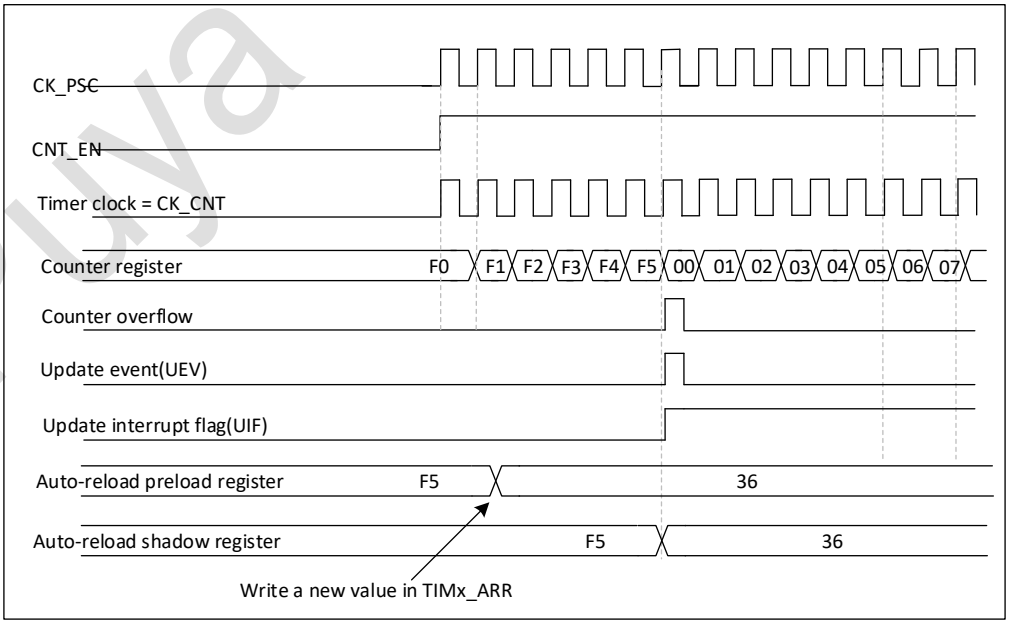

Figure 16-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

16.3.2. 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.

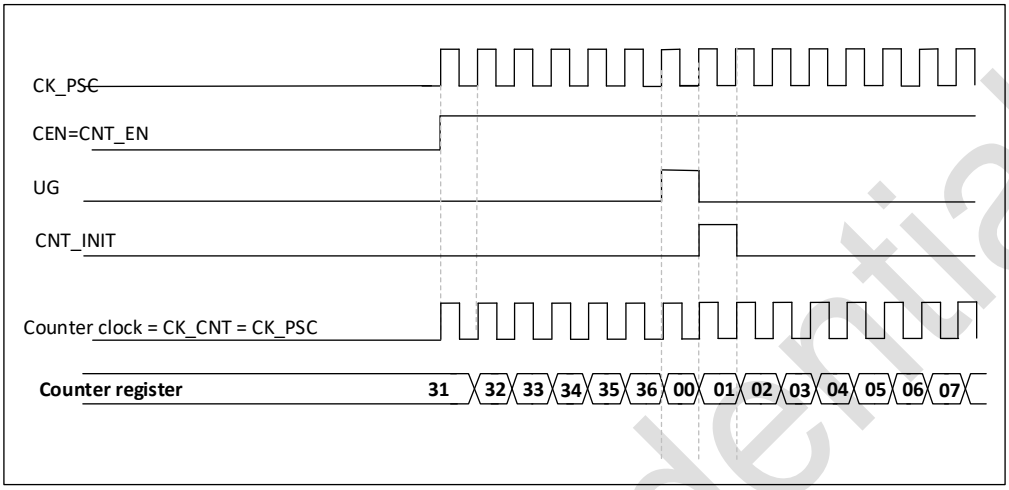

Figure 16-10 Control circuit in normal mode, internal clock divided by 1

16.3.3. 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).

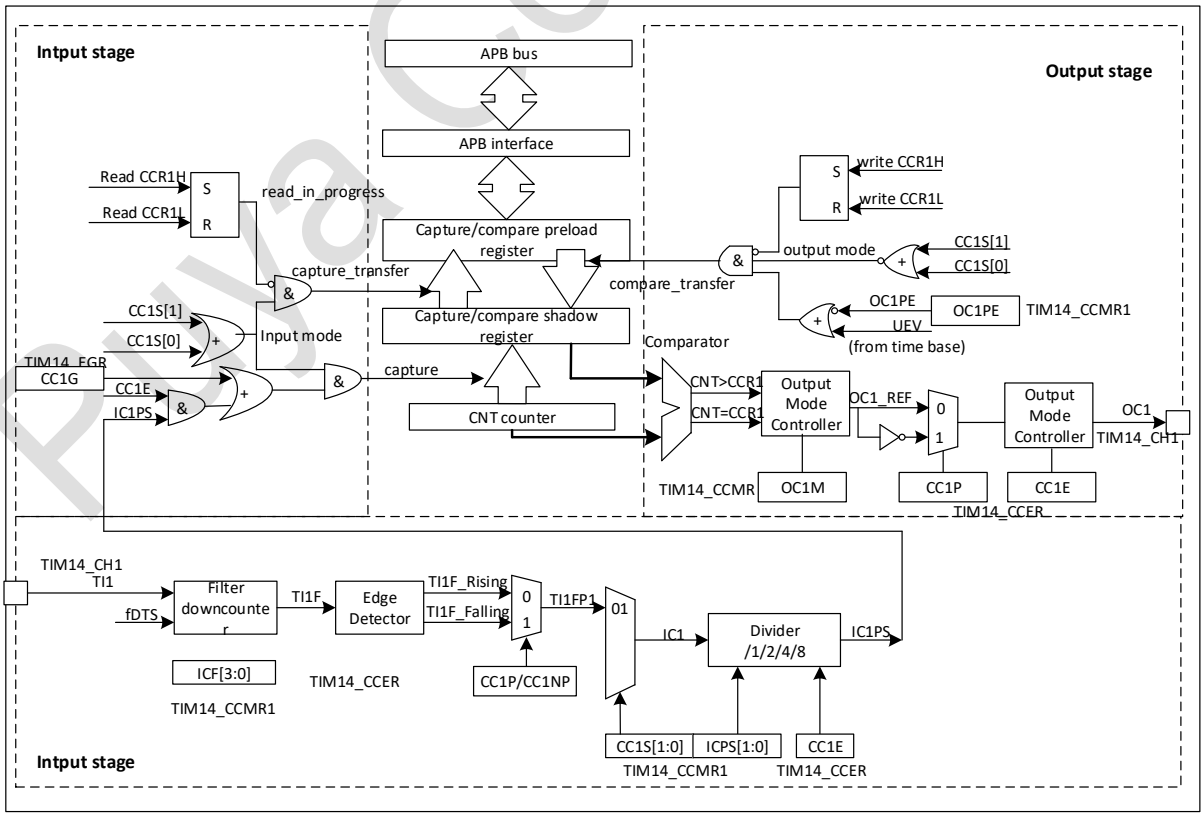

Figure 16-11 TIM14 capture/compare channel

The input stage samples the corresponding TIx input to generate a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx) which can be used as trigger input by the slave mode controller or as the capture command. It is prescaled before the capture register (ICxPS).

The output stage generates an intermediate waveform (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.

16.3.4. Input capture mode:

In Input capture mode, the capture/compare registers (TIM14_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIM14_SR register) is set. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when 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 CC1IE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIM14_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.

- An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before the overcapture flag. This is to avoid missing an overcapture which could happen after reading the flag and before reading the data.

Note: IC interrupt requests can be generated by software by setting the corresponding CCxG bit inthe TIMx_EGR register.

16.3.5. 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. This is described in the output compare mode section below.

16.3.6. 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.

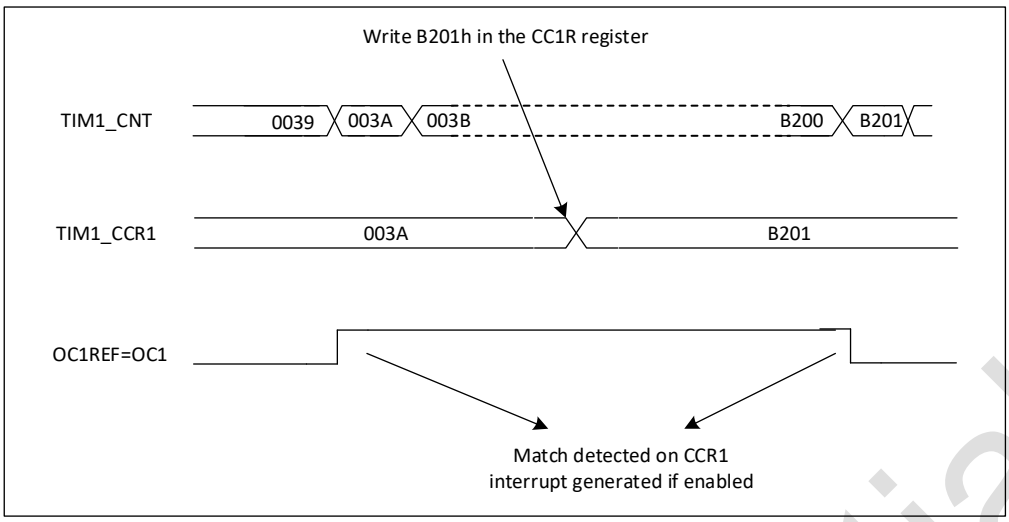

Figure 16-12 Output compare mode, toggle on OC1

16.3.7. 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.

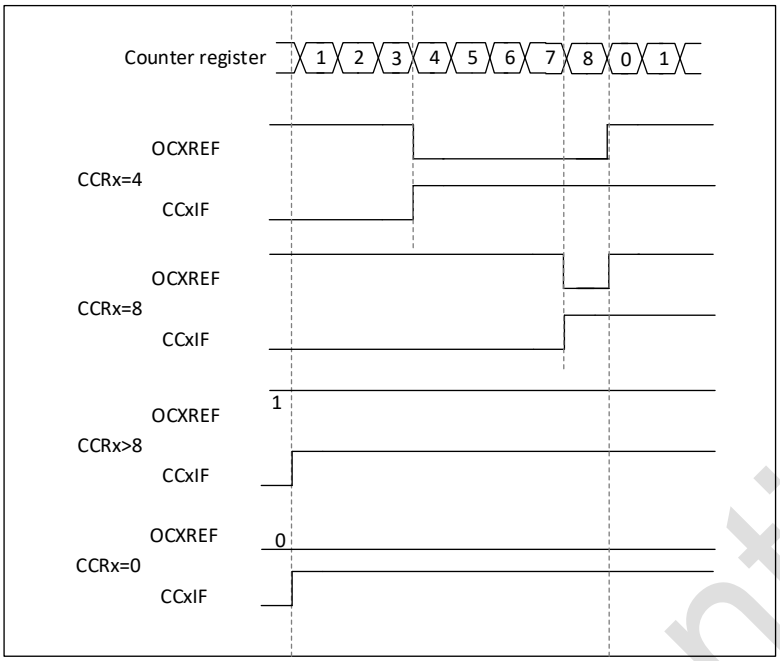

Figure 16-13 Edge-aligned PWM waveforms (ARR=8)

16.3.8. Timer synchronization

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.

16.3.9. Debug mode

When the microcontroller enters debug mode (Cortex®-M0+ core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

16.4. TIM14 registers

16.4.1. 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	-	-	-
9:8	CKD[1:0]	RW	2'b0	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	-	-	-
2	URS	RW	0	Update request source

Bit	Name	R/W	Reset Value	Function
				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.

16.4.2. 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	-	-	-
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

16.4.3. 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							CC1IF	UIF
-						RC_W0	-							RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	-
20	IC1IF	RC_W0	0	Falling edge capture 1 flag

Bit	Name	R/W	Reset Value	Function
				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	Reserved	-	-	-
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.
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	-	-	-
1	CC1IF	RC_W0	0	Capture/Compare 1 interrupt flag **If channel CC1 is configured as output:** This flag is set by hardware when the counter matches the compare value. It is cleared by 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. – When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register.

16.4.4. 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	-	-	-
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:**

Bit	Name	R/W	Reset Value	Function
				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).

16.4.5. 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	-	-	-
6:4	OC1M [2:0]	RW	3'b0	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen. The comparison between the output compare register TIM1_CCR1 and the counter TIMx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle OC1REF toggles when TIMx_CNT = TIMx_CCR1. 100: Force inactive level OC1REF is forced low. 101: Force active level OC1REF is forced high. 110 PWM mode 1- In upcounting, channel 1 is active as long as 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.
1:0	CC1S [1:0]	RW	2'b0	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.

Bit	Name	R/W	Reset value	Function
				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	-	-	-
7:4	IC1F [3:0]	RW	4’b0000	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at 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 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 0110: $f_{SAMPLING}=f_{DTS}/4$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
3:2	IC1PSC [1:0]	RW	2’b00	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	2’b00	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).

16.4.6. 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	-	-	-

Bit	Name	R/W	Reset Value	Function
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 de-fineTI1FP1 polarity (refer to CC1P description).
2	Reserved	-	0	Reserved, must be kept at reset value.
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 trigger or 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-The OC1 signal is output to the corresponding output pin. The CC1 channel is configured as an input: This bit determines whether the value of the counter can be captured into the TIMx_CCR1 register. 0: Capture disabled. 1: Capture enabled.

CcxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx= OCxREF+Polarity, OCx_EN=1

16.4.7. 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	-	-	-
15:0	CNT[15:0]	RW	16'h0	Counter value

16.4.8. 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	-	-	-
15:0	PSC[15:0]	RW	16'h0	Prescaler value

				The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). The PSC contains the value loaded into the current prescaler register when an update event occurs. Update event includes counter Clear 0 by the UG bit of TIM_EGR or by the slave controller operating in reset mode.

16.4.9. 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	-	-	-
15:0	ARR[15:0]	RW	16'hFFFF	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.

16.4.10. 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	-	-	-
15:0	CCR1 [15:0]	RW	16'h0	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).

16.4.11. 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	

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	-
1:0	TI1_RMP	RW	2'b0	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: Reserved 10: Reserved 11: TIM14 channel 1 is connected to the MCU clock output (MCO). This configuration is determined by setting the MCO [2: 0] of the RCC_CFG register.

17. Low-power timer (LPTIM)

17.1. Introduction

The LPTIM is a 16-bit timer. The ability of LPTIM to wake up the system from low power mode makes it suitable for implementing low power applications.

The LPTIM can be counted using high frequency clocks (APB clocks) and low frequency clocks (LSI/LSE) that provide the LPTIM with the required functionality and performance while minimizing power consumption.

17.2. LPTIM main features

- 16 bit upcounter
- 3-bit prescaler with 8 possible dividing factors (1, 2, 4, 8, 16, 32, 64, 128)
- Selectable clock
  - Internal clock sources: LSE, LSI or APB clock
- 16 bit ARR autoreload register
- Continuous/One-shot mode

17.3. LPTIM functional description

17.3.1. LPTIM block diagram

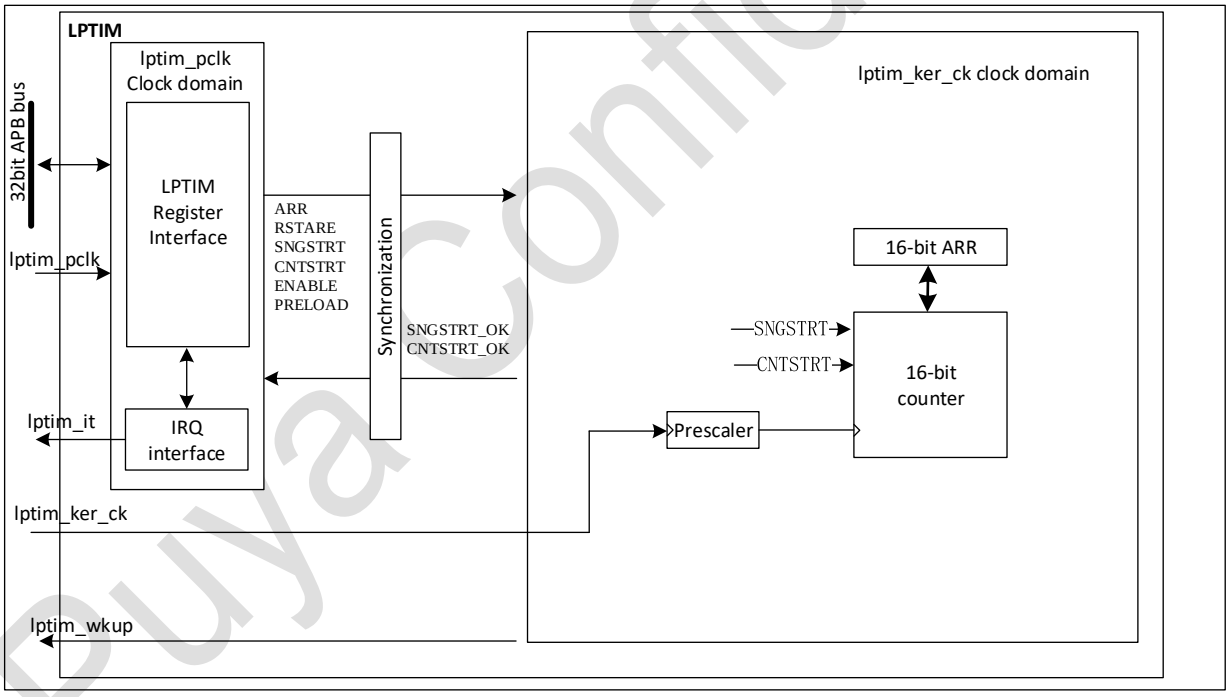

Figure 17-1 Low-power timer block diagram

17.3.2. LPTIM reset and clocks

The LPTIM can be clocked using several clock sources.

With the RCC module, it can be clocked using an internal clock signal (this clock signal can be selected among APB clock, LSI, LSE sources).

17.3.3. Prescaler

The LPTIM 16-bit counter is preceded by a configurable power-of-2 prescaler. The prescaler division ratio is controlled by the PRESC[2:0].

The table below lists all the possible division ratios:

Table 17-1 Prescaler division ratios

programming	dividing factor
000	/1
001	/2
010	/4
011	/8
100 COMP block diagram .	/16
101	/32
110 COMP block diagram .	/64
111	/128

Note: When LPTIM_KER_CK selects PCLK (configured via RCC_CCIPR.LPTIMSEL), the prescaler must be set to a division factor of 2 or higher.

17.3.4. Operating mode

LPTIM has two operating modes as follows:

- **One-shot mode**: the timer is started from a trigger event and stops when reaching the ARR value.

To enable the one-shot counting, the SNGSTRT bit must be set.

A new trigger event will re-start the timer. Any trigger event occurring after the counter starts and before the counter reaches ARR will be discarded.

- **The Continuous mode**: the timer is free running, the timer is started from a trigger event and never stops until the timer is disabled.

To enable the continuous counting, the LPTIM_CR.CNTSTRT bit must be set.

LPTIM_CR.CNTSTRT is set will start the counter for continuous counting.

The one-shot mode and the continuous mode can be converted to each other:

- If the Continuous mode was previously selected, setting SNGSTRT will switch the LPTIM to the One-shot mode. The counter (if active) will stop as soon as it reaches ARR.
- If the One-shot mode was previously selected, setting LPTIM_CR.CNTSTRT will switch the LPTIM to the Continuous mode. The counter (if active) will restart as soon as it reaches ARR.

17.3.5. Register update

The PRELOAD bit controls how the LPTIM_ARR register is updated:

- When the PRELOAD bit is reset to ‘0’, the LPTIM_ARR and the LPTIM_CMP registers are immediately updated after any write access.
- When the PRELOAD bit is set to ‘1’, the LPTIM_ARR and the LPTIM_CMP registers are updated at the end of the current period, if the timer has been already started.

The LPTIM APB interface and the LPTIM kernel logic use different clocks, so there is some latency between the APB write and the moment when these values are available to the counter comparator. Within this latency period, any additional write into these registers must be avoided.

The ARROK flag in the LPTIM_ISR register indicates when the write operation is completed to the LPTIM_ARR register.

After a write to the LPTIM_ARR register, a new write operation to the same register can only be performed when the previous write operation is completed. Any successive write before respectively the ARROK flag be set, will lead to unpredictable results.

17.3.6. Timer enable

The LPTIM counter can be used to count external events.

The ENABLE bit located in the LPTIM_CR register is used to enable/disable the LPTIM kernel logic. After setting the ENABLE bit, a delay of two counter clock is needed before the LPTIM is actually enabled.

The LPTIM_CFGR and LPTIM_IER registers must be modified only when the LPTIM is disabled.

17.3.7. Timer counter reset

In order to reset the content of LPTIM_CNT register to zero, reset mechanism is implemented:

The synchronous reset mechanism:

The synchronous reset is controlled by the COUNTRST bit in the LPTIM_CR register. After setting the COUNTRST bit-field to '1', the reset signal is propagated in the LPTIM kernel clock domain. So it is important to note that a few clock pulses of the LPTIM kernel logic will elapse before the reset is taken into account. This will make the LPTIM counter count few extra pluses between the time when the reset is trigger, and it becomes effective.

Since the COUNTRST bit is located in the APB clock domain and the LPTIM counter is located in the LPTIM kernel clock domain, a delay of 3 clock cycles of the kernel clock is needed to synchronize the reset signal issued by the APB clock domain when writing '1' to the COUNTRST bit.

17.3.8. Debug mode

When the microcontroller enters debug mode (core halted), the LPTIM counter either continues to work normally or stops, depending on the DBG_LPTIM_STOP configuration bit in the DBG module.

17.4. LPTIM low-power modes

Table 17-2 Effect of low-power modes on the LPTIM

Mode	Description
Sleep	No effect. LPTIM interrupts cause the device to exit Sleep mode.
Stop	No effect when LPTIM is clocked by LSE or LSI. LPTIM interrupts cause the device to exit Stop mode.

17.5. LPTIM interrupts

The following events generate an interrupt/wake-up event, if they are enabled through the LPTIM_IER register:

Auto-reload match

Note: If any bit in the LPTIM_IER register (Interrupt Enable Register) is set after that its corresponding flag in the LPTIM_ISR register (Status Register) is set, the interrupt is not asserted.

Table 17-3 Interrupt events

Interrupt event	Description
Auto-reload match	Interrupt flag is raised when the content of the Counter register (LPTIM_CNT) matches the content of the Auto-reload register (LPTIM_ARR).
Auto-reload register update OK	Interrupt flag is raised when the write operation to the LPTIM_ARR register is complete.

17.6. LPTIM registers

17.6.1. LPTIM interrupt and status register (LPTIM_ISR)

Reset value: 0x0000 0000

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:5	Reserved	-	-	-
4	ARROK	R	0	Auto-reload register update OK ARROK is set by hardware to inform application that the APB bus write operation to the LPTIM_ARR register has been successfully completed. ARROK flag can be cleared by writing 1 to the ARROKCF bit in the LPTIM_ICR register.
3:2	Reserved	-	-	-
1	ARRM	R	0	Auto-reload match ARRM is set by hardware to inform application that LPTIM_CNT register's value reached the LPTIM_ARR register's value. ARRM flag can be cleared by writing 1 to the ARRMCF bit in the LPTIM_ICR register.
0	Reserved	-	-	-

17.6.2. LPTIM interrupt clear register (LPTIM_ICR)

Address offset: 0x004

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:5	Reserved	-	-	-
4	ARROKCF	RW	0	Auto-reload register update OK clear flag Writing 1 to this bit clears the ARRM flag in the LPTIM_ISR register
3:2	Reserved	-	-	-
1	ARRMCF	RW	0	Auto-reload match clear flag Writing 1 to this bit clears the ARRM flag in the LPTIM_ISR register
0	Reserved	-	-	-

17.6.3. LPTIM interrupt enable register (LPTIM_IER)

Address offset: 0x008

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	-
4	ARROKIE	RW	0	Auto-reload register update OK interrupt enable 0: ARROK interrupt disabled 1: ARROK interrupt enabled
3:2	Reserved	-	-	-
1	ARRMIE	RW	0	Auto-reload match interrupt enable 0: ARRM interrupt disabled 1: ARRM interrupt enabled
0	Reserved	-	-	-

17.6.4. LPTIM configuration register (LPTIM_CFGR)

Address offset: 0x00C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	-
22	PRELOAD	RW	0	Register update mode The preload bit controls the LPTIM_ARR register update modality 0: Registers are updated after each APB bus write access 1: Registers are updated at the end of the current LPTIM period
21:12	Reserved	-	-	-
11:9	PRESC[2:0]	RW	3'h0	Clock prescaler The PRESC bits configure the prescaler division factor. It can be one among the following division factors: 000:/1 001:/2 010:/4 011:/8 100:/16 101:/32 110:/64 111:/128
8:0	Reserved	-	-	-

17.6.5. LPTIM control register (LPTIM_CR)

Address offset: 0x010

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:5	Reserved	-	-	-
4	RSTARE	RW	0	Reset after read enable This bit is set and cleared by software. When RSTARE is set to '1', any read access to LPTIM_CNT register will asynchronously reset LPTIM_CNT register content.
3	COUNTRST	RS	0	Timer counter reset This bit is set by software and cleared by hardware. When set to '1' this bit will trigger a synchronous reset of the LPTIM_CNT counter register. Due to the synchronous nature of this reset, it only takes place after a synchronization delay of 3 LPTIM core clock cycles (LPTIM core clock may be different from APB clock). Caution: COUNTRST must never be set to '1' by software before it is already cleared to '0' by hardware. Software should consequently check that COUNTRST bit is already cleared to '0' before attempting to set it to '1'.
2	CNTSTRT	RW	0	Timer start in Continuous mode This bit is set by software and setting this bit starts the LPTIM in continuous mode. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between the LPTIM_ARR and LPTIM_CNT registers. The LPTIM counter keeps counting in continuous mode. Note: This bit can be set only when the LPTIM is enabled. It will be automatically reset by hardware.
1	SNGSTRT	RW	0	LPTIM start in Single mode This bit is set by software and cleared by hardware. Setting this bit starts the LPTIM in single pulse mode Note: This bit can be set only when the LPTIM is enabled. It will be automatically reset by hardware.
0	ENABLE	RW	0	LPTIM enable. The ENABLE bit is set and cleared by software. 0: LPTIM is disabled 1: LPTIM is enabled

17.6.6. LPTIM autoreload register (LPTIM_ARR)

Address offset: 0x018

Reset value: 0x0000 0001

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:0	ARR	RW	16'h1	Auto reload value ARR is the auto-reload value for the LPTIM. The LPTIM_ARR register must only be modified when the LPTIM is enabled.

17.6.7. LPTIM counter register (LPTIM_CNT)

Address offset: 0x01C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:16	Reserved	-	-	-
15:0	CNT	R	16'h0	Counter value When the LPTIM is running with an asynchronous clock, reading the LPTIM_CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical. A read access can be considered reliable when the values of the two consecutive read accesses are equal.

18. Independent watchdog (IWDG)

18.1. Introduction

The devices feature an embedded watchdog peripheral that offers a combination of high safety level, timing accuracy and flexibility of use. The Independent watchdog peripheral detects and solves malfunctions due to software failure, and triggers system reset when the counter reaches a given timeout value.

The IWDG is clocked by LSI/LSE 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.

18.2. IWDG main features

- Free-running downcounter
- Clocked from LSI/LSE (can operate in Stop mode)
- Conditional reset
  - Reset when the downcounter value becomes lower than 0x000

18.3. IWDG functional description

18.3.1. IWDG block diagram

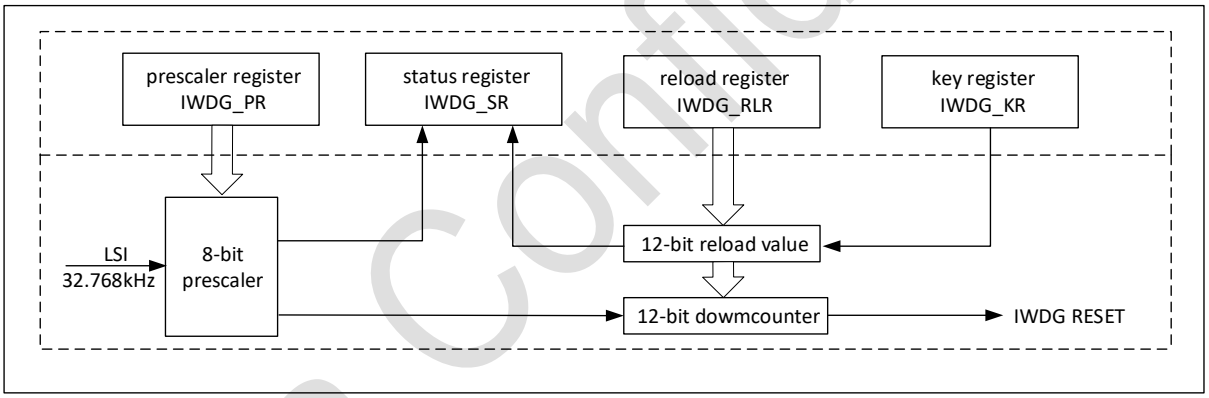

Figure 18-1 IWDG block diagram

When the independent watchdog is started by writing the value 0x0000 CCCC in the IWDG key register (IWDG_KR), the counter starts counting down from the reset value of 0xFFF. When it reaches the end of count value (0x000) a reset signal is generated (IWDG reset).

Whenever the key value 0x0000 AAAA is written in the IWDG key register (IWDG_KR), the IWDG_RLR value in IWDG reload register is reloaded in the counter and the watchdog reset is prevented.

Once running, the IWDG cannot be stopped.

18.3.2. 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.

18.3.3. Register access protection

Write access to IWDG prescaler register (IWDG_PR) and IWDG reload register (IWDG_RLR) is protected. A write access to this register with a different value breaks the sequence.

A status register is available to indicate that an update of the prescaler or of the downcounter reload value is ongoing.

18.3.4. Debug 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.

18.4. IWDG registers

18.4.1. 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	-	-	-
15:0	KEY[15:0]	W	16'h0	Key value These bits must be written by software at regular intervals with the key value 0xAAAA, otherwise the watchdog generates a reset when the counter reaches 0. Writing the key value 0x5555 to enable access to the IWDG_PR and IWDG_RLR registers. Writing the key value 0xCCCC starts the watchdog (except if the hardware watchdog option is selected).

18.4.2. IWDG prescaler register (IWDG_PR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	-
2:0	PR[2:0]	RW	3'h0	Prescaler divider They are written by software to select the prescaler divider feeding the counter clock. PVU bit of the IWDG status register (IWDG_SR) must be reset in order to be able to change the prescaler divider. 000: divider /4. 001: divider /8. 010: divider /16. 011: divider /32. 100: divider /64. 101: divider /128.

Bit	Name	R/W	Reset Value	Function
				110: divider /256. 111: divider /256.

18.4.3. 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]											
-				RW											

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	-
11:0	RL[11:0]	RW	12'h0	IWDG counter reload value They are written by software to define the value to be loaded in the watchdog counter each time the value0xAAAA is written in the IWDG key register (IWDG_KR). The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. The RVU bit in the IWDG status register (IWDG_SR) must be reset to be able to change the reload value.

18.4.4. 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
-														R	R

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	-
1	RVU	R	0	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the window value is ongoing. It is reset by hardware when the re-load value update operation is completed.
0	PVU	R	0	Watchdog prescaler value update This bit is set by hardware to indicate that an update of the prescaler value is ongoing. It is reset by hardware when the prescaler update operation is completed.

Note: It is mandatory to wait until RVU bit is reset before changing the reload value, to wait until PVU bit is reset before changing the prescaler value. 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.

19. Inter-integrated circuit (I^2C) interface

19.1. 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) and Fast-mode (Fm).

19.2. I^2C main features

- Slave and master modes
- Multi-master function: can be used as a master or a slave
- Supports different communication speeds
  - Standard mode (Sm): up to 100 kHz
  - Fast mode (Fm): up to 400 kHz
- 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 bit
  - Transmit/receive mode flag bit
  - Byte transfer completion flag bit
  - I^2C busy flag bit
- Error flag bit
  - Host Arbitration Lost
  - ACK failure after address/data transfer
  - Detection of misplaced start or stop condition
  - Overrun/Underrun (clock stretching function disable)
- Optional clock elongation function
- Software reset
- Analog noise filter function

19.3. I²C functional description

19.3.1. I²C block diagram

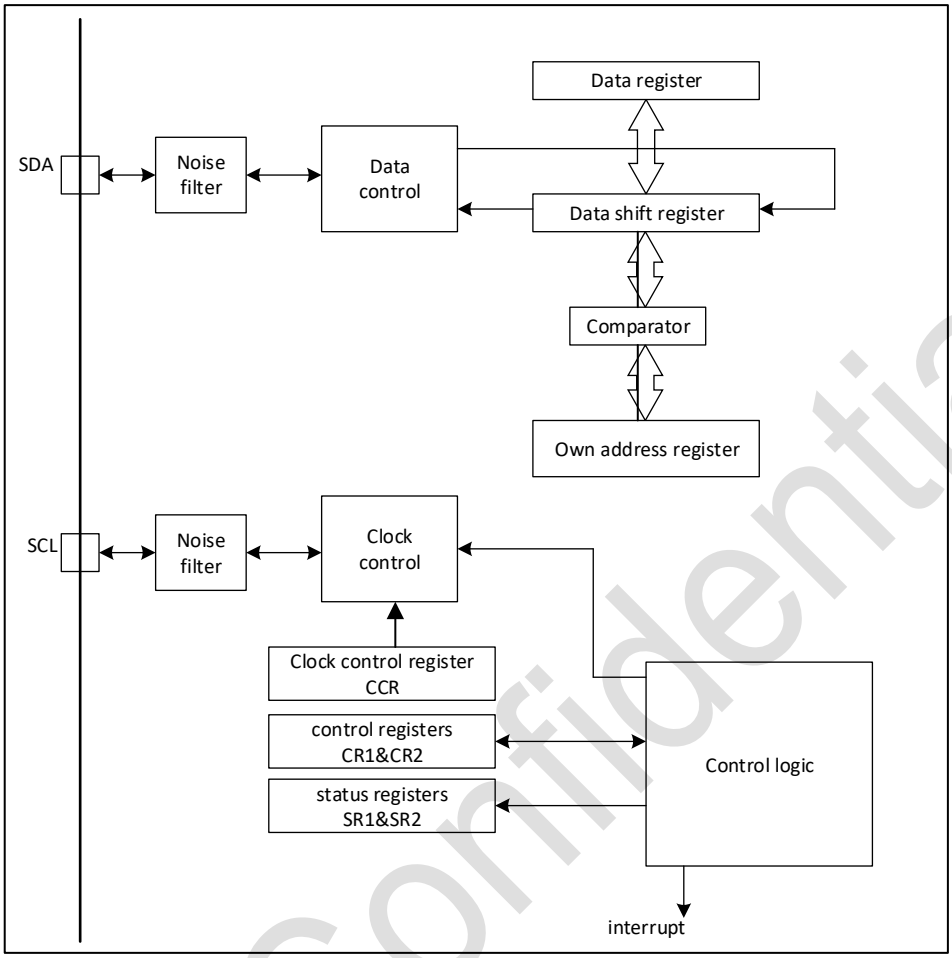

Figure 19-1 I²C block diagram

19.3.2. 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.

19.3.2.1. 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.

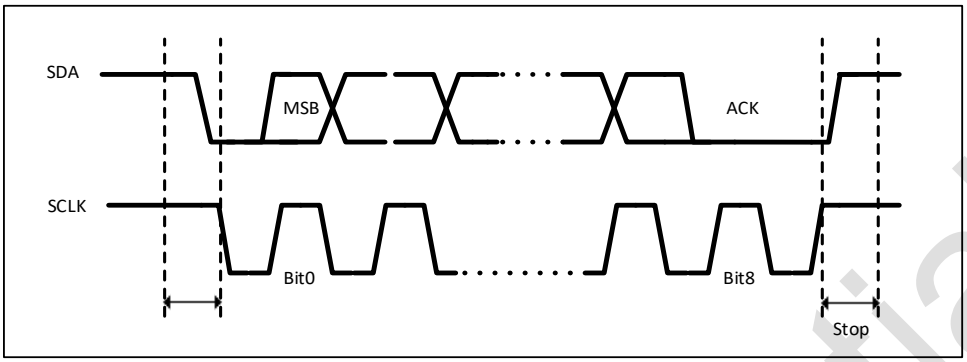

Figure 19-2 I²C bus protocol

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.

19.3.3. I²C initialization

19.3.3.1. Enable/turn off I²C module

The I²C peripheral clock must be configured and enabled by the I2C_EN bit in the RCC_APBENR1 register. Then the I²C can be enabled by setting the PE bit in the I2C_CR1 register.

19.3.3.2. 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.

19.3.4. 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

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.

19.3.4.1. Slave transmitter

Following the address reception and after clearing ADDR, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave stretches SCL low until ADDR is cleared and DR filled with the data to be sent.

When the acknowledge pulse is received: The TxE bit is set by hardware with an interrupt if the ITEVFEN and the ITBUFEN bits are set.

If TxE is set and some data were not written in the I2C_DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read to I2C_SR1 followed by a write to the I2C_DR register, stretching SCL low.

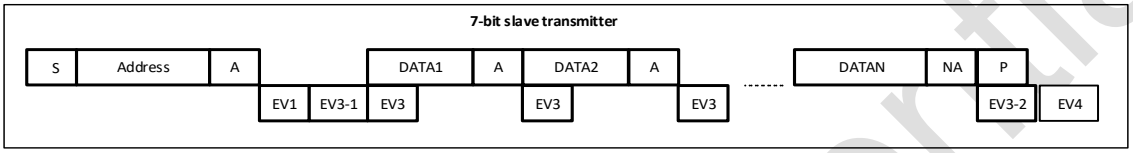

Figure 19-3 Transfer sequence diagram for slave transmitter

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, AF is cleared by writing '0'.

Notes:

1. The EV1 event stretches SCL low until the end of the corresponding software sequence.
2. The EV3 software sequence must be completed before the end of the current byte transfer.

19.3.4.2. 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. If ITEVTEN and ITBUFEN bits are set, an interrupt is generated.

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).

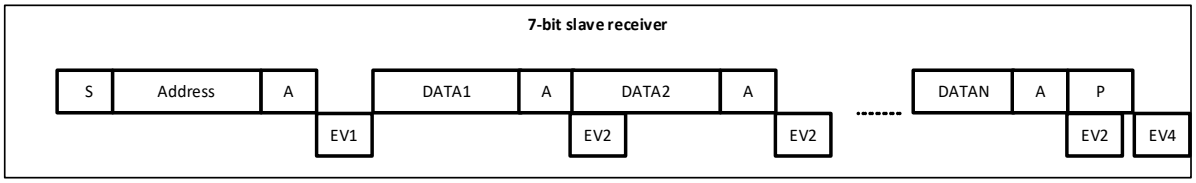

Figure 19-4 Transfer sequence diagram for slave receiver

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 I^2C 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.

19.3.4.3. 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 I2C_SR1 register followed by a write to the I2C_CR1 register. (See EV4 in figure above)

19.3.5. I^2C master mode

In Master mode, the I^2C 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

19.3.5.1. 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 I2C_TRISE register after rising edge generation.

- If the SCL line is low, it means that a slave is stretching the bus, and the high level counter stops until the SCL line is detected high. This allows to guarantee the minimum HIGH period of the SCL clock parameter.
- If the SCL line is high, the high level counter keeps on counting.

Indeed, the feedback loop from the SCL rising edge generation by the peripheral to the SCL rising edge detection by the peripheral takes time even if no slave stretches the clock. This loopback duration is linked to the SCL rising time, plus delay due to the noise filter present on the SCL input path, plus delay due to internal SCL input synchronization with APB clock. The maximum time used by the feedback loop is programmed in the TRISE bits, so that the SCL frequency remains stable whatever the SCL rising time.

19.3.5.2. 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 I2C_SR1 register followed by a write in the I2C_DR register with the Slave address.

19.3.5.3. 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 I2C_SR1 register followed by a read of the I2C_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.

19.3.5.4. Master transmitter

Following the address transmission and after clearing ADDR, the master sends bytes from the DR register to the SDA line via the internal shift register.

The master waits until the first data byte is written into DR register (see EV8_1 in figure below).

When the acknowledge pulse is received, the TxE bit is set by hardware and an interrupt isgenerated if the ITEVFEN and ITBUFEN bits are set.

If TxE is set and some data were not written in the DR register before the end of the next data transmission, the BTF bit is set. The interface waits until BTF is cleared by a read from I2C_SR1 followed by a write to I2C_DR, stretching SCL low.

Closing slave communication

Figure 19-5 Transfer sequence diagram for master transmitterAfter the last byte is written to the DR register, the STOP bit is set by software to generate a Stop condition (see EV8_2 in figure below). The interface automatically goes back to slave mode (MSL bit cleared).

Note: Stop condition should be programmed during EV8_2 event, when either TxE or BTF is set.

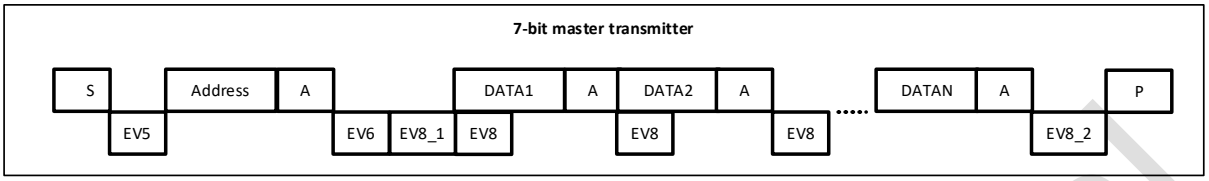

Figure 19-5 Transfer sequence diagram for master transmitter

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event

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 is not empty, data register bit wide, write Data2 to DR register, this bit is cleared

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.

19.3.5.5. 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 hardware setting RxNE = 1 generates an interrupt if the INEVFEN and ITBUFEN bits are set.

If the RxNE bit is set and the data in the DR register is not read before the end of the last data reception, the BTF bit is set by hardware and the interface waits until BTF is cleared by a read in the I2C_SR1 register followed by a read in the I2C_DR register, stretching SCL low.

Closing slave communication

Method 1: This method is for the case when the 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.

- 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).
- 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).

- 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). After the Stop condition generation, the interface goes automatically back to slave mode (MSL bit cleared).

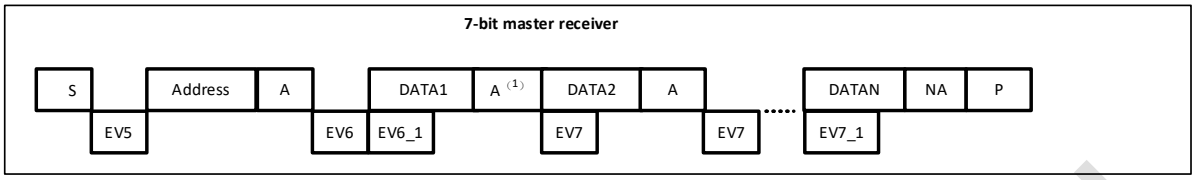

Figure 19-6 Method 1: transfer sequence diagram for master receiver

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event

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

Notes:

1. If a single byte is received, it is NA.
2. The EV5 and EV6 events stretch SCL low until the end of the corresponding software sequence.
3. The EV7 software sequence must complete before the end of the current byte transfer. In 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.

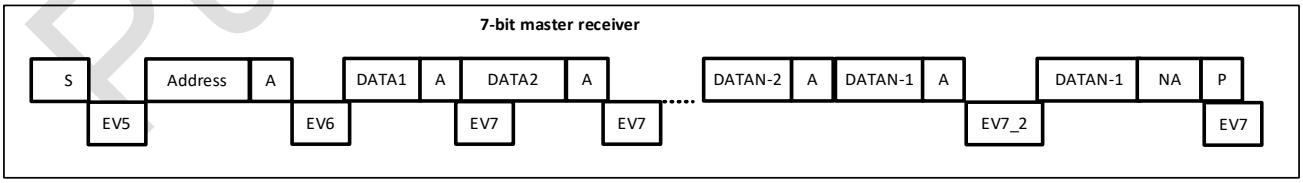

Figure 19-7 Method 2: transfer sequence diagram for master receiver when N>2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event

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

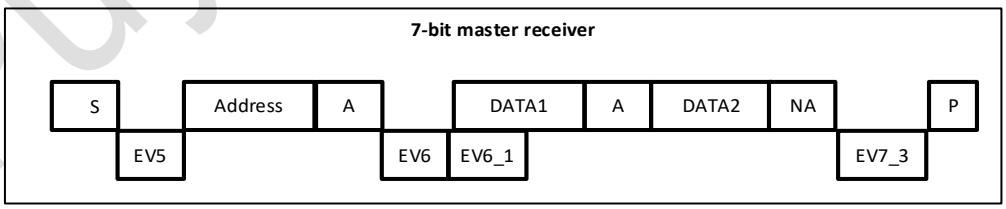

Figure 19-8 Method 2: transfer sequence diagram for master receiver when N=2

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event

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.

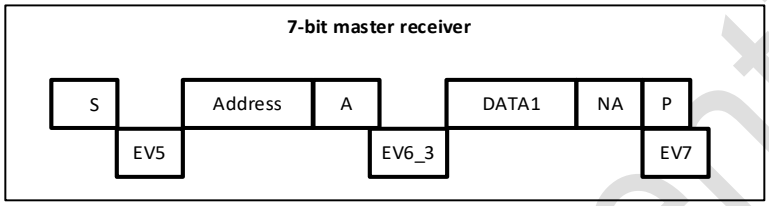

Figure 19-9 Method 2: transfer sequence diagram for master receiver when N=1

Legend: S= Start, Sr = Repeated Start, P= Stop, A= Acknowledge, EVx= Event

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 and EV6_3 events stretch SCL low until the end of the corresponding software sequence.

19.3.6. Error conditions

19.3.6.1. Bus error (BERR)

This error occurs when the I^2C 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.

19.3.6.2. 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

19.3.6.3. 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.
- Lines are released by hardware

19.3.6.4. 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:

- The last received byte is lost.
- In case of Overrun error, software should clear the RxNE bit and the transmitter should re-transmit the last received byte.

Underrun error can occur in slave mode when clock stretching is disabled and the 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.

19.3.7. SDA/SCL line control

- If clock stretching is enabled:
  - Transmitter mode: If TxE=1 and BTF=1: the interface holds the clock line low before transmission to wait for the microcontroller to read I2C_SR1 and then write the byte in the Data register (both DR and shift register are empty).
  - Receiver mode: If RxNE=1 and BTF=1: the interface holds the clock line low after reception to wait for the microcontroller to read SR1 and then read the byte in the Data register (both DR and shift register are full).
- If clock 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.

19.4. I²C interrupts

Table 19-1 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	
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	

19.5. I²C registers

The peripheral registers have to be accessed by half-words (16 bits) or words (32 bits).

19.5.1. I²C control register 1 (I2C_CR1)

Address offset: 0x00

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res	Res	Res	POS	ACK	STOP	START	NO STRETCH	ENGC	Res	Res	Res	Res	Res	PE
RW	-	-	-	RW	RW	RW	RW	RW	RW	-	-	-	-	-	RW

Bit	Name	R/W	Reset value	Function
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	-	-	-
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.
8	START	RW	0	Start generation

Bit	Name	R/W	Reset value	Function
				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.
5:1	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.

19.5.2. I^2C 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	ITBUFEN	ITEVTEN	ITERREN	Res	Res	FREQ[5:0]					
-	-	-	-	-	RW	RW	RW	-	-	RW					

Bit	Name	R/W	Reset Value	Function
15:11	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 This interrupt is generated when: 1. SB = 1 (Master) 2. ADDR = 1 (Master/Slave) 3. STOPF = 1 (Slave) 4. BTF = 1 with no TxE or RxNE event 5. TxE event to 1 if ITBUFEN = 1 6. RxNE event to 1if ITBUFEN = 1
8	ITERREN	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled This interrupt is generated when: 1.BERR=1 2.ARLO=1 3.AF=1 4.OVR=1
7:6	Reserved	-	-	-
5:0	FREQ	RW	6'h0	I^2C clock frequency

Bit	Name	R/W	Reset Value	Function
				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 allowed frequency is 4 MHz (Standard mode that is 100k) and 8 MHz (400k) the maximum frequency is limited by the maximum APB frequency. 000000: Disabled 000001: Disabled 000100: 4 MHz ….. 100100: 36 MHz ….. 110000: 48 MHz > 100100: Disabled.

19.5.3. 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	Res		ADD[7:1]							Res
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	-

Bit	Name	R/W	Reset Value	Function
14:8	Reserved	-	-	-
7:1	ADD[7:1]	RW	6'h0	Bits 7:1 of interface address
0	Reserved	-	-	-

19.5.4. 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							

Bit	Name	R/W	Reset value	Function
15:8	Reserved	-	-	-
7:0	DR[7:0]	RW	8'h0	The 8-bit data register, actually two independent caches inside the chip share an address, which are used to store the received data (RX_DR) and place the data to be sent to the bus (TX_DR) respectively. **Transmitter mode:** Byte transmission starts automatically when a byte is written in the DR (TX_DR actually) register. A continuous transmit stream can be maintained if the next data to be transmitted is put in DR once the transmission is started (TxE=1). **Receiver mode:** Received byte is copied into DR (RX_DR actually) register (RxNE=1). A continuous transmit stream can be maintained if DR is read before the next data byte is received (RxNE=1). Notes: 1. In slave mode, the address is not copied into 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.

19.5.5. 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	Res	Res	Res	OVR	AF	ARLO	BERR	TxE	RxNE	Res	STOPF	Res	BTF	ADDR	SB
-	-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	R	R	-	R	-	R	R	R

Bit	Name	R/W	Reset value	Function
15:12	Reserved	-	-	-
11	OVR	RC_W0	0	Overrun/Underrun 0: No overrun/underrun 1: Overrun or underrun Set by hardware in slave mode when NOSTRETCH=1 and: In reception when a new byte is received (including ACK pulse) and the DR register has not been read yet. New received byte is lost. In transmission when a new byte should be sent and the DR register has not been written yet. The same byte is sent twice. Cleared by software writing 0, or by hardware when PE=0. Note: If the DR write occurs very close to SCL rising edge, the sent data is unspecified and a hold timing error occurs.
10	AF	RC_W0	0	Acknowledge failure 0: No acknowledge failure 1: Acknowledge failure Set by hardware when no acknowledge is returned. Cleared by software writing 0, or by hardware when PE=0.
9	ARLO	RC_W0	0	Arbitration lost (master mode) 0: No Arbitration Lost detected 1: Arbitration Lost detected Set by hardware when the interface loses the arbitration of the bus to another master. Cleared by software writing 0, or by hardware when PE=0. After an ARLO event the interface switches back automatically to Slave mode (MSL=0).
8	BERR	RC_W0	0	Bus error 0: No misplaced Start or Stop condition 1: Misplaced Start or Stop condition Set by hardware when the interface detects wrong start or end conditions during a byte transfer. Cleared by software writing 0, or by hardware when PE=0.
7	TxE	R	0	Data register empty (transmitters) 0: Data register not empty 1: Data register empty 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	-	-	-
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. Note: The STOPF bit is not set after a NACK reception.
3	Reserved	-	-	-

Bit	Name	R/W	Reset value	Function
2	BTF	R	0	Byte transfer complete flag 0: Data byte transfer not done 1: Data byte transfer succeeded 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 matched (Slave):** 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.

19.5.6. 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
15:5	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	-	-	-
2	TRA	R	0	Transmit/receive flags 0: Data bytes received 1: Data bytes transmitted This bit is set depending on the R/W bit of the address byte, at the end of total address phase. It is also cleared by hardware after detection of Stop condition (STOPF=1), repeated Start condition, loss of bus arbitration (ARLO=1), or when PE=0.
1	BUSY	R	0	Bus busy 0: No communication on the bus 1: Communication ongoing on the bus

Bit	Name	R/W	Reset value	Function
				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 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.

19.5.7. 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	Res	Res	CCR[11:0]											
RW	RW	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset value	Function
15	F/S	RW	0	I²C master mode selection 0: Sm mode I²C 1: Fm mode I²C
14	DUTY	RW	0	Fm mode duty cycle 0: Fm mode t_{low}/t_{high} = 2 1: Fm mode t_{low}/t_{high} = 16/9
13:12	Reserved	-	-	-
11:0	CCR[11:0]	RW	12'h0	Clock control register in Fm/Sm mode (Master mode) Controls the SCL clock in master mode. ■ Standard mode: — T_{high}=CCR x T_{pclk} — Tlow =CCR x Tpclk ■ Fast mode: — DUTY=0: Thigh=CCR x Tpclk Tlow =2 x CCR x Tpclk — DUTY=1: -T_{high}=9 x CCR x T_{pclk} -T_{low} =16 x CCR x T_{pclk} Notes: 1. The minimum allowed value is 0x04, except in Fm mode where the minimum allowed value is 0x01. $T_{high}=t_{r(SCL)}+t_{w(SCLH)}$ $T_{low}=t_{r(SCL)}+t_{w(SCLL)}$ 2. These delays have no filters 3. This register can only be configured when PE = 0.

19.5.8. I²C TRISE register (I2C_TRISE)

Address offset: 0x20

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
15:6	Reserved	-	-	-
5:0	TRISE	RW	6'h0	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.

Bit	Name	R/W	Reset value	Function
				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, in the I2C_CR2 register, the value of FREQ[5:0] bits is equal to 0x08 and TPCLK1 = 125 ns therefore the TRISE[5:0] bits must be programmed with 09h.(1000 ns / 125 ns = 8 + 1 = 9). The filter value can also be added to TRISE[5:0]. 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).

20. Universal synchronous asynchronous receiver transmitter (USART)

20.1. Introduction

The USART offers a flexible means to perform Full-duplex data exchange with external equipments. USART utilizes a fractional baud rate generator to provide a wide range of baud rate selections.

The USART supports both synchronous one-way and Half-duplex Single-wire communications. It also supports multiprocessor communications.

20.2. USART main features

- Full-duplex asynchronous communication
- NRZ Standard Format
- Configurable with 16x or 8x oversampling for increased flexibility in speed and clock tolerance
- A common programmable transmit and receive baud rate
- Auto baud rate detection
- Programmable data length 8 bits or 9 bits
- Configurable STOP bits (1 or 2 bits)
- Synchronous mode and clock output function for synchronous slave communication
- Single-wire Half-duplex communications
- Independent transmit and receive enable bits
- Hardware flow control
- Detection flag
  - Receive buffer full
  - Send buffer empty
  - End of transmission
- Parity control
  - Send check digit
  - Verify received data
- Flagged interrupt source
  - CTS change
  - Send register empty
  - Send Complete
  - Receive full data register
  - Idle bus detected
  - Overrun error
  - Framing error
  - Noisy operation
- Multiprocessor communication
  - If the addresses do not match, enter silent mode
  - Wake-up from silent mode: by idle detection and address flag detection

20.3. USART functional description

The interface is externally connected to another device by three pins. Any USART bidirectional communications require a minimum of two pins: Receive Data In (R_X) and Transmit Data Out (T_X).

R_X: Receive data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

T_X: Transmit data output. When the transmitter is disabled, the output pin returns to its IO port configuration. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level. In single-wire mode, this IO is used to transmit and receive the data.

USART mainly includes the following features:

- An idle line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) the least significant bit first
- 1, 2 Stop bits indicating that the frame is complete
- A status register (USART_SR)
- A Data register (USART_DR)
- A baud rate register (USART_BRR), 12-bit mantissa and 4-bit fraction.

The following pin is required to interface in synchronous mode:

C_K: Transmitter clock output.

This pin outputs the transmitter data clock for synchronous transmission (no clock pulses on start bit and stop bit, and a software option to send a clock pulse on the last data bit). In parallel data can be received synchronously on RX. This can be used to control peripherals that have shift registers (e.g. LCD drivers). The clock phase and polarity are software programmable.

The following pins are required in hardware flow control mode:

- nCTS: Clear to send blocks the data transmission at the end of the current transfer when high
- nRTS: Request to send indicates that the USART is ready to receive a data (when low).

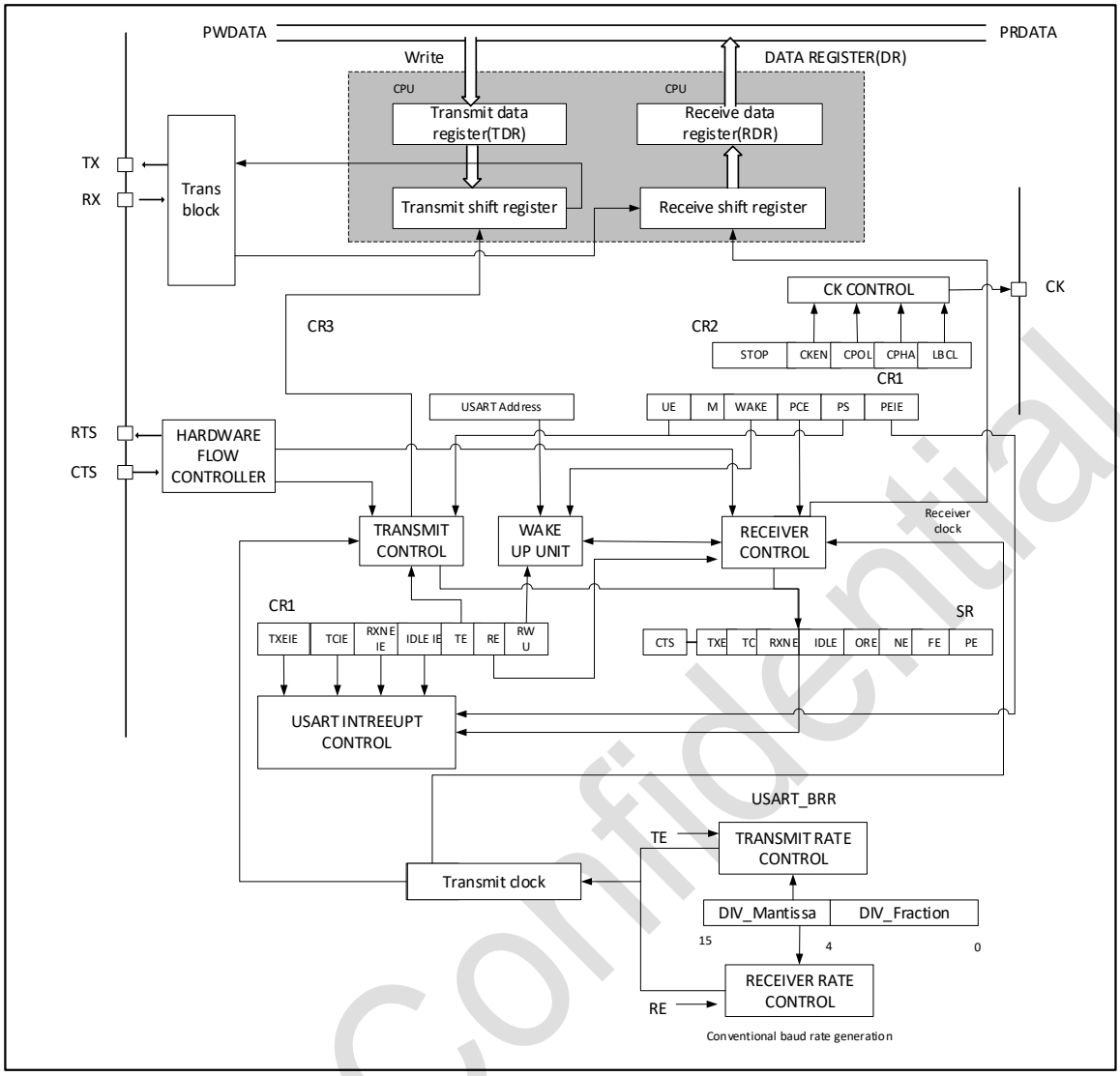

Figure 20-1 USART block diagram

20.3.1. USART character description

Word length may be selected as being either 8 or 9 bits by programming the M bit in the USART_CR1 register. The T_X pin is in low state during the start bit. It is in high state during the stop bit.

An Idle character is interpreted as an entire frame of 1 followed by the start bit of the next frame which contains data (The number of 1's will include the number of stop bits).

Transmission and reception are driven by a common baud rate generator, the clock for each is generated when the enable bit is set respectively for the transmitter and receiver.

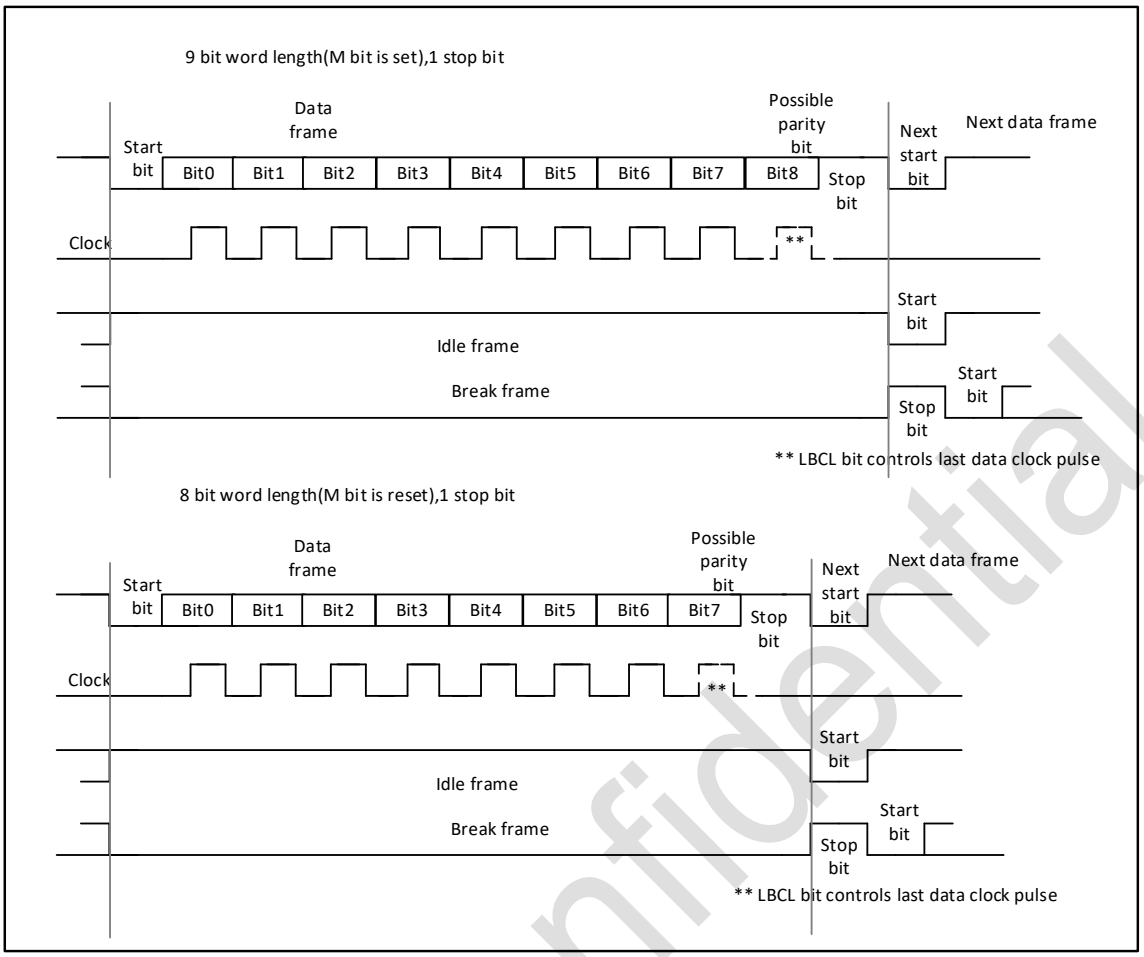

Figure 20-2 Word length programming

20.3.2. Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the transmit enable bit (TE) is set, the data in the transmit shift register is output on the T_X pin.

20.3.2.1. Character transmission

During a USART transmission, data shifts out the least significant bit first on the T_X pin. In this mode, the USART_DR register consists of a buffer (TDR) between the internal bus and the transmit shift register.

Every character is preceded by a start bit, which is a logic level low for one bit period. The character is terminated by a configurable number of stop bits. The following stop bits are supported by USART: 1 and 2 stop bits.

Notes:

The TE bit cannot be reset during data transfer, otherwise the data on the TX pin will be destroyed. Because the baud rate counter stops counting, the current data being transmitted will be lost.

An idle frame will be sent after the TE bit is enabled.

20.3.2.2. Configurable stop bits

The number of stop bits to be transmitted with every character can be programmed in STOP bit, CR2 register.

An idle frame transmission will include the stop bits.

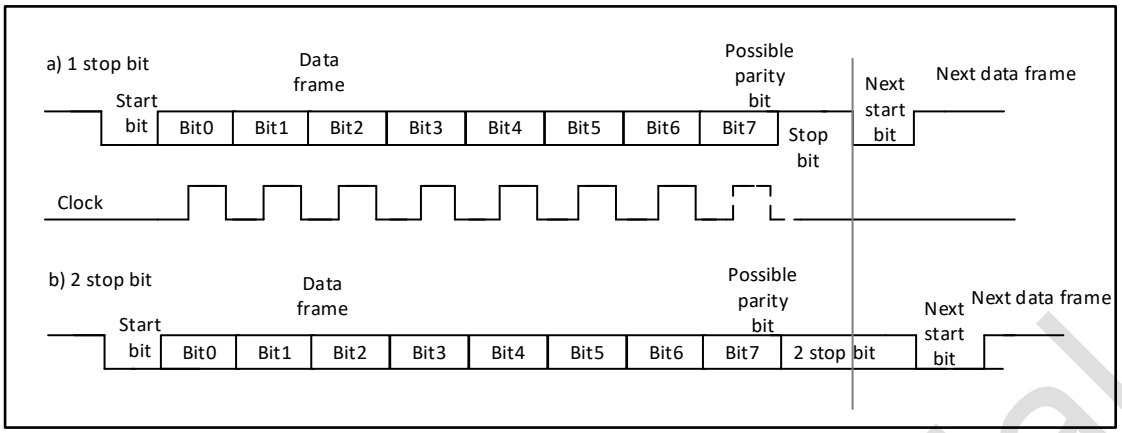

Figure 20-3 Configurable stop bits

Procedure:

1) Enable the USART by writing the UE bit in USART_CR1 register to 1.

2) Program the M bit in USART_CR1 to define the word length.

3) Program the number of stop bits in USART_CR2.

4) Select the desired baud rate using the USART_BRR register.

5) Set the TE bit in USART_CR1 to send an idle frame as first transmission.

6) Write the data to send in the USART_DR register (this clears the TXE bit). Repeat this for each data to be transmitted in case of single buffer.

7) After writing the last data into the USART_DR register, wait until TC=1. This indicates that the transmission of the last frame is complete.

This is required for instance when the USART is disabled or enters the Halt mode to avoid corrupting the last transmission.

20.3.2.3. Single byte communication

The TXE bit is always cleared by a write to the data register. The TXE bit is set by hardware and it indicates:

- The data has been moved from USART_DR to the shift register and the data transmission has started.
- The USART_DR register is empty.
- The next data can be written in the USART_DR register without overwriting the previous data.

This flag generates an interrupt if the TXEIE bit is set.

When a transmission is taking place, a write instruction to the USART_DR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the USART_DR register places the data directly in the shift register, the data transmission starts, and the TXE bit is immediately set.

If a frame is transmitted (after the stop bit) and the TXE bit is set, the TC bit goes high. An interrupt is generated if the TCIE bit is set in the USART_CR1 register.

After writing the last data into the USART_DR register, it is mandatory to wait for TC=1 before disabling the USART or causing the microcontroller to enter the low-power mode.

Note: The TC bit is cleared by writing 0 to it. This clearing sequence is recommended only for Multibuffer communication.

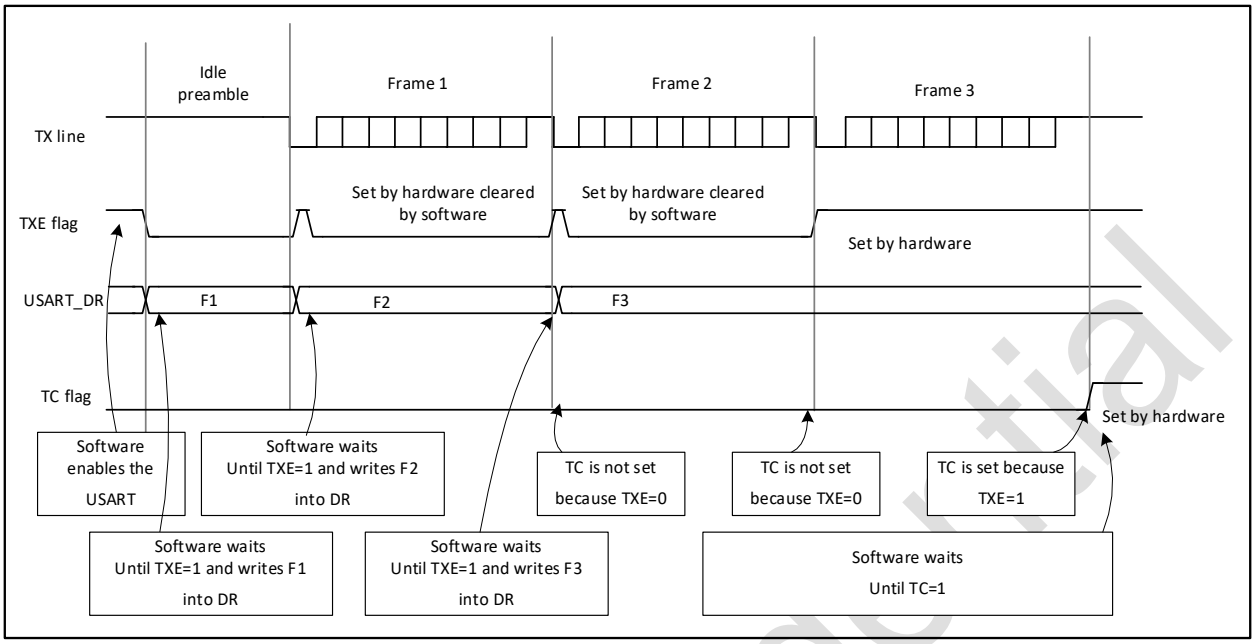

Figure 20-4TC/TXE behavior when transmitting

20.3.2.4. Idle characters

Setting the TE bit drives the USART to send an idle frame before the first data frame.

20.3.2.5. Start bit detection

In the USART, the start bit is detected when a specific sequence of samples is recognized.

The sequence is : 1 1 1 0 X 0 X 0 X 0 0 0 0.

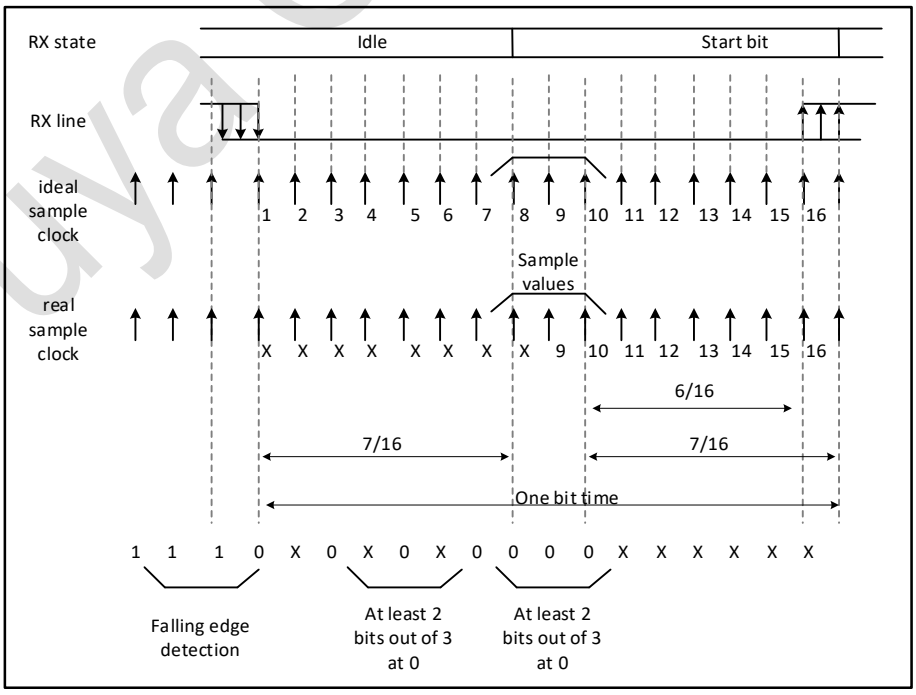

Figure 20-5 Start bit detection

If the sequence is not complete, the start bit detection aborts and the receiver returns to the idle state (no flag is set) where it waits for a falling edge. The start bit is confirmed (RXNE flag set, interrupt generated if RXNEIE=1) if the 3 sampled bits are at 0 (first sampling on the 3rd, 5th and 7th bits finds the 3 bits at 0 and second sampling on the 8th, 9th and 10th bits also finds the 3 bits at 0).

The start bit is validated (RXNE flag set, interrupt generated if RXNEIE=1) but the NE noise flag is set if, for both samplings, at least 2 out of the 3 sampled bits are at 0 (sampling on the 3rd, 5th and 7th bits and sampling on the 8th, 9th and 10th bits). If this condition is not met, the start detection aborts and the receiver returns to the idle state (no flag is set).

If, for one of the samplings (sampling on the 3rd, 5th and 7th bits or sampling on the 8th, 9thand 10th bits), 2 out of the 3 bits are found at 0, the start bit is validated but the NE noise flag bit is set.

20.3.2.6. Character reception

During a USART reception, data shifts in the least significant bit first through the R_X pin. In this mode, the USART_DR register consists of a buffer (RDR) between the internal bus and the received shift register.

Procedure:

1. Enable the USART by writing the UE bit in USART_CR1 register to 1.
2. Program the M bit in USART_CR1 to define the word length.
3. Program the number of stop bits in USART_CR2.
4. Select the desired baud rate using the baud rate register USART_BRR
5. Set the RE bit USART_CR1. This enables the receiver which begins searching for a start bit.

When a character is received:

- The RXNE bit is set. It indicates that the content of the shift register is transferred to the RDR. In other words, data has been received and can be read (as well as its associated error flags).
- An interrupt is generated if the RXNEIE bit is set.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.
- In single buffer mode, clearing the RXNE bit is performed by a software read to the USART_DR register. The RXNE flag can also be cleared by writing a zero to it. The RXNE bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Note: The RE bit should not be reset while receiving data. If the RE bit is disabled during reception, the reception of the current byte will be aborted.

20.3.2.7. Idle characters

When an idle frame is detected, there is the same procedure as a data received character plus an interrupt if the IDLEIE bit is set.

20.3.2.8. Overrun error

An overrun error occurs when a character is received when RXNE has not been reset. Data can not be transferred from the shift register to the RDR register until the RXNE bit is cleared. The RXNE flag is set after every byte received. An overrun error occurs if RXNE flag is set when the next data is received.

When an overrun error occurs:

- The ORE bit is set.

- The RDR content will not be lost. The previous data is available when a read to USART_DR is performed.
- The shift register will be overwritten. After that point, any data received during overrun is lost.
- An interrupt is generated if either the RXNEIE bit is set or EIE bit is set.
- The ORE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

Note: The ORE bit, when set, indicates that at least 1 data has been lost. There are two possibilities:

- If RXNE=1, then the last valid data is stored in the receive register RDR and can be read.
- If RXNE=0, then it means that the last valid data has already been read and thus there is nothing to be read in the RDR.

20.3.2.9. Noise error

Over-sampling techniques are used (except in synchronous mode) for data recovery by discriminating between valid incoming data and noise.

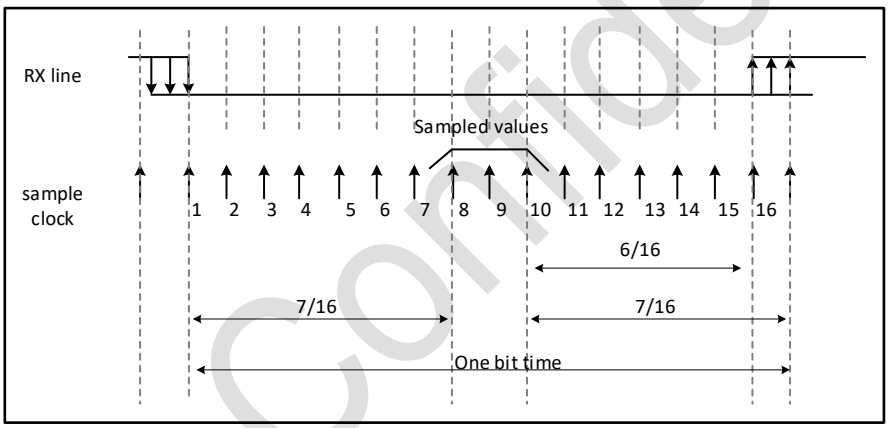

Figure 20-6 Data sampling for noise detection

Table 20-1 Data sampling for noise detection

Sampled value	NE status	Received bit value	Data validity
000	0	0	Valid
001	1	0	Not Valid
010	1	0	Not Valid
011	1	1	Not Valid
100 COMP block diagram .	1	0	Not Valid
101	1	1	Not Valid
110 COMP block diagram .	1	1	Not Valid
111	0	1	Valid

When noise is detected in a frame:

- The NE is set at the rising edge of the RXNE bit.
- The invalid data is transferred from the Shift register to the USART_DR register.
- No interrupt is generated in case of single byte communication. However, since the NE flag bit and the RXNE flag bit are set simultaneously, the RXNE will be set. In case of multibuffer communication an

interrupt will be issued if the EIE bit is set in the USART_CR3 register. The NE bit is reset by a USART_SR register read operation followed by a USART_DR register read operation.

20.3.2.10. Framing error

A framing error is detected when: The stop bit is not recognized on reception at the expected time.

When the framing error is detected:

- The FE bit is set by hardware
- The invalid data is transferred from the Shift register to the USART_DR register.
- No interrupt is generated in case of single byte communication. However, this bit rises at the same time as the RXNE bit which itself generates an interrupt. In case of multibuffer communication an interrupt is issued if the EIE bit is set in the USART_CR3 register.

The FE bit is reset by a read to the USART_SR register followed by a USART_DR register read operation.

20.3.2.11. Configurable stop bits during reception

The number of stop bits to be received can be configured through the STOP bits of CR2: it can be either 1 or 2 in normal mode.

- 1 stop bit: sampling for 1 stop bit is done on the 8th, 9th and 10th samples.
- 2 stop bits: sampling for 2 stop bits is done on the 8th, 9th and 10th samples of the first stop bit. The framing error flag is set if a framing error is detected during the first stop bit. The second stop bit is not checked for framing error. The RXNE flag is set at the end of the first stop bit.

20.3.3. USART baud rate generation

The baud rate for the receiver and transmitter (Rx and Tx) are both set to the value programmed in the Mantissa and Fraction values of USARTDIV.

Tx / Rx baud $=f_{CK}$ */(16*USARTDIV)*

Here f_{CK} is the clock given to the peripheral and USARTDIV is an unsigned number. The 12-bit value is set in the USART_BRR register.

Note: The baud counters are updated with the new value of the Baud registers after a write to USART_BRR. Hence, the Baud rate register value should not be changed during communication.

Example:

If DIV_Mantissa = 27，DIV_Fraction = 12 (USART_BRR=0x1BC), then Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = 12/16 = 0.75

Therefore, USARTDIV = 27.75

20.3.4. Tolerance of the USART receiver to clock deviation

The USART asynchronous receiver operates correctly only if the total clock system deviation is less than the tolerance of the USART receiver. The causes which contribute to the total deviation are:

- DTRA: deviation due to the transmitter error (which also includes the deviation of the transmitter’s local oscillator)
- DQUANT: error due to the baud rate quantization of the receiver
- DREC: deviation of the receiver local oscillator

- DTCL: deviation due to the transmission line (generally due to the transceivers which can introduce an asymmetry between the low-to-high transition timing and the high-to-low transition timing).

DTRA + DQUANT + DREC + DTCL < USART receiver tolerance

The USART receiver tolerance to properly receive data is equal to the maximum tolerated deviation and depends on the following choices:

- 10- or 11-bit character length defined by the M bit in the USART_CR1 register
- Use of fractional baud rate or not

Table 20 -2Tolerance of the USART receiver when DIV_Fraction is 0

M bit	NE is an error	NE doesn’t care
0	3.75 %	4.375%
1	3.41%	3.97%

Table 20-3 Tolerance of the USART receiver when DIV_Fraction is different from 0

M bit	NE is an error	NE doesn’t care
0	3.33%	3.88%
1	3.03%	3.53 %

20.3.5. USART Auto baud rate detection

The USART can detect and automatically set the USART_BRR register value based on the reception of one character. Automatic baud rate detection is useful under following circumstances:

1. The communication speed of the system is not known in advance.
2. The system is using a relatively low accuracy clock source and this mechanism allows the correct baud rate to be obtained without measuring the clock deviation.

The clock source frequency must be compatible with the expected communication speed.

Before activating the auto baud rate detection, the auto baud rate detection mode must be selected.

In these auto baud rate modes, the baud rate is measured several times during the synchronization data reception and each measurement is compared to the previous one.

These modes are the following:

Mode 0: Any character starting with a bit at ‘1’. In this case the USART measures the duration of the start bit (falling edge to rising edge).

Mode 1: Any character starting with a 10xx bit pattern. In this case, the USART measures the duration of the Start and of the 1st data bit. The measurement is done falling edge to falling edge, to ensure a better accuracy in the case of slow signal slopes.

Prior to activating the auto baud rate detection, the USART_BRR register must be initialized by writing a non-zero baud rate value.

The automatic baud rate detection is activated by setting the ABREN bit in the USART_CR3 register. The USART will then wait for the first character on the RX line. The auto baud rate operation completion is indicated by the setting of the ABRF flag in the USART_SR register. If the line is noisy, the correct baud rate detection cannot be guaranteed. In this case the BRR value may be corrupted and the ABRE error flag

will be set. This also happens if the communication speed is not compatible with the automatic baud rate detection range (bit duration not between 16 and 65536 clock periods).

The RXNE interrupt marks that communication is complete. The auto baud rate detection can be re-launched later by resetting the ABRF flag (by writing a '0').

Attention: The BRR value might be corrupted if the USART is disabled (UE = 0) during an auto baud rate operation.

20.3.6. Multiprocessor communication

It is possible to perform USART multiprocessor communications (with several USARTs connected in a network). For instance one of the USARTs can be the master with its TX output connected to the RX inputs of the other USARTs, while the others are slaves with their respective TX outputs logically ANDed together and connected to the RX input of the master.

In multiprocessor configurations, it is often desirable that only the intended message recipient actively receives the full message contents, thus reducing redundant USART service overhead for all non-addressed receivers.

The non-addressed devices can be placed in Mute mode. When the Mute mode is enabled:

- None of the reception status bits can be set.
- All the receive interrupts are inhibited.
- The RWU bit in USART_CR1 register is set to '1'. RWU can be controlled automatically by hardware or by software.

The USART can enter or exit from Mute mode using one of two methods, depending on the WAKE bit in the USART_CR1 register:

- Idle Line detection if the WAKE bit is reset,
- Address Mark detection if the WAKE bit is set.

20.3.6.1. Idle bus detection (WAKE=0)

The USART enters mute mode when the RWU bit is written to 1. It wakes up when an Idle frame is detected. Then the RWU bit is cleared by hardware, but the IDLE bit is not set in the USART_SR register. RWU can also be written to 0 by software. An example of mute mode behavior using idle line detection is given in the figure below.

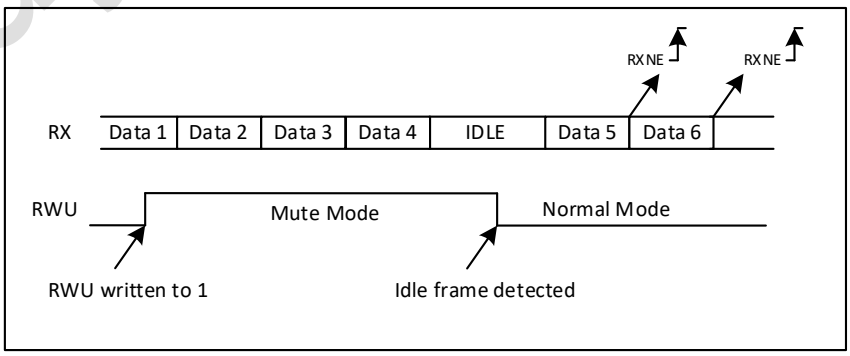

Figure 20-7 Mute mode using Idle line detection

20.3.6.2. Address mark detection (WAKE=1)

In this mode, bytes are recognized as addresses if their MSB is a '1' else they are considered as data. In an address byte, the address of the targeted receiver is put in the 4 LSBs. This 4-bit word is compared by the receiver with its own address which is programmed in the ADD bits in the USART_CR2 register.

The USART enters mute mode when an address character is received which does not match its programmed address. In this case, the RWU bit is set by hardware.

The RXNE flag is not set for this address byte and no interrupt is issued as the USART would have entered mute mode.

It exits from mute mode when an address character is received which matches the programmed address. Then the RWU bit is cleared, and subsequent bytes are received normally. The RXNE bit is set for the address character since the RWU bit has been cleared.

The RWU bit can be written to as 0 or 1 when the receiver buffer contains no data (RXNE=0 in the USART_SR register). Otherwise, the write attempt is ignored. An example of mute mode behavior using idle line detection is given in the figure below.

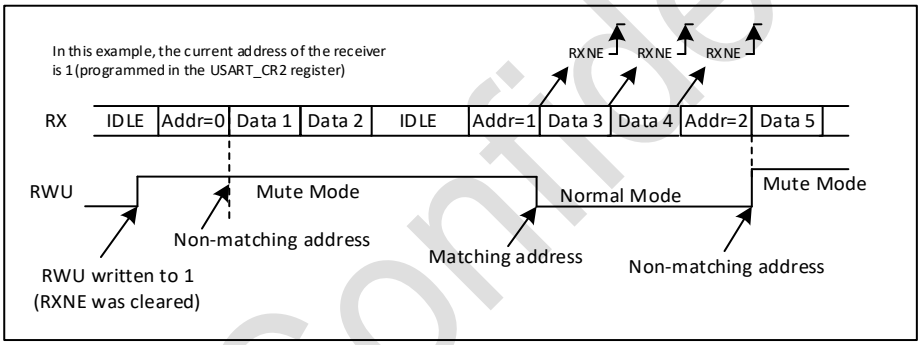

Figure 20-8 Mute mode using address mark detection

20.3.6.3. Parity control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the USART_CR1 register. Depending on the frame length defined by the M bit, the possible USART frame formats are as listed in the table below. SB: Start Bit, STB: Stop Bit, PB: Parity Bit

Table 20-4 Frame formats

M bit	PCE bit	USART frame
0	0	SB—8 bit data—STB
0	1	SB—7 bit data—PB—STB
1	0	SB—9 bit data—STB
1	1	SB—8 bit data—PB—STB

In case of wake up by an address mark, the MSB bit of the data is taken into account and not the parity bit (the MSB is the last one issued in the data bit, followed by the check bit or stop bit).

20.3.6.4. Even parity

The parity bit is calculated to obtain an even number of "1s" inside the frame.

As an example, if data = 00110101 and 4 bits are set, the parity bit is equal to 0 if even parity is selected (PS bit in USART_CR1 = 0).

20.3.6.5. Odd parity

The parity bit is calculated to obtain an odd number of “1s” inside the frame.

As an example, if data = 00110101 and 4 bits set, then the parity bit is equal to 1 if odd parity is selected (PS bit in USART_CR1 = 1).

20.3.6.6. Transmission mode:

If the PCE bit is set in USART_CR1, then the MSB bit of the data written in the data register is transmitted but is changed by the parity bit (even number of “1s” if even parity is selected or an odd number of “1s” if odd parity is selected). If the parity check fails, the PE flag is set in the USART_SR register and an interrupt is generated if PEIE is set in the USART_CR1 register.

20.3.7. USART synchronous mode

The synchronous master mode is selected by programming the CLKEN bit in the USART_CR2 register to ‘1’. In synchronous mode, the following bits must be kept cleared:

- HDSEL bit in the USART_CR3 register

The USART can be used to control bidirectional synchronous serial communications in master mode. The CK pin is the output of the USART transmitter clock. No clock pulses are sent to the CK pin during start bit and stop bit. Depending on the state of the LBCL bit in the USART_CR2 register, clock pulses are, or are not, generated during the last valid data bit. The CPOL bit in the USART_CR2 register isused to select the clock polarity, and the CPHA bit in the USART_CR2 register is used to select the phase of the external clock.

During the Idle state, preamble and send break, the external CK clock is not activated.

In synchronous master mode, the USART transmitter operates exactly like in asynchronous mode. However, since CK is synchronized with TX (according to CPOL and CPHA), the data on TX is synchronous.

In synchronous master mode, the USART receiver operates in a different way compared to asynchronous mode. If RE is set to 1, the data are sampled on CK (rising or falling edge, depending on CPOL and CPHA), without any oversampling. A given setup and a hold time must be respected (which depends on the baud rate: 1/16 bit time).

Attention:

The CK pin operates in conjunction with the TX pin. Thus, the clock is provided only if the transmitter is enabled (TE = 1) and data are being transmitted (USART_TDR data register written). This means that it is not possible to receive synchronous data without transmitting data.

The LBCL, CPOL and CPHA bits have to be selected when both the transmitter and the receiver are disabled to ensure that the clock pulses function correctly. These bits should not be changed while the transmitter or the receiver is enabled.

It is advised that TE and RE are set in the same instruction to minimize the setup and the hold time of the receiver.

The USART supports master mode only: it cannot receive or send data related to an input clock (CK is always an output).

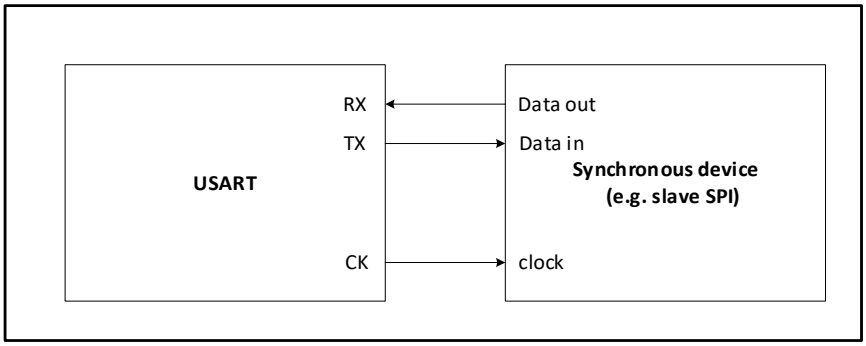

Figure 20-9 USART example of synchronous transmission

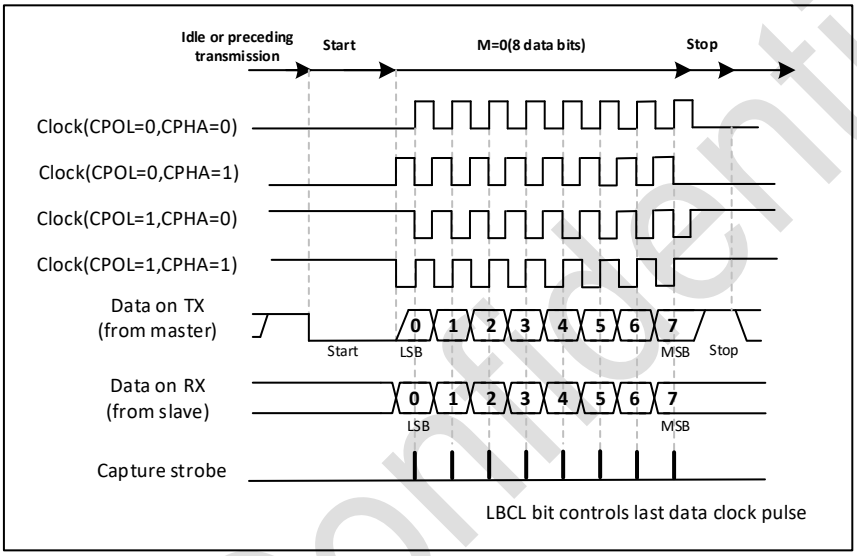

Figure 20-10 USART data clock timing diagram (M=0)

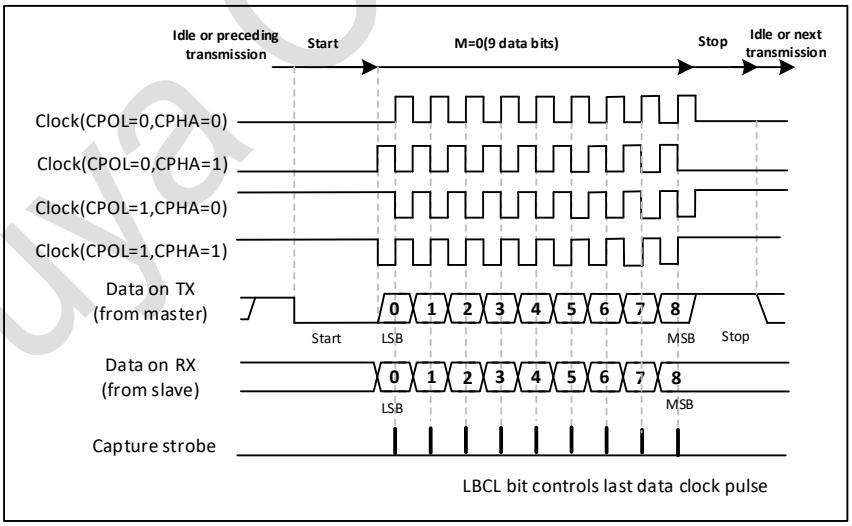

Figure 20-11 USART data clock timing diagram (M=1)

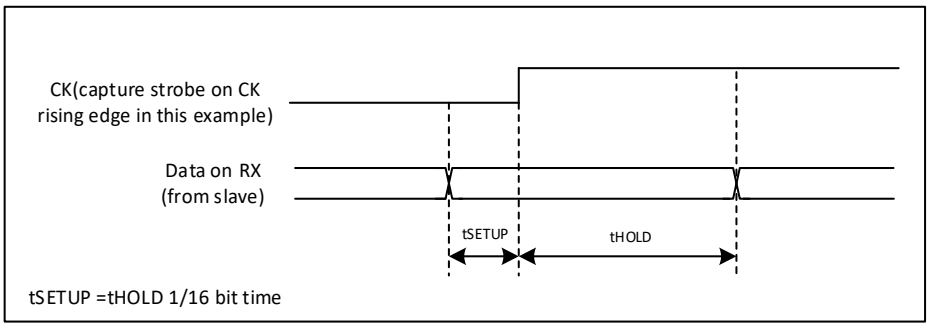

Figure 20-12 RX data setup/hold time

20.3.8. Single-wire Half-duplex communications

The single-wire half-duplex mode is selected by setting the HDSEL bit in the USART_CR3 register. In this mode, the following bits must be kept cleared.

The USART can be configured to follow a Single-wire Half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are connected internally. The selection between half- and Full-duplex communication is made with a control bit HDSEL in USART_CR3. The TX pin is always released when no data is transmitted. Thus, it acts as a standard I/O in idle or in reception. It means that the I/O must be configured so that TX is configured as alternate function open-drain with an external pull-up.

Apart from this, the communication protocol is similar to normal USART mode. Any conflict on the line must be managed by software. The transmission is never blocked by hardware and continues as soon as data are written in the data register while the TE bit is set.

20.3.9. Hardware flow control

It is possible to control the serial data flow between 2 devices by using the nCTS input and the nRTS output. The figure below shows how to connect two devices in this mode.

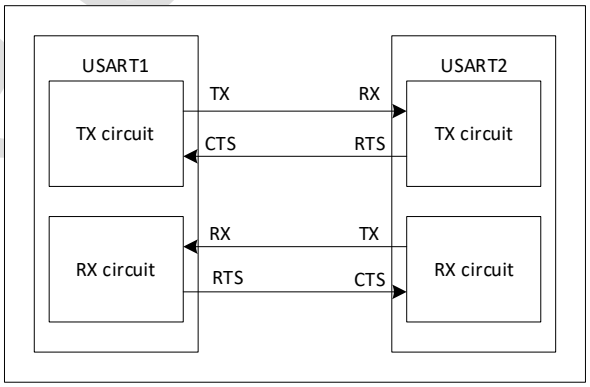

Figure 20-13 Hardware flow control between two USARTs

20.3.9.1. RTS flow control

If the RTS flow control is enabled (RTSE=1), then nRTS is asserted (tied low) as long as the USART receiver is ready to receive a new data. When the receive register is full, nRTS is deasserted, indicating that the transmission is expected to stop at the end of the current frame.

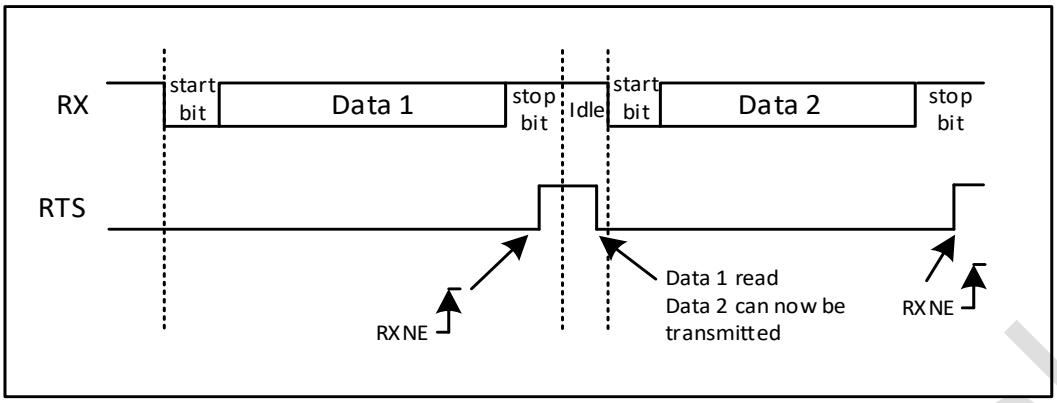

Figure 20-14 RTS flow control

20.3.9.2. CTS flow control

If the CTS flow control is enabled (CTSE=1), then the transmitter checks the nCTS input before transmitting the next frame. If nCTS is asserted (tied low), then the next data is transmitted. When CTS is deasserted during a transmission (high level), the current transmission is completed before the transmitter stops.

When CTSE = 1, the CTSIF status bit is automatically set by hardware as soon as the nCTS input toggles. It indicates when the receiver becomes ready or not ready for communication. An interrupt is generated if the CTSIE bit in the USART_CR3 register is set.

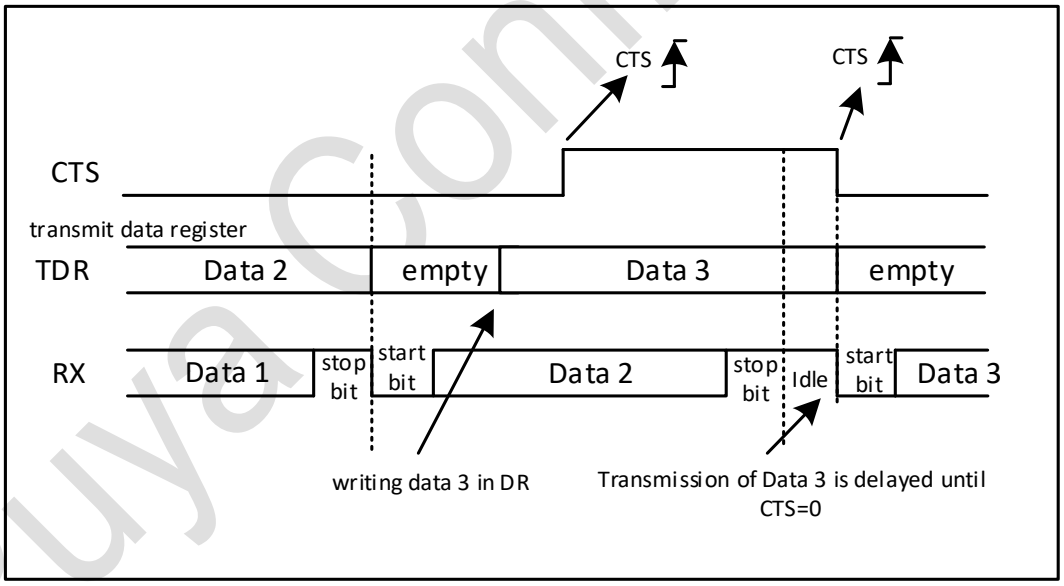

Figure 20-15 CTS flow control

20.4. USART interrupt requests

No.	Interrupt event	Event flag	Enable Control bit	Transmission/Reception
1	Transmit data register empty	TXE	TXEIE	Transmission
2	CTS interrupt	CTSIF	CTSIE	Transmission
3	Transmission is complete	TC	TCIE	Transmission
4	Receive data register not empty (data ready to be read)	RXNE	RXNEIE	Reception
5	Overrun error detected	ORE		Reception
6	Idle line detected	IDLE	IDLEIE	Reception
7	Parity error	PE	PEIE	Reception
8	Noise error/Overrun error/Framing Error in multibuffer communication	NR/ORE/FE	EIE	Reception

The USART interrupt events are connected to the same interrupt vector.

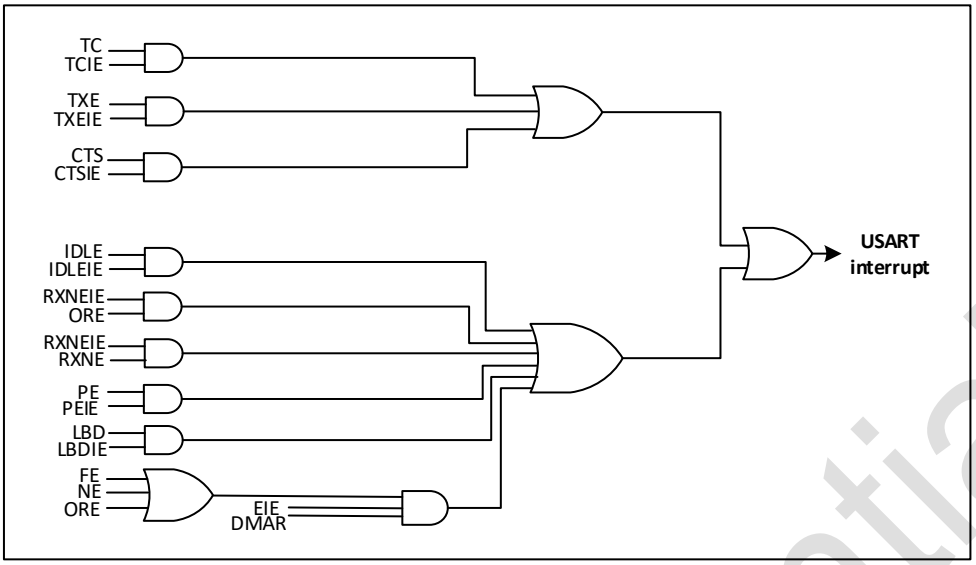

Figure 20-16 USART interrupt mapping diagram

20.5. USART register

20.5.1. USART status register (USART_SR)

Address offset: 0x00

Reset value: 0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	ABRRQ	ABRE	ABRF	CTS	Res	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
-	-	-	W	R	R	RC_W0	-	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset value	Function
31:13	Reserved	-	-	-
12	ABRRQ	W	0	Auto baud rate request Writing 1 to this bit resets the ABRF flag and requests an automatic baud rate measurement on the next received data frame.
11	ABRE	R	0	Auto baud rate error This bit is set by hardware if the baud rate measurement failed (baud rate out of range or character comparison failed). It is cleared by software, by writing 1 to the ABRRQ bit.
10	ABRF	R	0	Auto baud rate detection flag. This bit is set by hardware when the automatic baud rate has been set (RXNE is also set, generating an interrupt if RXNEIE = 1) or when the auto baud rate operation was completed without success (ABRE = 1) (ABRE, RXNE and FE are also set in this case). It is cleared by software, by writing 1 to the ABRRQ in the USART_SR register.
9	CTS	RC_W0	0	CTS flag If the CTSE bit is set, it is set high by the hardware when the nCTS input changes state. 0: CTS input unchanged 1: CTS input changed
8	Reserved	-	-	-
7	TXE	R	1	Transmit data register empty This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TXEIE bit =1. It is cleared by a write to the USART_DR register.

Bit	Name	R/W	Reset value	Function
				0: Data is not transferred to the shift register 1: Data is transferred to the shift register
6	TC	RC_W0	1	Transmission complete This bit is set by hardware if the transmission of a frame containing data is complete and if TXE is set. An interrupt is generated if TCIE=1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). The TC bit can also be cleared by writing a '0' to it. 0: Transmission not complete 1: Transmission is complete
5	RXNE	RC_W0	0	Read data register not empty This bit is set by hardware when the content of the shift register has been transferred to the USART_DR register. It is cleared by a read to the USART_DR register. The RXNE flag can also be cleared by writing a zero to it. An interrupt is generated if the RXNEIE bit =1. 0: Data is not received 1: Received data is ready to be read
4	IDLE	R	0	DLE line detected This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the IDLEIE bit =1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No Idle line is detected 1: Idle line is detected
3	ORE	R	0	Overrun error This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RXNE=1. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). An interrupt is generated if the RXNEIE bit =1. 0: No Overrun error 1: Overrun error is detected Note: When this bit is set, the RDR register content will not be lost, but the shift register will be overwritten. An interrupt is generated on ORE flag if the EIE bit is set.
2	NE	R	0	Noise error flag This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No noise is detected 1: Noise is detected Note: This bit does not generate interrupt as it appears at the same time as the RXNE bit which itself generates an interrupting. Interrupt is generated on NE flag in case of Multi-Buffer communication if the EIE bit is set.
1	FE	R	0	Framing error This bit is set by hardware when a de-synchronization or excessive noise is detected. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). 0: No frame error detected 1: Framing error is detected Note: This bit does not generate interrupt as it appears at the same time as the RXNE bit which itself generates an interrupting. If the word currently being transferred causes both frame error and overrun error, it will be transferred, and only the ORE bit will be set. Interrupt is generated on FE flag in case of MultiBuffer communication if the EIE bit is set.
0	PE	R	0	Parity error

Bit	Name	R/W	Reset value	Function
				This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read from the USART_SR register followed by a write to the USART_DR register). The software must wait for the RXNE flag to be set before clearing the PE bit. An interrupt is generated if the PEIE bit =1. 0: No parity error 1: Parity error

20.5.2. USART data register (USART_DR)

Address offset: 0x04

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:9	Reserved	-	-	-
8:0	DR[8:0]	RW	9'h0	Data value Contains the Received or Transmitted data character, depending on whether it is read from or written to. The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR). The TDR register provides the parallel interface between the internal bus and the output shift register. The RDR register provides the parallel interface between the input shift register and the internal bus. When transmitting with the parity enabled, the value written in the MSB (bit 7 or bit 8 depending on the data length) has no effect because it is replaced by the parity. When receiving with the parity enabled, the value read in the MSB bit is the received parity bit.

20.5.3. Baud rate register (USART_BRR)

Address offset: 0x08

Reset value: 0x0000 0000

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

In automatic baud rate detection mode, the hardware updates this register.

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	-
15:4	DIV_Mantissa[15:4]	RW	12'h0	12-bit integer
3:0	DIV_Faction[3:0]	RW	4'h0	4 decimal places

20.5.4. USART control register 1 (USART_CR1)

Address offset: 0x0C

Reset value: 0x0000 0000

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

Res	Res	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	Res
-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

Bit	Name	R/W	Reset value	Function
31:14	Reserved	-	-	-
13	UE	RW	0	USART enable When this bit is cleared the USART prescalers are stopped. This bit is set and cleared by software. 0: USART prescaler and outputs disabled 1: USART enabled The software needs to wait for USART_ISR.TC to be set before it can clear the UE bit and enter the low power mode.
12	M	RW	0	0: 1 Start bit, 8 Data bits 1: 1 Start bit, 9 Data bits
11	WAKE	RW	0	Receive wakeup mode. Wakeup method Set and cleared by software. 0: Idle Line 1: Address Mark
10	PCE	RW	0	Parity control enable 0: Parity control disabled 1: Parity control enabled
9	PS	RW	0	Parity selection Set and cleared by software. 0: Even parity 1: odd parity
8	PEIE	RW	0	PE interrupt enable Set and cleared by software. 0: Disabled 1:PE interrupt enabled
7	TXEIE	RW	0	TXE interrupt enable Set and cleared by software. 0: Disabled 1:TXE interrupt enabled
6	TCIE	RW	0	Transmission complete interrupt enable Set and cleared by software. 0: Disabled 1:TC interrupt enabled
5	RXNEIE	RW	0	RXNE interrupt enable. Set and cleared by software. 0: Disabled 1:ORE or RXNE interrupt enabled
4	IDLEIE	RW	0	IDLE interrupt enable Set and cleared by software. 0: Disabled 1:IDLE interrupt enabled
3	TE	RW	0	Transmitter enable 0: Transmitter is disabled 1: Transmitter is enabled
2	RE	RW	0	Receiver enable 0: Receiver is disabled 1: Receiver is enabled and begins searching for a start bit
1	RWU	RW	0	Receiver wakeup This bit indicates if the USART is in mute mode. It is cleared when a wakeup sequence is recognized. The specific wake-up sequence (address or idle bus) is controlled by the register USART_CR1. WAKE bit. 0: Receiver in active mode 1: Receiver in mute mode Note: 1: Before selecting Mute mode, the USART must first receive a data byte, otherwise it cannot function in Mute mode with wakeup by Idle line detection. Note: 2: In Address Mark Detection wakeup configuration (WAKE bit=1) the RWU bit cannot be modified by software while the RXNE bit is set.
0	Reserved	-	-	-

20.5.5. USART control register 2 (USART_CR2)

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset value	Function
31:14	Reserved	-	-	-
13	STOP	RW	0	Stop bits 0: 1 Stop bit 1: 2 Stop bits
12	Reserved	-	-	-
11	CLKEN	RW	0	Clock enable 0: Disabled 1: CK pin enabled This bit is reserved when synchronous mode is not supported.
10	CPOL	RW	0	Clock polarity This bit allows the user to select the polarity of the clock output on the CK pin in synchronous mode. 0: Steady low value on CK pin when the bus is idle. 1: Steady high value on CK pin when the bus is idle.
9	CPHA	RW	0	This bit allows the user to select the phase of the clock output on the CK pin in synchronous mode. It works in conjunction with the CPOL bit to produce the desired clock/data relationship. 0: Data capture at first edge of clock 1: Data capture at second edge of clock
8	LBCL	RW	0	Whether the clock pulse of the last bit of data is output at CK. 0: The clock pulse of the last data bit is not output to the CK pin 1: The clock pulse of the last data bit is output to the CK pin
7:4	Reserved	-	-	-
3:0	ADD[3:0]	RW	4'h0	Address of the USART node This is used in multiprocessor communication during mute mode, for a 4-bit wake up with address mark detection.

20.5.6. USART control register 3 (USART_CR3)

Address offset: 0x14

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		ABRMOD	ABREN	OVER8	CTSIE	CTSE	RTSE	Res	Res	Res	Res	HDSEL	Res	Res	EIE
-		RW	RW	RW	RW	RW	RW	-	-	-	-	RW	-	-	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	-
13	ABRMOD	RW	0	Auto baud rate mode 0: Measurement of the start bit is used to detect the baud rate 1: Falling edge to falling edge measurement This bitfield can only be written when ABREN = 0 or the USART is disabled (UE=0).
12	ABREN	RW	0	Auto baud rate enable 0: Disabled 1: Auto baud rate detection is enabled
11	OVER8	RW	0	Oversampling mode 0: Oversampling by 16 1: Oversampling by 8 This bit is only writeable when UE = 0.
10	CTSIE	RW	0	CTS interrupt enable 0: Interrupt is inhibited 1: CTSIF interrupt enable.
9	CTSE	RW	0	CTS enable 0: CTS hardware flow control disabled

Bit	Name	R/W	Reset Value	Function
				1: CTS mode enabled Data is only transmitted when the CTS input is asserted (tied to 0). If a data is written into the data register while CTS is deasserted, the transmission is postponed until CTS is asserted.
8	RTSE	RW	0	RTS enable 0: RTS hardware flow control disabled 1: RTS interrupt enabled, data is only requested when there is space in the receive buffer. The transmission of data is expected to cease after the current character has been transmitted. The RTS output is asserted (tied to 0) when a data can be received.
7:4	Reserved	-	-	-
3	HDSEL	RW	0	Half-duplex selection 0: Half duplex mode is not selected 1: Half duplex mode is selected
2:1	Reserved	-	-	-
0	EIE	RW	0	Error interrupt enable 0: Interrupt is inhibited 1: Frame error FE, over normal operation error ORE, noise NE interrupt enable.

21. Serial peripheral interface (SPI)

21.1. 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.

21.2. SPI main features

- Master or slave operation
- 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 transport frame selection
- Support multi-master mode
- 8 master mode baud rate prescalers up to 12 M
- Slave mode frequency up to 6 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
- Motorola mode
- Primary mode failure, overload that can cause interruption
- Two 32-bit embedded Rx and Tx FIFOs

21.3. SPI functional description

21.3.1. Overview

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.

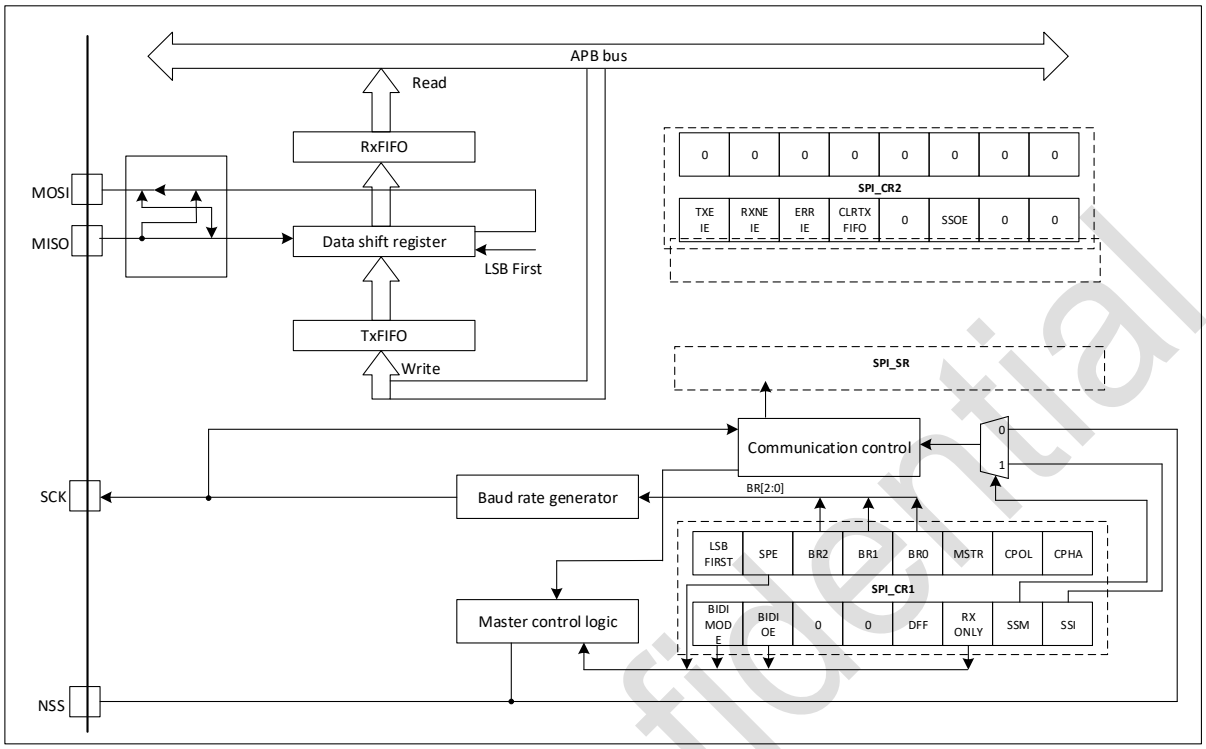

Figure 21-1 SPI block diagram

21.3.2. Communications between one master and one slave

The SPI allows the MCU to communicate using different configurations, depending on the device targeted and the application requirements. These configurations use 2 or 3 wires (with software NSS management) or 4 wires (with hardware NSS management).

21.3.2.1. Full-duplex communication

By default, the SPI is configured for full-duplex communication. In this configuration, the shift registers of the master and slave are linked using two unidirectional lines between the MOSI and the MISO pins. During SPI communication, data is shifted synchronously on the SCK clock edges provided by the master. The master transmits the data to be sent to the slave via the MOSI line and receives data from the slave via the MISO line. When the data frame transfer is complete, the information between the master and slave is exchanged.

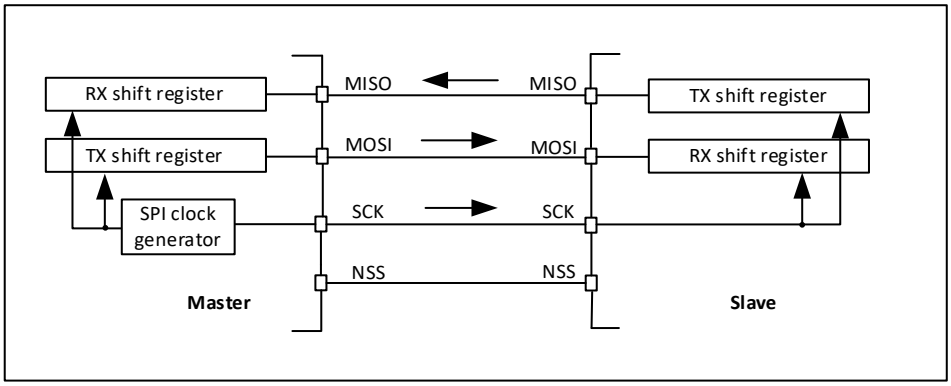

Figure 21-2 Full-duplex single master/ single slave application

21.3.2.2. 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 master′ s MISO pin and the slave′ s MOSI pin are free for other application uses and act as GPIOs.

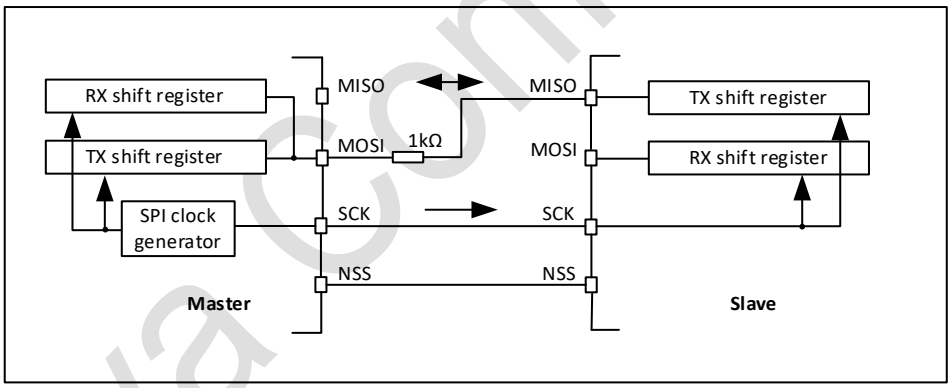

Figure 21-3 Half-duplex single master/ single slave application

The NSS pins can be used to provide a hardware control flow between master and slave. NSS simplex communication can also be controlled using software.

The SPI can communicate in simplex mode by setting the SPI in transmit-only or in receive-only using the RXONLY bit in the SPI_CR1 register. In this configuration, only one line is used for the transfer between the shift registers of the master and slave. 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 slave configuration, the MISO output is disabled and the pin can be used as a GPIO.

The slave continues to receive data from the MOSI pin while its slave select signal is active. Received data events appear depending on the data buffer configuration. In the master configuration, the MOSI output is disabled and the pin can be used as a GPIO. The clock signal is generated continuously as long as the SPI is enabled. The only way to stop the clock is to clear the RXONLY bit or the SPE bit and wait until the incoming pattern from the MISO pin is finished.

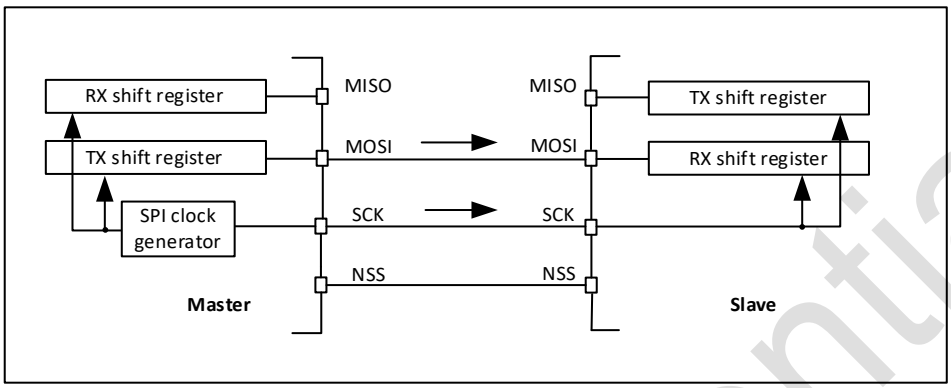

Figure 21-4 Single master/ single slave application
(master in transmit-only/slave in receive-only mode)

(1) The NSS pins can be used to provide a hardware control flow between master and slave. Then the flow has to be handled internally.

(2) An accidental input information is captured at the input of transmitter Rx shift register. All the events associated with the transmitter receive flow must be ignored in standard transmit only mode.

(3) In this configuration, both the MISO pins can be used as GPIOs.

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).

21.3.3. Multi-slave communication

In a configuration with two or more independent slaves, the master uses GPIO pins to manage the chip select lines 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.

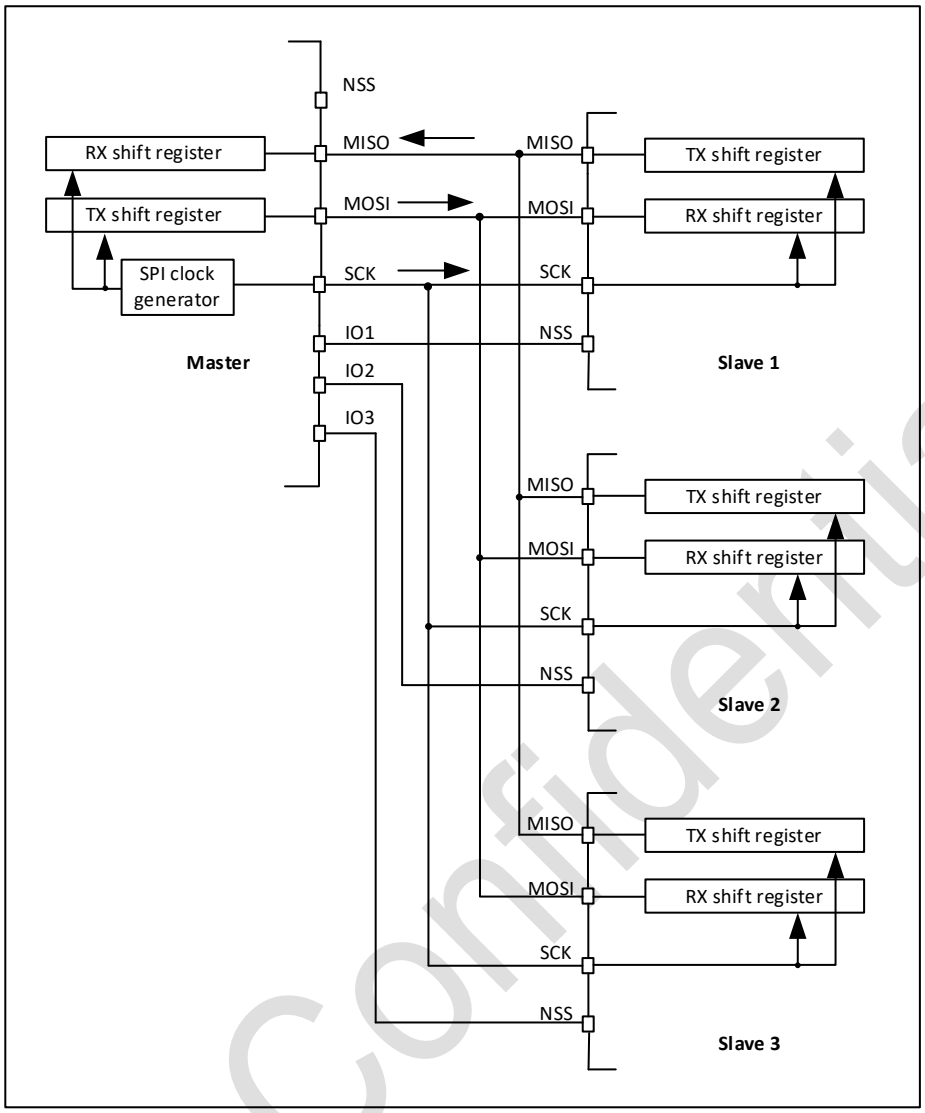

Figure 21-5 Master and three independent slaves

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.

21.3.4. Multi-master communication

Unless SPI bus is not designed for a multi-master capability primarily, the user can use build in feature which detects a potential conflict between two nodes trying to master the bus at the same time. For this detection, NSS pin is used configured at hardware input mode.

The connection of more than two SPI nodes working at this mode is impossible as only one node can apply its output on a common data line at time.

When nodes are non active, both stay at slave mode by default. Once one node wants to overtake control on the bus, it switches itself into master mode and applies active level on the slave select input of the other

node via dedicated GPIO pin. After the session is completed, the active slave select signal is released and the node mastering the bus temporary returns back to passive slave mode waiting for next session start.

If potentially both nodes raised their mastering request at the same time a bus conflict event appears (see mode fault MODF event). Then the user can apply some simple arbitration process.

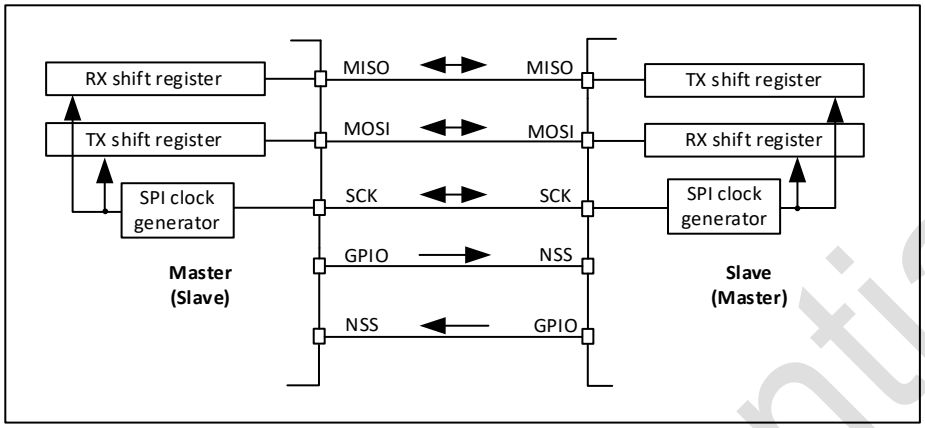

Figure 21-6 Multi-master application

21.3.5. 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.

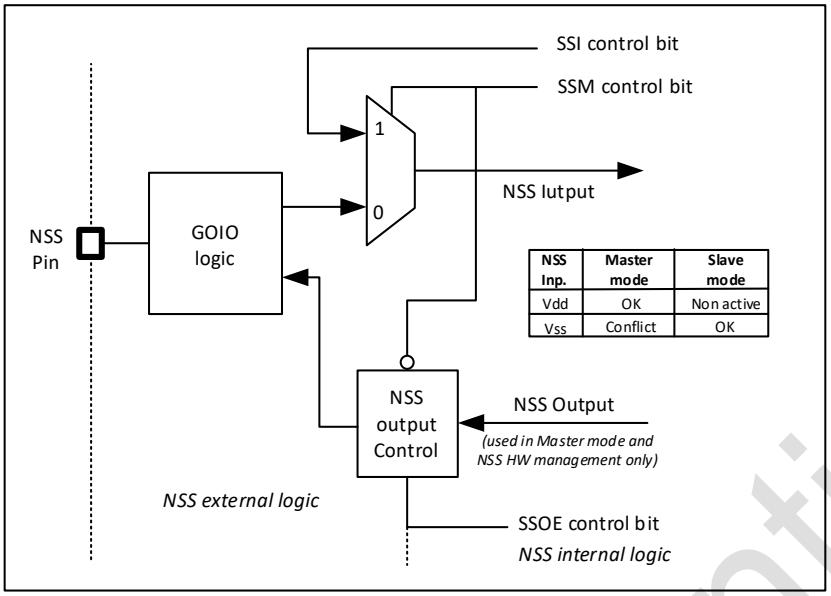

Figure 21-7 Hardware/software slave select management

21.3.6. Communication formats

During SPI communication, receive and transmit operations are performed simultaneously The serial clock (SCK) synchronizes the shifting and sampling of the information on the data lines. The communication format depends on the clock phase, the clock polarity and the data frame format. To be able to communicate together, the master and slaves devices must follow the same communication format.

21.3.6.1. 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. The CPOL (clock polarity) bit controls the idle state value of the clock when no data is being transferred. 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.

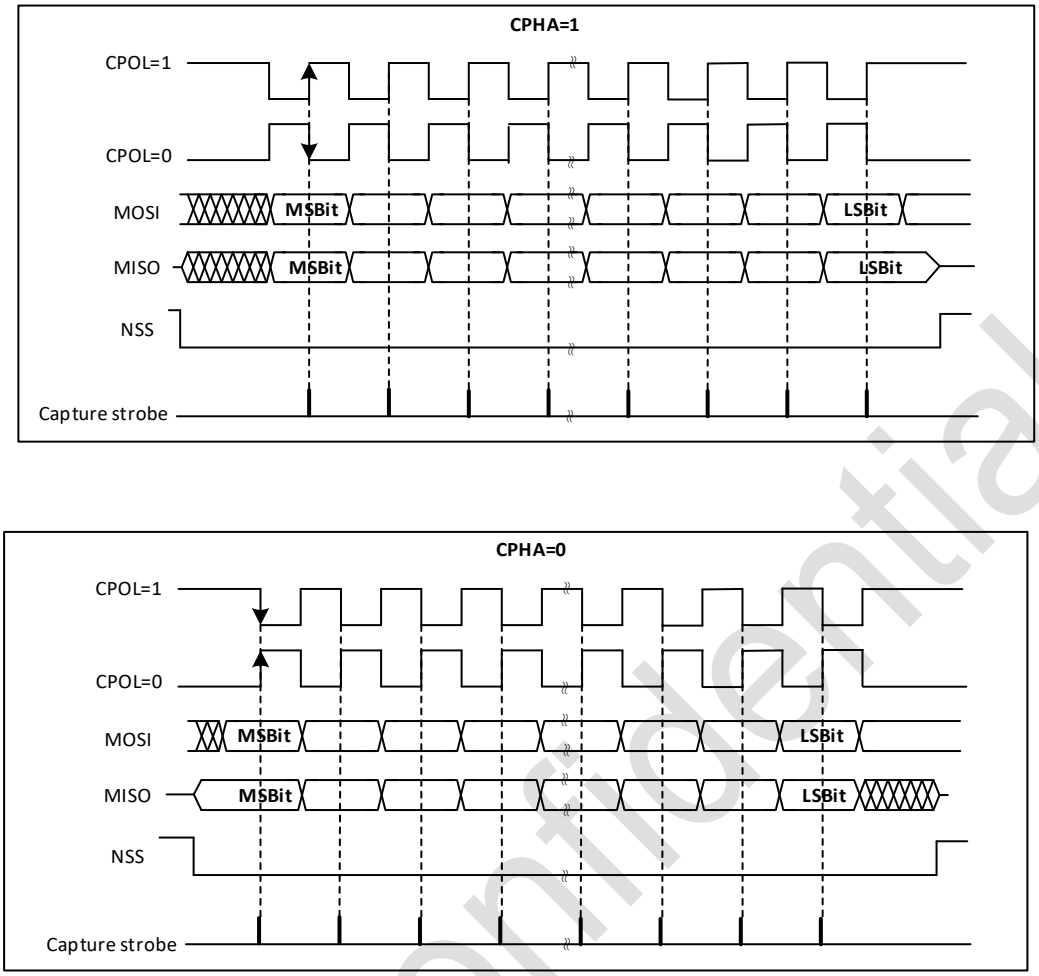

Figure 21-8 Data clock timing diagram

The order of data bits depends on LSBFIRST bit setting.

21.3.6.2. 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.

21.3.7. Configuration of SPI

The configuration procedure is almost the same for master and slave. When a standard communication is to be initialized, perform these steps:

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:
   - Configure the serial clock baud rate using the BR[2:0] bits (not required in slave mode)
   - Configure the CPOL and CPHA bits
   - Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE
   - Configure the LSBFIRST bit
   - Configure SSM and SSI
   - 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:
   - Configure the DS bit to select the data length for the transfer
   - Configure SSOE (not required for slave mode)

21.3.8. Procedure for enabling SPI

It is recommended to enable the SPI slave before the master sends the clock. If not, undesired data transmission might occur. The data register of the slave must already contain data to be sent before starting communication with the master. The SCK signal must be settled at an idle state level corresponding to the selected polarity before the SPI slave is enabled.

The master at full-duplex (or in any transmit-only mode) starts to communicate when the SPI is enabled and TXFIFO is not empty, or with the next write to TXFIFO.

In any master receive only mode (RXONLY=1 or BIDIMODE=1 & BIDIOE=0), master starts to communicate, and the clock starts running immediately after SPI is enabled.

21.3.9. Data transmission and reception procedures

21.3.9.1. RXFIFO and TXFIFO

All SPI data transactions pass through the 32-bit embedded FIFOs. This enables the SPI to work in a continuous flow, and prevents overruns when the data frame size is short. Each direction has its own FIFO called TXFIFO and RXFIFO. These FIFOs are used in all SPI modes.

The handling of FIFOs depends on the data exchange mode (duplex, simplex) and data frame format.

A read access to the SPI_SR register returns the oldest value stored in RXFIFO that has not been read yet. A write access to the SPI_DR stores the written data in the TXFIFO at the end of a send queue. The read access must be always aligned with the RXFIFO threshold. The FTLVL and FRLVL bits show the current occupancy levels of the two FIFOs.

A read access to the SPI_DR register must be managed by the RXNE event. This event is triggered when data is stored in RXFIFO and the threshold is reached. When RXNE is cleared, RXFIFO is considered to be empty.

In a similar way, write access of a data frame to be transmitted is managed by the TXE event. This event is triggered when the TXFIFO level is less than or equal to half of its capacity. Otherwise, TXE is cleared and the TXFIFO is considered as full.

In this way, RXFIFO can store up to 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.

21.3.9.2. Sequence handling

A few data frames can be passed at single sequence to complete a message. When transmission is enabled, a sequence begins and continues while any data is present in the TXFIFO of the master. The clock signal is provided continuously by the master until TXFIFO becomes empty, then it stops waiting for additional data.

In receive-only modes, half-duplex (BIDIMODE=1, BIDIOE=0) or simplex (BIDIMODE=0,RXONLY=1) the master starts the sequence immediately when both SPI is enabled and receive-only mode is activated. The clock signal is provided by the master, and it does not stop until either SPI or receive-only mode is disabled by the master. The master receives data frames continuously up to this moment

While the master can provide all the transactions in continuous mode (SCK signal is continuous) it has to respect slave capability to handle data flow and its content at anytime. When necessary, the master must slow down the communication and provide either a slower clock or separate frames or data sessions with sufficient delays. Be aware there is no underflow error signal for master or slave in SPI mode, and data from the slave is always transacted and processed by the master even if the slave could not prepare it correctly intime.

Each sequence must be encased by the NSS pulse in parallel with the multi-slave system to select just one of the slaves for communication. In a single slave system it is not necessary to control the slave with NSS, but it is often better to provide the pulse here too, to synchronize the slave with the beginning of each data sequence. NSS can be managed by both software and hardware.

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.

21.3.9.3. Procedure for disabling the SPI

When SPI is disabled, it is mandatory to follow the disable procedures described in this paragraph. It is important to do this before the system enters a low-power mode when the peripheral clock is stopped. Ongoing transactions can be corrupted in this case. In some modes the disable procedure is the only way to stop continuous communication running.

Master in full-duplex or transmit only mode can finish any transaction when it stops providing data for transmission. In this case, the clock stops after the last data transaction. Before the SPI is disabled in these modes, the user must follow standard disable procedure. When the SPI is disabled at the master transmitter while a frame transaction is ongoing or next data frame is stored in TXFIFO, the SPI behavior is not guaranteed.

When the master is in any receive only mode, the only way to stop the continuous clock is to disable the peripheral by SPE=0. Specific procedure must be followed when disabling SPI in this mode.

Data received but not read remains stored in RXFIFO when the SPI is disabled, and must be processed the next time the SPI is enabled, before starting a new sequence. To prevent having unread data, ensure that RXFIFO is empty when disabling the SPI.

Standard disable procedure is based on pulling BSY status together with FTLVL to check if a transmission session is fully completed. This check can be done in specific cases, too, when it is necessary to identify the end of ongoing transactions, for example:

- When NSS signal is managed by software and master has to provide proper end of NSS pulse for slave.
- When transactions' streams from FIFO are completed while the last data frame is still ongoing in the peripheral bus.

The correct disable procedure is (except when receive only mode is used):

1. Wait until FTLVL[1:0] = 00 (no more data to transmit)
2. Wait until BSY=0 (the last data frame is processed)
3. Disable the SPI (SPE=0).
4. Read data until FRLVL = 00 (read all the received data)

The correct disable procedure for certain receive only modes is:

1. Interrupt the receive flow by disabling SPI (SPE=0) in the specific time window while the last data frame is ongoing.
2. Wait until BSY=0 (the last data frame is processed).
3. Read data until FRLVL = 00 (read all the received data)

21.3.9.4. Data packing

When the data frame size fits into 8 bits, data packing is used automatically when any read or write 16-bit access is performed on the SPI_DR register. The double data frame pattern is handled in parallel in this case.

The figure below provides an example of data packing mode sequence handling. Two data frames are sent after the single 16-bit access the SPI_DR register of the transmitter. This sequence can generate just one RXNE event in the receiver if the RXFIFO threshold is set to 16 bits. The receiver then has to access both data frames by a single 16-bit read of SPI_DR as a response to this single RXNE event. The RxFIFO threshold setting and the following read access must be always kept aligned at the receiver side, as data can be lost if it is not in line.

On the transmitter side, writing the last data frame of any odd sequence with an 8-bit access to SPI_DR is enough. The receiver has to change the Rx_FIFO threshold level for the last data frame received in the odd sequence of frames in order to generate the RXNE event.

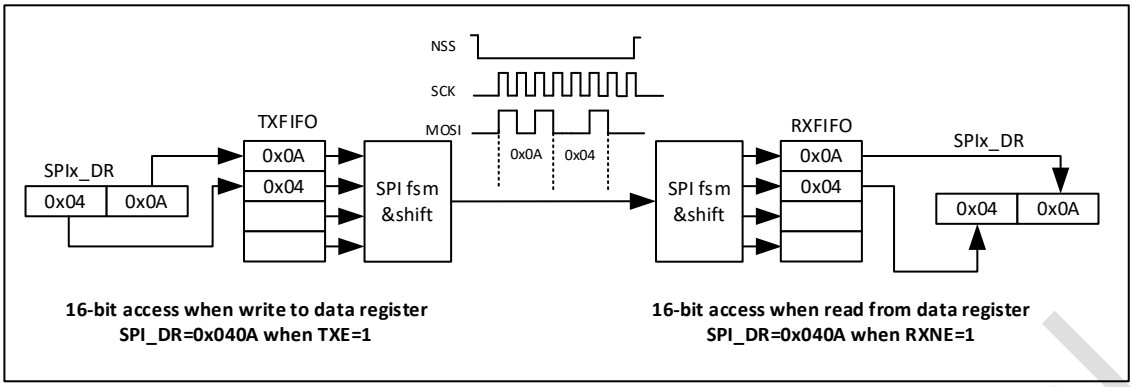

Figure 21-9 Packing data in FIFO for transmission and reception

21.3.9.5. Communication diagrams

Some typical timing schemes are explained in this section. These schemes are valid no matter if the SPI events are handled by polling, interrupts. For simplicity, the LSBFIRST=0, CPOL=0 and CPHA=1 setting is used as a common assumption here.

1. The slave starts to control MISO line as NSS is active and SPI is enabled, and is disconnected from the line when one of them is released. Sufficient time must be provided for the slave to prepare data dedicated to the master in advance before its transaction starts. At the master, the SPI peripheral takes control at MOSI and SCK signals (occasionally at NSS signal as well) only if SPI is enabled. If SPI is disabled the SPI peripheral is disconnected from GPIO logic, so the levels at these lines depends on GPIO setting exclusively.
2. At the master, BSY stays active between frames if the communication is continuous. At the slave, BSY signal always goes down for at least one clock cycle between data frames.
3. The TXE signal is cleared only if TXFIFO is full.
4. The TXE interrupt is generated just after the TXEIE is set. As the TXE signal is at an active level, data transfers to TxFIFO start, until TxFIFO becomes full.
5. In data packed mode, TxE and RxNE events are paired and each read/write access to the FIFO is 16 bits wide until the number of data frames are even.

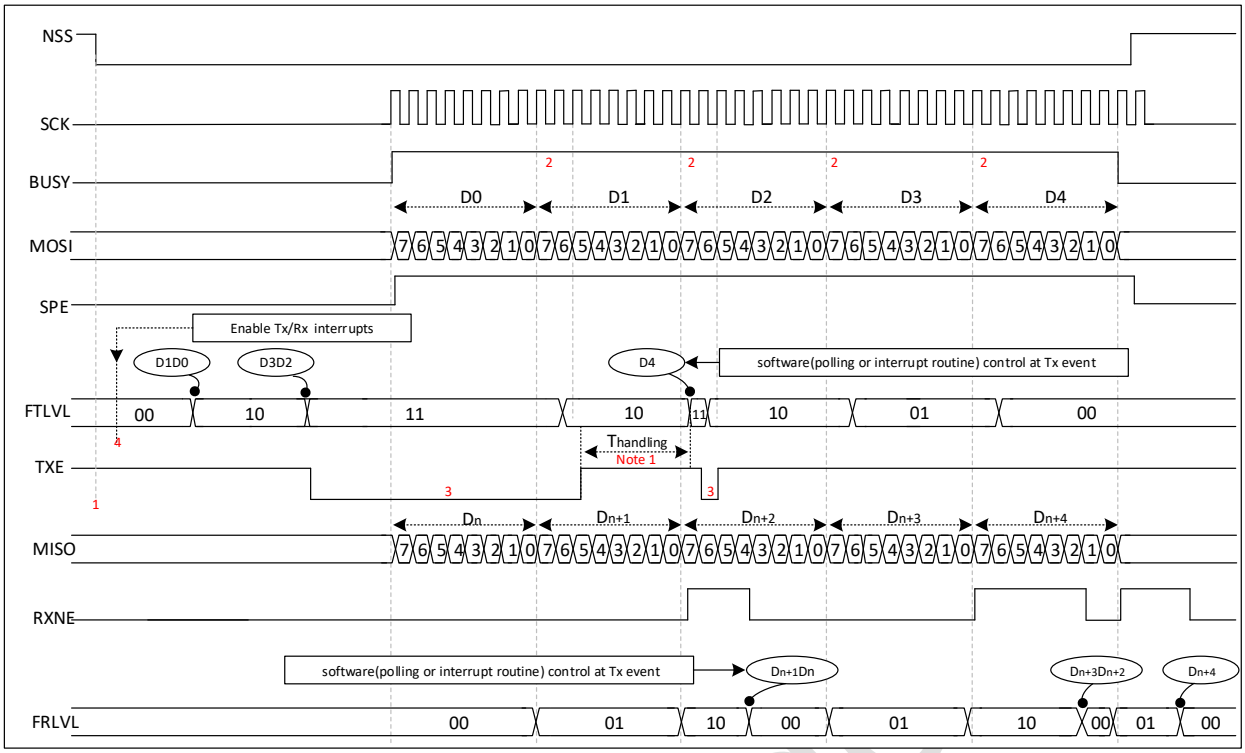

Figure 21-10 Master full-duplex communication (bit frame=8)

21.3.10. SPI status flags

The application can fully monitor the status of the SPI bus through 3 status flags.

21.3.10.1. Tx FIFO empty flag (TXE)

The TXE flag is set when transmission TXFIFO has enough space to store data to send. TXE flag is linked to the TXFIFO level. The flag goes high and stays high until the TXFIFO level is lower or equal to 1/2 of the FIFO depth. An interrupt can be generated if the TXEIE bit in the SPI_CR2 register is set. The bit is cleared automatically when the TXFIFO level becomes greater than 1/2.

21.3.10.2. Rx buffer not empty (RXNE)

The RXNE flag is set depending on the FRXTH bit value in the SPI_CR2 register.

21.3.10.3. 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 disabled
- When a fault is detected in Master mode (MODF bit set to 1)
- In Master mode, when it finishes a data transmission and no new data is ready to be sent
- In Slave mode, when the BSY flag is set to '0' for at least one SPI clock cycle between each data transfer.

Note: Do not use the BSY flag for each data transmission and reception. It is more appropriate to use TXE and RXNE.

21.3.11. Error flags

21.3.11.1. 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:

- Make a read or write access to the SPI_SR register while the MODF bit is set.
- 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.

21.3.11.2. Overflow pattern

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 an overrun condition occurs, the newly received value does not overwrite the previous one in the RXFIFO. The newly received value is discarded, and all data transmitted subsequently is lost.

Clearing the OVR bit is done by a read access to the SPI_DR register followed by a read access to the SPI_SR register.

21.3.12. SPI interrupts

Table 21-1 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

21.4. 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.

21.4.1. 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
BIDIMODE	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	-	-	-
11	DFF	RW	0	Data frame format 0: 8-bit data frame format is selected for transmission / reception 1: 16-bit data frame format is selected for transmission /reception Note: This bit should be written only when SPI is disabled (SPE = '0') for correct operation.
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.
6	SPE	RW	0	SPI enable 0: Peripheral disabled 1: Peripheral enabled

Bit	Name	R/W	Reset value	Function
5:3	BR[2:0]	RW	3'b000	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: This bit should not be changed when communication is ongoing. Note: In slave mode, the fastest baud rate only supports f_{PCLK}/8.
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.

21.4.2. 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								TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	Res	
-								RW	RW	RW	RW	-	RW	-	

Bit	Name	R/W	Reset value	Function
31:8	Reserved	-	-	-
7	TXEIE	RW	0	Tx buffer empty interrupt enable 0: TXE interrupt masked 1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set.
6	RXNEIE	RW	0	R_X buffer not empty interrupt enable 0: RXNE interrupt masked 1: RXNE interrupt not masked. Used to generate an interrupt request when the RXNE flag is set.
5	ERRIE	RW	0	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled When CRCERR, OVR, or MODF is 1, an interrupt request is generated.
4	CLRTXFIFO	RW	0	Clear TXFIFO Set by software and reset by hardware 0: No action 1: Clear TXFIFO Note: This bit should be written only when SPI is disabled (SPE = '0') for valid operation.
3	Reserved	-	-	-
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	-	-	-

21.4.3. SPI status register (SPI_SR)

Address offset: 0x08

Reset value: 0x0000 0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				FTLVL	Res	FRLVL	Res	BSY	OVR	MODF	Res	Res	Res	TXE	RXNE
-				R	-	R	-	R	R	R	-			R	R

Bit	Name	R/W	Reset value	Function
31:12	Reserved	-	-	-
11	FTLVL	R	0	FIFO transmission level These bits are 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	-	-	-
9	FRLVL	R	0	FIFO reception level These bits are set and cleared by hardware. 0: FIFO empty 1: FIFO is full Note: These bits are not used in I²S mode and in SPI receive-only mode while CRC calculation is enabled.
8	Reserved	-	-	-
7	BSY	R	0	Busy flag 0: SPI not busy 1: SPI is busy in communication or Tx buffer is not empty
6	OVR	R	0	Overrun flag 0: No overflow error 1: Overrun occurred This flag is set by hardware and reset by a software sequence.
5	MODF	R	0	Mode fault 0: No mode fault occurred 1: Mode fault occurred This flag is set by hardware and reset by a software sequence.
4:2	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 1: Rx buffer not empty 0: Rx buffer empty

21.4.4. 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	16'h0	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 (DS in SPI_CR1 register), the data sent or received is either 8-bit or 16-bit. For an 8-bit data frame, the buffers are 8-bit and only the LSB of the register is used for transmission/reception. When in reception mode, the MSB of the register (SPI_DR[15:8]) is forced to 0. For a 16-bit data frame, the buffers are 16-bit and the entire register, SPI_DR[15:0] is used for transmission/reception.

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 extensions allow the core to be stopped either on a given instruction fetch (breakpoint) or data access (watchpoint). When stopped, the core's internal state and the system's external state may be examined. Once examination is complete, the core and the system may be restored and program execution resumed.

The debug features are used by the debugger host when connecting to and debugging the MCUs. The debugging interface is SW-DP. The debug features embedded in the Cortex®-M0+ core are a subset of the Arm® CoreSight Design Kit.

M0 + provides integrated on-chip debug support. It is comprised of:

- SW-DP: Serial wire
- BPU: Break point unit
- DWT: Data watchpoint trigger

It also includes debug features dedicated to the device:

- Flexible debug pinout assignment, SWIO @ PB6, SWCLK @ PA2
- MCU debug box (support for low-power modes, control over peripheral clocks, etc.)

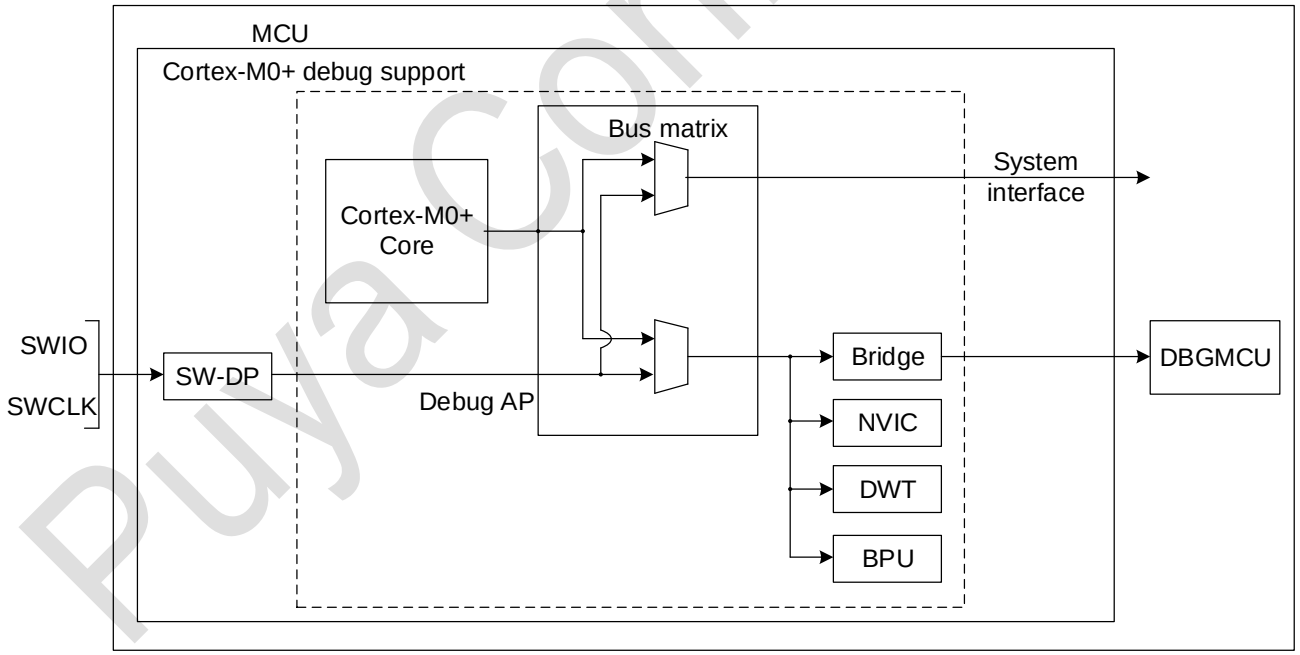

Figure 22-1 DBG block diagram

22.2. Pinout and debug port pins

22.2.1. SWD port

Two pins are related to debugging functions which are available on all packages.

Table 22-1 DBG block diagram

SW-DP pin name	SWD debug port pins		Pin assignment
	Type	Debug assignment	
SWDIO	I/O	Serial Wire Data Input/Output	PB6
SWCLK	I	Serial Wire Clock	PA2

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

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 DBGMCU_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 request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target
- Data transfer phase (33 bits) transmitted by the host or the target

Table 22-2 Packet request (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	Single bit parity of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as “1” by the target because of the pull-up

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drive the line.

Table 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

- Read accesses to the DP are not posted: the target response can be immediate (if ACK=OK) or can be delayed (if ACK=WAIT).
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. That is, the result of the previous read operation can only be obtained at the next operation. If the next access to be done is NOT an AP access, then the DP-RDBUFF register must be read to obtain the result.
- The READOK flag of the DP-CTRL/STAT register is updated on every AP read accessor RDBUFF read request to know if the AP read access was successful.
- The SW-DP implements a write buffer (for both DP or AP writes), that enables it to accept a write operation even when other transactions are still outstanding. If the write buffer is full, the target acknowledge response is “WAIT”. With the exception of IDCODE read or CTRL/STAT read or ABORT write which are accepted even if the write buffer is full.
- Because of the asynchronous clock domains SWCLK and HCLK, two extra SWCLK cycles are needed after a write transaction (after the parity bit) to make the write effective internally. These cycles should be applied while driving the line low (IDLE state). This is particularly important when writing the

CTRL/STAT for a power-up request. If the next transaction (requiring a power-up) occurs immediately, it will fail.

22.4.5. SW-DP registers

Access to these registers are initiated when APnDP=0

A[3:2]	R/W	CTRLSEL bit or SELECT registers	Register	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

Access to these registers are initiated when APnDP=1

There are many AP Registers addressed as the combination of:

- The shifted value A[3:2]
- The current value of the DP SELECT register.

22.5. Core debug

Core debug is accessed through the core debug registers. Debug access to these registers is by means of the debug access port. It consists of four registers:

Table 22-5 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 are only reset by a power-on reset. To Halt on reset, it is necessary to:

- Enable the bit0 (VC_CORRESET) of the Debug and Exception Monitor Control Register
- Enable the bit0 (VC_DEBUGEN) of the Debug Halting Control and Status Register

22.6. Break point unit (BPU)

The Cortex®-M0+ BPU implementation provides four breakpoint registers.

22.6.1. BPU functionality

The processor breakpoints implement PC based breakpoint functionality.

Refer the ARMv6-M Arm and the Arm CoreSight Components Technical Reference Manual for more information about the BPU CoreSight.

22.7. Data watchpoint trace (DWT)

The Cortex®-M0+ DWT implementation provides two watchpoint register sets.

22.7.1. DWT functionality

The processor watchpoints implement PC based watchpoint functionality.

22.7.2. DWT Program Counter Sample Register

A processor that implements the data watchpoint unit also implements the ARMv6-M optional DWT Program Counter Sample Register (DWT_PCSR). This register permits a debugger to periodically sample the PC without halting the processor.

The Cortex®-M0+ DWT_PCSR records both instructions that pass their condition codes and those that fail.

22.8. MCU debug component (DBG)

The MCU debug component helps the debugger provide support for:

- Low-power modes
- Clock control for timers and watchdog during a breakpoint

22.8.1. Debug support for low-power modes

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU implements several low-power modes which can either deactivate the CPU clock or reduce the power of the CPU.

The core does not allow FCLK or HCLK to be turned off during a debug session. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug software in low-power modes.

For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior: In Stop mode, the DBG_STOP bit must be previously set by the debugger.

22.8.2. Debug support for timers and watchdog

During a breakpoint, it is necessary to choose how the counter of timers and watchdog should behave:

- They can continue to count inside a breakpoint.
- They can stop to count inside a breakpoint.

22.9. DBG register

22.9.1. DBG device ID code register (DBG_IDCODE)

Address offset:0x00

Reset value: 0x2020 0061

Only 32-bit access supported. Read-only

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

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

Bit	Name	R/W	Reset Value	Function
31:0	DBG_IDCODE[31:0]	R	32'h2020 0061	This field indicates the device ID.

22.9.2. DBG configuration register (DBG_CR)

This register configures the low-power modes of the MCU under debug.

It is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support this feature, it is still possible for the user software to write to this register.

Address offset: 0x04

Reset value: 0x0000 0000 (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	Res
-														RW	-

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	-
1	DBG_STOP	RW	0	Debug Stop mode 0: (FCLK=Off, HCLK=Off). In Stop mode, the clock controller disables all clocks (including HCLK and FCLK). When exiting from Stop mode, the clock configuration is identical to the one after RESET (CPU clocked by the internal RC oscillator (HSI)). Consequently, the software must reprogram the clock controller. 1: (FCLK=On, HCLK=On). In this case, when entering Stop mode, FCLK and HCLK are provided by the internal RC oscillator (HSI). When exiting Stop mode, the software must reprogram the clock controller if the clock control needs to be changed.
0	Reserved	-	-	-

22.9.3. DBG APB freeze register 1 (DBG_APB_FZ1)

This register configures the clocking of timers and IWDG of the MCU under debug: The register is asynchronously reset by the POR (and not the system reset). It can be written by the debugger under system reset.

Address offset: 0x08

Power on reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM_STOP	RW	0	Clocking of LPTIM counter when the core is halted 0: Enable 1: Disable
30:13	Reserved			
12	DBG_IWDG_STOP	RW	0	Clocking of IWDG counter when the core is halted 0: Enable 1: Disable
11:0	Reserved	-	-	-

22.9.4. 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

Power on reset value: 0x0000 0000

Only 32-bit access supported. Read-only

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-															
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	-	-	-
15	DBG_TIM14_STOP	RW	0	Clocking of TIM14 counter when the core is halted 0: Enable 1: Disable
14:12	Reserved	-	-	-
11	DBG_TIM1_STOP	RW	0	Clocking of TIM1 counter when the core is halted 0: Enable 1: Disable
10:0	Reserved	-	-	-

23. Revision history

Version	Date	Descriptions
V1.0	2024.03.05	Initial version
V1.1	2024.12.23	Update 17.6.5.LPTIM control register (LPTIM_CR)
V1.2	2025.01.14	1. Modify 13.10.6. ADC sampling time register (ADC_SMPR) SMP [2:0] 2. Modify DBG_IDCODE register definition. 3. Add IWDG_STOP function. 4. V_{REFBUF}:1.5 V, 2.048 V and 2.5 V 5. Modify Figure 14-1 Comparator architecture block diagram 6. Modify 4.4.1. Flash option byte
V1.3	2025.06.27	Official version

Puya Semiconductor Co., Ltd.