megawin
make you win

User Guide

MG32F04A016

Arm®Cortex® M0 Core

User Guide

Version 1.1

Date 2025/01/03

Contents

Figures

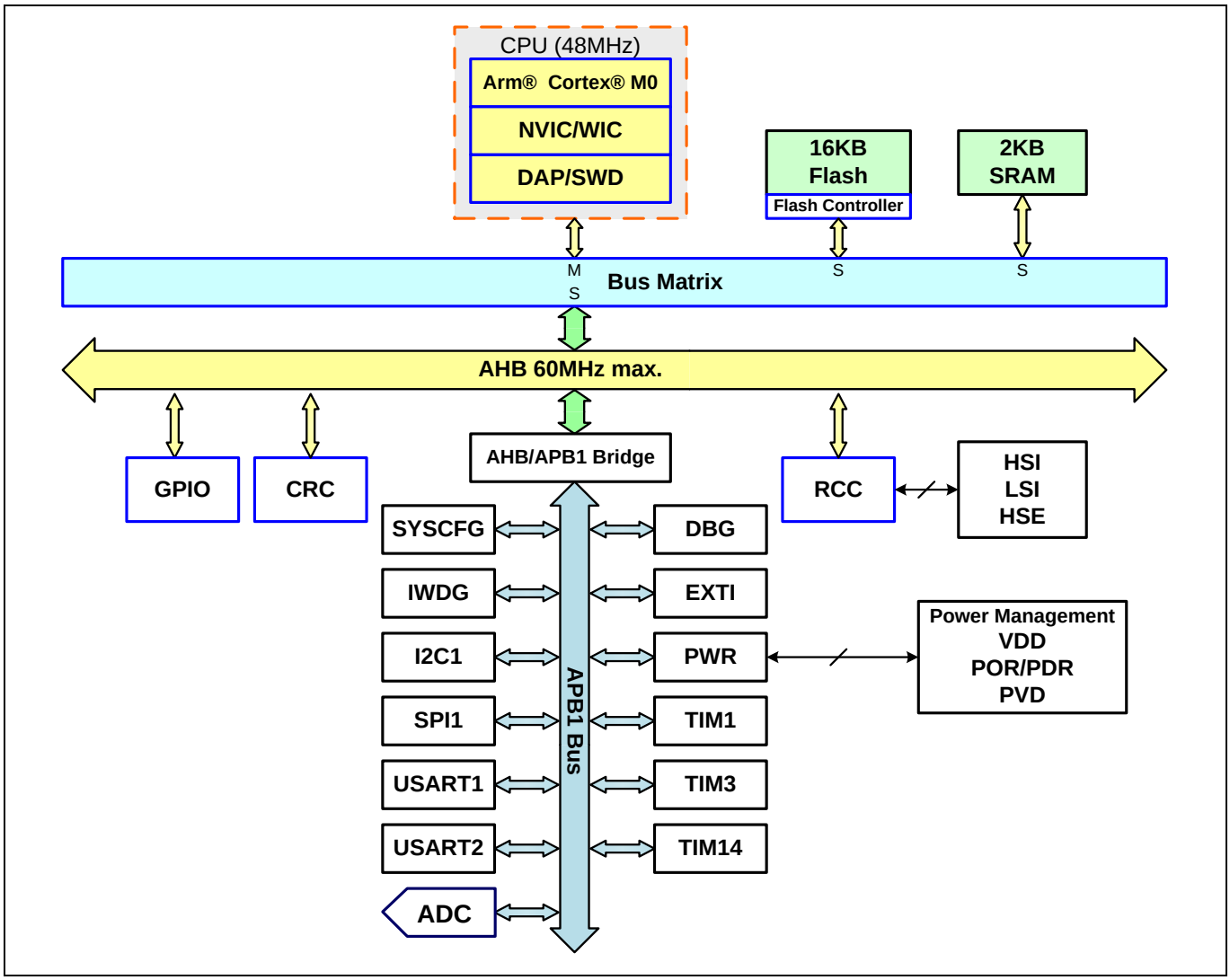
Tables

1. System and Memory Architecture

1.1 System architecture overview

MG32F04A016 product is a 32-bit microcontroller product based on Arm®Cortex®-M0 processor. It features high performance and low power consumption. It adopts the matrix bus structure, which includes one AHB master: CPU and the following slaves: SRAM, Flash memory, AHB bus (including bus bridge from AHB to APB), and a variety of devices connected to APB bus.

Figure 1-1 System architecture block diagram



1.1.1 System bus

The system bus works to achieve data transfer by connecting the CPU core and bus matrix to build the path between CPU and bus matrix. The CPU can work as the host drive bus, enabling the read and write operations to slaves such as APB peripheral, SRAM, Flash, etc.

1.1.2 BUS matrix

The bus matrix consists of an AHB interconnected matrix, an AHB bus and a bridged APB bus. Peripherals (RCC, GPIO and CRC) of the AHB bus are connected to the system bus via the AHB interconnected matrix. The connections between the APB and AHB buses exchange data via the AHB1APB bridge. When the APB registers are accessed by 8-bit, 16-bit, the APB is automatically widened to 32-bit, and similarly, the AHB1APB bridge has an automatic widening function.

1.2 Memory introduction

Program memory, data memory, registers, and I/O interfaces are located in memory address space (linear 4GB address space) with different address range. 4GB address space is divided into 8 blocks, with 512MB per block. The space allocated to the on-chip memory and peripherals is fixed, and not subject to change. The remaining address space is reserved, and can be defined and used by the chip vendor.

1.2.1 Memory map and register addressing

For the memory map, please refer to the memory map diagram in the corresponding chapter of each peripheral.

Table 1-1 Memory mapping

Bus	Address range	Size	Peripheral
Flash	0x0000 0000 - 0x0000 3FFF	16 KB	Could be mapped as main memory, system memory or SRAM, based on SYSCFG registers configuration
	0x0000 8000 - 0x07FF FFFF	~127 MB	Reserved
	0x0800 0000 - 0x0800 3FFF	16 KB	Main memory
	0x0800 8000 - 0x1FFD FFFF	~383 MB	Reserved
	0x1FFE 0000 - 0x1FFF F3FF	125KB	Reserved
	0x1FFF F400 - 0x1FFF F7FF	1 KB	System memory
	0x1FFF F800 - 0x1FFF F9FF	0.5KB	Option byte
	0x1FFF FA00 - 0x1FFF FFFF	1.5KB	Reserved
SRAM	0x2000 0000 - 0x2000 07FF	2 KB	SRAM
	0x2000 0800 - 0x2FFF FFFF	~255 MB	Reserved
APB1	0x4000 0000 – 0x4000 03FF	1KB	Reserved
	0x4000 0400 – 0x4000 07FF	1KB	TIM3
	0x4000 0800 – 0x4000 2FFF	11KB	Reserved
	0x4000 3000 – 0x4000 33FF	1KB	IWDG
	0x4000 3400 – 0x4000 43FF	4KB	Reserved
	0x4000 4400 – 0x4000 47FF	1KB	USART2
	0x4000 4800 – 0x4000 53FF	3KB	Reserved
	0x4000 5400 – 0x4000 57FF	1KB	I2C1
	0x4000 5800 – 0x4000 6FFF	6KB	Reserved
	0x4000 7000 – 0x4000 73FF	1KB	PWR
	0x4000 7400 – 0x4000 FFFF	35KB	Reserved
	0x4001 0000 – 0x4001 03FF	1KB	SYSCFG
	0x4001 0400 – 0x4001 07FF	1KB	EXTI
	0x4001 0800 – 0x4001 23FF	7KB	Reserved
	0x4001 2400 – 0x4001 27FF	1KB	ADC1
	0x4001 2800 – 0x4001 2BFF	1KB	Reserved
	0x4001 2C00 – 0x4001 2FFF	1KB	TIM1
	0x4001 3000 – 0x4001 33FF	1KB	SPI1
	0x4001 3400 – 0x4001 37FF	1KB	DBG
	0x4001 3800 – 0x4001 3BFF	1KB	USART1
	0x4001 3C00 – 0x4001 3FFF	1KB	Reserved
	0x4001 4000 – 0x4001 43FF	1KB	TIM14
	0x4001 4400 – 0x4001 FFFF	47KB	Reserved
AHB	0x4002 0000 – 0x4002 0FFF	4KB	Reserved
	0x4002 1000 – 0x4002 13FF	1KB	RCC
	0x4002 1400 – 0x4002 1FFF	3KB	Reserved
	0x4002 2000 – 0x4002 23FF	1KB	Flash Interface

	0x4002 2400 – 0x4002 2FFF	3KB	Reserved
	0x4002 3000 – 0x4002 33FF	1KB	CRC
	0x4002 3400 – 0x4002 FFFF	51KB	Reserved
	0x4800 0000 – 0x4800 03FF	1KB	PORT A
	0x4800 0400 – 0x4800 07FF	1KB	PORT B

1.2.2 Embedded SRAM

It features up to 2K bytes of a built-in static SRAM. It can be accessed in byte (8 bits), half-word (16 bits) or word (32 bits). The starting address of SRAM is 0x2000 0000.
SRAM can be accessed by CPU with the fastest system clock and without any waiting.

1.2.3 FLASH memory overview

The Flash memory is composed of two storage areas:

The main Flash memory block includes the application data and user data area.

The information block includes option bytes and system memory:

- Option bytes - Containing hardware and user storage protection configuration options.
- System memory - Containing system information.

The Flash memory interface based on the AHB protocol, executes commands and data access. The prefetch buffer function of the Flash interface can speed up the code execution of CPU.

2. Embedded FLASH

2.1 Key Flash features

The Flash memory is up to 16K bytes

Memory organization:

- Main Flash memory block: Up to 4K bytes (4K x 32 bits)
- Information block:
  - System memory: Up to 1K bytes (1K x 8 bits)
  - Option bytes: Up to 2 x 8 bits

The Flash memory interface features:

Read interface with prefetch buffer (2 x 32-bit words)

Option byte loader

Flash Program/Erase operation

Read/Write protection

Low power consumption mode

2.2 Flash memory function description

2.2.1 Flash memory organization

The Flash memory is organized as 32-bit wide memory cells that can be used for storing both code and data. The main Flash memory block is partitioned by 16 pages (1K bytes per page) or 4 sectors (4K bytes per sector), with write protection on a sector-by-sector basis (refer to the memory protection part).

Table 2-1 Flash module organization

Module	Name	Address	Size(byte)
Main memory block	Page 0	0x0800 0000 - 0x0800 03FF	1K
	Page 1	0x0800 0400 - 0x0800 07FF	1K
	Page 2	0x0800 0800 - 0x0800 0BFF	1K
	Page 3	0x0800 0C00 - 0x0800 0FFF	1K
	Page 15	0x0800 3C00 - 0x0800 3FFF	1K
Information block	System memory	0x1FFF F400 - 0x1FFF F7FF	1K
	Option bytes	0x1FFF F800 - 0x1FFF F80F	16
Flash memory interface registers	FLASH_ACR	0x4002_2000 - 0x4002 2003	4
	FLASH_KEYR	0x4002_2004 - 0x4002 2007	4
	FLASH_OPTKEYR	0x4002_2008 - 0x4002 200B	4
	FLASH_SR	0x4002_200C - 0x4002 200F	4
	FLASH_CR	0x4002_2010 - 0x4002 2013	4
	FLASH_AR	0x4002_2014 - 0x4002 2017	4
	Reserved	0x4002_2018 - 0x4002 201B	4
	FLASH_OBR	0x4002_201C - 0x4002 201F	4
	FLALSH_WRPR	0x4002_2020 - 0x4002 2023	4

2.2.2 Read operation of the Flash memory

The embedded Flash module can be addressed directly, as a common memory space. Any read operation that accesses the content of the Flash module should go through a dedicated judgmental process.

Instruction and data fetches are performed through the AHB bus in the way that is specified as the option in the Flash access control register (FLASH_ACR).

● Instruction fetch: CPU operation speed can be accelerated after enabling the prefetch buffer.

● Latency: The number of wait bits for correct reading.

Instruction fetch

The CPU fetches the instructions over the AHB bus. The prefetch block aims at increasing the efficiency of instruction fetch.

Prefetch buffer

Prefetch buffer (2 x 32-bit blocks): The buffer is enabled automatically after each reset. A block can be completely updated on a single read from the Flash memory as the size of each block (32 bits) matches the bandwidth of the Flash memory. The implementation of this prefetch buffer makes a faster CPU execution possible. The CPU can fetch one word up to 32 bits at one time, as it fetches one instruction at a time with the next instruction readily available in the prefetch buffer.

Prefetch controller

The prefetch controller decides to access the Flash memory depending on the available space in the prefetch buffer. The controller initiates a read request when there is at least one block free in the prefetch buffer. After reset, the default state of the prefetch buffer is on. The prefetch buffer should be switched on/off only when SYSCLK is lower than 24MHz and no prescaler is applied on the AHB clock (SYSCLK must be equal to HCLK). The prefetch buffer is usually switched on/off during the initialization routine, while the MCU is running on the internal 8MHz oscillator.

Note: The latency for prefetch buffer must be kept on when using a prescaler different from 1 on the AHB clock.

Access latency

In order to maintain the correct reading from the Flash memory, the speed ratio of the prefetch controller has to be specified in the LATENCY [2:0] of the Flash access control register. This value gives the number of cycles needed to insert between each access of the Flash memory and next access. After reset, the value is zero, i.e. no wait cycle is inserted.

2.2.3 Flash write and erase operations

The embedded Flash memory supports the in-circuit programming as well as the in-application programming.

The in-circuit programming (ICP) method refers to updating the contents of the Flash memory in the circuit by loading the user codes into the microcontroller. ICP offers a quick and efficient method and eliminates unnecessary socketing of devices during chip programming.

In contrast to the ICP method, in-application programming (IAP) can use any communication interface supported by the MCU (I/Os, USART, I2C, SPI, etc.) to download programs or data. IAP allows the user to re-program the application while the application is running. Nevertheless, part of the application has to be previously programd in the Flash memory using ICP.

The write and erase operations can be performed within the whole operating voltage range of the product. They are performed by the following 7 registers:

● Key register (FLASH_KEYR)
● Option byte key register (FLASH_OPTKEYR)
● Flash control register (FLASH_CR)
● Flash status register (FLASH_SR)
● Flash address register (FLASH_AR)
● Option byte register (FLASH_OBR)
● Write protection register (FLASH_WRPR)

An ongoing Flash memory write operation will not block the CPU as long as the CPU does not access the Flash memory. That is to say, during a write/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 write/erase operation has completed. This means that instruction or data fetches cannot be made while a write/erase operation is ongoing. For write/erase operations on the Flash memory, the internal oscillator (HSI) must be ON.

Unlocking the Flash memory

After reset, the Flash memory is protected in default to prevent unexpected erase operation. The FLASH_CR register is not allowed to be rewritten, unless an unlocking sequence is executed for the FLASH_CR register to enable access permission to the FLASH_CR. This sequence consists of two write operations:

● Write Key 1 = 0x45670123
● Write Key 2 = 0xCDEF89AB

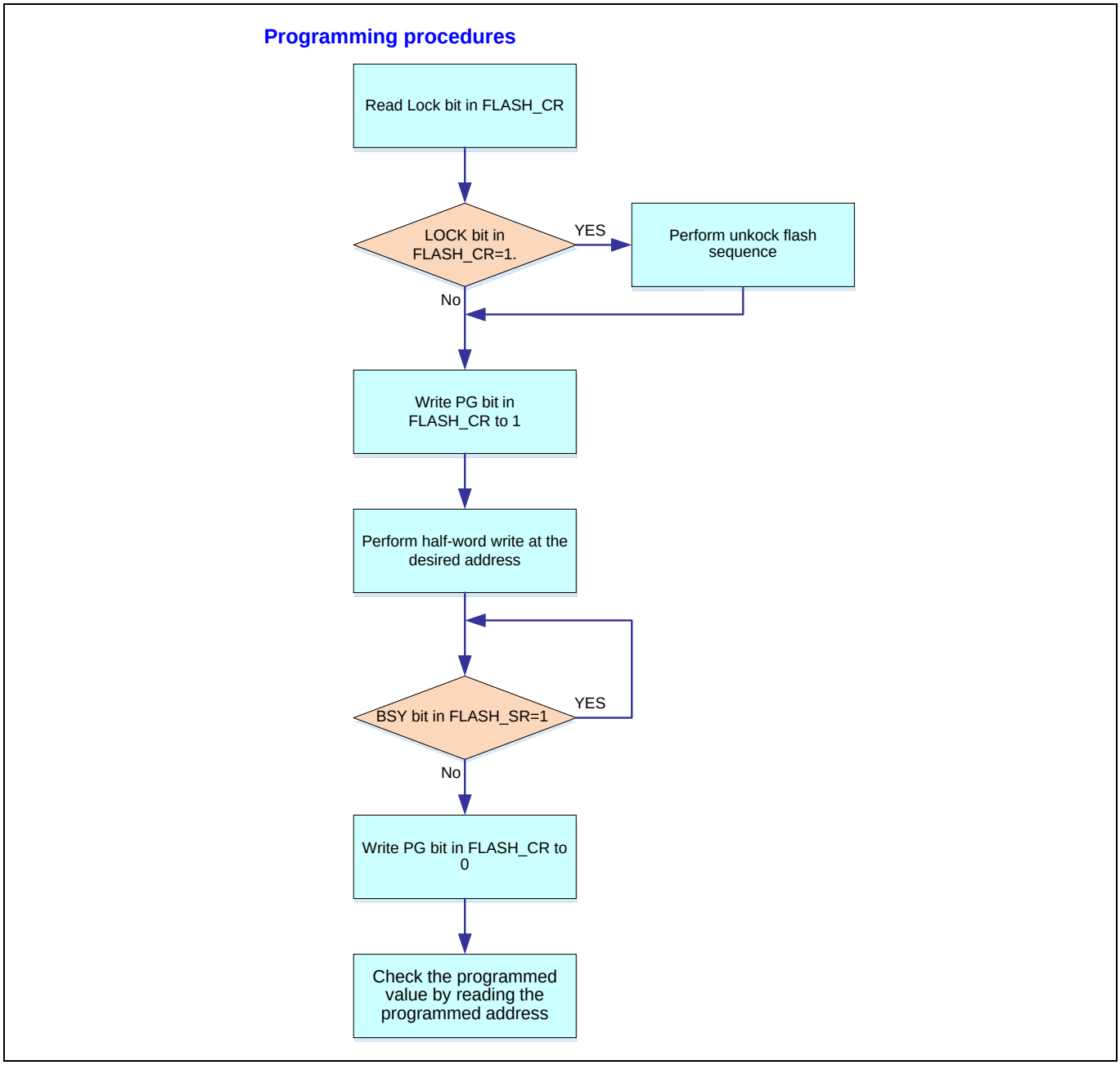
Any wrong sequence locks up the FLASH_CR register until the next reset.

Also a bus error is returned on a wrong key sequence, which will then trigger a hardware error response. This is done immediately if KEY1 is wrong, or if KEY1 has been correctly written but KEY2 is wrong.

2.2.3.1 Main Flash memory programming

The main Flash memory can be programd 16 bits at a time. One programming operation is done when a half-word (16 bits) is written into the corresponding address with the PG bit of the FLASH_CR to be 1. Any attempt to write data that are not half-word long will result in a hardware error response.

Figure 2-1 Programming procedures



The Flash memory interface prefetches the bytes to be programd and checks whether they are all set to 1. If not, the program operation is skipped and a programming error warning is issued by the PGERR bit in the FLASH_SR register.

If the address to be programd is write-protected by the FLASH_WRPR register, the program operation is skipped and a programming error warning is issued, either. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.

The main Flash memory programming sequence in standard mode is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Set the PG bit in the FLASH_CR register
- Perform the data write in a unit of half-word at the desired address
- Wait for the BSY bit in the FLASH_SR register to return to zero

● Read the data and verify

Note: These registers are not to be written when the BSY bit of the FLASH_SR register is set to 1.

Flash memory erase

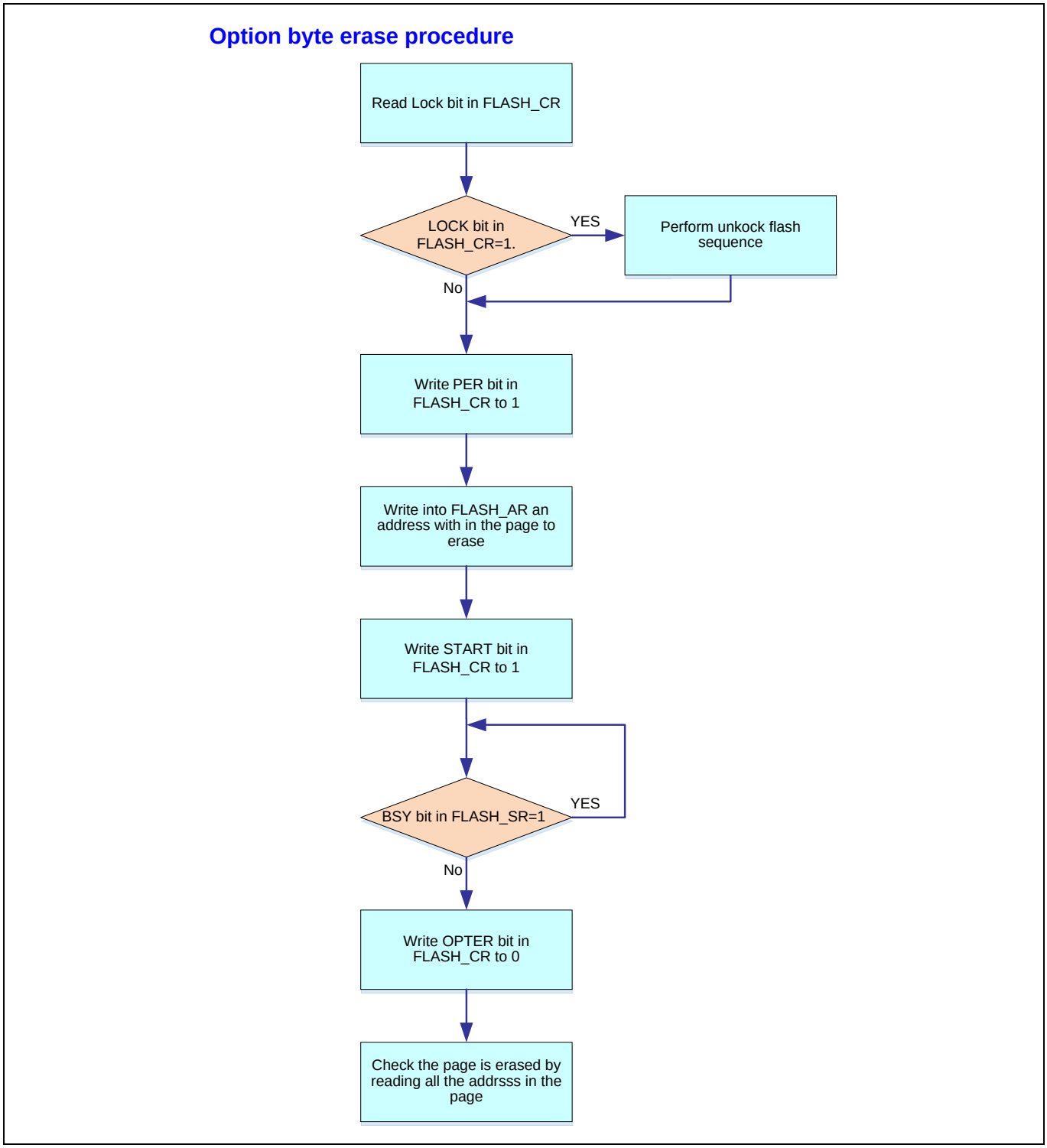
The Flash memory can be erased page by page or completely as mass erase.

2.2.3.2 Page erase

To erase a page, the procedure below should be followed:

● Check that last operation is finished by checking the BSY bit in the FLASH_SR register
● Set the PER bit to 1 in the FLASH_CR register
● Write the FLASH_AR register to select a page to erase
● Set the STRT bit to 1 in the FLASH_CR register
● Wait for the BSY bit in the FLASH_SR register to return to zero
● Read the erased page and verify

Figure 2-2 Flash register page erase procedures

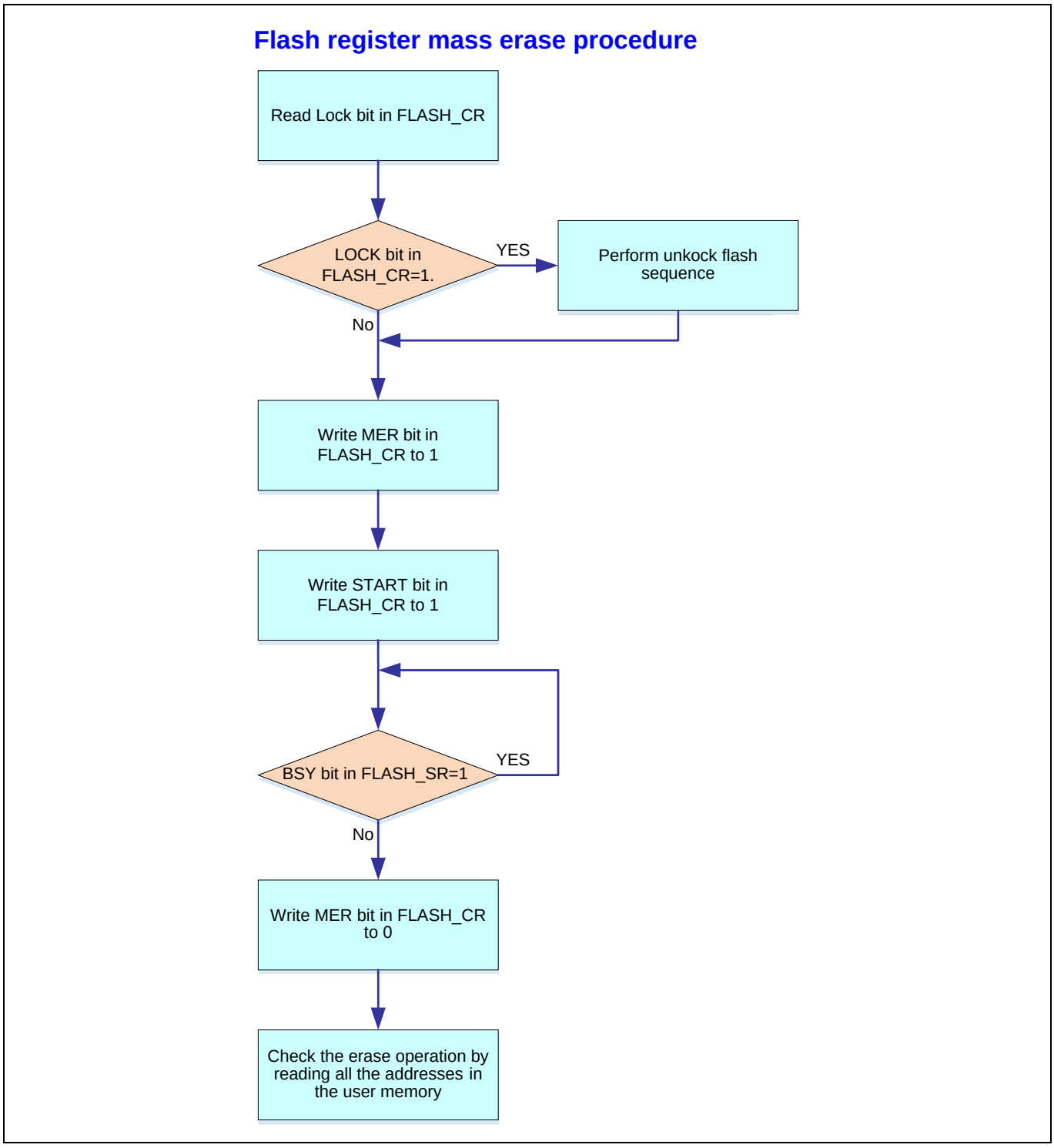


2.2.3.3 Mass Erase

The Mass Erase command can be used to completely erase the whole Flash user area. The information block is unaffected by this instruction. The specific sequence is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Set the MER bit to 1 in the FLASH_CR register
- Set the STRT bit to 1 in the FLASH_CR register
- Wait for the BSY bit to return to zero
- Read all the pages and verify

Figure 2-3 Flash register mass erase procedures

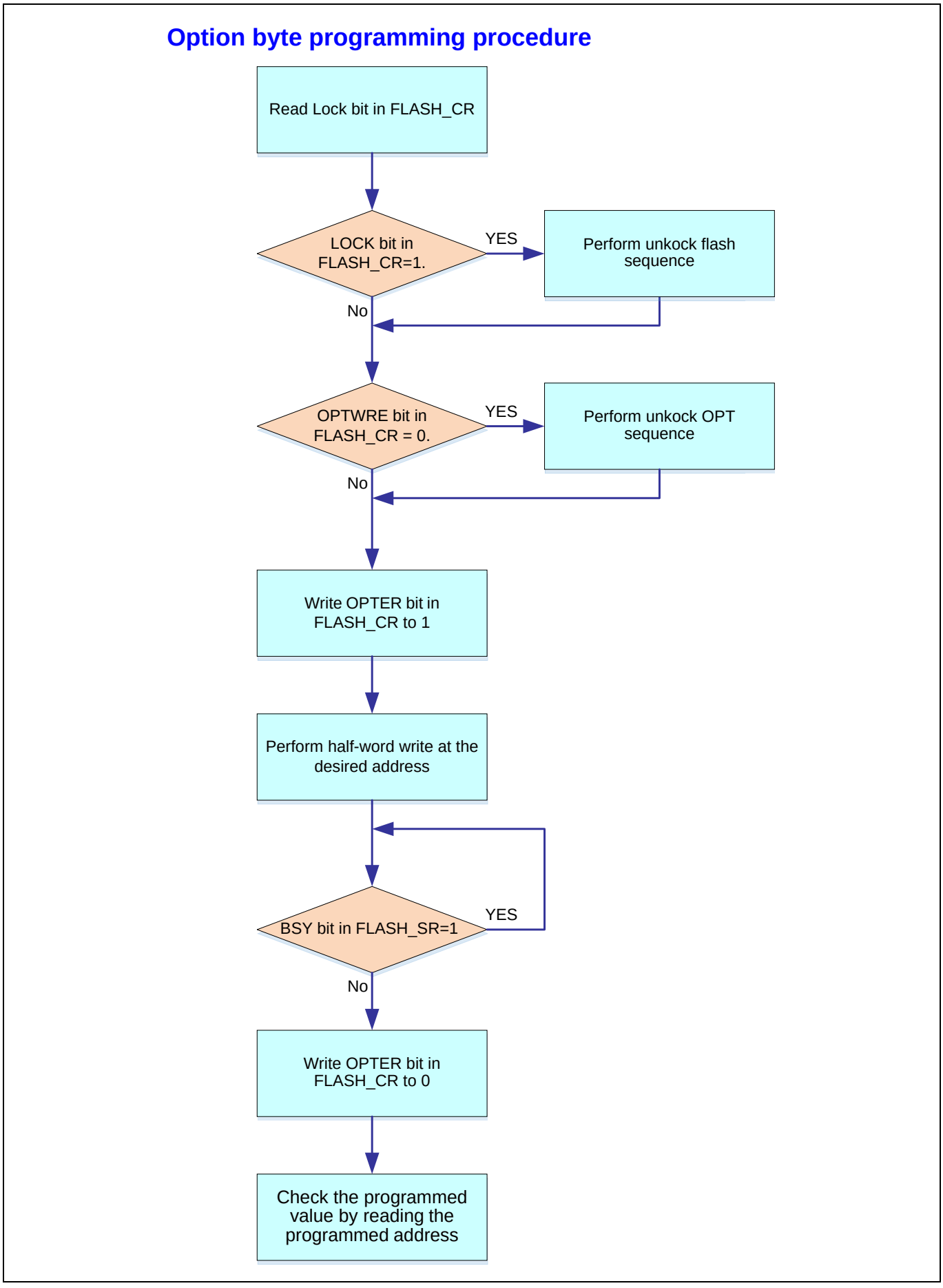


2.2.3.4 Option byte programming

The option bytes are programd differently from normal user addresses, including two for write protection and one for hardware configuration. After unlocking the access permission to the Flash, the writing of keys into the FLASH_OPTKEYR register has to be performed. Once it is done, the OPTWRE bit in the FLASH_CR register will be set to ’1’. Then the user has to set the OPTPG bit in the FLASH_CR register and perform a half-word write operation at the desired address. Similarly, the option bytes are checked to confirm they are set as 1. If not, the relevant operation is skipped and an error warning is issued by the PGERR bit in the FLASH_SR register. The end of the program operation is indicated by the EOP bit in the FLASH_SR register.
The option byte is 16-bit data. The significant data is the 8 lower bits, while the 8 higher bits are the complement of the 8 lower bits. During the programming process, the user has to calculate the complement of the 8 lower bits and consider it as the 8 higher bits first, and write it together with the 8 lower bits as a half-word. Ensure that the value written into the option byte always makes the 8 higher bits and the 8 lower bits as the radix-minus-one complements to each other. The sequence is as follows:

- Check that last operation is finished by checking the BSY bit in the FLASH_SR register
- Unlock the OPTWRE bit in the FLASH_CR register
- Set the OPTPG bit to 1 in the FLASH_CR register
- Write the data (half-word) to the desired address
- Wait for the BSY bit to return to zero
- Read the data and verify that a mass erase is automatically performed when the protection option byte is changed from protected to unprotected. If the user wants to change an option other than the protection option, then the mass erase is not performed. This mechanism is used to protect the content of the Flash memory.

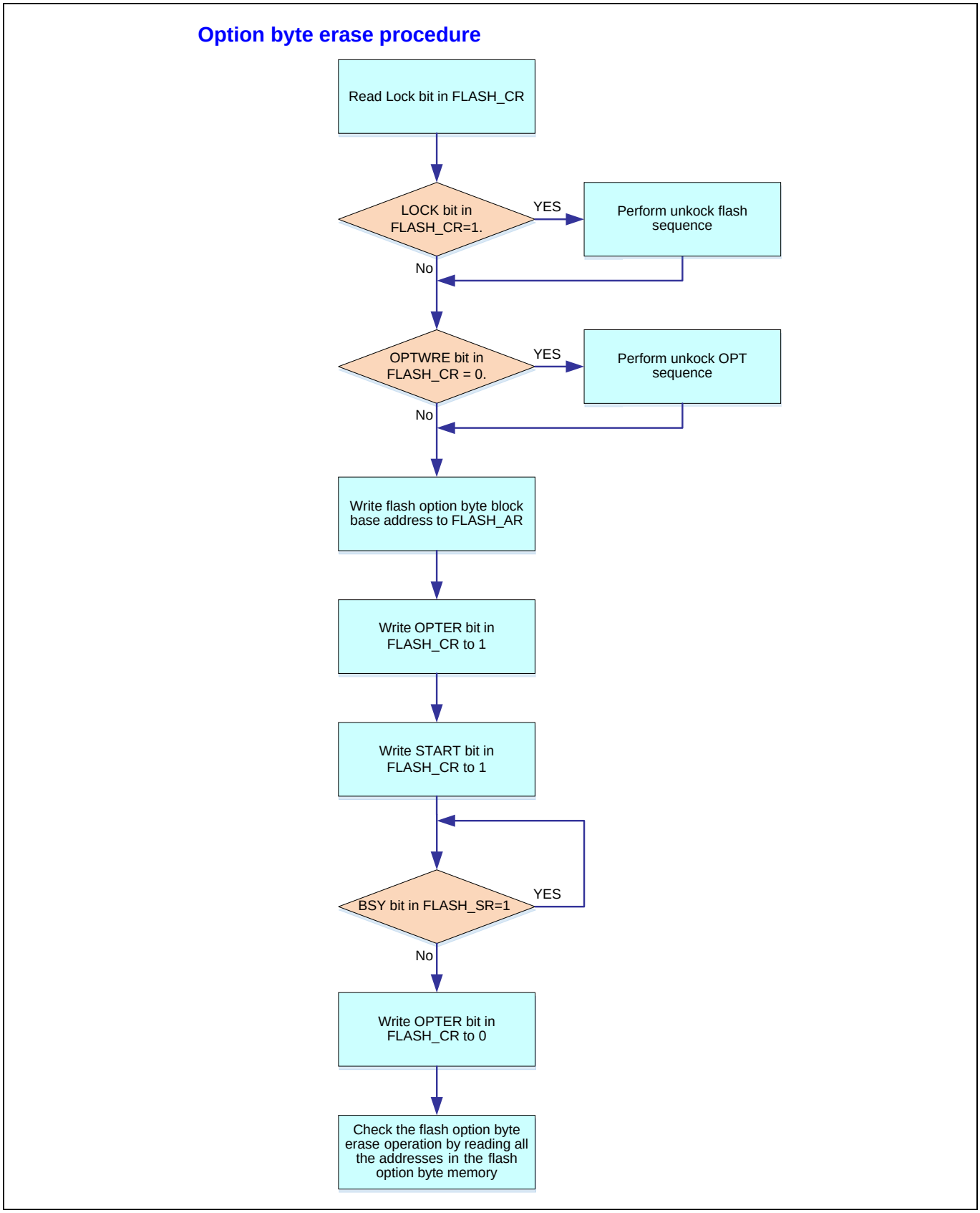
Figure 2-4 Option byte programming procedures



2.2.3.5 Erase procedure

The option byte erase sequence is as follows:

Figure 2-5 Option byte erase procedure



2.3 Memory protection

The user area of the Flash memory can be protected against read by untrusted code. The Flash memory can also be protected against unwanted erase due to program runaway. The write-protection granularity is then of one sector (four pages).

2.3.1 Read operation

The read protection is activated by setting the RDP option to a half word and then, by re-applying a system POR (power-on reset) to reload new RDPs.
Note: If the read protection is set while the debugger is still connected through SWD/JTAG, apply a POR instead of a system reset (without debugger connection).

2.3.1.1 Set as read protection

Refer to the half-word programming method in the option byte area, and write a half word for the RDP option in sequence to the corresponding address.
Set the Flash AR address value to 0x1FFFF800 and execute the block erase
Write the desired value of 0x807F to 0x1FFFF800
Reset the desired system via software, or turn off the power before turning on it again
Once corresponding values have been written to the protection byte:
● Main Flash memory read access is not allowed except for the user code (when booting from main Flash memory itself with the non-debug mode).
● After the read protection is activated, the Flash is inaccessible in the debug mode (sram boot and debug modes). The flash's own programs should not occupy the 4KB space.
● Pages 0 ~ 3 are automatically write-protected. The rest of the memory can be programd by the code executed from the main Flash memory (for IAP or data storage, etc.), but it is protected against write/erase (but not against mass erase) in debug mode or when booting from the embedded SRAM.
● All features for loading code into and executing code from the embedded SRAM via SWD are still active. Booting from embedded SRAM via SWD is also allowed and this can be used to disable the read protection.
● When executing code from the embedded SRAM to access the main Flash memory, Flash memory accesses through the SWD (serial wire debug) is not allowed.
The Flash memory is protected when the RDPs contain the following values:

Table 2-2 Flash memory read protection status

RDPs Byte Value	Read Protection Status
RDP=0x807F @0x1FFFF800	Protection

Note: If the corresponding address value for the option byte block is not 0xFFFF, the erase of option byte block should be executed first. Erasing the option byte block will not trigger an automatic mass erase or change the protection status.

2.3.1.2 Unprotection

To disable the read protection from the embedded SRAM:
Refer to the half-word programming method in the option byte area, and write a half word for the RDP option in sequence to the corresponding address:
Set the Flash AR address value to 0x1FFFF800 and execute the block erase
Write the desired value of 0x5AA5 to 0x1FFFF800 to trigger a mass erase of the main Flash memory. At this stage the read protection is not activated.
Apply a POR reset to reload the option bytes, to disable the read protection.

Table 2-3 Flash memory read unprotecting status

RDPs Byte Value	Read Protection Status
Erase the 0x1FFFF800 option block Write RDP=0x5AA5 @0x1FFFF800 **Trigger the main Flash mass erase for 0x08000000**	Protection is disabled

2.3.2 Write protection of main memory

The write protection is controlled in a unit of sector (4 pages). The WRP bit in the option bytes should be configured, and then the system reset will load the new option bytes to enable the protection. If a write or an erase operation is performed on a protected sector. The WRPRTERR flag bit is set on the FLASH_SR.

2.3.3 Option byte write protection

The option byte blocks are always read-accessible and write-protected by default. To gain write access (program/erase) to the option byte blocks, a sequence of keys (same as for lock) has to be written into the OPTKEYR. It then gives write access to the option byte blocks and this is indicated by OPTWRE bit in the FLASH_CR register. Write access can be disabled by clearing the bit.

2.4 Flash interrupt

Table 2-4 Flash interrupt requests

Interrupt Event	Event Symbol	Enable Control Bit
End of operation	EOP	EOPIE
Write protection error	WRPRTERR	ERRIE
Programming error	PGERR	ERRIE

2.5 Option byte description

Option bytes are configured by the user depending on the application requirements. As an example, the watchdog may be selected in hardware or software mode.
A 32-bit word is split up as follows in the option bytes:

Table 2-5 Option byte format

Bits 31 ~ 24	Bits 23 ~ 16	Bits 15 ~ 8	Bits 7 ~ 0
Complemented option byte 1	Option byte 1	Complemented option byte 0	Option byte 0

Note: Complements must be implemented by the user application software.
The organization of these option bytes inside the option byte block is as below.
The option bytes can be read from the memory locations listed in the table below or from the option byte register (FLASH_OBR).
Note: The new programd option bytes (user, read/write protection) take effect after a system reset.

Table 2-6 Option byte organization

Address	[31：24]	[23：16]	[15：8]	[7：0]
0x1FFF F800	nUSER	USER	nRDP	RDP
0x1FFF F804	nData1	Data1	nData0	Data0
0x1FFF F808			nWRP0	WRP0

Table 2-7 Description of the option bytes

Flash memory address	Option bytes
0x1FFF F800	Bits [31：24] nUSER Bits [23:16] USER: User option byte (stored in FLASH_OBR[9:2]) This byte is used to con- figure the following features: Select the watchdog event: hardware or software Note: Bits [23: 18] are not used. Bit 17：nRST_STOP 0: Reset is generated when entering STOP mode 1: No reset when entering STOP mode Bit 16：WDG_SW 0: Hardware watchdog 1: Software watchdog Bits [15：8]: nRDP Bits [7：0]: RDP: read protection option byte The read protection helps the user protect the code stored in Flash memory. It is activated by setting the RDP option byte. When this option byte is programd to a correct value (RDP = 0x807F), read access to the Flash memory is forbidden. (The result of RDP enabled/ disabled is stored in FLASH_OBR [1].)
0x1FFF F804	Datax: two bytes for user data The address can be programd using the option byte programming procedure. Bits [31：24]：nData1 Bits [23：16]：Data1(stored in FLASH_OBR[25：18]) Bits [15：8]：nData0 Bits [7：0]：Data0(stored in FLASH_OBR[17：10]) WRPx: Flash memory write protection option bytes Bits [15：8]：nWRP0 Bits [7：0]：WRP0(stored in FLASH_WRPR[7：0])
0x1FFF F808	Each bit of the option byte WRPx is used to protect 4 pages in main memory: 0: write protection active 1: write protection not active In total, one user option byte is used to protect the 32-Kbyte main Flash memory. WRP0: write-protects for pages 0 ~ 31

On every system reset, the option byte loader (OBL) reads the information block and stores the data into the option byte register (FLASH_OBR). Each option bit also has its complement in the information block. During option bit loading, by verifying the option bit and its complement, it is possible to check that the loading has correctly taken place. If this is not the case, an option byte error (OPTERR) is generated. When an error occurs, the corresponding option byte is forced to 0xFF. The verification is disabled when the option byte and its complement are both equal to 0xFF (after-erase state).
All option bits (but not their complements) are available to configure the microcontroller. The option byte registers are accessible in read mode by the CPU.

2.6 Register

2.6.1 Register overview

Table 2-8 FLASH register overview

Offset	Acronym	Register Name	Reset
0x00	FLASH_ACR	Flash access control register	0x00000038
0x04	FLASH_KEYR	FPEC key register	0x00000000
0x08	FLASH_OPTKEYR	Flash OPTKEY register	0x00000000

0x0C	FLASH_SR	Flash status register	0x00000000
0x10	FLASH_CR	Flash control register	0x00000080
0x14	FLASH_AR	Flash address register	0x00000000
0x1C	FLASH_OBR	Option byte register	0x03FFFC1C
0x20	FLASH_WRPR	Write protection register	0xFFFFFFFF

2.6.2 FLASH_ACR Flash Access Control Register

Offset address: 0x00

Reset value: 0x0000 0008

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.	PRFTBE	Res.	LATENCY		
											rw		rw		

31：6	Res.	Reserved, must retain the reset value
5	Res.	Reserved, must retain the reset value
4	PRFTBE	Prefetch buffer enable 0: Close prefetch buffer. 1: Open prefetch buffer. Note 1: Only when LATENCY is set to 0, can the prefetch buffer be controlled through this bit.
3	Res.	Reserved, must retain the reset value
2：0	LATENCY	Latency These bits indicate the ratio of SYSCLK (system clock) period to Flash memory access time. 000: zero wait state, 0 < SYSCLK ≤ 24MHz 001: one wait state, 24MHz < SYSCLK ≤ 48MHz

2.6.3 FLASH_KEYR FPEC Key Register

Offset address: 0x04

Reset value: 0x0000 0000

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

Bit	Field	Description
31：0	FKEYR	FPEC Flash key These bits are used to input FPEC unlock key.

Note: All these bits are write-only and return to 0 when read.

2.6.4 FLASH_OPTKEYR Flash OPTKEY Register

Offset address: 0x08

Reset value: 0x0000 0000

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

Bit	Field	Description
31：0	OPTKEYR	Option byte key These bits are used as the input option byte key to disable OPTWRE.

Note: All these bits are write-only and return to 0 when read.

2.6.5 FLASH_SR Flash Status Register

Offset address: 0x0C

Reset value: 0x0000 0000

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

Bit	Field	Description
31:6	Reserved	Reserved, must retain the reset value
5	EOP	End of operation When the Flash memory operation (programming I erasing) is completed, the hardware sets this bit as “1” and write “1” to clear the bit status. Note: EOP status is set at the end of each successful program or erase operation.
4	WRPRTERR	Write protection error When an attempt is made to program the write-protected Flash memory address, the hardware sets this bit as “1” and write “1” to clear the bit status.
3	Res.	Reserved, must retain the reset value
2	PGERR	Programming error When an attempt is made to program an address whose content is not “0x FFFF”, the hardware sets this bit as “1” and write “1” to clear the bit status Note: Prior to programming, clear the STRT bit of FLASH_CR register. Note: When the PGERR bit is 1, the CPU cannot be cleared and cannot be written again. It can only be reset.
1	Res.	Reserved, must retain the reset value
0	BSY	Busy This bit indicates that the operation of Flash memory is in progress. Before Flash operation starts, this bit is set as “1”. After the operation or in case of an error, this bit is cleared as “0”.

2.6.6 FLASH_CR Flash Control Register

Offset address: 0x10

Reset value: 0x0000 0080

31	30	29	28	28	27	26	25	24	23	22	21	20	19	18	17
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			EOPIE	Res.	ERRIE	OPTWRE	Res.	LOCK	STRT	OPTER	OPTPG	Res.	MER	PER	PG
						rw		rw	rw	rw	rw		rw	rw	rw

Bit	Field	Description
31:10	Res.	Reserved, must retain the reset value
12	EOPIE	End of operation interrupt enable This bit enables the interrupt generation when the EOP bit in the FLASH_SR register goes to '1'. 0: interrupt generation disabled 1: interrupt generation enabled
11	Res.	Reserved, must retain the reset value
10	ERRIE	Error interrupt enable This bit enables the interrupt generation on an FPEC error (when PGERR/WRPRTERR are set to '1'in the FLASH_SR register). 0: interrupt generation disabled 1: interrupt generation enabled
9	OPTWRE	Option byte write enable When this bit is "1", the option byte is allowed to be program. When the correct key sequence is written in the FLASH_OPTKEYR register, the bit is set as "1". This bit can be cleared by writing 0.
8	Res.	Reserved, must retain the reset value
7	LOCK	Lock Only write "1". When the bit is "1", it indicates that FPEC and FLASH_CR are locked. When the correct unlocking sequence is detected, the hardware automatically clears this bit as "0". After an unsuccessful unlocking operation, this bit cannot be changed before the next system reset.
6	STRT	Start An erasing is triggered when this bit is "1". The bit can only be set to "1" by software and automatically be cleared to "0" when BSY turns into "1".
5	OPTER	Option byte erase Erase option byte.
4	OPTPG	Option byte programming Program option byte.
3	Res.	Reserved, must retain the reset value
2	MER	Mass erase Choose to erase all user pages.
1	PER	Page erase Choose to erase page.
0	PG	Programming Choose to conduct programming.

2.6.7 FLASH_AR Flash Address Register

Offset address: 0x14

Reset value: 0x0000 0000

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

Bit	Field	Description
31：0	FAR	Flash Address

		Select the page to be erased when performing page erasing. Note: When the BSY bit in FLASH_SR is “1”, the register cannot be written.

The address is modified by the hardware to the currently used one. During the operation of page erasing, the register is modified to specify the page to be erased.

2.6.8 FLASH_OBR Option Byte Register

Offset address: 0x1C

Reset value: 0x03FF FC1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.								Data1						Data0	
								r						r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data0						Res.						nRST_STOP	WDG_SW	RDPRT	OPTERR
r												r	r	r	r

Bit	Field	Description
31：26	Res.	Reserved, must retain the reset value
25：18	Data1	Data1
17：10	Data0	Data0
9: 4	Res.	Res.
3	nRST_STOP	Reset event in stop mode 0: Reset in stop mode 1: No reset in stop mode
2	WDG_SW	Select watchdog event 0: Hardware watchdog 1: Software watchdog
1	RDPRT	Read protection level status When setting to “1”, it indicates that the Flash memory is read protected. Note: The bit is read-only.
0	OPTERR	Option byte error When this bit is “1”, it indicates that the option byte does not match its inverse code. Note: The bit is read-only.

2.6.9 FLASH_WRPR Write Protection Register

Offset address: 0x20

Reset value: 0xFFFF FFFF

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

Bit	Field	Description
31:4	Reserved	Reserved, must retain the reset value
3:0	WRP	Write protection The register includes the write protection option byte loaded by OBL. 0: Write protection enabled 1: Write protection disabled

3. CRC Cyclic Redundancy Check Calculation Unit

3.1 Introduction

The CRC calculation unit utilizes a fixed polynomial to compute the CRC checksum value of 32-bit data, used to verify the integrity of data transmission or storage.

3.2 Key Features

Supports CRC-32/MPEG-2 (Ethernet) polynomial: 0x4C11DB7

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

Supports 32-bit wide data register for input/output

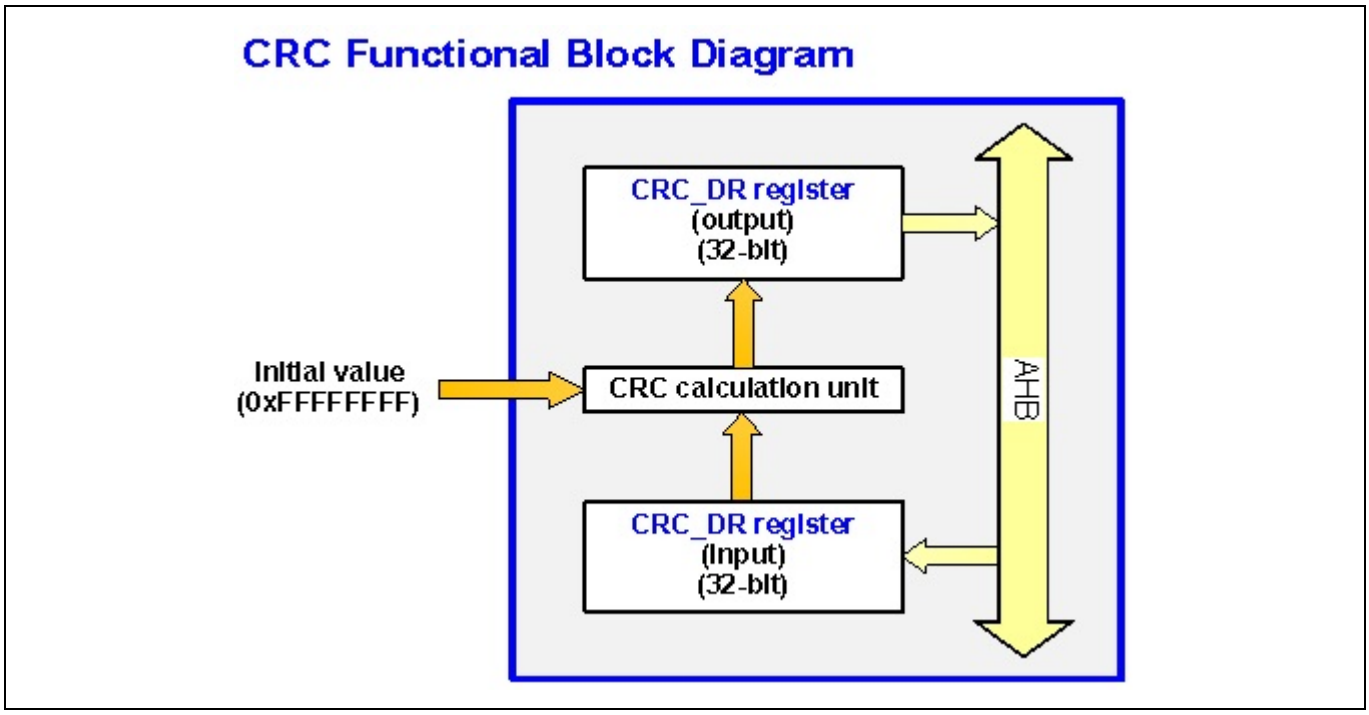
Hardware computation time is 3 HCLK cycles

8-bit independent data register for storing temporary data

3.3 Function Description

3.3.1 Functional Block Diagram

Figure 3-1 CRC Functional Block Diagram



3.3.2 Functional Overview

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

When written to, it acts as an input register, accepting new data for CRC calculation.

When read from, it returns the result of the previous CRC calculation.

Each write to the data register combines the previous CRC result with the new computation result (CRC calculation is done on the entire 32-bit word, not byte by byte).

During CRC calculation, write operations are paused, allowing for consecutive writes or continuous write-read operations on register CRC_DR.

The data register CRC_DR can be reset to 0xFFFFFFFF by setting the RST bit in register CRC_CR. This operation does not affect the data inside register CRC_IDR.

3.3.3 Usage

3.3.3.1 CRC Calculation Operation Steps

Enable the clock to the CRC module.
Reset the CRC module.
Restore the CRC to its initial state by configuring the RST bit in the CRC control register (CRC_CR).
Write data sequentially into the CRC data register (CRC_DR).
Read the CRC data register (CRC_DR) to obtain the CRC calculation result.

3.4 Register

3.4.1 Register Overview

Table 3-1 CRC Register Overview

Offset	Acronym	Register Name	Reset
0x00	CRC_DR	CRC Data Register	0xFFFFFFFF
0x04	CRC_IDR	CRC Independent Data Register	0x00000000
0x08	CRC_CR	CRC Control Register	0x00000000

3.4.2 CRC_DR CRC Data Register

Offset Address: 0x00
Reset Value: 0xFFFF FFFF

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

Bit	Field	Description
31：0	DR	Data Register When written to, acts as an input register, performing CRC calculation on the written data and the previous result. When read from, returns the result of the CRC calculation.

3.4.3 CRC_IDR CRC Independent Data Register

Offset Address: 0x04
Reset Value: 0x0000 0000

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

Bit	Field	Description
31：8	Reserved	Reserved, must be kept at reset value
7：0	IDR	General-purpose 8-bit Data Register Temporarily holds space for one byte of data. Not affected by the RST bit in the CRC Control Register (CRC_CR).

3.4.4 CRC_CR CRC Control Register

Offset Address: 0x08

Reset Value: 0x0000 0000

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

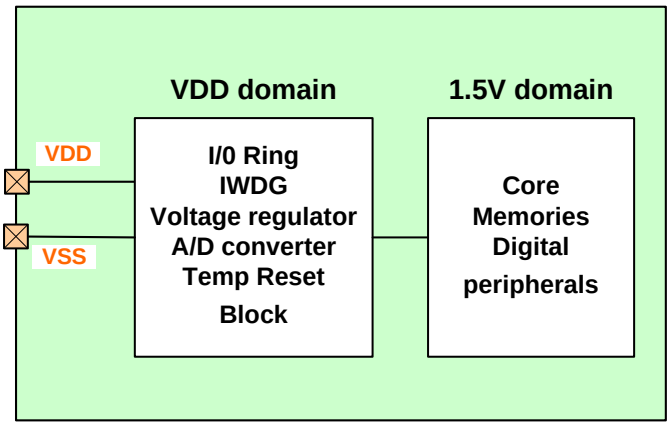
Bit	Field	Description
31：1	Reserved	Reserved, must be kept at reset value
0	RST	CRC Reset CRC Data Register (CRC_DR) is reset to 0xFFFFFFFF. This bit can only be written as "1," and hardware automatically clears it to "0".

4. PWR Power Controller

4.1 Power supplies

The chip requires a 2.0V ~ 5.5V operating voltage supply (V_{DD}). An embedded regulator is used to supply the required core power.

Figure 4-1 Power supply block diagram



4.1.1 Voltage regulator

The voltage regulator is always enabled after reset. It works in three different modes depending on the application modes.

In Run mode, the regulator supplies power to the 1.5V domain (core, memories and peripherals) normally.

In Stop mode, the regulator supplies low-power to the 1.5V domain, preserving contents of registers and SRAM.

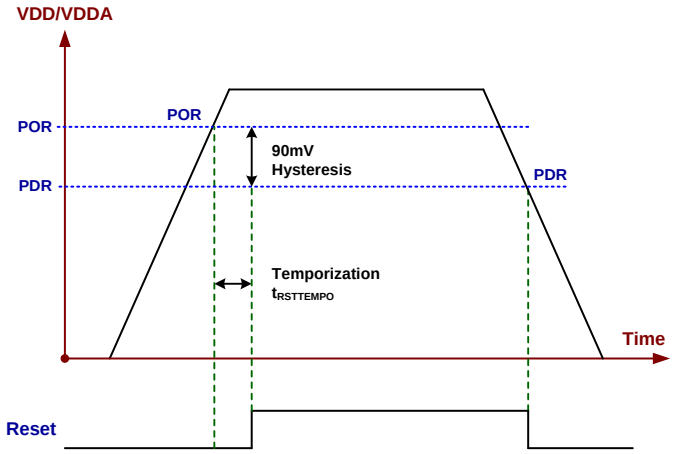
In Deep Stop mode, the core works in a lower power consumption mode, all clocks stop, and the contents of registers and SRAM are saved.

4.2 Power supply supervisor

4.2.1 Power on reset and power down reset

The chip has an integrated POR/PDR circuitry that allows proper operation starting from/down to 2.0V of supply. The device remains in Reset mode when VDD/VDDA is below a specified threshold, VPOR/VPDR, without the need for an external reset circuitry. For more details concerning the power-on/power-down reset, refer to the electrical characteristics of the datasheet.

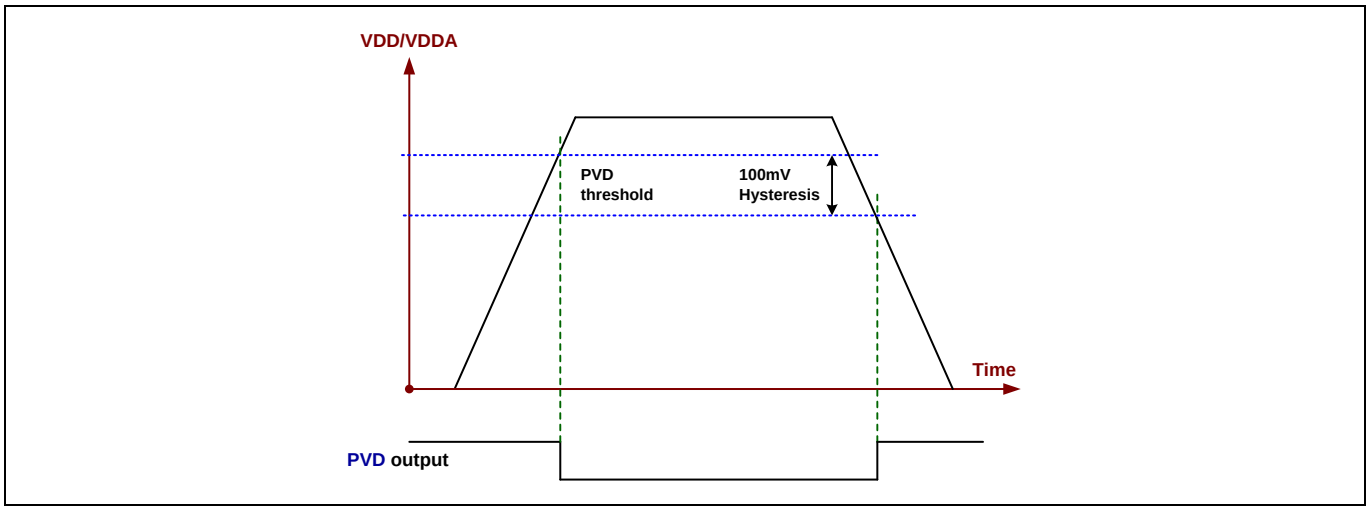
Figure 4-2 Power-on reset and power-down reset waveform



4.2.2 Programmable voltage detector

The PVD can be used by the user to monitor the power supply by comparing VDD to the PLS bits in the power control register (PWR_CR). These bits can select the threshold of the monitored voltage.
The PVD is enabled by setting the PVDE bit.
A PVDO flag is available, in the power control/status register (PWR_CSR), to indicate if VDD is higher or lower than the PVD voltage threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if the interrupt is enabled through the EXTI registers. The PVD interrupt can be generated when VDD drops below the PVD threshold or when VDD rises above the PVD threshold depending on EXTI line16 rising/falling edge configuration. As an example the service routine could perform emergency shutdown tasks.

Figure 4-3 PVD threshold waveform



4.3 Low-power modes

By default, the microcontroller is in Run mode after a system or a power reset. Several low-power modes are available to save power when the CPU does not need to be kept running, for example when waiting for an external event. It is up to the user to select the mode that gives the best compromise between low-power consumption, short startup time and available wakeup sources.
The chip features three low-power modes:

Sleep mode (CPU stops working, but all peripherals including CPU peripherals like NVIC, SysTick, etc. are kept running)

Stop mode (all clocks are stopped while the SRAM and register contents are kept intact)

Deep Stop mode, the core works in a lower power consumption mode, all clocks stop, and the contents of registers and SRAM are saved.

In addition, the power consumption in Run mode can be reduced by one of the following means:

Reduce the system clock frequency

Turn off the clocks on the APB and AHB buses when they are unused

Table 4-1 Low power modes

Mode	Entry mode	Wake-up mode	Influence on 1.5V domain clock	Influence on VDD domain clock	Voltage regulator
SLEEP(SLEEP NOW or SLEEP ON EXIT)	WFI (Wait for Interrupt)	Any interrupt	CPU clock off, no influence on other clock and ADC clock	off	On
	WFE (Wait for Event)	Wake-up event			
Stop	Clear LPDS bit; set SLEEPDEEP bit; WFI or WFE;	Any external interrupt (setup in external interrupt register)	All 1.5V domain clocks close	HSI off	On
DeepStop	Set LPDS bit; SLEEPDEEP bit; WFI or WFE;	Any external interrupt (setup in external interrupt register)			On

4.3.1 Run mode

Lower system clock

In Run mode, the speed of any system clock (SYSCLK, HCLK, PCLK1) can be reduced by programming the prescaler registers. These prescalers can also be used to reduce peripheral clocks before entering Sleep mode. For more details refer to: Clock configuration register (RCC_CFGR).

Peripheral clock gating

In Run mode, the HCLK and PCLKx for individual peripherals and memories can be stopped at any time to reduce power consumption. To further reduce power consumption in Sleep mode the peripheral clocks can be disabled prior to executing the WFI or WFE instructions.

Peripheral clock gating is controlled by the AHB peripheral clock enable register (RCC_AHBENR) and APB1 peripheral clock enable register (RCC_APB1ENR).

4.3.2 Sleep Mode

Entering Sleep mode

The Sleep mode is entered by executing the WFI or WFE instructions. Two options are available to select the Sleep mode entry mechanism, depending on the SLEEPONEXIT bit in the CPU 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 interrupt service routine.

In the Sleep mode, all I/O pins keep the same state as in the Run mode. Refer to and for details on how to enter Sleep mode.

Table 4-2 SLEEPNOW mode

SLEEP NOW mode	Description
Entry	Execute WFI (Wait for Interrupt) or WFE (Wait for Event) instructions under conditions below: SLEEPDEEP = 0 SLEEPONEXIT = 0
Exit	If executing WFI, enter Sleep Mode: interrupt (refer to interrupt vector table) If executing WFE, enter Sleep Mode: wake-up event (refer to wake-up event management)
Wake-up latency	Immediate wake-up

Table 4-3 SLEEPONEXIT mode

SLEEP ON EXIT mode	Description
Entry	Execute WFI (Wait for Interrupt) instructions under conditions below: SLEEPDEEP = 0 SLEEPONEXIT = 1
Exit	Interrupt (refer to interrupt vector table)
Wake-up latency	Immediate wake-up

4.3.3 Stop Mode

The Stop mode is based on the CPU deepsleep mode combined with peripheral clock gating. In Stop mode, the voltage regulator can be configured in normal mode, all clocks in the 1.5V domain are stopped, and the HSI and the HSE oscillators are disabled. SRAM and register contents are preserved. In Stop mode, all I/O pins keep the same state as in the Run mode.

Entering Stop mode

The following features can be selected by programming individual control bits:

Independent watchdog (IWDG): the IWDG is started by writing to its key register or by hardware option.

Internal oscillator (LSI oscillator): this is configured by the LSION bit in the control/status register (RCC_CSR).

The ADC can also consume power during the Stop mode, unless it is disabled before entering it. To disable it, the ADEN bit in the ADC_ADCFG register must be written to 0. Other unused GPIOs also consume power unless they are configured with analog inputs.

Exiting Stop mode

When exiting Stop mode by issuing an interrupt or a wakeup event, the HSI oscillator is selected as system clock. The clock frequency is the HSI divided by 6.

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

Refer to for details on how to exit the Stop mode.

Table 4-4 Stop mode

Stop mode	Description
Entry	The WFI (Wait for Interrupt) or WFE (Wait for Event) instructions are executed under the following conditions: The SLEEPDEEP bit in the CPU System Control Register is set; The LPDS bit in the Power Control Register (PWR_CR) is reset; The system clock is switched to LSI or HSI Note: To enter the Stop mode, all request bits for external interrupts (interrupt event pending registers EXTI_PR) must be cleared. Otherwise, the entry process of the Stop mode will be skipped, and the program will continue running.
Exit	The WFI (Wait for Interrupt) instruction is executed under the following conditions: Any external interrupt line is set to interrupt mode (the corresponding external interrupt vector must be enabled in NVIC). See the interrupt vector table for Wait for Event; The WFE (Wait for Event) instruction is executed under the following conditions: Any external interrupt line is set to event mode, such as a watchdog interrupt;
Wake-up latency	The wakeup time for LSI or HSI and the additional startup delay caused by the voltage regulator wakeup

4.3.4 DeepStop Mode

Deep Stop is a low power consumption mode combining the clock control and voltage regulator control mechanism of peripherals on the basis of CPU deep sleep mode. In the deep stop mode, all clocks in the 1.5V domain are stopped, HSI functions are disabled, and SRAM and register contents are retained. In the deep stop mode, all I/O pins remain in the state of run mode.

4.3.4.1 Entering Deep Stop mode

In Deep Stop mode, the following features can be selected by setting individual control bits:

Independent watchdog (IWDG): the IWDG is started by writing to its key register or by hardware option.

Internal oscillator (LSI oscillator): this is configured by the LSION bit in the control/status register (RCC_CSR).

4.3.4.2 Exiting Deep Stop mode

When an interrupt or wake-up event causes the exit of the deep stop mode, the HSI oscillator is selected as the system clock. The clock frequency is 6 division of HSI. When the voltage regulator is in the normal power consumption mode and the system exits from the deep stop mode, there will be an additional start delay.

Refer to for details on how to enter/exit the Deep Stop mode.

Table 4-5 Deep Stop mode

Deep Stop Mode	Description
Entry	The WFI (Wait for Interrupt) or WFE (Wait for Event) instructions are executed under the following conditions: The SLEEPDEEP bit in the CPU System Control Register is set; The LPDS bit in the Power Control Register (PWR_CR) is set;
Exit	Any external interrupt or wake-up event.
Wake-up latency	The wakeup time for LSI or HSI and the additional startup delay caused by the voltage regulator wakeup

4.3.4.3 Debug mode

By default, the debug connection is lost if the application puts the MCU in Stop or Standby mode while the debug features are used. This is due to the fact that the CPU core is no longer clocked.
However, by setting some configuration bits in the DBGMCU_CR register, the software can be debugged even when using the low-power modes extensively. For more details, refer to debug support for low-power modes.

4.4 Power control register

Table 4-6 Power control register overview

Offset	Acronym	Register Name	Reset
0x00	PWR_CR	Power control register	0x00000000
0x04	PWR_CSR	Power control status register	0x00000000
0x24	PWR_CFGR	Power configuration register	0x00000040
0x30	PWR_MEMCR	Power memory control register	0x00000000

4.4.1 PWR_CR Power Control Register

Address offset: 0x00
Reset value: 0x00000000

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		PLS			Res		Res		PVDE		Res		LPDS
				rw							rw				rw

Bit	Field	Description
31:13	Reserved	Reserved, always read as 0
12：9	PLS	PVD level selection These bits are used to select the voltage threshold of the power voltage detector. 0000：1.8V 0100：3.0V 1000：4.2V 0001：2.1V 0101：3.3V 1001：4.5V 0010：2.4V 0110：3.6V 1010：4.8V 0011：2.7V 0111：3.9V Others: Reserved Note: Refer to the electrical characteristics section in the datasheet for detailed information.
8	Reserved	Reserved, always read as 0
7:5	Reserved	Reserved, always read as 0
4	PVDE	Power voltage detector enable 1 = PVD enabled 0 = PVD disabled
3:1	Reserved	Reserved, always read as 0
0	LPDS	Low Power DeepStop: 1: When entering Stop Mode, the voltage regulator operates in low-power mode. 0: When entering Stop Mode, the voltage regulator operates in normal power mode. When entering Stop Mode, the current with LPDS = 1 is lower than the current with LPDS = 0. Refer to the corresponding datasheet of this chip for more details.

4.4.2 PWR_CSR Power Control/Status Register

Address offset: 0x04
Reset value: 0x00000000 (not cleared when wake up from the Standby Mode)

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res													PVDO	Res	
													r		

Bit	Field	Description
31:3	Reserved	Reserved, always read as 0
2	PVDO	PVD output This bit is valid when PVD is enabled by the PVDE bit. 1: VDD/VDDA is lower than the PVD threshold selected by PLS[3:0] 0: VDD/VDDA is higher than the PVD threshold selected by PLS[3:0] Note: In Standby mode, the PVD is disabled. Therefore, after Standby mode or a reset, this bit is 0 until the PVDE bit is set.
1:0	Reserved	Reserved, always read as 0

5. RCC Clock and Reset

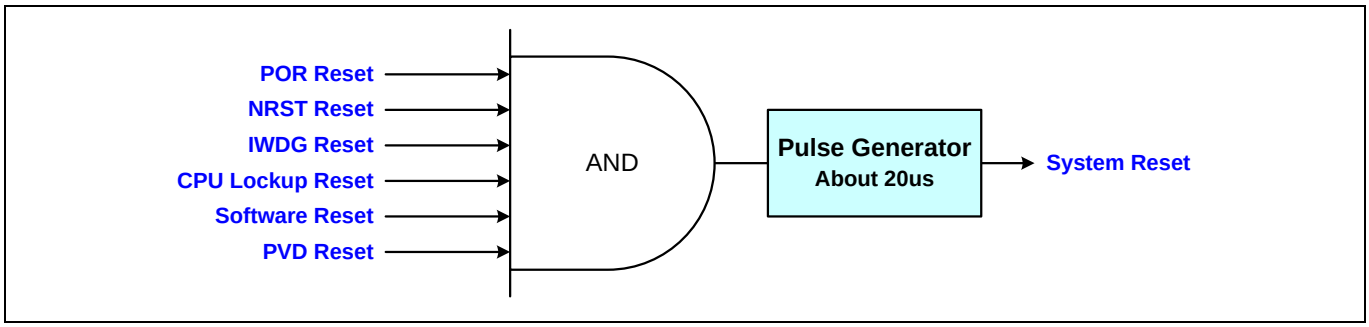
5.1 Reset unit

5.1.1 Overview

There are two types of resets: power reset, system reset.

5.1.2 Functional block diagram

Figure 5-1 Reset functional block diagram



5.1.3 Main characteristics

Reset event judgment: Judge through the reset flag bit of the control status register (RCC_CSR).
Power on reset: Reset all registers.
System reset: Except that the reset flag and internal low-speed oscillator enable flag in clock control register (RCC_CSR), standby and wake-up flag in power control register (PWR_CSR), DBG control register (DBG_CR) and register in the backup domain are not affected by system reset, other registers are reset by the system.

5.1.4 Functional description

5.1.4.1 POR reset

Power reset includes:

Power-on Reset
Power-down Reset

5.1.4.2 System reset

System reset includes:

NRST Reset
IWDG Reset
Software Reset
CPU Lockup Reset
PVD Reset

5.1.4.3 NRST Reset:

When input the low level through NRST Pin, the NRST reset will produce.

5.1.4.4 IWDG Reset:

The counter begins ramp down from its reset value 0x0FFF. When it ramps down to 0x0000, the independent watchdog reset will produce.
If the program is anomaly, the feed will not be normal, and the independent watchdog reset will produce.
Refer to the chapter of window watchdog for details

5.1.4.5 Software Reset:

Set SCB_AIRCR [SYSRESETREQ] as 1, and software reset will produce.

5.1.4.6 CPU Lockup Reset:

Configure the LOCKUPEN bit of control status register (RCC_CSR) as 1, and enable CPU lockup reset; CPU lockup reset will produce when CPU enters the lockup status.

5.1.4.7 PVD Reset:

Configure the PVDRSTEN bit of control status register (RCC_CSR) as 1, enable PVD reset;
Configure the PVDE bit of power control register 1 (PWR_CR1) as 1, enable PVD;
Configure the PLS bit of power control register 1 (PWR_CR1) and select the PVD threshold;
Detect VDD power. When the VDD power is lower than the selected threshold voltage, PVD reset will produce.

5.2 Clock unit

5.2.1 Overview

Four configurable independent clocks:

High-speed external clock (HSE)

High-speed internal clock divided by 6 (HSIDIV)

High-speed internal clock (HSI)

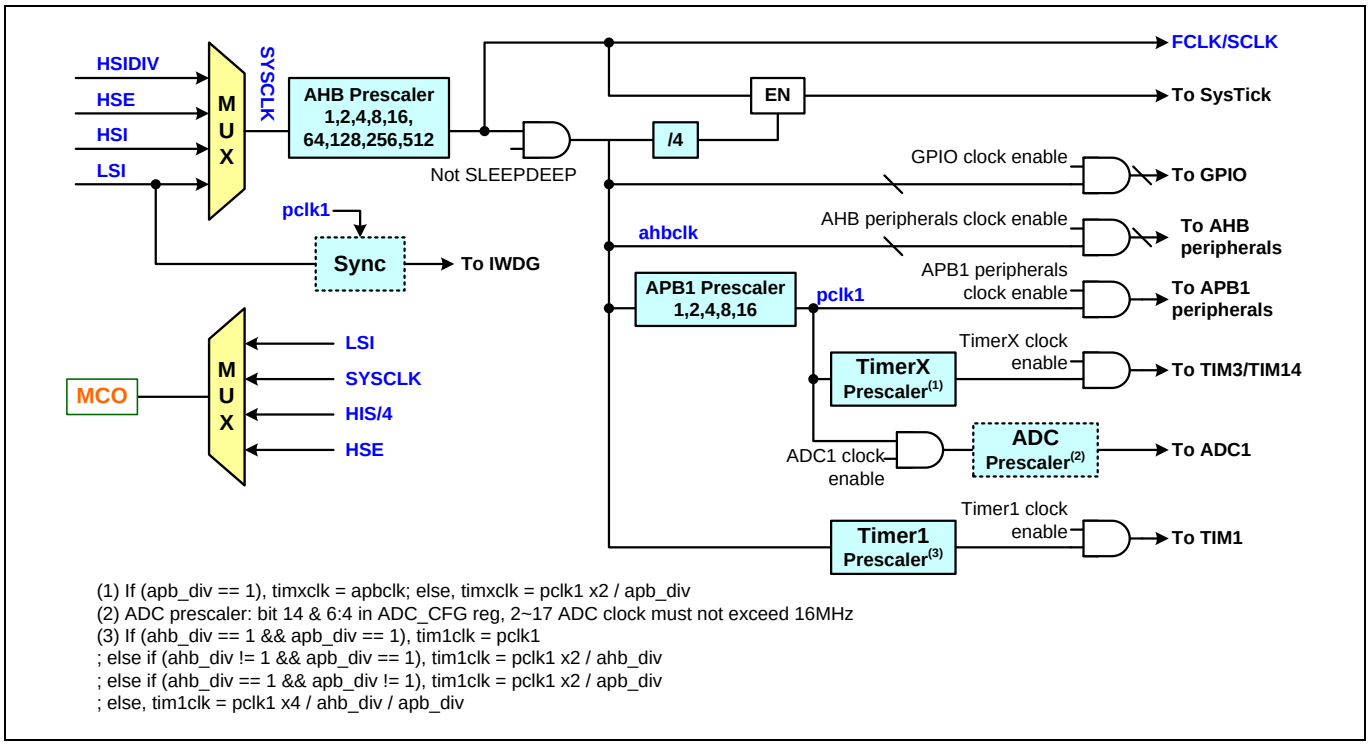
Low-speed internal clock (LSI)

One configurable independent peripheral clock sources:

Low-speed external clock (LSE)

5.2.2 Functional block diagram

Figure 5-2 Clock tree



(1) If (apb_div == 1), timxclk = apbclk; else, timxclk = pclk1 x2 / apb_div
(2) ADC prescaler: bit 14 & 6:4 in ADC_CFG reg, 2~17 ADC clock must not exceed 16MHz
(3) If (ahb_div == 1 && apb_div == 1), tim1clk = pclk1
; else if (ahb_div != 1 && apb_div == 1), tim1clk = pclk1 x2 / ahb_div
; else if (ahb_div == 1 && apb_div != 1), tim1clk = pclk1 x2 / apb_div
; else, tim1clk = pclk1 x4 / ahb_div / apb_div

5.2.3 Main characteristics

Respectively configure the clock frequency of AHB and APB1 bus via the prescaler control bit of the clock register (RCC_CFGR). The maximum frequency of AHB and APB1 bus clock is 48MHz.

5.2.4 Functional description

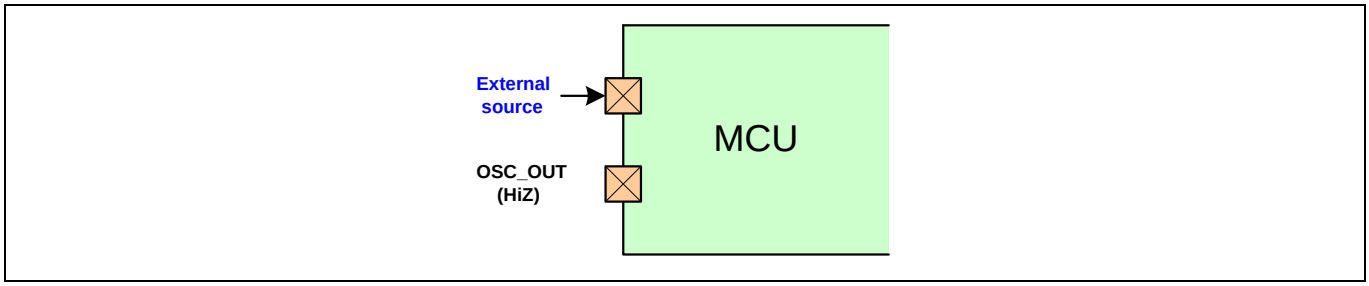
5.2.4.1 External high-speed clock (HSE)

One variety of high-speed external clock source:

External high-speed input speed

This clock input mode is selected by configuring the HSEON bit of the clock control register (RCC_CR).

Figure 5-3 External high-speed input speed



The figure above is the module block diagram of high-speed external input clock

The high-speed external clock has characteristics below:

External high-speed clock source or crystal oscillator must be provided

Input frequency range 4~24MHz

External high-speed clock signal (square wave, sine wave) with 50% duty cycle. Please refer to the chapter of electric characteristics of the data manual for details

When HSE is enabled, OSC_IN pin must be driven while OSC_OUT pin should remain floating when not being used.

Precaution:

Once HSE is enabled, relevant configuration of HSE cannot be changed. When the configuration needs to be changed, it's mandatory to disable HSE first.

The process to open the HSE configuration:

Configure the HSEON bit of clock control register (RCC_CR) as 1, and enable HSE;

Set the HSERDY bit of clock control register (RCC_CR) as 1. It means that HSE is ready, and valid clock signal will be output. At this time, it may be selected as the system clock or peripheral clock source.

5.2.4.2 High-speed internal clock (HSI)

HSI clock signal is produced by the internal oscillator. HSI clock source is opened by default when the chip is power on.

Steps to enable HSI configuration:

Configure the HSION bit of clock control register (RCC_CR) as 1, and enable HSI;

Set the HSIRDY bit of clock control register (RCC_CR) as 1. It means that HSI is ready, and valid clock signal will be output. At this time, it may be selected as the system clock or peripheral clock source.

Precaution:

Once HSI is enabled, relevant configuration of HSI cannot be changed. When the configuration needs to be changed, it's mandatory to disable HSI first.

5.2.4.3 Low-speed internal clock (LSI)

LSI acts as the low power clock source, and it provides the clock source for the independent watchdog. The clock center frequency is around 40 kHz. Refer to the chapter of electric characteristics of data manual.

Steps to enable LSI configuration:

Set the LSION bit of control status register (RCC_CSR) as 1, enable LSI;

Set the LSIRDY bit of control status register (RCC_CSR) as 1. It means that LSI is ready, and valid clock can be exported.

Precaution:

Once LSI is enabled, relevant LSI configuration cannot be changed. If it's required to change configuration, disable LSI first.

5.2.4.4 Interrupt

Table 5-1 RCC global interrupt

Interrupt event	Event flag bit	Enable control bit	Flag clear bit
RCC_HSERDY	HSERDYF	HSERDYIE	HSERDYC
RCC_HSIRDY	HSIRDYF	HSIRDYIE	HSIRDYC
RCC_LSIRDY	LSIRDYF	LSIRDYIE	LSIRDYC

Note: The above flag bit, control bit and clear bit can be configured by the clock interrupt register (RCC_CIR).

5.2.4.5 System clock selection (SWS)

Four system clock sources:

High-speed internal clock divided by 6 (default after power on)
High-speed internal clock (HSI)
High-speed external clock (HSE)
Low-speed internal clock (LSI)

System clock configuration steps:

Enable the required system clock source (HSI divided by 6, HSI, HSE, LSI). Each clock is enabled differently. For specific method, please view (chapter of HSI, HSE, LSI);
Set the bit of the selected clock source RDY signal as 1. It means that the system clock source is ready (when the target clock source is ready, the system clock may change over);
Select the system clock by configuring the SW bit of clock configuration register (RCC_CFGR);
Read the SWS bit of clock configuration register (RCC_CFGR), and judge the current system clock source.

5.2.4.6 System clock frequency switch

The system clock frequency switches from low speed to high speed or from high speed to low speed. It's recommended to switch the medium speed frequency for transition. The interval switch time from high speed to low speed is 1us at least.

5.2.4.7 Peripheral reset

Achieve the corresponding peripheral software reset by APB1 peripheral reset register (RCC_APB1RSTR) and AHB peripheral reset register (RCC_AHBRSTR).

5.2.4.8 Microcontroller clock output (MCO)

The microcontroller clock output (MCO) allows the clock to export to the external MCO pin. The configuration register of the corresponding GPIO port must be configured as the multiplex output function. Select any one of the 5 clock signals below as MCO output clock:

Table 5-2 Correlation of MCO and clock source

MCO bit of clock configuration register (RCC_CFGR)	Clock source
00x	No clock output
010	LSI
100	SYSCLK
101	HSI/4
110	HSE

5.2.4.9 Independent watchdog clock

When the independent watchdog is enabled by hardware, LSI oscillator will be automatically opened, and cannot be closed;
When the independent watchdog is enabled by software, LSI oscillator needs to be enabled and opened by software. When LSI oscillator is ready, the clock is supplied to IWDG, and LSI can be closed by software.

5.3 Register description

5.3.1 Register overview

Table 5-3 RCC register overview

Offset	Acronym	Register Name	Reset
0x00	RCC_CR	Clock Control Register	0x00000001
0x04	RCC_CFGR	Clock Configuration Register	0x00000000
0x08	RCC_CIR	Clock Interrupt Register	0x00000000
0x10	RCC_APB1RSTR	APB1 Peripheral Reset Register	0x00000000
0x14	RCC_AHBENR	AHB Peripheral Clock Enable Register	0x00000014
0x1C	RCC_APB1ENR	APB1 Peripheral Clock Enable Register	0x00000000
0x24	RCC_CSR	Control Status Register	0x08000000
0x28	RCC_AHBRSTR	AHB Peripheral Reset Register	0x00000000
0x40	RCC_SYSCFG	System Configuration Register	0x00000003

5.3.2 RCC_CR Clock Control Register

Address offest: 0x00
Reset value: 0x0000 0001
Access: No wait state, word, half word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														HSERDY	HSEON
														r	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved														HSIRDY	HSION
														r	rw

Bit	Field	Description
31：18	Reserved	Reserved, must be kept at reset value.
17	HSERDY	External High-speed Clock Ready Flag Set by hardware. 0: high-speed external crystal oscillator is not ready 1: high-speed external crystal oscillator is ready
16	HSEON	External High-speed Clock Enable Set '1'or clear '0' by software. When entering the standby or stop mode, the bit is cleared '0' by hardware. When HSE has been or will be selected as the clock source by system clock, this bit is forbidden to be reset. 0: Disable high-speed external crystal oscillator 1: Enable high-speed external crystal oscillator
15：2	Reserved	Reserved, must be kept at reset value.
1	HSIRDY	Internal High-speed Clock Ready Flag Set ‘1’ by hardware. It means that the internal clock is ready. When HSION bit is cleared, HSIRDY turns "0" after 3 AHB clock periods. 0: Internal high-speed clock is not ready 1: Internal high-speed clock is ready
0	HSION	Internal High-speed Clock Enable Set "1"or clear "0" by software. When leaving standby or stop mode or external oscillator used as system clock, produce fault, and the bit is set as "1"by hardware to force opening of internal oscillator. When the system clock has used or will use HSI as the clock source, disable resetting the bit.

		0: Disable internal high-speed clock 1: Enable internal high-speed clock

5.3.3 RCC_CFGR Clock Configuration Register

Address offest: 0x04
Reset value: 0x0000 0000
Access: No wait state, word, half word and byte access
Only when access occurs during clock change over, insert 1 or 2 wait state.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.					MCO			Res.							
					rw										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					PPRE1			HPRE				SWS		SW	
					rw			rw				r		rw	

Bit	Field	Description
31：27	Reserved	Reserved, must be kept at reset value.
26：24	MCO	Micro Controller Clock Output Set '1' or clear '0' by software 00x: No clock output 010: LSI clock output 100: System clock (SYSCLK) 8 division output 101: HSI 4 division output 110: HSE clock output Other: No clock output Note: This clock output may be stopped when starting or switching MCO clock source. When system clock is exported to MCO pin, please guarantee that the output clock frequency is no more than 50MHz
23：11	Reserved	Reserved, must be kept at reset value.
10：8	PPRE1	APB1 Prescaler Coefficient Set the software to control the APB1 clock (PCLK2) prescaler factor. 0xx: HCLK no division 100: HCLK 2 division 101: HCLK 4 division 110: HCLK 8 division 111: HCLK 16 division
7：4	HPRE	AHB Prescaler Coefficient Set the software to control the APB clock prescaler factor. 0xxx: SYSCLK no division 1000: SYSCLK 2 division 1001: SYSCLK 4 division 1010: SYSCLK 8 division 1011: SYSCLK 16 division 1100: SYSCLK 64 division 1101: SYSCLK 128 division 1110: SYSCLK 256 division 1111: SYSCLK 512 division Note: When the prescaler coefficient of AHB clock is greater than 1, it's mandatory to enable prefetch buffer. See the chapter of flash memory for details.
3：2	SWS	System Clock Switch Status 00: Select HSI divided by 6 output as system clock 01: Select HSE output as system clock 10: Select HSI output as system clock 11: Select LSI output as system clock
1：0	SW	System Clock Switch Select system clock source by software configuration When returning from stop or standby mode or directly or indirectly as system clock HSE produces fault, the hardware will enforce selection of HSI as system clock. 00: Select HSI divided by 6 output as system clock 01: Select HSE output as system clock 10: Select HSI output as system clock 11: Select LSI output as system clock

5.3.4 RCC_CIR Clock Interrupt Register

Address offest: 0x08
Reset value: 0x0000 0000
Access: No wait period, word, half word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.												HSERDYC	HSIRDYC	Res.	LSIRDYC
												w1c	w1c		w1c
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.				HSERDYIE	HSIRDYIE	Res.	LSIRDYIE	Res.				HSERDYF	HSIRDYF	Res.	LSIRDYF
				rw	rw		rw					r	r		r

Bit	Field	Description
31：20	Reserved	Reserved, must be kept at reset value.
19	HSERDYC	HSE Ready Interrupt Clear Set ‘1’ by software to clear HSE ready interrupt flag bit HSERDYF. 0: Invalid 1: Clear HSE ready interrupt flag bit HSERDYF
18	HSIRDYC	HSI Ready Interrupt Clear Set ‘1’ by software to clear HSI ready interrupt flag bit HSIRDYF. 0: Invalid 1: Clear HSI ready interrupt flag bit HSIRDYF
17	Reserved	Reserved, must be kept at reset value.
16	LSIRDYC	LSI Ready Interrupt Clear Set ‘1’ by software to clear LSI ready interrupt flag bit LSIRDYF 0: Invalid 1: Clear LSI ready interrupt flag bit LSIRDYF
15：12	Reserved	Reserved, must be kept at reset value.
11	HSERDYIE	HSE Ready Interrupt Enable Set ‘1’ by software to enable or clear "0" to disable external oscillator ready interrupt. 0: Disable HSE ready interrupt 1: Enable HSE ready interrupt
10	HSIRDYIE	HSI Ready Interrupt Enable Set ‘1’ by software to enable or clear "0" to disable external oscillator ready interrupt. 0: Disable HSI ready interrupt 1: Enable HSI ready interrupt
9	Reserved	Reserved, must be kept at reset value.
8	LSIRDYIE	LSI Ready Interrupt Enable Set ‘1’ by software to enable or clear "0" to disable internal 40KHz oscillator ready interrupt. 0: Disable LSI ready interrupt 1: Enable LSI ready interrupt
7：4	Reserved	Reserved, must be kept at reset value.
3	HSERDYF	HSE Ready Interrupt Flag When the high-speed external clock is ready, set ‘1’ by hardware. Clear by setting HSERDYC bit as "1"by software. 0: Produce clock ready interrupt without external oscillator 1: Clock ready interrupt caused by external oscillator
2	HSIRDYF	HSI Ready Interrupt Flag When the high-speed internal clock is ready, set ‘1’ by hardware. Clear by setting HSIRDYC bit as "1"by software. 0: Produce clock ready interrupt without internal HSI oscillator 1: Clock ready interrupt caused by internal HSI oscillator
1	Reserved	Reserved, must be kept at reset value.
0	LSIRDYF	LSI Ready Interrupt Flag When the low-speed internal clock is ready, set ‘1’ by hardware. Clear by setting LSIRDYC bit as "1"by software. 0: Produce clock ready interrupt without internal 40KHz oscillator 1: Clock ready interrupt caused by internal 40KHz oscillator

5.3.5 RCC_APB1RSTR APB1 Peripheral Reset Register

Address offest: 0x10
Reset value: 0x0000 0000
Access: No wait period, word, half word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SYSCFG	DBG	PWR	Res.						I2C1	Res.			USART2	USART1
	rw	rw	rw							rw				rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			SPI1	Res.		ADC1	Res.					TIM14	TIM1	TIM3	Res.
			rw			rw						rw	rw	rw	

Bit	Field	Description
31	Reserved	Reserved, must be kept at reset value.
30	SYSCFG	SYSCFG Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
29	DBG	DBG Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
28	PWR	Power Interface Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
27：22	Reserved	Reserved, must be kept at reset value.
21	I2C1	I2C1 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
20：18	Reserved	Reserved, must be kept at reset value.
17	USART2	USART2 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
16	USART1	USART1 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
15：13	Reserved	Reserved, must be kept at reset value.
12	SPI1	SPI1 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
11：10	Reserved	Reserved, must be kept at reset value.
9	ADC1	ADC1 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
8：4	Reserved	Reserved, must be kept at reset value.
3	TIM14	TIM14 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
2	TIM1	TIM1 Reset Set to "1" or clear "0" by software 0: Invalid 1: Reset
1	TIM3	TIM3 Reset

		Set to "1" or clear "0" by software 0: Invalid 1: Reset
0	Reserved	Reserved, must be kept at reset value.

5.3.6 RCC_AHBENR AHB Peripheral Clock Enable Register

Address offest: 0x14
Reset value: 0x0000 0014
Access: No wait period, word, half word and byte access

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

Bit	Field	Description
31：19	Reserved	Reserved, must be kept at reset value.
18	GPIOB	GPIOB Clock Enable Set "1" or clear "0" by software 0: Clock disable 1: Clock enable
17	GPIOA	GPIOA Clock Enable Set "1" or clear "0" by software 0: Clock disable 1: Clock enable
16：7	Reserved	Reserved, must be kept at reset value.
6	CRC	CRC Clock Enable Set "1" or clear "0" by software 0: Clock disable 1: Clock enable
5	Reserved	Reserved, must be kept at reset value.
4	Flash	FLASH Clock Enable Set "1" or clear "0" by software 0: Clock disable 1: Clock enable
3	Reserved	Reserved, must be kept at reset value.
2	SRAM	SRAM Clock Enable Set "1" or clear "0" by software 0: Clock disable 1: Clock enable
1：0	Reserved	Reserved, must be kept at reset value.

5.3.7 RCC_APB1ENR APB1 Peripheral Clock Enable Register

Address offest: 0x1C
Reset value: 0x0000 0000
Access: No wait period, word, half word and byte access
Note: When peripheral clock is not enabled, the software cannot read the value of peripheral register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	SYSCFG	DBG	PWR	Res.						I2C1	Res.			USART2	USART1
	rw	rw	rw							rw				rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			SPI1	Res.		ADC1	Res.					TIM14	TIM1	TIM3	Res.
			rw			rw						rw	rw	rw	

Bit	Field	Description
31	Reserved	Reserved, must be kept at reset value.
30	SYSCFG	SYSCFG Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
29	DBG	DBG Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
28	PWR	Power Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
27：22	Reserved	Reserved, must be kept at reset value.
21	I2C1	I2C1 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
20：18	Reserved	Reserved, must be kept at reset value.
17	USART2	USART2 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
16	USART1	USART1 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
15：13	Reserved	Reserved, must be kept at reset value.
12	SPI1	SPI1 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
11：10	Reserved	Reserved, must be kept at reset value.
9	ADC1	ADC1 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
8：4	Reserved	Reserved, must be kept at reset value.
3	TIM14	TIM14 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
2	TIM1	TIM1 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable 1: Clock enable
1	TIM3	TIM3 Clock Enable Set to "1" or clear to "0" by software 0: Clock disable

		1: Clock enable
0	Reserved	Reserved, must be kept at reset value.

5.3.8 RCC_CSR Control Status Register

Address offest: 0x24
Reset value: 0x0800 0000
Access: 0-3 wait state, word, half word and byte access
In case of continuous access to the register, insert the wait state.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.		IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	Res.	RMVF	LOCKUPF	PVDRSTF	Res.					
		r	r	r	r		w1c	r	r						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.								LOCKUPEN	PVDRSTEN	Res.				LSIRDY	LSION
								rw	rw					r	rw

Bit	Field	Description
31：30	Reserved	Reserved, must be kept at reset value.
29	IWDGRSTF	Independent Watchdog Reset Flag In case of independent watchdog reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software. 0: No independent watchdog reset 1: Independent watchdog reset
28	SFTRSTF	Software Reset Flag In case of software reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software. 0: No software reset 1: Software reset
27	PORRSTF	POR/PDR Reset Flag In case of POR/PDR reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software. 0: No POR/PDR reset 1: POR/PDR reset
26	PINRSTF	NRST PIN Reset Flag In case of NRST PIN reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software 0: No NRST PIN reset 1: NRST PIN reset
25	Reserved	Reserved, must be kept at reset value.
24	RMVF	Remove Reset Flag Clear the reset flag by setting '1'by software. 0: Invalid 1: Clear reset flag
23	LOCKUPF	CPU Lockup Reset Flag In case of CPU lockup reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software 0: No CPU lockup reset 1: CPU lockup reset
22	PVDRSTF	PVD Reset Flag In case of PVD reset, set'1' by hardware, and enable clear by power reset or write RMVF bit to clear by software 0: No PVD reset 1: PVD reset
21：8	Reserved	Reserved, must be kept at reset value.
7	LOCKUPEN	CPU Lockup Reset Enable 0: Disable CPU lockup reset 1: Enable CPU lockup reset
6	PVDRSTEN	PVD Reset Enable 0: Disable PVD reset 1: Enable PVD reset

5：2	Reserved	Reserved, must be kept at reset value.
1	LSIRDY	Internal Low-speed Oscillator Ready Setting '1'or clearing '0' by hardware indicates whether the internal 40KHz oscillator is ready. When LSION is cleared, LSIRDY is cleared after 3 AHB clocks. 0: Internal 40KHz oscillator clock not ready 1: Internal 40KHz oscillator clock ready
0	LSION	Internal Low-speed Oscillator Enable Setting '1'or clearing '0' by software, or clear by power reset 0: Disable internal 40KHz oscillator 1: Enable internal 40KHz oscillator

5.3.9 RCC_AHBRSTR AHB Peripheral Reset Register

Address offest: 0x28
Reset value: 0x0000 0000
Access: No wait period, word, half word and byte access

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

Bit	Field	Description
31：19	Reserved	Reserved, must be kept at reset value.
18	GPIOB	GPIOB Reset Set "1" or clear "0" by software 0: Invalid 1: Reset
17	GPIOA	GPIOA Reset Set "1" or clear "0" by software 0: Invalid 1: Reset
16：0	Reserved	Reserved, must be kept at reset value.

5.3.10 RCC_SYSCFG System Configuration Register

Address offest: 0x40
Reset value: 0x0000 0003
Access: 0-3 wait state, word, half-word and byte access

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.													SFT_NRST_RMP	SECTOR1KCFG	PROG_CHECK_EN
													rw	rw	r

Bit	Field	Description
31：3	Reserved	Reserved, must be kept at reset value.
2	SFT_NRST_RMP	SFT_NRST_RMP: software mapping of nRST. This bit is only cleared when the power is reset or when it is set to ‘0’by software. 1: PA14 mapped to nRST 0: no action Note: The SFT_NRST_RMP bit of RCC_SYSCFG is set to‘1’, PA14 is mapped to the external reset of nRST, and the low level is at least 4us.
1	SECTOR_1K_CFG	SECTOR_1K_CFG: size for Flash erase. 1: 1K bytes 0: 512 bytes
0	PROG_CHECK_EN	Check whether the data in Flash is 0xFF when writing Flash 1: Check (hardware is fixed to 1) 0: Not check

6. GPIO General-Purpose I/Os

6.1 Overview

Each general-purpose IO (GPIO) port, can be individually configured by two 32-bit control registers (GPIOx_CRL/GPIOx_CRH) and two 32-bit multiplexed control registers (GPIOx_AFRL, GPIOx_AFRH) in eight modes: analog input、input floating、input pull-up、input pull-down、output push-pull、output open-drain、alternate function push-pull and alternate function open-drain.
Each I/O port bit is freely programmable. All registers can be accessed as 32-bit (words), 16-bit (half-words) or 8-bit (bytes). The GPIOx_BSRR and GPIOx_BRR control registers in the GPIO register group are able to modify the GPIOx_ODR bit to output 0 or 1 independently through write accesses.

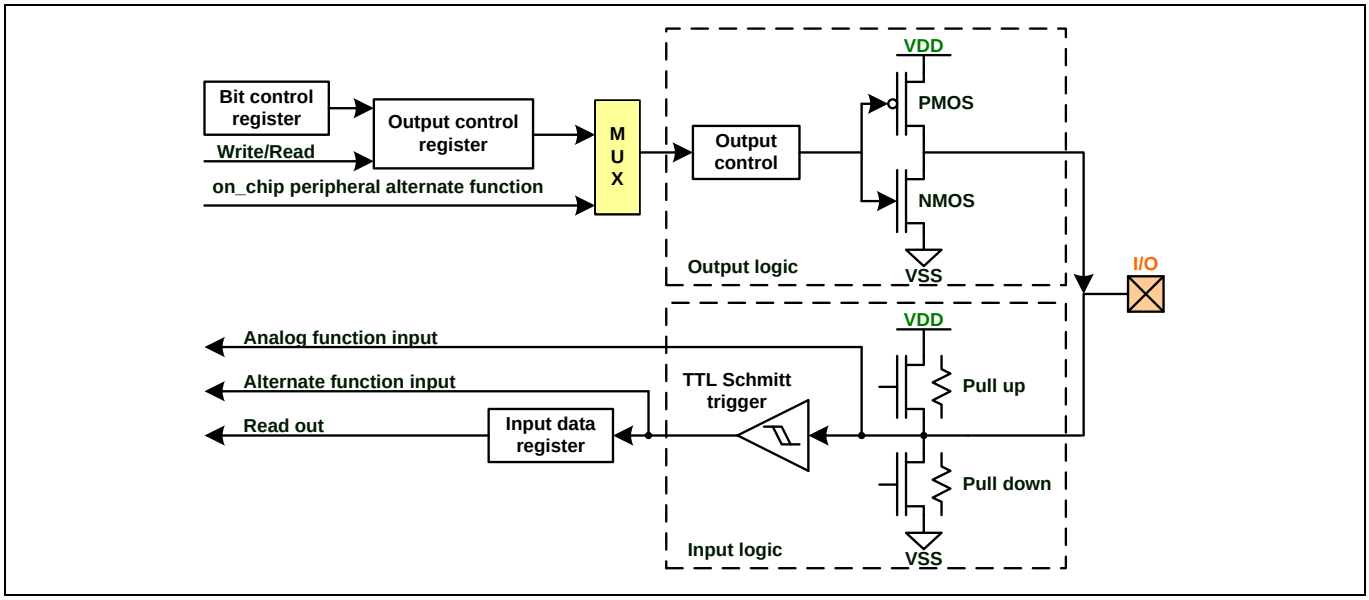
6.2 Main characteristics

A single AHB write access changes one or multiple bits for GPIOx_ODR
All I/Os support programmable EXTI configuration register to output externally triggered interrupts
Support GPIO locking mechanism
Supported input states: floating, pull-up/down, analog
Supported output states: push-pull or open drain with pull-up/down
Input floating as default, configurable input/output direction
Configurable input/output speed

6.3 Functional description

6.3.1 Functional block diagram

Figure 6-1 Standard I/O port



6.3.2 GPIO port configuration

Table 6-1 Port bit configuration table（take port0 as an example）

Pin_mode		Pull up/down	DCR[1:0]		CNF0		MODE0	ODRx
Analog input		x	x	x	0	0	00	
General input Alternate input		Floating	x	x	0	1		
		Pull-up	x	x	1	0		
		Pull-down	x	x	1	0		0
General output	Push-pull	x	x	x	0	0	01 10 11	0 or 1
	Open-drain	Floating	x	0	0	1		0 or 1
		Pull-up	1	1	0	1		0 or 1

		Pull-down	0	1	0	1		0 or 1
Alternate output	Push-pull	x	x	x	1	0		x
	Open-drain	Floating	x	0	1	1		x
		Pull-up	1	1	1	1		x
		Pull-down	0	1	1	1		x

Note: x represents “don’t care” for I/Os in the corresponding mode. ODR0 represents the bit 0 of the output data register.

Input and output follow these configurations:

General-purpose input:

The user should configure the CNF0 in the GPIOx_CRL to choose the Input mode.

General-purpose output:

Push-Pull output: The user configures MODE0 to choose the output speed and sets CNF0=00;

Open-Drain output: The user configures MODE0 to choose the output speed and sets CNF0=01. To activate the pull-up/pull-down feature for pins, the GPIOx_DCR register should be configured individually. This feature is invalid if it is not an output open-drain mode.

Alternate function:

Configure the AFRLx [3:0] and AFRHx [3:0] registers to choose the alternate function:

Alternate push-pull output: The user configures MODE0 to choose the output speed and sets CNF0=10;

Alternate open-drain output: The user configures MODE0 to choose the output speed and sets CNF0=11.

To activate the pull-up/pull-down feature for IO in the output mode, the GPIOx_DCR register should be configured individually. This feature is invalid if it is not an output open-drain mode.

During and just after reset, the GPIO ports are configured in Input Floating mode. The Serial-Wired Debug pins default to input pull-up (PU)/pull-down (PD).

When configured in general purpose output mode, the value of the output data register (GPIOx_ODR) is output on the corresponding I/O pin. The Input Data register (GPIOX_IDR) captures the data present on the I/O pin at every AHB clock cycle.

Note: Not all chips contain the JTAG and SWD ports. For specific chip configuration, refer to the chip datasheet.

PA14: SWCLK in PD

PA13: SWDIO in PU

6.3.3 Alternate function

The corresponding IO alternate functions are enabled by setting the alternate function register.

When an IO is configured as Alternate Function Input, the port should choose pull-up, pull-down or floating input.

When an IO is configured as Alternate Function Output, the port should choose output push-pull or open-drain mode.

When an IO is configured with bidirectional alternate function, the port should choose output push-pull or open-drain mode. In this case the input is configured in input floating mode. In the open-drain mode, the GPIOx_DCR register is configured to choose weak pull-up or weak pull-down resistor.

If a port bit is configured as Alternate Function Output, this connects the port to the output signal of an on-chip peripheral. If a GPIO pin is configured as Alternate Function Output solely via software, but peripheral is not activated, its output is not specified.

6.3.4 GPIO locking mechanism

It is possible to freeze the IO configuration by applying the GPIO locking mechanism. When the LOCK mechanism has been applied on a port, it is no longer possible to modify the port configuration until the next reset. The write sequence of the LOCK key:

GPIOx_LCKR [16] =’1’+LCKR [15:0].

GPIOx_LCKR [16] =’0’+LCKR [15:0].

GPIOx_LCKR [16] =’1’+LCKR [15:0].

To lock the PA [0] port of GPIOA, follow these configurations:

GPIOA->GPIOA_LCKR=0x10001.

GPIOA->GPIOA_LCKR=0x00001.

GPIOA->GPIOA_LCKR=0x10001.

After three steps above are finished, the bit 16 of the GPIOA_LCKR register is set. The GPIOA_LCKR register can no longer be written until the next soft reset. The bit 16 of the GPIOA_LCKR register remains 1 and PA [0] keeps its configuration before the locking.

When the port is locked, the value of the port bit can no longer be modified until the soft reset. Each lock bit in the GPIOx_LCKR register freezes 4 bits in the port configuration registers (GPIOx_CRL and GPIOx_CRH).

Notes:

Only the value of PA [0] is locked due to the proceeding configuration. It is still possible to configure the values of PA [15:1] and other GPIO control registers.

6.3.5 Input configuration

When the I/O port is program as Input:

The Schmitt Trigger Input is activated.

The Output Buffer is disabled.

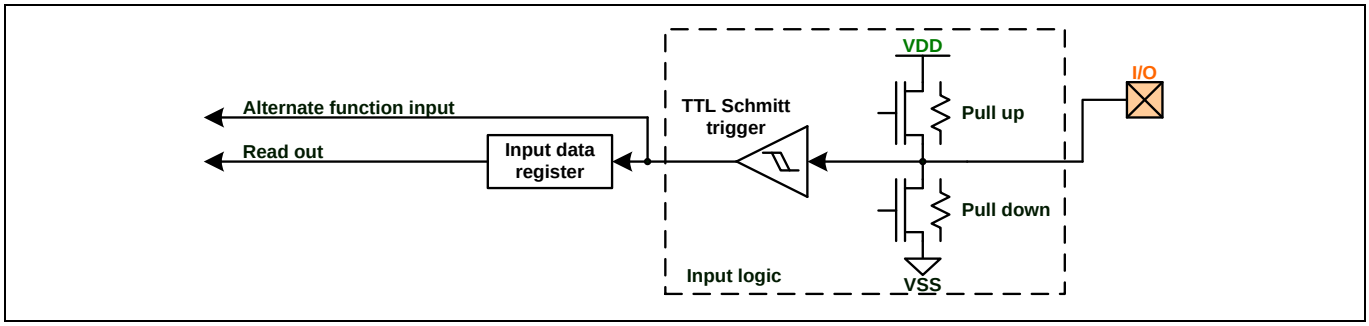
Floating, pull-up or pull-down input mode can be used.

The data present on the I/O pin is sampled into the Input Data register every AHB clock cycle.

A read access to the Input Data register gets the I/O state.

The figure below shows the input configuration of the I/O port:

Figure 6-2 Floating/pull-up/pull-down input configuration



To configure the PA [0] as pull-up input in the GPIOA, refer to the following:

GPIOA->GPIOA_ODR=0x0001.

GPIOA->GPIOA_CRL=0x00000008.

To configure the PA [0] as pull-down input in the GPIOA, refer to the following:

GPIOA->GPIOA_ODR=0x0000.

GPIOA->GPIOA_CRL=0x00000008.

Notes:

To configure the port as pull-up input, the corresponding bit in the GPIO_ODR register should output 1 firstly.

To configure the port as pull-down input, the corresponding bit in the GPIO_ODR register should output 0 firstly.

6.3.6 Output configuration

When the GPIO pin is configured as output:

The Schmitt Trigger Input is activated.

The Output Buffer is enabled.

In general purpose output mode, the weak pull-up and pull-down resistors are disabled.

Open Drain Mode: When the Port Output Data Register is set to 0, the corresponding pin outputs low level; when the Port Output Data Register is set to 1, the corresponding pin is in Hi-Z.

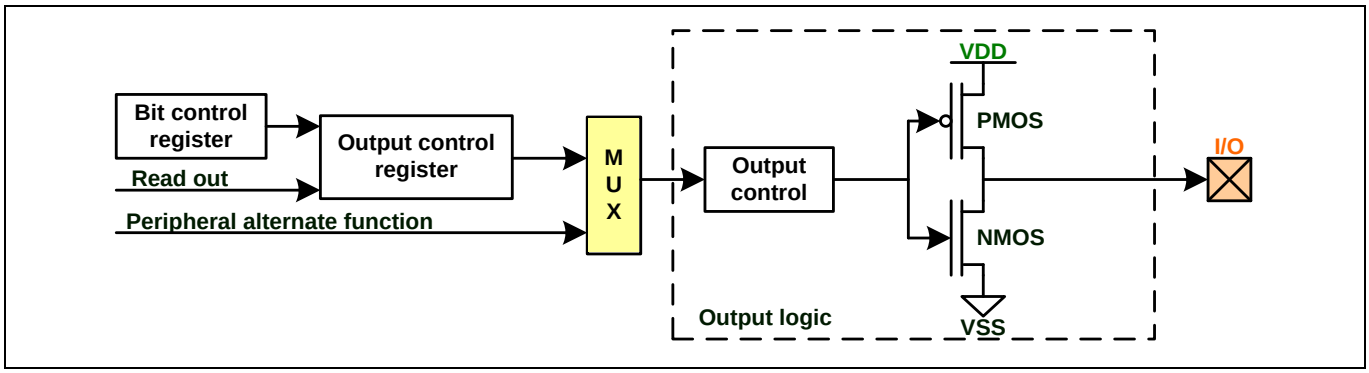
Push-Pull Mode: When the output register is set to 0, the corresponding pin outputs low level; when the output register is set to 1, the corresponding pin outputs high level.

A read access to the Port Output Data register gets the last written value.

A read access to the Port Input Data register gets the current I/O state.

The figure below shows the output configuration of the I/O port:

Figure 6-3 Output configuration



6.3.7 Alternate function configuration

When the pin is configured as alternate function:

The Schmitt Trigger Input is activated.

The Output Buffer can be configured as Open Drain or Push-Pull.

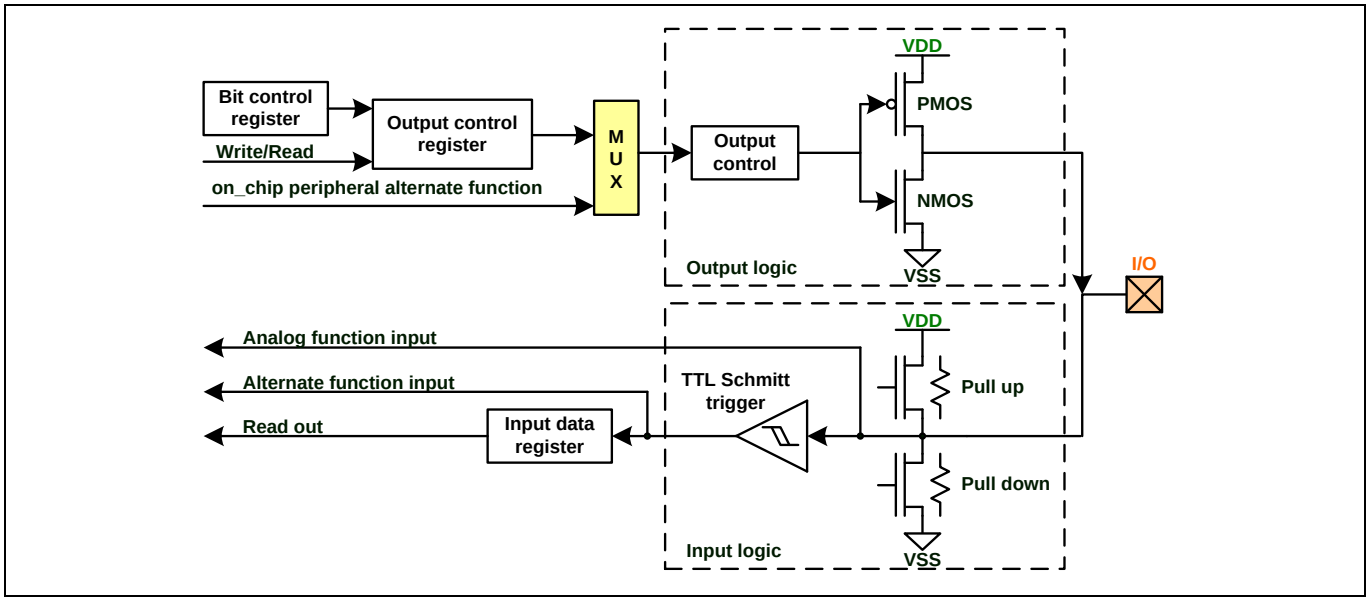
In the output open-drain mode, the GPIOx_DCR register is configured to choose weak pull-up or weak pull-down resistor.

The weak pull-up or weak pull-down resistor can be chosen when the pin is configured as Input.

The data present on the I/O pin is sampled into the Input Data register every AHB clock cycle.

The figure below shows the alternate function configuration of the I/O port. Refer to AFRL and AFRH registers as well as the datasheet for further information.

Figure 6-4 Alternate function configuration



6.3.8 Analog input configuration

When the I/O port is configured as analog input configuration:

The Output Buffer is disabled.

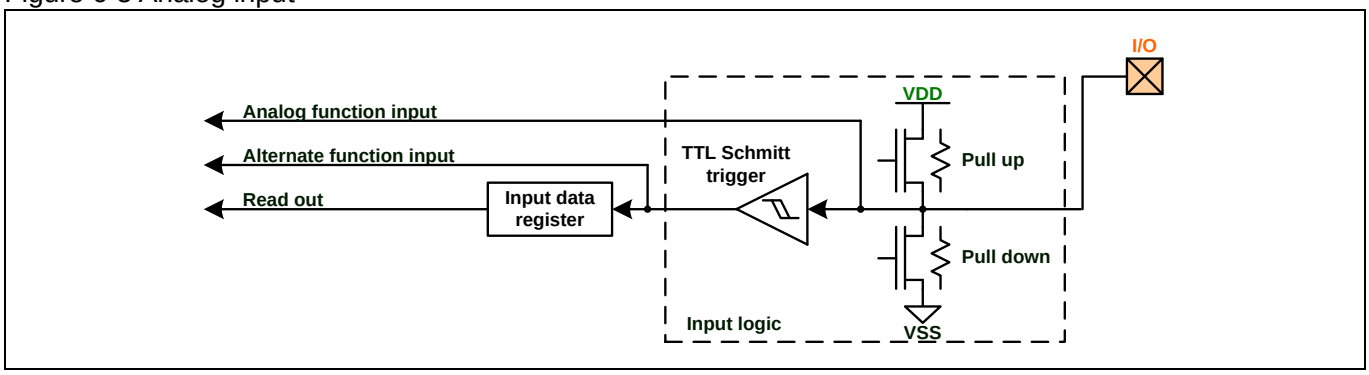
The Schmitt Trigger Input is de-activated.

The weak pull-up and pull-down resistors are disabled.

The Port Input Data Register remains 0.

The figure below shows the analog input configuration of the I/O port:

Figure 6-5 Analog input



6.3.9 SWD alternate function remapping

The SWD interface signals are mapped on the GPIO ports as shown in the table below:

Table 6-2 SWD alternate function remapping

Alternate function	GPIO port
SWDIO	PA13
SWCLK	PA14

6.4 Register

6.4.1 Overview of registers

Table 6-3 Overview of GPIO registers

Offset	Acronym	Register Name	Reset
0x00	GPIOx_CRL	Port configuration register low	See description below
0x04	GPIOx_CRH	Port configuration register high	See description below
0x08	GPIOx_IDR	Port input data register	0x0000XXXX
0x0C	GPIOx_ODR	Port output data register	0x00000000
0x10	GPIOx_BSRR	Port bit set/reset register	0x00000000
0x14	GPIOx_BRR	Port bit reset register	0x00000000
0x18	GPIOx_LCKR	Port configuration lock register	0x00000000
0x1C	GPIOx_DCR	Port output open drain control register	0x00000000
0x20	GPIOx_AFRL	Port alternate function register low	See description below
0x24	GPIOx_AFRH	Port alternate function register high	See description below

6.4.2 GPIOx_CRL Port Configuration Register Low

Address offset: 0x00

Reset value: GPIOA_CRL：0x4444 4444，GPIOB_CRL：0x0000 0044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF7		MODE7		CNF6		MODE6		CNF5		MODE5		CNF4		MODE4	
rw		rw		rw		rw		rw		rw		rw		rw	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNF3		MODE3		CNF2		MODE2		CNF1		MODE1		CNF0		MODE0	
rw		rw		rw		rw		rw		rw		rw		rw	

Bit	Field	Description
31:30	CNF7	Port configuration bits (y=7..0) Set MODEy to be 0 and the port as input mode while configuring the CNFy bit to choose the input mode： 00: Analog input mode 01: Input floating mode 10: Input pull-up/pull-down mode
27:26	CNF6	
23:22	CNF5	
19:18	CNF4	
15:14	CNF3	

11:10	CNF2	11: Reserved Set MODEy not to be 0 and the port as output mode while configuring the CNFy bit to choose the output mode： 00: General purpose output push-pull 01: General purpose output open-drain 10: Alternate function output push-pull 11: Alternate function output open-drain Port input/output configuration (MODEy)(y = 0..7) The corresponding I/O port is configured via software; refer to Port bit configuration table. Set MODEy is not equal to 0, different configurations result in different rates： 00: input mode; As for the other configuration output speeds, please refer to the datasheet.
7:6	CNF1	
3:2	CNF0	
29:28	MODE7	
25:24	MODE6	
21:20	MODE5	
17:16	MODE4	
13:12	MODE3	
9:8	MODE2	
5:4	MODE1	
1:0	MODE0	

6.4.3 GPIOx_CRH Port Configuration Register High

Address offset: 0x04

Reset value: GPIOA_CRH：0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF15		MODE15		CNF14		MODE14		CNF13		MODE13		CNF12		MODE12	
rw		rw		rw		rw		rw		rw		rw		rw	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNF11		MODE11		CNF10		MODE10		CNF9		MODE9		CNF8		MODE8	
rw		rw		rw		rw		rw		rw		rw		rw	

Bit	Field	Description
31:30	CNF15	Port configuration bits (y=15..8) Set MODEy to be 0 and the port as input mode while configuring the CNFy bit to choose the input mode： 00: Analog input mode 01: Input floating mode 10: Input pull-up/pull-down mode 11: Reserved Set MODEy not to be 0 and the port as output mode while configuring the CNFy bit to choose the output mode： 00: General purpose output push-pull 01: General purpose output open-drain 10: Alternate function output push-pull 11: Alternate function output open-drain Port input/output configuration (MODEy)(y = 15..8) The corresponding I/O port is configured via software; refer to Port bit configuration table. Set MODEy is not equal to 0, different configurations result in different rates： 00: input mode; As for the other configuration output speeds, please refer to the datasheet.
27:26	CNF14	
23:22	CNF13	
19:18	CNF12	
15:14	CNF11	
11:10	CNF10	
7:6	CNF9	
3:2	CNF8	
29:28	MODE15	
25:24	MODE14	
21:20	MODE13	
17:16	MODE12	
13:12	MODE11	
9:8	MODE10	
5:4	MODE9	
1:0	MODE8	

6.4.4 GPIOx_IDR Port Input Data Register

Address offset: 0x08

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDRy(y=15~0)															
r															

Bit	Field	Description
31:16	Reserved	Always read as 0

15:0	IDRy	Port input data (y=15..0) The reads represent the corresponding I/O states.

6.4.5 GPIOx_ODR Port Output Data Register

Address offset: 0xC

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
ODRy(y=15~0)															
rw															

Bit	Field	Description
31:16	Reserved	Always read as 0
15:0	ODRy	Port output data (y=15..0) When configured in general purpose output mode, the written value is output to the corresponding I/O. Note: The ODR bits can be individually set and cleared by setting the GPIOx_BSRR (x=A..H) Register.

6.4.6 GPIOx_BSRR Port Bit Set/Reset Register

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BRy(y=15~0)															
w															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BSy(y=15~0)															
w															

Bit	Field	Description
31:16	BRy	Port Reset bit y (y=15..0) Writing it to 0 will make the corresponding ODRY bit unchanged Writing it to 1 will clear the corresponding ODRY bit
15:0	BSy	Port Set bit y (y=15..0) Writing it to 0 will make the corresponding ODRY bit unchanged Writing it to 1 will set the corresponding ODRY bit to 1 Note: When both BSy and BRy bits are written to 1, BSy has priority over BRy.

6.4.7 GPIOx_BRR Port Bit Reset Register

Address offset: 0x14

Reset value: 0x0000 0000

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

Bit	Field	Description
31:16	Reserved	Always read as 0
15:0	BRy	Port Reset bit y (y=15..0) Writing it to 0 will make the corresponding ODRY bit unchanged Writing it to 1 will clear the corresponding ODRY bit

6.4.8 GPIOx_LCKR Port Configuration Lock Register

Address offset: 0x18

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCKy(y=15~0)															
w															

Bit	Field	Description
31:17	Reserved	Always read as 0
16	LCKK	Lock key This bit can be read anytime. It can only be modified using the Lock Key Writing Sequence. 0: port configuration lock key not active 1: port configuration lock key active. GPIOx_LCKR register is locked until the next soft reset. LOCK key sequence: write 1 -> write 0 -> write 1
15:0	LCKy	Port x Lock bit y (y = 15..0) These bits are read write but can only be written when the LCKK bit is 0. 0: port configuration not locked 1: port configuration locked

6.4.9 GPIOx_DCR Port Output Open Drain Control Register

Address offset: 0x1C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PX15		PX14		PX13		PX12		PX11		PX10		PX9		PX8	
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PX7		PX6		PX5		PX4		PX3		PX2		PX1		PX0	
rw															

Bit	Field	Description
31:2	PX15-PX1	See PX0
1:0	PX0	PX0[1:0]: 11: output open-drain mode with port pull-up 01: output open-drain mode with port pull-down x0: output open-drain mode without port pull-up/pull-down

6.4.10 GPIOx_AFRL Port Alternate Function Register Low

Address offset: 0x20

Reset value: GPIOA_AFRL：0xFFFF FFFF，GPIOB_AFRL：0x0000 00FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7				AFR6				AFR5				AFR4			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR3				AFR2				AFR1				AFR0			
rw				rw				rw				rw			

Bit	Field	Description
31:0	AFRy	Port x alternate function bit y (y = 0..7), which can be accessed by software to configure the function. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

6.4.11 GPIOx_AFRH Port Alternate Function Register High

Address offset: 0x24

Reset value: GPIOA_AFRH: 0xF00F FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR15				AFR14				AFR13				AFR12			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR11				AFR10				AFR9				AFR8			
rw				rw				rw				rw			

Bit	Field	Description
31:0	AFRy	Port x alternate function bit y (y = 8..15), which can be accessed by software to configure the function. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

7. EXTI Interrupt and Event

7.1 Overview

Nestled vector interrupt controller (NVIC) connects the processor core, and manages the anomaly and interrupt processing with low latency. NVIC includes 2-bit interrupt priority levels, which can thus provide 4 interrupt priority levels. For other more anomalies and NVIC programming details, please refer to the Cortex-Mx Technology Reference Manual.

The EXTI module includes the edge detection circuit, which can produce the interrupt request or wake-up event, and the edge detection supports the rising edge, falling edge or any edge configuration. Each edge detection circuit supports the independent enable and mask.

7.2 Main characteristics

Independently trigger and mask each interrupt

Configure interrupt/event output by software

Produce the wake up event to wake up the low power mode

The pending register saves the status of each corresponding interrupt line

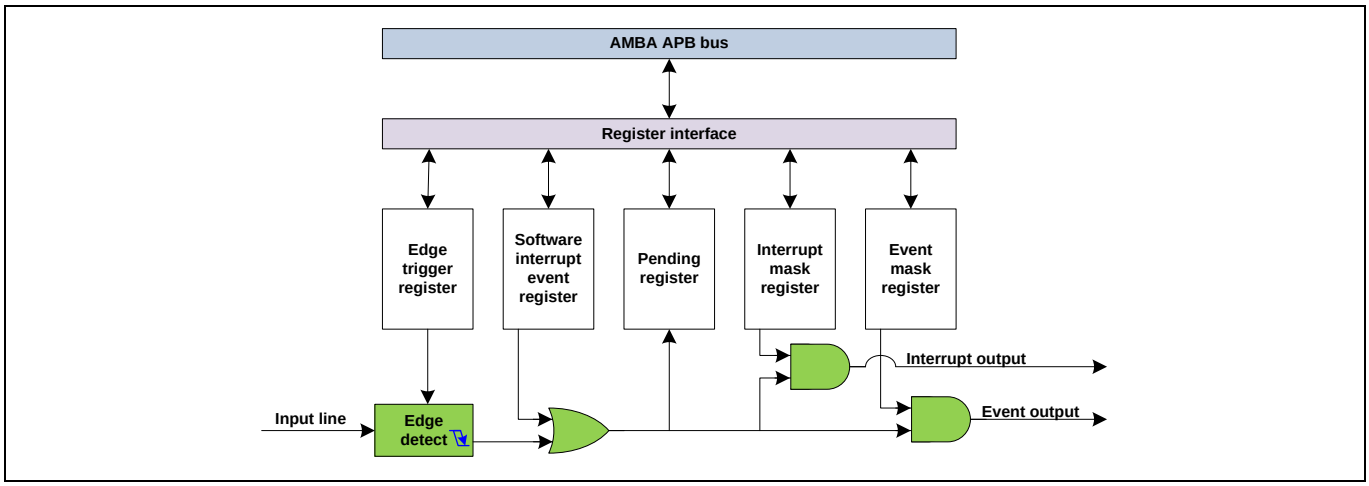
All GPIOs supports the trigger source configured as EXTI

Support the rising edge trigger, falling edge trigger and any edge trigger

7.3 Function description

7.3.1 Function block diagram

Figure 7-1 EXTI structure block diagram



7.3.2 Interrupt and anomaly vector

Under Handler mode, Cortex-M0 processor and nested interrupt vector control (NVIC) can process all anomalies based on priority. In case of anomaly, the system will pop the current processing work stack after interrupt service. Simultaneous processing of vector and current work status improves the interrupt efficiency. The table below presents the anomaly type and interrupt vector.

Table 7-1 Abnormal vector

Position	Priority	Priority type	Name	Description	Address
				Reserved	0x0000 0000
	-3	Fixed	Reset	Reset	0x0000 0004
	-2	Fixed	NMI	Non-maskable interrupt RCC Clock security system (CSS) connected to NMI	0x0000 0008
	-1	Fixed	Hardware fault	All types of faults	0x0000 000C

Table 7-2 Interrupt vector

Position	Priority	Priority type	Name	Description	Address
	3	Settable	SVCall	System service call via SWI instruction	0x0000 002C
	4	-		Reserved	0x0000 0030
		-		Reserved	0x0000 0034
	5	Settable	PendSV	Pending system service	0x0000 0038
	6	Settable	SysTick	System tick timer	0x0000 003C
0	7	Settable	WWDG_IWDG	WWDG and IWDG thru EXTI 17	0x0000_0040
1	8	Settable	PVD	PVD thru EXTI 16	0x0000_0044
2	9	Settable	Reserved	Reserved	0x0000_0048
3	10	Settable	FLASH	Flash interrupt	0x0000_004C
4	11	Settable	RCC	RCC interrupt	0x0000_0050
5	12	Settable	EXTI[1:0]	EXTI line [1:0] interrupt	0x0000_0054
6	13	Settable	EXTI[3:2]	EXTI line [3:2] interrupt	0x0000_0058
7	14	Settable	EXTI[15:4]	EXTI line [15:4] interrupt	0x0000_005C
8	15	Settable	Reserved	Reserved	0x0000_0060
9	16	Settable	Reserved	Reserved	0x0000_0064
10	17	Settable	Reserved	Reserved	0x0000_0068
11	18	Settable	Reserved	Reserved	0x0000_006C
12	19	Settable	ADC	ADC1 interrupt	0x0000_0070
13	20	Settable	TIM1_BRK_UP_TRG_COM	TIM1 brake refresh trigger COM interrupt	0x0000_0074
14	21	Settable	TIM1_CC	TIM1 capture compare interrupt	0x0000_0078
15	22	Settable	Reserved	Reserved	0x0000_007C
16	23	Settable	TIM3	TIM3 global interrupt	0x0000_0080
17	24	Settable	Reserved	Reserved	0x0000_0084
18	25	Settable	Reserved	Reserved	0x0000_0088
19	26	Settable	TIM14	TIM14 global interrupt	0x0000_008C
20	27	Settable	Reserved	Reserved	0x0000_0090
21	28	Settable	Reserved	Reserved	0x0000_0094
22	29	Settable	Reserved	Reserved	0x0000_0098
23	30	Settable	I2C1	I2C1 global interrupt	0x0000_009C
24	31	Settable	Reserved	Reserved	0x0000_00A0
25	32	Settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	Settable	Reserved	Reserved	0x0000_00A8
27	34	Settable	USART1	USART1 global interrupt	0x0000_00AC
28	35	Settable	USART2	USART2 global interrupt	0x0000_00B0
29	36	Settable	Reserved	Reserved	0x0000_00B4
30	37	Settable	Reserved	Reserved	0x0000_00B8
31	38	Settable	Reserved	Reserved	0x0000_00BC

7.3.3 Wake-up event management

All EXTI lines support the generation of interrupt or the event is used to wake the system from the low power mode. The user executes the WFE instruction to enter the corresponding low power mode. Configure the EXTI line event output to wake up the system. The user executes the WFI to enter the low power mode, and configures EXTI line interrupt output to wake up the system. For the detailed configuration, please refer to the section of power control.

7.3.4 Interrupt function description

To enable interrupt function and produce interrupt, configure the edge detection trigger register as the required trigger type, and open the corresponding bit of interrupt mask register allowing interrupt request. When the corresponding external interrupt line detects the configured trigger condition, it produces an interrupt request and the corresponding pending position of register is 1. Write the corresponding bit of pending register as 1 and

clear the interrupt.

To configure the generation event, configure the edge detection trigger register as the required trigger type, and open the corresponding bit of interrupt mask register allowing interrupt request. When the corresponding external interrupt line detects the configured trigger condition, it produces an interrupt request.

Enabling the corresponding bit of the software interrupt/event register can also produce the interrupt/event request.

7.3.5 Hardware interrupt output

Specific steps to configure hardware interrupt source:

Open the mask bit of the corresponding interrupt line (EXTI IMR), enable interrupt.

Configure the trigger register bit (EXTI_RISE/EXTI_FALL) of the corresponding interrupt line;

Open the NVIC connected interrupt channel so that the interrupt request can be transmitted to CPU and correctly responded.

When configuring EXTIx (x=31~0) line and produce interrupt output, the corresponding bit of EXTI_PR register will be set 1. It's required to clear the corresponding pending bit of EXTI_PR register before again detecting the rolling of EXTIx (x=31~0) line, and producing the interrupt.

There are three ways to clear the pending bit of EXTI_PR register:

Write the pending bit of EXTI_PR register as 1.

When configuring the rising edge trigger selection register (EXTI_RTSR), writing 0 to the corresponding bit will clear the pending bit. When configuring the falling edge trigger selection register (EXTI_FTSR), writing 0 in the corresponding bit will clear the pending bit.

Clear by changing the edge detection polarity of EXTI line.

7.3.6 Hardware event output

Specific steps to configure hardware event source:

Open the mask bit (EXTI_EMR) of the corresponding event line;

Configure the trigger register bit (EXTI_RISE/EXTI_FALL) of the corresponding event line.

7.3.7 Software interrupt and event output

Specific steps to support configuration of interrupt and event by software:

Enable event or interrupt enable bit (EXTI IMR, EXTI EMR).

Configure the corresponding bit of the software interrupt event register as 1 (EXTI SWIER).

7.3.8 External interrupt mapping

All GPIOs are used as the EXTI trigger source for producing the interrupt or event request. Configure SYSCFG_EXTICRx register, and meanwhile support the internal module (including PVD and IWDG) trigger. The specific connection relationships are shown as below:

Table 7-3 EXTI trigger source

External interrupt line	IO mapping	Control bit
EXTI0	PA0;PB	SYSCFG_EXTICR1 register EXTI0
EXTI1	PA1;PB1	SYSCFG_EXTICR1 register EXTI1
EXTI2	PA2	SYSCFG_EXTICR1 register EXTI2
EXTI3	PA3	SYSCFG_EXTICR1 register EXTI3
EXTI4	PA4	SYSCFG_EXTICR2 register EXTI4
EXTI5	PA5	SYSCFG_EXTICR2 register EXTI5
EXTI6	PA6	SYSCFG_EXTICR2 register EXTI6
EXTI7	PA7	SYSCFG_EXTICR2 register EXTI7
EXTI8	PA8	SYSCFG_EXTICR3 register EXTI8
EXTI9	PA9	SYSCFG_EXTICR3 register EXTI9
EXTI10	PA10	SYSCFG_EXTICR3 register EXTI10
EXTI11	PA11	SYSCFG_EXTICR3 register EXTI11

EXTI12	PA12	SYSCFG_EXTICR4 register EXTI12
EXTI13	PA13	SYSCFG_EXTICR4 register EXTI13
EXTI14	PA14	SYSCFG_EXTICR4 register EXTI14
EXTI15	PA15	SYSCFG_EXTICR4 register EXTI15

Other external interrupt/event controller connection:

EXTI line 16 connected to PVD output

EXTI line 17 connected to IWDG output

7.4 Register

7.4.1 Register overview

Table 7-4 EXTI register overview

Offset	Acronym	Register Name	Reset
0x00	EXTI_IMR	Interrupt mask register	0x00000000
0x04	EXTI_EMR	Event mask register	0x00000000
0x08	EXTI_RTSR	Rising edge trigger selection register	0x00000000
0x0C	EXTI_FTSR	Falling edge trigger selection register	0x00000000
0x10	EXTI_SWIER	Software interrupt event register	0x00000000
0x14	EXTI_PR	Pending register	0x00000000

7.4.2 EXTI_IMR Interrupt Mask Register

Address offset: 0x0

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														IMR17	IMR16
														rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	IMRx	Line x interrupt enable bit 1: Configure the bit as 1, enable line x corresponding interrupt 0: Configure the bit as 0, disable line x corresponding interrupt

7.4.3 EXTI_EMR Event Mask Register

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														EMR17	EMR16
														rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EMR15	EMR14	EMR13	EMR12	EMR11	EMR10	EMR9	EMR8	EMR7	EMR6	EMR5	EMR4	EMR3	EMR2	EMR1	EMR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	EMRx	Line x event enable bit 1: Configure the bit as 1, enable line x corresponding event 0: Configure the bit as 0, disable line x corresponding event

7.4.4 EXTI_RTSR Rising Edge Trigger Selection Register

Address offset: 0x08
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														TR17	TR16
														rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	TRx	Line x corresponding interrupt or event trigger polarity 1: Configure the bit as 1, enable line x corresponding rising edge trigger interrupt or event 0: Configure the bit as 0, disable line x corresponding rising edge trigger interrupt or event

7.4.5 EXTI_FTSR Falling Edge Trigger Selection Register

Address offset: 0x0C
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														TR17	TR16
														rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	TRx	Line x corresponding interrupt or event trigger polarity 1:Configure the bit as 1, enable line x corresponding falling edge trigger interrupt or event 0:Configure the bit as 0, disable line x corresponding falling edge trigger interrupt or event

7.4.6 EXTI_SWIER Software Interrupt Event Register

Address offset: 0x10
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														SWIER17	SWIER16
														rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIER15	SWIER14	SWIER13	SWIER12	SWIER11	SWIER10	SWIER9	SWIER8	SWIER7	SWIER6	SWIER5	SWIER4	SWIER3	SWIER2	SWIER1	SWIER0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	SWIERx	Line x software interrupt or event enable Writing 1 will set the corresponding pending bit of EXTI_PR register. Meanwhile, configuring the corresponding bit of EXTI_IMR or EXTI_EMR as 1 can produce interrupt or event.
		Note: Writing 1 to the corresponding bit of EXTI_PR register may clear the bit

7.4.7 EXTI_PR Software Interrupt Event Pending Register

Address offset: 0x14
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.														PR17	PR16
														rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit	Field	Description
31:25	Reserved	Reserved, always read as 0
24:0	PRx	Line x trigger pending bit 1: Selected trigger request 0: No trigger request In case of the selected edge event on the external interrupt line, the bit is set as 1. Writing 1 clears the bit, or clear by changing the edge detection polarity.

8. ADC Analog-to-Digital Converter

8.1 ADC introduction

The 12-bit ADC is a successive approximation analog-to-digital converter (SAR A/D converter).
A/D converter supports multiple operation modes: single conversion, continuous conversion, or automatic scan through selected channel. The ADC converter can be started by software, external pin trigger, and individual timers.
The window comparator (analog watchdog) allows the application to detect if the input voltage goes outside the user-defined high or low thresholds.
The ADC input clock is generated from the PCLK1 clock divided by a prescaler and it must not exceed 16MHz...

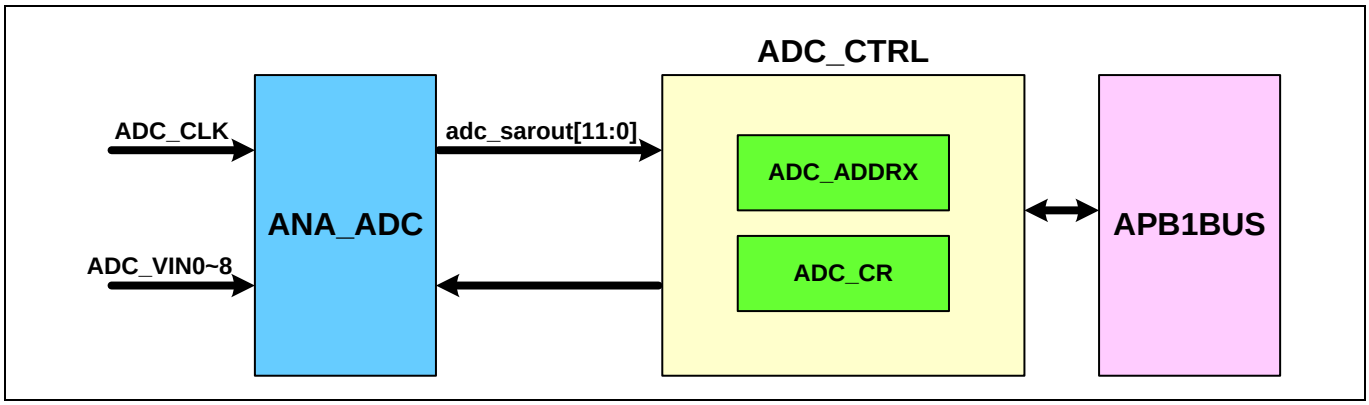
8.2 ADC main features

- **SAR ADC with up to 12-bit programmable resolution; no less than 8 multiplexed external input channels and 1 internal channel**
- **Conversion rate up to 1Msps**
- **Support normal operation mode:**
  - Single conversion mode: A/D converter does one conversion in the specified channel
  - One-cycle scan mode: A/D converter does one-cycle conversion in all specified channels (from low sequence number channel to high sequence number channel, or from high sequence number channel to low sequence number channel)
  - Continuous scan mode: A/D converter does continuous one-cycle conversions until the software stops A/D conversion. A/D conversion must stop if the conversion channel needs to be changed in the midway. The conversion can restart after configuring the corresponding register.
- Support arbitrary channel operation mode:
  - Single conversion mode: A/D converter does one conversion in the specified channel
  - One-cycle scan mode: A/D converter does one-cycle conversion in all specified channels (in any sequence)
  - Continuous scan mode: A/D converter does continuous one-cycle conversions until the software stops A/D conversion. A/D conversion doesn't have to stop if the conversion channel needs to be changed. The conversion in a new channel will start in the next scan cycle after configuring the corresponding register
- Support the configuration of channel sampling time and resolution via software
- A/D conversion start condition:
  - By software
  - By external trigger with the configuration of delay via software
- A/D conversion start conditions
  - Software start
  - Triggered start with configurable trigger delay
  - Timer1/3 matching or TRGO signal, external EXTI signal source
- Analog watchdog for comparing the conversion result with the specified value; An interrupt generation can be set by the user when the conversion result matches with the specified value.

8.3 ADC system block diagram

The system block diagram of the ADC is as follows:

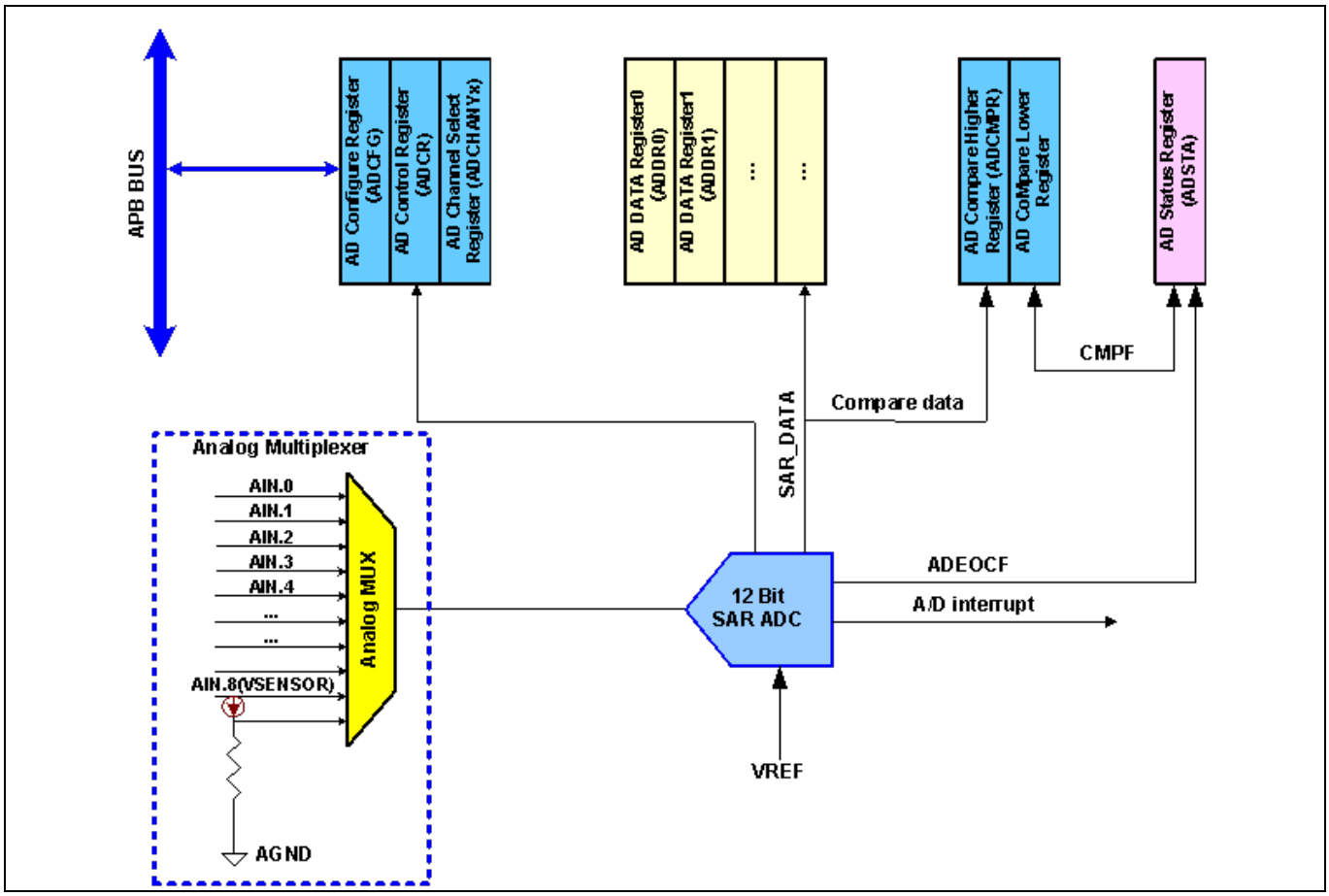
Figure 8-1 ADC system block diagram



8.4 ADC Functional Description

The ADC diagram is as follows:

Figure 8-2 ADC diagram



Note: V_SENSOR (internal reference voltage) is in the ADC AIN8 channel.

8.4.1 ADC on-off control

The ADC can be powered-on by setting the ADON bit in the ADCFG register. When the ADEN bit is set for the first time, it wakes up the ADC from Power Down mode.

Conversion starts when ADST bit is set for the ADCR register after an ADC power-up delay has elapsed (around 200ns).

The conversion can be stopped by resetting the ADST bit, and the ADC put in power down mode by resetting the ADEN bit

8.4.2 Channel Selection

ADC1 has 8 multiplexed channels including external input channels and internal 1.2V reference voltage channel. Each external input channel has an independent enable bit that can be set by CHANY_NUM, ADC_CHANY0 and ADC_CHANY1.

8.5 Arbitrary channel operation mode

8.5.1 Single Conversion Mode

In single conversion mode the A/D converter does one conversion in the corresponding channel. The specific procedures are as follows:

Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, the conversion channel, and the CHANY_MDEN bit. (only CHANY_SEL0 has to be set for the single conversion mode)

Set the ADST bit in the ADCR register by software, external trigger input, or timer overflow and start the A/D conversion.

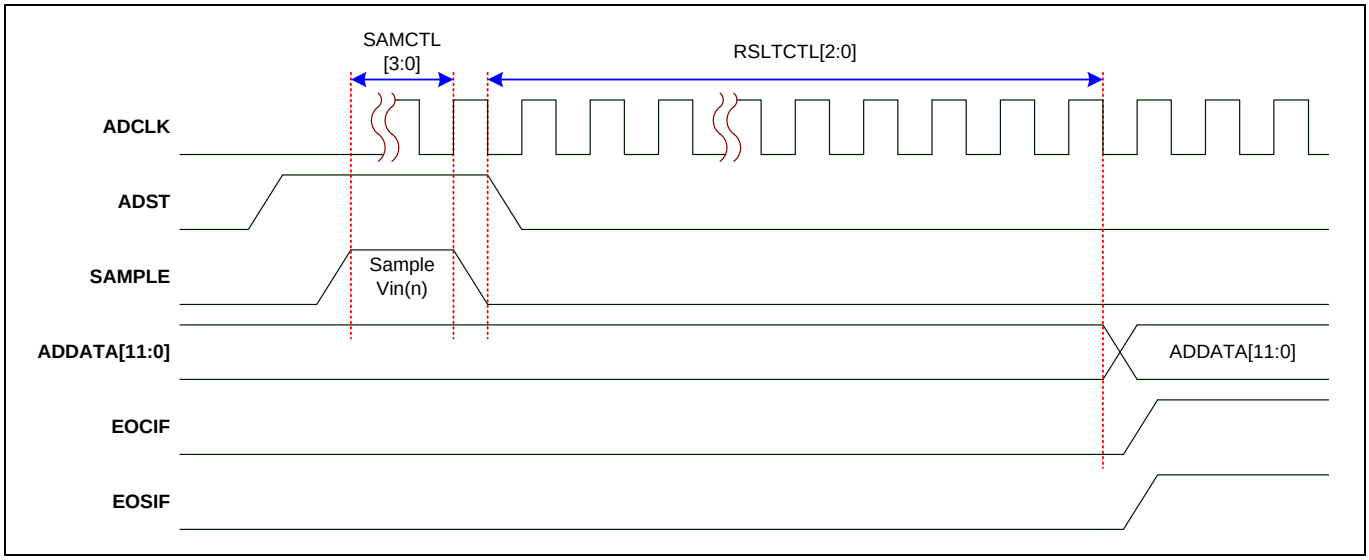
After A/D conversion, the converted data is stored in the data registers (ADDATA and ADDRn).

After A/D conversion, set the ADIF bit of the status register ADSTA to '1'. At this time, an AD EOC (End Of Conversion) interrupt request is generate if the ADIE bit in the control register ADCR is set to '1' .

ADST bit remains 1 during the A/D conversion period. Once the A/D channel sampling is complete, the ADST is automatically cleared and the A/D converter enters the idle mode.

If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of currently configured channels is complete, waiting for setting ADST by software next time.

Figure 8-3 Channel conversion timing diagram in single conversion mode



8.5.2 One-cycle scan mode

In one-cycle scan mode the A/D converter does one conversion according to the software configuration. The specific procedures are as follows:

Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, desired channels and quantity, and the CHANY_MDEN bit.

Set the ADST bit in the ADCR register by software or external trigger with the configuration of trigger delay via software. A/D conversion direction is from CHANY_SEL0 to CHANY_SEL8. The

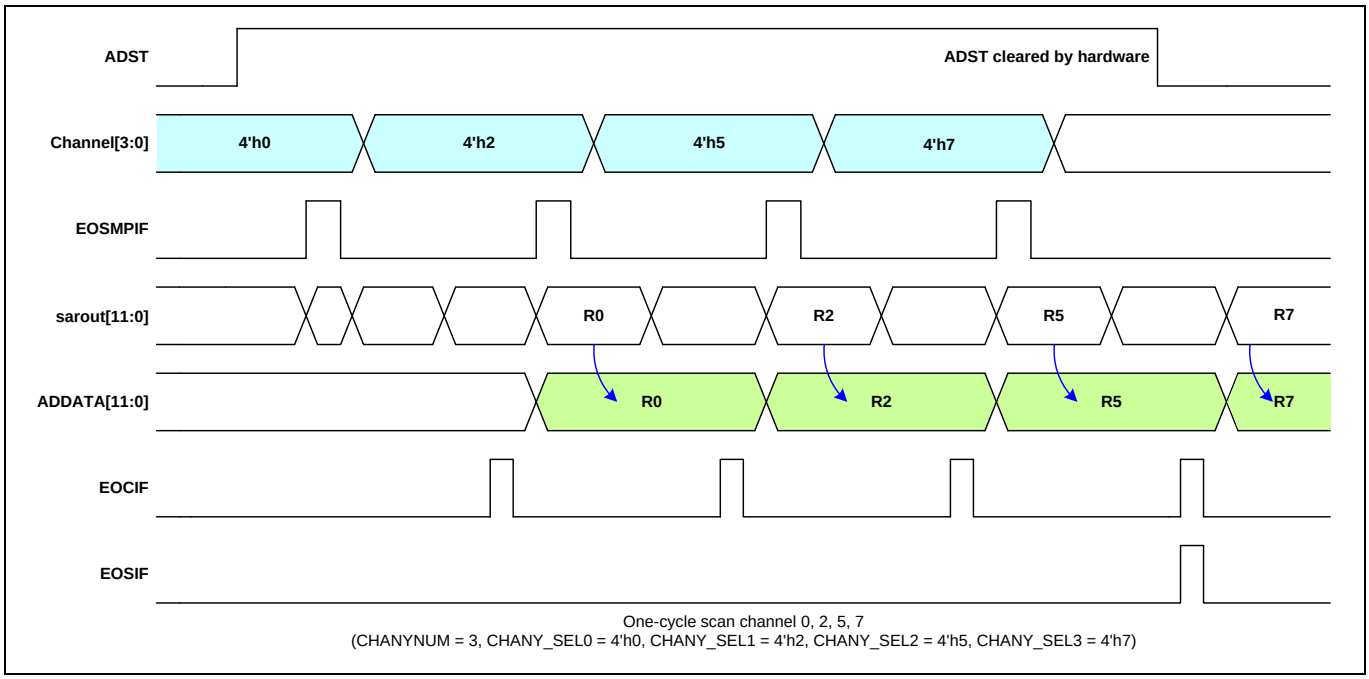
number of conversion channels is configured by CHANY_NUM. The number is arbitrarily configured from CHANY_SEL0 to CHANY_SEL8, which can be identical or not.

After the end of A/D conversion in each channel, the converted values will be loaded to the data register of corresponding channel in order. The ADIF EOC flag is set. At this time, an AD EOC interrupt request is generate if the ADIE bit in the control register ADCR is set to '1' .

Once the final A/D channel sampling is complete, the ADST is automatically cleared and the A/D converter enters the idle mode.

If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of currently configured channels is complete, waiting for setting ADST by software next time.

Figure 8-4 Channel conversion timing diagram in One-cycle conversion mode



8.5.3 Continuous Scan Mode

In continuous scan mode the A/D converter does continuous conversions according to the software configuration until the software stops conversion. The specific procedures are as follows:

Set by software the ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 registers, desired channels and quantity, and the CHANY_MDEN bit.

Set the ADST bit in the ADCR register by software or external trigger with the configuration of trigger delay via software. A/D conversion direction is from CHANY_SEL0 to CHANY_SEL8. The number of conversion channels is configured by CHANY_NUM. The number is arbitrarily configured from CHANY_SEL0 to CHANY_SEL8, which can be identical or not.

After the end of A/D conversion in each channel, the converted values will be loaded to the data register of corresponding channel in order. The ADIF EOC flag is set. At this time, an AD EOC interrupt request is generate if the ADIE bit in the control register ADCR is set to '1' .

The A/D conversion continues as long as the ADST bit remains 1. Once the ADST bit is cleared and the current A/D conversion is complete, the A/D converter enters the idle mode.

If the software updates ADC_ANY_CFG, ADC_CHANY0, and ADC_CHANY1 during the A/D conversion period, the hardware will not update these settings immediately. The updates will be applied when the conversion of current configured channels is complete, i.e. a conversion in the new channel starts in the next scan cycle.

Figure 8-5 Channel conversion timing diagram in Continuous scan mode

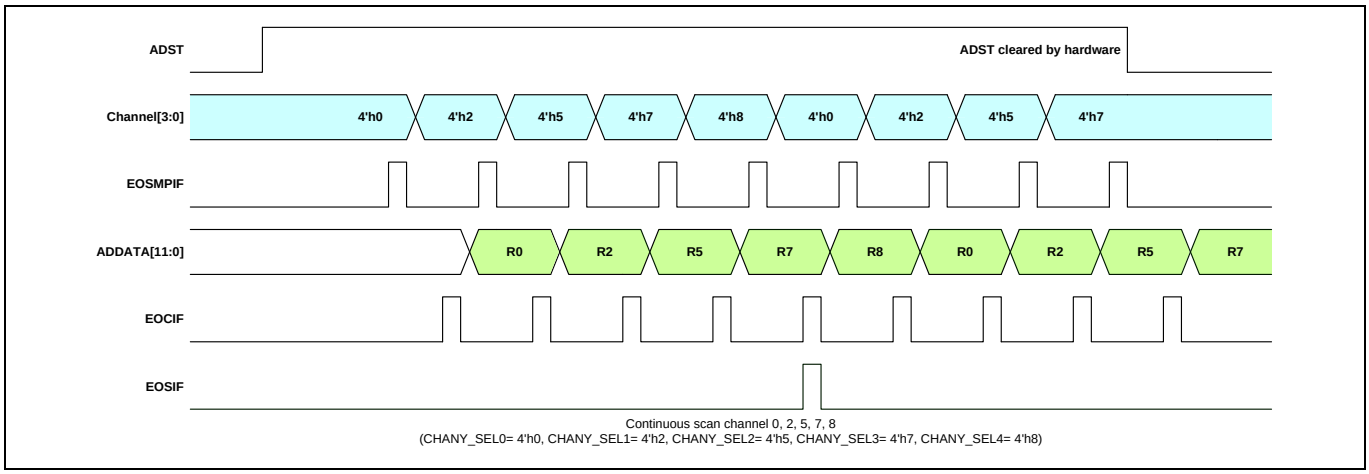
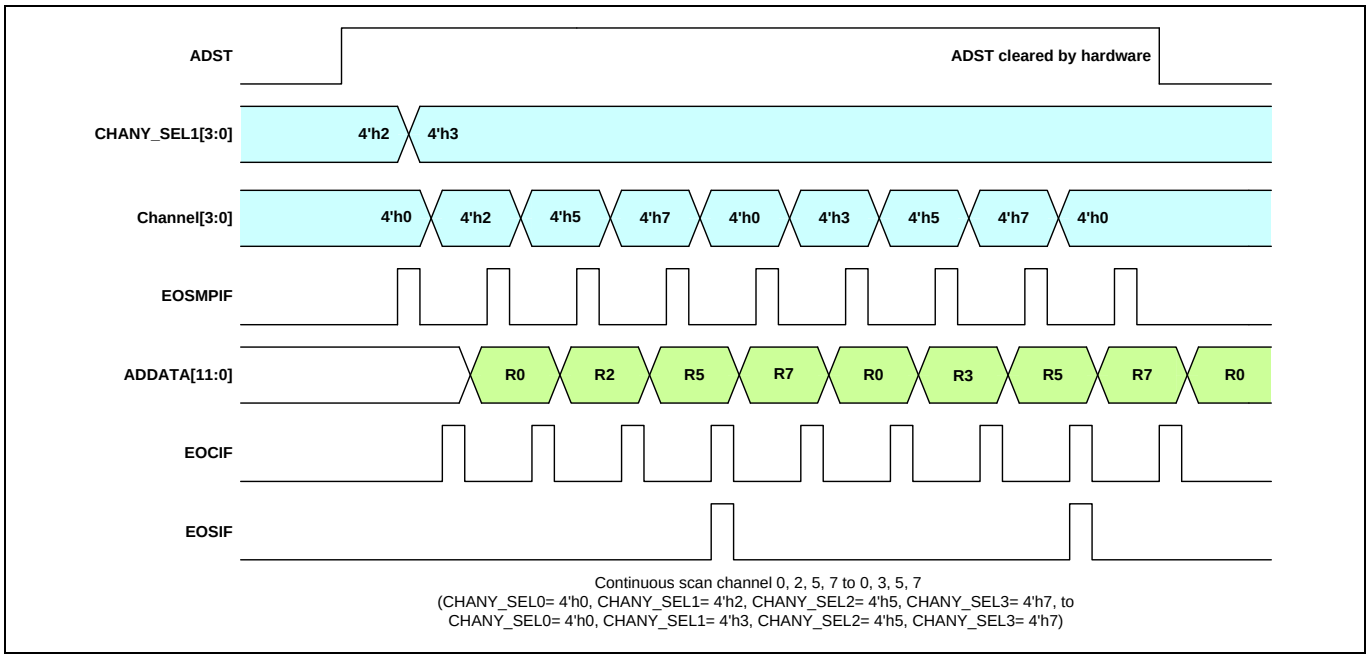


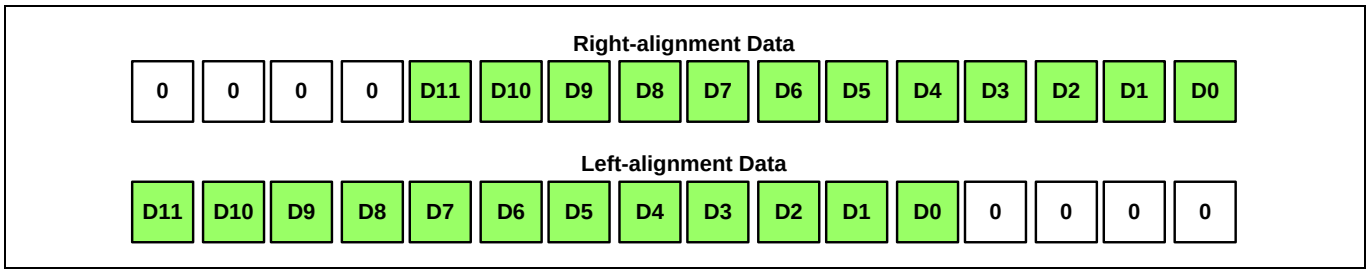
Figure 8-6 Timing diagram with dynamically updated configuration, in Continuous scan mode



8.6 Data Alignment

The ALIGN bit in the ADCR register selects the alignment mode for storing converted data. The data can be left-aligned or right-aligned, as shown in the figure below.

Figure 8-7 Data alignment mode



8.7 Programmable Resolution

The ADC conversion significance can be modified using the RSLTCTL [2:0] bits in the ADC_CFG register to accelerate the data conversion speed. The significant data bits are aligned from the higher bits of 12-bit data.

8.8 Programmable Sampling Time

The ADC clock ADCLK is generated from the PCLK 1 divided by a prescaler. The prescale coefficient can be determined by setting the ADCPRE bit in the ADCFG register, i.e. PCLK1/ (N + 2) divided by a prescaler to be

the ADC clock. ADC samples the input voltage for a number of ADC_CLK cycles which can be modified using the SAMCTL [3:0] bits in the ADC_CFG registers.
Set the ADC resolution as n bits (n=8, 9, 10, 11, 12). The sampling cycles of each channel is m.
The sampling frequency is calculated as follows:
$F_{sample} = F_{ADCLK} / (m + n + 0.5)$.
Assuming the resolution is configured as 12 bits and each channel has a sampling period of 3.5T, then $F_{sample} = F_{ADCLK} / 16$.
The total conversion time is calculated as follows:
T_{CONV} = Sampling time + 12.5 conversion cycles.
For example:
When ADCCLK = 16MHz and the sampling time is 3.5 periods, TCONV = 3.5 + 12.5 = 16, cycle=1μs.

8.9 Conversion on external trigger

ADC conversion can be triggered by an external event (e.g. timer capture, EXTI line). After the trigger signal is generated, the sampling is delayed by N PCLK1 clock cycles before starting again. In case of trigger scan mode, only the first channel sample is delayed, the rest of the channels are started immediately after the end of the previous sample.
If the TRGEN bit of the ADCR register is set, an external event can be used to trigger the conversion. The external trigger source can be selected by setting the TRGSEL bit.
For specific external trigger source selection, refer to the description of the relevant bits in the ADCR control register (ADCR.TRGSEL).
External triggering can be set for delay control, refer to the description of the relevant bits in the ADCR control register (ADCR.TRGSHIFT).

8.10 Internal Reference Voltage

The internal signal source channel of ADC is connected to an internal reference voltage of 1.2V. This channel converts the 1.2V reference voltage output to a digital value.
The internal reference voltage has an independent enable bit that can be enabled or disabled by setting the corresponding bit in the register.

8.11 Monitoring ADC Conversion Results in Window Comparator Mode

An upper limit comparison register and a lower limit comparison register are provided in the comparison mode. The monitoring channel can be selected by setting the CMPCH bit via software.
Providing CPMHDATA ≥ CPMLDATA, and the comparison result is greater than or equal to the specified CMPHDATA value or less than the specified CMPLDATA value in the ADCMPR register, the ADWIF bit in the status register ADSTA is set to 1.
Providing CPMHDATA < CPMLDATA, and the comparison result is equal to the specified CPMHDATA value or between two specified values, the ADWIF bit in the status register ADSTA is set to 1.
An interrupt request is generate if the ADWIE bit in the control register ADCR is set.

8.12 ADC register description

8.12.1 Register Overview

Table 8-1 ADC register overview

Offset	Acronym	Register Name	Reset
0x00	ADC_ADDATA	A/D data register	0x00000000
0x04	ADC_ADCFG	A/D configuration register	0x00000000
0x08	ADC_ADCR	A/D control register	0x00000000
0x10	ADC_ADCMPR	A/D analog watchdog comparison register	0x00000000
0x14	ADC_ADSTA	A/D status register	0x00000000
0x18~0x38	ADC_ADDR 0~8	A/D channel data register	0x00000000
0x5C	ADC_CHANY0	A/D arbitrary channel selection register 0	0x00000000
0x60	ADC_CHANY1	A/D arbitrary channel selection register 1	0x00000000

0x64	ADC_ANY_CFG	A/D arbitrary channel configuration register	0x00000000
0x68	ADC_ANY_CR	A/D arbitrary channel control register	0x00000000

8.12.2 A/D data register (ADC_ADDATA)

Address offset: 0x00
Reset value: 0x00000000

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

Bit	Field	Description
31:22	Reserved	Reserved, always read as 0.
21	VALID	Data Valid Flag (Read-only) 1: DATA[11:0] data is valid 0: DATA[11:0] data is invalid This bit is set after the corresponding analog channel conversion is complete. This bit is cleared by hardware after reading the ADDATA register.
20	OVERRUN	Data Overrun Flag (Read-only) 1: DATA[11:0] data is overrun 0: DATA[11:0] data is the last conversion result Before the new conversion result is loaded to the register, OVERRUN is set to 1 if the DATA [11:0] is not read. This bit is cleared by hardware after reading the ADDATA register.
19:16	CHANNELSEL	4 bits show the channel corresponding to the current data (Channel selection) 0000: conversion data of channel 0. 0001: conversion data of channel 1. 0010: conversion data of channel 2. 0011: conversion data of channel 3. 0100: conversion data of channel 4. 0101: conversion data of channel 5. 0110: conversion data of channel 6. 0111: conversion data of channel 7. 1000: conversion data of channel 8. Other: Invalid.
15:0	DATA	12-bit A/D conversion result (Transfer data) Data can be left or right aligned as configured.

8.12.3 A/D configuration register (ADC_ADCFG)

Address offset: 0x04
Reset value: 0x00000000

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.	ADCPREL	SAMCTL				RSLTCTL			ADCPREH			VSEN	Res.	ADWEN	ADEN
	rw	rw				rw			rw			rw		rw	rw

Bit	Field	Description
31:15	Reserved	Reserved, always read as 0.
14	ADCPREL	ADC Prescaler Low Bits Prescaler coefficient ADCPRE={ADCPREH, ADCPREL}
13: 10	SAMCTL	Channel x Sample time selection These bits are used to independently select the sample time for each channel. In a sample cycle, the channel selection bit should remain unchanged. 0000: 2.5 cycles 0100: 42.5 cycles

		0001: 8.5 cycles 0101: 56.5 cycles 0010: 14.5 cycles 0110: 72.5 cycles 0011: 29.5 cycles 0111: 240.5 cycles 1000: 3.5 cycles 1001: 4.5 cycles 1010: 5.5 cycles 1011: 6.5 cycles 1100: 7.5 cycles Others: Reserved
9: 7	RSLTCTL	ADC Conversion Data Resolution Selection 000: 12-bit effective 001: 11-bit effective 010: 10-bit effective 011: 9-bit effective 100: 8-bit effective Others: Reserved
6: 4	ADCPREH	ADC Prescaler High Bits Prescaler coefficient ADCPRE={ADCPREH, ADCPREL} ADC clock division：div= (ADCPRE+2)
3	VSEN	Voltage Sensor Enable 1: Internal voltage sensor enabled 0: Internal voltage sensor disabled
2	Reserved	Reserved and always read as 0.
1	ADWEN	ADC window comparison enable 1: A/D window comparison enabled 0: A/D window comparison disabled
0	ADEN	A/D conversion enable (ADC enable) 1：Enable 0：Disable

8.12.4 ADC_ADCR Control Register

Offset address: 0x08

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.						TRG_EDGE		Res.		TRGSHIFT			TRGSELH		Res.
						rw				rw			rw		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPCH				ALIGN	ADMD		ADST	Res.	TRGSELL			Res.	TRGEN		ADIE
rw				rw	rw		rw		rw				rw	rw	rw

Bit	Field	Description
31:26	Reserved	Reserved, always read as 0.
25 : 24	TRG_EDGE	Trigger Sources Edge Selection 00: Dual edge trigger 01: Falling edge trigger 10: Rising edge trigger 11: Trigger blocked
23:22	Reserved	Reserved, always read as 0.
21 : 19	TRGSHIFT	External trigger shift sample After a trigger signal is generated, the sampling starts after a delay of N PCLK1 clock cycles. In case of trigger scan mode, the sampling in the rest channels starts immediately after the last sampling is complete. 000: No delay 001: 4 cycles 010: 16 cycles 011: 32 cycles 100: 64 cycles 101: 128 cycles 110: 256 cycles 111: 512 cycles
18 :17	TRGSELH	External Trigger Sources Select For High Bits TRGSEL={TRGSELH,TRGSELL}
16	Reserved	Reserved, always read as 0.
15 : 12	CMPCH	Window comparison channel selection

		0000: conversion result for selecting comparison channel 0 0001: conversion result for selecting comparison channel 1 0010: conversion result for selecting comparison channel 2 0011: conversion result for selecting comparison channel 3 0100: conversion result for selecting comparison channel 4 0101: conversion result for selecting comparison channel 5 0110: conversion result for selecting comparison channel 6 0111: conversion result for selecting comparison channel 7 1000: conversion result for selecting internal comparison reference voltage 1111: All scan channels Others: Invalid
11	ALIGN	Data Alignment 1: Left alignment 0: Right alignment
10 : 9	ADMD	A/D conversion mode (ADC mode) 00: Single conversion 01: Single-cycle scan 10: Continuous scan 11: Reserved When the conversion mode is changed, the software has to disable the ADST bit.
8	ADST	A/D conversion start 1: Conversion start 0: Conversion end or idle state There are two ways of setting ADST: In the single or one-cycle mode, ADST is automatically cleared by hardware after conversion. In continuous scan mode the A/D converter does continuous conversions until the software writes‘0’ to the bit or the system is reset.
7	Reserved	Reserved, always read as 0.
6: 4	TRGSELL	External trigger selection, bits [18:17,6:4]: Select the external trigger source. 00000: TIM1_CC1 00001: TIM1_CC2 00010: TIM1_CC3 00011: Reserved 00100: TIM3_TRGO 00101: TIM1_CC4 and TIM1_CC5 00110: TIM3_CC1 00111: EXTI Line 11 01000: TIM1_TRGO 01001: Reserved 01010: Reserved 01011: Reserved 01100: TIM3_CC4 01101: Reserved 01110: Reserved 01111: EXTI Line 15 10000: TIM1_CC4 10001: TIM1_CC5 Others: Invalid
3	Reserved	Reserved, always read as 0.
2	TRGEN	External hardware trigger source (External trigger enable) 1: enable A/D conversion by external trigger signal 0: not enable A/D conversion by external trigger signal
1	AWDIE	ADC Window Comparator Interrupt Enable 1: Enable A/D window comparator interrupt 0: Disable A/D window comparator interrupt
0	ADIE	ADC Interrupt Enable 1: Enable A/D interrupt 0: Disable A/D interrupt An A/D EOC interrupt request is generated if the ADIF bit is set.

8.12.5 A/D window comparison register (ADC_ADCMPR)

Address offset: 0x10
Reset value: 0x00000000

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

Bit	Field	Description
31:28	Reserved	Reserved, always read as 0.
27: 16	CMPHDATA	Compare data high limit The 12-bit value will be compared with the conversion result of the specified channel.
15:12	Reserved	Reserved, always read as 0.
11 : 0	CMPLDATA	Compare data low limit The 12-bit value will be compared with the conversion result of the specified channel.

8.12.6 ADC_ADSTA Status Register

Offset address: 0x14
Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.			OVERRUN									Res.			VALID
			r												r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALID								CHANNEL				Res.	BUSY	AWDIF	ADIF
r								r					r	rc_w1	rc_w1

Bit	Field	Description
31:29	Reserved	Reserved, always read as 0
28:20	OVERRUN	Data overrun flag for channel 0 ~ 8 (Overrun flag) Read-only. Channel 8 is the voltage sensor channel.
19:17	Reserved	Reserved, always read as 0.
16:8	VALID	Valid flag for channel 0 ~ 8 (Valid flag) Read-only. Channel 8 is the voltage sensor channel.
7:4	CHANNEL	Current conversion channel These 4 bits indicate the channel in conversion when BUSY = 1. These 4 bits indicate the channel for the next conversion when BUSY = 0
3	Reserved	Reserved, always read as 0.
2	BUSY	Busy/Idle 1: A/D converter is busy 0: A/D converter is idle
1	AWDIF	ADC window comparator interrupt flag Providing CPMHDATA ≥ CPMLDATA, and the selected ADC channel comparison result is greater than or equal to the specified CMPHDATA value or less than the specified CMPLDATA value in the ADCMPR register, the ADWIF bit in the status register ADSTA is set to 1. Providing CPMHDATA < CPMLDATA, and the selected ADC channel comparison result is equal to the specified CPMHDATA value or between two specified values, the ADWIF bit in the status register ADSTA is set to 1. This flag bit is cleared by writing '1'
0	ADIF	ADC interrupt flag

		This bit is set by the hardware when the channel group conversion is completed and cleared by software. 1: A/D conversion completed 0: A/D conversion not completed This flag bit is cleared by writing '1'

8.12.7 A/D data register (ADC_ADDR0 ~ 8)

Address offset: 0x18~0x38

Reset value: 0x00000000

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

Bit	Field	Description
31:22	Reserved	Reserved, always read as 0.
21	VALID	Valid Flag (read-only): 1: DATA [11:0] bits data is valid. 0: DATA [11:0] bits data is invalid. This bit is set after the corresponding analog channel conversion is complete. This bit is cleared by hardware after reading the ADDATA register.
20	OVERRUN	Data overrun flag (read-only) 1: DATA [11:0] data is overrun 0: DATA [11:0] data is the last conversion result Before the new conversion result is loaded to the register, OVERRUN is set to '1' if the DATA [11:0] is not read. This bit is cleared by hardware after reading the ADDATA register.
19 : 16	Reserved	Reserved, always read as 0.
15 : 0	DATA	12-bit A/D conversion result in the channel (Transfer data) Data can be left or right aligned as configured.

8.12.8 A/D arbitrary channel selection register (ADC_CHANY0)

Address offset: 0x5C

Reset value: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHANY_SEL7				CHANY_SEL6				CHANY_SEL5				CHANY_SEL4			
rw				rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHANY_SEL3				CHANY_SEL2				CHANY_SEL1				CHANY_SEL0			
rw				rw				rw				rw			

Bit	Field	Description
31:28	CHANY_SEL7	Can be configured to any channel from channel 0 ~ 8.
27:24	CHANY_SEL6	Can be configured to any channel from channel 0 ~ 8.
23:20	CHANY_SEL5	Can be configured to any channel from channel 0 ~ 8.
19:16	CHANY_SEL4	Can be configured to any channel from channel 0 ~ 8.
15:12	CHANY_SEL3	Can be configured to any channel from channel 0 ~ 8.
11:8	CHANY_SEL2	Can be configured to any channel from channel 0 ~ 8.
7:4	CHANY_SEL1	Can be configured to any channel from channel 0 ~ 8.
3:0	CHANY_SEL0	Can be configured to any channel from channel 0 ~ 8.

Note: In the one-cycle or continuous scan modes, the ADC_CHANY0 shadow register is started by hardware. Before ADC starts to work, writing to ADC_CHANY0 by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_CHANY0 value is changed. Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_CHANY0, so as to complete the dynamic channel switch.

8.12.9 A/D arbitrary channel selection register 1 (ADC_CHANY1)

Address offset: 0x60
Reset value: 0x00000000

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.												CHANY_SEL8			
												rw			

Bit	Field	Description
31:4	Reserved	Reserved, always read as 0.
3:0	CHANY_SEL8	Can be configured to any channel from channel 0 ~ 8.

Note: In the one-cycle or continuous scan modes, the ADC_CHANY1 shadow register is started by hardware. Before ADC starts to work, writing to ADC_CHANY1 by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_CHANY1 value is changed. Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_CHANY1, so as to complete the dynamic channel switch.

8.12.10 A/D arbitrary channel configuration register (ADC_ANY_CFG)

Address offset: 0x64
Reset value: 0x00000000

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.												CHANY_NUM			
												rw			

Bit	Field	Description
31:4	Reserved	Reserved, always read as 0.
3: 0	CHANY_NUM	Number of Any Channel Mode: 0000：Channel 0 0001：Channel 0~1 0010：Channel 0~2 0011：Channel 0~3 0100：Channel 0~4 0101：Channel 0~5 0110：Channel 0~6 0111：Channel 0~7 1000：Channel 0~8 Other：Invalid

Note: In the one-cycle or continuous scan modes, the ADC_NUM shadow register is started by hardware. Before ADC starts to work, writing to ADC_NUM by software also writes to its shadow register. When ADC is working, only the shadow register is updated if the ADC_NUM value is changed. Besides that, when ADC starts to convert the last channel, the value of the shadow register will update to ADC_NUM, so as to complete the dynamic channel switch.

8.12.11 A/D arbitrary channel control register (ADC_ANY_CR)

Address offset: 0x68

Reset value: 0x00000000

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.															CHANY_ MDEN
															rw

Bit	Field	Description
31:1	Reserved	Reserved, always read as 0.
0	CHANY_ MDEN	Arbitrary channel configuration mode enable bit: 1: Enable 0: Disable When this bit is enabled, the function of ADC channel configuration become changed. There are two controlling parts: CHANY_NUM is used to configure the number of channels from channel 0 ~ 8. CHANY_SEL0 ~ CHANY_SEL8 is used to configure them as an arbitrary ADC channel.

Note: In the arbitrary mode plus the one-cycle/continuous scan mode, the ADST bit in the ADC_ADCR must be disabled before closing the ADC. Then the user should judge whether the BUSY bit in the ADC_ADSTA is 0. That is to say, the CHANY_MDEN bit in the ADC_ANY_CR should be disabled when the ADC conversion is complete.

9. TIM1 Advanced-Control Timer

9.1 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 as described in the general-purpose timer synchronization section.

9.2 Main features

TIM1 timer features include:

16-bit up, down, up/down auto-reload counter.

16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 ~ 65536.

5 comparison channels (channels 1~4 support output, while channel 5 only provides interrupt and internal trigger signals)

4 output channels, channel 1/2/3 has complementary output channels, and channel 4 has no complementary output channels

- Output compare
- PWM generation (Edge and Center-aligned Mode)
- One-pulse mode output

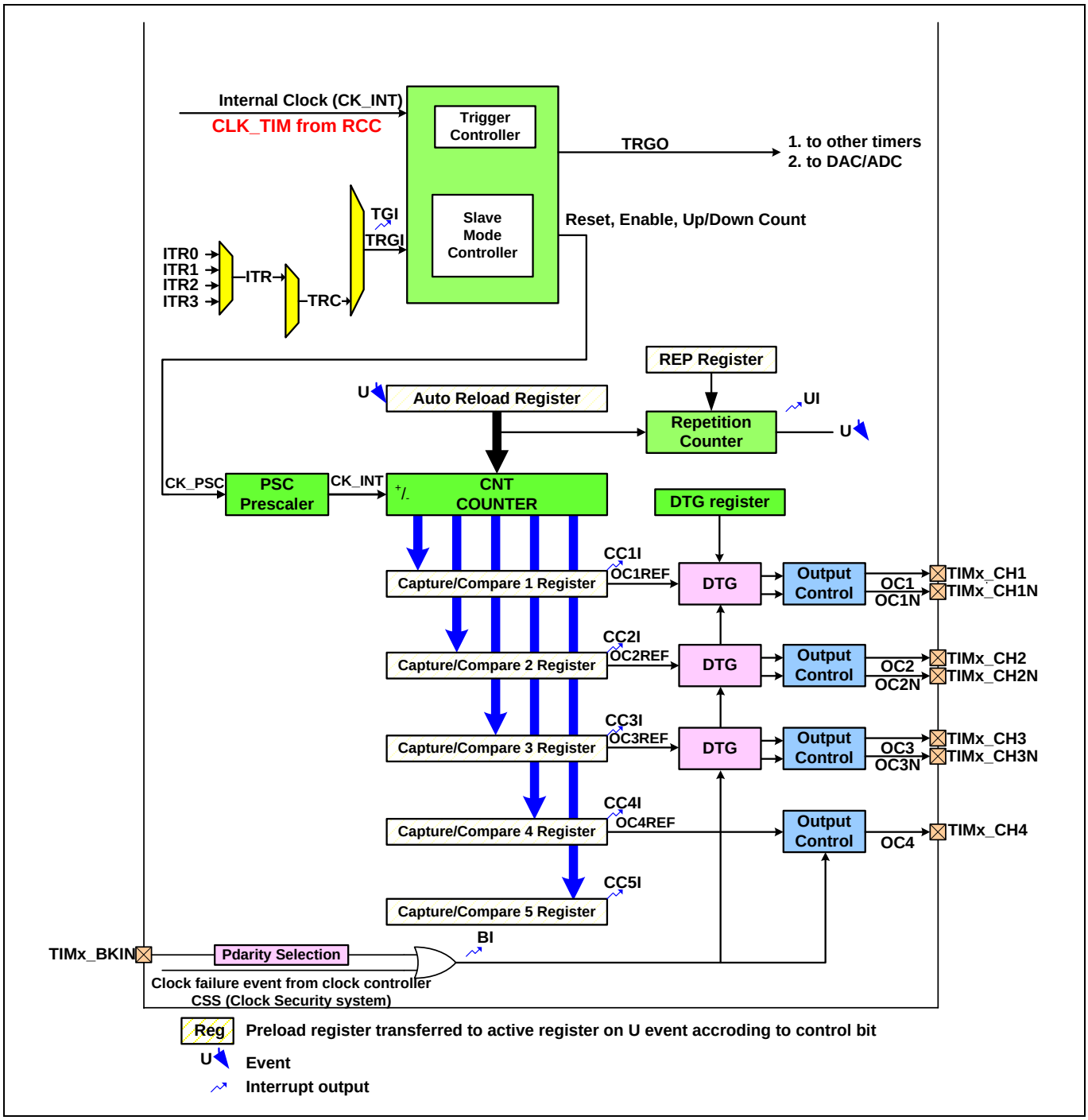
Complementary outputs with programmable dead-time

Synchronization circuit to control the timer with external signals and to interconnect several timers together

Repetition counter to update the timer registers only after a given number of cycles of the counter

Break input to put the timer’s output signals in reset state or in a known state

Figure 9-1 Advanced-control timer block diagram



9.3 Functional description

9.3.1 Time-base unit

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

The time-base unit includes:

Counter register (TIMx_CNT)

Prescaler register (TIMx_PSC)

Auto-reload register (TIMx_ARR)

Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the autoreload 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 generation of the update event is described in detailed for each configuration.
The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).
Note: that the counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR1 register.

9.3.1.1 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.
The following two diagrams each give examples of changing the counter parameters while the prescaler is running.

Figure 9-2 Counter timing diagram with prescaler division change from 1 to 2

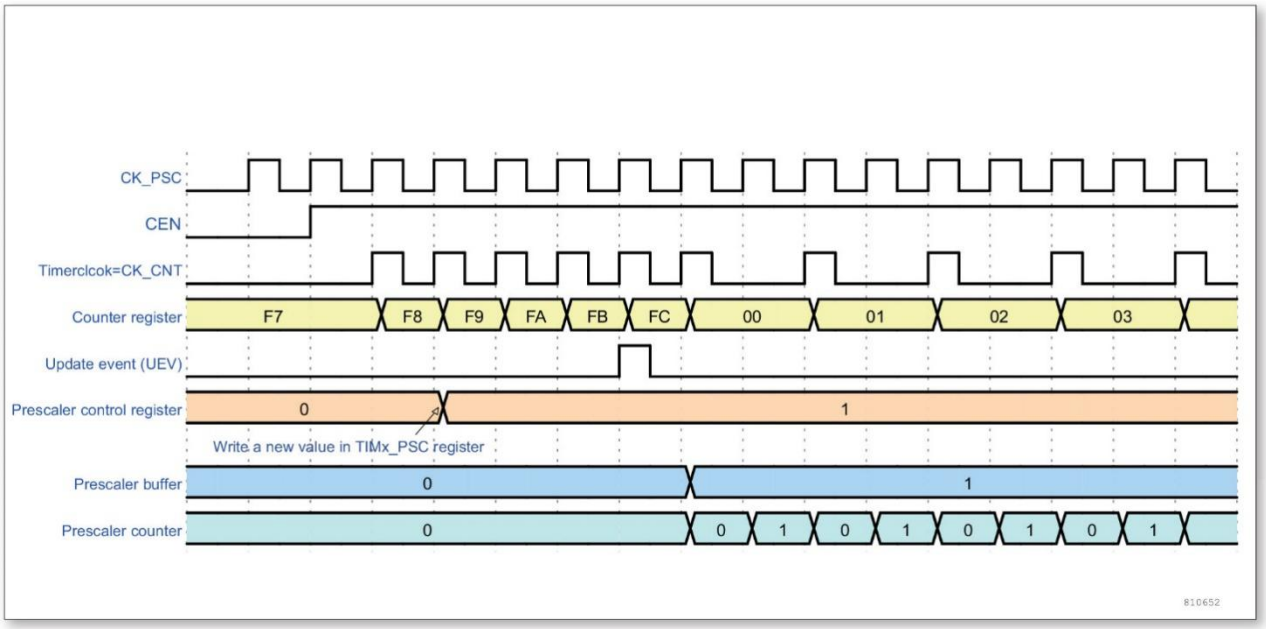
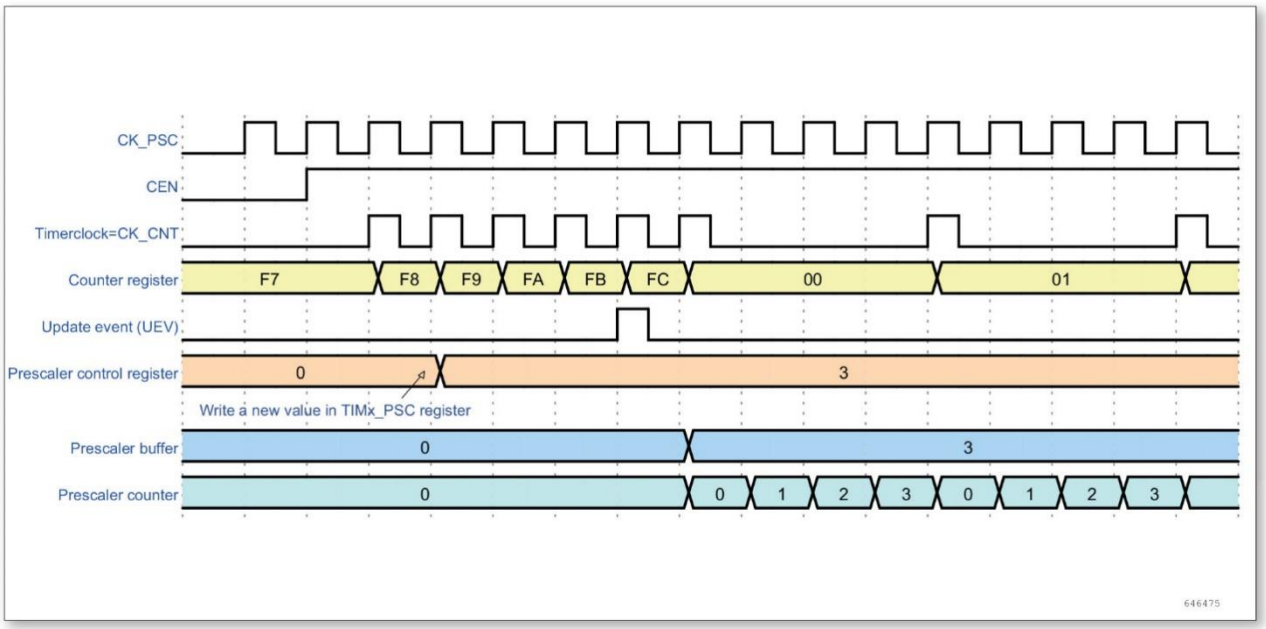


Figure 9-3 Counter timing diagram with prescaler division change from 1 to 4



9.3.2 Counter modes

9.3.2.1 Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.
If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times program in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter overflow.
Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.
The UEV event can be disabled by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. 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 by hardware (thus no interrupt is sent). When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set by hardware (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 9-4 Counter timing diagram, internal clock divided by 1

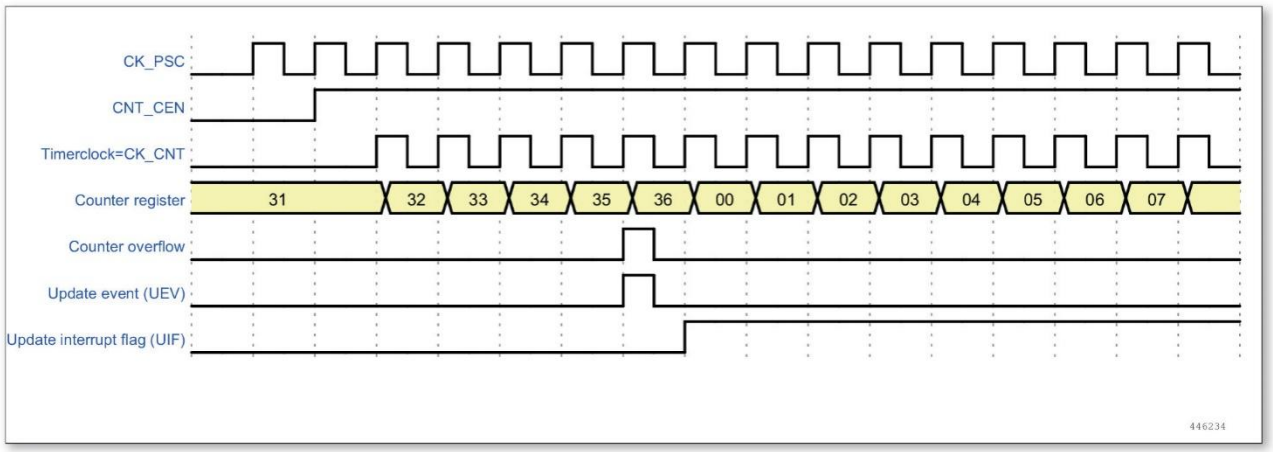


Figure 9-5 Counter timing diagram, internal clock divided by 2

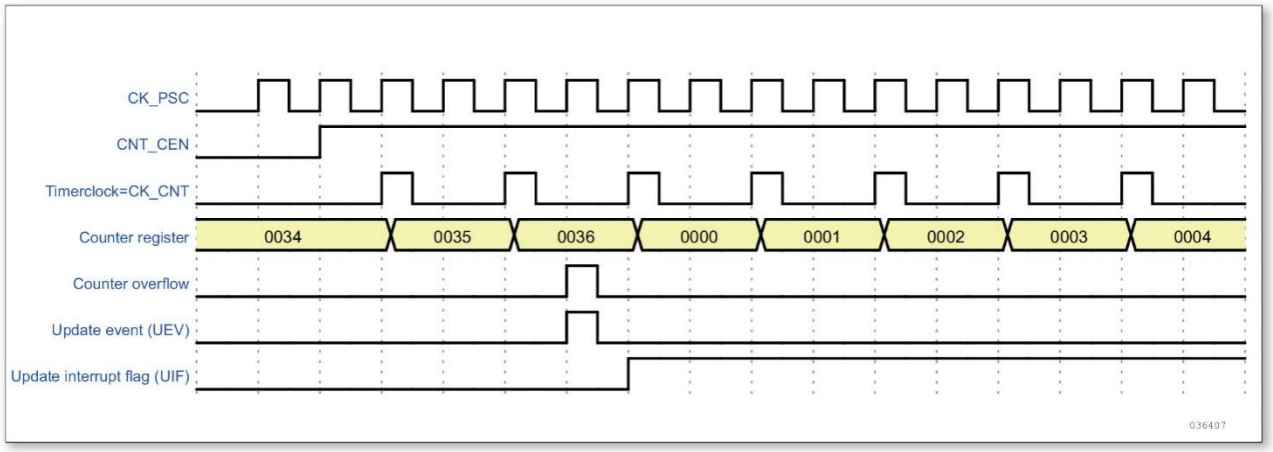


Figure 9-6 Counter timing diagram, internal clock divided by 4

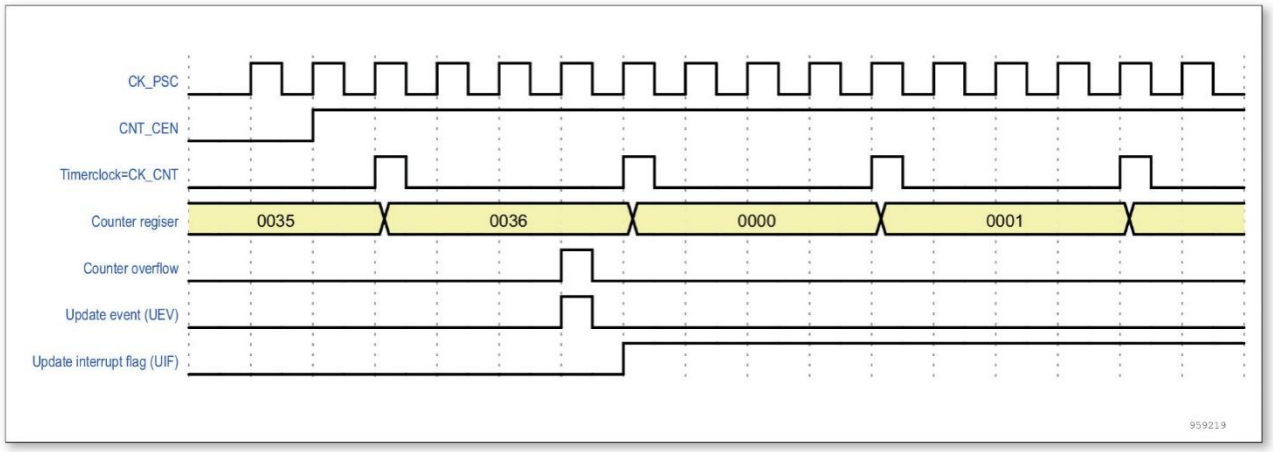


Figure 9-7 Counter timing diagram, internal clock divided by N

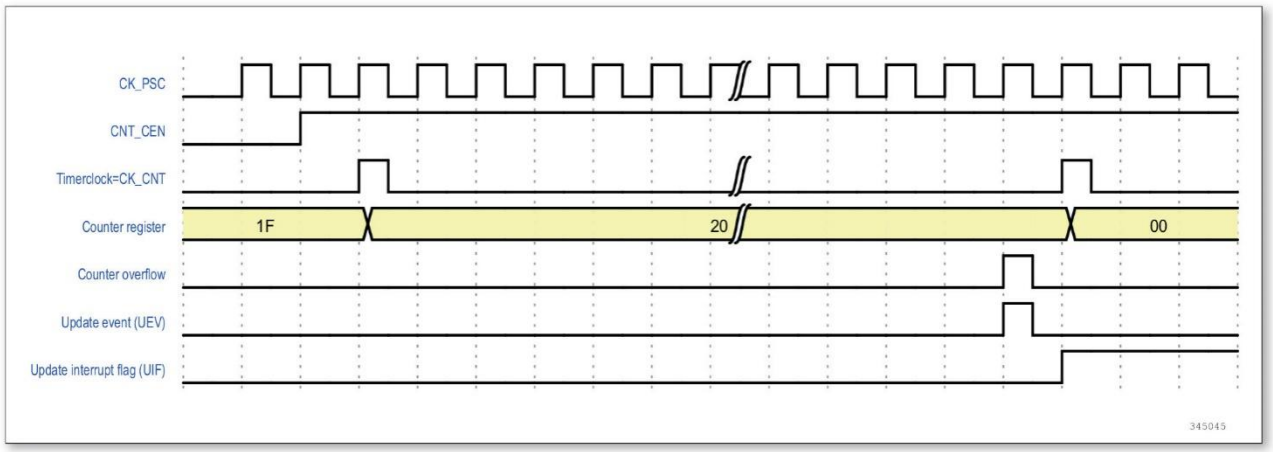


Figure 9-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

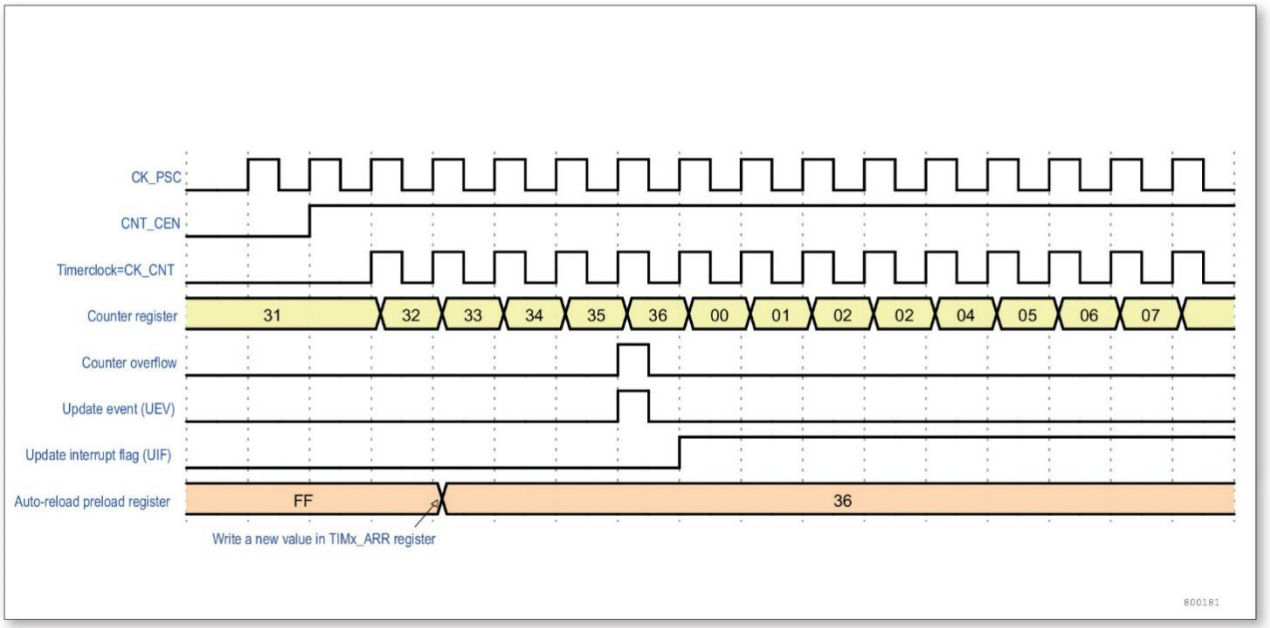
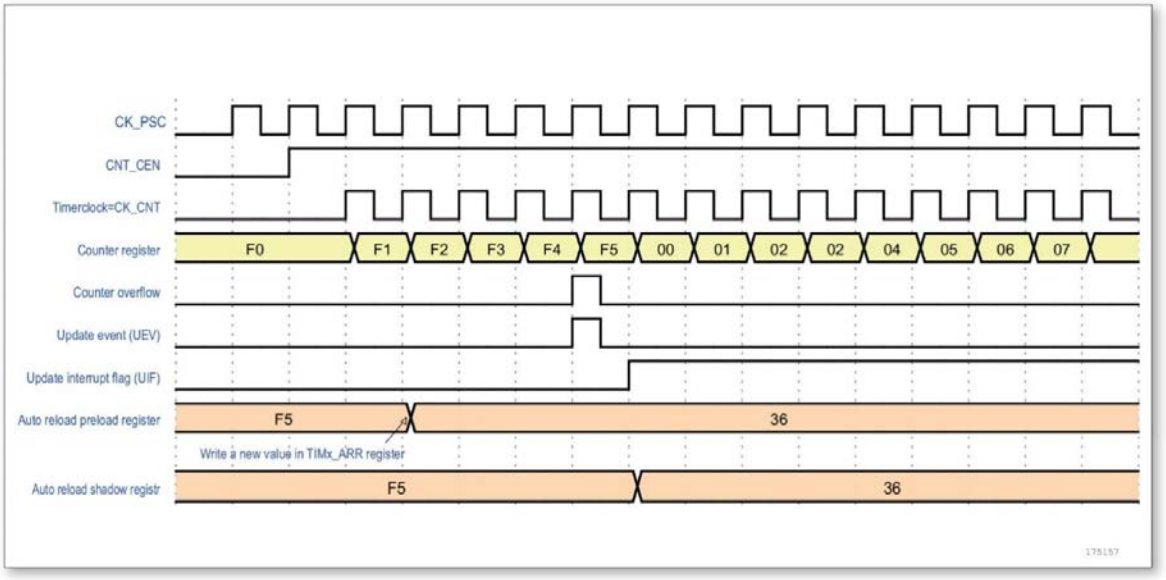


Figure 9-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)



9.3.2.2 Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (value of the TIMx_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.
If the repetition counter is used, the update event (UEV) is generated after downcounting is repeated for the number of times program in the repetition counter register (TIMx_RCR). Else the update event is generated at each counter underflow.
Setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller) also generates an update event.
The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).
In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent).
When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

The repetition counter is reloaded with the content of TIMx_RCR register.

The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note: the auto-reload is updated before the counter is reloaded, so that the next period is the expected one.
The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36.

Figure 9-10 Counter timing diagram, internal clock divided by 1

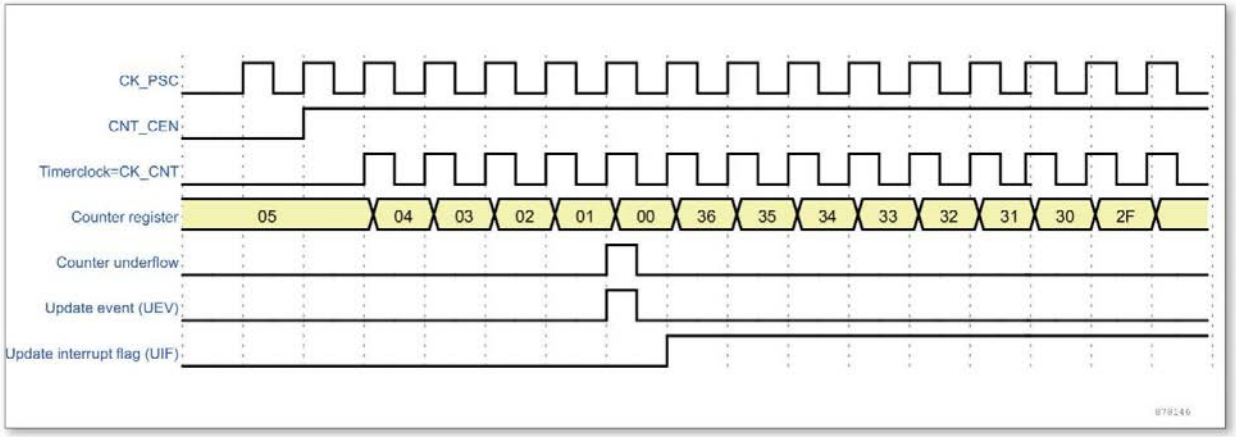


Figure 9-11 Counter timing diagram, internal clock divided by 2

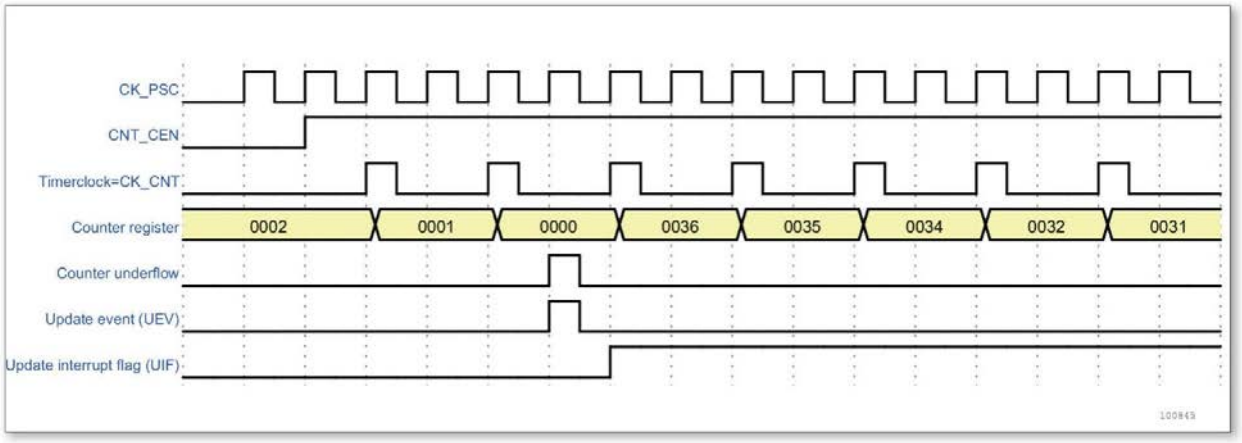


Figure 9-12 Counter timing diagram, internal clock divided by 4

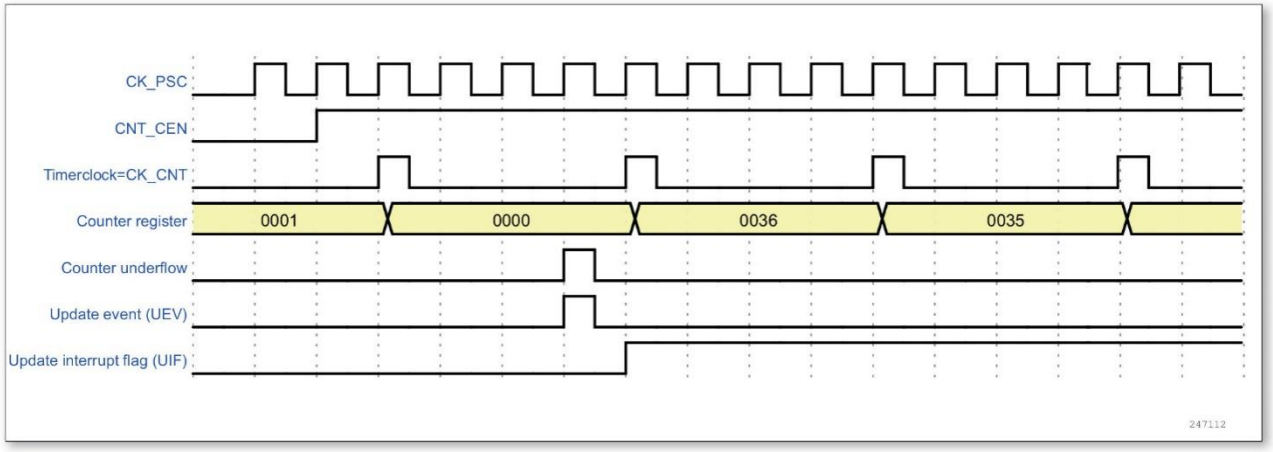


Figure 9-13 Counter timing diagram, internal clock divided by N

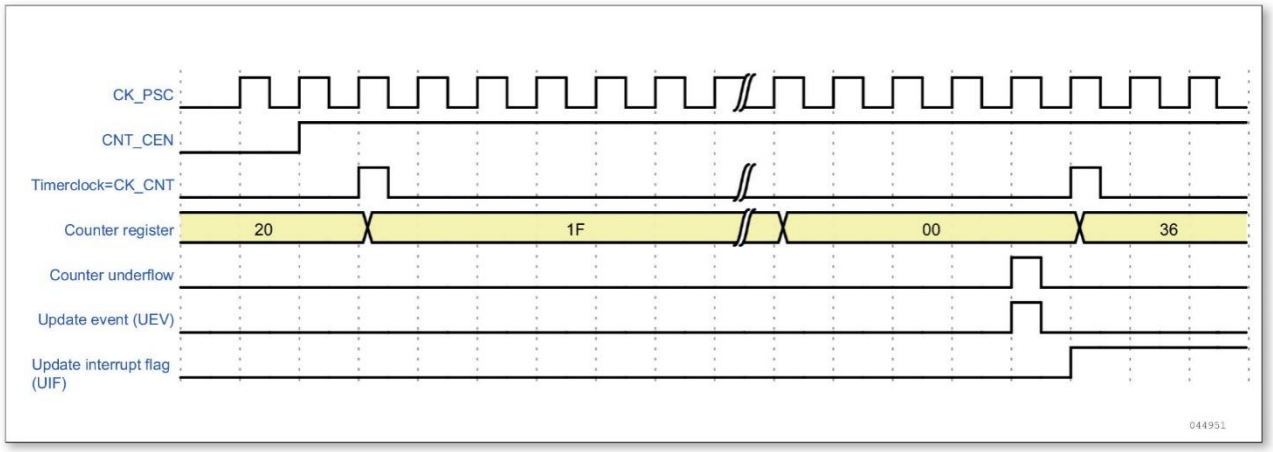
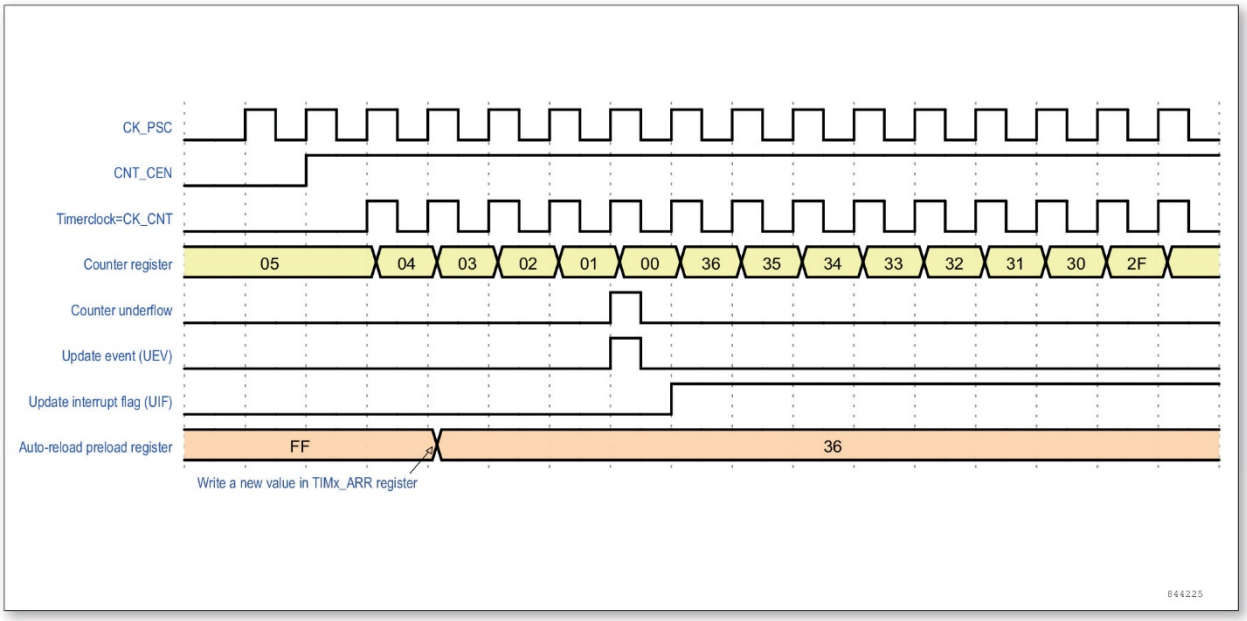


Figure 9-14 Counter timing diagram, update event when repetition counter is not used



9.3.2.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (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.

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). In this case, the counter restarts counting from 0, as well as the counter of the prescaler.

The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers.

Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

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

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 9-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x06

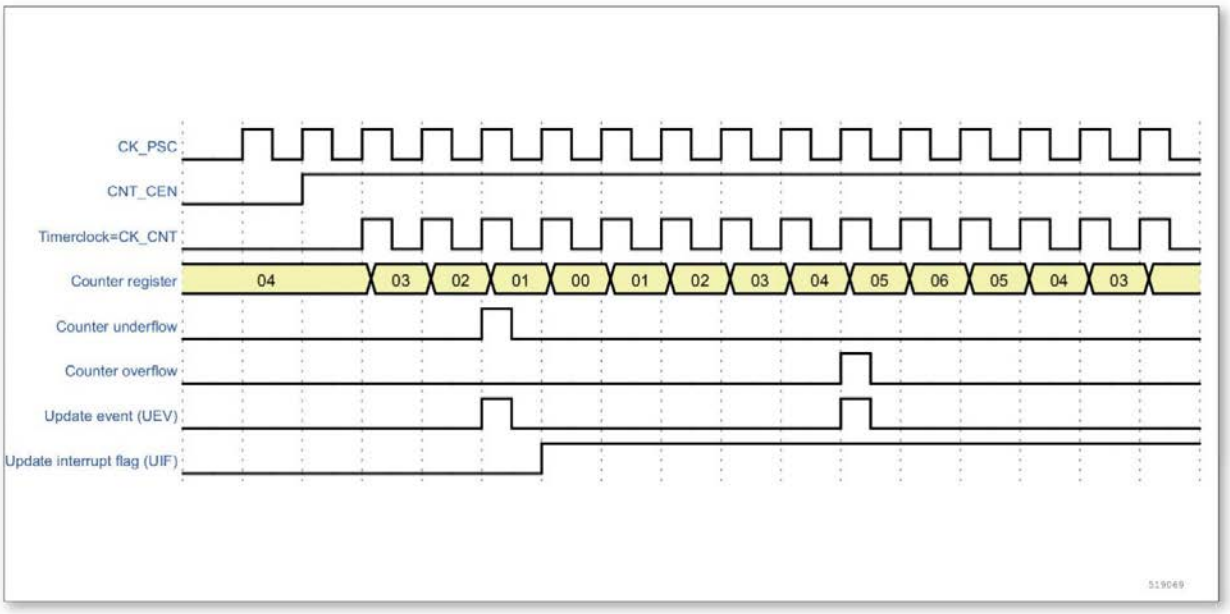


Figure 9-16 Counter timing diagram, internal clock divided by 2

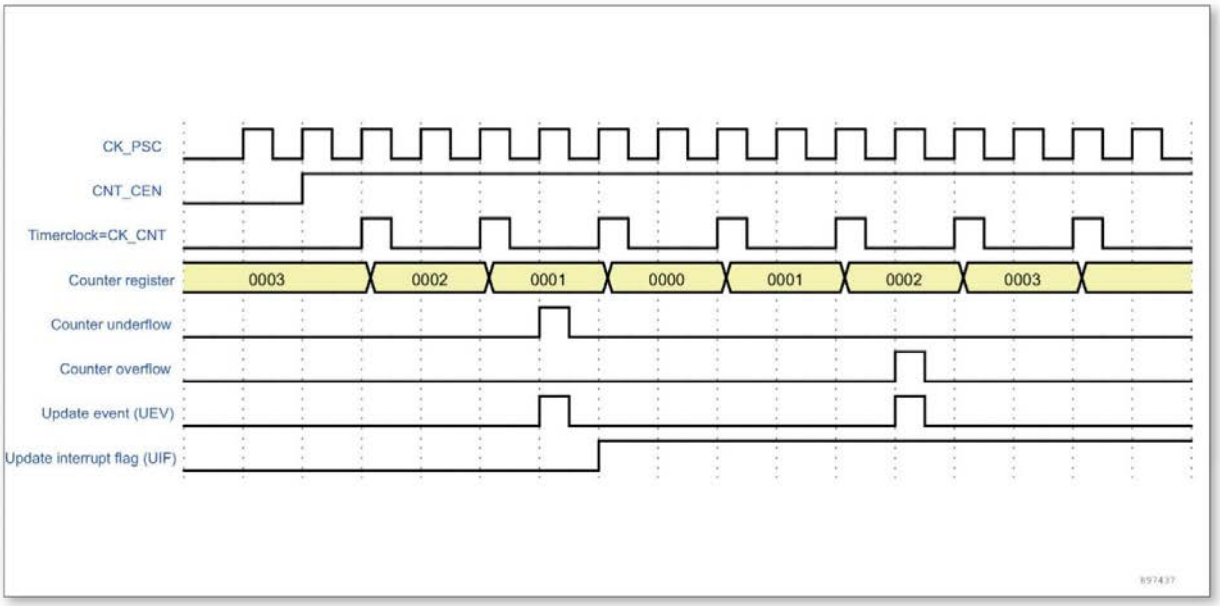


Figure 9-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR=0x03

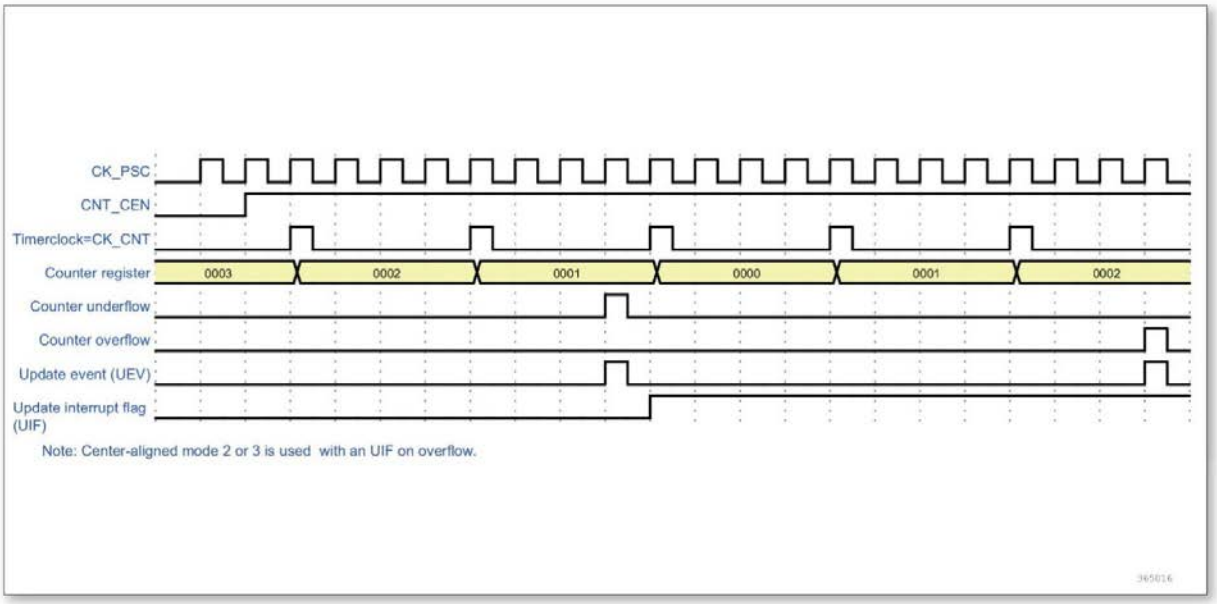


Figure 9-18 Counter timing diagram, internal clock divided by N

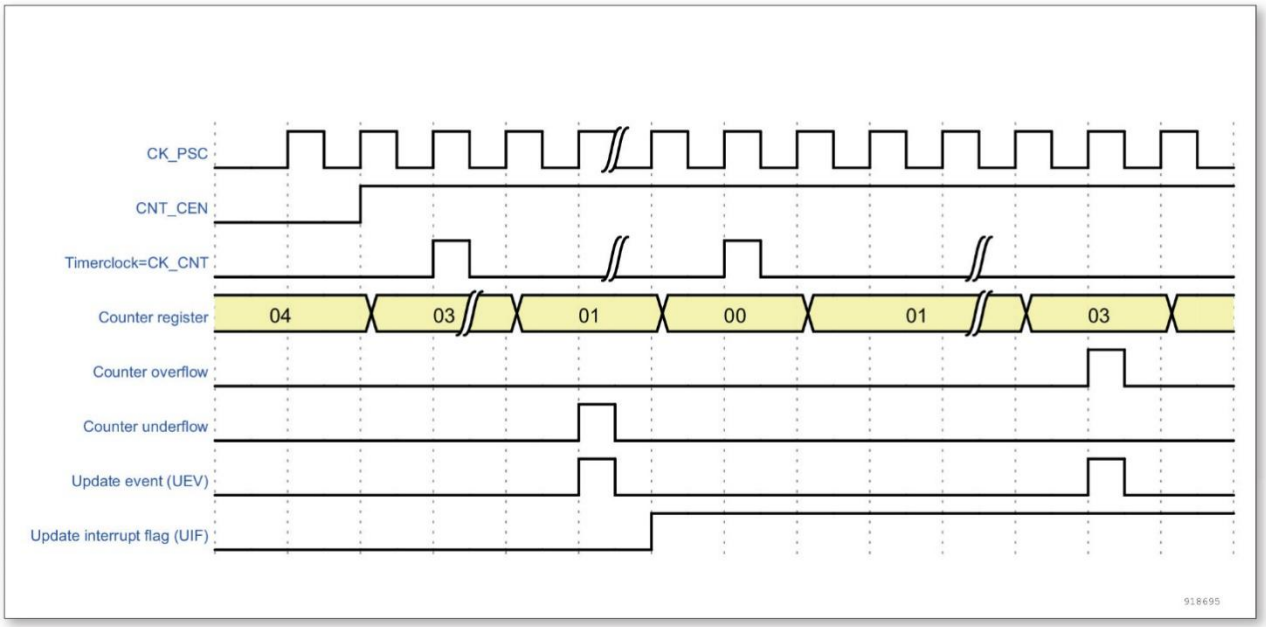


Figure 9-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

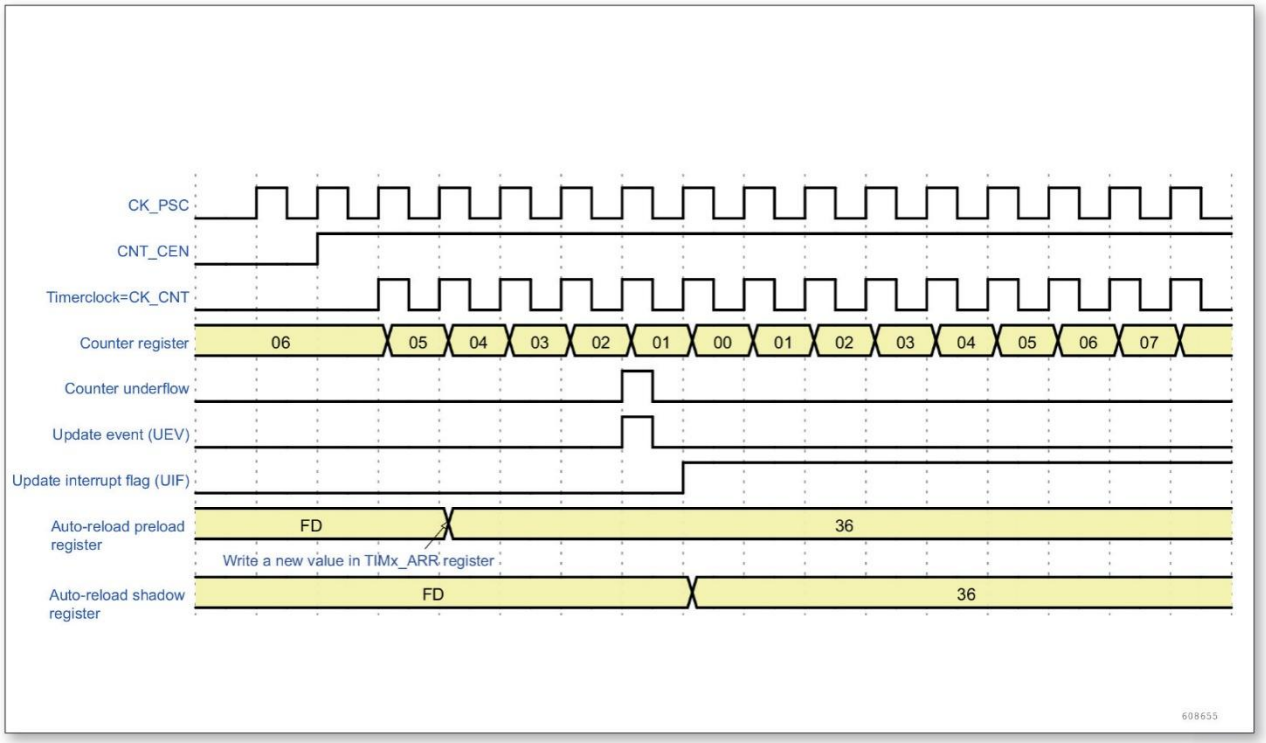
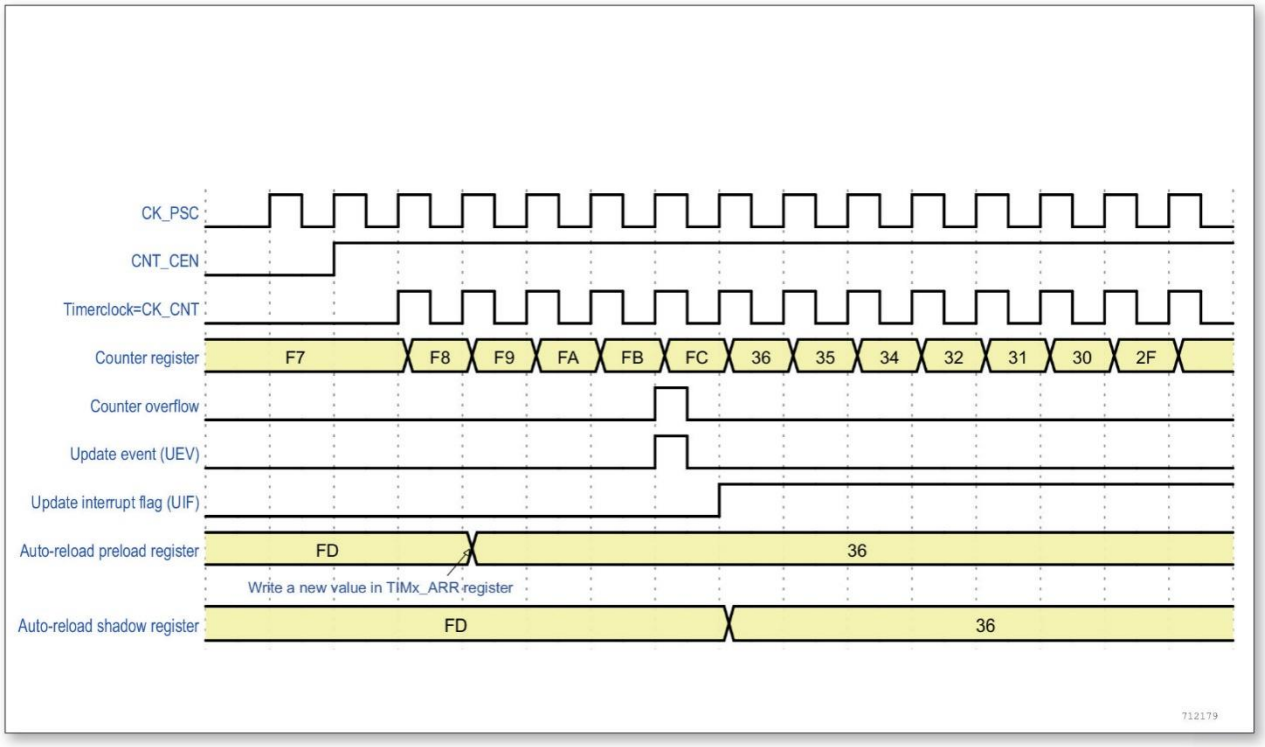


Figure 9-20 Counter timing diagram, update event with ARPE=1 (counter overflow)



9.3.3 Repetition counter

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

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

The repetition counter is decremented:

At each counter overflow in upcounting mode

At each counter underflow in downcounting mode

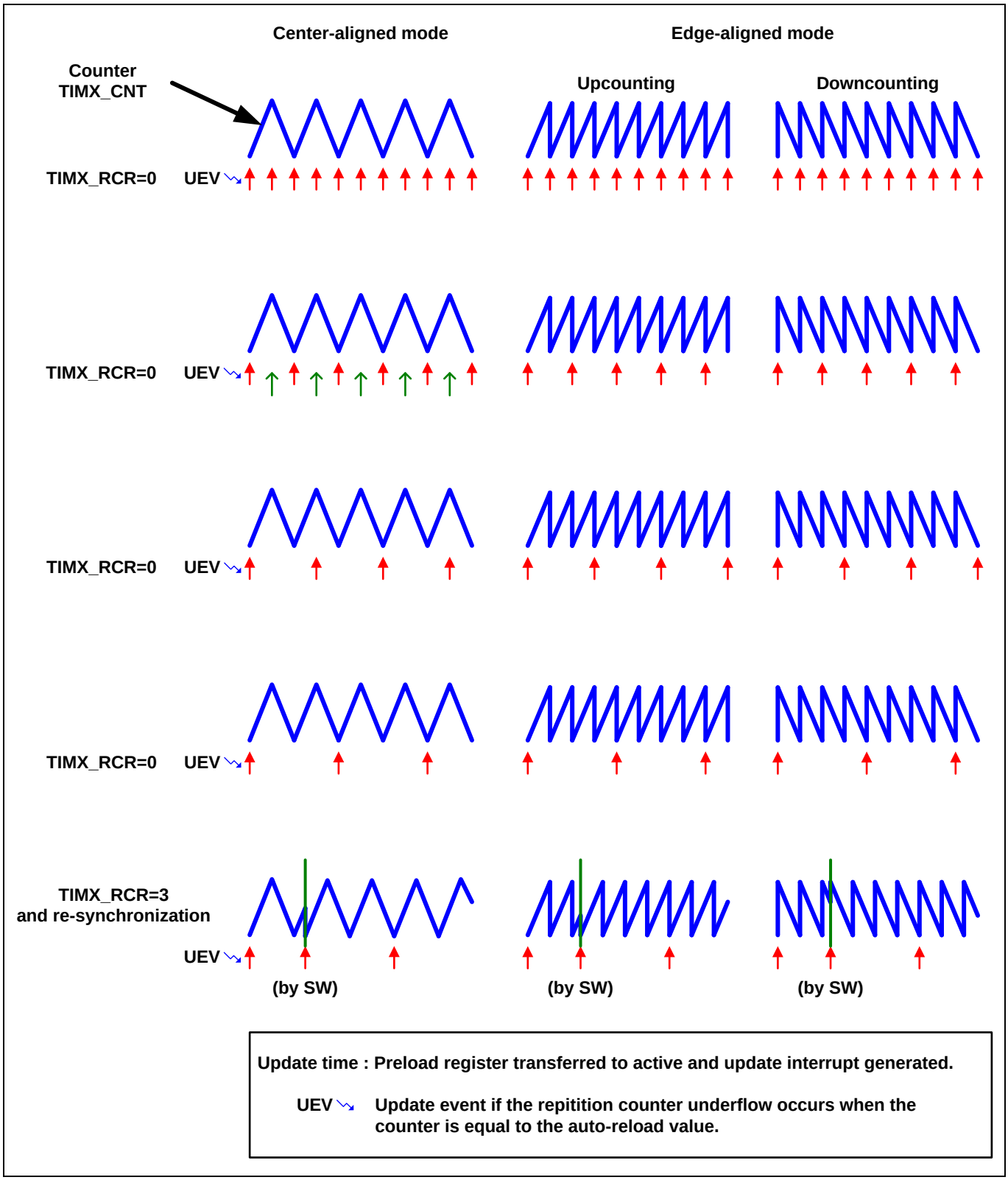
At each counter overflow and at each counter underflow in center-aligned mode.

Although this limits the maximum number of repetition to 128 PWM cycles, it makes it possible to update the duty cycle twice per PWM period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is 2xTck, due to the symmetry of the pattern.

The repetition counter is an auto-reload type; The repetition rate is maintained as defined by the TIMx_RCR register value (refer to Figure 48). 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.

The value of the repetition counter is written to a new REP_CNT register (Repetition counter value of real-time writing) in real time. This is used in the repetition counter modification mode, to move the update interrupt flag (UIF) to left by (REP-REP_CNT) phases (to right when the subtracted value is negative) by shifting UIF detection point in real time. These bits should be written after the update event UG is generated (note writing to REP_CNT before the update event is generated makes the displacement invalid).

Figure 9-21 Update rate examples depending on mode and TIMx_RCR register settings



9.3.4 Clock selection

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

Internal clock (CK_INT)

Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 3.

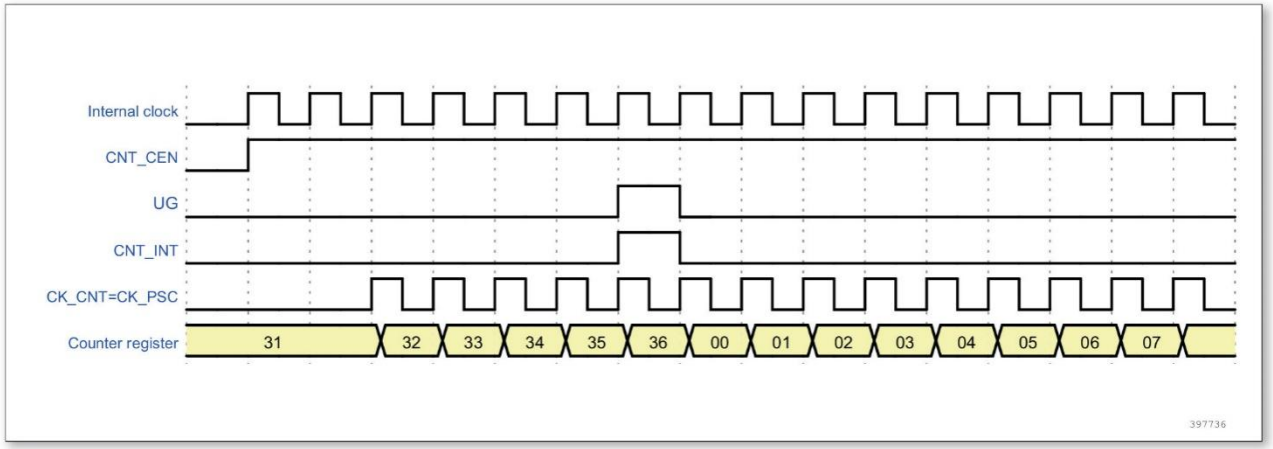
9.3.4.1 Internal clock source (CK_INT)

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

CK_INT.

The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

Figure 9-22 Control circuit in normal mode, internal clock divided by 1



9.3.5 Compare channel

Figure 9-23 Capture/compare channel 1 main circuit

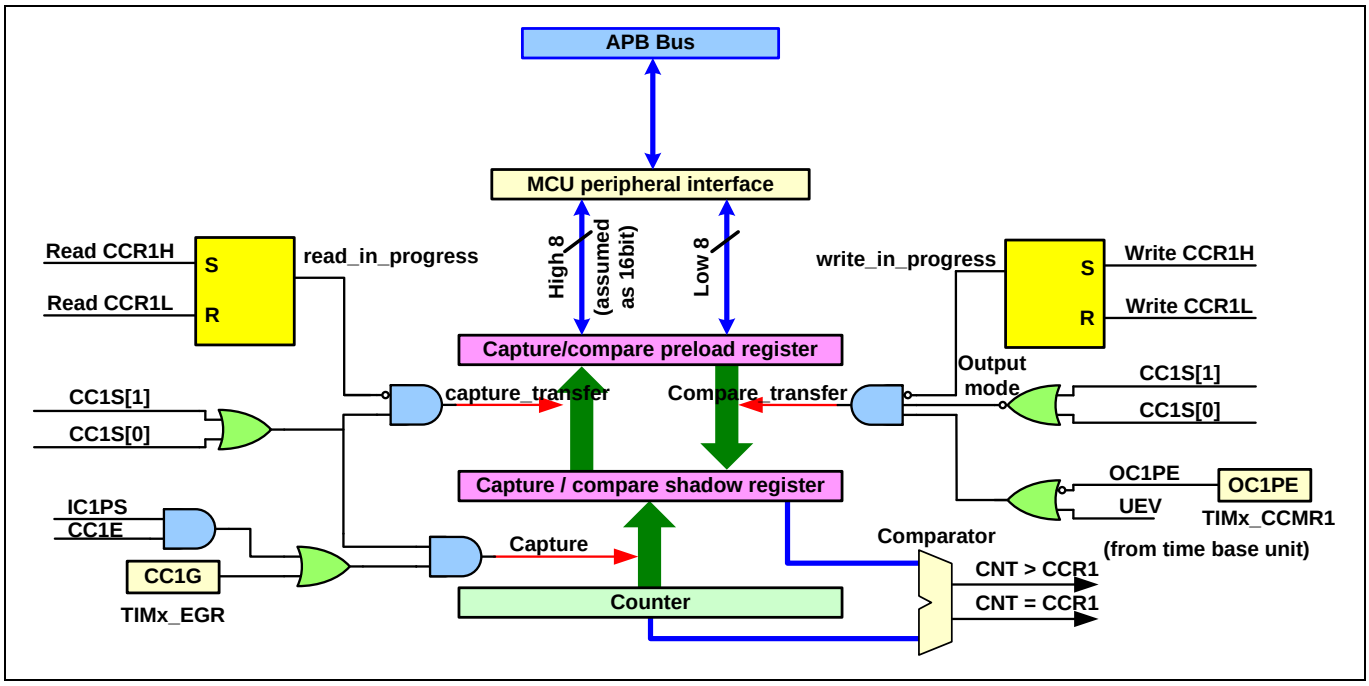


Figure 9-24 Output of the compare channel (channel 1 to 3)

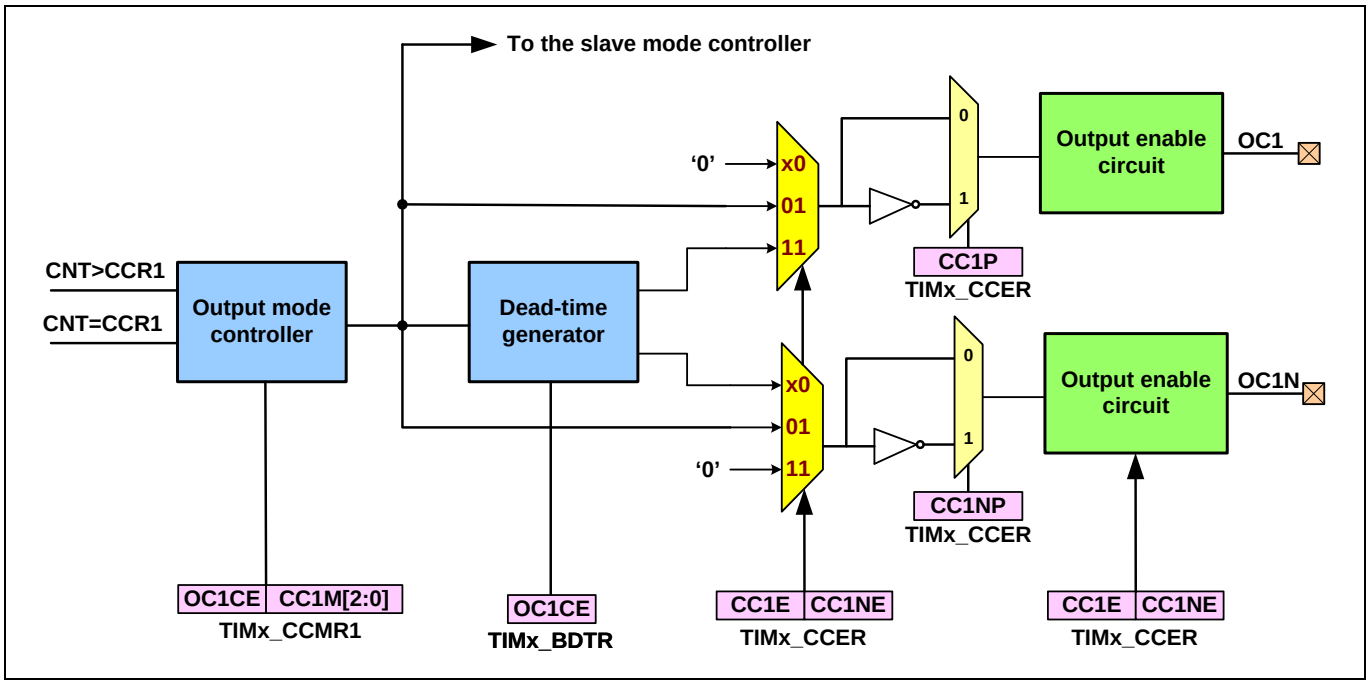
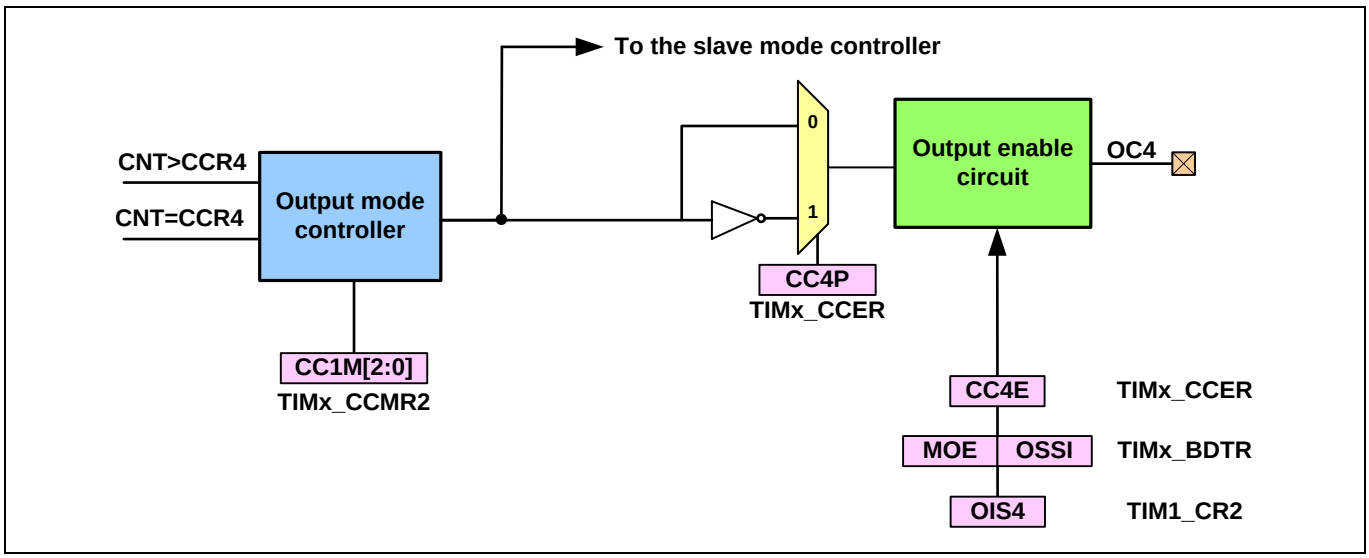


Figure 9-25 Output of the compare channel (channel 4)



The compare block is made of one preload register and one shadow register. Write and read always access the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

9.3.6 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx/OCxN) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write OCxM = 101 in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx gets 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 OCxM = 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 requests can be sent accordingly. This is described in the output compare mode section below.

9.3.7 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.
When a match is found between the compare register and the counter, the output compare function:

Assign the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit 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.

Set a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).

Generate an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be program 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 synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).
The output compare mode is configured as follows:

Select the counter clock (internal, external, and prescaler)

Write the desired data in the TIMx_ARR and TIMx_CCRx registers

Set the CCxIE bit if an interrupt request is to be generated

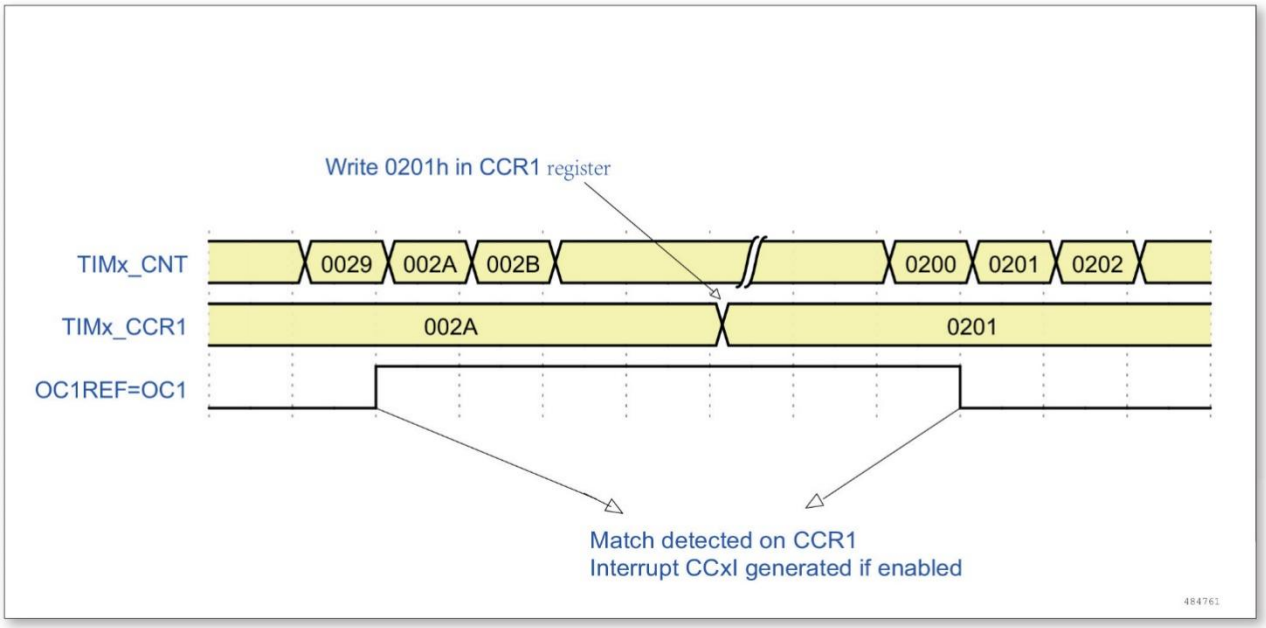
Select the output mode. For example:

- 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 9-26 Output compare mode, toggle on OC1



9.3.8 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the TIMx_CCRx register.
The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110'

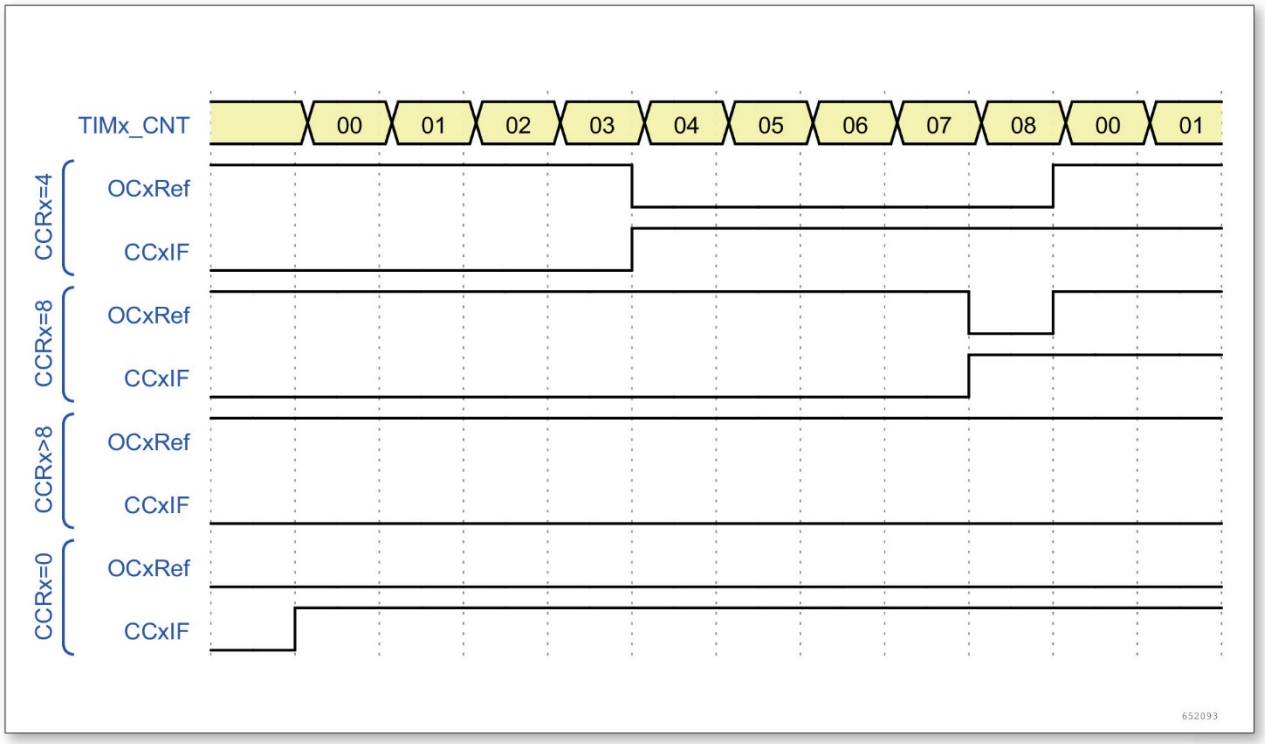
(PWM mode 1) or '111' (PWM mode 2) in the OCxM bit in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.
As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIMx_EGR register.
OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programd 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.

9.3.8.1 PWM edge-aligned mode

9.3.8.1.1 Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to the counter modes section.
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'. Below shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

Figure 9-27 Edge-aligned PWM waveforms (ARR = 8)



9.3.8.1.2 Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high. Refer to the counter modes section.
In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx; Else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at'1'. 0% PWM is not possible in this mode.

9.3.8.2 PWM center-aligned mode

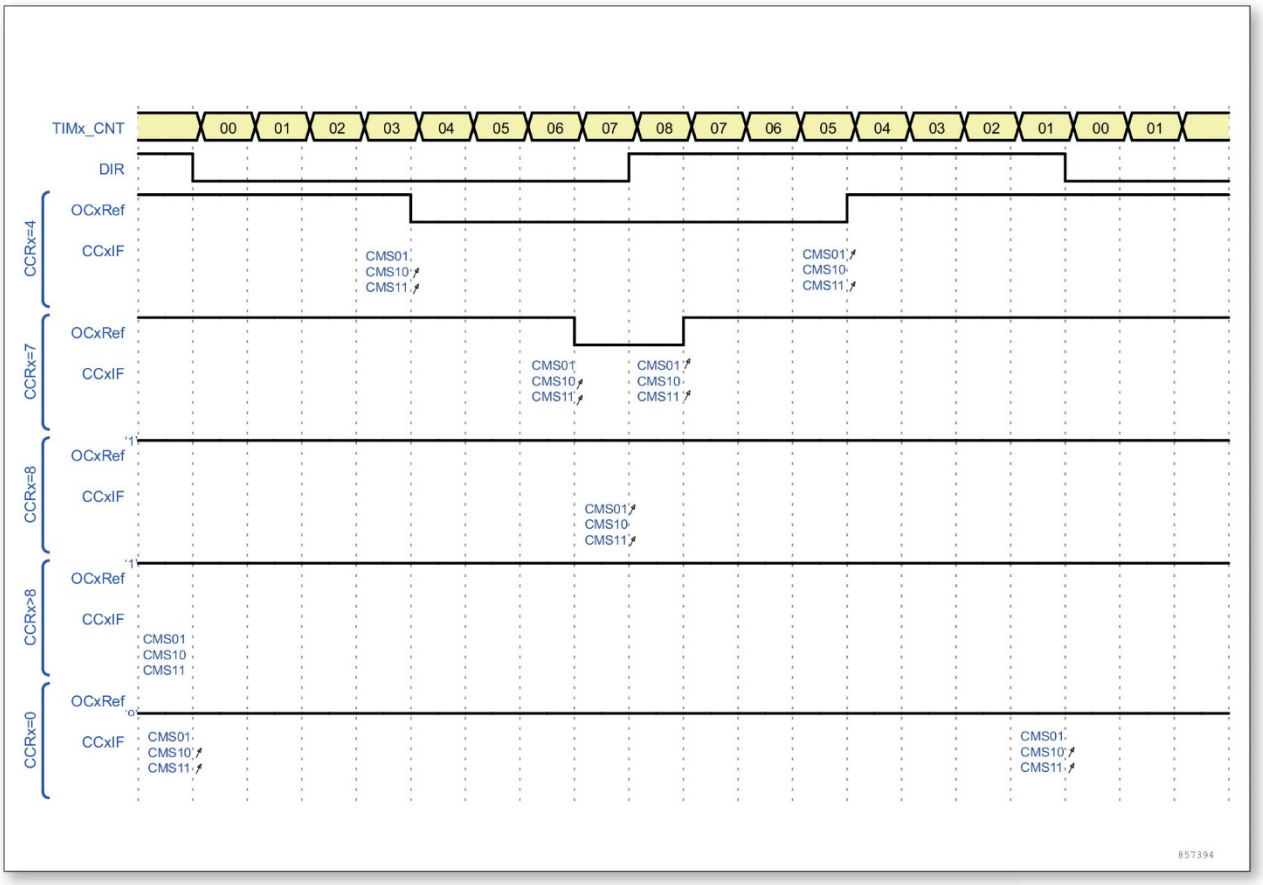
Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals).
The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software. Refer to the Center-aligned mode in counter modes section.
The figure below shows some center-aligned PWM waveforms in an example where:

TIMx_ARR = 8

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.

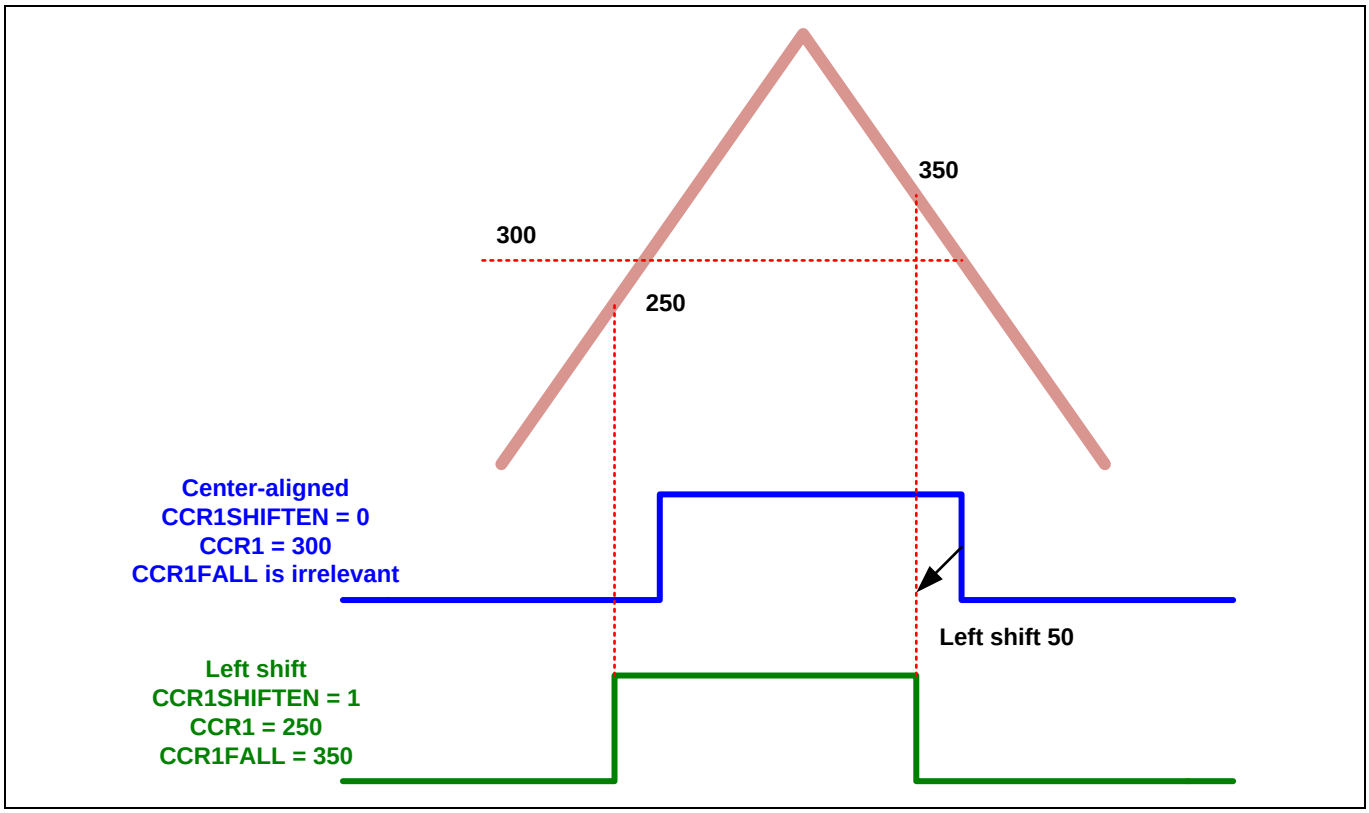
Figure 9-28 Center-aligned PWM waveforms (ARR = 8)



9.3.8.3 Phase shift in the PWM center-aligned mode

PDER (channel x output PWM phase shift enable bit) and CCRxFALL (channel x compare value when downcounting in the PWM center-aligned mode) are added to allow 5 channels to output PWM phase shift. To realize PWM outputs with programmable phase-shift waveform (left shift or right shift as required), the user should enable the PWM phase-shift function in the PDER register and set the CCRxFALL and CCRx.

Figure 9-29 Phase shift diagram



Hints on using center-aligned mode:

When entering the center-aligned mode, the current up-down counting configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.

Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:

- The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR).
- For example, if the counter was counting up, it will continue to count up.
- The direction is updated if the user writes 0 or writes 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.

9.3.9 Complementary output and dead-time insertion

The advanced-control timer (TIM1) can output two complementary signals and manage the switching-off and the switching-on instants of the outputs. This time is generally known as dead-time and it has to be adjusted depending on the devices connected to the output and their characteristics (delay of level-shifters, delay of power switches...)

User can select the polarity of the output (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. For more details, refer to

Table 9-3 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 a 10-bit dead-time generator in 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 9-30 Complementary output with dead-time insertion

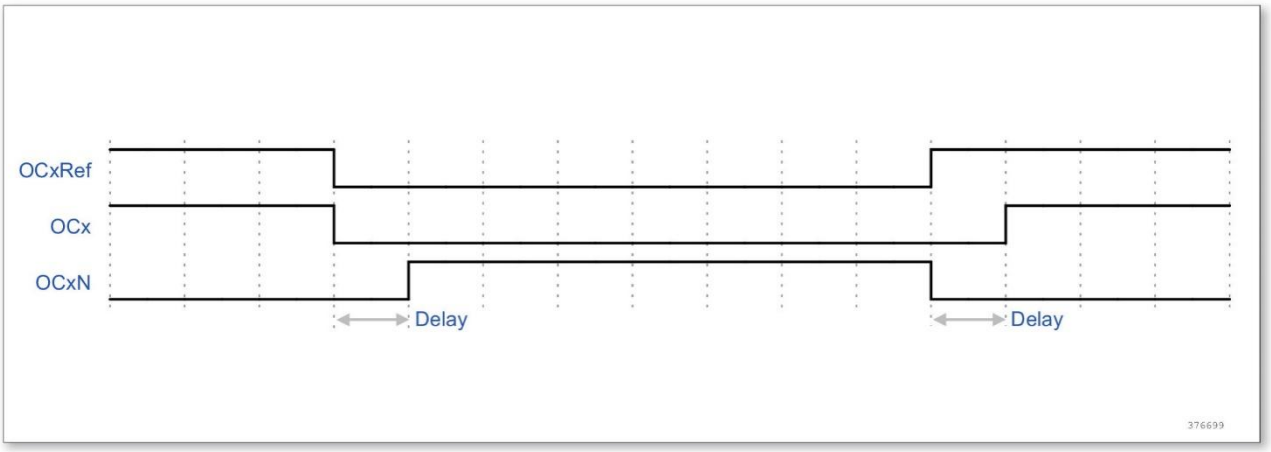


Figure 9-31 Dead-time waveforms with delay greater than the negative pulse

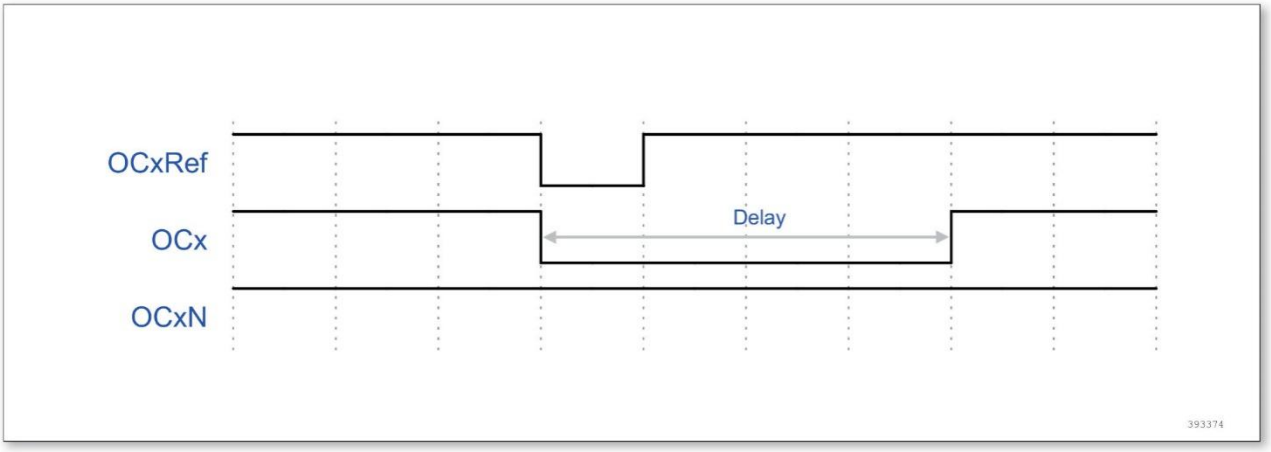
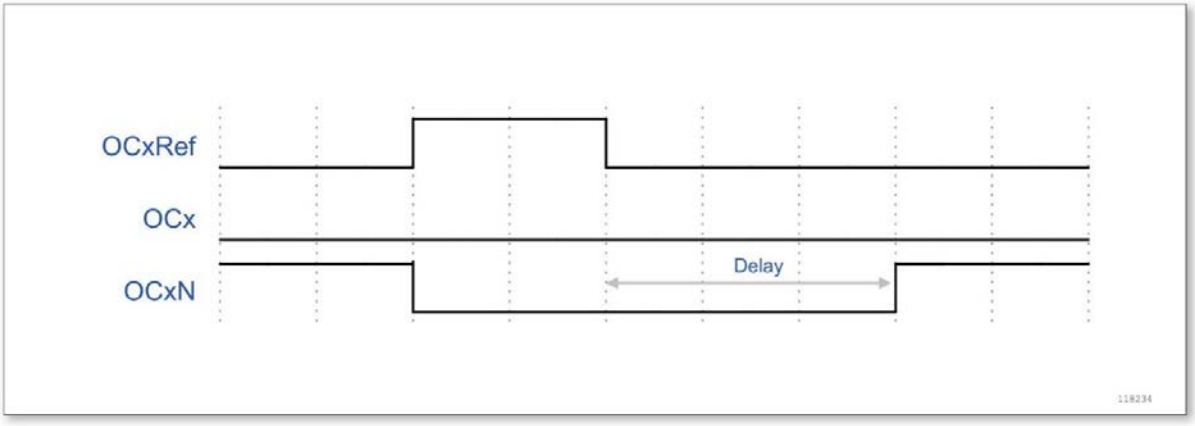


Figure 9-32 Dead-time waveforms with delay greater than the positive pulse



The dead-time delay is the same for each of the channels and is programmable with the DTG bits in the TIMx_BDTR register. Refer to the delay calculation in the register section.

9.3.9.1 Re-directing OCxREF to OCx or OCxN

In output mode (forced, output compare or PWM), OCxREF can be re-directed to the OCx output or to OCxN output by configuring the CCxE and CCxNE bits in the TIMx_CCER register.
This allows the user to send a specific waveform (such as PWM or static active level) on one output while the complementary remains at its inactive level. Other possibilities are to have both outputs at inactive level or both outputs active and complementary with dead-time.
Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it is not complemented and becomes active as soon as OCxREF is high. For example, if CCxNP = 0 then OCxN = OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE = 1), OCx becomes active when OCxREF is high whereas OCxN is complemented and becomes active when OCxREF is low.

9.3.10 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to corresponding control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to output control bits for complementary OCx and OCxN channels with break feature in the table of registers.
The break source can be either the break input pin or a clock failure event, generated by the Clock Security System (CSS), from the Reset Clock Controller.
After the system reset, the break circuit is disabled and the MOE bit is low. User can enable the break function by setting the BKE bit in the TIMx_BDTR register. The break input polarity can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified at the same time.
Because MOE falling edge can be asynchronous, a resynchronization circuit has been inserted between the actual signal (acting on the outputs) and the synchronous control bit (accessed in the TIMx_BDTR register). It results in some delays between the asynchronous and the synchronous signals. In particular, if MOE is written to 1 whereas it was low, a delay (dummy instruction) must be inserted before reading it correctly. This is because the user writes an asynchronous signal, but reads a synchronous signal.
When a break occurs (selected level on the break input):

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 programd 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 output is first put in reset state, namely 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 programd in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 CK_TIM clock cycles).
- If OSSI = 0 then the timer releases the enable outputs, else the enable outputs remain; Or the enable outputs become high as soon as one of the CCxE or CCxNE bits is high.

An interrupt can be generated if the BIE bit in the TIMx_DIER register is set and the break status flag (BIF bit in the TIMx_SR register) is set to '1'.

If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance.

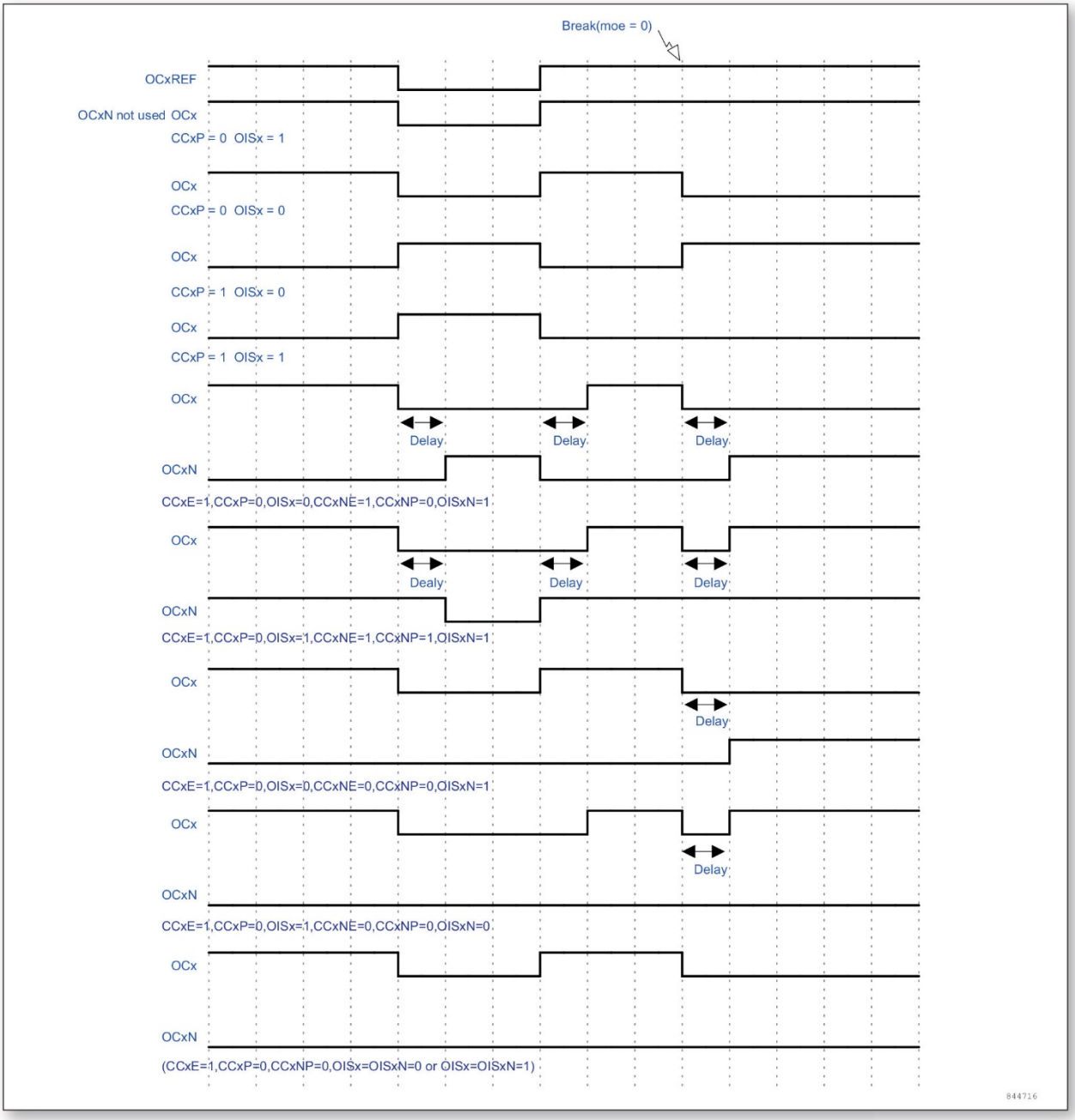
Else, MOE remains low until it is written to '1' again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.
Note: The break input is acting on level. Thus, the MOE cannot be set while the break input is active (neither automatically nor by software). In the meantime, the status flag BIF cannot be cleared.

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

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

The figure below shows an example of behavior of the output in response to a break:

Figure 9-33 Output behavior in response to a break



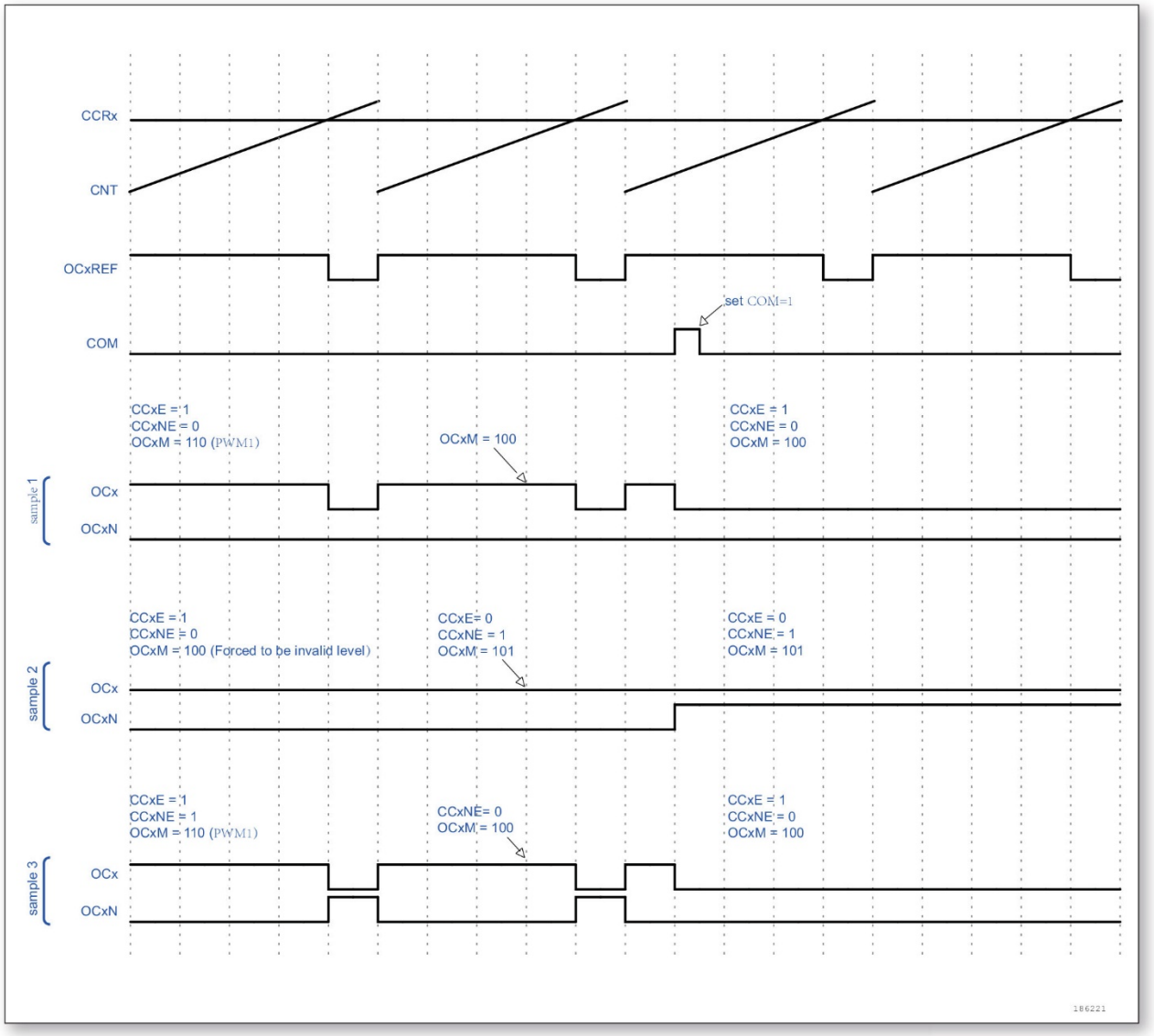
9.3.11 6-step PWM generation

When complementary output is used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. The user can thus program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIMx_EGR register or by hardware (on TRGI rising edge).

A flag is set when the COM event occurs (COMIF bit in the TIMx_SR register), which can generate an interrupt

(if the COMIE bit is set in the TIMx_DIER register).
The figure below describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programd configurations.

Figure 9-34 6-step generation, COM example (OSSR=1)



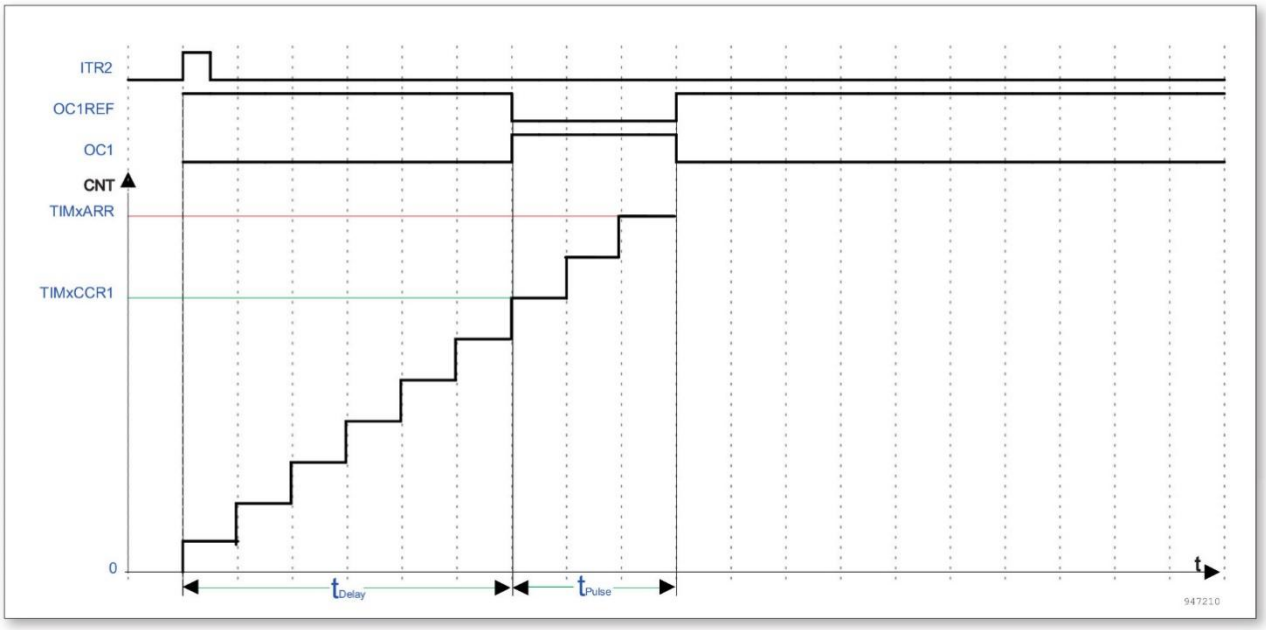
9.3.12 One-pulse mode

Onepulse 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. Select Onepulse mode by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.
A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)

In downcounting: CNT > CCRx

Figure 9-35 Example of one pulse mode



For example, the user may want to generate a positive pulse on OC2 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the ITR2 input pin.

Configure ITR2 as trigger source:

Configure TS=010 in the TIMx_SMCR register, and ITR2 serves as the trigger (TRGI) of the slave mode controller.

Configure SMS=110 in the TIMx_SMCR register, select the trigger mode, and ITR1 enables the counter to operate.

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

tDELAY is defined by the value written in the TIMx_CCR1 register.

tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR-TIMx_CCR1).

Assuming that a waveform from 0 to 1 is generated when a comparison match occurs, and a waveform from 1 to 0 is generated when the counter reaches the preload value; First, set OC1M=111 in the TIMx_CCMR1 register, entering PWM mode 2; Optionally enable preload registers as needed: set OC1PE=1 in TIMx_CCMR1 register and ARPE in TIMx_CR1 register; Then fill the comparison value in the TIMx_CCR1 register and fill the auto load value in the TIMx_ARR register, set the UG bit to generate an update event, and then wait for an external trigger event on ITR2. In this example, CC1P=1.

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

The user only wants one pulse, so OPM=1 must be written 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, if a waveform with a minimum delay is required to be output, set OCxFE bit inTIMx_ CCMRx register; At this point, OCxREF (and OCx) is forced to respond directly to the excitation without relying on the comparison result, and the output waveform is the same as the waveform at the time of comparison and matching. OCxFE only works when the channel is configured for PWM1 and PWM2 modes.

9.3.13 Timer synchronization

TIMx timers can synchronize with an internal trigger in multiple modes: reset mode, gated mode, and trigger mode.

9.3.13.1 Slave mode: reset mode

When a trigger input event occurs, the counter and its prescaler can be reinitialized, and if URS bit of the TIMx_CR1 register is low, and an update event UEV is generated. Then, all pre-loaded registers (TIMx_ARR, TIMx_CCRx) are updated.

For example, ITR2 triggers a counter restart:

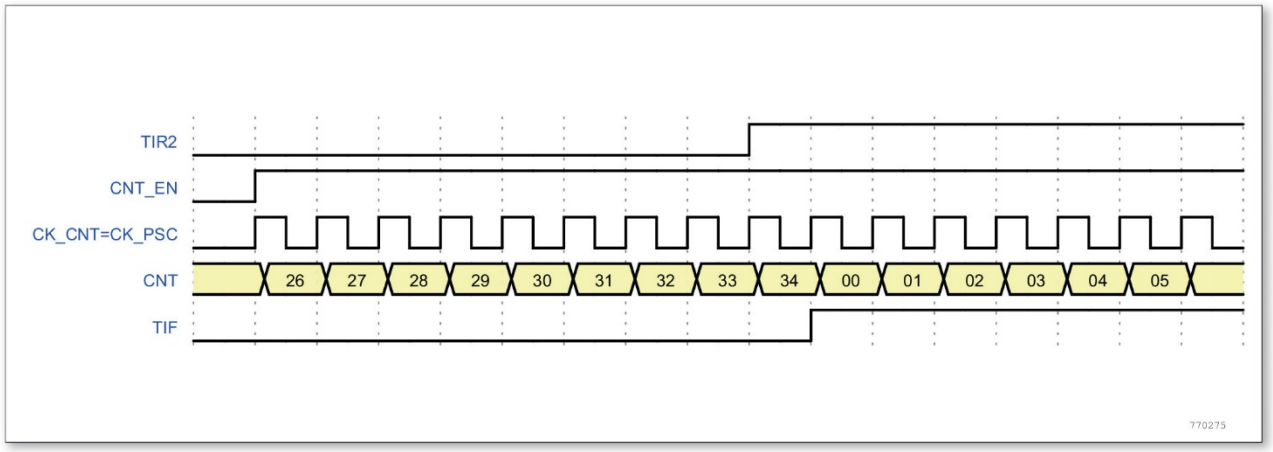
Configure SMS=100 in TIMx_SMCR register, select the reset mode as the salve mode; Configure TS=010 in the TIMx_SMCR register, and select ITR2 as the trigger input of the synchronization counter.

Configure DIR=0 in the TIMx_CR1 register, and select the counting direction as the upcounting; Configure PSC=0 without frequency division; Configure CEN=1 to enable the counter.

The counter starts counting on the internal clock, then behaves normally until a rising edge emerged on ITR2. At that time, 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 the TIE bit (interrupt enable) is set in TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR = 0x36.

Figure 9-36 Control circuit in reset mode



9.3.13.2 Slave mode: gated mode

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

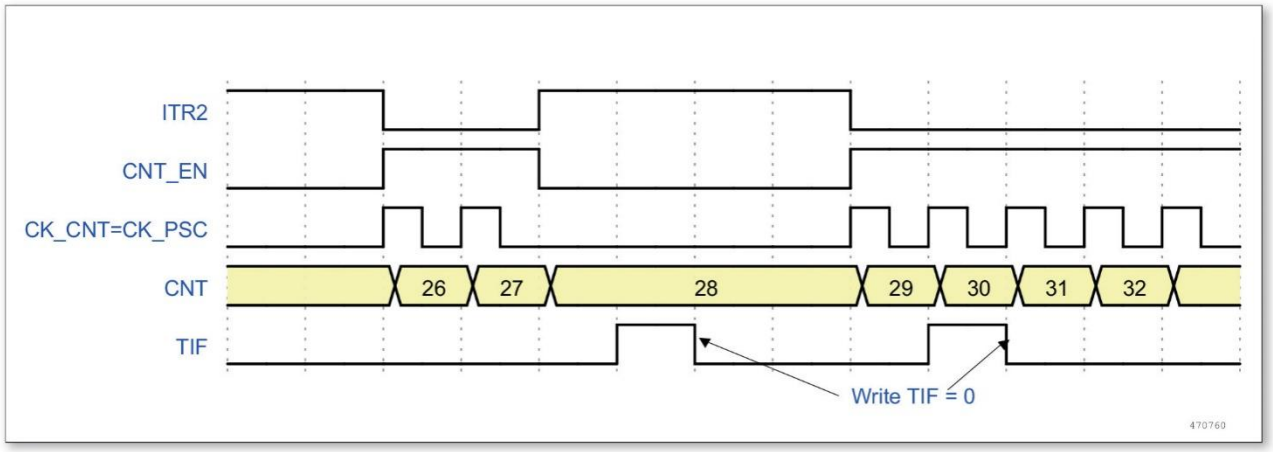
In the following example, the counter counts only when ITR2 is low:

Configure SMS=101 in the TIMx_SMCR register, select the gated mode as the slave mode; Configure TS=010 in the TIMx_SMCR register and select ITR2 as the trigger input of the synchronization counter.

Configure DIR=0 in the TIMx_CR1 register, and select the counting direction as the upcounting; Configure PSC=0 without frequency division; Configure CEN=1 to enable the counter.

As long as ITR2 is low, the counter starts counting based on the internal clock, and stops counting when ITR2 becomes high. Set TIF flag in TIMx_SR when the counter starts or stops.

Figure 9-37 Control circuit in gated mode



9.3.13.3 Slave mode: trigger mode

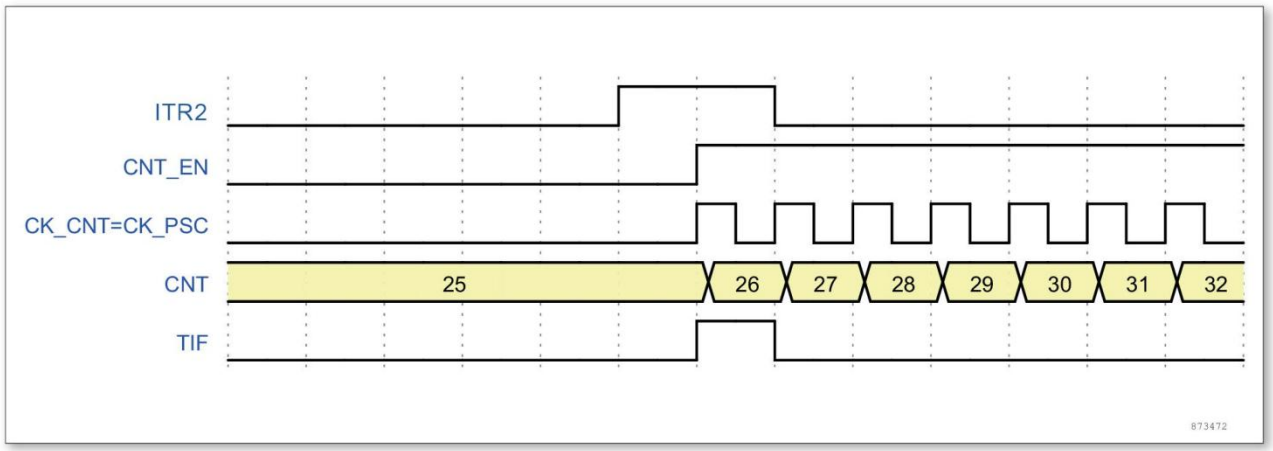
The counter can start in response to an event on a selected input.
For example, the counter starts counting at the rising edge of ITR2:

Configure SMS=110 of the TIMx_SMCR register, select the trigger mode as the slave mode; Configure TS=010 of the TIMx_SMCR register, select ITR2 as the trigger input of the counter.

Configure DIR=0 of the TIMx_CR1 register, and select the counting direction upcounting; Configure PSC=0 without frequency division.

When a rising edge appears in ITR2, the counter starts counting under the internal clock drive, while setting the TIF flag.

Figure 9-38 Control circuit in trigger mode



9.3.14 Debug mode

When the microcontroller enters debug mode (CPU core halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module. For more details, refer to subsequent debug section.

9.4 Register

Table 9-1 TIM1 register overview

Offset	Acronym	Register Name	Reset
0x00	TIM1_CR1	Control Register 1	0x0000
0x04	TIM1_CR2	Control Register 2	0x0000
0x08	TIM1_SMCR	Slave Mode Control Register	0x0000
0x0C	TIM1_DIER	Interrupt Enable Register	0x0000 0000
0x10	TIM1_SR	Status Register	0x0000 0000
0x14	TIM1_EGR	Event Generation Register	0x0000 0000
0x18	TIM1_CCMR1	Compare Mode Register 1	0x0000
0x1C	TIM1_CCMR2	Compare Mode Register 2	0x0000
0x20	TIM1_CCER	Compare Enable Register	0x0000
0x24	TIM1_CNT	Counter	0x0000
0x28	TIM1_PSC	Prescaler	0x0000
0x2C	TIM1_ARR	Auto Reload Register	0x0000
0x30	TIM1_RCR	Repeat Count Register	0x0000
0x34	TIM1_CCR1	Compare Register 1	0x0000
0x38	TIM1_CCR2	Compare Register 2	0x0000
0x3C	TIM1_CCR3	Compare Register 3	0x0000
0x40	TIM1_CCR4	Compare Register 4	0x0000
0x44	TIM1_BDTR	Break And Dead-Time Register	0x0000 0000
0x54	TIM1_CCMR3	Compare Mode Register 3	0x0000
0x58	TIM1_CCR5	Compare Register 5	0x0000
0x5C	TIM1_PDER	PWM Phase Shift Enable Register	0x0000
0x60 ~ 0x70	TIM1_CCRxFALL	PWM Phase Shift Count Down Compare Register	0x0000

9.4.1 TIM1_CR1 Control Register 1

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

Bit	Field	Description
15：10	Reserved	Reserved, must be kept at reset value.
9：8	CKD	Clock division 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 (TIx). 00：$t_{DTS} = t_{INT\_CK}$ 01：t_{DTS} =2x t_{INT_CK} 10：t_{DTS} =4x t_{INT_CK} 11：Reserved, do not program this value
7	ARPE	Auto reload preload enable 0: Disable the shadow register of TIM1_ARR register 1: Enable the shadow register of TIM1_ARR register
6：5	CMS	Center alignment mode selection 00: Edge alignment mode. Count direction depends on DIR bit 01: Central alignment mode 1. The counter alternatively conducts up and down count. The channel is in output mode. Only in down count, compare interrupt flag bit is set

		10: Central alignment mode 2. The counter alternatively conducts up and down count. The channel is in output mode. Only in up count, compare interrupt flag bit is set 11: Central alignment mode 3. The counter alternatively conducts up and down count. The channel is in output mode. In up and down count, compare interrupt flag bit is set Note: During count, the alignment mode change is disabled.
4	DIR	Count direction 0: up count 1: down count Note: When the counter is configured as central alignment mode, the bit is read only.
3	OPM	one-pulse mode 0: Disable one-pulse mode. In case of update event, the counter count continues 1: Enable one-pulse mode. In case of the next update event (clear CEN bit), the counter count stops
2	URS	Update request source This bit is set and cleared by software, select update event source. 0: The event below may generate a update interrupt request: - Counter overflow/underflow - Set UG bit - Update generation through the slave mode controller 1: Only in counter overflow/underflow, generate update interrupt request
1	UDIS	Update disable This bit is used to enable or disable the update event 0: Update event (UEV) enabled. 1: Update event 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 EGR.UG bit is set, the counter is reinitialized if a hardware reset is received from the slave mode controller.
0	CEN	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated 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.

9.4.2 TIM1_CR2 Control Register 2

Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	Res.	MMS			Res.	CCUS	Res.	CCPC
	rw	rw	rw	rw	rw	rw	rw		rw				rw		rw

Bit	Field	Description
15	Reserved	Reserved, must be kept at reset value.
14	OIS4	Output idle state 4 (OC4 output). Refer to OIS1 bit.
13	OIS3N	Output idle state 3 (OC3N output). Refer to OIS1N bit.
12	OIS3	Output idle state 3 (OC3 output). Refer to OIS1 bit.
11	OIS2N	Output idle state 2 (OC2N output). Refer to OIS1N bit.
10	OIS2	Output idle state 2 (OC2 output). Refer to OIS1 bit.
9	OIS1N	(Output idle state 1) (OC1N output) 0: In case of MOE =0, OC1N =0 after dead-time 1: In case of MOE =0, OC1N =1 after dead-time Note: After setting LOCK (TIM1_BKR register) level 1, 2 or 3, this bit cannot be changed.
8	OIS1	(Output idle state 1) (OC1 output) 0: In case of MOE=0, if OC1N is implemented, OC1=0 after dead-time 1: In case of MOE=0, if OC1N is implemented, OC1 = 1 after dead-time

		Note: After setting LOCK (TIM1_BKR register) level 1, 2 or 3, this bit cannot be changed.
7	Reserved	Reserved, must be kept at reset value.
6：4	MMS	Master mode selection These bits control TRGO signal selection, used to select the sync information sent to the slave timers in master mode: 000: Reset TIM1_EGR register UG bit generate one pulse trigger output (TRGO). 001: Enable The Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected. 010: Update Update event is selected as TRGO. 011: Compare When a compare match occurred, send a positive pulse as TRGO. 100: Compare OC1REF signal is used as trigger output (TRGO) 101: Compare OC2REF signal is used as trigger output (TRGO) 110: Compare OC3REF signal is used as trigger output (TRGO) 111: Compare OC4REF signal is used as trigger output (TRGO)
3	Reserved	Reserved, must be kept at reset value.
2	CCUS	Compare control update selection 0: CCPC=1, they are updated by setting the COMG=1 only. 1：CCPC=1, they are updated by setting the COMG=1 or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	Reserved, must be kept at reset value.
0	CCPC	Compare preloaded control bit 0: CCxE, CCxNE and OCxM bit preload disable 1: CCxE, CCxNE and OCxM bit preload enable Note: This bit acts only on channels that have a complementary output.

9.4.3 TIM1_SMCR Slave Mode Control Register

Address offset: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MSM	TS			Res.	SMS		
								rw	rw				rw		

Bit	Field	Description
15：8	Reserved	Reserved, must be kept at reset value.
7	MSM	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	Trigger selection Trigger input source selection. 000：Internal trigger 0 (ITR0) 001：Internal trigger 1 (ITR1) 010：Internal trigger 2 (ITR2) 011：Internal trigger 3 (ITR3) 100：Reserved 101：Reserved

		110：Reserved 111：Reserved Note: After the slave mode enable, these bits cannot be changed
3	Reserved	Reserved, must be kept at reset value.
2：0	SMS	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. 000: Close slave mode - In case of CEN =1, the prescaler is directly driven by the internal clock. 001: Reserved 010: Reserved 011: Reserved 100: Reset mode -The rising edge of the selected trigger input (TRGI) reinitialize the counter and generate an update event. 101: Gate mode - When the trigger input (TRGI) is high, the counter count begins. When the trigger input turns low, the counter count stops (but without reset). The counter start and stop are controlled. 110: Trigger mode -The counter starts in the rising edge of the trigger input TRGI (but without reset). Only the counter start is controlled. 111: External clock mode 1 -The rising edge of the selected trigger input (TRGI) drives the counter.

Table 9-2 TIMx internal trigger connection

Slave timer	ITR0	ITR1	ITR2	ITR3
TIM1	-	-	TIM3	-
TIM3	TIM1	-	-	-

9.4.4 TIM1_DIER Interrupt Enable Register

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5IE
															rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31：17	Reserved	Reserved, must be kept at reset value.
16	CC5IE	Compare 5 interrupt enable 0: Compare 5 interrupt disable 1: Compare 5 interrupt enable
15：8	Reserved	Reserved, must be kept at reset value.
7	BIE	Break interrupt enable 0: Break interrupt disable 1: Break interrupt enable
6	TIE	Trigger interrupt enable 0: Break interrupt disable 1: Break interrupt enable
5	COMIE	Enable COM interrupt 0: COM interrupt disable 1: COM interrupt enable
4	CC4IE	Enable compare 4 interrupt 0: Compare interrupt 4 disable 1: Compare interrupt 4 enable
3	CC3IE	Enable compare 3 interrupt 0: Compare interrupt 3 disable 1: Compare interrupt 3 enable
2	CC2IE	Enable compare 2 interrupt

		0: Compare interrupt 2 disable 1: Compare interrupt 2 enable
1	CC1IE	Enable compare 1 interrupt 0: Compare interrupt 1 disable 1: Compare interrupt 1 enable
0	UIE	Enable update interrupt 0: Update event interrupt disable 1: Update event interrupt enable

9.4.5 TIM1_SR Status Register

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5IF
															rw0c
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
								r_w0c							

Bit	Field	Description
31：17	Reserved	Reserved, must be kept at reset value.
16	CC5IF	Compare 5 interrupt flag Refer to CC1IF description.
15：8	Reserved	Reserved, must be kept at reset value.
7	BIF	Break interrupt flag When the break input is active, the bit is set by hardware. If the Break input is inactive, the bit is set as 0 by software 0: No break event occurred 1: Detect active level on Break input
6	TIF	Trigger interrupt flag In case of trigger event (When the slave mode counter is in any other mode except for the gate mode, detect the active edge in TRGI input terminal, or detect any edge in gate mode), the bit is set by hardware. It's cleared by software. 0: No trigger event occurred 1: Trigger interrupt pending
5	COMIF	COM interrupt flag In case of COM event (compare control bit: CCxE, CCxNE, OCxM updated), the bit is set by hardware. It is cleared by software. 0：No COM event occurred 1: COM interrupt pending
4	CC4IF	Compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	Compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	Compare 2 interrupt flag Refer to CC1IF description.
1	CC1IF	Compare 1 interrupt flag When the counter value and compare value match, the bit is set by hardware except for the central alignment mode (the bit is set in the central alignment mode according to TIM1_CR1.CMS [1:0]). It's cleared by software. 0: No match 1: TIM1_CNT value and TIM1_CCR1 value match
0	UIF	Update interrupt flag In case of update event, the bit is set by hardware. It's cleared by software. 0: No update interrupt occurred 1: update interrupt pending This bit is set by hardware when the registers are updated: - In case of TIM1_CR1 register UDIS=0 and REP_CNT=0, when the counter generates overflow/underflow.

		-In case of TIM1_CR1 register UDIS=0, URS=0, when TIM1_EGR register UG =1. - In case of TIM1_CR1 register UDIS=0, URS=0, when update generation through the slave mode controller.

9.4.6 TIM1_EGR Event Generation Register

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5G
															w
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved								BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								w	w	w	w	w	w	w	w

Bit	Field	Description
31：17	Reserved	Reserved, must be kept at reset value.
16	CC5G	Compare 5 generation Refer to CC1G description.
15：8	Reserved	Reserved, must be kept at reset value.
7	BG	Break generation 0: No action 1: generate a break event. MOE =0, BIF =1; when producing the corresponding interrupt, generate the corresponding interrupt. It's cleared by hardware.
6	TG	Trigger generation 0: No action 1: generate a trigger event. TIM1_SR register TIF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. It's auto cleared by hardware.
5	COMG	Compare control update generation 0: No action 1: Compare event control update. It's auto cleared by hardware. In case of CCPC =1, enable update CCxE, CCxNE and OCxM bit. Note: This bit is only active for the complementary output channel.
4	CC4G	Compare 4 generation Refer to CC1G description.
3	CC3G	Compare 3 generation Refer to CC1G description.
2	CC2G	Compare 2 generation Refer to CC1G description.
1	CC1G	Compare 1 generation Refer to CC1G description. This bit is set by software. It generates a compare event, and is auto cleared by hardware. 0: No action 1：Generate a compare event in channel CC1: Set CC1IF =1. When enabling the corresponding interrupt, generate the corresponding interrupt.
0	UG	Update event generation 0：No action 1: Initialize the counter, and generate an update event. It's auto cleared by hardware. When selecting the central alignment or up counting mode, the counter is clear; otherwise (down counting mode) the counter auto reload value is loaded. The prescaler counter is cleared at the same time.

9.4.7 TIM1_CCMR1 Compare Mode Register 1

Address offset: 0x18

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC2M			OC2PE	OC2FE	Reserved			OC1M			OC1PE	OC1FE	Reserved	
	rw			rw	rw				rw			rw	rw		

Bit	Field	Description
15	Reserved	Reserved, must be kept at reset value.
14：12	OC2M	Channel 2 output and compare mode Refer to OC1M description.
11	OC2PE	Channel 2 output and compare preload enable Refer to OC1PE description.
10	OC2FE	Channel 2 output and compare quick enable Refer to OC1FE description.
9：7	Reserved	Reserved, must be kept at reset value.
6：4	OC1M	Channel 1 output compare mode The bit has defined the output reference signal OC1REF action. OC1REF has determined OC1 and OC1N value. OC1REF is active at high level. The active level of OC1, OC1N depends on CC1P and CC1NP bit. 000：Freeze. TIM1_CCR1 and TIM1_CNT compare results has no effect on OC1REF. 001: Set as high when configuration. When TIM1_CNT value and TIM1_CCR1 value are same, enforce OC1REF as high level 010: Set as low when configuration. When TIM1_CNT value and TIM1_CCR1 value are same, enforce OC1REF as low level 011: Toggle when match. When TIM1_CCR1=TIM1_CNT, OC1REF toggle. 100: Enforce as low. Enforce OC1REF at low level 101: Enforce as high. Enforce OC1REF at high level 110：PWM mode 1. During up count, in case of TIM1_CNT<TIM1_CCR1, enforce OC1REF is at high level. Or else, it's at low level. During down count, in case of TIM1_CNT > TIM1_CCR1, enforce OC1REF is at low level. Or else, it's at high level. 111：PWM mode 2. During up count, in case of TIM1_CNT<TIM1_CCR1, channel 1 enforce OC1REF is at low level. Or else, it's at high level. During down count, in case of TIM1_CNT > TIM1_CCR1, enforce OC1REF is at high level. Or else, it's at low level. Note 1: In case of LOCK level 3 (TIM1_BDTR register LOCK bit), the bit cannot be changed. Note 2: In PWM mode 1 or PWM mode 2, only when the compare result changes or it changes over from the freeze mode to PWM mode in the output compare mode, OC1REF level may change.
3	OC1PE	Channel 1 output compare preload enable 0: Disable TIM1_CCR1 register preload function. The value written into TIM1_CCR1 register becomes active immediately 1: Enable TIM1_CCR1 register preload function. Only read and write the preload register. TIM1_CCR1 preload value becomes active at the update event Note 1: When the LOCK level is 3 (TIM1_BDTR register LOCK bit), the bit cannot be changed. Note 2: Only in the one-pulse mode (TIM1_CR1 register OPM= 1), it has no influence whether the preload register is set. Under other scenarios, it's required to set the preload register. Otherwise, the follow up action is not certain.
2	OC1FE	Channel 1 output compare quick enable In case of the bit is set 1, when the channel is configured as PWM mode, it speeds up response of the compare output to the trigger time. The output channel deems the active edge of the trigger input signal as one compare match. Therefore, OC is set as compare level, but is irrelevant to the compare result. 0: channel 1 output compare fast enable. 1: channel 1 output compare fast disable.
1：0	Reserved	Reserved, must be kept at reset value.

9.4.8 TIM1_CCMR2 Compare Mode Register 2

Address offset: 0x1C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC4M			OC4PE	OC4FE	Reserved			OC3M			OC3PE	OC3FE	Reserved	
	rw			rw	rw				rw			rw	rw		

Bit	Field	Description
15	Reserved	Reserved, must be kept at reset value.
14：12	OC4M	Channel 4 output compare mode Refer to OC3M description
11	OC4PE	Channel 4 output compare preload enable Refer to OC3PE description
10	OC4FE	Channel 4 output compare rapid enable Refer to OC3FE description
9：7	Reserved	Reserved, must be kept at reset value.
6：4	OC3M	Channel 3 output compare mode The bit has defined the output reference signal OC3REF action. OC3REF has determined OC3 and OC3N value. OC3REF is active at high level. The active level of OC3, OC3N depends on CC3P and CC3NP bit. 000：Freeze. TIM1_CCR3 and TIM1_CNT compare results has no effect on OC3REF. 001: Set as high when configuration. When TIM1_CNT value and TIM1_CCR3 value are same, enforce OC3REF as high level 010: Set as low when configuration. When TIM1_CNT value and TIM1_CCR3 value are same, enforce OC3REF as low level 011: Toggle when match. When TIM1_CCR3=TIM1_CNT, OC3REF toggle. 100: Enforce as low. Enforce OC3REF at low level 101: Enforce as high. Enforce OC3REF at high level 110: PWM mode 1. During up count, in case of TIM1_CNT<TIM1_CCR3, enforce OC3REF as high level, otherwise its low level; during down count, in case of TIM1_CNT > TIM1_CCR3, enforce OC3REF as low level, otherwise its high level. 111: PWM mode 2. During up count, in case of TIM1_CNT<TIM1_CCR3, enforce OC3REF as low level, otherwise its high level; during down count, in case of TIM1_CNT > TIM1_CCR3, enforce OC3REF as high level, otherwise its low level. Note 1: In case of LOCK level 3 (TIM1_BDTR register LOCK bit), the bit cannot be changed. Note 2: In PWM mode 1 or PWM mode 2, only when the compare result changes or it changes over from the freeze mode to PWM mode in the output compare mode, OC3REF level may change.
3	OC3PE	Channel 3 output compare preload enable 0: Disable TIM1_CCR3 register preload function, the value written into TIM1_CCR3 register becomes active immediately 1: Enable TIM1_CCR3 register preload function. The read and write only works on the preload register. TIM1_CCR3 preload value becomes active when the update event is available Note 1: When the LOCK level is set 3 (TIM1_BDTR register LOCK bit), the bit cannot be changed. Note 2: Only in the one-pulse mode (TIM1_CR1 register OPM= 1), it's not required to set the preload register. Under other scenarios, it's required to set the preload register. Otherwise, the follow up action is not certain.
2	OC3FE	Channel 3 output compare quick enable In case of the bit 1, when the channel is configured as PWM mode, it speeds up response of the compare output to the trigger time. The output channel deems the active edge of the trigger input signal as one compare match. Therefore, OC is set as compare level, but is irrelevant to the compare result. 0: channel 3 output compare fast disable. 1: channel 3 output compare fast enable.
1：0	Reserved	Reserved, must be kept at reset value.

9.4.9 TIM1_CCER Compare Enable Register

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		rw													

Bit	Field	Description
15：14	Reserved	Reserved, must be kept at reset value.
13	CC4P	Channel 4 compare output polarity Refer to CC1P description.
12	CC4E	Channel 4 compare enable Refer to CC1E description
11	CC3NP	Channel 3 complementary output polarity Refer to CC1NP description
10	CC3NE	Channel 3 complementary output enable Refer to CC1NE description
9	CC3P	Channel 3 output polarity Refer to CC1P description
8	CC3E	Channel 3 output enable Refer to CC1E description
7	CC2NP	Channel 2 complementary output polarity Refer to CC1NP description
6	CC2NE	Channel 2 complementary output enable Refer to CC1NE description
5	CC2P	Channel 2 output polarity Refer to CC1P description
4	CC2E	Channel 2 output enable Refer to CC1E description
3	CC1NP	Channel 1 complementary output polarity This bit has defined the input signal polarity: 0: OC1N is active at high level 1: OC1N is active at low level Note: When LOCK level (TIM1_BDTR register LCCK bit) is set 3 or 2, the bit cannot be changed.
2	CC1NE	Channel 1 complementary output enable 0: Close channel 1 complementary output. OC1N output disable. 1: OC1N signal exports to the corresponding output pin. The output level relies on the values in MOE, OSSI, OSSR, OIS1, OIS1N and CC1E.
1	CC1P	Channel 1 output polarity This bit defines the output signal polarity: 0: OC1 is active at high level 1: OC1 is active at low level Note: When LOCK level (TIM1_BDTR register LCCK) is set 3 or 2, this bit cannot be changed.
0	CC1E	Channel 1 output enable 0: Close. OC1 output disable 1: Enable. The output level relies on the values in MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE.

Table 9-3 Output control bits for complementary OCx and OCxN channels with break feature

Control bits					Output states	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx output states	OCxN output states
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)	OCxREF + Polarity，

					OCx = 0，OCx_EN = 0	OCxN = OCxREF xor CCxNP， OCxN_EN = 1
		0	1	0	OCxREF + Polarity， OCx = OCxREF xor CCxP， OCx_EN = 1	Output Disabled (not driven by the timer) OCxN = 0，OCxN_EN = 0
		0	1	1	OCxREF + Polarity + deadtime, OCx_EN=1	Complementary to OCxREF + 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	Output Disabled (not driven by the timer) OCxN = CCxNP，OCxN_EN = 1
		1	1	1	OCxREF + Polarity + deadtime, OCx_EN = 1	Complementary to OCxREF + Polarity + dead-time, OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer) Asynchronously：OCx = CCxP ，OCx_EN = 0 ，OCxN = CCxNP，OCxN_EN = 0； If the clock is present: after a dead-time OCx = OISx ，OCxN = OISxN， Assuming that OISx and OISxN do not correspond to OCX and OCxN both in active.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	Output Disabled (not driven by the timer) Asynchronously：OCx = CCxP，OCx_EN = 1，OCxN = CCxNP，OCxN_EN = 1； If the clock is present: after a dead-time OCx = OISx ，OCxN = OISxN， Assuming that OISx and OISxN do not correspond to OCX and OCxN both in active.	
	1		0	1		
	1		1	0		
	1		1	1		

Note1: When both outputs of a channel are not used (CCxE = CCxNE = 0), the OISx, OISxN, CCxP and CCxNP bits must be kept cleared.

Note2: The state of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel state and the GPIO registers.

9.4.10 TIM1_CNT Counter

Address offset: 0x24

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
rw															

Bit	Field	Description
15：0	CNT	Counter value

9.4.11 TIM1_PSC Prescaler

Address offset: 0x28

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
rw															

Bit	Field	Description

15：0	PSC	Prescaler value Counter clock frequency (ck_cnt) =$f_{CK\_PSC/（PSC+1）}$ In case of update event, PSC value is loaded into the current prescaler register.

9.4.12 TIM1_ARR Auto Reload Register

Address offset: 0x2C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR															
rw															

Bit	Field	Description
15：0	ARR	Auto reload value These bits define the auto reload value of the counter. When auto reload value is 0, the counter doesn't work.

9.4.13 TIM1_RCR Repeat Count Register

Address offset: 0x30

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REP_CNT								REP							
rw								rw							

Bit	Field	Description
15：8	REP_CNT	Real-time value written by the repeat counter In the repeat count mode, it's to change real-time phase move of the detection point of update interrupt flag bit (UIF) Note: After the update event, write these bits. Write REP_CNT before the update event will make the bit move inactive.
7：0	REP	Repeat counter value The repeat counter value defines the generation speed of the update event. When the repeat counter value reduces to 0, it generate the update event. If the update interrupt is enabled, it will simultaneously affect the update interrupt generation speed. The written REP value becomes active at the next update event. In PWM mode, (REP+1) corresponds to: In the edge alignment mode, the number of PWM period In the central alignment mode, the number of PWM half period

9.4.14 TIM1_CCR1 Compare Register 1

Address offset: 0x34

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw															

Bit	Field	Description
15：0	CCR1	Channel 1 compare value If the preload function is not selected in TIM1_CCMR1 register (OC1PE bit), the write value is immediately transmitted to the current compare shadow register. Or else, the write value is only loaded into the compare register in case of update event. The current compare shadow register is involved in the compare with the counter TIM1_CNT, and the compare result is reflected to the output signal of OC1 port.

9.4.15 TIM1_CCR2 Compare Register 2

Address offset: 0x38
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2															
rw															

Bit	Field	Description
15：0	CCR2	Channel 2 compare value Refer to CCR1 description.

9.4.16 TIM1_CCR3 Compare Register 3

Address offset: 0x3C
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3															
rw															

Bit	Field	Description
15：0	CCR3	Channel 3 compare value Refer to CCR1 description.

9.4.17 TIM1_CCR4 Compare Register 4

Address offset: 0x40
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
rw															

Bit	Field	Description
15：0	CCR4	Channel 4 compare value Refer to CCR1 description.

9.4.18 TIM1_BDTR Break and Dead-Time Register

Address offset: 0x44
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															DOE
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
rw	rw	rw	rw	rw	rw	rw		rw							

Note: According to the lock setting, DOE, AOE, BKP, BKE, OSSI, OSSR and DTG bits can be write protected; it's necessary to configure them during the first write into TIM1_BDTR register. Details are given in the chapter of complementary output and dead-time insert.

Bit	Field	Description
31：17	Reserved	Reserved, must be kept at reset value.
16	DOE	Direct output enable When the break is active, MOE is set 0, active. 0：After break output, wait for a dead time and export the idle state (output disable). 1：Immediately export the idle state (output disable). Note: When LOCK level 1 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed.

15	MOE	Main output enable When the channel x is configured as output, according to AOE bit setting value, the bit can be cleared or auto set by software. When the break input is active, the bit is cleared by hardware. 0：Disable OCx and OCxN output or enforce as the idle state (output disable). 1：When setting the corresponding enable bit (TIM1_CCER register CCxE, CCxNE), OCx and OCxN output enable
14	AOE	Auto output enable 0: MOE cannot be set by hardware 1: MOE can be set by software or auto set by hardware at the next update event when the break is inactive Note: When LOCK level 1 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed.
13	BKP	Break input polarity 0: Break input is active low 1: Break input is active high Note: When LOCK level 1 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed.
12	BKE	Break function enable 0：Break input disabled 1：Break input enabled Note 1: When LOCK level 1 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed. Note 2: Break input includes pin input and comparator compare result input. Before break function is enabled, configure BKIN_SEL bit in TIM1_BKINF register to select break source.
11	OSSR	Off state selection in the run mode This bit only applies in case of MOE =1 and the channel is in complementary output. 0: When the timer inactive, disable OC/OCN output 1: When the timer inactive, in case of CCxE = 1 or CCxNE = 1, first enable OC/OCN and export inactive level, and then set OC/OCN enable output signal Note: When LOCK level 2 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed.
10	OSSI	Off state selection in the idle mode This bit only applies in case of MOE =0 and the channel is in output. 0: When the timer inactive, disable OC/OCN output 1: When the timer inactive, in case of CCxE = 1 or CCxNE = 1, first OC/OCN exports inactive level, and then set OC/OCN enable output signal. Note: When LOCK level 2 (TIM1_BDTR register LOCK bit) is set, the bit cannot be changed.
9：8	LOCK	Lock configuration This bit defines the register write protection function. 00: Write protection function closes, the register doesn't have the write protection 01: Lock level 1, unable to write TIM1_BDTR register DOE, DTG, BKE, BKP and AOE bit and TIM1_CR2 register OISx/OISxN bit 10: Lock level 2, unable to write lock level 1 bit, or CC polarity level as well as OSSR/OSSI bit 11：Lock level 3, unable to write lock level 2 bit, or CC control bit 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	Dead-time generator setup adjustment These bits define the dead-time duration of the complementary output. DTG[7: 5] = 0xx: DT = (DTG [7: 0] + 1) × tdtg, tdtg = tDTS; DTG[7: 5] = 10x: DT = (DTG [5: 0] + 1 + 64) × tdtg, tdtg = 2 × tDTS; DTG[7: 5] = 110: DT = (DTG [4: 0] + 1 + 32) × tdtg, tdtg = 8 × tDTS; DTG[7: 5] = 111: DT = (DTG [4: 0] + 1 + 32) × tdtg, tdtg = 16 × tDTS; Example if tDTS = 125ns(8MHz), dead-time possible values are:

		125ns to 15875ns(step time is 125ns) 16µs to 31750ns(step time is 250ns) 32µs to 63µs(step time is 1µs) 64µs to 126µs(step time is 2µs) Note: When LOCK level (TIM1_BDTR register LOCK bit) is set 1, 2 or 3, the bit cannot be changed.

9.4.19 TIM1_CCMR3 Compare Mode Register 3

Address offset: 0x54
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												OC5PE	Reserved		
												rw			

Bit	Field	Description
15：4	Reserved	Reserved, must be kept at reset value.
3	OC5PE	Output compare 5 preload enable 0: Disable TIM1_CCR5 register preload function, and the value written into TIM1_CCR5 register will become effective immediately. 1: Enable TIM1_CCR5 register preload function. Only read and write the preload register. TIM1_CCR5 preload value becomes active at the update event Note 1: When LOCK level 3 is set (TIM1_BDTR register LOCK level), this bit cannot be changed.
2：0	Reserved	Reserved, must be kept at reset value.

9.4.20 TIM1_CCR5 Compare Register 5

Address offset: 0x58
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR5															
rw															

Bit	Field	Description
15：0	CCR5	Compare 5 value CC5 channel is only configured as output: If the preload function is not selected in the TIM1_CCMR3 register (OC5PE bit), the write value will immediately be transmitted to the corresponding compare shadow register. Or else, only in case of the update event, the preload value will be transmitted to the corresponding compare shadow register. The compare shadow register is involved in the counter TIM1_CNT compare. Because CC5 channel is the internal channel and cannot be transmitted to the pin, the compare result is used for the internal trigger event.

9.4.21 TIM1_PDER PWM Phase Shift Enable Register

Address offset: 0x5C
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										CCR5_SHIFT_EN	CCR4_SHIFT_EN	CCR3_SHIFT_EN	CCR2_SHIFT_EN	CCR1_SHIFT_EN	Res.
										rw	rw	rw	rw	rw	

Bit	Field	Description

15：6	Reserved	Reserved, must be kept at reset value.
5	CCR5_SHIFT_EN	Enable channel 5 output PWM phase shift bit 0: Channel 5 output PWM phase shift disable 1: Channel 5 output PWM phase shift enable Details given in CCRxFALL register phase shift operation
4	CCR4_SHIFT_EN	Enable channel 4 output PWM phase shift bit 0: Channel 4 output PWM phase shift disable 1: Channel 4 output PWM phase shift enable Details given in CCRxFALL register phase shift operation
3	CCR3_SHIFT_EN	Enable channel 3 output PWM phase shift bit 0: Channel 3 output PWM phase shift disable 1: Channel 3 output PWM phase shift enable Details given in CCRxFALL register phase shift operation
2	CCR2_SHIFT_EN	Enable channel 2 output PWM phase shift bit 0: Channel 2 output PWM phase shift disable 1: Channel 2 output PWM phase shift enable Details given in CCRxFALL register phase shift operation
1	CCR1_SHIFT_EN	Enable channel 1 output PWM phase shift bit 0: Channel 1 output PWM phase shift disable 1: Channel 1 output PWM phase shift enable Details given in CCRxFALL register phase shift operation
0	Reserved	Reserved, must be kept at reset value.

9.4.22 TIM1_CCRxFALL PWM Phase Shift Count Down Compare Register

Address offset: 0x60 ~ 0x70

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCRxFALL															
rw															

Bit	Field	Description
15：0	CCRxFALL	Channel x compare value during count down in PWM central alignment mode PWM phase shift function: Enable PDER register PWM phase shift. According to the required phase shift, configure CCRxFALL and CCRx, PWM exports programmable phase shift waveform, and can shift left or right.

10. TIM3 16-Bit General-purpose Timer

10.1 Overview

The general-purpose timers consist of a 16-bit auto-reload counter driven by a programmable prescaler. They may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

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

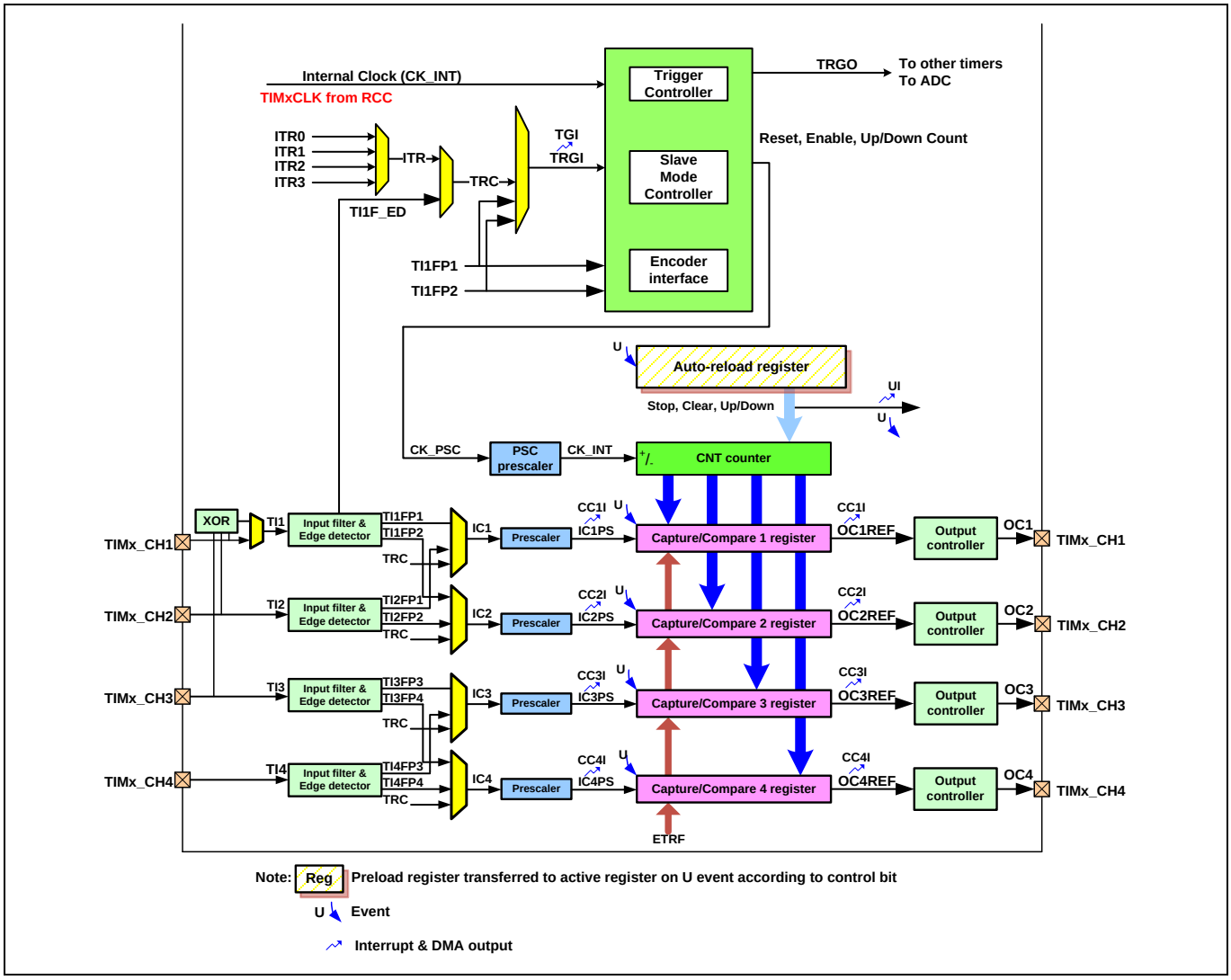
The TIMx timers are completely independent, and do not share any resources. They can be synchronized together.

10.2 Main characteristics

General-purpose TIMx timer features include:

1. 16-bit up, down, up/down auto-reload counter.
2. 16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 ~ 65536.
3. Up to 4 independent channels for:
   - Input capture
   - Output compare
   - PWM generation (Edge and Center-aligned Mode)
   - One-pulse mode output
4. Synchronization circuit to control the timer with external signals and to interconnect several timers together
5. Supports incremental (quadrature) encoder and hall sensor circuitry for positioning purposes
6. Trigger input for external clock or cycle-by-cycle current management

Figure 10-1 General-purpose timer block diagram



10.3 Function description

10.3.1 Time-base unit

The main block of the programmable general-purpose timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock can be divided by a prescaler. The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running. The time-base unit includes:

Counter register (TIMx_CNT)

Prescaler register (TIMx_PSC)

Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the autoreload 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 generation of the update event is described in detailed for each configuration.

The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIMx_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

10.3.1.1 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.

The figures below give some examples of changing the counter parameters while the prescaler is running:

Figure 10-2 Counter timing diagram with prescaler division change from 1 to 2

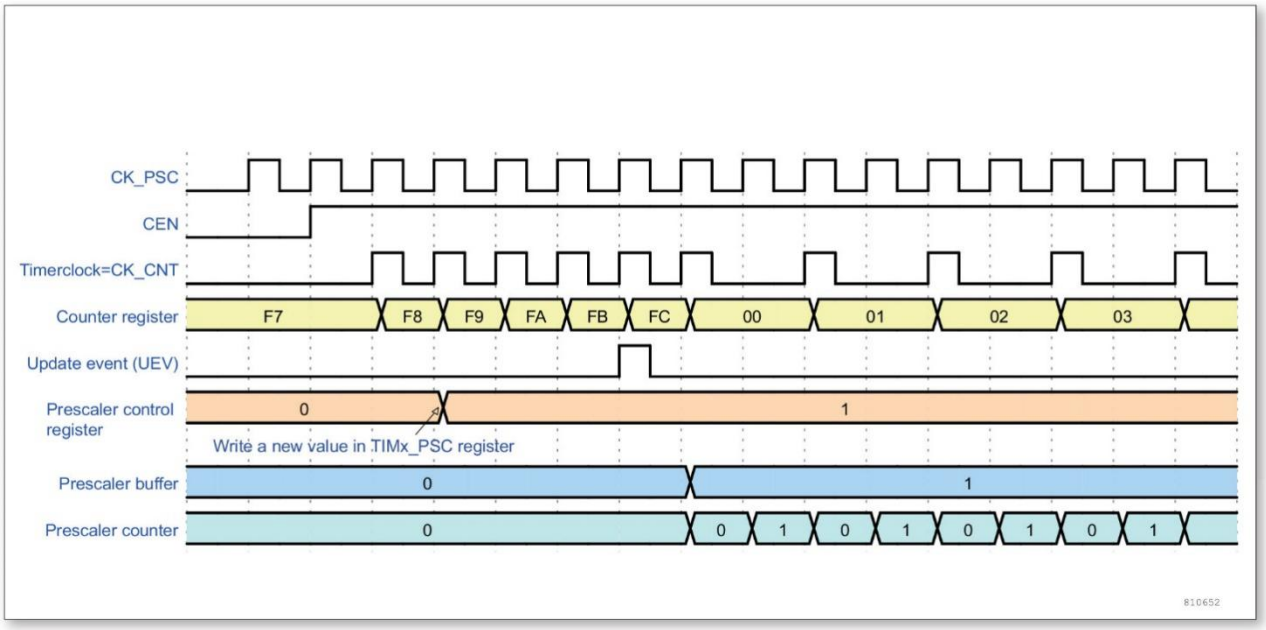
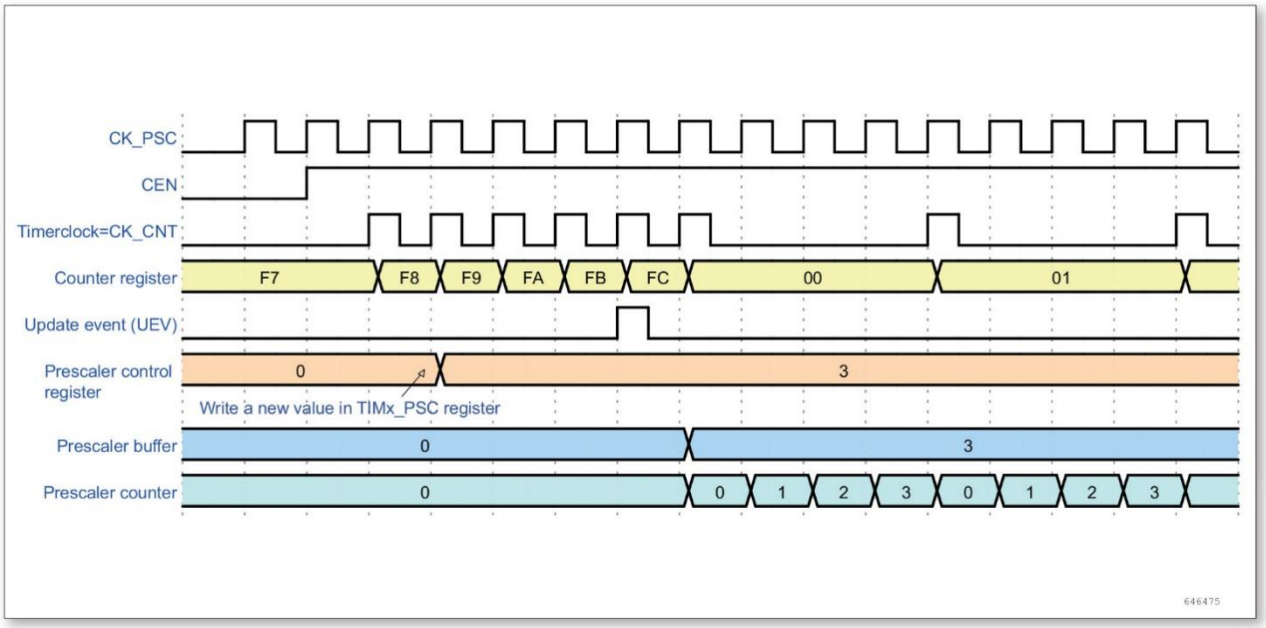


Figure 10-3 Counter timing diagram with prescaler division change from 1 to 4



10.3.2 Counter modes

10.3.2.1 Upcounting mode

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

An Update event can be generated at each counter overflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).

The UEV event can be disabled by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers.

Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. 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 by hardware (thus no interrupt is sent).

This is to avoid generating both update and capture interrupts when clearing the counter on the capture mode.

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

The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The auto-reload shadow register is updated with the preload value (TIMx_ARR).

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

Figure 10-4 Counter timing diagram, internal clock divided by 1

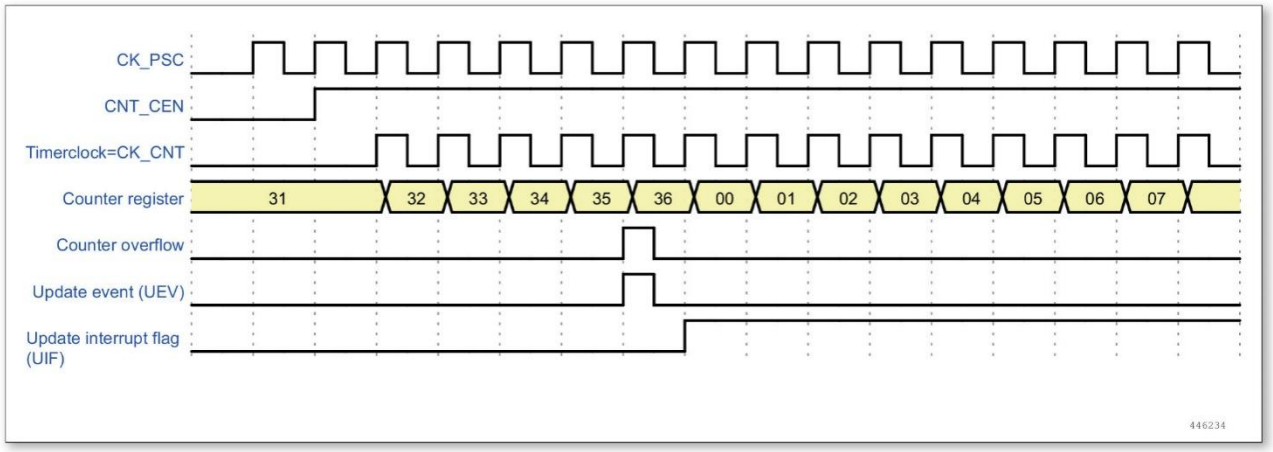


Figure 10-5 Counter timing diagram, internal clock divided by 2

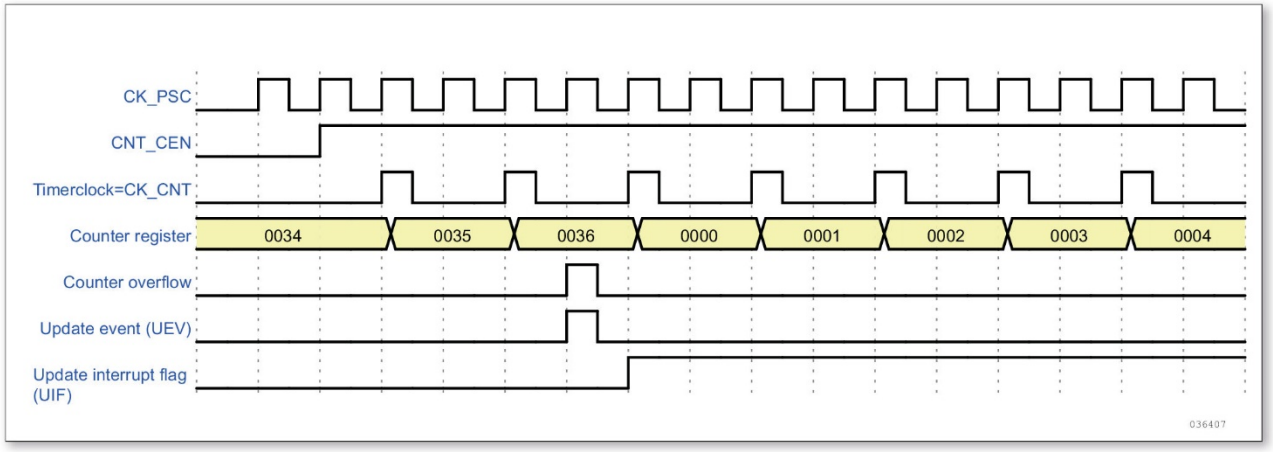


Figure 10-6 Counter timing diagram, internal clock divided by 4

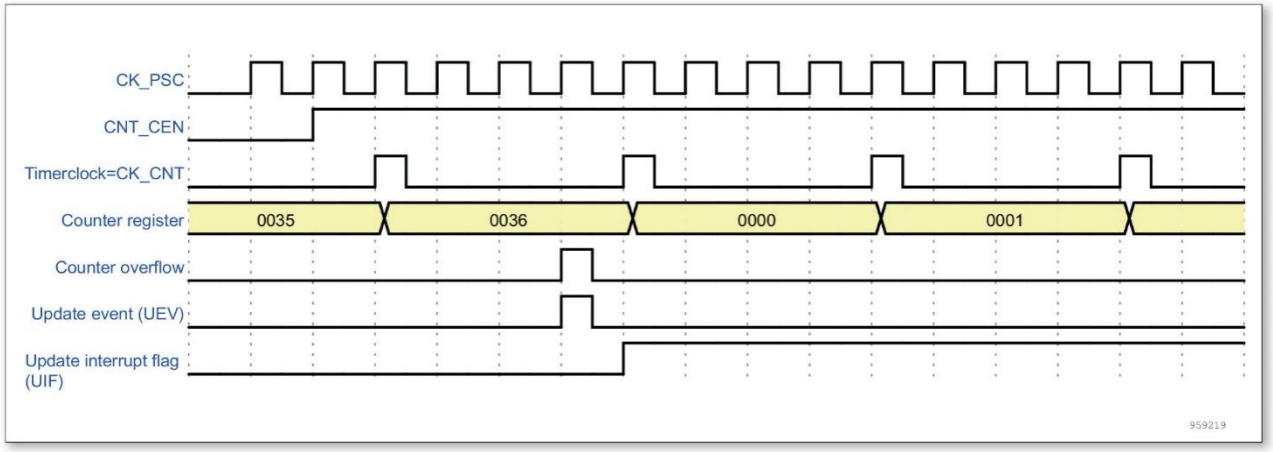


Figure 10-7 Counter timing diagram, internal clock divided by N

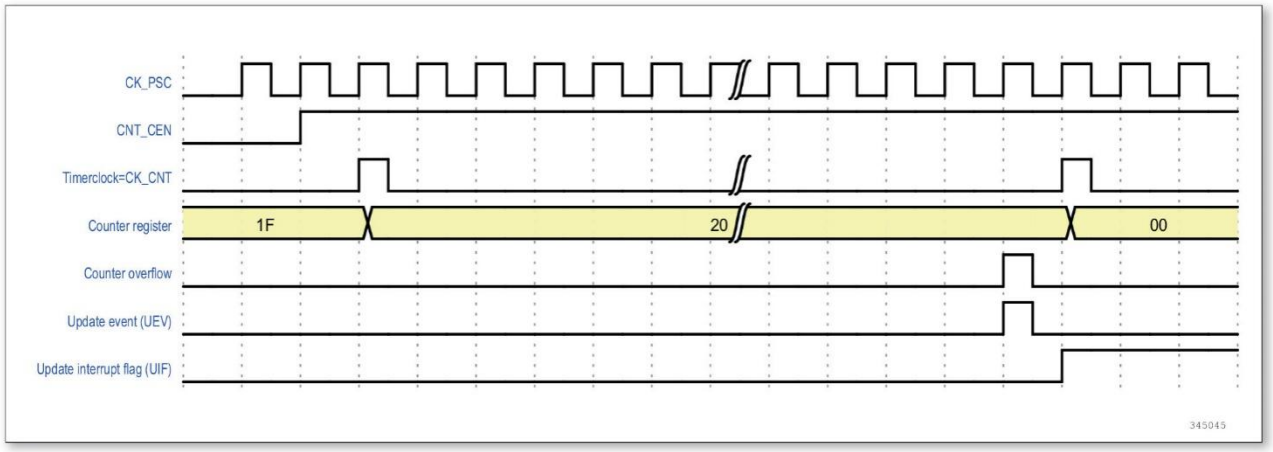


Figure 10-8 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

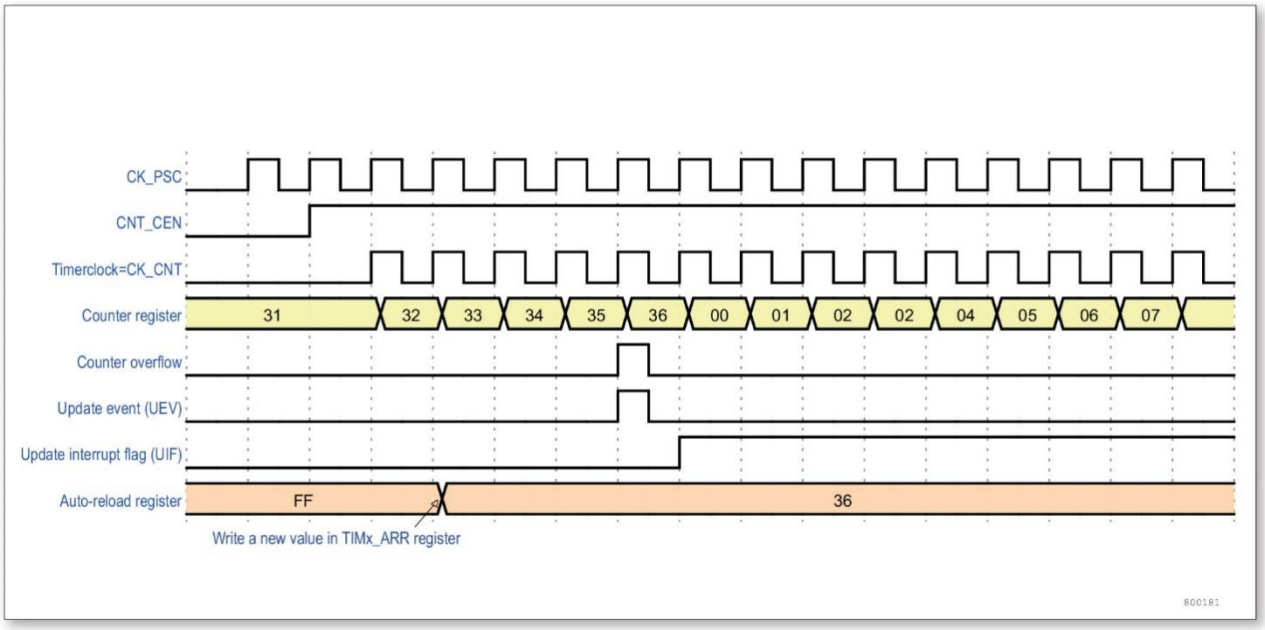
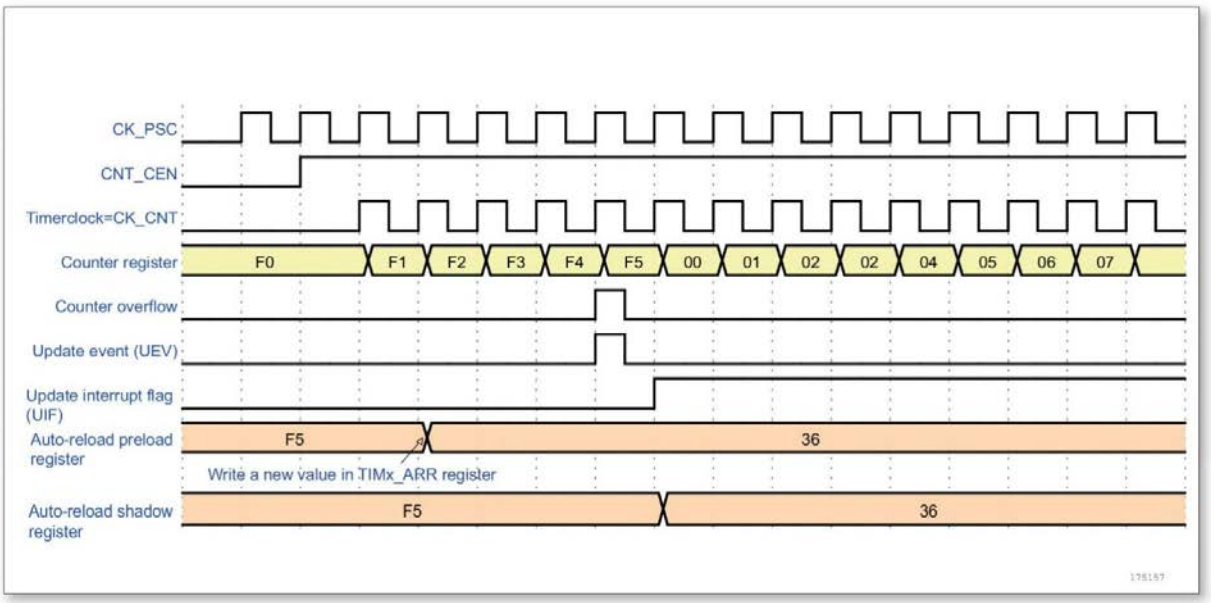


Figure 10-9 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)



10.3.2.2 Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMx_ARR register) down to 0, then restarts from the auto-reload value and generates a counter underflow event.
An Update event can be generated at each counter overflow or by setting the UG bit in the TIMx_EGR register (by software or by using the slave mode controller).
The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers.
Then no update event occurs until UDIS bit has been written to 0. However, the counter restarts from the current auto-reload value, whereas the counter of the prescaler restarts from 0 (but the prescale rate doesn't change).
In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.
When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register).

The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note: The auto-reload is updated before the counter is reloaded, so that the next period is the expected one.
The following figures show some examples of the counter behavior for different clock frequencies when TIMx_ARR = 0x36:

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

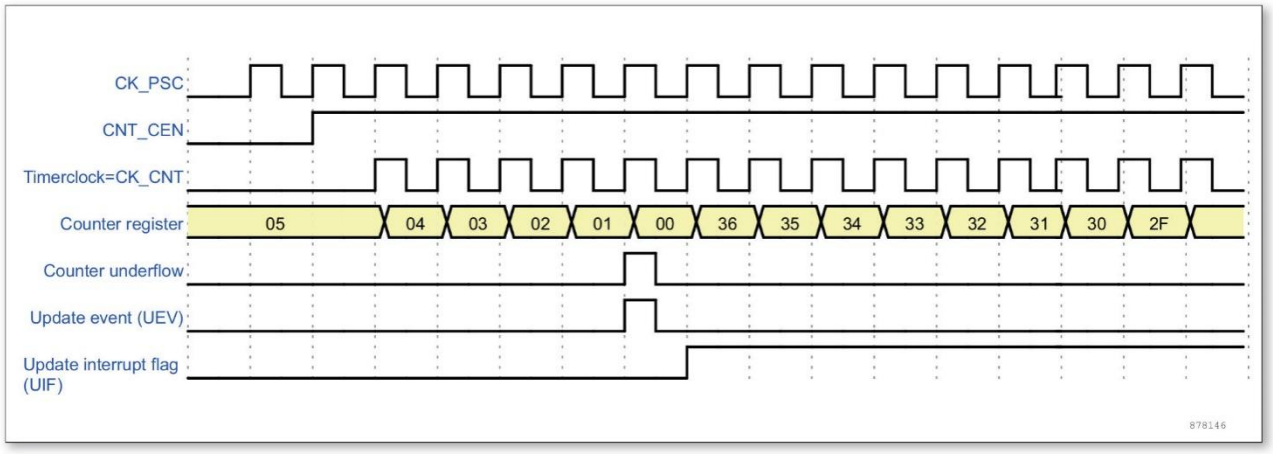


Figure 10-11 Counter timing diagram, internal clock divided by 2

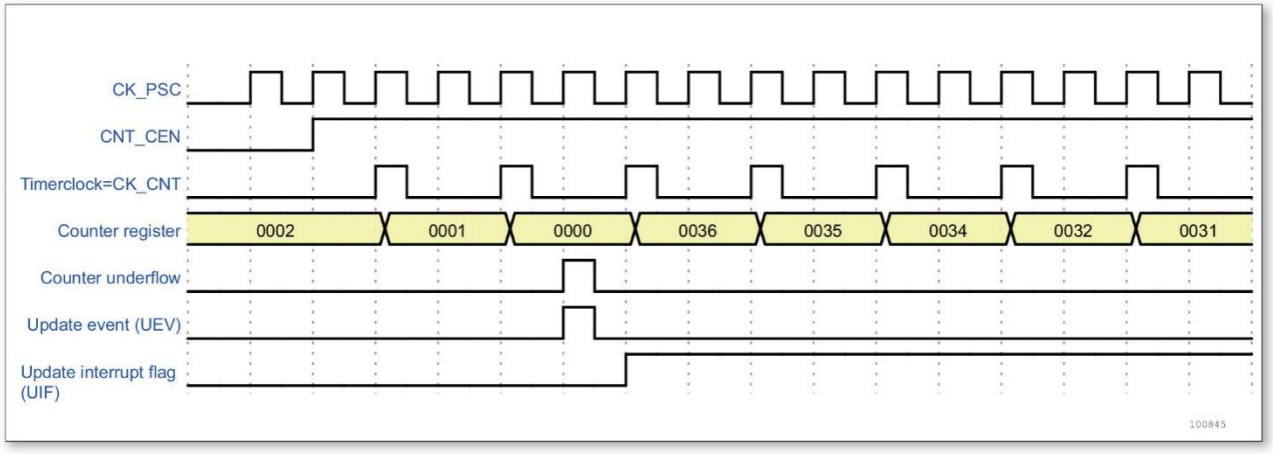


Figure 10-12 Counter timing diagram, internal clock divided by 4

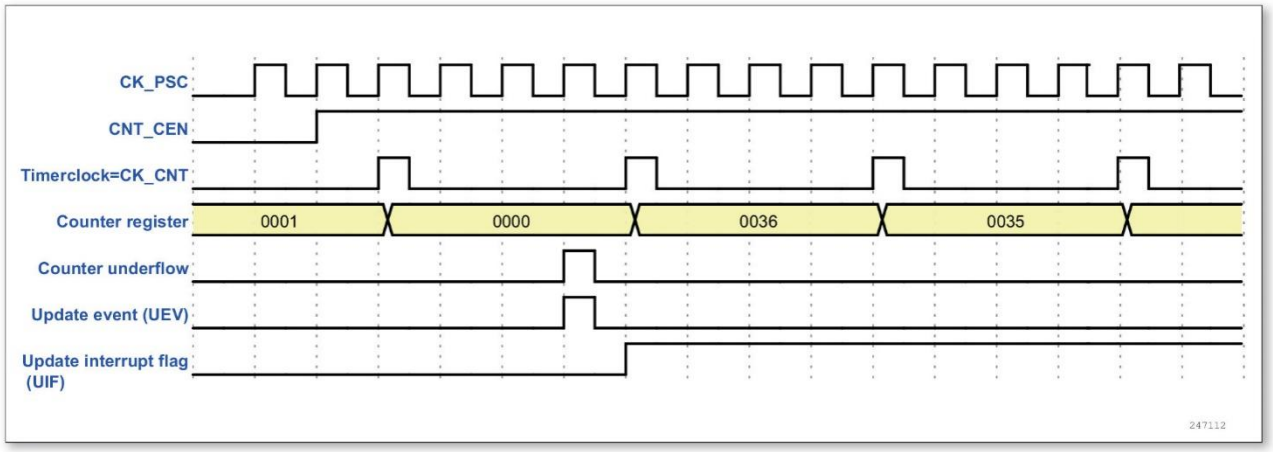


Figure 10-13 Counter timing diagram, internal clock divided by N

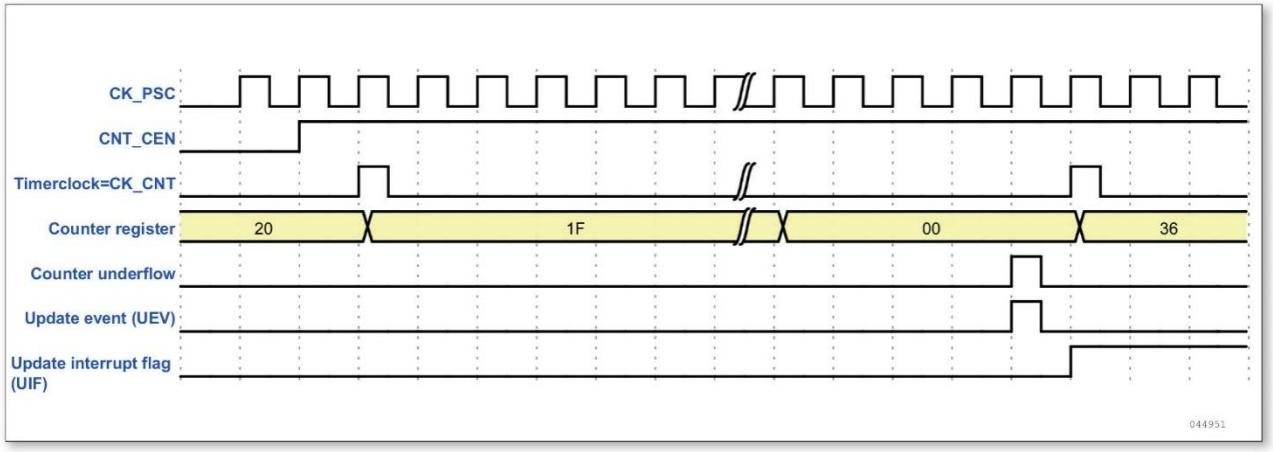
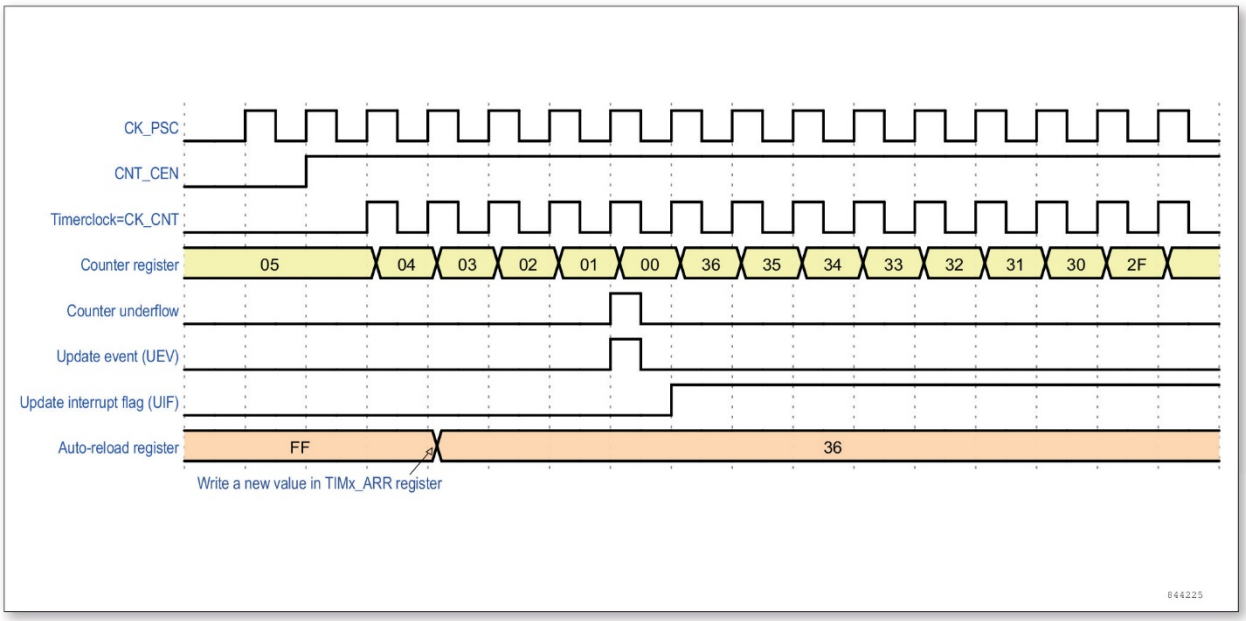


Figure 10-14 Counter timing diagram, update event when repetition counter is not used



10.3.2.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (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.
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). In this case, the counter restarts counting from 0, as well as the counter of the prescaler.
The UEV update event can be disabled by setting the UDIS bit in TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers.
Then no update event occurs until UDIS bit has been written to 0. However, the counter continues counting up and down, based on the current auto-reload value.
In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.
When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

The buffer of the prescaler is reloaded with the preload value (content of the TIMx_PSC register.

The auto-reload active register is updated with the preload value (content of the TIMx_ARR register).

Note: If the update source is a counter overflow, the 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 10-15 Counter timing diagram, internal clock divided by 1, TIMx_ARR = 0x06

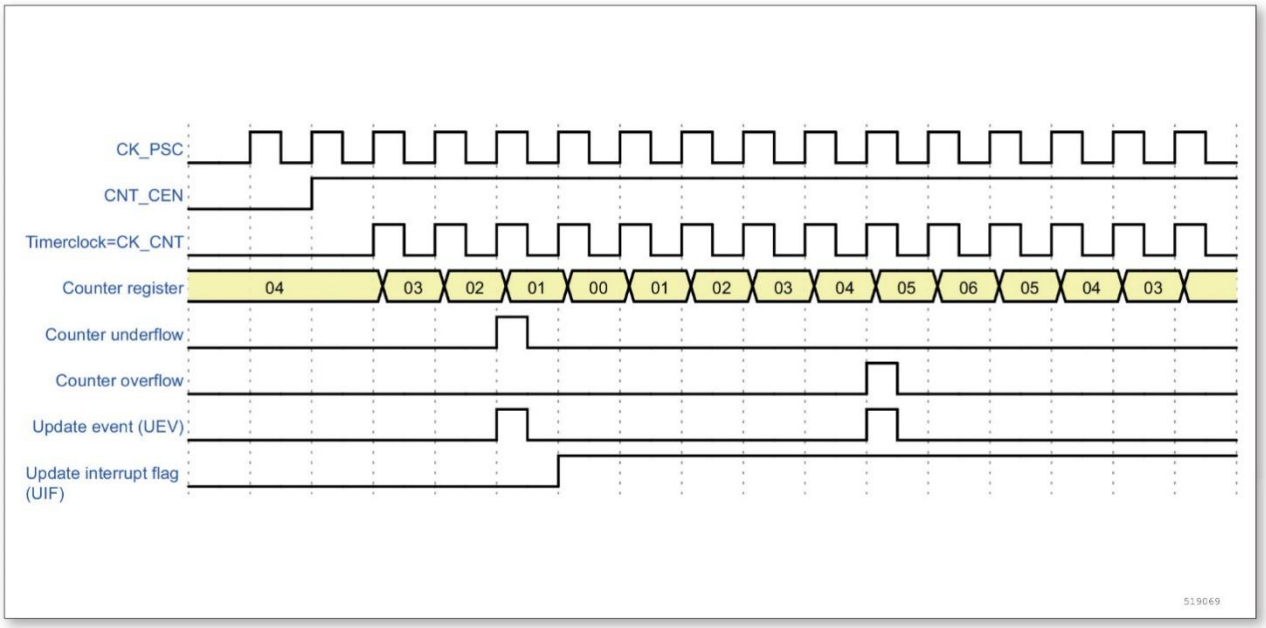


Figure 10-16 Counter timing diagram, internal clock divided by 2

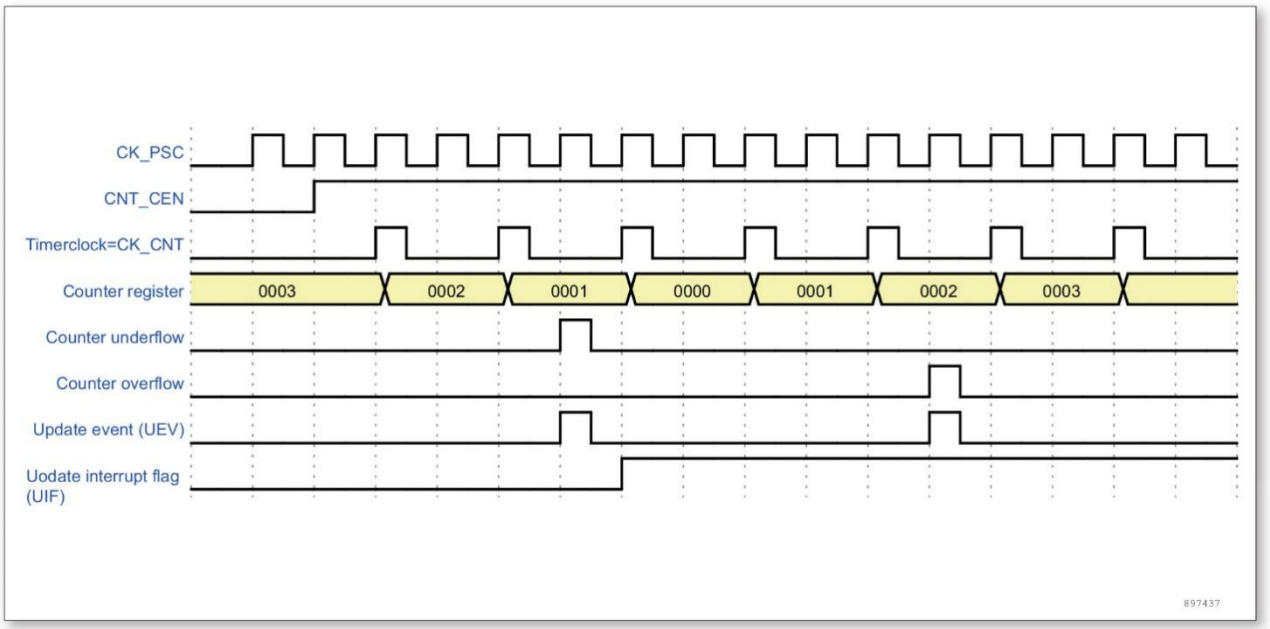


Figure 10-17 Counter timing diagram, internal clock divided by 4, TIMx_ARR = 0x03

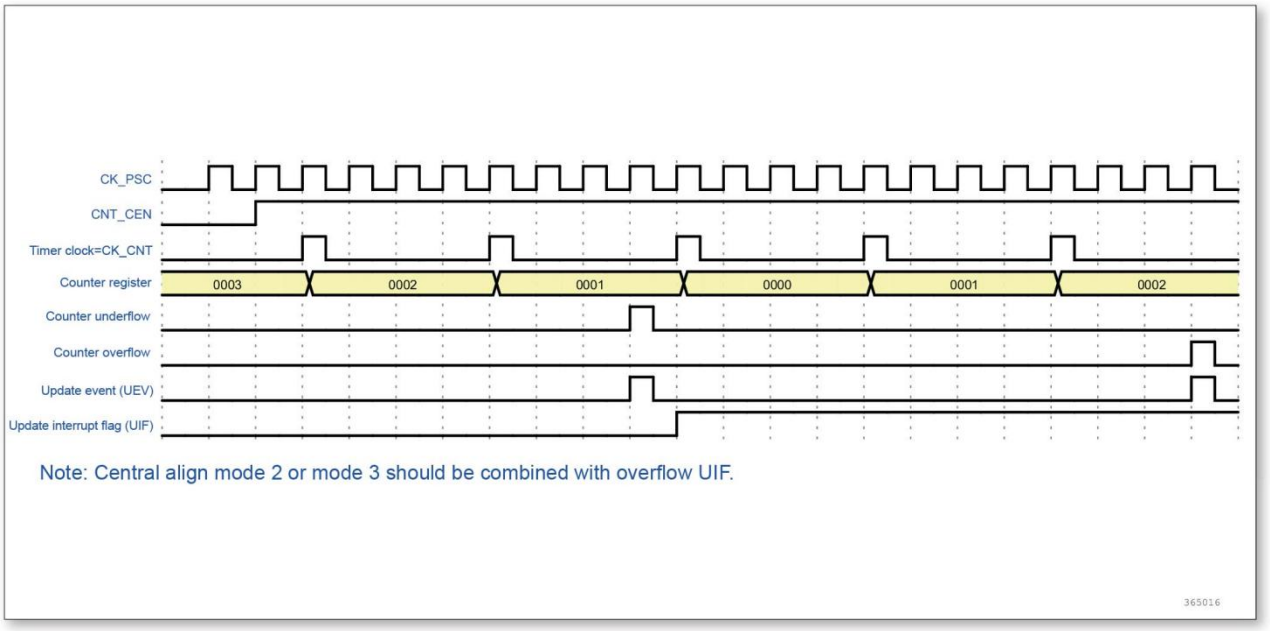


Figure 10-18 Counter timing diagram, internal clock divided by N

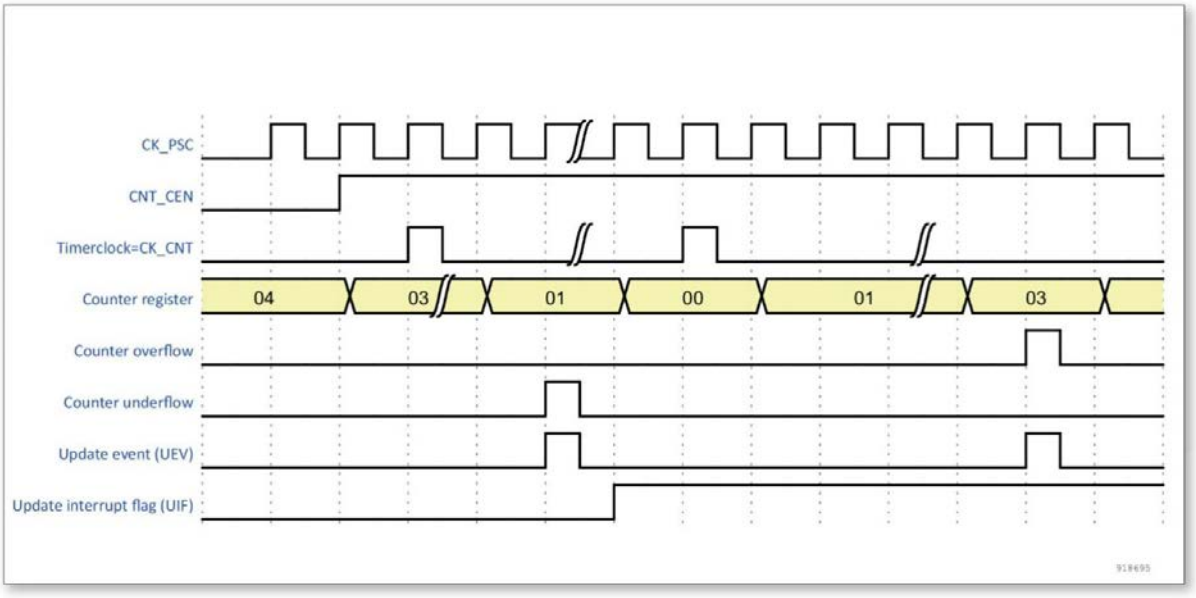


Figure 10-19 Counter timing diagram, update event with ARPE=1 (counter underflow)

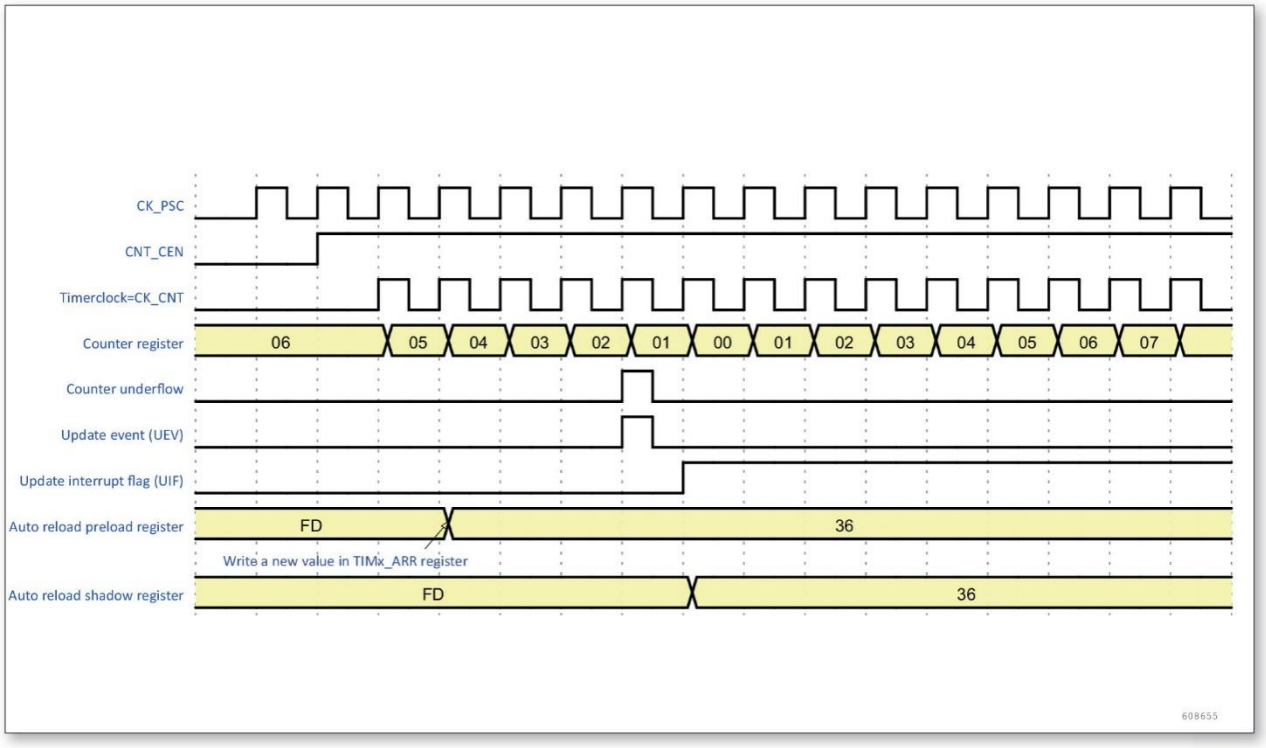
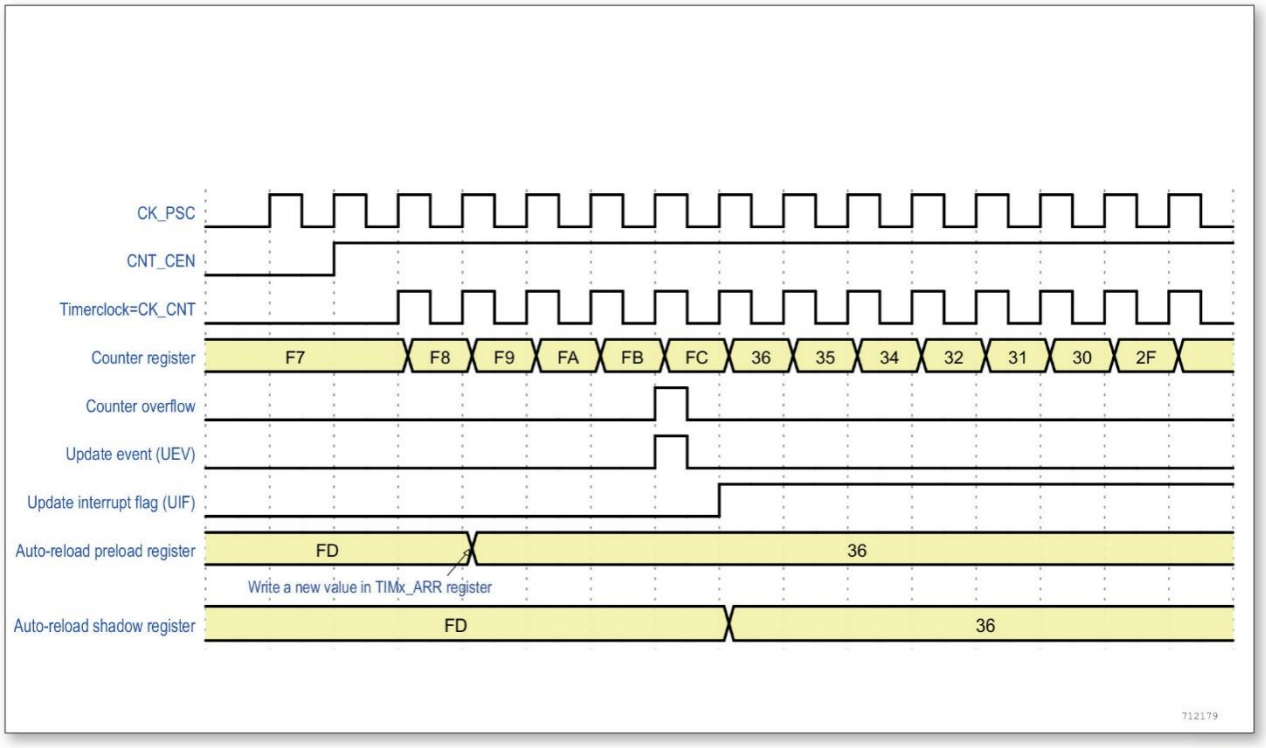


Figure 10-20 Counter timing diagram, update event with ARPE=1 (counter overflow)



10.3.3 Clock selection

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

Internal clock (CK_INT)

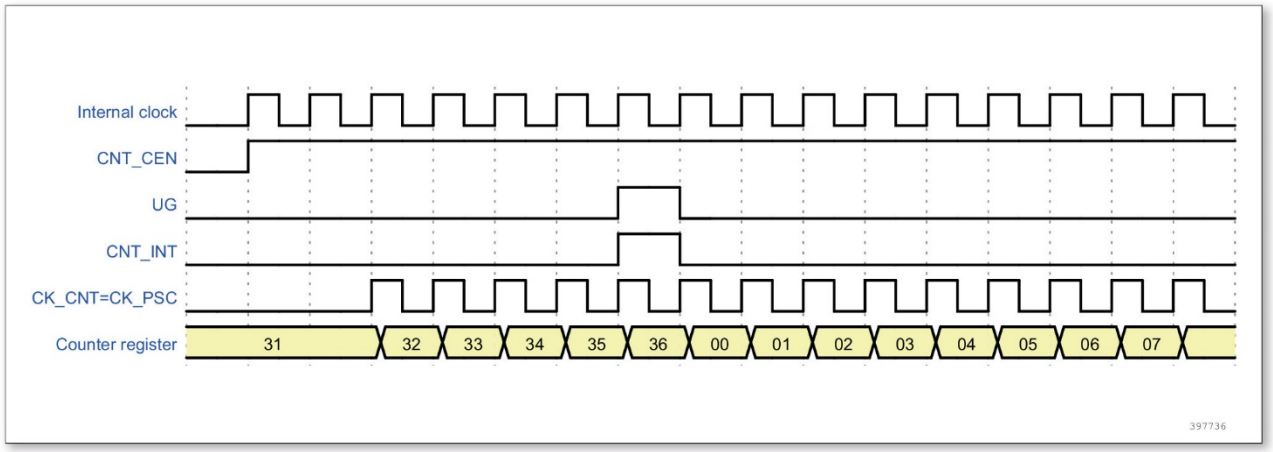
External clock mode 1: external input pin (TIx)

Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 3

10.3.3.1.1 Internal clock source (INT _CK)

If the slave mode controller is disabled (SMS = 000), then the CEN, DIR (in the TIMx_CR1 register) and UG bits (in the TIMx_ EGR register) are actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.
The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

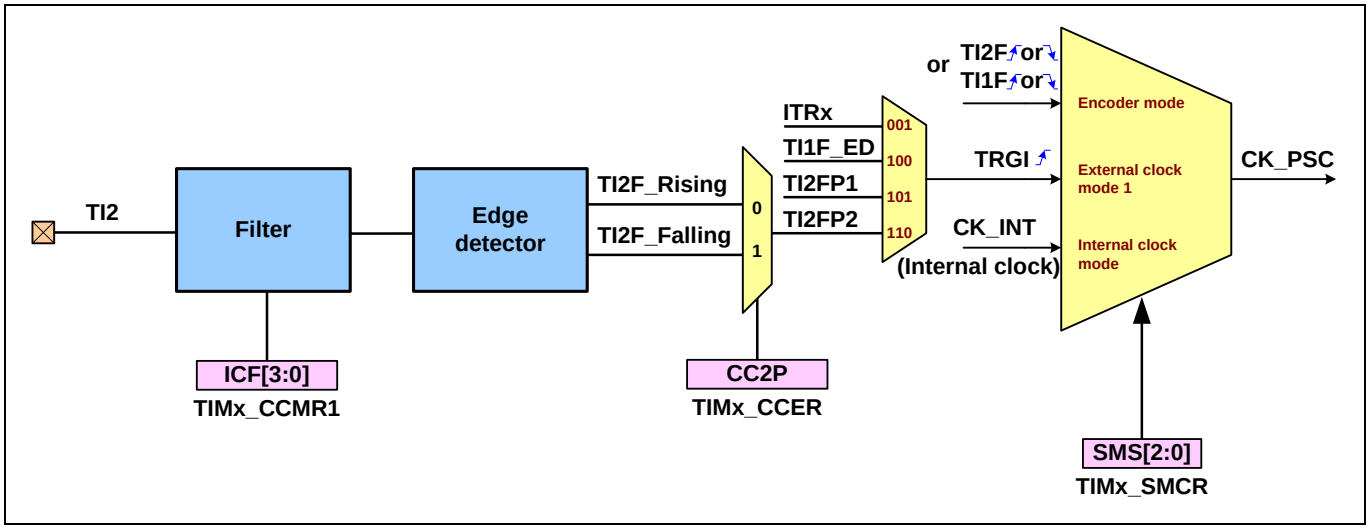
Figure 10-21 Control circuit in normal mode, internal clock divided by 1



10.3.3.1.2 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 10-22 TI2 external clock connection example



For example, to configure the upcounter to count in response to a rising edge on the TI2 input, use the following procedure:

Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = 01 in the TIMx_CCMR1 register.

Configure the input filter duration by writing the IC2F [3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F = 0000). Note: The capture prescaler is not used for triggering, so the user does not need to configure it.

Select rising edge polarity by writing CC2P = 0 in the TIMx_CCER register.

Configure the timer in external clock mode 1 by writing SMS = 111 in the TIMx_SMCR register.

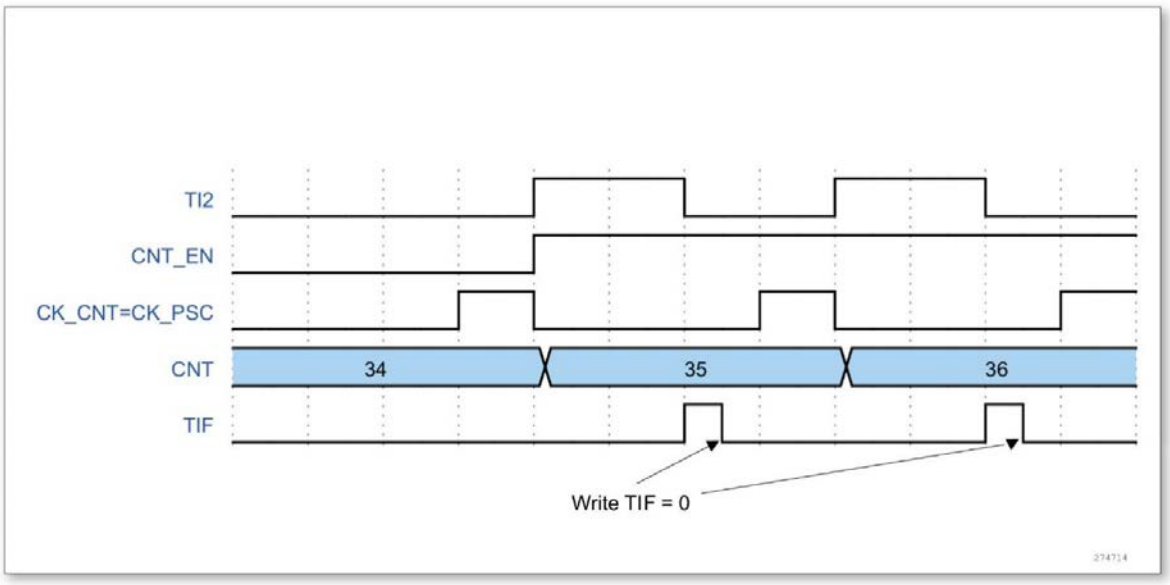
Select TI2 as the trigger input source by writing TS = 110 in the TIMx_SMCR register.

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

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

The delay between the rising edge on TI2 and the actual clock of the counter is due to the resynchronization circuit on TI2 input.

Figure 10-23 Control circuit in external clock mode 1

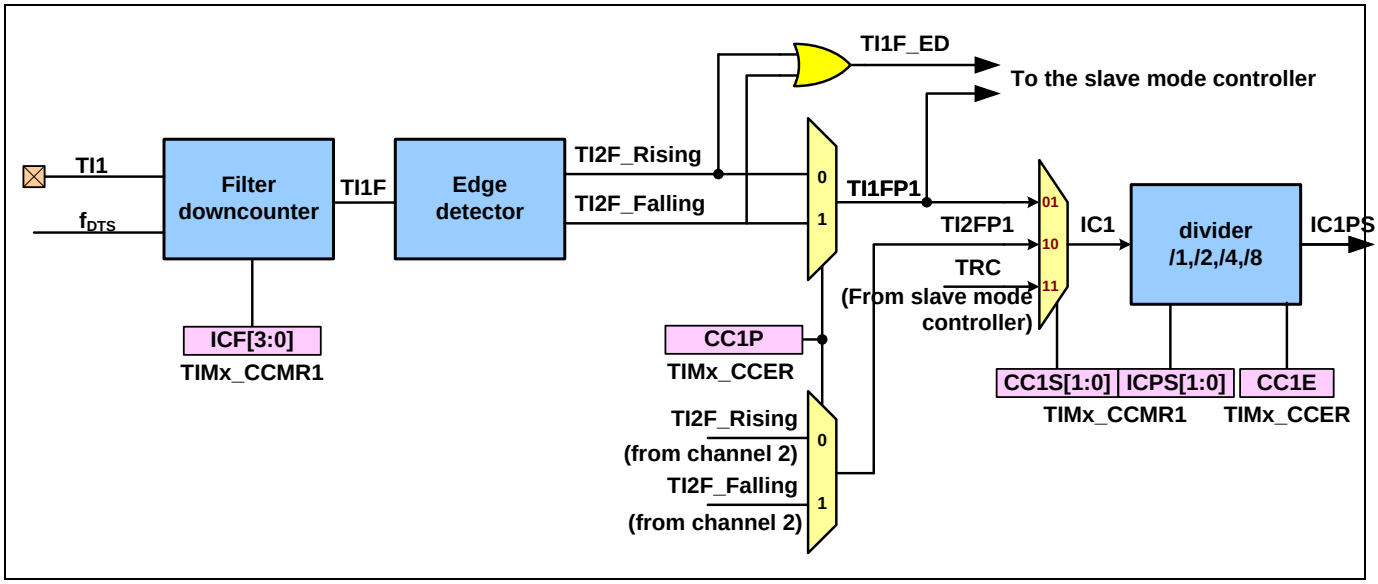


10.3.4 Capture/compare channel

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

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

Figure 10-24 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 10-25 Capture/compare channel 1 main circuit

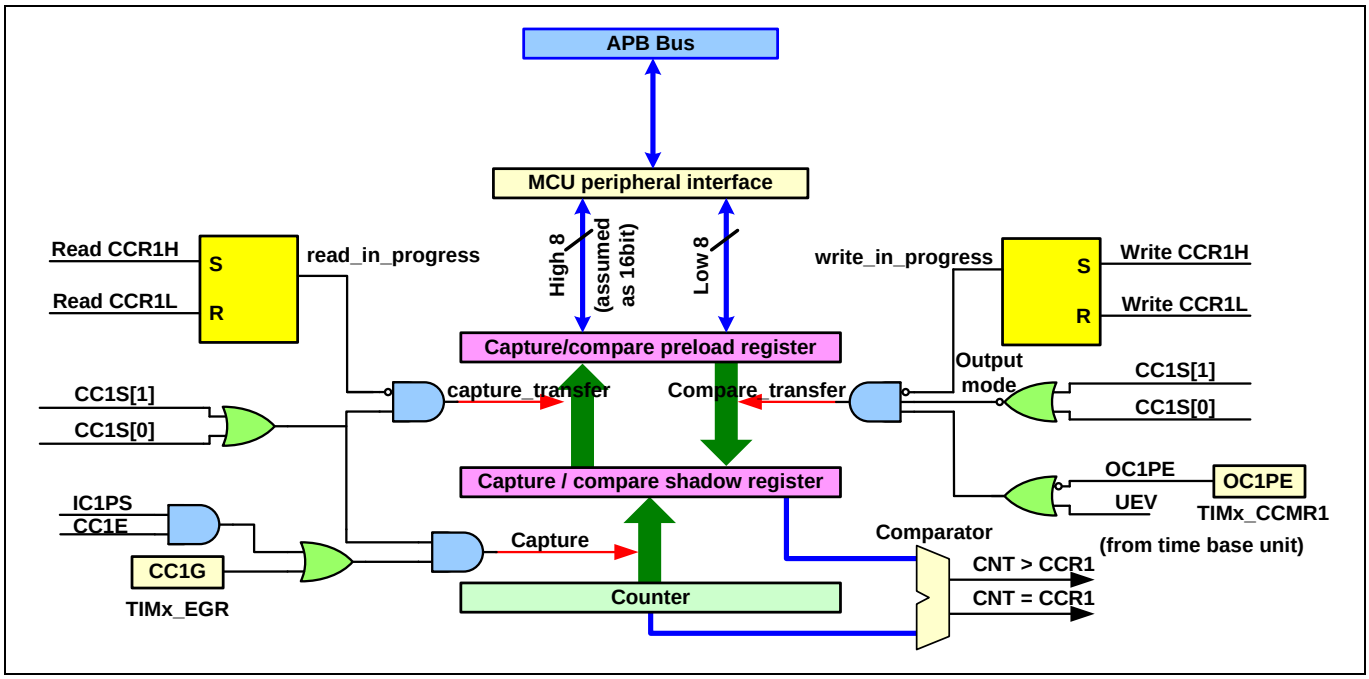
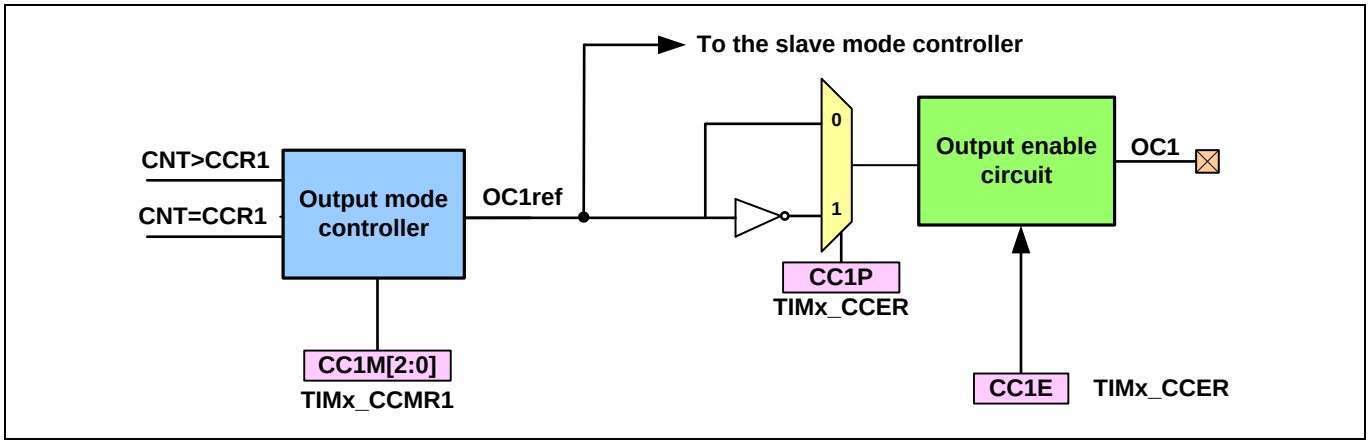


Figure 10-26 Output stage of capture/compare channel (channel 1)



The capture/compare block is made of one preload register and one shadow register. Write and read always access the preload register. In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

10.3.5 Input capture mode

In Input capture mode, the Capture/Compare registers (TIMx_CCRx) are used to latch the value of the counter after an edge is 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 it is enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to '0'.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises. To do this, use the following procedure:

Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write CC1S = 01 in the TIMx_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TM1_CCR1 register becomes read-only.

Program the needed input filter duration with respect to the input signal (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input

signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate an edge transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F = 0011 in the TIMx_CCMR1 register.

Select the edge of the active transition on the TI1 channel by writing CC1P = 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 = 00 in the TIMx_CCMR1 register).

Enable capture from the counter into the capture register by setting the CC1E = 1 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 level transition.

CC1IF flag is set (interrupt flag). CC1IF flag was not cleared if at least two consecutive captures occurred.

CC1OF is also set to '1'.

An interrupt is generated depending on the CC1IE bit.

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

Note: Input capture interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

10.3.6 PWM input mode

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

Two ICx signals are mapped on the same TIx input

These 2 ICx signals are active on edges with opposite polarity

One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode

For example, the user can measure the 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 = 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 = 0 (active on rising edge).

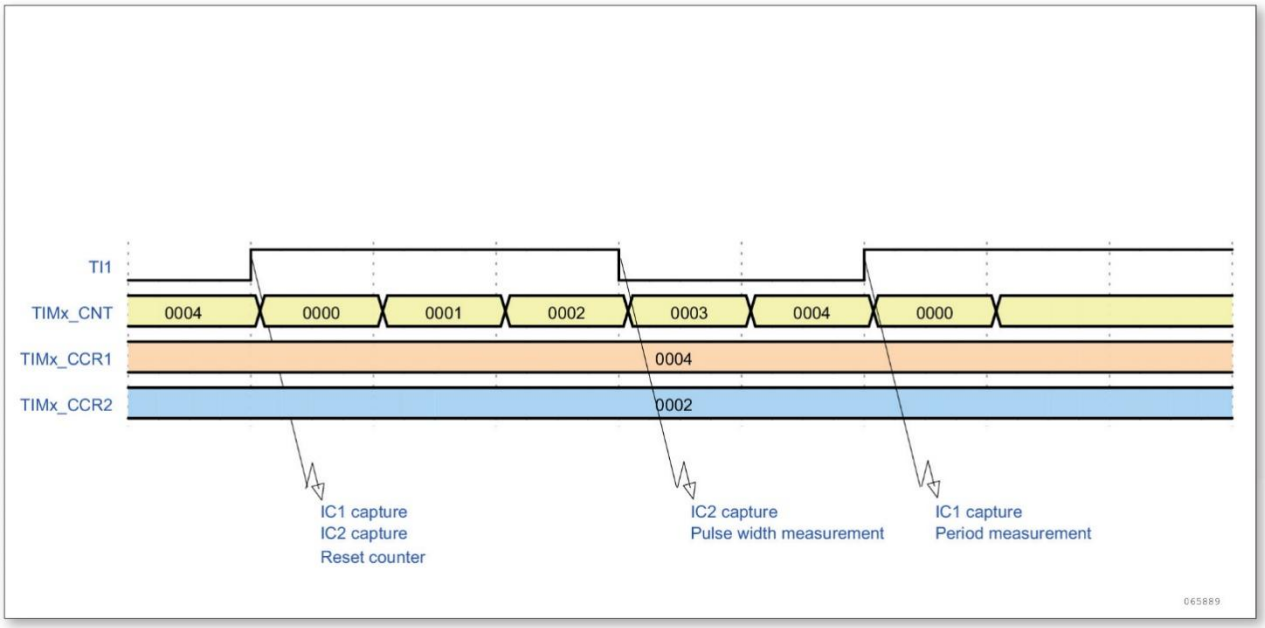
Select the active input for TIMx_CCR2: write the CC2S = 10 in the TIMx_CCMR1 register (TI1 selected).

Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P = 1 (active on falling edge).

Select the valid trigger input: write the TS = 101 in the TIMx_SMCR register (TI1FP1 selected).

Configure the slave mode controller in reset mode: write SMS = 100 in the TIMx_SMCR register.

Figure 10-27 PWM input mode timing



The PWM input mode can be used only with the TIMx_CH1/TIMx_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

10.3.7 Forced output mode

In output mode (CCxS bits = 00 in the TIMx_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write OCxM = 101 in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx gets 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 OCxM = 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 requests can be sent accordingly. This is described in the output compare mode section below.

10.3.8 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed. When a match is found between the capture/compare register and the counter, the output compare function:

Assign the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit 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.

Set a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).

Generate an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).

The TIMx_CCRx registers can be programd 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 synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

The output compare mode is configured as follows:

Select the counter clock (internal, external, prescaler)

Write the desired data in the TIMx_ARR and TIMx_CCRx registers

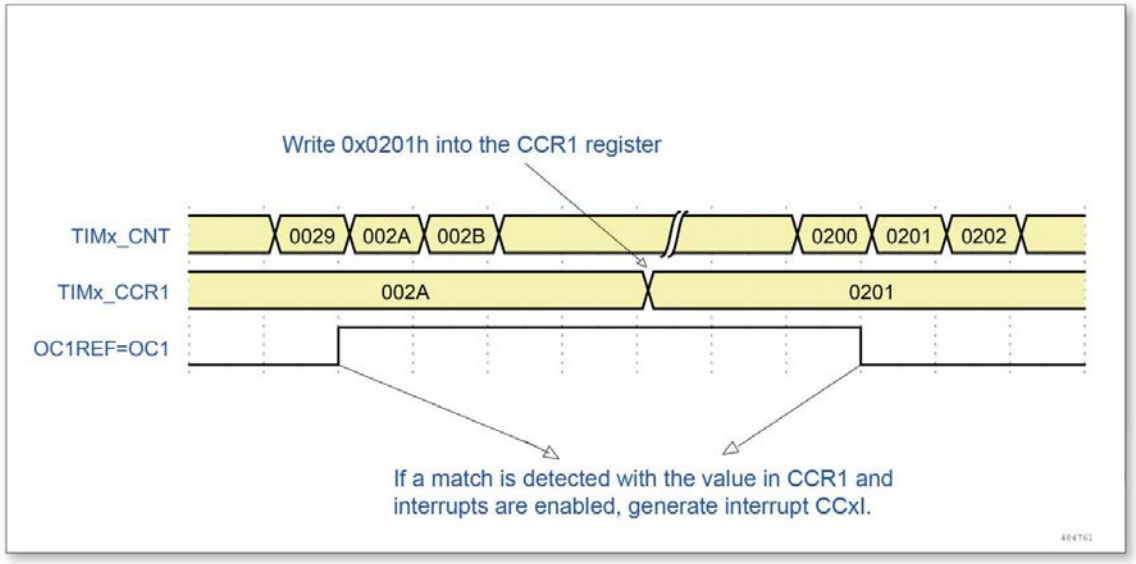
Set the CCxIE bit 3 if an interrupt request is to be generated

Select the output mode. For example, the user write OCxM = 011, OCxPE = 0, CCxP = 0 and CCxE = 1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high

Enable the counter 5 by setting the CEN bit in the TIMx_CR1 register

The TIMx_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE = 0, else TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

Figure 10-28 Output compare mode, toggle on OC1



10.3.9 PWM mode

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

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

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

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programd as active high or active low. OCx output is enabled by using the CCxE bit in the TIMx_CCER register. For details, refer to the TIMx_CCER register description.

In PWM mode (1 or 2), TIMx_CNT and TIM1_CCRx are always compared to determine whether TIM1_CCRx ≤ TIM1_CNT or TIM1_CNT ≤ TIM1_CCRx (depending on the direction of the counter). The OCxREF signal can be generated when:

When the result of the comparison changes

When the output compare mode (OCxM bit in the TIMx_CCMRx register) switches from “frozen” mode (no comparison, OCxM = 000) to “PWM” mode (OCxM = 110 or 111).

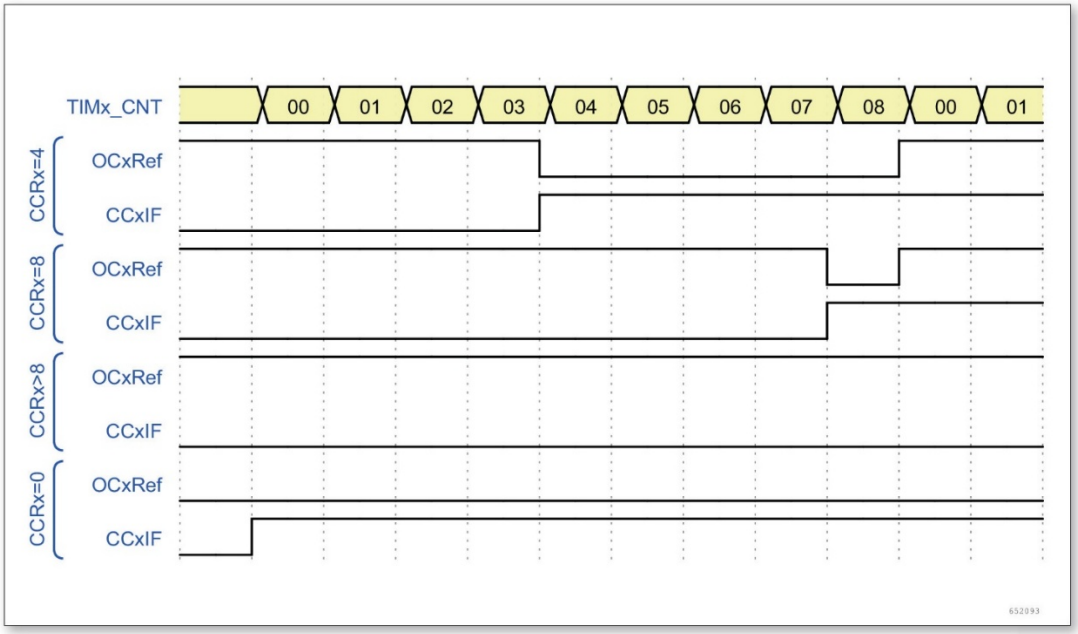
Therefore, the PWM output can be forced on the fly by software. The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

10.3.9.1 PWM edge-aligned mode

10.3.9.1.1 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'. The following figure shows some edge-aligned PWM waveforms in an example where TIMx_ARR = 8.

Figure 10-29 Edge-aligned PWM waveforms (ARR = 8)



10.3.9.1.2 Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.
In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx; Else it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at'1'. 0% PWM is not possible in this mode.

10.3.9.2 PWM center-aligned mode

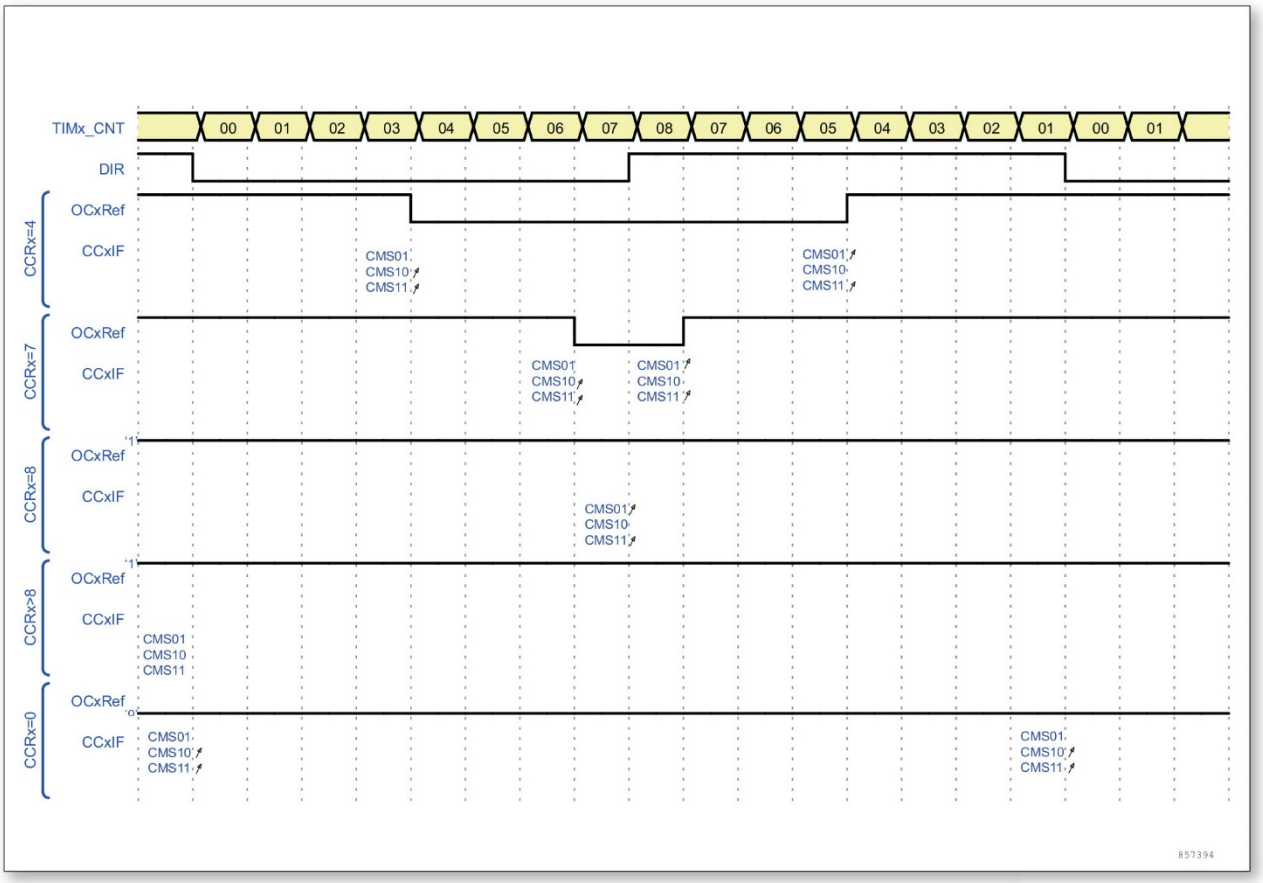
Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from '00' (all the remaining configurations having the same effect on the OCxRef/OCx signals).
The compare flag is set when the counter counts up, when it counts down or both when it counts up and down depending on the CMS bits configuration. The direction bit (DIR) in the TIMx_CR1 register is updated by hardware and must not be changed by software.
Refer to center-aligned mode section.
The figure below shows some center-aligned PWM waveforms in an example where:

TIMx_ARR = 8

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 10-30 Center-aligned PWM waveforms (ARR = 8)



Hints on using center-aligned mode:

When entering the center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.

Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:

- The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT > TIMx_ARR). For example, if the counter was counting up, it will continue to count up.
- The direction is updated if the user writes 0 or writes 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.

10.3.10 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

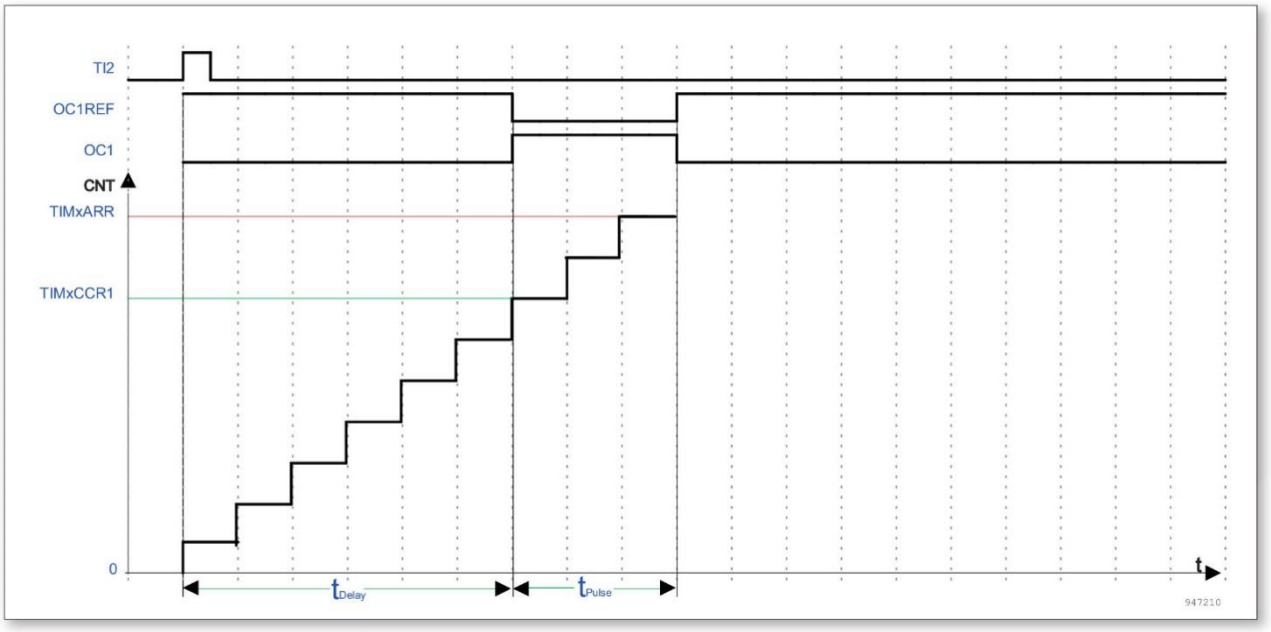
Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. Select One-pulse mode by setting the OPM bit in the TIMx_CR1 register. This makes the counter stop automatically at the next update event UEV.

A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

In upcounting: CNT < CCRx ≤ ARR (in particular, 0 < CCRx)

In downcounting: CNT > CCRx

Figure 10-31 Example of one pulse mode



For example the user may want to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a positive edge is detected on the TI2 input pin.

Assume TI2FP2 as trigger 1:

Map TI2FP2 to TI2 by writing CC2S = 01 in the TIMx_CCMR1 register

TI2FP2 must detect a rising edge, write CC2P = 0 in the TIMx_CCER register.

Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS = 110 in the TIMx_SMCR register.

TI2FP2 is used to start the counter by writing SMS = 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).

tDELAY is defined by the value written in the TIMx_CCR1 register.

tPULSE is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).

Let us say the user wants 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, enable PWM mode 2 by writing OC1M = 111 in the TIMx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE = 1 in the TIMx_CCMR1 register and ARPE in the TIMx_CR1 register. In this case the compare value must be written in the TIMx_CCR1 register, the auto-reload value in the TIMx_ARR register, generate an update by setting the UG bit and wait for external trigger event on TI2. CC1P is written to '1' in this example.

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

The user only wants one pulse, so '1' must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

Particular case: OCx fast enable:

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

If the user wants to output a waveform with the minimum delay, the OCxFE bit in the TIMx_CCMRx register must be set. Then OCxRef (and OCx) are forced in response to the stimulus, without taking in account the comparison. Its new level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

10.3.11 Encoder interface mode

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

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

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to table below. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence, the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR before starting. In the same way, the capture, compare, prescaler, trigger output features continue to work as normal.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction correspond to the rotation direction of the connected sensor.

The following table summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Table 10-1 Counting direction versus encoder signals

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI1FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally be used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The following figure 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:

C1S = ' 01' (TIMx_CCMR1 register, IC1FP1 mapped on TI1)

CC2S = ' 01' (TIMx_CCMR2 register, IC2FP2 mapped on TI2)

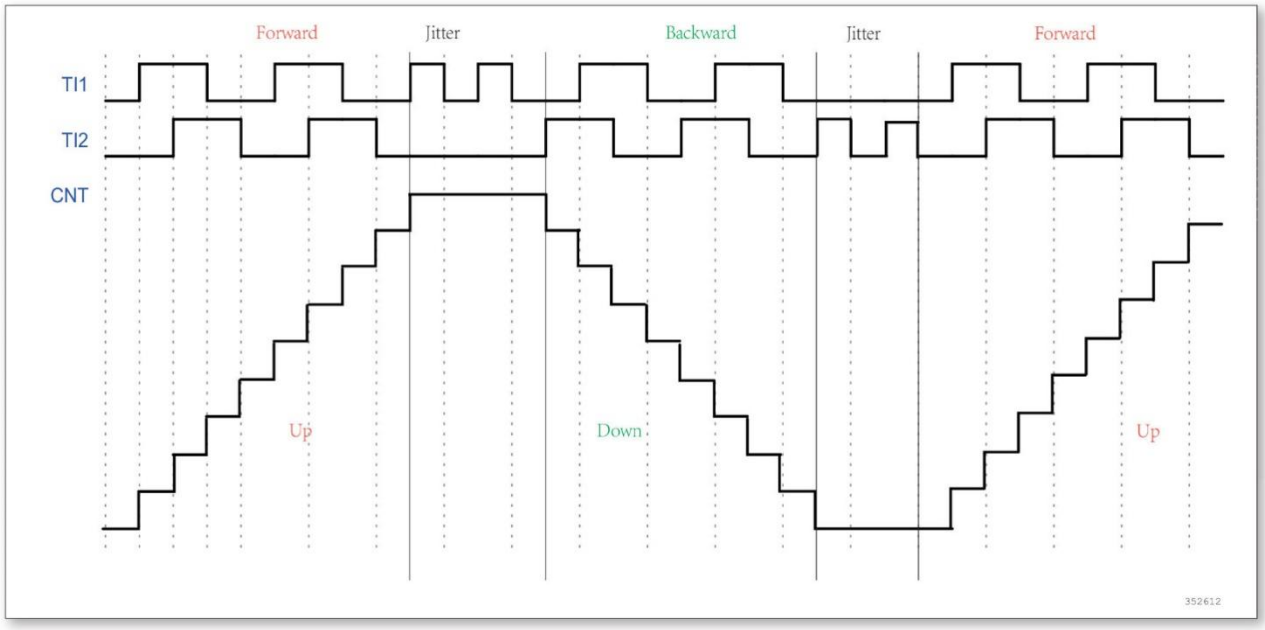
CC1P = ' 0' (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1 = TI1)

CC2P = ' 0' (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2 = TI2)

SMS = ' 011' (TIMx_SMCR register, both inputs are active on both rising and falling edges)

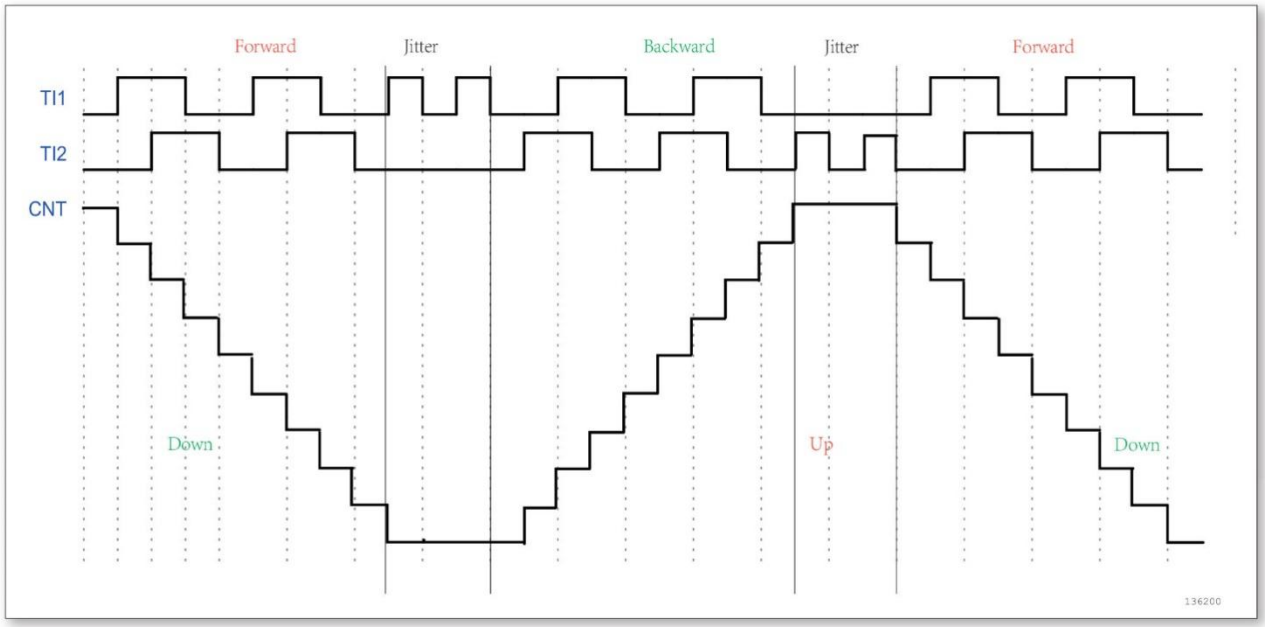
CEN = ' 1' (TIMx_CR1 register, counter enabled)

Figure 10-32 Example of counter operation in encoder interface mode



The following figure gives an example of counter behavior when IC1FP1 polarity is inverted (same configuration as above except CC1P =' 1').

Figure 10-33 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. The user can obtain dynamic information (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).

10.3.12 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.

10.3.13 Timers and external trigger synchronization

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

10.3.13.1 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 don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC1S bits select the input capture source only, i.e. CC1S = 01 in the TIMx_CCMR1 register. Write CC1P = 0 in TIMx_CCER register to validate the polarity (and detect rising edges only).

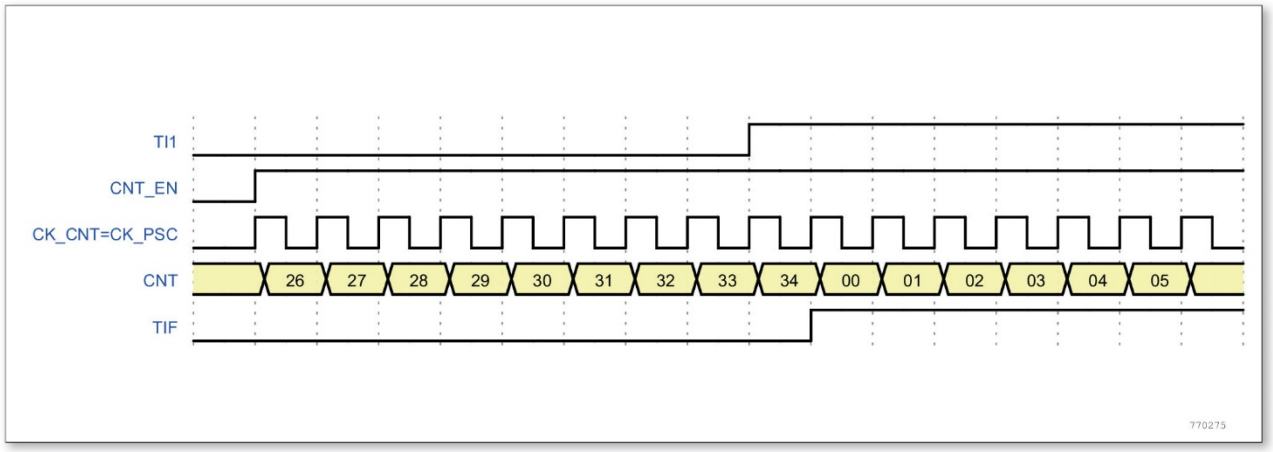
• Configure the timer in reset mode by writing SMS = 100 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.

• 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 depending on the TIE (interrupt enable) 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 10-34 Control circuit in reset mode



10.3.13.2 Slave mode: Gated mode

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

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

Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F = 0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only. Write CC1S = 01 in TIMx_CCMR1 register. Write CC1P = 1 in TIMx_CCER register to validate the polarity (and detect low level only).

Configure the timer in gated mode by writing SMS = 101 in TIMx_SMCR register. Select TI1 as the input source by writing TS = 101 in TIMx_SMCR register.

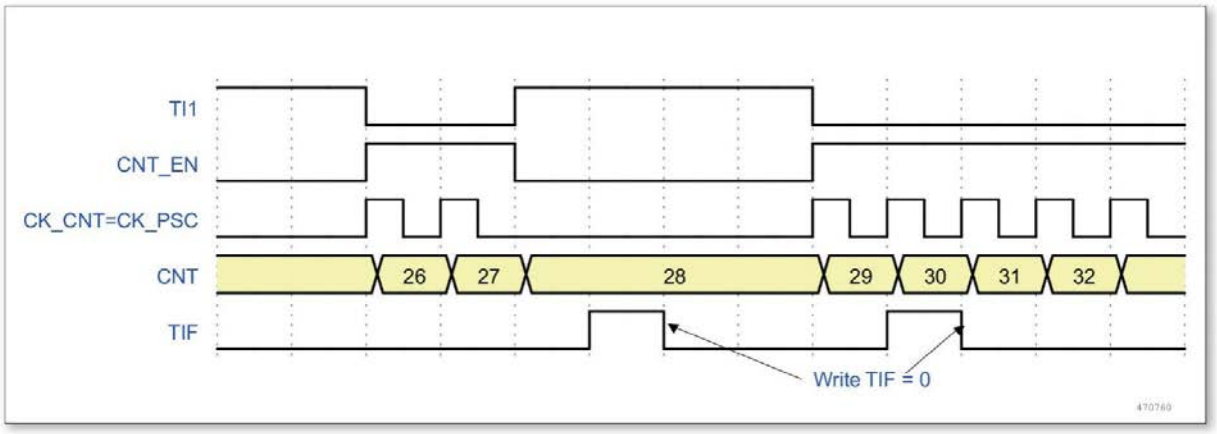
Enable the counter by writing CEN = 1 in the TIMx_CR1 register (in gated mode, the counter doesn't

start if CEN = 0, whatever is the trigger input level).

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set either 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 10-35 Control circuit in gated mode



10.3.13.3 Slave mode: Trigger mode

The counter can start in response to an event on a selected input.

In the following example, the upcounter starts in response to a rising edge on TI2 input:

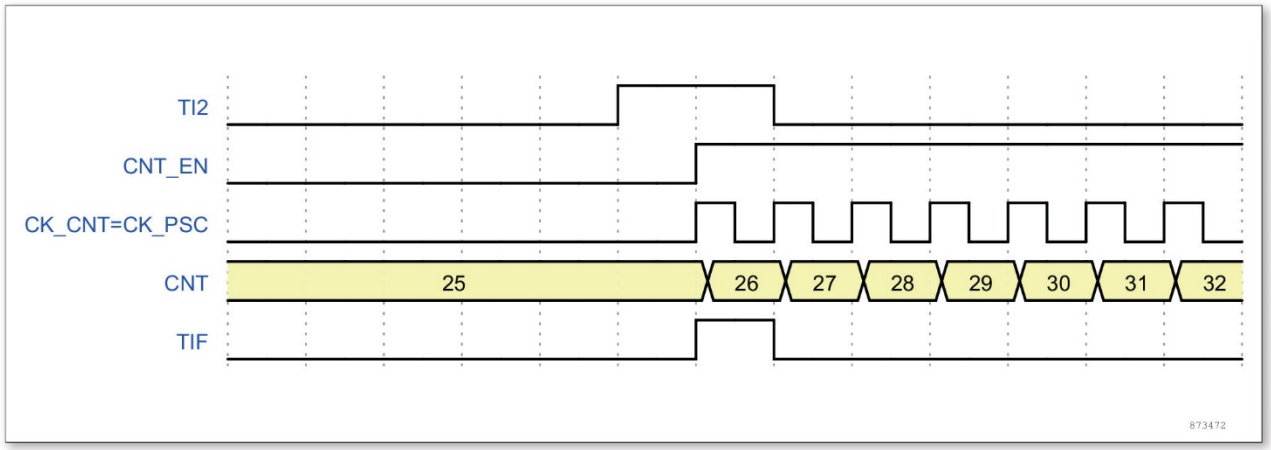
Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F = 0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits select the input capture source only. Write CC2S = 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 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 10-36 Control circuit in trigger mode

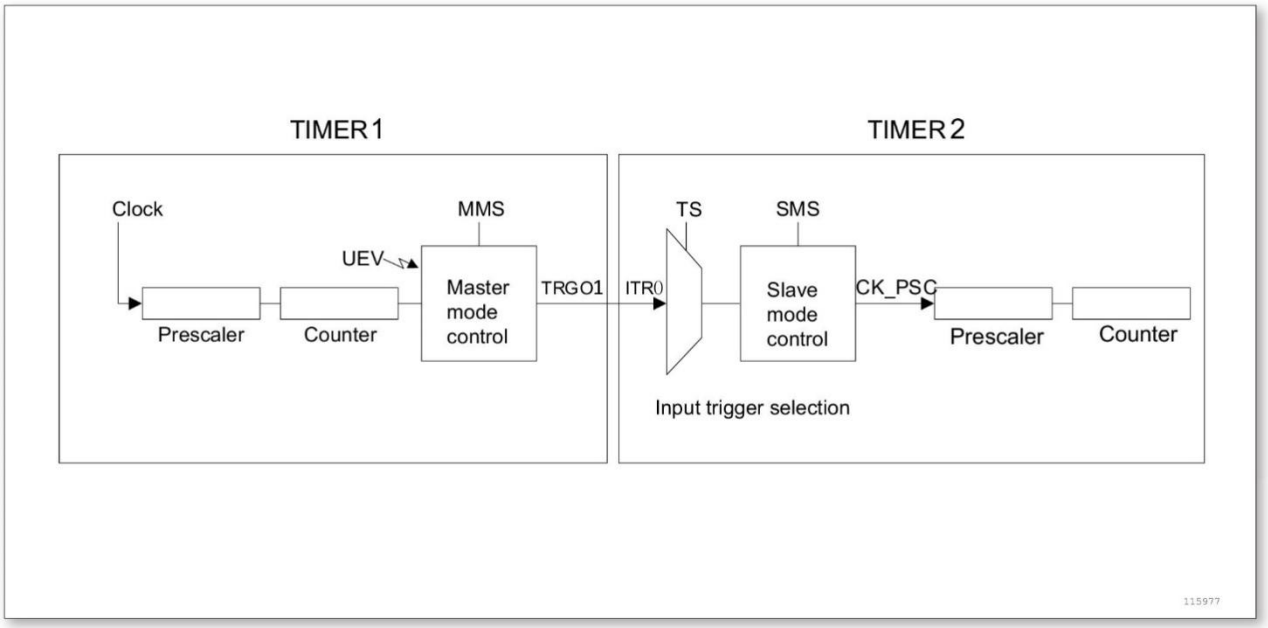


10.3.14 Timer synchronization

The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master mode, it can reset, start, stop or clock the counter of another Timer configured in Slave mode.

The figure below presents an overview of the trigger selection and the master mode selection blocks.

Figure 10-37 Master/Slave timer example



10.3.14.1 Using one timer as prescaler for another timer

For example, the user can configure Timer 1 to act as a prescaler for Timer 3. Refer to the above diagram and perform the following operations:

Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV. If you write MMS = 010 in the TIM1_CR2 register, a rising edge is output on TRGO1 each time an update event is generated.

To connect the TRGO1 output of Timer 1 to Timer 3, Timer 3 must be configured in slave mode using ITR1 as internal trigger. You select this through the TS bits in the TIM3_SMCR register (writing TS = 000).

Then you put the slave mode controller in external clock mode 1 (write SMS = 111 in the TIM3_SMCR register). This causes Timer 3 to be clocked by the rising edge of the periodic Timer 1 trigger signal (which corresponds to the Timer 1 counter overflow).

Finally both timers must be enabled by setting their respective CEN bits (TIMx_CR1 register).

Note: If OCx is selected on Timer 1 as trigger output (MMS = 1xx), its rising edge is used to clock the counter of Timer 3.

10.3.14.2 Using one timer to enable another timer

In this example, we control the enable of Timer 3 with the output compare of Timer 1. Refer to the diagram below. Timer 3 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3(fCK_CNT = fCK_INT/3) by the prescaler.

Configure Timer 1 master mode to send its Output compare 1 Reference (OC1REF) signal as trigger output (MMS = 100 in the TIM1_CR2 register)

Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register)

Configure Timer 3 to get the input trigger from Timer 1 (TS = 001 in the TIM3_SMCR register)

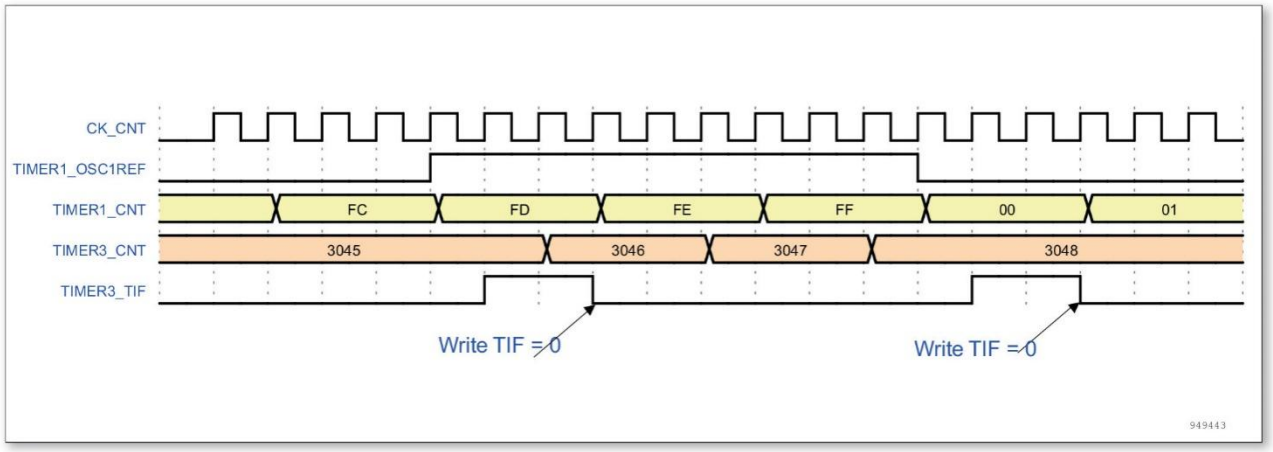
Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register)

Enable Timer 3 by writing '1' in the CEN bit (TIM3_CR1 register)

Start Timer 1 by writing '1' in the CEN bit (TIM1_CR1 register)

Note: The counter 3 clock is not synchronized with counter 1, this mode only affects the Timer 3 counter enable signal.

Figure 10-38 Gating timer 3 with OC1REF of timer 1



In the above example, the Timer 3 counter and prescaler are not initialized before being started. So they start counting from their current value. It is possible to start from a given value by resetting both timers before starting Timer 1. You can then write any value you want in the timer counters. The timers can easily be reset by writing the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 3. Timer 1 is the master and starts from 0. Timer 3 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 3 stops when Timer 1 is disabled by writing ‘0’ to the CEN bit in the TIM1_CR1 register.

Configure Timer 1 master mode to send its Output compare 1 Reference (OC1REF) signal as trigger output (MMS = 100 in the TIM1_CR2 register)

Configure the Timer 1 OC1REF waveform (TIM1_CCMR1 register)

Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register)

Configure Timer 3 in gated mode (SMS = 101 in TIM3_SMCR register)

Reset Timer 1 by writing ‘1’ in UG bit (TIM1_EGR register)

Reset Timer 3 by writing ‘1’ in UG bit (TIM3_EGR register)

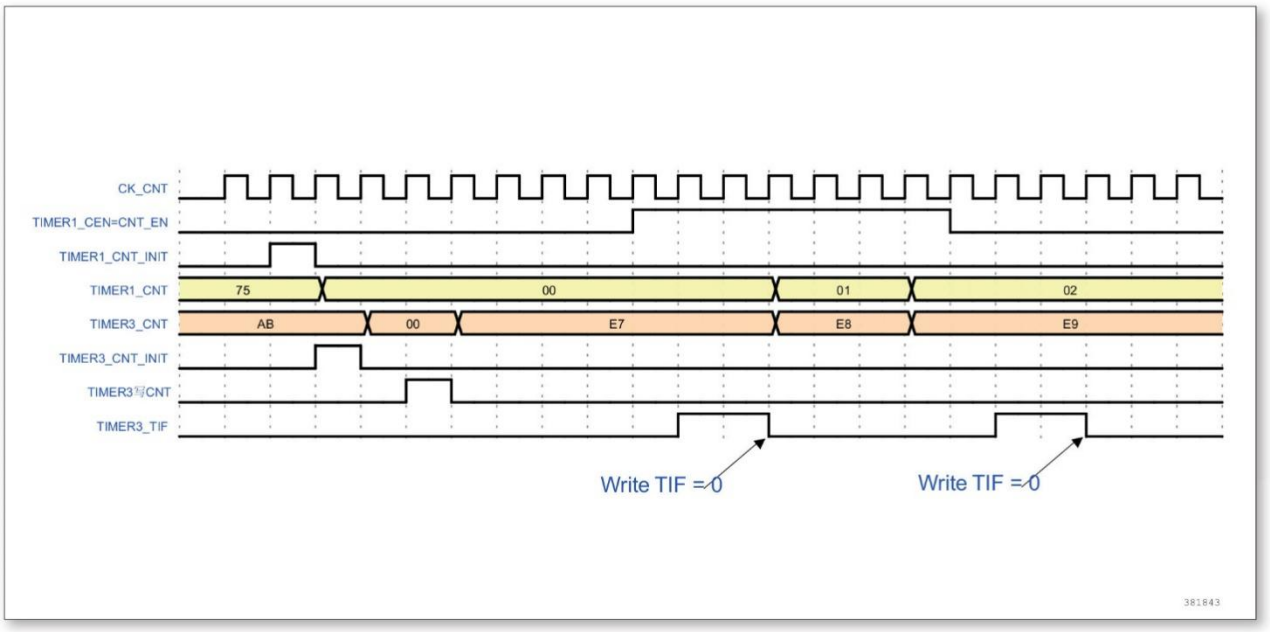
Initialize Timer 3 to 0xE7 by writing ‘0xE7’ in the timer 3 counter (TIM3_CNT)

Enable Timer 3 by writing ‘1’ in the CEN bit (TIM3_CR1 register)

Start Timer 1 by writing ‘1’ in the CEN bit (TIM1_CR1 register)

Stop Timer 1 by writing ‘0’ in the CEN bit (TIM1_CR1 register)

Figure 10-39 Gating timer 3 with enable of timer 1



10.3.14.3 Using one timer to start another timer

In this example, we set the enable of Timer 3 with the update event of Timer 1. Refer to the diagram below. Timer 3 starts counting from its current value (which can be nonzero) on the divided internal clock as soon as the update event is generated by Timer 1. When Timer 3 receives the trigger signal, its CEN bit is automatically set to‘1’ and the counter counts until we write ‘0’ to the CEN bit in the TIM3_CR1 register. Both counter clock frequencies are divided by 3(fCK_CNT=fCK_INT/3) by the prescaler.

Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS = 010 in the TIM1_CR2 register)

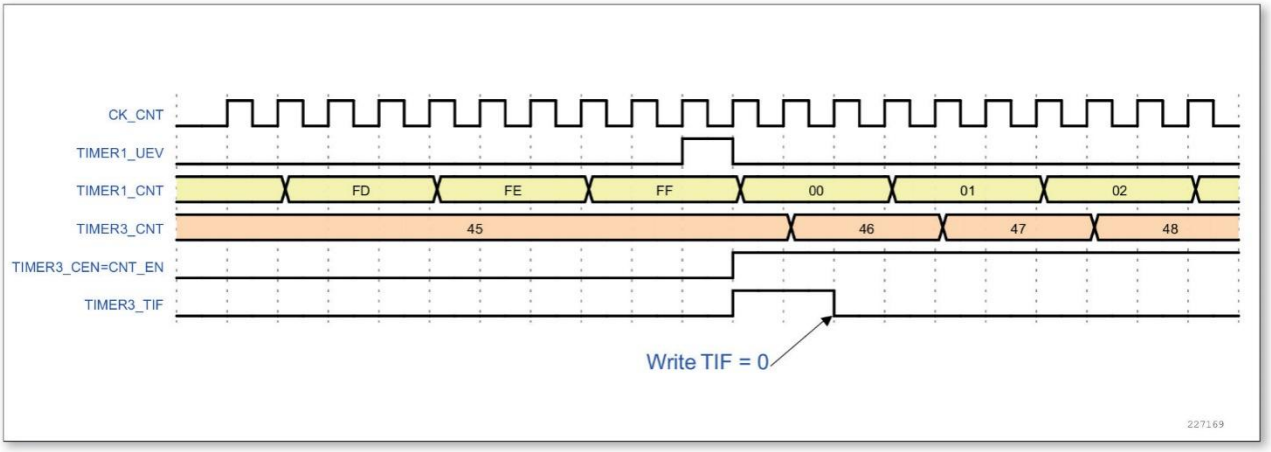
Configure the Timer 1 period (TIM1_ARR register)

Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register)

Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register)

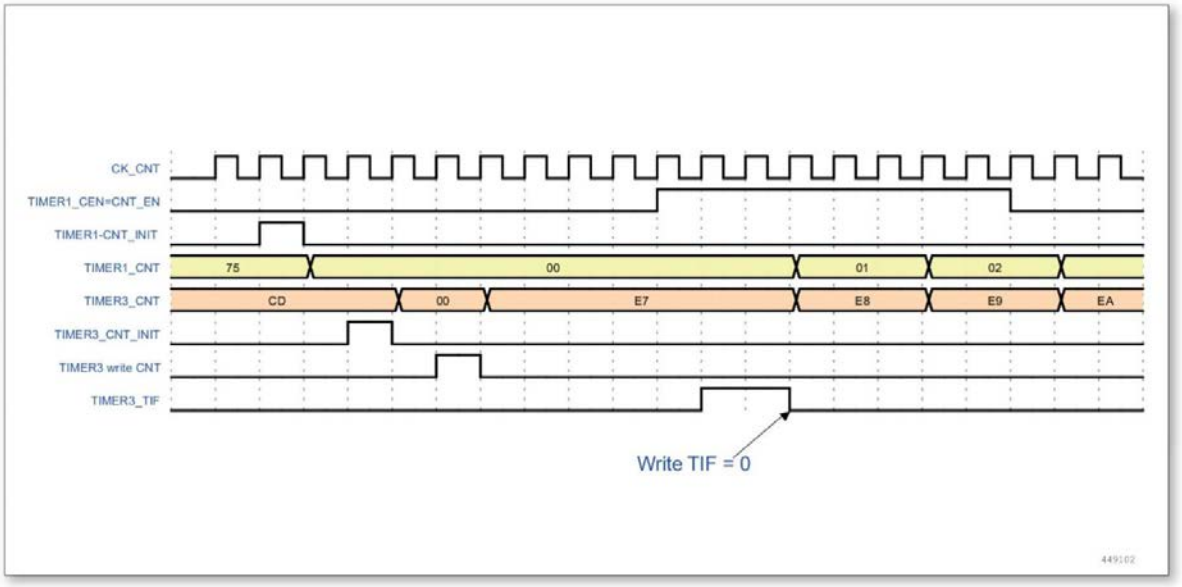
Start Timer 1 by writing ‘1’ in the CEN bit (TIM1_CR1 register)

Figure 10-40 Triggering timer 3 with update of timer 1



As in the previous example, the user can initialize both counters before starting counting. The diagram below shows the behavior with the same configuration but in trigger mode instead of gated mode (SMS = 110 in the TIM3_SMCR register).

Figure 10-41 Triggering timer 3 with enable of timer 1



10.3.14.4 Starting 2 timers synchronously in response to an external trigger

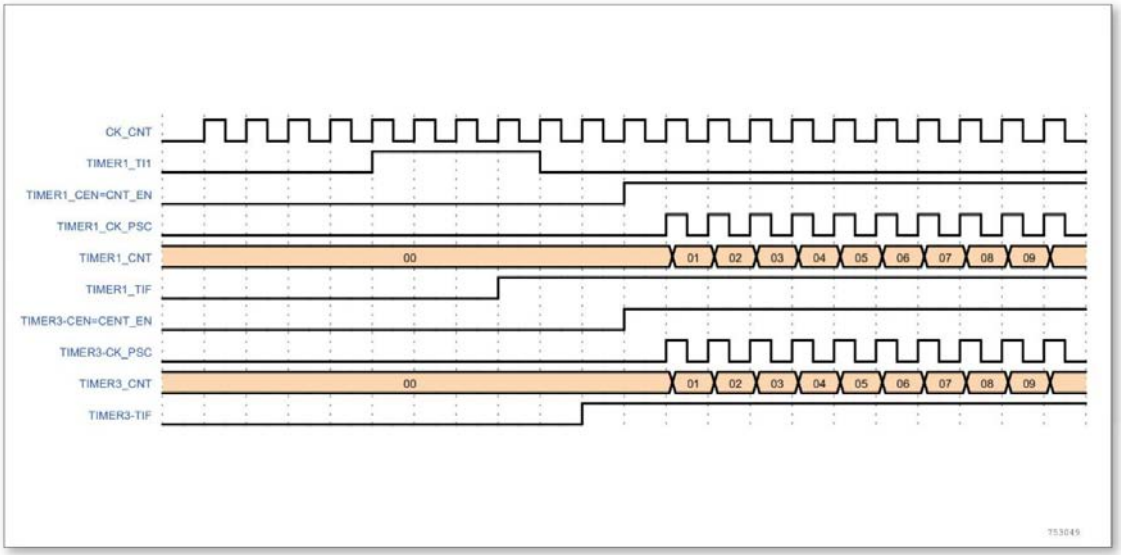
In this example, we set the enable of Timer 1 when its TI1 input rises, and the enable of Timer 3 with the enable of Timer 1. To ensure the counters are aligned, Timer 1 must be configured in Master/Slave mode (slave with respect to TI1, master with respect to Timer 3):

- Configure Timer 1 master mode to send its Enable as trigger output (MMS = 001 in the TIM1_CR2 register)
- Configure Timer 1 slave mode to get the input trigger from TI1 (TS = 100 in the TIM1_SMCR register)
- Configure Timer 1 in trigger mode (SMS = 110 in the TIM1_SMCR register)
- Configure Timer 3 to get the input trigger from Timer 1 (TS = 000 in the TIM3_SMCR register)
- Configure Timer 3 in trigger mode (SMS = 110 in TIM3_SMCR register)

When a rising edge occurs on TI1 (Timer 1), both counters starts counting synchronously on the internal clock and both TIF flags are set.

Note: In this example both timers are initialized before starting (by setting their respective UG bits). Both counters starts from 0, but you can easily insert an offset between them by writing any of the counter registers (TIMx_CNT). You can see from the following diagram that there is a delay between CNT_EN and CK_PSC on Timer 1 in the master/slave mode.

Figure 10-42 Triggering timer 1 and 3 with timer 1 TI1 input



10.3.15 Debug mode

When the microcontroller enters debug mode (CPU core - halted), the TIMx counter either continues to work normally or stops, depending on DBG_TIM3_STOP configuration bit in DBG module. For more details, refer to debug module section.

10.4 Register

Table 10-2 TIMx register overview

Offset	Acronym	Register Name	Reset
0x00	TIMx_CR1	Control Register 1	0x0000
0x04	TIMx_CR2	Control Register 2	0x0000
0x08	TIMx_SMCR	Slave Mode Control Register	0x0000
0x0C	TIMx_DIER	Interrupt Enable Register	0x0000
0x10	TIMx_SR	Status Register	0x0000
0x14	TIMx_EGR	Event Generation Register	0x0000
0x18	TIMx_CCMR1	Capture/Compare Mode Register 1	0x0000
0x1C	TIMx_CCMR2	Capture/Compare Mode Register 2	0x0000
0x20	TIMx_CCER	Capture/Compare Enable Register	0x0000
0x24	TIMx_CNT	Counter	0x0000
0x28	TIMx_PSC	Prescaler	0x0000

0x2C	TIMx_ARR	Auto Reload Register	0x0000
0x34	TIMx_CCR1	Capture/Compare Register 1	0x0000
0x38	TIMx_CCR2	Capture/Compare Register 2	0x0000
0x3C	TIMx_CCR3	Capture/Compare Register 3	0x0000
0x40	TIMx_CCR4	Capture/Compare Register 4	0x0000
0x50	TIMx_OR	Input Option Register	0x0000

10.4.1 TIMx_CR1 Control Register 1

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

Bit	Field	Description
15：10	Reserved	Reserved, must be kept at reset value.
9：8	CKD	Clock division 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 (TIx). 00：$t_{DTS} = t_{INT\_CK}$ 01：t_{DTS} =2x t_{INT_CK} 10：t_{DTS} =4x t_{INT_CK} 11：Reserved, do not program this value
7	ARPE	Auto reload preload enable 0: Disable the shadow register of TIMx_ARR register 1: Enable the shadow register of TIMx_ARR register
6：5	CMS	Center alignment mode selection 00: Edge alignment mode. Count direction depends on DIR bit 01: Central alignment mode 1. The counter alternatively conducts up and down count. The channel is in output mode. Only in down count, compare interrupt flag bit is set 10: Central alignment mode 2. The counter alternatively conducts up and down count. The channel is in output mode. Only in up count, compare interrupt flag bit is set 11: Central alignment mode 3. The counter alternatively conducts up and down count. The channel is in output mode. In up and down count, compare interrupt flag bit is set Note: During count, the alignment mode change is disabled.
4	DIR	Count direction 0: up count 1: down count Note: When the counter is configured as central alignment mode or encoder mode, the bit is read only.
3	OPM	one-pulse mode 0: Disable one-pulse mode. In case of update event, the counter count continues 1: Enable one-pulse mode. In case of the next update event (clear CEN bit), the counter count stops
2	URS	Update request source This bit is set and cleared by software, select update event source. 0: The event below may generate a update interrupt request: - Counter overflow/underflow - Set UG bit - Update generation through the slave mode controller 1: Only in counter overflow/underflow, generate update interrupt request
1	UDIS	Update disable This bit is used to enable or disable the update event 0: Update event (UEV) enabled.

		1: Update event 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 EGR.UG bit is set or a hardware reset is received from the slave mode controller.
0	CEN	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.

10.4.2 TIMx_CR2 Control Register 2

Address offset: 0x04
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TI1S	MMS			Reserved			
								rw	rw						

Bit	Field	Description
15：8	Reserved	Reserved, must be kept at reset value.
7	TI1S	TI1 selection 0: TIMx_CH1 pin is connected to TI1 input 1: TIMx_CH1, TIMx_CH2 and TIMx_CH3 pins are connected to the TI1 input after XOR combination
6：4	MMS	Master mode selection These bits control TRGO signal selection, used to select the sync information sent to the slave timers in master mode: 000: Reset TIMx_EGR register UG bit generate one pulse trigger output (TRGO). 001: Enable The Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected. 010: Update Update event is selected as TRGO. 011: Capture/compare When a capture or a compare match occurred, send a positive pulse as TRGO. 100: Compare OC1REF signal is used as trigger output (TRGO) 101: Compare OC2REF signal is used as trigger output (TRGO) 110: Compare OC3REF signal is used as trigger output (TRGO) 111: Compare OC4REF signal is used as trigger output (TRGO)
3：0	Reserved	Reserved, must be kept at reset value.

10.4.3 TIMx_SMCR Slave Mode Control Register

Address offset: 0x08
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MSM	TS			Res.	SMS		
								rw	rw				rw		

Bit	Field	Description
15：8	Reserved	Reserved, must be kept at reset value.
7	MSM	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	Trigger selection 000：Internal trigger 0 (ITR0) 001：Internal trigger 1 (ITR1) 010：Internal trigger 2 (ITR2) 011：Internal trigger 3 (ITR3) 100：TI1 edge detector (TI1F_ED) 101：Timer input after filter 1 (TI1FP1) 110：Timer input after filter 2 (TI2FP2) 111：External trigger input (ETR) others：Reserved More details on ITRx refer to the table below. Note: After the slave mode enable, these bits cannot be changed
3	Reserved	Reserved, must be kept at reset value.
2：0	SMS	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. 000: Close slave mode - In case of CEN =1, the prescaler is directly driven by the internal clock. 001: Encoder mode 1 - Based on TI1FP1 level, the counter conducts up/down count along TI2FP2 edge. 010: Encoder mode 2 - Based on TI2FP2 level, the counter conducts up/down count along TI1FP1edge. 011: Encoder mode 3- Based on another input level, the counter conducts up/down count along TI1FP1 and TI2FP2 edge. 100: Reset mode -The rising edge of the selected trigger input (TRGI) reinitialize the counter and generate an update register signal. 101: Gate mode - When the trigger input (TRGI) is high, the counter count begins. When the trigger input turns low, the counter count stops (but without reset). The counter start and stop are controlled. 110: Trigger mode -The counter starts in the rising edge of the trigger input TRGI (but without reset). Only the counter start is controlled. 111: External clock mode 1 -The rising edge of the selected trigger input (TRGI) drives the counter. Note: If TI1F_EN is selected as trigger input (TS =100), don't use the gate mode. This is because that EI1F_ED exports a pulse at each TI1F change. However, in the gate mode, it's required to check the trigger input level.

Table 10-3 TIMx internal trigger connection

Slave TIM	ITR0	ITR1	ITR2	ITR3
TIM3	TIM1	-	-	-

10.4.4 TIMx_DIER Interrupt Enable Register

Address offset: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
									rw		rw	rw	rw	rw	rw

Bit	Field	Description
15：7	Reserved	Reserved, must be kept at reset value.
6	TIE	Trigger interrupt enable 0: Break interrupt disable 1: Break interrupt enable

5	Reserved	Reserved, must be kept at reset value.
4	CC4IE	Enable capture/compare 4 interrupt 0: Capture/compare interrupt 4 disable 1: Capture/compare interrupt 4 enable
3	CC3IE	Enable capture/compare 3 interrupt 0: Capture/compare interrupt 3 disable 1: Capture/compare interrupt 3 enable
2	CC2IE	Enable capture/compare 2 interrupt 0: Capture/compare interrupt 2 disable 1: Capture/compare interrupt 2 enable
1	CC1IE	Enable capture/compare 1 interrupt 0: Capture/compare interrupt 1 disable 1: Capture/compare interrupt 1 enable
0	UIE	Enable update interrupt 0: Update event interrupt disable 1: Update event interrupt enable

10.4.5 TIMx_SR Status Register

Address offset: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Reserved		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			r_w0c						r_w0c		r_w0c				

Bit	Field	Description
15：13	Reserved	Reserved, must be kept at reset value.
12	CC4OF	Capture/compare 4 over capture flag Refer to CC1OF description.
11	CC3OF	Capture/compare 3 over capture flag Refer to CC1OF description.
10	CC2OF	Capture/compare 2 over capture flag Refer to CC1OF description.
9	CC1OF	Capture/compare 1 over capture flag Only when the channel 1 is configured as input capture mode and CC1F is configured as 1, and the capture event occurs again. The flag is set by hardware. Write 0 may clear the bit. 0: No overcapture 1: Overcapture
8：7	Reserved	Reserved, must be kept at reset value.
6	TIF	Trigger interrupt flag In case of trigger event (When the slave mode counter is in any other mode except for the gate mode, detect the active edge in TRGI input terminal, or detect any edge in gate mode), the bit is set by hardware. It's cleared by software. 0: No trigger event occurred 1: Trigger interrupt pending
5	Reserved	Reserved, must be kept at reset value.
4	CC4IF	Capture/compare 4 interrupt flag Refer to CC1IF description
3	CC3IF	Capture/compare 3 interrupt flag Refer to CC1IF description
2	CC2IF	Capture/compare 2 interrupt flag Refer to CC1IF description.
1	CC1IF	Capture/compare 1 interrupt flag Channel 1 in output mode: When the counter value and compare value match, the bit is set by hardware except for the central alignment mode (the bit is set in the central alignment mode according to TIMx_CR1.CMS [1:0]). It's cleared by software. 0: No match 1: TIMx_CNT value and TIMx_CCR1 value match Channel 1 as output mode:

		In case of capture event, the bit is set by hardware. It's cleared by software or reading TIMx_CCR1 register 0: No input capture occurred 1: Counter value is captured to TIMx_CCR1
0	UIF	Update interrupt flag In case of update event, the bit is set by hardware. It's cleared by software. 0: No update interrupt occurred 1: update interrupt pending This bit is set by hardware when the registers are updated: - In case of TIMx_CR1 register UDIS=0 and REP_CNT=0, when the counter generates overflow/underflow. -In case of TIMx_CR1 register UDIS=0, URS=0, when TIMx_EGR register UG =1. - In case of TIMx_CR1 register UDIS=0, URS=0, when update generation through the slave mode controller.

10.4.6 TIMx_EGR Event Generation Register

Address offset: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

Bit	Field	Description
15：7	Reserved	Reserved, must be kept at reset value.
6	TG	Trigger generation 0: No action 1: generate a trigger event. TIMx_SR register TIF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. It's auto cleared by hardware.
5	Reserved	Reserved, must be kept at reset value.
4	CC4G	Capture/compare 4 generation Refer to CC1G description.
3	CC3G	Capture/compare 3 generation Refer to CC1G description.
2	CC2G	Capture/compare 2 generation Refer to CC1G description.
1	CC1G	Capture/compare 1 generation Refer to CC1G description. This bit is set by software. It generates a capture/compare event, and is auto cleared by hardware. 0: No action 1：Generate a capture/compare event in channel CC1: When the channel CC1 is configured as output: Set CC1IF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. When the channel CC1 is configured input: The current value of the counter is captured in TIMx_CCR1 register, set CC1IF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. When CC1IF is set, set CC1OF =1.
0	UG	Update event generation 0：No action 1: Initialize the counter, and generate an update event. It's auto cleared by hardware. When selecting the central alignment or up counting mode, the counter is clear; otherwise (down counting mode) the counter auto reload value is loaded. The prescaler counter is cleared at the same time.

10.4.7 TIMx_CCMR1 Capture/Compare Mode Register 1

Address offset: 0x18

Reset value: 0x0000

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

Res.	OC2M	OC2PE	OC2FE	CC2S	Res.	OC1M	OC1PE	OC1FE	CC1S
IC2F		IC2PSC		CC2S	IC1F		IC1PSC		CC1S
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

The channel may be used for input (capture mode) or output (compare mode). The channel direction is defined by the corresponding CCxS. The register's bit other than CCxS functions differently in the input mode and output mode. OCxx describes the channel function in the output mode. ICxx describes the channel function in the input mode.

Output compare mode:

Bit	Field	Description
15	Reserved	Reserved, must be kept at reset value.
14：12	OC2M	Channel 2 output and compare mode Refer to OC1M description.
11	OC2PE	Channel 2 output and compare preload enable Refer to OC1PE description.
10	OC2FE	Channel 2 output and compare quick enable Refer to OC1FE description.
9：8	CC2S	Channel 2 capture/compare 2 selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00：Channel 2 is configured as output 01：Channel 2 is configured as input, IC2 is mapped on TI2 10：Channel 2 is configured as input, IC2 is mapped on TI1 11：Channel 2 is configured as input, IC2 is mapped on TRC. The mode only works when the internal generator input is selected (TIMx_SMCR register TS bit selection)
7	Reserved	Reserved, must be kept at reset value.
6：4	OC1M	Channel 1 output compare mode The bit has defined the output reference signal OC1REF action. OC1REF has determined OC1 and OC1N value. OC1REF is active at high level. The active level of OC1, OC1N depends on CC1P and CC1NP bit. 000：Freeze. TIMx_CCR1 and TIMx_CNT compare results has no effect on OC1REF. 001: Set as high when configuration. When TIMx_CNT value and TIMx_CCR1 value are same, enforce OC1REF as high level 010: Set as low when configuration. When TIMx_CNT value and TIMx_CCR1 value are same, enforce OC1REF as low level 011: Toggle when match. When TIMx_CCR1=TIMx_CNT, OC1REF toggle. 100: Enforce as low. Enforce OC1REF at low level 101: Enforce as high. Enforce OC1REF at high level 110：PWM mode 1. During up count, in case of TIMx_CNT<TIMx_CCR1, enforce OC1REF is at high level. Or else, it's at low level. During down count, in case of TIMx_CNT > TIMx_CCR1, enforce OC1REF is at low level. Or else, it's at high level. 111：PWM mode 2. During up count, in case of TIMx_CNT<TIMx_CCR1, channel 1 enforce OC1REF is at low level. Or else, it's at high level. During down count, in case of TIMx_CNT > TIMx_CCR1, enforce OC1REF is at high level. Or else, it's at low level. Note 1: In case of LOCK level 3 (TIMx_BDTR register LOCK bit) and CC1S = 00 (the channel is configured as output), the bit cannot be changed. Note 2: In PWM mode 1 or PWM mode 2, only when the compare result changes or it changes over from the freeze mode to PWM mode in the output compare mode, OC1REF level may change.
3	OC1PE	Channel 1 output compare preload enable 0: Disable TIMx_CCR1 register preload function. The value written into TIMx_CCR1 register becomes active immediately 1: Enable TIMx_CCR1 register preload function. Only read and write the preload register. TIMx_CCR1 preload value becomes active at the update event

		Note 1: When the LOCK level is 3 (TIMx_BDTR register LOCK bit) and CC1S = 00 (when the channel is configured as output), the bit cannot be changed. Note 2: Only in the one-pulse mode (TIMx_CR1 register OPM= 1), it has no influence whether the preload register is set. Under other scenarios, it's required to set the preload register. Otherwise, the follow up action is not certain.
2	OC1FE	Channel 1 output compare quick enable In case of the bit is set 1, when the channel is configured as PWM mode, it speeds up response of the capture/compare output to the trigger time. The output channel deems the active edge of the trigger input signal as one compare match. Therefore, OC is set as compare level, but is irrelevant to the compare result. 0: channel 1 output compare fast enable. 1: channel 1 output compare fast disable.
1：0	CC1S	Channel 1 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00：Channel 1 is configured as output 01：Channel 1 is configured as input, IC1 is mapped on TI1 10：Channel 1 is configured as input, IC1 is mapped on TI2 11：Channel 1 is configured as input, IC1 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)

Input capture mode:

Bit	Field	Description
15：12	IC2F	Input capture 2 filter Refer to IC1F description
11：10	IC2PSC	Input/capture 2 prescaler Refer to IC1PSC description
9：8	CC2S	Channel 2 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00：Channel 2 is configured as output 01：Channel 1 is configured as input, IC2 is mapped on TI2 10：Channel 2 is configured as input, IC2 is mapped on TI1 11：Channel 2 is configured as input, IC2 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)
7：4	IC1F	Channel 1 input capture filter The digital filter is composed of an event counter. It records an output jump after N input events. These bits define IC1 input signal sampling frequency and digital filter length. 0000: No filter, f_{DTS} sampling 001: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =2 0010: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =4 0011: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =8 0100: Sampling frequency $f_{sampling}$ =$f_{DTS/2}$, N =6 0101: Sampling frequency $f_{sampling}$ =$f_{DTS/2}$, N =8 0110: Sampling frequency $f_{sampling}$ =$f_{DTS/4}$, N =6 0111: Sampling frequency $f_{sampling}$ =$f_{DTS/4}$, N =8 1000: Sampling frequency $f_{sampling}$ =$f_{DTS/8}$, N =6 1001:Sampling frequency $f_{sampling}$ =$f_{DTS/8}$, N =8 1010: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =5 1011: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =6 1100: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =8 1101: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =5 1110: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =6 1111: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =8
3：2	IC1PSC	Channel 1 input/capture prescaler This bit defines the ratio of the prescaler acting on IC1.The prescaler is reset as soon as CC1E=0(TIMx_CCER register).

		00: No prescaler, the capture input detects each edge to trigger one capture 01: Trigger one capture for every 2 events 10: Trigger one capture for every 4 events 11: Trigger one capture for every 8 events
1：0	CC1S	Channel 1 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00: Channel 1 is configured as output 01: Channel 1 is configured as input, IC1 is mapped on TI1 10: Channel 1 is configured as input, IC1 is mapped on TI2 11: Channel 1 is configured as input, IC1 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)

10.4.8 TIMx_CCMR2 Capture/Compare Mode Register 2

Address offset: 0x1C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OC4M			OC4PE	OC4FE	CC4S		Res.	OC3M			OC3PE	OC3FE	CC3S	
IC4F				IC4PSC		CC4S		IC3F				IC3PSC		CC3S	
rw	rw			rw	rw	rw		rw	rw			rw	rw	rw	

Compare output mode:

Bit	Field	Description
15	Reserved	Reserved, must be kept at reset value.
14：12	OC4M	Channel 4 output compare mode Refer to OC3M description
11	OC4PE	Channel 4 output compare preload enable Refer to OC3PE description
10	OC4FE	Channel 4 output compare rapid enable Refer to OC3FE description
9：8	CC4S	Channel 4 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00: Channel 4 is configured as output 01: Channel 4 is configured as input, IC4 is mapped on TI4 10: Channel 4 is configured as input, IC4 is mapped on TI3 11: Channel 4 is configured as input, IC4 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection
7	Reserved	Reserved, must be kept at reset value.
6：4	OC3M	Channel 3 output compare mode The bit has defined the output reference signal OC3REF action. OC3REF has determined OC3 and OC3N value. OC3REF is active at high level. The active level of OC3, OC3N depends on CC3P and CC3NP bit. 000：Freeze. TIMx_CCR3 and TIMx_CNT compare results has no effect on OC3REF. 001: Set as high when configuration. When TIMx_CNT value and TIMx_CCR3 value are same, enforce OC3REF as high level 010: Set as low when configuration. When TIMx_CNT value and TIMx_CCR3 value are same, enforce OC3REF as low level 011: Toggle when match. When TIMx_CCR3=TIMx_CNT, OC3REF toggle. 100: Enforce as low. Enforce OC3REF at low level 101: Enforce as high. Enforce OC3REF at high level 110: PWM mode 1. During up count, in case of TIMx_CNT<TIMx_CCR3, enforce OC3REF as high level, otherwise its low level; during down count, in case of TIMx_CNT > TIMx_CCR3, enforce OC3REF as low level, otherwise its high level. 111: PWM mode 2. During up count, in case of TIMx_CNT<TIMx_CCR3, enforce OC3REF as low level, otherwise its high level; during down count, in

		case of TIMx_CNT > TIMx_CCR3, enforce OC3REF as high level, otherwise its low level. Note 1: In case of LOCK level 3 (TIMx_BDTR register LOCK bit) and CC3S = 00 (the channel is configured as output), the bit cannot be changed. Note 2: In PWM mode 1 or PWM mode 2, only when the compare result changes or it changes over from the freeze mode to PWM mode in the output compare mode, OC3REF level may change.
3	OC3PE	Channel 3 output compare preload enable 0: Disable TIMx_CCR3 register preload function, the value written into TIMx_CCR3 register becomes active immediately 1: Enable TIMx_CCR3 register preload function. The read and write only works on the preload register. TIMx_CCR3 preload value becomes active when the update event is available Note 1: When the LOCK level is set 3 (TIMx_BDTR register LOCK bit) and CC3S =0 (the channel is configured as output), the bit cannot be changed. Note 2: Only in the one-pulse mode (TIMx_CR1 register OPM= 1), it's not required to set the preload register. Under other scenarios, it's required to set the preload register. Otherwise, the follow up action is not certain.
2	OC3FE	Channel 3 output compare quick enable In case of the bit 1, when the channel is configured as PWM mode, it speeds up response of the capture/compare output to the trigger time. The output channel deems the active edge of the trigger input signal as one compare match. Therefore, OC is set as compare level, but is irrelevant to the compare result. 0: channel 3 output compare fast disable. 1: channel 3 output compare fast enable.
1：0	CC3S	Channel 3 capture/compare selection The bit has defined the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00: Channel 3 is configured as input 01: Channel 3 is configured as input, IC3 is mapped on TI3 10: Channel 3 is configured as input, IC3 is mapped on TI4 11: Channel 3 is configured as input, IC3 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)

Input capture mode:

Bit	Field	Description
15：12	IC4F	Input capture 4 filter Refer to IC3F description
11：10	IC4PSC	Input/capture 4 prescaler Refer to IC3PSC description
9：8	CC4S	Channel 4 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00：Channel 4 is configured as output 01：Channel 4 is configured as input, IC4 is mapped on TI4 10：Channel 4 is configured as input, IC2 is mapped on TI3 11：Channel 4 is configured as input, IC4 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)
7：4	IC3F	Channel 3 input capture filter The digital filter is composed of an event counter. It records an output jump after N input events. These bits define IC1 input signal sampling frequency and digital filter length. 0000: No filter, f_{DTS} sampling 001: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =2 0010: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =4 0011: Sampling frequency $f_{sampling}$ =f_{INT_CK}, N =8 0100: Sampling frequency $f_{sampling}$ =$f_{DTS/2}$, N =6 0101: Sampling frequency $f_{sampling}$ =$f_{DTS/2}$, N =8 0110: Sampling frequency $f_{sampling}$ =$f_{DTS/4}$, N =6 0111: Sampling frequency $f_{sampling}$ =$f_{DTS/4}$, N =8

		1000: Sampling frequency $f_{sampling}$ =$f_{DTS/8}$, N =6 1001:Sampling frequency $f_{sampling}$ =$f_{DTS/8}$, N =8 1010: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =5 1011: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =6 1100: Sampling frequency $f_{sampling}$ =$f_{DTS/16}$, N =8 1101: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =5 1110: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =6 1111: Sampling frequency $f_{sampling}$ =$f_{DTS/32}$, N =8
3：2	IC3PSC	Channel 3 input/capture prescaler This bit defines the ratio of the prescaler acting on IC3.The prescaler is reset as soon as CC3E=0(TIMx_CCER register). 00: No prescaler, the capture input detects each edge to trigger one capture 01: Trigger one capture for every 2 events 10: Trigger one capture for every 4 events 11: Trigger one capture for every 8 events
1：0	CC3S	Channel 3 capture/compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00: Channel 3 is configured as output 01: Channel 3 is configured as input, IC3 is mapped on TI3 10: Channel 3 is configured as input, IC3 is mapped on TI4 11: Channel 3 is configured as input, IC3 is mapped on TRC. This mode only works when the internal trigger input is selected (TIMx_SMCR register TS bit selection)

10.4.9 TIMx_CCER Capture/Compare Enable Register

Address offset: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
rw		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

Bit	Field	Description
15	CC4NP	Channel 4 input/capture complementary output polarity Refer to CC1NP description
14	Reserved	Reserved, must be kept at reset value.
13	CC4P	Channel 4 capture/compare output polarity Refer to CC1P description.
12	CC4E	Channel 4 capture/compare enable Refer to CC1E description
11	CC3NP	Channel 3 input/capture complementary output polarity Refer to CC1NP description
10	Reserved	Reserved, must be kept at reset value.
9	CC3P	Channel 3 input capture output polarity Refer to CC1P description
8	CC3E	Channel 3 input/capture output enable Refer to CC1E description
7	CC2NP	Channel 2 input/capture complementary output polarity Refer to CC1NP description
6	Reserved	Reserved, must be kept at reset value.
5	CC2P	Channel 2 input capture output polarity Refer to CC1P description
4	CC2E	Channel 2 input/capture output enable Refer to CC1E description
3	CC1NP	Channel 1 input/capture complementary output polarity When channel 1 is configured as output, this bit has defined the input signal polarity:

		0: OC1N is active at high level 1: OC1N is active at low level When channel 1 is configured as input, CC1P/CC1NP match use has defined the input signal polarity and level. Details are given in the ICx polarity/level selection table. Note: When LOCK level (TIMx_BDTR register LCCK bit) is set 3 or 2 and CC1S = 00 (channel is configured as output), the bit cannot be changed.
2	Reserved	Reserved, must be kept at reset value.
1	CC1P	Channel 1 input/capture output polarity When channel 1 is configured as output, the bit defines the output signal polarity: 0: OC1 is active at high level 1: OC1 is active at low level When channel 1 is configured as input, CC1P/CC1NP match use has defined the input signal polarity and level. Details are given in the ICx polarity/level selection table. Note: When LOCK level (TIMx_BDTR register LCCK) is set 3 or 2, this bit cannot be changed.
0	CC1E	Channel 1 input/capture output enable When channel 1 is configured as output: 0: Close. OC1 output disable 1: Enable. The output level relies on the values in MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE. CC1channel is configured as input: This bit determines whether the input capture function is enabled. 0：Capture disable 1：Capture enable

In input mode, ICx polarity/level selection is given in the Table below:

Table 10-4 ICx polarity/level selection

CCxP	CCxNP	ICx polarity/level
0	0	Rising edge active/high level active
1	0	Falling edge active/low level active
1	1	Rising edge or falling edge active/low level active
0	1	Reserved

10.4.10 TIMx_CNT Counter

Address offset: 0x24
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
rw															

Bit	Field	Description
15：0	CNT	Counter value

10.4.11 TIMx_PSC Prescaler

Address offset: 0x28
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
rw															

Bit	Field	Description
15：0	PSC	Prescaler value Counter clock frequency (ck_cnt) =f_{CK_PSC}/ （PSC+1） In case of update event, PSC value is loaded into the current prescaler register.

10.4.12 TIMx_ARR Auto Reload Register

Address offset: 0x2C
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR															
rw															

Bit	Field	Description
15：0	ARR	Auto reload value These bits define the auto reload value of the counter. When auto reload value is 0, the counter doesn't work.

10.4.13 TIMx_CCR1 Capture/Compare Register 1

Address offset: 0x34
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw															

Bit	Field	Description
15：0	CCR1	Channel 1 capture/compare value When channel 1 is configured as input: CCR1 value determines the counter value of the previous capture event (this register is only readable). When channel 1 is configured as output: If the preload function is not selected in TIMx_CCMR1 register (OC1PE bit), the write value is immediately transmitted to the current capture/compare shadow register. Or else, the write value is only loaded into the capture/compare register in case of update event. The current capture/compare shadow register is involved in the compare with the counter TIMx_CNT, and the compare result is reflected to the output signal of OC1 port.

10.4.14 TIMx_CCR2 Capture/Compare Register 2

Address offset: 0x38
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2															
rw															

Bit	Field	Description
15：0	CCR2	Channel 2 capture/compare value Refer to CCR1 description.

10.4.15 TIMx_CCR3 Capture/Compare Register 3

Address offset: 0x3C
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3															
rw															

Bit	Field	Description
15：0	CCR3	Channel 3 capture/compare value Refer to CCR1 description.

10.4.16 TIMx_CCR4 Capture/Compare Register 4

Address offset: 0x40
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
rw															

Bit	Field	Description
15：0	CCR4	Channel 4 capture/compare value Refer to CCR1 description.

10.4.17 TIMx_OR Input Option Register

Address offset: 0x50
Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ETR_RMP	
														rw	

Bit	Field	Description
15:2	Reserved	Reserved, must be kept at reset value.
1:0	ETR_RMP	ETR multiplex 00：Reserved 01：LSI clock input 10：Reserved 11：OSCIN_128 frequency division input

11. TIM14 Basic Timer

11.1 Overview

The basic timer TIM14 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 and PWM).
Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.
The basic timer TIM14 is completely independent, and does not share any resources.

11.2 Main characteristics

16-bit auto-reload counter

16-bit programmable prescaler allowing dividing (also “on the fly”) the counter clock frequency either by any factor between 1 and 65536.

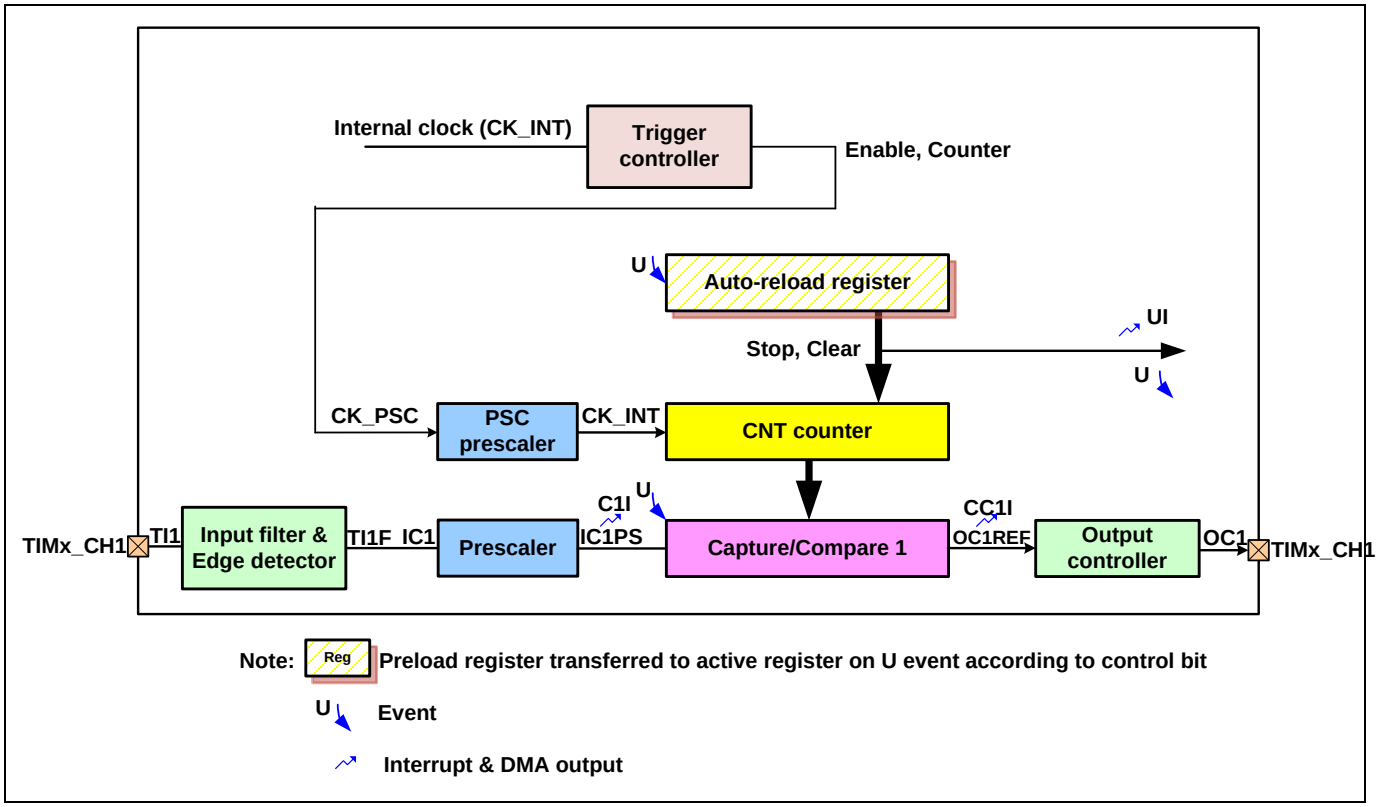
Independent channels for:

- 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 11-1 Basic timer block diagram



11.3 Function description

11.3.1 Time-base unit

The main block of the programmable basic timer is a 16-bit counter with its related autoreload register. The counter can count up. The counter clock can be divided by a prescaler.
The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running. The time-base unit includes:

Counter register (TIM14_CNT)

Prescaler register (TIM14_PSC)

Auto-reload register (TIM14_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the autoreload 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 TIM1_CR1 register. It can also be generated by software. The generation of the update event is described in detailed for each configuration.
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 (refer also to the slave mode controller description to get more details on counter enabling).
Note that the actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

11.3.1.1 Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536.
It is based on a 16-bit counter controlled through a 16-bit register (in the TIM14_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.
The following diagrams each give an example of changing the counter parameters while the prescaler is running.

Figure 11-2 Counter timing diagram with prescaler division change from 1 to 2

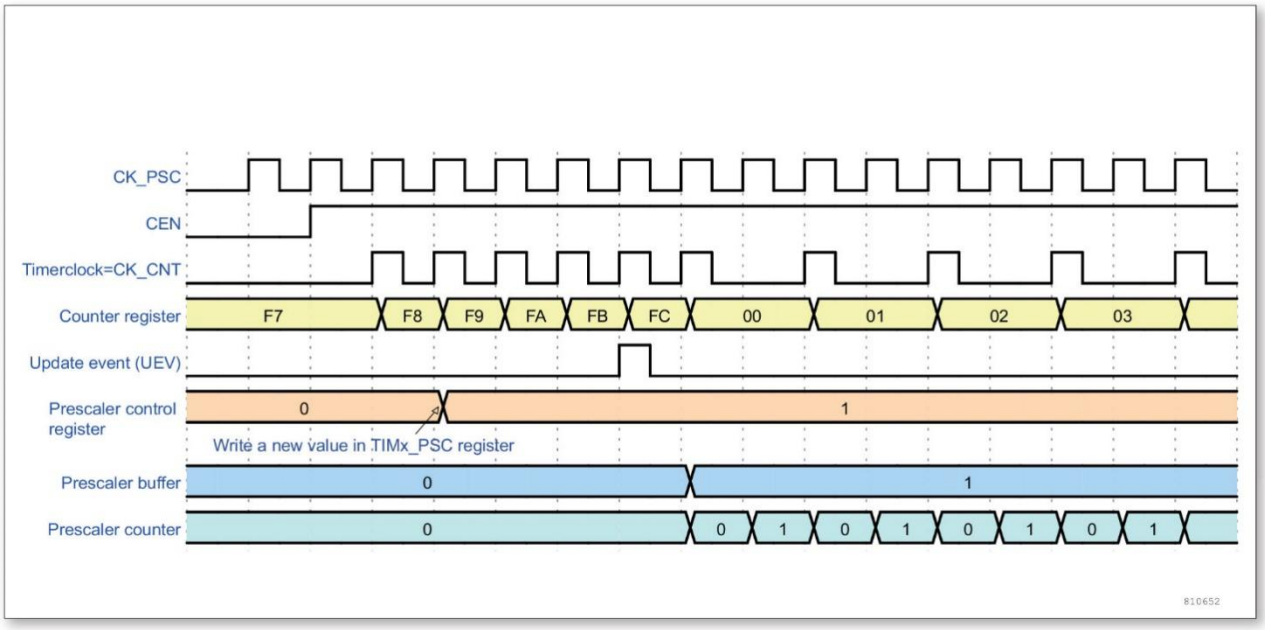
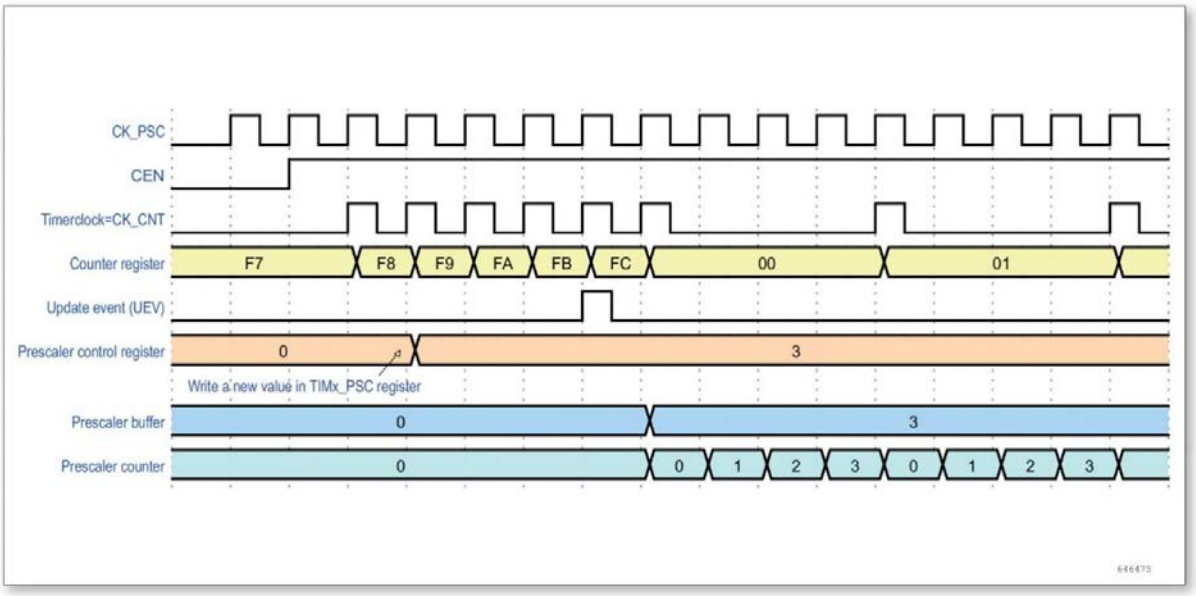


Figure 11-3 Counter timing diagram with prescaler division change from 1 to 4



11.3.2 Counting mode

11.3.2.1 Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIM14_ARR counter), then restarts from 0 and generates a counter overflow event.

An update event can be generate by setting the UG bit in the TIM14_EGR register.

The UEV event can be disabled by setting the UDIS bit in the TIM14_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers.

Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change) when an update event should be generated. 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 the UIF flag is not set by hardware (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter in the capture mode.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM14_SR register) is set by hardware (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 TIM14_ARR = 0x36:

Figure 11-4 Counter timing diagram, internal clock divided by 1

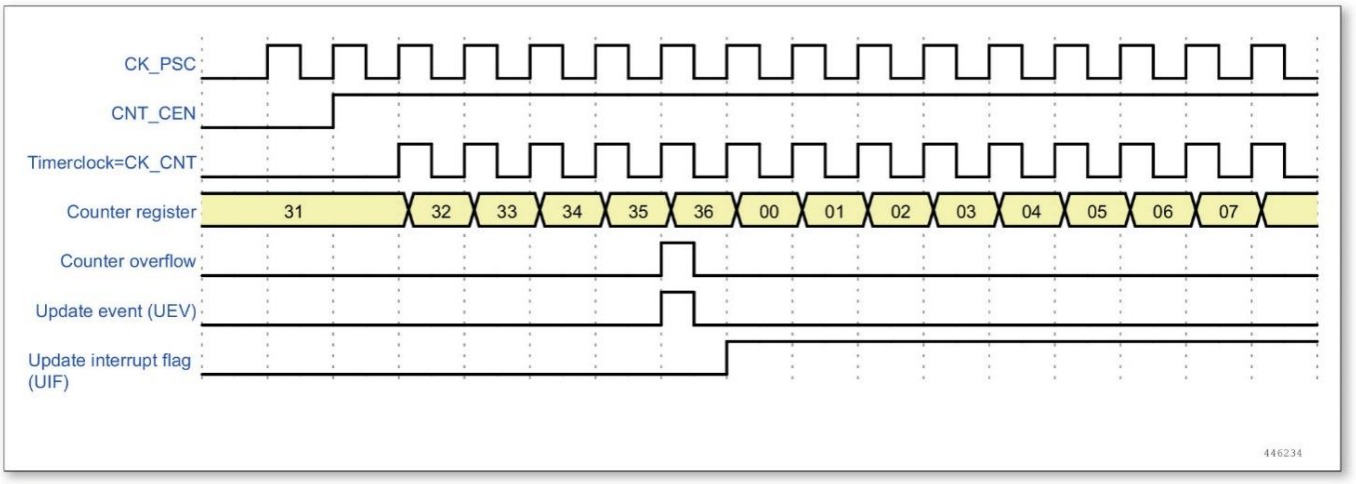


Figure 11-5 Counter timing diagram, internal clock divided by 2

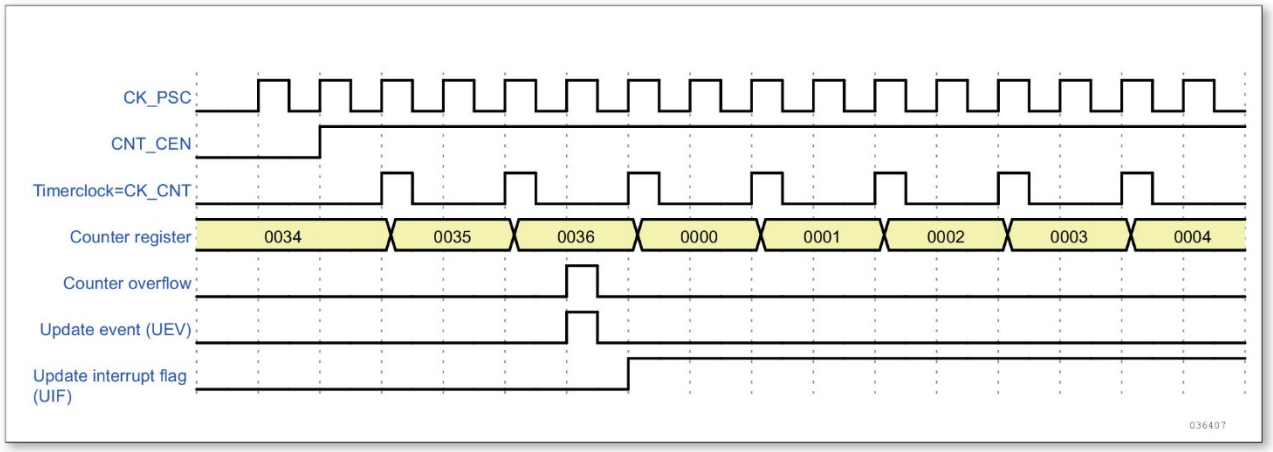


Figure 11-6 Counter timing diagram, internal clock divided by 4

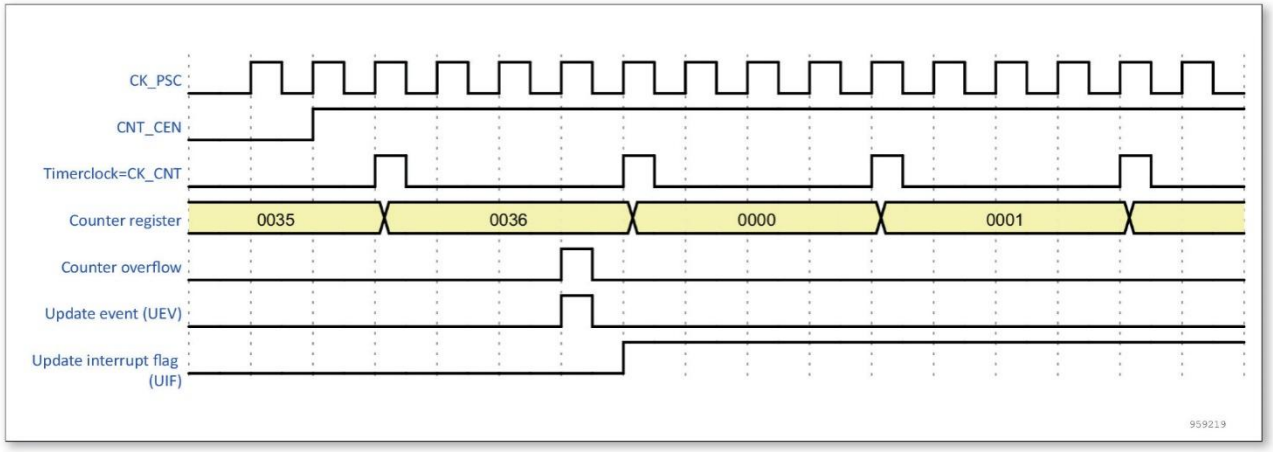


Figure 11-7 Counter timing diagram, internal clock divided by N

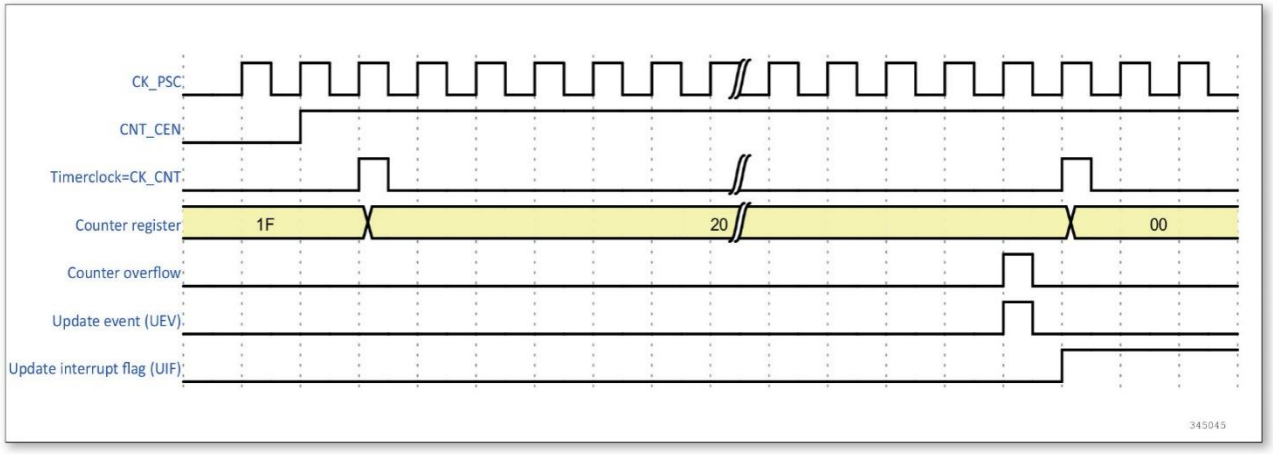


Figure 11-8 Counter timing diagram, update event when ARPE=0 (TIM14_ARR not preloaded)

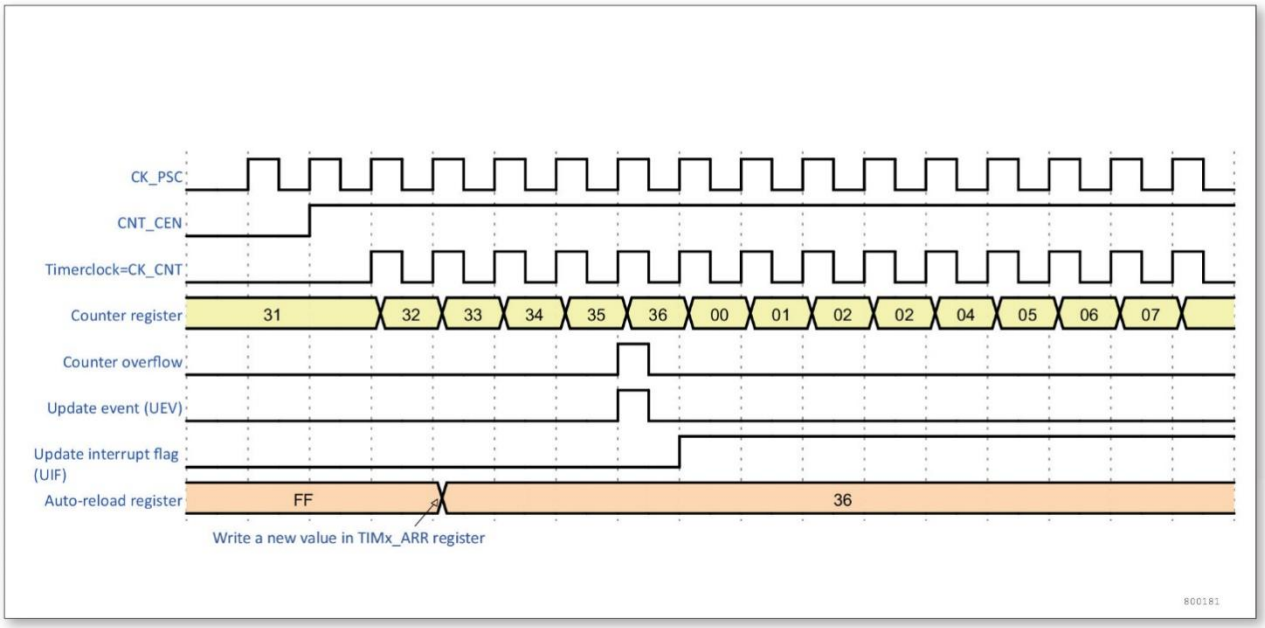
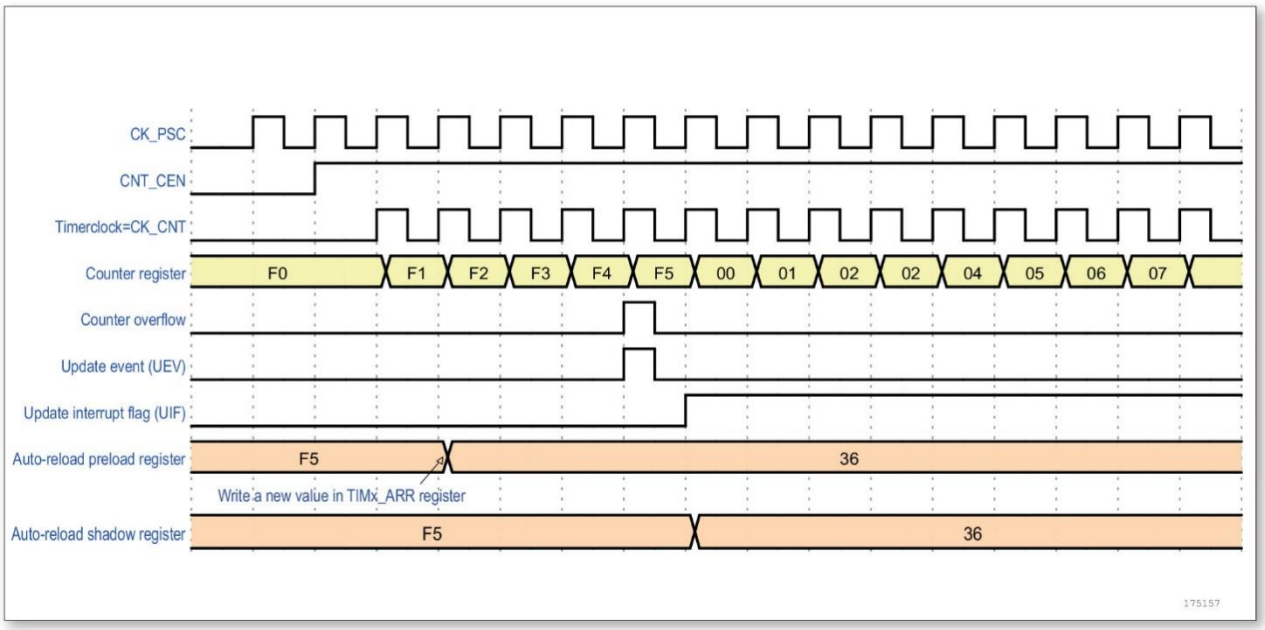


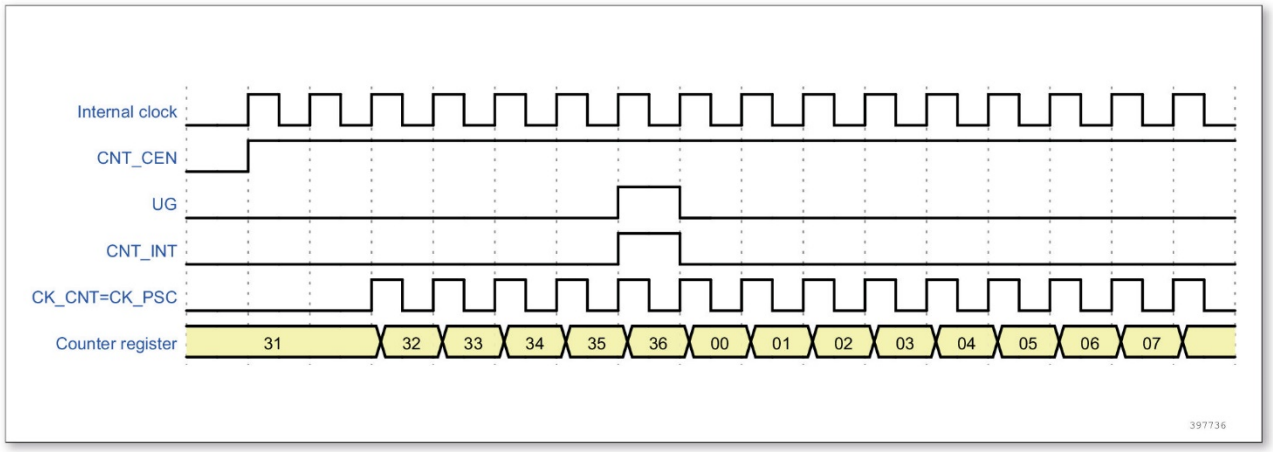
Figure 11-9 Counter timing diagram, update event when ARPE=1 (TIM14_ARR preloaded)



11.3.3 Clock source

The counter clock is provided by the internal clock (CK_INT) source.
The CEN (in the TIM14_CR1 register) and UG bits (in the TIM14_EGR register) are actual control bits and can be changed only by software (except for UG that remains cleared automatically). As soon as the CEN bit is written to ‘1’, the prescaler is clocked by the internal clock CK_INT.
The figure below shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

Figure 11-10 Control circuit in normal mode, internal clock divided by 1

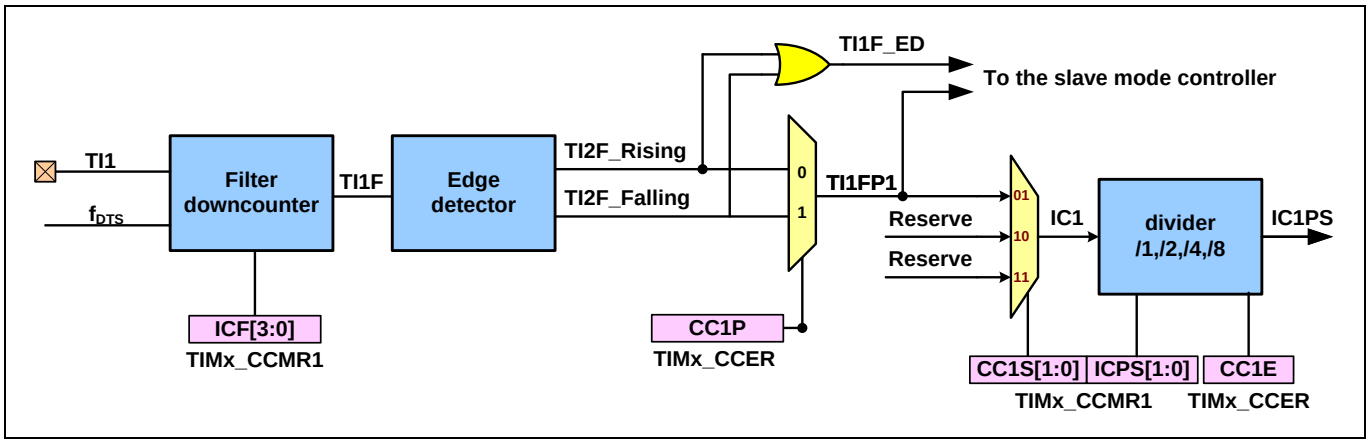


11.3.4 Capture/compare channel

Each Capture/Compare channel is built around a capture/compare register (including a shadow register), an input stage for capture (with digital filter, multiplexing and prescaler) and an output stage (with comparator and output control).

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

Figure 11-11 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 11-12 Capture/compare channel 1 main circuit

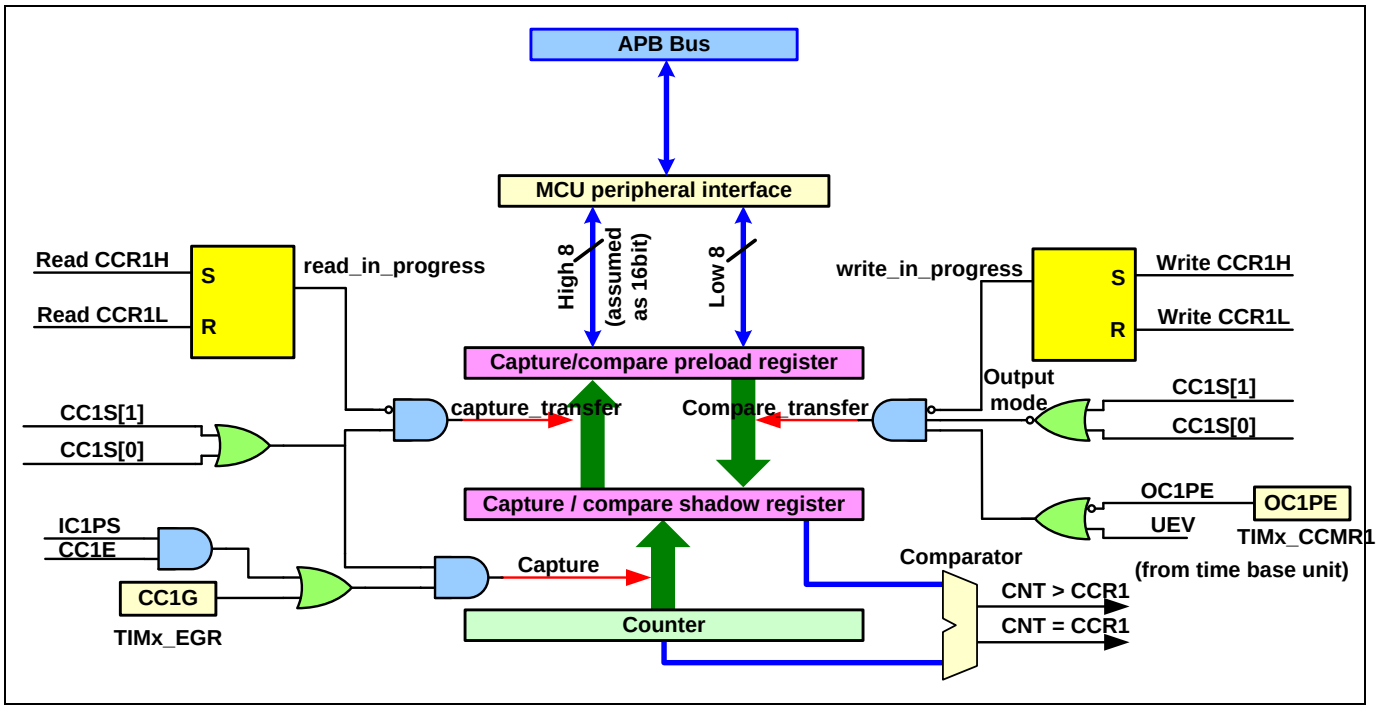
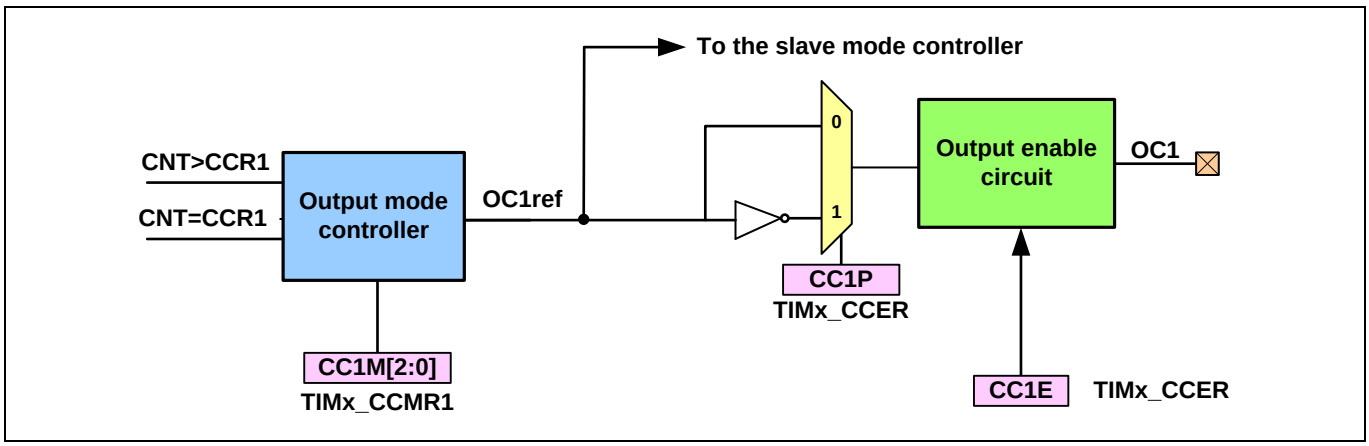


Figure 11-13 Output stage of capture/compare channel (channel 1)



The capture/compare block is made of one preload register and one shadow register.

Write and read always access the preload register. In capture mode, captures are actually done in the shadow register, which is copied into the preload register.

In compare mode, the content of the preload register is copied into the shadow register which is compared to the counter.

11.3.5 Input capture mode

In Input capture mode, the Capture/Compare registers (TIM14_CCRx) are used to latch the value of the counter after an edge is detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIM14_SR register) is set and an interrupt can be sent if it is enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIM14_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIM14_CCRx register. CCxOF is cleared when written to '0'.

The following example shows how to capture the counter value in TIM14_CCR1 when TI1 input rises. To do this, use the following procedure:

Select the active input: TIM14_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM14_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM14_CCR1 register becomes read-only.

Program the needed input filter duration with respect to the input signal (by programming ICxF bits in the TIM14_CCMRx register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration

longer than these five clock cycles. We can validate an edge transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIM14_CCMR1 register.

Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 and CC1NP to 0 in the TIM14_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 TIM14_CCMR1 register).

Enable capture from the counter into the capture register by setting the CC1E bit to '1' in the TIM14_CCER register.

If needed, enable the related interrupt request by setting the CC1IE bit in the TIM14_DIER register.

When an input capture occurs:

The TIM14_CCR1 register gets the value of the counter on the active level transition. CC1IF flag is set (interrupt flag). CC1OF is also set to '1' if at least two consecutive captures occurred whereas the CC1IF 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: Input capture interrupt requests can be generated by software by setting the corresponding CCxG bit in the TIM14_EGR register.

11.3.6 Forced output mode

In output mode (CCxS bits = 00 in the TIM14_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCxREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIM14_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx gets 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 bit to 100 in the TIM14_CCMRx register.

Anyway, the comparison between the TIM14_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly.

This is described in the output compare mode section below.

11.3.7 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

Assign the corresponding output pin to a programmable value defined by the output compare mode (OCxM bit in the TIM14_CCMRx register) and the output polarity (CCxP bit in the TIM14_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.

Set a flag in the interrupt status register (CCxIF bit in the TIM14_SR register).

Generate an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM14_DIER register).

The TIM14_CCRx registers can be programd with or without preload registers using the OCxPE bit in the TIM14_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The synchronization resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

The output compare mode is configured as follows:

Select the counter clock (internal, external, and prescaler)

Write the desired data in the TIM14_ARR and TIM14_CCRx registers

Set the CCxIE bit if an interrupt request is to be generated

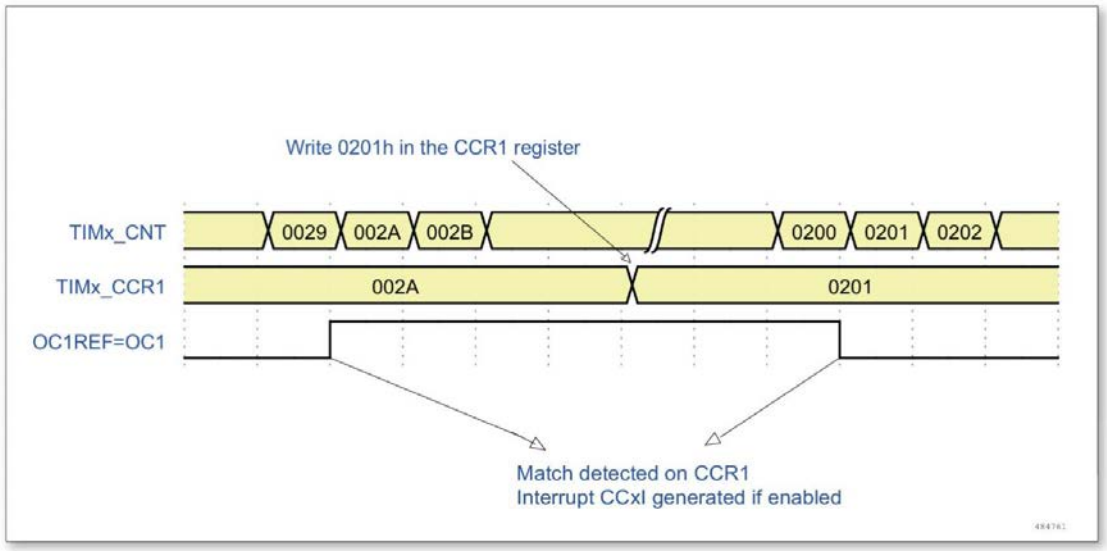
Select the output mode

- Write OCxM = 011 to toggle OCx output pin when CNT matches CCRx
- Write OCxPE = 0 to disable preload
- Write CCxP = 0 to select active high
- Write CCxE = 1 to enable output

Enable the counter by setting the CEN bit in the TIM14_CR1 register

The TIM14_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 TIM14_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

Figure 11-14 Output compare mode, toggle on OC1



11.3.8 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIM14_ARR register and a duty cycle determined by the value of the TIM14_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 bit in the TIM14_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register (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, the user must initialize all the registers 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 programd as active high or active low. OCx output is enabled by using the CCxE bit in the TIM14_CCER register. For details, refer to the TIM14_CCERx register description.

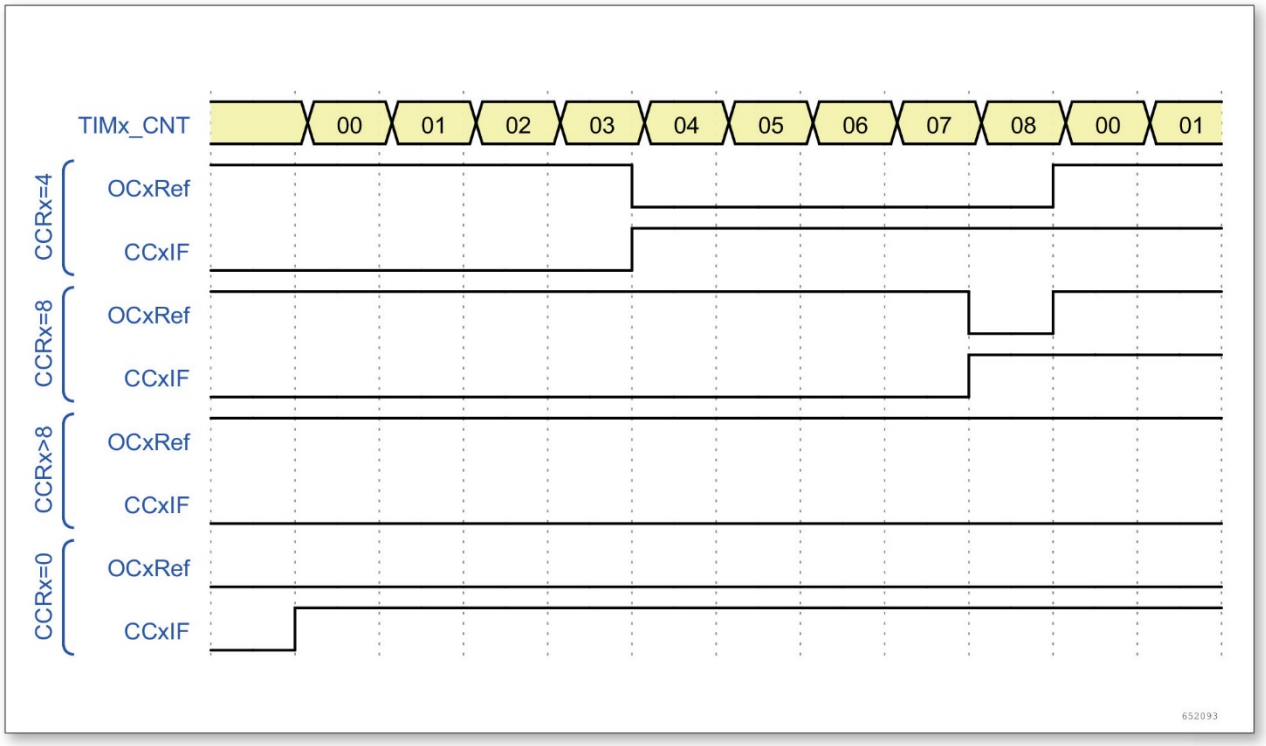
In PWM mode (1 or 2), TIM14_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT ≤ TIM14_CCRx.

The reason is that this counter is counting up and can only generate PWM in edge-aligned mode.

11.3.8.1 PWM edge-aligned mode

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIM14_CNT < TIM14_CCRx; Else it becomes low. If the compare value in TIM14_CCRx is greater than the auto-reload value (in TIM14_ARR), then OCxREF is held at '1'. If the compare value is 0 then OCxREF is held at '0'. The following figure shows some edge-aligned PWM waveforms in an example where TIM14_ARR = 8.

Figure 11-15 Edge-aligned PWM waveforms (ARR = 8)



11.3.9 Debug mode

When the microcontroller enters debug mode (CPU core halted), the TIM14 counter either continues to work normally or stops, depending on DBG_TIM14_STOP configuration bit in DBG module.

11.4 Register

Table 11-1 TIMx register overview

Offset	Acronym	Register Name	Reset
0x00	TIMx_CR1	Control Register 1	0x0000
0x0C	TIMx_DIER	Interrupt Enable Register	0x0000
0x10	TIMx_SR	Status Register	0x0000
0x14	TIMx_EGR	Event Generation Register	0x0000
0x18	TIMx_CCMR1	Capture/Compare Mode Register 1	0x0000
0x20	TIMx_CCER	Capture/Compare Enable Register	0x0000
0x24	TIMx_CNT	Counter	0x0000
0x28	TIMx_PSC	Prescaler	0x0000
0x2C	TIMx_ARR	Auto Reload Register	0x0000
0x34	TIMx_CCR1	Capture/Compare Register 1	0x0000
0x44	TIMx_BDTR	Break And Dead-Time Register	0x0000

11.4.1 TIMx_CR1 Control Register 1

Address offset：0x00

Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD		ARPE	Reserved				URS	UDIS	CEN
						rw		rw					rw	rw	rw

Bit	Field	Description
15：10	Reserved	Reserved, must be kept at reset value.
9：8	CKD	Clock division 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 (TI1). 00：$t_{DTS} = t_{INT\_CK}$ 01：t_{DTS} =2x t_{INT_CK} 10：t_{DTS} =4x t_{INT_CK} 11：Reserved, do not program this value
7	ARPE	Auto reload preload enable 0: Disable the shadow register of TIMx_ARR register 1: Enable the shadow register of TIMx_ARR register
6：3	Reserved	Reserved, must be kept at reset value.
2	URS	Update request source Software configures this bit, select update event source. 0: The event below may generate an update interrupt request: - Counter overflow - Set UG bit 1: Only in counter overflow, generate update interrupt request
1	UDIS	Update disable This bit is used to enable or disable the update event 0: Update event (UEV) enabled. 1: Update event disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCR1). However the counter and the prescaler are reinitialized if the EGR.UG bit is set.
0	CEN	Counter enable 0: Counter disabled 1: Counter enabled

11.4.2 TIMx_DIER Interrupt Enable Register

Address offset：0x0C
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CC1IE	UIE
														rw	rw

Bit	Field	Description
15：2	Reserved	Reserved, must be kept at reset value.
1	CC1IE	Enable capture/compare 1 interrupt 0: Capture/compare interrupt 1 disable 1: Capture/compare interrupt 1 enable
0	UIE	Enable update interrupt 0: Update event interrupt disable 1: Update event interrupt enable

11.4.3 TIMx_SR Status Register

Address offset：0x10
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CC1OF	Reserved							CC1IF	UIF
						r_w0c								r_w0c	r_w0c

Bit	Field	Description
15：10	Reserved	Reserved, must be kept at reset value.
9	CC1OF	Capture/compare 1 over capture flag Only when the channel 1 is configured as input capture mode and CC1F is configured as 1, and the capture event occurs again. The flag is set by hardware. Write 0 may clear the bit. 0: No overcapture 1: Overcapture
8：2	Reserved	Reserved, must be kept at reset value.
1	CC1IF	Capture/compare 1 interrupt flag Channel 1 in output mode: When the counter value and compare value match, the bit is set by hardware, It's cleared by software. 0: No match 1: TIMx_CNT value and TIMx_CCR1 value match Channel 1 as output mode: In case of capture event, the bit is set by hardware. It's cleared by software or reading TIMx_CCR1 register 0: No input capture occurred 1: Counter value is captured to TIMx_CCR1
0	UIF	Update interrupt flag In case of update event, the bit is set 1 by hardware. It's cleared 0 by software. 0: No update interrupt occurred 1: update interrupt pending TIM1_EGR register UG =1 or counter overflow generate update event.

11.4.4 TIMx_EGR Event Generation Register

Address offset：0x14
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CC1G	UG
														w	w

Bit	Field	Description
15：2	Reserved	Reserved, must be kept at reset value.
1	CC1G	Capture/compare 1 generation This bit is set by software. It generates a capture/compare event, and is auto cleared by hardware. 0: No action 1：Generate a capture/compare event in channel CC1: When the channel CC1 is configured as output: Set CC1IF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. When the channel CC1 is configured input: Set CC1IF =1. When enabling the corresponding interrupt, generate the corresponding interrupt. When CC1IF is set, set CC1OF =1.
0	UG	Update generation 0：No action 1: Initialize the counter, and generate an update event. It's auto cleared by hardware.

11.4.5 TIMx_CCMR1 Capture/Compare Mode Register 1

Address offset：0x18
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Res.	OC1M			OC1PE	Res.	CC1S	
								IC1F				IC1PSC		CC1S	
								rw	rw			rw	rw	rw	

The channel may be used for input (capture mode) or output (compare mode). The channel direction is defined by the corresponding CC1S. The register's other bits functions differently in the input mode and output mode. OC1x describes the channel function in the output mode. IC1x describes the channel function in the input mode.
Output compare mode:

Bit	Field	Description
15：7	Reserved	Reserved, must be kept at reset value.
6：4	OC1M	Output compare mode The bit has defined the output reference signal OC1REF action. OC1REF has determined OC1 value. OC1REF is active at high level. The active level of OC1 depends on CC1P bit. 000：Freeze. TIMx_CCR1 and TIMx_CNT compare results has no effect on OC1REF. 001: Set as high when configuration. When TIMx_CNT value and TIMx_CCR1 value are same, enforce OC1REF as high level 010: Set as low when configuration. When TIMx_CNT value and TIMx_CCR1 value are same, enforce OC1REF as low level 011: Toggle when match. When TIMx_CCR1=TIMx_CNT, OC1REF toggle. 100: Enforce as low. Enforce OC1REF at low level 101: Enforce as high. Enforce OC1REF at high level 110：PWM mode 1. During up count, in case of TIMx_CNT<TIMx_CCR1, enforce OC1REF is at high level. Or else, it's at low level. 111：PWM mode 2. During up count, in case of TIMx_CNT<TIMx_CCR1, channel 1 enforce OC1REF is at low level. Or else, it's at high level. Note: In PWM mode 1 or PWM mode 2, only when the compare result changes or it changes over from the freeze mode to PWM mode in the output compare mode, OC1REF level may change.
3	OC1PE	Output compare preload enable 0: Disable TIMx_CCR1 register preload function. The value written into TIMx_CCR1 register becomes valid immediately

		1: Enable TIMx_CCR1 register preload function. Only read and write the preload register. TIMx_CCR1 preload value becomes valid during the update event Note: Only in the one-pulse mode (TIMx_CR1 register OPM= 1), it has no influence whether the preload register is set. Under other scenarios, it's required to set the preload register. Otherwise, the follow up action is not certain.
2	Reserved	Reserved, must be kept at reset value.
1：0	CC1S	Channel 1 Capture/Compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00：Channel 1 is configured as output 01：Channel 1 is configured as input 10：Reserved 11：Reserved

Input capture mode

Bit	Field	Description
15：8	Reserved	Reserved, must be kept at reset value.
7：4	IC1F	Channel 1 input capture filter The digital filter is composed of an event counter. It records an output jump after N input events. These bits define IC1 input signal sampling frequency and digital filter length. 0000：No filter, f_{DTS} sampling 0001：Sampling frequency $f_{sampling}=f_{INT\_CK}$，N=2 0010：Sampling frequency $f_{sampling}=f_{INT\_CK}$，N=4 0011：Sampling frequency $f_{sampling}=f_{INT\_CK}$，N=8 0100：Sampling frequency $f_{sampling}=f_{DTS}$ /2，N=6 0101：Sampling frequency $f_{sampling}=f_{DTS}$ /2，N=8 0110：Sampling frequency $f_{sampling}=f_{DTS}$ /4，N=6 0111：Sampling frequency $f_{sampling}=f_{DTS}$ /4，N=8 1000：Sampling frequency $f_{sampling}=f_{DTS}$ /8，N=6 1001：Sampling frequency $f_{sampling}=f_{DTS}$ /8，N=8 1010：Sampling frequency $f_{sampling}=f_{DTS}$ /16，N=5 1011：Sampling frequency $f_{sampling}=f_{DTS}$ /16，N=6 1100：Sampling frequency $f_{sampling}=f_{DTS}$ /16，N=8 1101：Sampling frequency $f_{sampling}=f_{DTS}$ /32，N=5 1110：Sampling frequency $f_{sampling}=f_{DTS}$ /32，N=6 1111：Sampling frequency $f_{sampling}=f_{DTS}$ /32，N=8
3：2	IC1PSC	Channel 1 input capture prescaler This bit defines the ratio of the prescaler acting on IC1.The prescaler is reset as soon as CC1E=0(TIMx_CCER register). 00: No prescaler, the capture input detects each edge to trigger one capture 01: Trigger one capture for every 2 events 10: Trigger one capture for every 4 events 11: Trigger one capture for every 8 events
1：0	CC1S	Channel 1 capture/Compare selection This bit defines the channel direction and input signal selection. Only when the channel closes, these bits can be written: 00: Channel 1 is configured as output 01: Channel 1 is configured as input 10: Reserved 11: Reserved

11.4.6 TIMx_CCER Capture/Compare Enable Register

Address offset：0x20

Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CC1NP	Res.	CC1P	CC1E
												rw		rw	rw

Bit	Field	Description
15：4	Reserved	Reserved, must be kept at reset value.
3	CC1NP	Channel 1 input capture 1 polarity This bit is not valid when Channel 1 is configured as an output. When channel 1 is configured as input, CC1P/CC1NP match use has defined the input signal polarity and level. Details are given in the IC1 polarity/level selection table.
2	Reserved	Reserved, must be kept at reset value.
1	CC1P	Channel 1 capture/Compare output polarity When Channel 1 is configured as an output, this bit defines the output signal polarity. 0: OC1 active at high level 1: OC1 active at low level When channel 1 is configured as input, CC1P/CC1NP match use has defined the input signal polarity and level. Details are given in the IC1 polarity/level selection table.
0	CC1E	Channel 1 capture/compare output enable When channel 1 is configured as an output. 0: off. OC1 output is disabled 1: on. OC1 signal is output to the corresponding output pin CC1 Channel configured as input. This bit determines whether the input capture function is enabled. 0: Capture disable 1: Capture enable

The polarity/level selection of IC1 in the input mode is shown in the following table:

Table 11-2 IC1 polarity/level selection table

CC1P	CC1NP	IC1 polarity/level
0	0	Rising edge active/high level active
1	0	Falling edge active/low level active
1	1	Rising or falling edge active/low level active
0	1	Reserved

11.4.7 TIMx_CNT Counter

Address offset：0x24
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT															
rw															

Bit	Field	Description
15：0	CNT	Count value

11.4.8 TIMx_PSC Prescaler

Address offset：0x28
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC															
rw															

Bit	Field	Description
15：0	PSC	Prescaler value Counter clock frequency（ck_cnt）=f_{CK_PSC}/（PSC+1） In case of an update event, PSC value is loaded into the current prescaler register.

11.4.9 TIMx_ARR Auto Reload Register

Address offset：0x2C
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR															
rw															

Bit	Field	Description
15：0	ARR	Auto-reload value These bits define the auto reload value of the counter. When auto reload value is 0, the counter doesn't work.

11.4.10 TIMx_CCR1 Capture/Compare Register 1

Address offset：0x34
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
rw															

Bit	Field	Description
15：0	CCR1	Channel 1 capture/compare value When channel 1 is configured as input: CCR1 value determines the counter value of the previous capture event (this register is only readable). When channel 1 is configured as output: If the preload function is not selected in TIMx_CCMR1 register (OC1PE bit), the write value is immediately transmitted to the current capture/compare shadow register. Or else, the write value is only loaded into the capture/compare register in case of update event. The current capture/compare shadow register is involved in the compare with the counter TIMx_CNT, and the compare result is reflected to the output signal of OC1 port.

11.4.11 TIMx_BDTR Break and Dead-Time Register

Address offset：0x44
Reset value：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	Reserved														
rw															

Bit	Field	Description
15	MOE	Main output enable 0：Disable OC1 output or enforce as the idle state 1: If the corresponding enable bit (CC1E of TIMx_CCER register) is set, the OC1 output is turned on
14：0	Reserved	Reserved, must be kept at reset value.

12. IWDG Independent watchdog

12.1 Introduction

The independent watchdog is designed to detect and resolve malfunctions due to software failure. Its principle can be briefly described as follows: a system reset signal is generated when the independent watchdog (IWDG) counter decrements to a given value, thus triggering a system reset and improving the overall safety level of the system.
The independent watchdog is best suited to applications which require the watchdog to run as a totally independent process outside the main application, but have lower timing accuracy constraints.
The independent watchdog is clocked by the internal low-speed clock (LSI) and thus stays active even if the main clock fails.

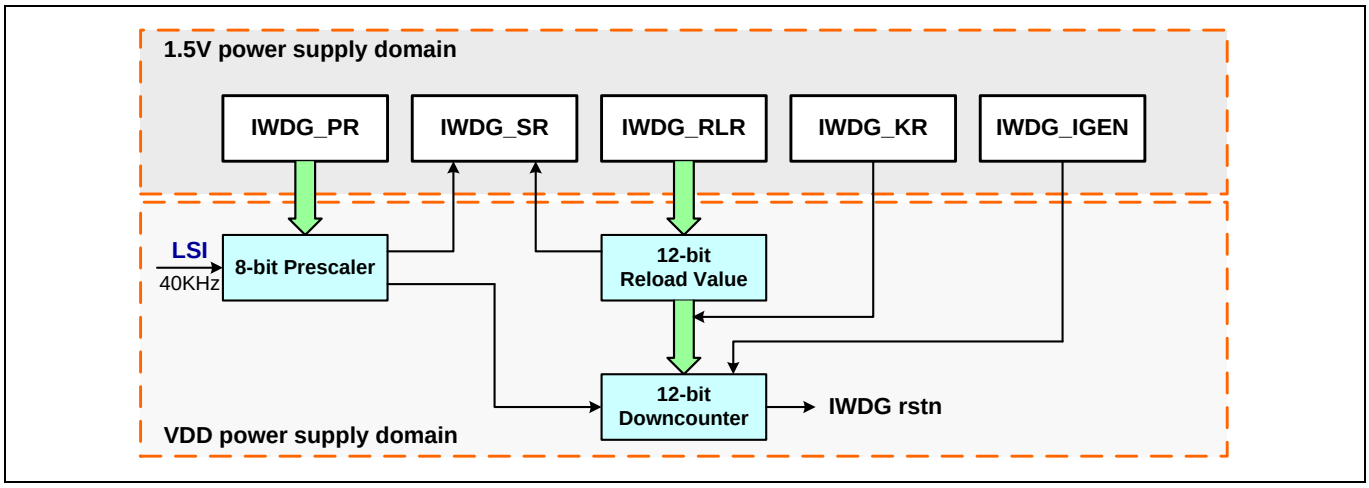
12.2 Main performance of IWDG

Free running down counter
The clock is provided by an independent oscillator (can operate in shutdown mode)
After the watchdog is activated, a reset is generated when the counter counts to 0x0000.

12.3 IWDG Function Description

Write 0xCCCC in the key register (IWDG_KR). Start activating the independent watchdog; At this point, the counter starts to count down from its reset value 0xFFF. When the counter reaches the end of 0x000, a reset signal (IWGD_RESET) will be generated. At any time, as long as the key register IWDG_ Write 0xAAAA and IWDG in KR_ The value in RLR will be reloaded into the counter to avoid a watchdog reset.
The following figure shows the functional block diagram of the independent watchdog module.

Figure 12-1 Independent Watchdog Block Diagram



Note: The watchdog function is in the VDD power supply area, which means it can still operate normally in shutdown mode.

Table 12-1 IWDG timeout (For example, the LSI clock frequency is 40 KHz)

Prescale Coefficient	PR[2:0] Bits	Min time (ms) RL[11:0]=0x000	Max time (ms)
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 or 7)	6.4	26214.4

Note: These times are given based on a 40 KHz clock. In fact, the oscillator frequency inside the MCU will vary between 30 KHz and 60 KHz. In addition, even if the frequency of the oscillator is precise, the exact timing still depends on the phase difference between the APB interface clock and the oscillator clock, so there will always be a complete oscillator cycle that is uncertain.

12.3.1 Hardware Watchdog

If the user activates the 'Hardware Watchdog' function in selecting bytes (please refer to the 'Embedded Flash' section), the watchdog will automatically start running after the system is powered on and reset; If the software does not write the corresponding value to the key register before the counter count ends, the system will generate a reset.

12.3.2 Register Access Protection

IWDG_ PR, IWDG_ RLR and IWDG_ The IGEN register has write protection function. To modify the values of these three registers, you must first report to IWDG_ Write 0x5555 to the KR register. Writing different values to this register will disrupt the order of operations and the register will be protected again. The reload operation (i.e. writing 0xAAAA) will also activate the write protection function. The status register indicates whether the pre division value and down counter are being updated.

12.3.3 Debug mode

When the microcontroller enters the Debug menu (CPU core stops), according to the DBG in the debugging module_ IWDG_ The status of the STOP configuration bit allows the IWDG counter to continue working or stop. Please refer to the chapter on debugging module for details.

12.4 Register

12.4.1 Overview of registers

Table 12-2 Overview of IWDG registers

Offset	Acronym	Register Name	Reset
0x00	IWDG_KR	Key register	0x00000000
0x04	IWDG_PR	Prescaler register	0x00000000
0x08	IWDG_RLR	Reload register	0x00000FFF
0x0C	IWDG_SR	Status register	0x00000000
0x10	IWDG_CR	Control register	0x00000000
0x14	IWDG_IGEN	Interrupt generate register	0x00000FFF
0x18	IWDG_CNT	Counter register	0x00000000

12.4.2 IWDG_KR Key Register

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**
KEY															
w															

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
15:0	KEY	Key value (write-only register) These bits must be written by software at regular intervals with 0xAAAA to feed the dog, otherwise a reset signal is generated to reset the system when the counter decrements to 0x0000. Writing 0x5555 by software may disable the protection and enable access to other configuration registers (IWDG_PR, IWDG_RLR, IWDG_CR (bit0), IWDG_IGEN). Writing 0xCCCC by software starts the watchdog.

12.4.3 IWDG_PR Prescaler Register

Address offset: 0x04
Reset value: 0x0000 0000

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

Bit	Field	Description
31:3	Reserved	always read as 0
2:0	PR	Prescaler divider These bits have write protection settings. Select the pre division factor of the counter clock by setting these bits. To change the pre division factor, IWDG_ The PVU bit of the SR register must be 0. 000: prescaler divider = 4 100: prescaler divider = 64 001: prescaler divider = 8 101: prescaler divider = 128 010: prescaler divider = 16 110: prescaler divider = 256 011: prescaler divider = 32 111: prescaler divider = 256 Note: Performing a read operation on this register will return the pre divided frequency value from the VDD voltage domain. If a write operation is in progress, the value read back may be invalid. Therefore, only for that IWDG_ The read value is only valid when the PVU bit of the SR register is 0.

12.4.4 IWDG_RLR Reload Register

Address offset: 0x08
Reset value: 0x0000 0FFF

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

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11:0	RL	Watchdog counter reload value These bits serve to define the reload value of the watchdog counter. This value will update to the counter each time the dog is fed (write 0xAAAA to IWDG_KR register). Subsequently, the watchdog counter counts down from this value. To modify the reload values, the protection should be disabled at first (write 0x5555 to IWDG_KR). After the reload values are updated, the RVU bit in the register becomes 0. At this time, the value read from this register is valid. The watchdog timeout period is a function of this reload value and the prescaler.

12.4.5 IWDG_SR Status Register

Address offset: 0x0C
Reset value: 0x0000 0000

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

Bit	Field	Description
31:4	Reserved	Reserved, must be kept at reset value
3	UPDATE	Watchdog reload value update flag

		The UPDATE is set when 0xAAAA is written into the IWDG_KR register. It is cleared automatically when the watchdog counter is updated and the reload value is written into the counter.
2	IVU	Watchdog Interrupt Generate value update This bit is set by hardware to indicate that an update of the interrupt generate value is ongoing. It is cleared by hardware when the interrupt generate value update operation is completed in the VDD voltage domain (takes up to 5 oscillator cycles at LSI). The interrupt generate value can be updated only when IVU bit is cleared.
1	RVU	Watchdog counter reload value update This bit is set when an update of the reload value is ongoing. It is cleared when the reload value update operation is completed (takes up to 5 oscillator cycles at LSI). The reload value can be updated only when RVU bit is cleared.
0	PVU	Watchdog prescaler value update This bit is set when an update of the prescaler value is ongoing. It is cleared when the prescaler value update operation is completed (takes up to 5 oscillator cycles at LSI). The prescaler value can be updated only when PVU bit is cleared.

Note: If multiple reload values, pre frequency values, or interrupt generation values are used in the application, the pre load value must be changed again after the RVU bit is cleared, the pre frequency value can be changed again after the PVU bit is cleared, and the interrupt generation value must be changed again after the IVU bit is cleared. However, after the pre division or reinstallation values are updated, there is no need to wait for the RVU or PVU to reset, and the following code can continue to be executed.

12.4.6 IWDG_CR Control Register

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Field	Description
31:2	Reserved	Reserved, must be kept at reset value
1	IRQ_CLR	IWDG interrupt clear 1: Interrupt cleared 0: No effect, interrupt flag still pending Note: It is not necessary to disable the KEY protection before writing to this bit.
0	IRQ_SEL	IWDG overflow operation select 1: interrupt generation enabled after overflow 0: reset generation enabled after overflow

12.4.7 IWDG_IGEN Interrupt Generate Register

Address offset: 0x14

Reset value: 0x0000 0FFF

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

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11:0	IGEN	IWDG Interrupt Generate value These bits are write protected. Used to define the threshold for the watchdog counter to generate an interrupt. Whenever the counter value decreases to the threshold, an interrupt is generated.

	Only when IWDG_ Only when the IVU bit in the SR register is 0 can modifications be made to this register. Note: Performing a read operation on this register will return the interrupt generated value from the VDD voltage domain. If a write operation is in progress, the return value may be invalid. Only when IWDG_ When the IVU bit of the SR register is 0, the return value must be valid.

12.4.8 IWDG_CNT Counter Register

Address offset: 0x18

Reset value: 0x0000 0001(reset by RCC and system)

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

Bit	Field	Description
31:19	Reserved	Reserved, must be kept at reset value
18:8	IWDG_CNT	IWDG counter value
7：0	IWDG_PS	Value of the prescaler counter of the IWDG clock

13. SPI Serial Peripheral Interface

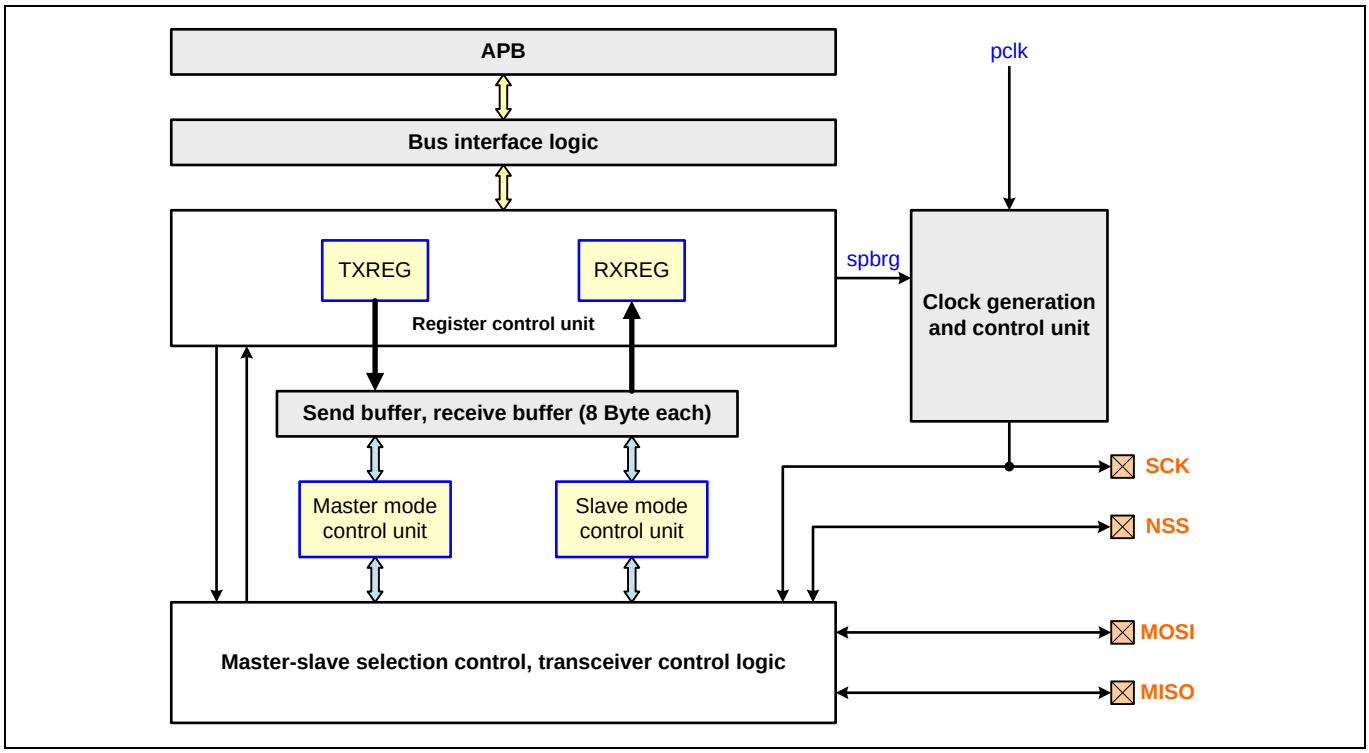
13.1 Overview

The SPI (Serial Peripheral Interface) interface is widely used on board level communication between different devices, such as extended serial Flash and ADC. Many IC manufacturers produce devices that support SPI interfaces.

SPI allows the MCU to communicate with external devices in full duplex, synchronous and serial mode. The application software can communicate by the query status or SPI interrupt.

13.2 Function block diagram

Figure 13-1 SPI function block diagram



13.3 SPI function description

13.3.1 Overview

SPI supports the simultaneous receipt and transmission of 1~32 bit data. SPI can be configured as the slave mode or the master mode under the host environment. The software configures the CPOL bit and the CPHA bit of the universal control register (CCTL) and select four possible transmission sequences among the clock and data. Configure LSBFE bit to select whether MSB is in front or LSB is in front for data transmission.

SPI transmits data in the rising edge or falling edge of SCK, and receives data in the effective edge of the opposite clock.

SPI is used to exchange data. Data must be read after the transmission, even if the data is not valid data. Besides, the clock phase bit and polarity of the host and other communication slave unit must be same.

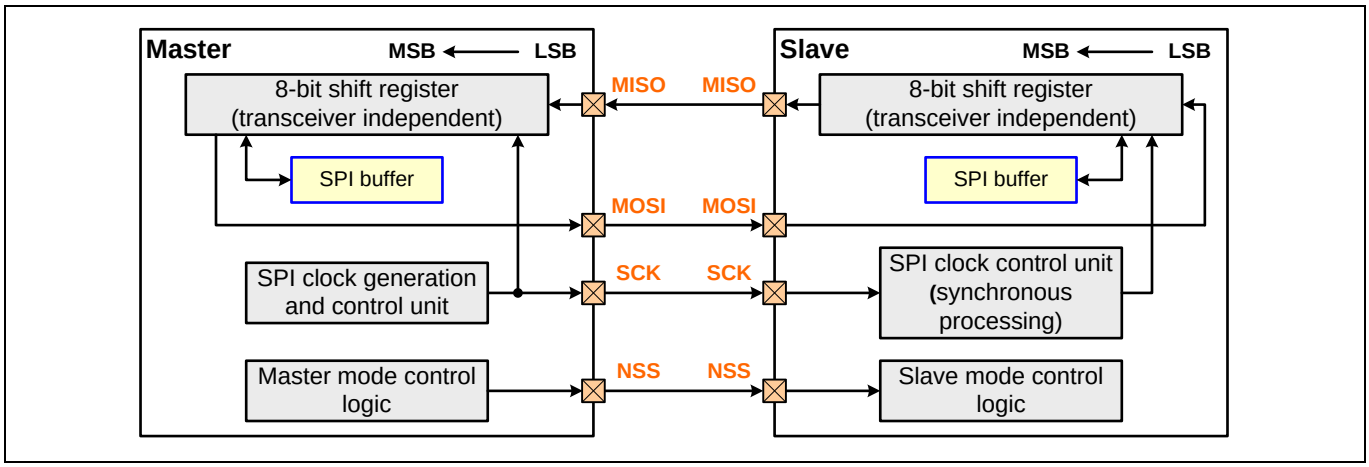
SPI is connected with the external device via 4 pins:

1. MISO: Master equipment input, slave device output pin. The transmission direction is from the slave equipment to the master equipment.
2. MOSI: Master equipment output, slave device input pin. The transmission direction is from the master equipment to the slave equipment
3. SCK: SCK: Serial port clock, generated by master equipment and transmitted to the slave equipment.
4. NSS: Slave equipment selection. This pin is the selectable function under SPI protocol for master equipment to select the slave equipment to communicate with it (when there are multiple slave equipment) to achieve

the one-to-one communication between the master and slave equipment and avoid the conflict between equipment on the data line. When the NSS pin function is activated, configure SPI_I2S_GCTL.MODE as '1' and SPI works in the master mode. Configure SPI_I2S_NSSR.NSS as '0' and NSS pin exports the low level, thereby enabling data communication between the slave equipment connected with the master equipment (configure SPI_I2S_GCTL.MODE as '0') and it.

The figure below shows one-to-one interconnected communication between the master equipment and slave equipment:

Figure 13-2 Single master and slave application



Same pins of SPI master and slave equipment are connected. The data transmission in the figure is serial communication from the highest valid bit to the lowest valid bit.

The master equipment is responsible for initiating the communication request and the slave equipment is responsible for response. The slave equipment gets the clock signal provided by master equipment via SCK pin, thereby enabling master and slave equipment to use the same clock for synchronous full duplex communication.

For the slave equipment, MOSI pin inputs the transmission data from the master equipment, and MISO pin exports the response data to the master equipment.

13.3.1.1 Phase bit and polarity of clock signal

CPOL and CPHA bit of SPI_CCTL register respectively control the clock polarity and phase bit of the clock. By combination, 4 sequence relations of clock/data can be made available.

Clock polarity refers to the level of SCK clock under the clock idle status: When configuring CPOL bit as '0', SCK clock keeps the low level under the idle status; otherwise, SCK clock keeps the high level under the idle status. Both master and slave equipment will be affected by CPOL control bit.

The clock phase bit determines the input data sampling sequence: When configuring CPHA bit as '0', the first data bit will be sampled in the second clock edge of SCK; otherwise, the first data bit will be sampled in the first clock edge of SCK.

Besides, CPHASEL bit is '0' after system power on and reset. When the software adjusts this bit as '1', change the data sampling sequence and CPHA bit function may change. For example during CPHASEL=1, CPHA=0: The first data bit will be sampled in the first clock edge of SCK (CPOL bit '0' rising edge; '1' falling edge).

Therefore, the expected use of clock/data sequence relations should be based on the configuration of CPOL, CPHA and CPHASEL bits.

Precautions for the time sequence configuration:

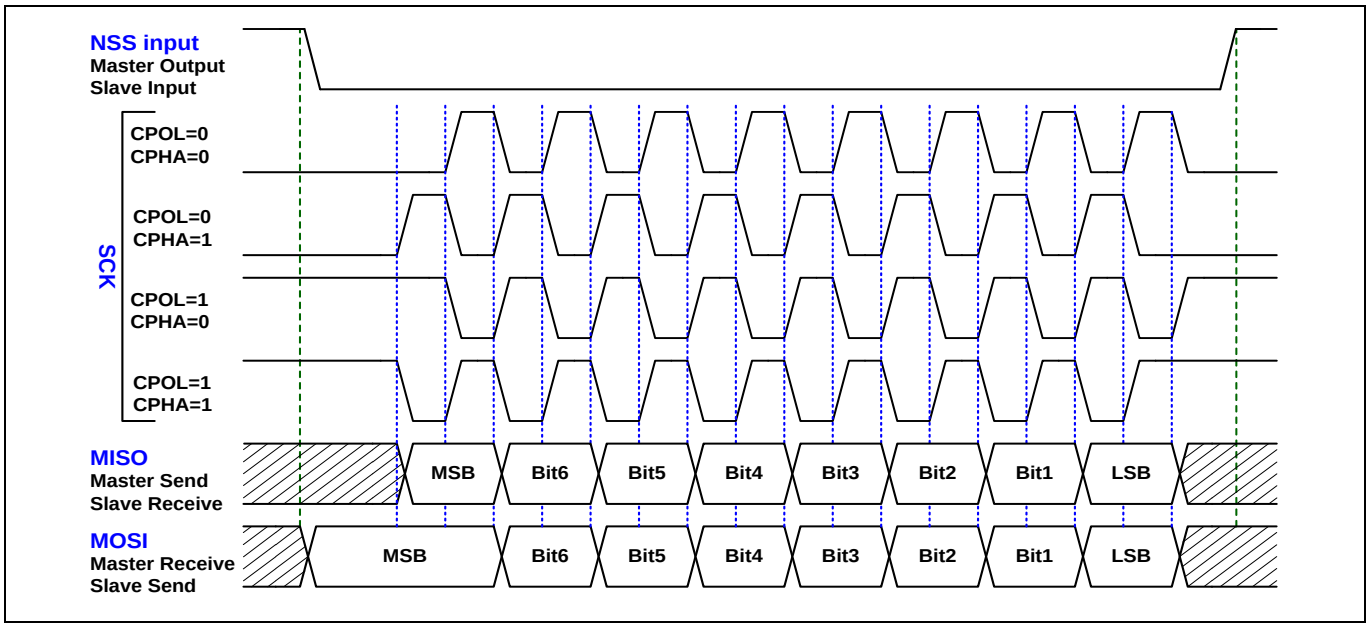
SPI cannot change CPOL/CPHA during work (When change is needed, first close the SPI enable bit SPIEN).

The master/slave equipment communicate in step. Therefore, the time sequence configuration should be consistent.

In the idle status, SCK level must be consistent with the corresponding polarity of CPOL configuration.

The figure below illustrates 4 scenarios of different CPHA and CPOL bit combinations during SPI transmission (configure CPHASEL bit as '1'), as well as the sequence of master/slave equipment SCK/MISO/MOSI/NSS pins.

Figure 13-3 Data clock sequence



It should be noted, configure SPI_GCTL.NSS=1, namely when NSS pin function is auto controlled by hardware. After data communication, the hardware will auto pull up this pin (as illustrated in the figure above). When configuring SPI_GCTL.NSS=0, NSS output status will be controlled by the NSS bit of register SPI_NSSR from the chip (changeover of NSS pin output status by software).

13.3.1.2 High speed transmission

With response to the board level sensitivity in the high speed transmission mode, configure SPI_CCTL register TXEDGE/RXEDGE bit to adjust the time point of data transmission/receipt sampling. During high speed transmission, the baud rate generator for the master equipment is configured at SPBRG<=4 and output SCK clock is quicker (>=10MHz); during low speed transmission, the baud rate generator for the master equipment is configured at SPBRG>4, and output SCK clock is slower (<10MHz).
In the slave mode, when TXEDGE bit is ‘1’, once the software configures TXREG to write data, rather than waiting for SCK clock input edge, make transmission to the MISO pin line; when TXEDGE bit is ‘0’, the slave equipment will always wait for a valid clock edge before transmitting data to MISO pin line.

13.3.1.3 Data frame format

Configure LSBFE bit of SPI_CCTL register, and determine the data bit output sequence, from the lowest valid bit to the highest valid bit when the LSBFE bit is ‘1’or from the highest valid bit to the lowest valid bit in case of ‘0’ (default).
Configure SPILEN bit of SPI_CCTL register, and determine the data frame length. When SPILEN bit is ‘1’ (default), the data frame length is 8 bit. In case of ‘0’, the data frame length is 7 bits. SPI transmission and receipt is controlled by the data frame format configuration.
Besides, configure SPI_GCTL.DW8_32=1 and register SPI_EXTCTL to achieve the data format of any frame length (frame length range within 1~32 bit); any frame length configuration also supports LSBFE bit function (LSB or MSB preferred).
When cooperating with DMA for data transmission, DMA data length is configured as 8bit.

13.3.2 Main characteristics of SPI

1 Fully compatible with Motorola SPI specification
2 Support full duplex sync transmission in 3 lines
3 Bit programmable baud rate generator
4 Support master and slave mode
5 Support communication between one host and multiple slave units
6 When the SPI is in the master/slave mode, the clock can be up to PCLK/2、PCLK/4 (PCLK as APB clock)
7 Programmable clock polarity and phase bit
8 Programmable data frame length (fixed 8 bit or 7 bit frame length, any frame length of 1~32 bits)

9 Programmable data sequence, MSB is in front or LSB is in front (support transmission and receipt of data with any frame length of 1~32 or fixed frame length)
10 4 byte receipt/ transmission buffer, the following interrupt event or status should be available for software configuration and use:
- Transmission buffer is empty
- Transmission buffer and transmission shift register are simultaneously empty
- Transmission end underflow
- Receive valid byte
- Receipt buffer overflow
- Receipt buffer full
- Receive specified byte in the master mode

13.3.3 SPI slave mode

When SPI serves as the slave equipment, SCK pin put comes from the serial clock of the master equipment. Therefore, when the baud rate generator is not used while the slave equipment works, it's not necessary to configure the register SPI_SPBRG (invalid under the slave equipment).

13.3.3.1 Configuration steps

1. Set SPI_GCTL.SPILEN to define the data frame format as 7 bit or 8 bit.
2. Configure the CPOL, CPHA/CPHASEL bit of the register SPI_CCTL to determine the sequence mode.
3. Configure SPI_CCTL.LSBFE, and determine the data frame receipt and transmission sequence (LSB or MSB bit preferred).
4. Configure MODE bit of register SPI_GCTL as '0' (slave mode), SPIEN bit as '1' (SPI function enable), and configure GPIO function pins required for SPI operation.
5. Configure TXEN and RXEN bit of register SPI_GCTL as '1', and open transmission and receipt permission (during transmission, it's required to write data into the register SPI_TXREG). SPI will receive MOSI pin data under the slave mode, and export data from MISO pin.

Note: It's mandatory to keep consistent the sequence mode and data frame receipt and transmission of master and slave equipment to guarantee normal transmission of data.

13.3.3.2 Data transmission

Write data to the transmission data register SPI_TXREG, and the whole data is transmitted to the transmission buffer.
When the slave equipment receives the SCK clock signal as well as the first data bit of MOSI pin. The slave equipment uses SCK change edge and transmit the data to the MISO pin bit by bit. The transmission data process conforms to the relevant sequence of data/clock (determined by the CPOL, CPHA/CPHASEL bit).
However, during high speed transmission (configure SPI_CCTL.TXEDGE=1), the data won't change according to the input SCK clock edge, but instead transmit data to MISO pin in advance along the internal PCLK clock edge (it won't be earlier than the SCK clock edge of the previous bit of data receipt sampling).
When the first bit of data is transmitted, the hardware will assert SPI_INTSTAT.TX_INTF flag, and the software will use this flag to write TXREG for continuous data transmission (configure SPI_INTEN.TX_IEN bit as '1' to produce CPU interrupt).
Note: The slave unit clock signal is provided by the host. Therefore, the precondition of continuous transmission must be that the host can provide the continuous clock.

13.3.3.3 Data receipt

When the slave equipment receives a complete data from the MOSI pin:
This data along the shift register will be transmitted to the receipt buffer in the final sampling clock edge. The hardware will simultaneously assert the SPI_INTSTAT.RX_INTF flag. And then the software will read the SPI_RXREG and acquire the data from the receipt buffer.
The software configures SPI_INTEN.RX_IEN bit as '1' to open the interrupt enable, and use CPU interrupt to acquire the receipt data.

13.3.4 SPI master mode

When SPI serves as the master equipment, export the serial clock to the SCK pin for use by the slave equipment.

13.3.4.1 Configuration steps

1. Set SPI_SPBREG to define the serial clock baud rate.
2. Configure SPI_CCTL register CPOL, CPHA/CPHASEL, and determine the time sequence mode.
3. Configure SPI_CCTL.SPILEN to define 8 or 7 bit data frame format. Configure SPI_GCTL.DW8_32 as '1', and configure SPI_EXTCTL register to define any frame format (SPILEN should be fixed as '1').
4. Configure SPI_CCTL.LSBFE to determine the data receipt and transmission sequence (LSB or MSB bit preferred).
5. In case of only receipt of data rather than transmission, configure SPI_RXDNR register to define the received byte (upon receipt of the specified number of byte, SCK clock output will end and keep CPOL bit configuration status).
6. Configure MODE bit of the register SPI_GCTL as '1' (master mode), and SPIEN bit as '1' (SPI function enable), and configure GPIO function pin required for the SPI work.
7. Configure TXEN and RXEN bit of register SPI_GCTL as '1', and open transmission and receipt permission (during transmission, it's required to write data into the register SPI_TXREG after opening TXEN). SPI will export clock SCK and sync data MOSI to the pin in the master mode, and sample input data from MISO pin. NSS is the optional output function of master equipment.

Note: It's mandatory to keep consistent the sequence mode and data frame receipt and transmission of master and slave equipment to guarantee normal transmission of data.

13.3.4.2 Data transmission

Configure TXEN bit as '1', and write data to the transmission data register TXREG. The master data will be transmitted to the transmission buffer, and the master equipment begins transmission. The master equipment serially exports SCK clock and MOSI data to the pin according to the preset baud rate, and this process conforms to the relevant sequence of data/clock (as determined by CPOL, CPHA/CPHASEL). Besides, LSBFE bit determines the data serial transmission sequence.

When the 1st bit data is transmitted, the hardware will assert the SPI_INTSTAT.TX_INTF flag. The software uses this flag to write TXREG and achieve continuous data transmission (configuring SPI_INTEN.TX_IEN bit as '1' to produce CPU interrupt).

13.3.4.3 Data receipt

When the slave equipment receives a complete data from the MOSI pin:

This data along the shift register will be transmitted to the receipt buffer in the final sampling clock edge. The hardware will simultaneously assert the SPI_INTSTAT.RX_INTF flag. And then the software will read the SPI_RXREG and acquire the data from the receipt buffer.

The software configures SPI_INTEN.RX_IEN bit as '1' to open the interrupt enable, and use CPU interrupt to acquire the receipt data.

In case of only receipt, and after receipt of the number of bytes as defined by RXDNR, the hardware will assert the SPI_INTSTAT.RXMATCH_INTF flag. Meanwhile, the master equipment won't transmit the clock signal, and SCK output will be kept in the CPOL bit configuration status (fixed high or low level).

13.3.5 Baud rate setting

SCK pin output clock frequency conforms to the baud rate configuration, which is obtained by the division of internal clock PCK according to the configuration value of SPI_SPBRG register. The register SPBREG controls the count period of one 16-bit counter.

According to the expected baud rate and Fpclk (APB module PCLK clock frequency), and use the formula of the table below to calculate the configuration value to the register SPBRG (X in the table below), X within the range of 2~65535.

Table 13-1 Baud rate formula

Mode	Formula
SPI mode	Baud rate = Fpclk/X

13.3.6 Interrupt

13.3.6.1 Status flag

To facilitate software operation, the application may use 4 current status flags and 7 interrupt flags to monitor SPI bus status.
The current status flag is only readable, and auto set and cleared by the hardware.
The interrupt status flag bit is set during event occurrence, and produces CPU interrupt during interrupt enable, cleared by software.
SPI has an 8 byte transmission buffer and receipt buffer. According to SPI_GCTL register DW8_32 bit setting, CPU may read and write 1 or 4 bytes each time. According to DW8_32 setting, the transmission and receipt buffer respectively have one byte or one valid data status flag.
Note: Configure SPI_GCTL.DW8_32=1, and receipt and transmission buffer have two valid data at most. When the frame length is configured as 8bit and below, one valid data is 1 byte. Its 2 bytes when the frame length is configured as a valid data within the range of 9~16bit. Its 3 bytes when the frame length is configured as a valid data within the range of 17~24bit. Its 4 bytes when the frame length is configured as a valid data within the range of 25~32bit.

Table 13-2 SPI status

Class	Status flag	Buffer and signal status
Interrupt status	TX_INTF	Transmission buffer is empty, according to DW8_32 setting to complete the write operation of transmission data register TXREG
	RX_INTF	According to DW8_32 setting, there is at least one valid data space, and complete the read operation of one receipt data register RXREG
	UNDERRUN_INTF	Transmission buffer is empty and repeat transmission
	RXOERR_INTF	Receipt buffer is not empty and covered
	RXMATCH_INTF	Non empty, the final one data is transmitted to the receipt buffer (valid in the master mode)
	RXFULL_INTF	Receipt buffer is full, and cannot receive new data
	TXEPT_INTF	Transmission buffer and shift register both are empty
Current status	RXAVL_4BYTE	Receipt buffer has the valid data over 4 bytes
	TXFULL	Transmission buffer is full
	TXEPT	Transmission buffer and shift register both are empty
	RXAVL	Receipt buffer is not empty

13.4 Register description

13.4.1 Overview of registers

Table 13-3 SPI register overview

Offset	Acronym	Register Name	Reset
0x00	SPI_TXREG	Transmission Data Register	0x00000000
0x04	SPI_RXREG	Receipt Data Register	0x00000000
0x08	SPI_CSTAT	Current Status Register	0x00000001
0x0C	SPI_INTSTAT	Interrupt Status Register	0x00000000
0x10	SPI_INTEN	Interrupt Enable Register	0x00000000
0x14	SPI_INTCLR	Interrupt Clear Register	0x00000000
0x18	SPI_GCTL	Global Control Register	0x00000004
0x1C	SPI_CCTL	Universal Control Register	0x00000008
0x20	SPI_SPBRG	Baud Rate Generator	0x00000002
0x24	SPI_RXDNR	Receipt Data Number Register	0x00000001
0x28	SPI_NSSR	Slave Chip Selection Register	0x000000FF
0x2C	SPI_EXTCTL	Data Length Control Register	0x00000008

13.4.2 SPI_TXREG Transmission Data Register

Address offset: 0x00
Reset value: 0x0000 0000

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

Bit	Field	Description
31：0	TXREG	Transmission data register Valid data bit is controlled by DW8_32: DW8_32=0, only valid in low 8 bit DW8_32=1, TXREG[31：0] valid

13.4.3 SPI_RXREG Receipt Data Register

Address offset: 0x04
Reset value: 0x0000 0000

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

Bit	Field	Description
31：0	RXREG	Receipt data register Valid data bit is controlled by DW8_32. DW8_32=0: Only valid in low 8 bit DW8_32=1: RXREG[31:0] valid Note: The register is readable, but not writable.

13.4.4 SPI_CSTAT Current Status Register

Address offset: 0x08
Reset value: 0x0000 2001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.					RXFADDR			Res.	TXFADDR			RXAVL_4BYTE	TXFULL	RXAVL	TXEPT
					r				r			r	r	r	r

Bit	Field	Description
31：11	Reserved	Always read as 0
10：8	RXFADDR	Number of valid byte in the current buffer
7	Reserved	Always read as 0
6：4	TXFADDR	Number of valid byte in the transmission buffer
3	RXAVL_4BYTE	Valid data in the receipt buffer reaches 4 byte flag bit 0: Data in receipt buffer is less than 4 byte 1: Data in receipt buffer exceeds 4 byte Note: In the I2S mode, after receipt of one audio data, the bit is set. (For example CHLEN =0, the bit is set after receipt of 16bit).

2	TXFULL	Transmission buffer full flag bit 0: Transmission buffer is not full 1: Transmission buffer is full
1	RXAVL	Receipt valid byte data message bit When the receipt buffer receives a complete byte data, the bit is set. 0: Receipt buffer is empty 1: Receipt buffer is not empty Note: This bit is only readable, auto set and cleared by the hardware.
0	TXEPT	Transmission empty bit 0: Transmission buffer or Shift register is not empty 1: Transmission buffer and Shift register both are empty Note: This bit is only readable, auto set and cleared by the hardware.

13.4.5 SPI_INTSTAT Interrupt Status Register

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Field	Description
31：7	Reserved	Reserved, always read as 0
6	TXEPT_INTF	Transmission empty interrupt flag bit Auto set by the hardware, clear by writing the INTCLR. TXEPT_ICLR bit as ‘1’. 0: Transmission buffer or Shift register is not empty 1: Transmission buffer and Shift register both are empty Note: This bit is the interrupt status signal while TXEPT is the status signal.
5	RXFULL_INTF	Receipt buffer full interrupt flag bit Auto set by the hardware, clear by writing INTCLR. RXFULL_ICLR bit as ‘1’. 0: RX buffer is not full 1: RX buffer is full
4	RXMATCH_INTF	Receipt specified byte interrupt flag bit Auto set by the hardware, clear by writing INTCLR.RXMATCH_ICLR bit as ‘1’. 0: Still has not received specified byte in RXDNR register 1: Receive specified byte in RXDNR register
3	RXOERR_INTF	Receipt overflow error interrupt flag bit Auto set by the hardware, clear by writing INTCLR. RXOERR_ICLR bit as ‘1’. 0: No overflow error 1: Overflow error
2	UNDERRUN_INTF	SPI auxiliary mode underflow interrupt flag bit Auto set by the hardware, clear by writing INTCLR.UNDERRUN_ICLR bit as ‘1’. 0: No underrun error 1: Underrun error
1	RX_INTF	Receipt data valid interrupt flag bit Auto set by the hardware, clear by writing INTCLR. RX_ICLR bit as ‘1’. When the receipt buffer receives a complete valid data. 0: Receipt buffer is empty 1: Receipt buffer has received a complete valid data Note：a complete valid data’s byte numbers, refer to 13.3.6.1 Status flags
0	TX_INTF	Transmission buffer empty interrupt flag bit (buffer is empty, can write TXREG) Auto set by the hardware, auto clear when the transmission buffer is not empty. 0: Transmission buffer is not empty 1: Transmission buffer is empty

13.4.6 SPI_INTEN Interrupt Enable Register

Address offset: 0x10
Reset value: 0x0000 0000

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

Bit	Field	Description
31：7	Reserved	Reserved, always read as 0
6	TXEPT_IEN	Transmission empty interrupt enable bit 0：Interrupt disable 1：Interrupt enable
5	RXFULL_IEN	Receipt buffer full interrupt enable bit 0：Interrupt disable 1：Interrupt enable
4	RXMATCH_IEN	Receipt specified byte interrupt enable bit 0：Interrupt disable 1：Interrupt enable
3	RXOERR_IEN	Receipt overflow error interrupt enable bit 0：Interrupt disable 1：Interrupt enable
2	UNDERRUN_IEN	SPI auxiliary mode underflow interrupt enable bit 0：Interrupt disable 1：Interrupt enable
1	RX_IEN	Receipt data interrupt enable bit 0：Interrupt disable 1：Interrupt enable
0	TX_IEN	Transmission buffer empty interrupt enable bit 0：Interrupt disable 1：Interrupt enable

13.4.7 SPI_INTCLR Interrupt Clear Register

Address offset: 0x14
Reset value: 0x0000 0000

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

Bit	Field	Description
31：7	Reserved	Reserved, always read as 0
6	TXEPT_ICLR	Transmission empty interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt
5	RXFULL_ICLR	Receipt buffer full interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt
4	RXMATCH_ ICLR	Receipt specified byte interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt
3	RXOERR_ ICLR	Receipt overflow error interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt

2	UNDERRUN_ ICLR	SPI auxiliary mode underflow interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt
1	RX_ICLR	Receipt interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt
0	TX_ICLR	Transmission buffer empty interrupt clear bit 0：Write 0 meaningless 1：Write 1 to clear interrupt

13.4.8 SPI_GCTL Global Control Register

Address offset: 0x18

Reset value: 0x0000 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Res.	DW8_32	NSS	DMA MODE	Reserved				RXEN	TXEN	MODE	INTEN	SPIEN
				rw	rw	rw					rw	rw	rw	rw	rw

Bit	Field	Description
31：13	Reserved	Reserved, always read as 0
12	Reserved	Reserved, must retain the reset value.
11	DW8_32	Transmission and receipt data register valid data selection 0:Only valid in low 8 bit 1:Valid in 32 bit data Note：Fixed as 1 in the i2S mode
10	NSS	NSS output in the hardware or software control master mode 0:Control by the NSSR register value 1:Auto control by the hardware during data transmission Note：Fixed as 0 in the i2S mode
9	DMAMODE	DMA mode selection bit 0：Normal mode 1：Enable DMA mode
8：5	Reserved	Reserved, must retain the reset value.
4	RXEN	Receipt enable bit 0: Receipt disable. Clear RX buffer at the same time 1: Receipt enable Note: When the SPI only works in the host receipt mode, TXEN must be set as 0.
3	TXEN	Transmission enable bit 0: Transmission disable. Clear TX buffer at the same time 1: Transmission enable Note: In the host mode, the transmission and receipt occur simultaneously
2	MODE	Host mode bit 0: Slave mode (serial clock from external host) 1: Host mode (serial clock produced by the internal BRG)
1	INTEN	SPI/I2S interrupt enable bit 0: Disable SPI/I2S interrupt 1: Enable SPI/I2S interrupt
0	SPIEN	SPI/I2S selection bit 0 : SPI/I2S disable (reset status) 1 : SPI/I2S enable

13.4.9 SPI_CCTL Universal Control Register

Address offset: 0x1C

Reset value: 0x0000 0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.									CPHASEL	TXEDGE	RXEDGE	SPILEN	LSBFE	CPOL	CPHA
									rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31：7	Reserved	Reserved, always read as 0
6	CPHASEL	CPHA polarity inversion selection 0: CPHA setting remains unchanged. 1: Invert CPHA setting value. In case of CPHA as 1, the first data bit sampling begins from the second clock edge. In case of CPHA as 0, the first data bit sampling begins from the first clock edge. Note：Fixed as 1 in the I2S mode
5	TXEDGE	Transmission data phase bit adjustment (slave mode) 0: Transmit data to the bus after one valid clock edge. To be used in the low speed mode (slave mode input clock slower). 1: Immediately transmit data to the data bus. To be used in the high speed mode (slave mode input clock quicker, exceeding 10MHz). Note: It's recommended to configure the bit as 1 to prevent incompliance with AC characteristics because of quicker communication speed, or the data transmission in the slave mode failing to conform to the master sampling sequence requirement.
4	RXEDGE	Receipt data sampling clock edge selection bit (master mode) 0: Sample data in the middle of transmission data bit. 1: Sample clock in the tail clock edge of the transmission data (for high speed mode, move back the sampling time point to respond to the board level wiring and slave mode transmission data latency). Note: It's recommended to configure the bit as 1 to prevent incompliance with AC characteristics because of quicker communication speed, or the incorrect data receipt in the master mode
3	SPILEN	SPI data width bit This bit works after DW8_32 setting (DW8_32 =0); this bit should be kept as 1 during DW8_32=1. 0: 7 bit data 1: 8 bit data (default) Note: Fixed as 1 in the I2S mode
2	LSBFE	LSBFE:LSB front enable bit 0: Data transmission or receipt highest bit in front 1: Data transmission or receipt lowest bit in front Note: Fixed as 0 in the I2S mode
1	CPOL	Clock polarity flag bit 0: Clock in low level in idle status (between two transmissions) 1: Clock in high level in idle status (between two transmissions)
0	CPHA	Clock phase selection bit 0: The first data bit sampling begins from the second clock edge 1: The first data bit sampling begins from the first clock edge Note: Fixed as 0 in the I2S mode

13.4.10 SPI_SPBRG Baud Rate Generator

Address offset: 0x20
Reset value: 0x0000 0002

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

Bit	Field	Description
31：16	Reserved	Reserved, always read as 0
15：0	SPBRG	SPI baud rate control register for baud rate Baud rate formula: Baud rate = Fpclk/SPBRG (Fpclk is APB clock frequency) Note: Don’t write 0 and 1 in the register.

13.4.11 SPI_RXDNR Receipt Data Number Register

Address offset: 0x24
Reset value: 0x0000 0001

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

Bit	Field	Description
31：16	Reserved	Reserved, always read as 0
15：0	RXDNR	The register is used to hold a count of to be received in the next receipt process Note: The register value is valid when SPI is in the host receipt mode. The default value is 1. The register value changes by MCU write value, and don’t write the ‘0’ value in the register.

13.4.12 SPI_NSSR Slave Chip Selection Register

Address offset: 0x28
Reset value: 0x0000 00FF

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.															NSS
															rw

Bit	Field	Description
31：1	Reserved	Reserved, must retain the reset value.
0	NSS	The chip selection output signal in the master mode. Low valid, the bit is invalid in the slave mode 0: Slave device is not selected (allow communication between the slave device and master mode) 1: Slave device is selected

13.4.13 SPI_EXTCTL Data Length Control Register

Address offset: 0x2C

Reset value: 0x0000 0008

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.											EXTLEN				
											rw				

Bit	Field	Description
31：5	Reserved	Reserved, always read as 0
4：0	EXTLEN	Control SPI data length 0 0000：32 bit 0 0001：1 bit 0 0010：2 bit 0 0011：3 bit …… 1 1100：28 bit 1 1101：29 bit 1 1110：30 bit 1 1111：31 bit Note: It's valid only when the SPI_GCTL.DW8_32 bit is '1'; in case of DW8_32=0, it's mandatory to keep the initial value 5'h8). Configuration invalid in the I2S mode: In the I2S mode, in case of CHLEN=1, EXTLEN value is fixed as 5'b00000 (32 bit); In the I2S mode, in case of CHLEN=0, EXTLEN value is fixed as 5'b10000 (16 bit)

14. I2C Inter-Integrated Circuit Interface

14.1 Introduction

I2C (inter-integrated circuit) bus interface provides support for serial interconnection between chips in the microcontroller. It provides multi-master capability to control all I2C bus-specific sequencing, protocol, arbitration and timing.

The I2C bus is a two-wire serial interface, where the serial data line (SDA) and the serial clock line (SCL) pass information between the devices connected to the bus. Each device is addressable by a unique address and can operate as a transmitter or receiver. Besides, the device can also be considered as a master or slave when it conducts data transfer. A master is a device that initiates the data transfer on the bus and generates clock signals that allow the transfer. At this point, any device addressed is considered as a slave.

I2C has two optional speed modes: Standard mode (data transfer rate up to 100Kbps) and Fast mode (data transfer rate up to 400Kbps).

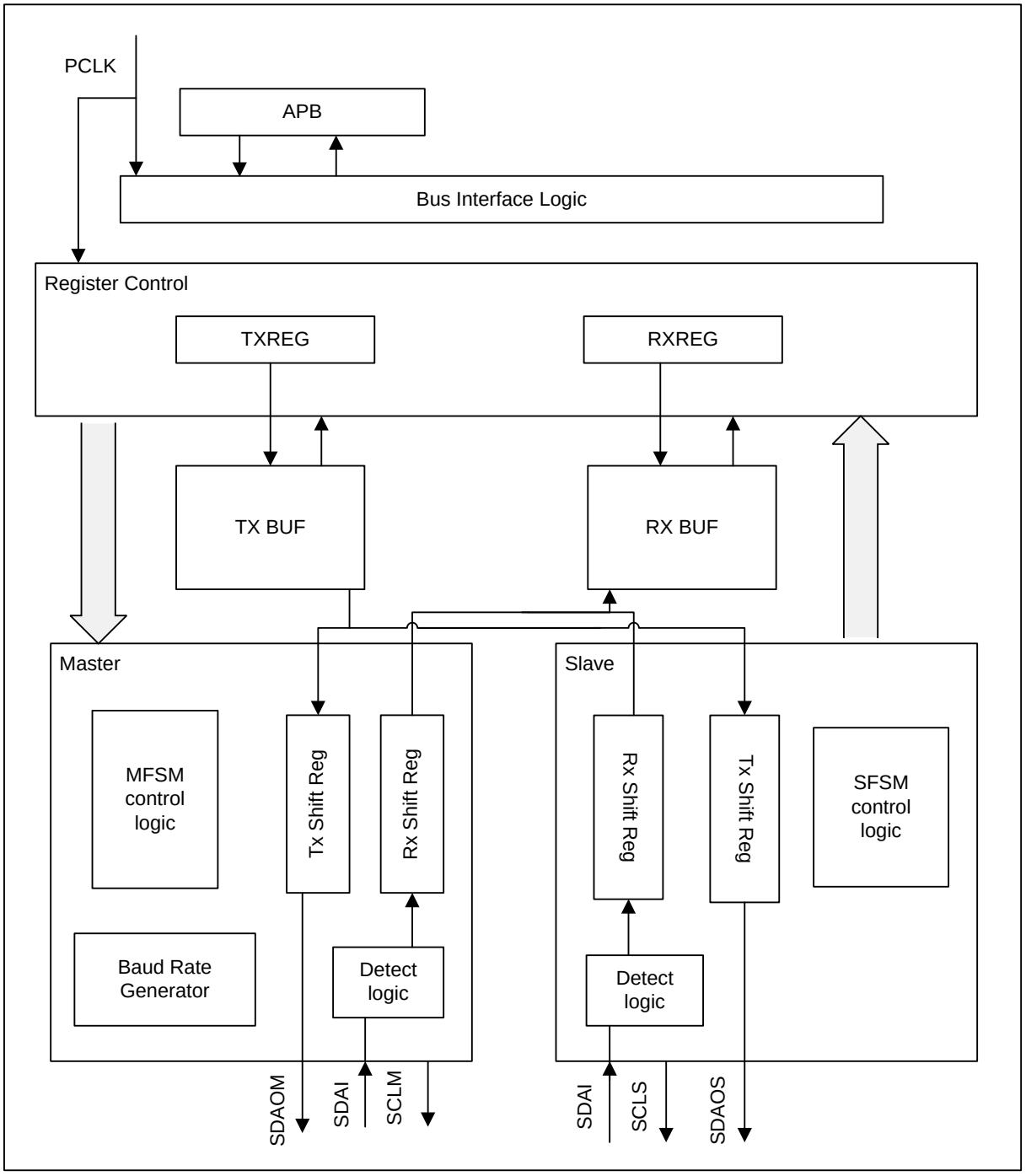
14.2 Main characteristics

1 I2C bus protocol converter/parallel-bus
2 Half-duplex synchronous operation
3 Master and Slave modes supported
4 7-bit/10-bit address format supported
5 START, STOP, RESTART, and ACK generation and detection supported
6 Standard mode (up to 100Kbps) and Fast mode (up to 400Kbps) supported
7 Respective 2-byte Tx and RX FIFOs
8 Spike-free circuit added to SCLI and SDAI
9 Interrupt or polled-mode operation supported
10 Multiple slave addresses supported (see the I2C_SLVMASK register description for details)

14.3 Functional description

14.3.1 Functional block diagram

Figure 14-1 I2C functional block diagram



14.3.2 Pin definitions

Table 14-1 Pin definitions

Pin Name	Attribute	Description
I2C_SCL	I/O	I2C clock
I2C_SDA	I/O	I2C data

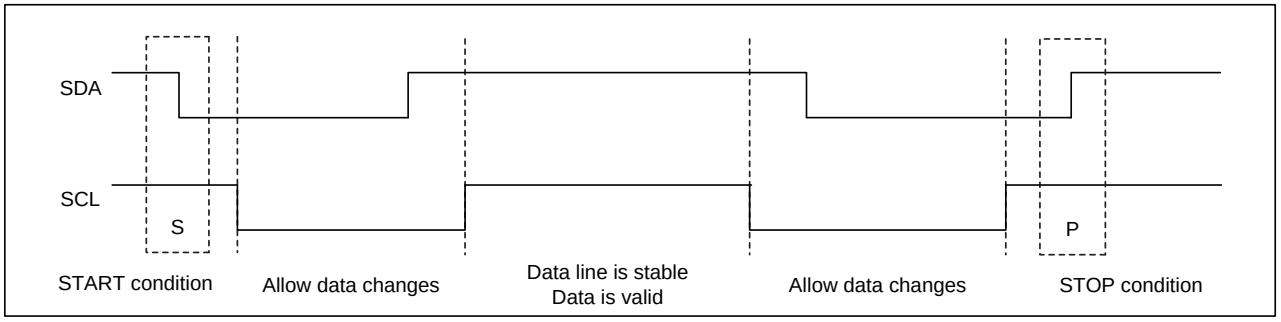
Note: When used, pins should be all in open-drain mode by configuration. Please refer to the GPIO section for configuration method.

14.3.3 I2C protocol

14.3.3.1 Start and STOP conditions

When the bus is idle, SCL and SDA are pulled up by external pull-up resistors. The data transfer initiated in the Master begins with a START condition. A high to low transition on the SDA line while SCL is high defines a START condition. The data transfer in the Master is terminated by the Master issuing a STOP condition. A low to high transition on the SDA line while SCL is high defines a STOP condition. A timing diagram of Start and STOP conditions is shown as below. SDA must maintain steady during the data transfer when SCL is 1.

Figure 14-2 Start and STOP conditions



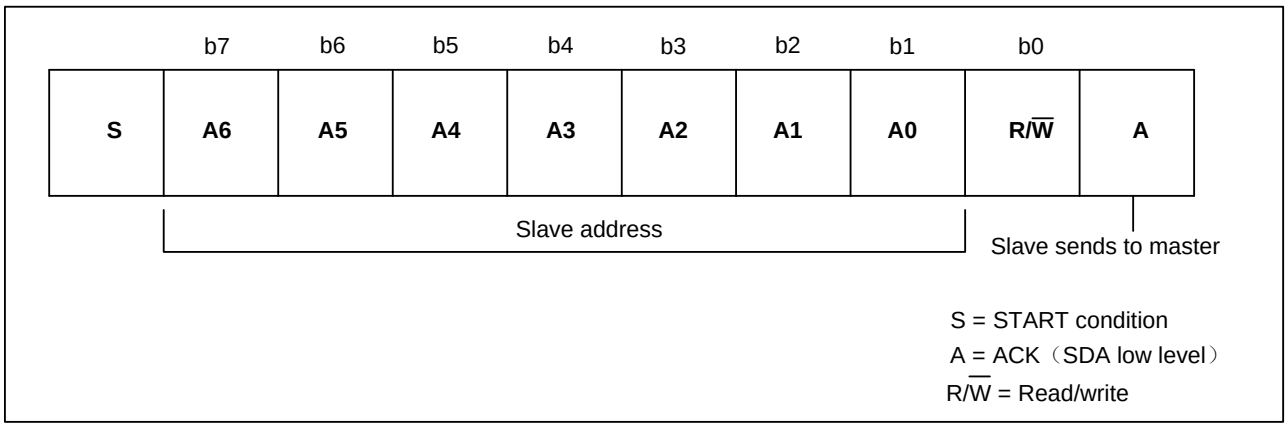
14.3.3.2 Addressing protocol

I2C has two address formats: 7-bit address format and 10-bit address format.

7-bit address format

The slave address is the first 7 bits (bit 7:1) of the first byte sent following a START condition (S). The least significant bit (bit 0) determines the data transfer direction. If bit 0 is 0, then the Master writes data to the Slave; if bit 0 is 1, then the Master reads data from the Slave.

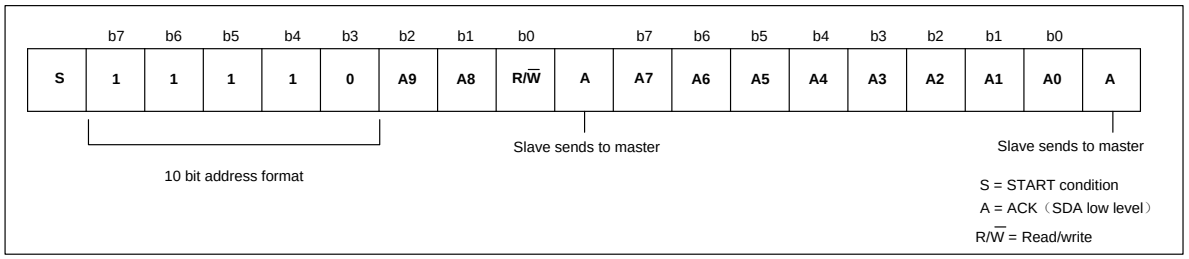
Figure 14-3 7-bit address format



10-bit address format

During 10-bit addressing, two bytes should be transferred to set the 10-bit address. The first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (bit 0) determines the data transfer direction (R/W). The second byte sets lower eight bits of the 10-bit address.
For details, see the diagram below:

Figure 14-4 10-bit address format



The following table defines the special purpose and reserved addresses of the first byte for I2C:

Table 14-2 First byte for I2C

Slave Address	R/W Bit	Description
0000 000	0	General Call address. I2C places the data in the receive buffer and issues a General Call interrupt.
0000 000	1	Start byte
0000 001	x	CBUS address. I2C ignores these accesses.
0000 010	x	Reserved
0000 011	x	Reserved
0000 1xx	x	Reserved
1111 1xx	x	Reserved
1111 0xx	x	10-bit slave addressing

14.3.3.3 Transmitting and receiving protocol

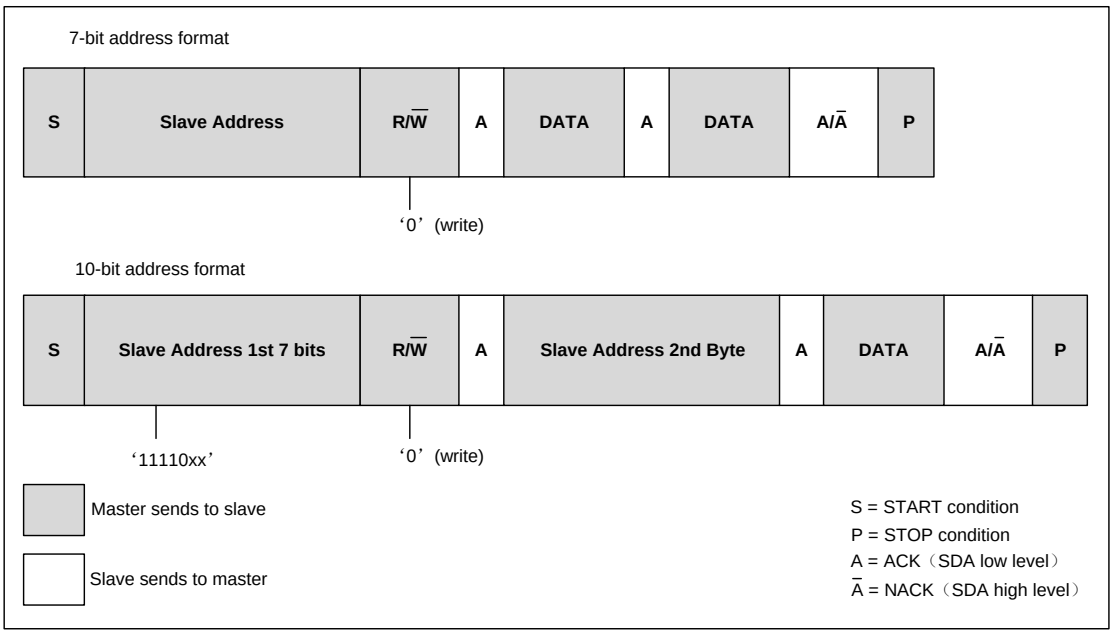
The master can initiate data transmission and reception to or from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master, acting as either a slave-transmitter or slave-receiver.

Master-transmitter and Slave-receiver

All data is transmitted in byte format, with no limit on the number of bytes per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. If a slave is not able to respond, the slave must leave the SDA line high so that the master can abort the transfer.

If the master-transmitter is transmitting data as shown in the following figure, then the slave-receiver responds to the master-transmitter with an ACK pulse after every byte of data is received.

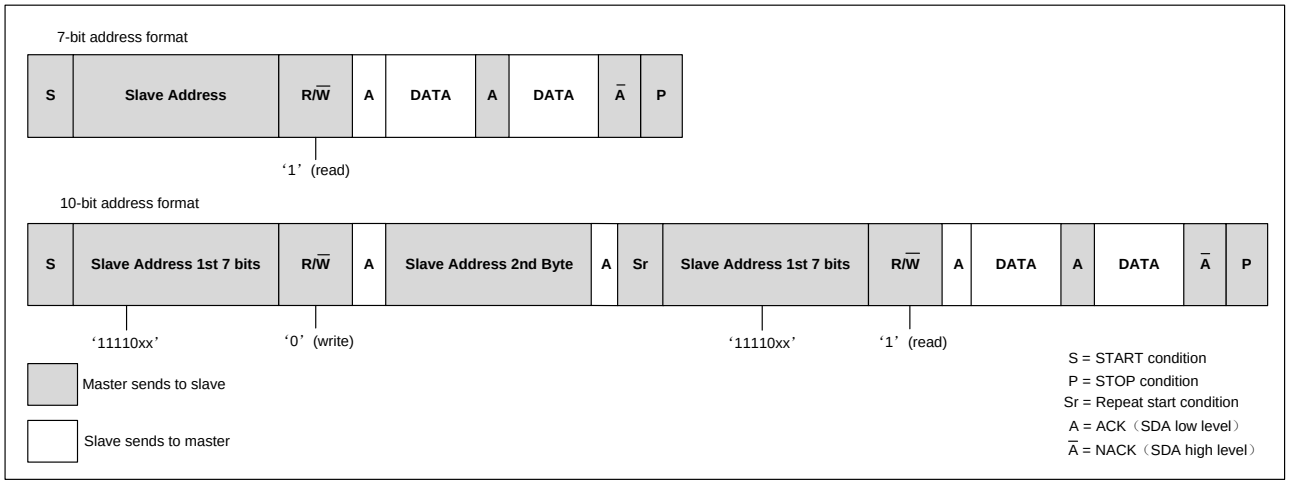
Figure 14-5 Master-transmitter protocol



Master-receiver and Slave-transmitter

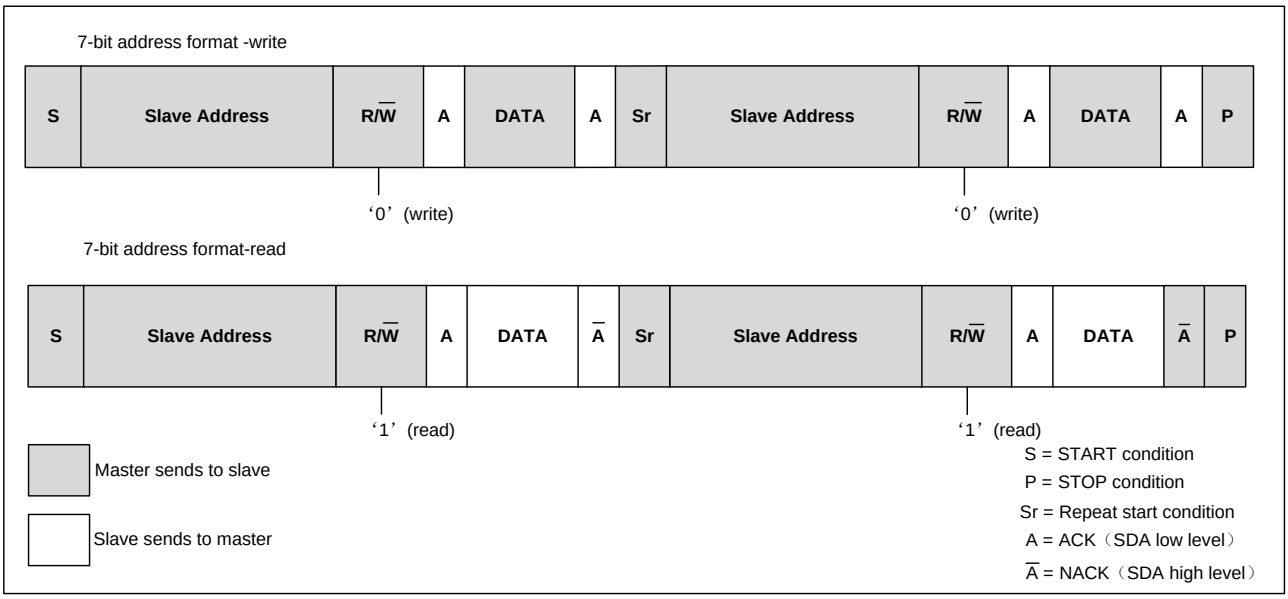
If the master is receiving data as shown in the following figure, then the master responds to the slave-transmitter with an ACK pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition.

Figure 14-6 Master-receiver protocol



When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK. Operating in Master mode, the I2C interface can then communicate with the same slave using a transfer of a different direction.

Figure 14-7 Master-transmitter and receiver protocol with a RESTART (SR)

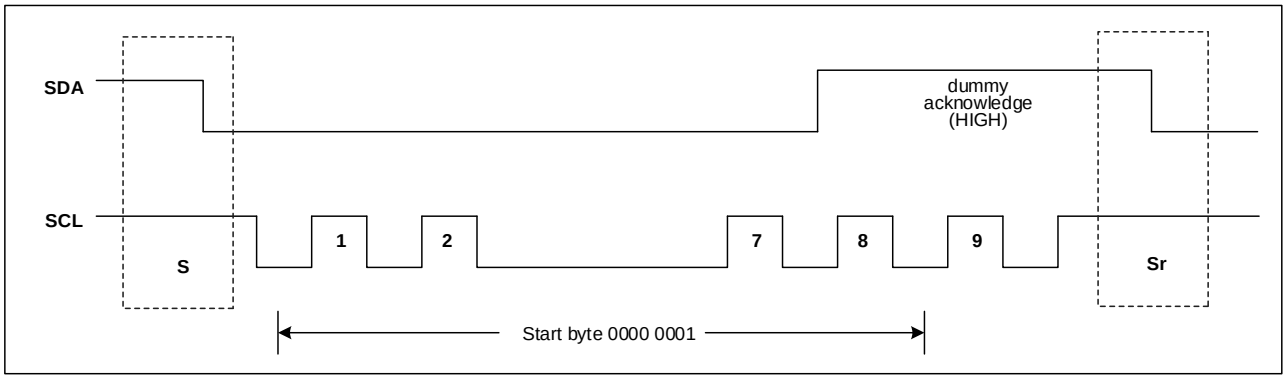


Start Byte transfer protocol

The Start Byte transfer protocol is set up for systems that do not have a dedicated I2C hardware module. When I2C is set as a master, it supports the generation of Start Byte transfers at the beginning of every transfer in case a slave device requires it.

This protocol consists of seven zeros followed by a 1, as illustrated in the following figure. This allows the processor that is polling the bus to under-sample the address phase. Once the processor detects a 0, it switches from the under sampling rate to the correct rate of the master.

Figure 14-8 Start Byte transfer



The Start Byte procedure:

1. Master generates a START condition
2. Master transmits the Start byte (0000 0001)
3. Master transmits the ACK clock pulse (to conform to the byte handling format used on the bus)
4. No slave responds with an ACK pulse
5. Master generates a RESTART condition

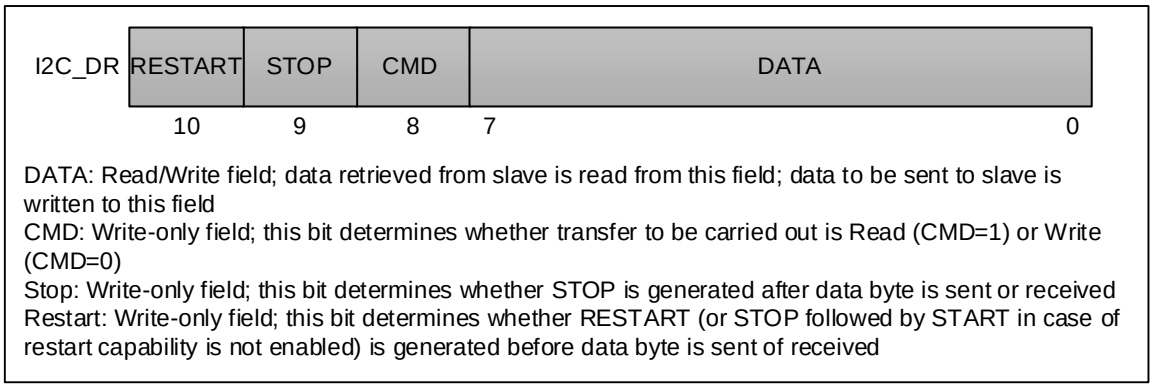
A hardware I2C receiver does not respond to the Start Byte because it is a reserved address and resets after the RESTART condition is generated.

14.3.3.4 TX FIFO management and Start, Stop, and Restart generation

When operating in host mode, the I2C module generates a stop condition on the bus whenever the TX FIFO is empty. If the repeated start generation function is enabled (RESTART = 1), a RESTART condition is generated when the transfer direction changes from read to write or write to read. If the RESTART condition is not enabled, a START condition will be generated after the STOP condition.

The following figure shows the bits of the DR register.

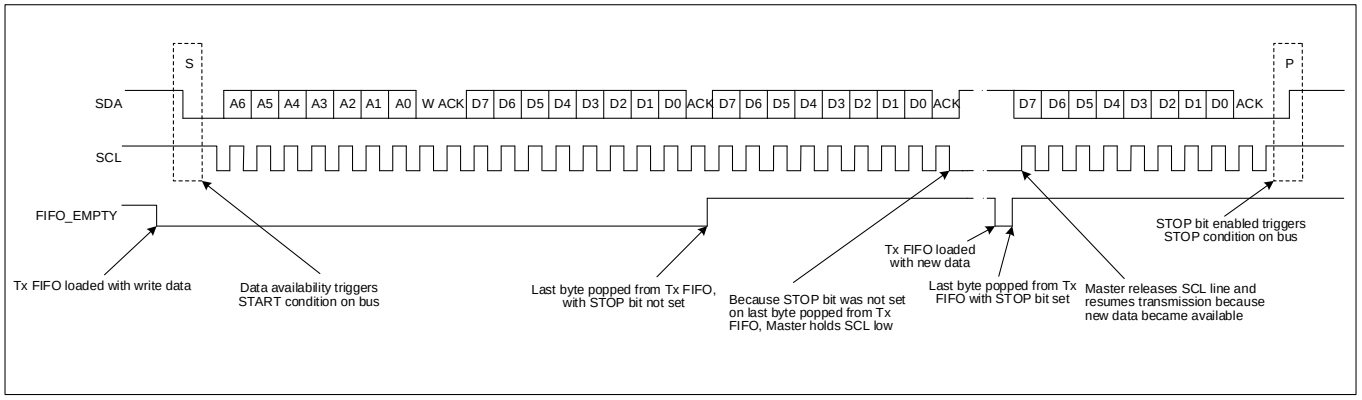
Figure 14-9 I2C_DR register



DATA: Read/Write field; data retrieved from slave is read from this field; data to be sent to slave is written to this field
CMD: Write-only field; this bit determines whether transfer to be carried out is Read (CMD=1) or Write (CMD=0)
Stop: Write-only field; this bit determines whether STOP is generated after data byte is sent or received
Restart: Write-only field; this bit determines whether RESTART (or STOP followed by START in case of restart capability is not enabled) is generated before data byte is sent of received

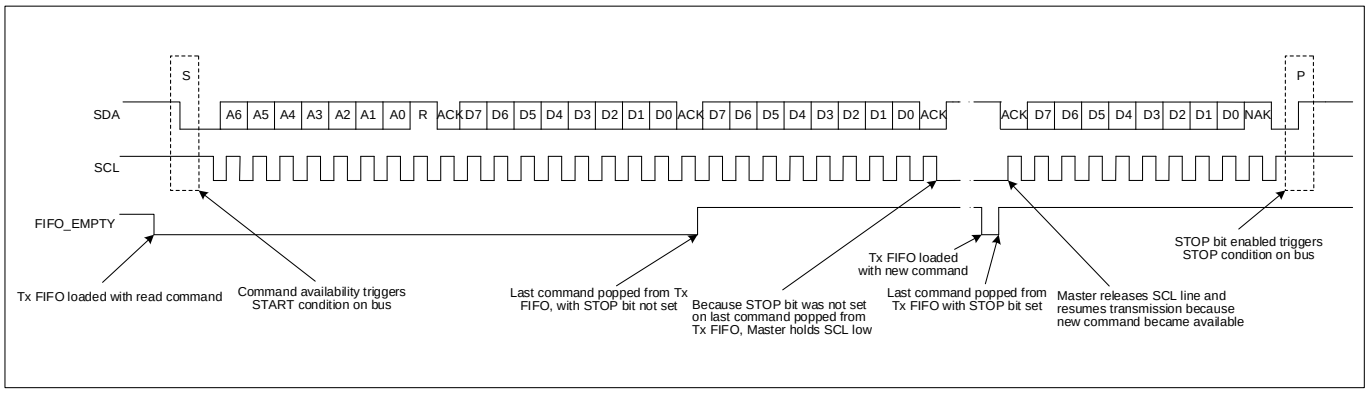
The following timing diagram depicts the behavior of I2C module operating in master transmit mode when TX FIFO becomes empty or generates STOP.

Figure 14-10 Master-transmitter, TX FIFO empty or generate STOP



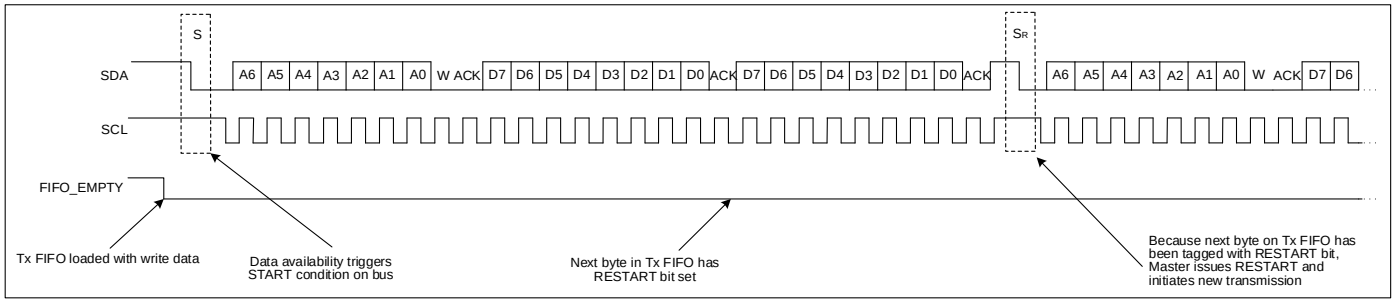
The following timing diagram depicts the behavior of I2C module operating in master receive mode when TX FIFO becomes empty or generates STOP.

Figure 14-11 Master-receiver, TX FIFO empty or generate STOP



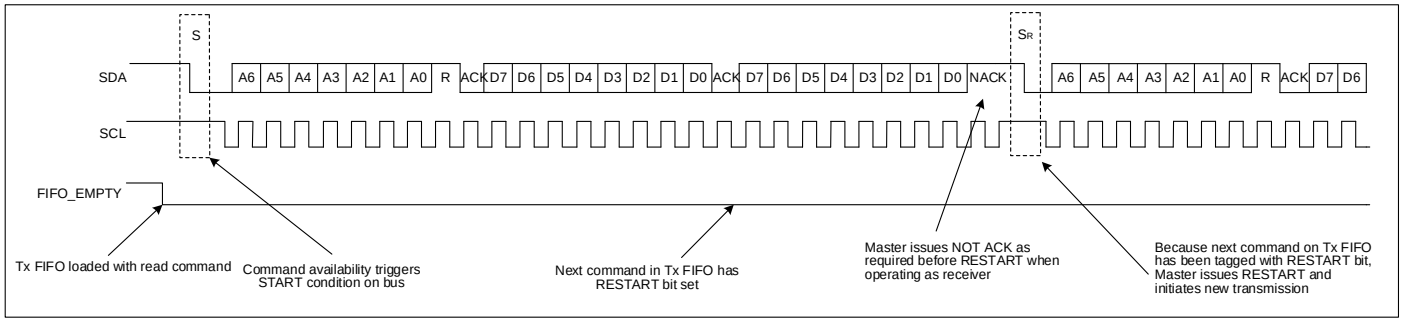
The following timing diagram illustrates the behavior of I2C operating in the master transmit mode when RESTART (I2C_CR.REPEN=1) is generated. If I2C_CR.REPEN (bit 5) is ‘0’, then a STOP followed by a START is issued, in order to replace the RESTART.

Figure 14-12 Master-transmitter, with RESTART



The following timing diagram illustrates the behavior of I2C operating in the master receive mode when RESTART is generated.

Figure 14-13 Master-receiver, with RESTART



14.3.3.5 Arbitration

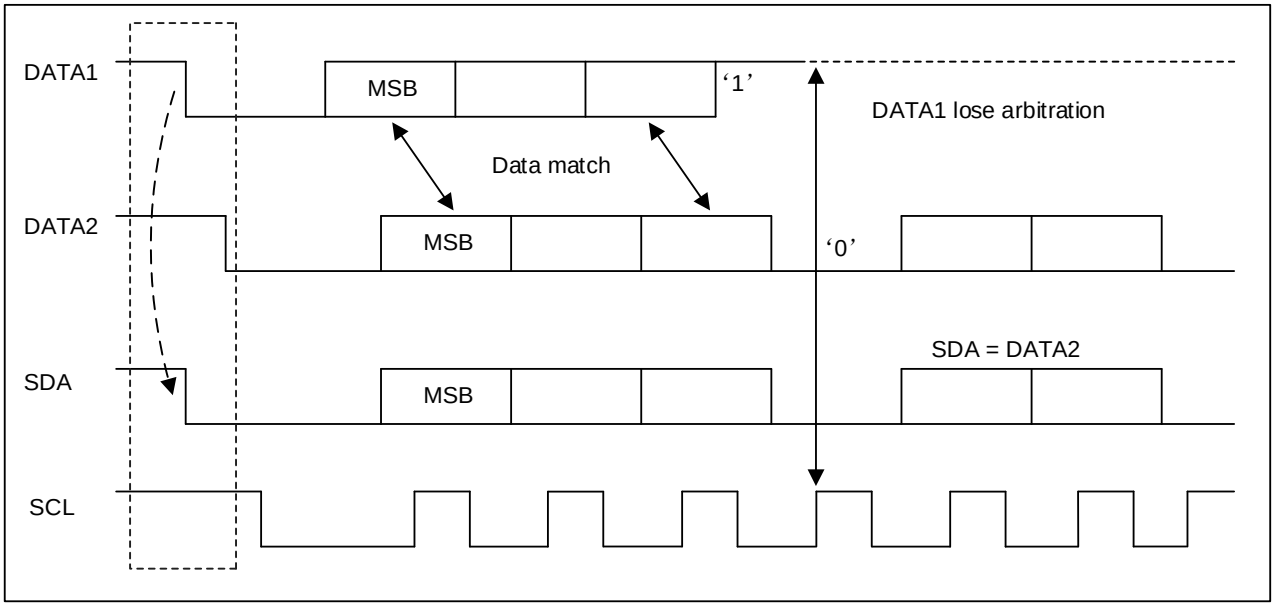
If there are several masters on the same bus, there is an arbitration procedure if they try to take control of the bus by generating the START condition at the same time (if several masters try to take control of the bus at the same time, only one master can take control of the bus with intact messages). Once a master has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state.

Arbitration takes place on the SDA line, while the SCL line is high. If there are two or more masters trying to send messages to the bus, the master, which transmits a 1 while the other master transmits 0, loses arbitration. The master that lost arbitration can continue to generate clock pulses until the end of the byte transfer. If all masters are addressing the same device, the arbitration could go into the data phase.

Upon detecting that it has lost arbitration, the I2C interface stops generating SCL.

The following figure illustrates the timing of two masters arbitrating on the bus.

Figure 14-14 Dual master arbitration



14.3.3.6 Clock synchronization

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is only valid during the high period of clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL and transitions the SCL to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a wait state until the SCL transitions to 1.

All masters then count off their high time, and the master with the shortest high time transitions the SCL to 0. The masters then count out their low time and the one with the longest low time forces the other masters into a wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in the following figure.

Figure 14-15 Clock synchronization (schematic diagram)

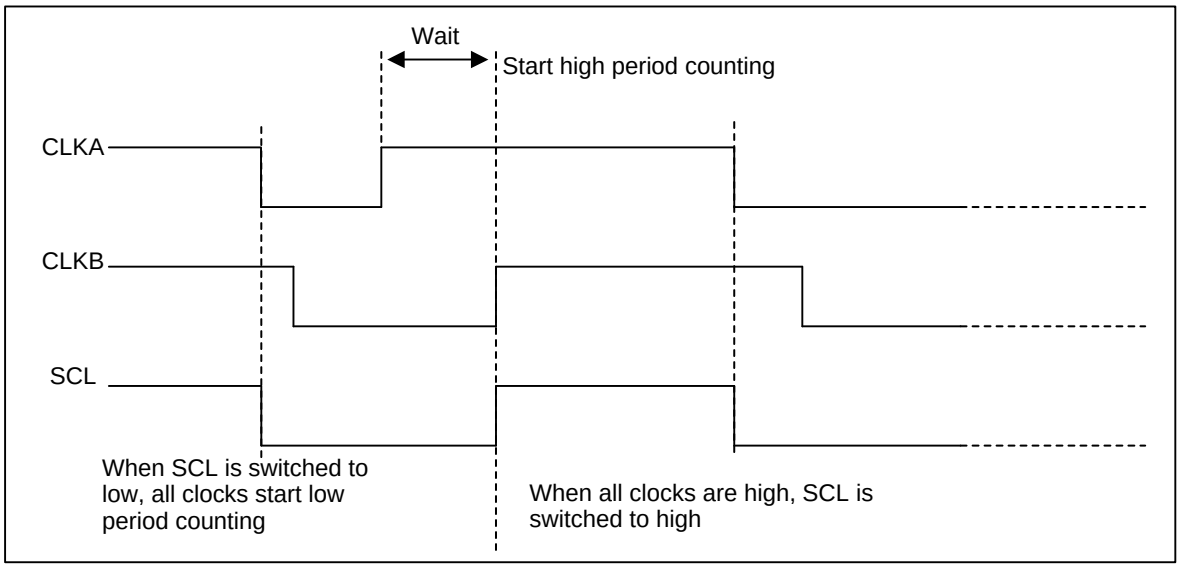
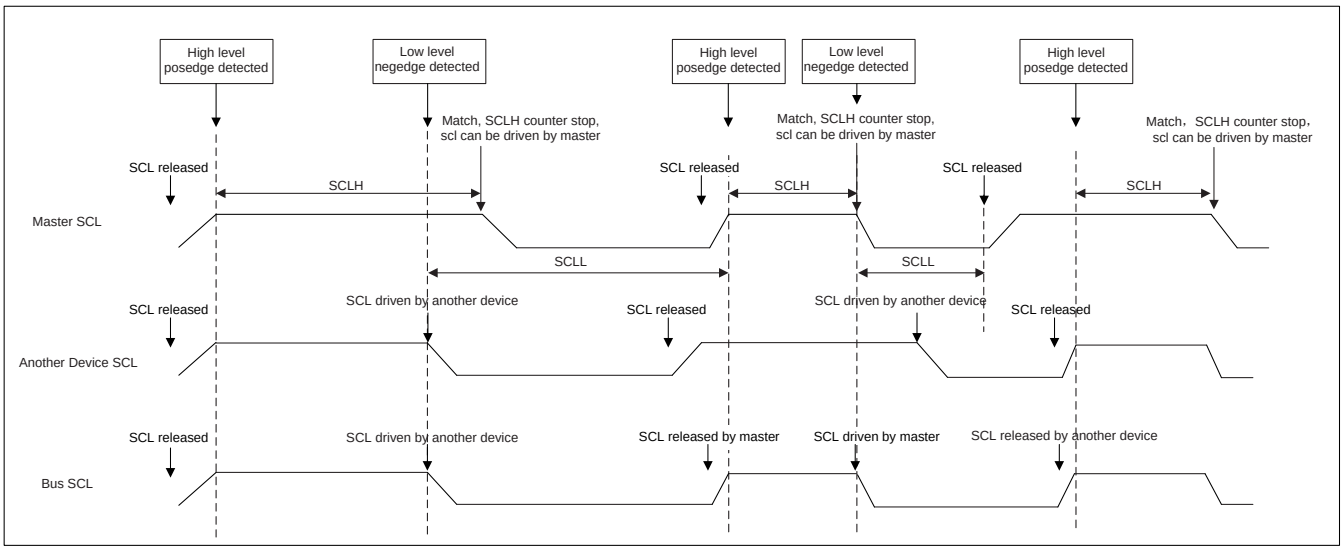


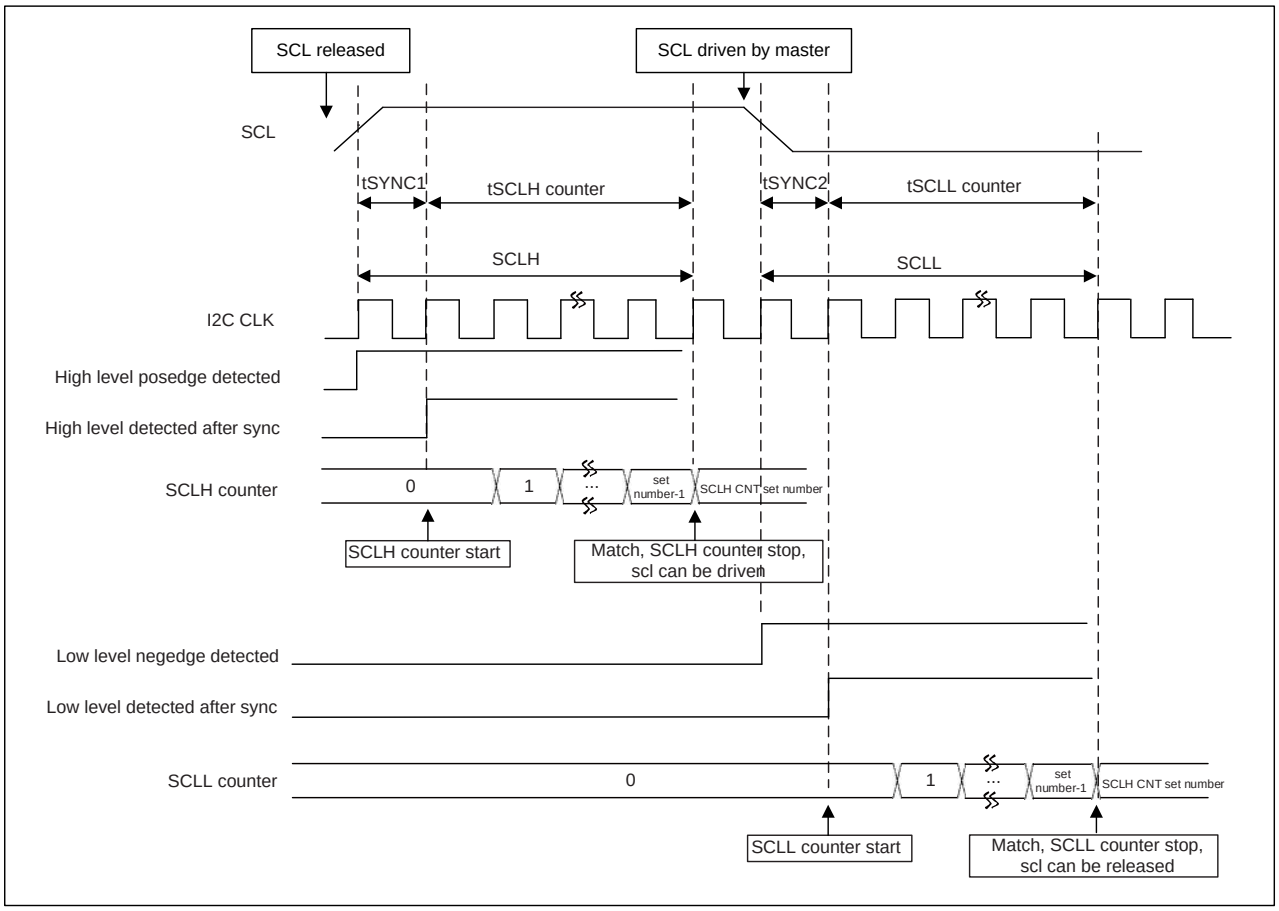
Figure 14-16 Clock synchronization (timing diagram)



14.3.3.7 SCL configuration

For SCL configuration of I2C, refer to the following:

Figure 14-17 SCL generation timing



Standard mode:

SCLH = (SSHR + 12) x I2C CLK + tSYNC1

SCLL = (SSLR + 1) x I2C CLK + tSYNC2

Note: tSYNC1 is equal to 0 ~ 1 I2C CLK and tSYNC2 is equal to 0 ~ 1 I2C CLK

Fast mode:

SCLH = (FSHR + 12) x I2C CLK + tSYNC1

SCLL = (FSLR + 1) x I2C CLK + tSYNC2

Note: tSYNC1 is equal to 2 ~ 3 I2C CLK and tSYNC2 is equal to 2 ~ 3 I2C CLK

14.3.4 Operating mode

I2C interface can operate in one of the four following modes:

Slave transmitter mode

Slave receiver mode

Master transmitter mode

Master receiver mode

Note: I2C can only operate in the Master or Slave mode but it cannot operate in both modes at the same time. Therefore, make sure that the register I2C_CR.DISSLAVE (bit 6) and I2C_CR.MASTER (bit 0) are not set to 0 and 1 (or 1 and 0 respectively).

14.3.4.1 Slave mode

Initial configuration

1. Disable I2C by writing a 0 to I2C_ENR.ENABLE (bit 0).
2. Configure the I2C_SAR register to set the slave address (this is the address to which the I2C responds).
3. Configure the I2C_CR.SLAVE10 (bit 3) to specify which type of addressing is supported (7- or 10-bit address); write 0 to I2C_CR.DISSLAVE (bit 6) and I2C_CR.MASTER (bit 0).
4. Enable the I2C by setting the I2C_ENR.ENABLE (bit 0) to 1.

Slave-transmitter operation for a single byte

When another I2C master device addresses the I2C interface and requests data, the I2C interface acts as a slave-transmitter and the following steps occur:

1. The other I2C master device initiates an I2C transfer with an address that matches the slave address in the I2C_SAR register.
2. The I2C interface acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.
3. The I2C interface asserts the RD_REQ interrupt (bit 5 of the I2C_RAWISR) and holds the SCL line low. The bus stays in the wait state until the software responds. If the RD_REQ interrupt has been masked (I2C_IMR bit 5 = 0), then it is recommended that CPU does periodic query of the I2C_RAWISR register:
   a. Reads that indicate I2C_RAWISR.RD_REQ (bit 5) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted.
   b. Software must then act to satisfy the I2C transfer requirements.
   c. The timing interval is generally 10 times the SCL clock period (for example, for 400kbps, the timing interval is 25us).
4. If there is any data remaining in the TX FIFO before receiving the read request, the I2C interface asserts a TX_ABRT interrupt (I2C_RAWISR bit 6) to flush the old data from the TX FIFO (when I2C_CR. SLV_TX_ABRT_DIS = 0).
5. Therefore, it is necessary for software to clear the TX_ABRT interrupt by reading the I2C_TX_ABRT register before attempting to write into the TX FIFO. If the TX_ABRT interrupt has been masked (I2C_IMR bit 6 = 0), then it is recommended that CPU does periodic query of the I2C_RAWISR register: Reads that indicate I2C_RAWIS.TX_ABORT (bit 6) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.
6. Software writes data to the I2C_DR register and writes a 0 in bit 8.
7. Software must clear the RD_REQ (I2C_RAWISR bit 5) and TX_ABRT (I2C_RAWISR bit 6) interrupts.
8. The I2C interface releases SCL and transmits the byte.
9. The master controls the bus by issuing a RESTART condition or releases the bus by issuing a STOP condition.

Slave-receiver operation for a single byte

When another master device addresses the I2C interface and sends data, the I2C interface acts as a slave-receiver and the following steps occur:

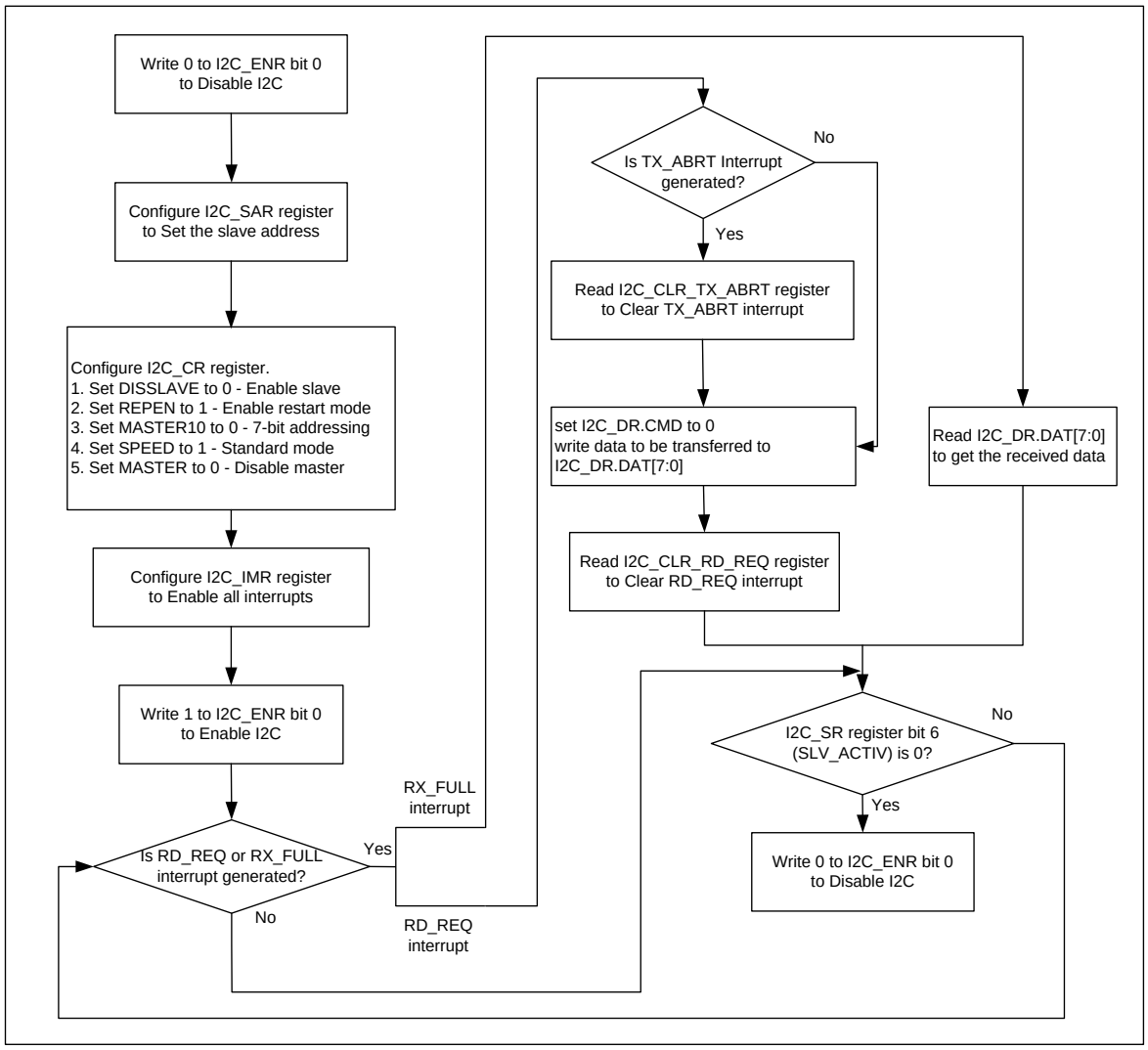
1. Another I2C master device initiates an I2C transfer with a sending address that matches the slave address in the I2C_SAR register.

2. The I2C interface acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-receiver.
3. I2C interface receives the transmitted data from master and places it in the receive buffer.
4. I2C interface asserts the RX_FULL interrupt (I2C_RAWISR bit 2).
5. If the RX_FULL interrupt has been masked (I2C_IMR bit 2 = 0), then it is recommended that CPU does periodic query of the I2C_SR register. Reads that indicate I2C_SR.RFNE (bit 3) being set to 1 can be treated as the equivalent of the RX_FULL interrupt being asserted.
6. Software may read the I2C_DR register (bits 7:0) to get the data that was received.
7. The master controls the bus by issuing a RESTART condition or releases the bus by issuing a STOP condition.

Program flow chart

The following chart shows a programming model when I2C interface is set as a slave:

Figure 14-18 Flow chart (I2C interface as a slave)



Slave-transfer operation for bulk transfers

In the standard I2C protocol, all transactions are single byte transactions and the program responds to a master read request by writing one byte into the slave's TX FIFO. When a slave (slave-transmitter) receives a read request (RD_REQ) from the master (master-receiver), there should be at least one entry placed into the slave-transmitter's TX FIFO. The I2C interface is designed to handle more data in the TX FIFO so that subsequent read requests can receive that data without raising an interrupt. Ultimately, this eliminates latencies significantly being incurred by the interrupt for data each time.

This mode is only applicable when I2C interface is acting as a slave-transmitter. If the master acknowledges the data sent by the slave-transmitter and there is no data in the slave's TX FIFO, the I2C interface holds the SCL line low while it raises the read request interrupt (RD_REQ); moreover, it will not release the SCL line unless the data in the TX FIFO is prepared.

If the RX_REQ interrupt has been masked (I2C_ISR bit 5 = 0), then it is recommended that the software does periodic query of the I2C_RAWISR register. Reads that indicate I2C_RAWISR.RX_REQ (bit 5) being set to 1 can be treated as the equivalent of the RX_REQ interrupt being asserted.

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must raise the RD_REQ interrupt request again because the master is requesting for more data.

If the master is to receive n bytes from the I2C interface but the program wrote a number of bytes larger than n to the TX FIFO, then when the slave finishes sending the requested n bytes, it clears the TX FIFO and ignores any excess bytes.

14.3.4.2 Master mode

Initial configuration

1. Disable I2C by writing a 0 to I2C_ENR.ENABLE (bit 0).
2. Set the I2C operation speed mode (Stand mode or Fast mode) by configuring I2C_CR.SPEED (bit 2:1). Meanwhile, make sure I2C_CR.DISSLAVE (bit 6) = 1 and I2C_CR.MASTER (bit 0) = 1.
3. Write to the I2C_TAR (this register can also be set to configure a General Call address or a Start Byte command) the address of the I2C device to be addressed.
4. Enable the I2C interface module by setting the I2C_ENR.ENABLE (bit 0) to 1.
5. Now write the transfer direction and data to I2C_DR. Thus, I2C interface will generate a START condition and send the address byte data.

If the I2C_DR register is written before the I2C interface is enabled, the data and commands are lost as the buffers are kept cleared when the I2C interface is not enabled.

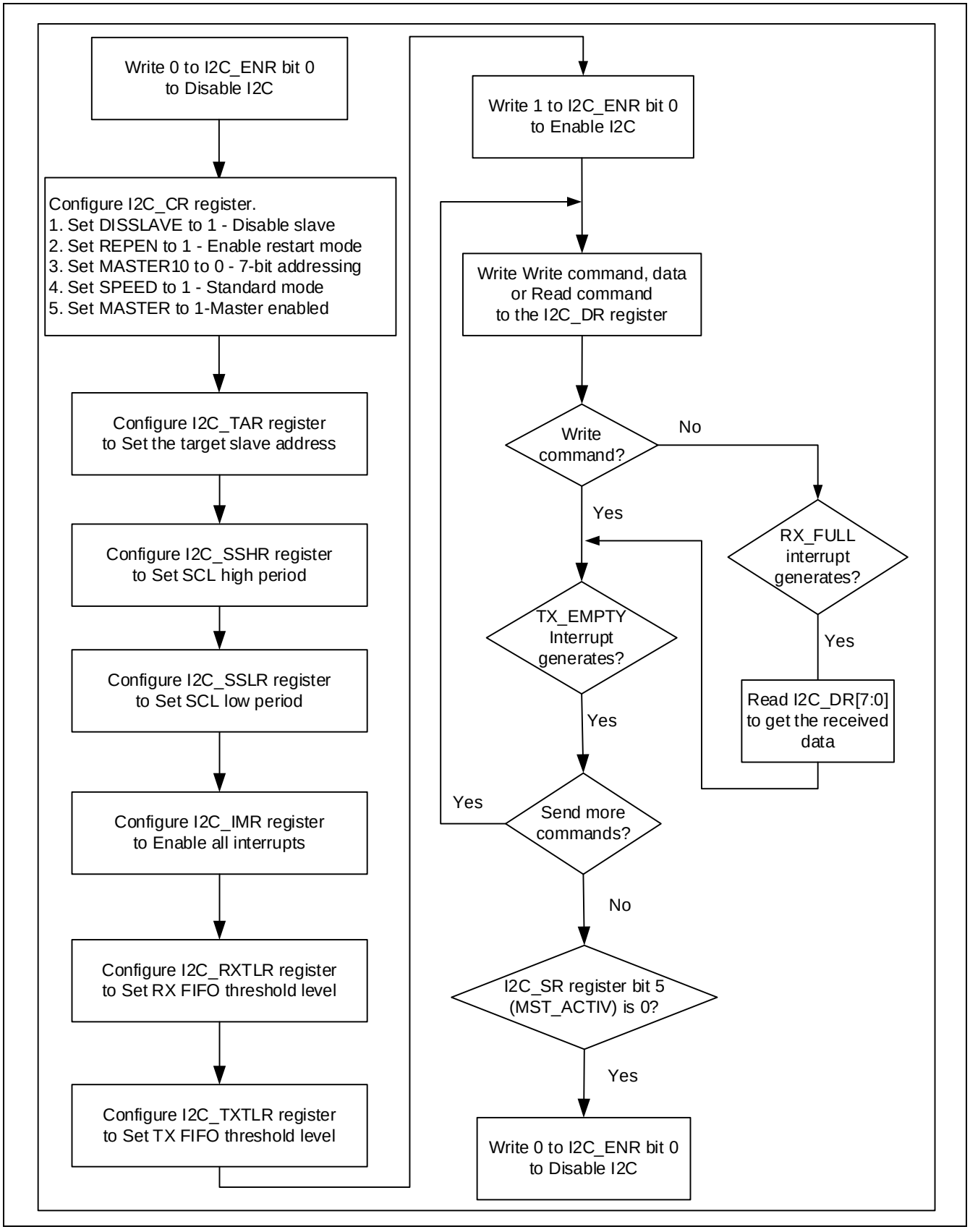
Master transmit and Master receive

The I2C interface supports switching back and forth between reading and writing dynamically. When transmitting data, write the data to be written to the lower byte of I2C_DR. The I2C_DR.CMD (bit 8) should be written to 0 for write operations. Subsequently, a read command does not require the configuration of lower byte of I2C_DR, and a 1 should be written to I2C_DR.CMD. If the TX FIFO is empty, the I2C module holds SCL low until the next command is written into the TX FIFO.

Program flow chart

The following flow chart shows a programming model when I2C interface is set as a master:

Figure 14-19 Flow chart (I2C interface as a master)



14.3.4.3 Abort transfer

The I2C_ENR.ABORT (bit 1) allows the software to relinquish the I2C bus before completing the issued transfer commands from the TX FIFO. In response to an ABORT request, the I2C module issues the STOP condition over the I2C bus, alongside TX FIFO flush. Aborting the transfer is allowed only in master mode of operation.

Procedure:

1. Stop filling the TX FIFO (I2C_DR) with new commands
2. When operating in DMA mode, disable the transmit DMA by setting I2C_DMA.TXEN (bit 1) to 0
3. Set I2C_ENR.ABORT (bit 1) to 1
4. Wait for the TX_ABRT interrupt

14.3.5 Interrupt

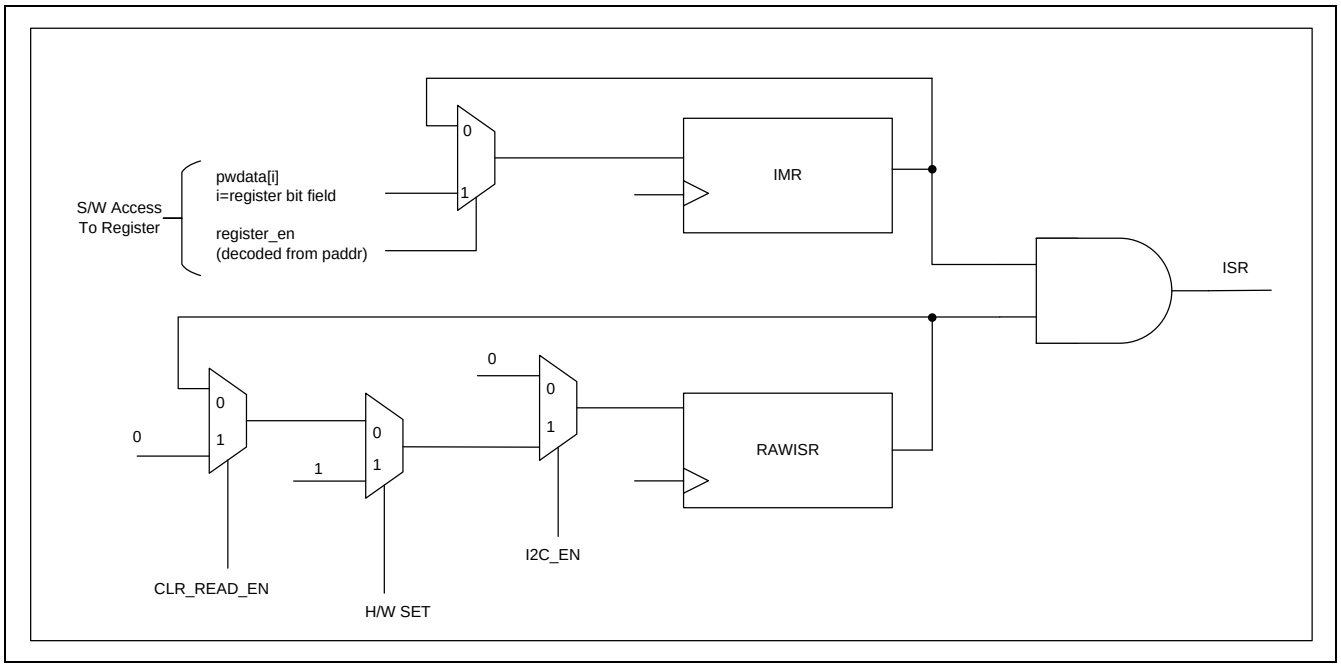
The table below gives the list of I2C interrupt bits, as well as the way to set and clear them. Some bits are set by hardware and cleared by software, others are set and cleared by hardware.

Table 14-3 Setting and clearing the interrupt bits

Interrupt bit	Set by hardware/Cleared by software	Set and cleared by hardware
GEN_CALL	√	x
START_DET	√	x
STOP_DET	√	x
ACTIVITY	√	x
RX_DONE	√	x
TX_ABRT	√	x
RD_REQ	√	x
TX_EMPTY	x	√
TX_OVER	√	x
RX_FULL	x	√
RX_OVER	√	x
RX_UNDER	√	x

The figure below describes the operation of the interrupt registers where the bits are set by hardware and cleared by software.

Figure 14-20 I2C interrupt mechanism



14.4 Register

14.4.1 Overview of registers

Table 14-4 Overview of I2C registers

Offset	Acronym	Register Name	Reset
0x00	I2C_CR	Control Register	0x0000007F
0x04	I2C_TAR	Target Address Register	0x00000055
0x08	I2C_SAR	Slave Address Register	0x00000055
0x10	I2C_DR	Data Command Register	0x00000000
0x14	I2C_SSHR	Standard Mode SCL High Count Register	0x00000190
0x18	I2C_SSLR	Standard Mode SCL Low Count Register	0x000001D6
0x1C	I2C_FSHR	Fast Mode SCL High Count Register	0x0000003C
0x20	I2C_FSLR	Fast Mode SCL Low Count Register	0x00000082
0x2C	I2C_ISR	Interrupt Status Register	0x00000000
0x30	I2C_IMR	Interrupt Mask Register	0x000008FF
0x34	I2C_RAWISR	RAW Interrupt Status Register	0x00000000
0x38	I2C_RXTLR	Receive Threshold Register	0x00000000
0x3C	I2C_TXTLR	Transmit Threshold Register	0x00000000
0x40	I2C_ICR	Combined and Independent Interrupt Clear Register	0x00000000
0x44	I2C_RX_UNDER	RX_UNDER Interrupt Clear Register	0x00000000
0x48	I2C_RX_OVER	RX_OVER Interrupt Clear Register	0x00000000
0x4C	I2C_TX_OVER	TX_OVER Interrupt Clear Register	0x00000000
0x50	I2C_RD_REQ	RD_REQ Interrupt Clear Register	0x00000000
0x54	I2C_TX_ABRT	TX_ABRT Interrupt Clear Register	0x00000000
0x58	I2C_RX_DONE	RX_DONE Interrupt Clear Register	0x00000000
0x5C	I2C_ACTIV	ACTIVITY Interrupt Clear Register	0x00000000
0x60	I2C_STOP	STOP_DET Interrupt Clear Register	0x00000000
0x64	I2C_START	START_DET Interrupt Clear Register	0x00000000
0x68	I2C_GC	GEN_CALL Interrupt Clear Register	0x00000000
0x6C	I2C_ENR	Enable Register	0x00000000
0x70	I2C_SR	Status Register	0x00000006
0x74	I2C_TXFLR	Transmit FIFO Level Register	0x00000000
0x78	I2C_RXFLR	Receive FIFO Level Register	0x00000000
0x7C	I2C_HOLD	SDA Hold Time Register	0x00000001
0x94	I2C_SETUP	SDA Setup Time Register	0x00000064
0x98	I2C_GCR	General Call ACK Register	0x00000001
0xB0	I2C_SLVMASK	Slave Address Mask Register	0x000003FF
0xB4	I2C_SLVRCVADDR	Slave Address Receive Register	0x00000000

14.4.2 I2C_CR Control Register

Address offset: 0x00

Reset value: 0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			PAD_SEL	Res.			EMPINT	STOPINT	DISSLAVE	REPEN	MASTER10	SLAVE10	SPEED		MASTER
			rw				rw	rw	rw	rw	rw	rw	rw		rw

Bit	Field	Description
31:13	Reserved	Reserved, must be kept at reset value
12	PAD_SEL	PAD MUX

		Exchange the corresponding PADs of SCL and SDA 0:PAD0 ≒ SCL；PAD1 ≒ SDA 1:PAD0 ≒ SDA；PAD1 ≒ SCL
11:9	Reserved	Reserved, must be kept at reset value
8	EMPINT	This bit controls whether a TX_EMPTY interrupt is generated. For details, refer to the I2C_RAWISR register.
7	STOPINT	This bit controls whether a STOP interrupt is generated in the Slave mode. 0: STOP interrupt is generated regardless of whether it's addressed or not. 1: STOP interrupt is generated only when it's addressed. Note: During a General Call address, if this bit is set, the slave doesn't generate a STOP interrupt. The STOP interrupt is generated only when the transmitted address is matched with the slave address (I2C_SAR).
6	DISSLAVE	This bit controls whether I2C has its slave disabled 0: slave enabled 1: slave disabled
5	REPEN	This bit determines whether RESTART conditions may be sent when I2C is acting as a master. 0: disabled 1: enabled RESTART condition can be replaced with a Stop and a subsequent START condition. When RESTART is disabled, I2C acting as a master is incapable of performing the following functions: Send a Start Byte Change transfer direction in the combined format Read operation with a 10-bit address If the above operations are performed, it will result in setting the I2C_RAWISR.TX_ABRT (bit 6) to 1.
4	MASTER10	Addressing mode when I2C is acting as a master 0: 7-bit addressing 1: 10-bit addressing
3	SLAVE10	When I2C is acting as a slave, this bit controls whether it responds to 7- or 10-bit addresses. 0: 7-bit addressing. I2C ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the I2C_SAR register are compared. 1: 10-bit addressing. I2C responds to only 10-bit addressing transfers that match the full 10 bits of the I2C_SAR register.
2:1	SPEED	These bits control at which speed the I2C operates in the Master mode. 01: Standard mode (up to 100Kbps) 10: Fast mode (up to 400Kbps)
0	MASTER	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled

I2C_CR.DISSLAVE (bit 6) and I2C_CR.MASTER (bit 0) are set according to the table below:

Table 14-5 DISSLAVE and MASTER settings

DISSLAVE (I2C_CR[6])	MASTER (I2C_CR[0])	Status
0	0	Slave device
0	1	Configuration disabled
1	0	Configuration disabled
1	1	Master device

14.4.3 I2C_TAR Target Address Register

Address offset: 0x04
Reset value: 0x0000 0055

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.				SPECIAL	GC	ADDR									
				rw	rw	rw									

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11	SPECIAL	This bit indicates whether software performs a General Call or Start Byte command. 0: Ignore GC (bit 10) and use ADDR (bit 9:0) normally 1: Perform special I2C command as specified in GC bit
10	GC	If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C. 0: General Call After issuing a General Call, only writes may be performed. I2C remains in General Call mode until the SPECIAL (bit 11) is cleared. 1: START BYTE
9:0	ADDR	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE command, the CPU needs to write only once into these bits.

14.4.4 I2C_SAR Slave Address Register

Address offset: 0x08
Reset value: 0x0000 0055

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.						ADDR									
						rw									

Bit	Field	Description
31:10	Reserved	Reserved, must be kept at reset value
9:0	ADDR	The slave address of I2C. For 7-bit addressing, only ADDR [6:0] is valid.

14.4.5 I2C_DR Data Command Register

Address offset: 0x10
Reset value: 0x0000 0000

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

Bit	Field	Description
31:11	Reserved	Reserved, must be kept at reset value
10	RESTART	Whether to generate RESTART before sending or receiving bytes 0: If REPEN is 1, RESTART is generated only when the previous command changes the transmission direction; if REPEN is 0, STOP is generated first and then START

		1: If REPEN is 1, RESTART is generated before data is received or sent (according to the value of CMD), whether the previous command changes the direction of data transmission or not; if REPEN is 0, STOP is generated first and then START
9	STOP	Whether to generate STOP after sending or receiving bytes 0: No STOP is generated after the current byte, whether TX FIFO is empty or not. If TX FIFO is not empty, the master device continues the current transmission (send or receive data based on the CMD value). If TX FIFO is empty, the master device will pull the SCL line low and hang the bus until TX FIFO receives a new command 1: STOP is generated after the current byte, whether the TX FIFO is empty or not. If TX FIFO is not empty, the host will immediately try to initiate a new transfer by sending START
8	CMD	This bit controls whether a read or a write is performed in the Master mode. 0: Write 1: Read When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, write operation to this bit is ignored. In slave-transmitter mode, a “0” indicates that data in I2C_DR register is to be transmitted.
7:0	DAT	This register contains the data to be transmitted or received on the I2C bus.

14.4.6 I2C_SSHR Standard Mode SCL High Count Register

Address offset: 0x14
Reset value: 0x0000 0190

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

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
15:0	CNT	This register sets the SCL clock high-period count (min valid value is 6) for Standard mode. Note: This register is programd to a value between 6 and 65525, because I2C uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of CNT+10.

14.4.7 I2C_SSLR Standard Mode SCL Low Count Register

Address offset: 0x18
Reset value: 0x0000 01D6

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

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
15:0	CNT	This register sets the SCL clock low-period count (min valid value is 8) for Standard mode.

14.4.8 I2C_FSHR Fast Mode SCL High Count Register

Address offset: 0x1C
Reset value: 0x0000 003C

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

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

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
15:0	CNT	This register sets the SCL clock high-period count (min valid value is 6) for Fast mode.

14.4.9 I2C_FSLR Fast Mode SCL Low Count Register

Address offset: 0x20

Reset value: 0x0000 0082

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

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
15:0	CNT	This register sets the SCL clock low-period count (min valid value is 8) for Fast mode.

14.4.10 I2C_ISR Interrupt Status Register

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.				R_GC	R_START	R_STOP	R_ACTIV	R_RX_DONE	R_TX_ABRT	R_RD_REQ	R_TX_EMPTY	R_TX_OVER	R_RX_FULL	R_RX_OVER	R_RX_UNDER
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11	R_GC	Refer to I2C_RAWISR register for more details
10	R_START	Refer to I2C_RAWISR register for more details
9	R_STOP	Refer to I2C_RAWISR register for more details
8	R_ACTIV	Refer to I2C_RAWISR register for more details
7	R_RX_DONE	Refer to I2C_RAWISR register for more details
6	R_TX_ABRT	Refer to I2C_RAWISR register for more details
5	R_RD_REQ	Refer to I2C_RAWISR register for more details
4	R_TX_EMPTY	Refer to I2C_RAWISR register for more details
3	R_TX_OVER	Refer to I2C_RAWISR register for more details
2	R_RX_FULL	Refer to I2C_RAWISR register for more details
1	R_RX_OVER	Refer to I2C_RAWISR register for more details
0	R_RX_UNDER	Refer to I2C_RAWISR register for more details

14.4.11 I2C_IMR Interrupt Mask Register

Address offset: 0x30

Reset value: 0x0000 08FF

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.				M_GC	M_START	M_STOP	M_ACTIV	M_RX_DONE	M_TX_ABRT	M_RD_REQ	M_TX_EMPTY	M_TX_OVER	M_RX_FULL	M_RX_OVER	M_RX_UNDER
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11	M_GC	Mask the interrupt status bit corresponding to I2C_ISR register
10	M_START	Mask the interrupt status bit corresponding to I2C_ISR register
9	M_STOP	Mask the interrupt status bit corresponding to I2C_ISR register
8	M_ACTIV	Mask the interrupt status bit corresponding to I2C_ISR register
7	M_RX_DONE	Mask the interrupt status bit corresponding to I2C_ISR register
6	M_TX_ABRT	Mask the interrupt status bit corresponding to I2C_ISR register
5	M_RD_REQ	Mask the interrupt status bit corresponding to I2C_ISR register
4	M_TX_EMPTY	Mask the interrupt status bit corresponding to I2C_ISR register
3	M_TX_OVER	Mask the interrupt status bit corresponding to I2C_ISR register
2	M_RX_FULL	Mask the interrupt status bit corresponding to I2C_ISR register
1	M_RX_OVER	Mask the interrupt status bit corresponding to I2C_ISR register
0	M_RX_UNDER	Mask the interrupt status bit corresponding to I2C_ISR register

14.4.12 I2C_RAWISR RAW Interrupt Status Register

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.				GC	START	STOP	ACTIV	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Description
31:12	Reserved	Reserved, must be kept at reset value
11	GC	General Call Set when a General Call address is received and acknowledged. It is cleared either by disabling I2C or when the CPU reads the I2C_GC.GC (bit 0). I2C stores the received data in the Rx buffer.
10	START	START condition detection This bit is set to 1 once a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in Master or Slave mode.
9	STOP	STOP condition detection The status of this bit differs based on the status of I2C_CR.STOPINT (bit 7): STOPINT = 0: This bit is set to 1 once a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in Master or Slave mode. In Slave mode, a STOP interrupt is generated regardless of whether the slave device is addressed. STOPINT = 1: In Master mode (MASTER = 1), this bit indicates whether a STOP condition has occurred on the I2C interface; In Slave mode (MASTER = 0), a STOP interrupt is generated only if the slave device is addressed.

8	ACTIV	This bit captures I2C module activity. Once this bit is set, it stays set unless one of the four methods is used to clear it: Disabling the I2C Reading the I2C_ACTIV register Reading the I2C_ICR register System reset Even if the I2C is idle, this bit remains set until cleared.
7	RX_DONE	Slave transmit done When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.
6	TX_ABRT	Transmit abort This bit is set to 1 when I2C, as an I2C transmitter, is unable to complete the data transfer. Note: The I2C flushes the TX FIFO and RX FIFO when there is a transmit abort. The TX FIFO remains in this flushed state until the I2C_TX_ABRT register is read. Once this read is performed, the TX FIFO is then ready to accept new data from the APB interface.
5	RD_REQ	Read request This bit is set to 1 when I2C is acting as a slave and another I2C master is attempting to read data from the I2C interface. The I2C interface holds the bus in a wait state (SCL = 0) until this interrupt is serviced, which means that the I2C interface has been addressed by another master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the I2C_DR register. This bit is set to 0 just after the processor reads the I2C_RD_REQ register.
4	TX_EMPTY	Transmit buffer empty The status of this bit differs based on the status of I2C_CR.EMPINT (bit 8): If EMPINT = 0, this bit is set to 1 when the transmit buffer is at or below the threshold value; If EMPINT = 1, this bit is set to 1 when the transmit buffer is at or below the threshold value and the most recent address/data transmission from the internal shift register is completed. It is automatically cleared by hardware when the transmit buffer level is above the threshold value.
3	TX_OVER	Transmit buffer over When the transmit buffer is full, the processor writes new data and causes overflow, which sets the bit to 1.
2	RX_FULL	Receive buffer full Set when the receive buffer is above the threshold. It is cleared by hardware when the receive buffer level is at or below the threshold.
1	RX_OVER	Receive buffer over Set if the receive buffer is full and an additional byte is received. The I2C interface acknowledges this, but new data will be lost.
0	RX_UNDER	Receive buffer over Set if the processor attempts to read the I2C_DR register when RX FIFO is empty.

14.4.13 I2C_RXTLR Receive Threshold Register

Address offset: 0x38

Reset value: 0x0000 0000

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

Bit	Field	Description
31:8	Reserved	Reserved, must be kept at reset value
7:0	TL	Receive FIFO threshold level Controls the level of entries that triggers the RX_FULL interrupt.

14.4.14 I2C_TXTLR Transmit Threshold Register

Address offset: 0x3C
Reset value: 0x0000 0000

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

Bit	Field	Description
31:8	Reserved	Reserved, must be kept at reset value
7:0	TL	Transmit FIFO threshold level Controls the level of entries that trigger the TX_EMPTY interrupt.

14.4.15 I2C_ICR Combined and Independent Interrupt Clear Register

Address offset: 0x40
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	ICR	Reading this register clears all combined and independent interrupts. This bit only clears software clearable interrupts, instead of hardware clearable interrupts.

14.4.16 I2C_RX_UNDER RX_UNDER Interrupt Clear Register

Address offset: 0x44
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	RX_UNDER	Reading this register clears the RX_UNDER interrupt (I2C_RAWISR [0]).

14.4.17 I2C_RX_OVER RX_OVER Interrupt Clear Register

Address offset: 0x48
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.															RX_OVER
															r

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	RX_OVER	Reading this register clears the RX_OVER interrupt (I2C_RAWISR [1]).

14.4.18 I2C_TX_OVER TX_OVER Interrupt Clear Register

Address offset: 0x4C
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	TX_OVER	Reading this register clears the TX_OVER interrupt (I2C_RAWISR [3]).

14.4.19 I2C_RD_REQ RD_REQ Interrupt Clear Register

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.															RD_REQ
															r

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	RD_REQ	Reading this register clears the RD_REQ interrupt (I2C_RAWISR [5]).

14.4.20 I2C_TX_ABRT TX_ABRT Interrupt Clear Register

Address offset: 0x54
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	TX_ABRT	Reading this register clears the TX_ABRT interrupt (I2C_RAWISR [6]). This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO.

14.4.21 I2C_RX_DONE RX_DONE Interrupt Clear Register

Address offset: 0x58
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.															RX_DONE
															r

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	RX_DONE	Reading this register clears the RX_DONE interrupt (I2C_RAWISR [7]).

14.4.22 I2C_ACTIV ACTIVITY Interrupt Clear Register

Address offset: 0x5C
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	ACTIV	Reading this register clears the ACTIV interrupt (I2C_RAWISR [8]) if the I2C is not active anymore. If the I2C is still active, the ACTIV interrupt bit continues to be set. It is cleared by hardware if the I2C module is disabled and if there is no further activity on the I2C bus. By reading this register, you get status of the I2C_RAWISR.ACTIV (bit 8).

14.4.23 I2C_STOP STOP_DET Interrupt Clear Register

Address offset: 0x60
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.															STOP
															r

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	STOP	Reading this register clears the STOP interrupt (I2C_RAWISR [9]).

14.4.24 I2C_START START_DET Interrupt Clear Register

Address offset: 0x64
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.															START
															r

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	START	Reading this register clears the START interrupt (I2C_RAWISR [10]).

14.4.25 I2C_GC GEN_CALL Interrupt Clear Register

Address offset: 0x68
Reset value: 0x0000 0000

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	GC	Reading this register clears the GC interrupt (I2C_RAWISR [11]).

14.4.26 I2C_ENR Enable Register

Address offset: 0x6C
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.														ABORT	ENABLE
														rw	rw

Bit	Field	Description
31:2	Reserved	Reserved, must be kept at reset value
1	ABORT	I2C transfer abort 0: ABORT not initiated or ABORT done 1: ABORT operation in progress The software can abort the I2C transfer by setting this bit when I2C acts as a master. The software cannot clear the ABORT bit once set. In response to an ABORT, the I2C controller issues a STOP and flushes the TX FIFO after completing the current transfer, then sets the TX_ABRT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation.
0	ENABLE	I2C module enable 0: I2C module disabled (TX and RX FIFOs are held in an erased state) 1: I2C module enabled

14.4.27 I2C_SR Status Register

Address offset: 0x70
Reset value: 0x0000 0006

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.									SLV_ ACTIV	MST_ ACTIV	RFF	RFN E	TFE	TFNF	ACTIV
									r	r	r	r	r	r	r

Bit	Field	Description
31:7	Reserved	Reserved, must be kept at reset value
6	SLV_ACTIV	Slave FSM activity status 0: Slave FSM is in IDLE state so the Slave part of I2C is not active 1: Slave FSM is not in IDLE state so the Slave part of I2C is active
5	MST_ACTIV	Master FSM activity status 0: Master FSM is in IDLE state so the Master part of I2C is not active 1: Master FSM is not in IDLE state so the Master part of I2C is active
4	RFF	Receive FIFO completely full

		0: Receive FIFO is not full 1: Receive FIFO is full
3	RFNE	Receive FIFO not empty 0: Receive FIFO is empty 1: Receive FIFO is not empty
2	TFE	Transmit FIFO completely empty 0: Transmit FIFO is not empty 1: Transmit FIFO is empty
1	TFNF	Transmit FIFO not full 0: Transmit FIFO is full 1: Transmit FIFO is not full
0	ACTIV	I2C activity status This bit is an ORed result of MST_ACTIV bit and SLV_ACTIV bit.

14.4.28 I2C_TXFLR Transmit FIFO Level Register

Address offset: 0x74
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.														CNT	
														r	

Bit	Field	Description
31:2	Reserved	Reserved, must be kept at reset value
1:0	CNT	Contains the number of valid data entries in the transmit FIFO (0~2)

14.4.29 I2C_RXFLR Receive FIFO Level Register

Address offset: 0x78
Reset value: 0x0000 0000

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

Bit	Field	Description
31:2	Reserved	Reserved, must be kept at reset value
1:0	CNT	Contains the number of valid data entries in the receive FIFO (0~2)

14.4.30 I2C_HOLD SDA Hold Time Register

Address offset: 0x7C
Reset value: 0x0000 0001

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

Bit	Field	Description
31:24	Reserved	Reserved, must be kept at reset value
23:16	RX_HOLD	Sets the internal SDA hold time (valid while SCL is high) in units of APB clock period, when I2C device acts as a receiver.
15:0	TX_HOLD	Sets the SDA hold time (after SCL goes from high to low) in units of APB clock period, when I2C device acts as a transmitter.

14.4.31 I2C_SETUP SDA Setup Time Register

Address offset: 0x94
Reset value: 0x0000 0064

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.								CNT							
								rw							

Bit	Field	Description
31:8	Reserved	Reserved, must be kept at reset value
7:0	CNT	SDA setup time (min value is 2) It is recommended that if the required delay is 1000ns, then for an APB clock frequency of 10MHz, this register should be programd to a value of 11.

14.4.32 I2C_GCR General Call ACK Register

Address offset: 0x98
Reset value: 0x0000 0001

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

Bit	Field	Description
31:1	Reserved	Reserved, must be kept at reset value
0	GC	ACK General Call 0: I2C neither responds with an ACK nor generates an interrupt, when it receives a General Call. 1: I2C responds with an ACK when it receives a General Call.

14.4.33 I2C_SLVMASK Slave Address Mask Register

Address offset: 0xB0
Reset value: 0x0000 03FF

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.						MASK									
						rw									

Bit	Field	Description
31:10	Reserved	Reserved, must be kept at reset value
9:0	MASK	Slave address mask 0: a corresponding bit in the I2C_SAR register is masked, not requiring comparison 1: a corresponding bit in the I2C_SAR register requires comparison

14.4.34 I2C_SLVRCVADDR Slave Address Receive Register

Address offset: 0xB4
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.						ADDR									
						r									

Bit	Field	Description
31:10	Reserved	Reserved, must be kept at reset value
9:0	ADDR	Slave address actually received by I2C

15. USART Universal Synchronous Asynchronous Receiver Transmitter

15.1 Introduction

The universal synchronous/asynchronous receiver transmitter (USART) offers a flexible means of full-duplex data exchange with external equipment. The USART offers a very wide range of baud rates using a built-in baud rate (including integer and decimal formats) generator.
It supports asynchronous mode (UART), synchronous mode, and the single wire half duplex communication is supported in the asynchronous mode (UART).

15.2 USART features

Supports full-duplex asynchronous communications and full-duplex clock synchronous communications

Baud rate generator (including integer and decimal formats)

- Programmable baud rate, available for a transmitter or receiver (with a minimum division factor of 1)

Separate transmit and receive FIFO registers, with separate enable bits for transmitter and receiver

Supports LSB/MSB reception and transmission modes

Programmable data word length (8 or 9 bits)

Configurable stop bits (1/2 stop bits)

Configurable parity check bit (odd parity, even parity, without parity check)

Supports the idle frame's generation (automatic output when TE is enabled) and receive detection

Support exchange of Tx and Rx pins and inversion of Tx and Rx signals

Supports following interrupt sources:

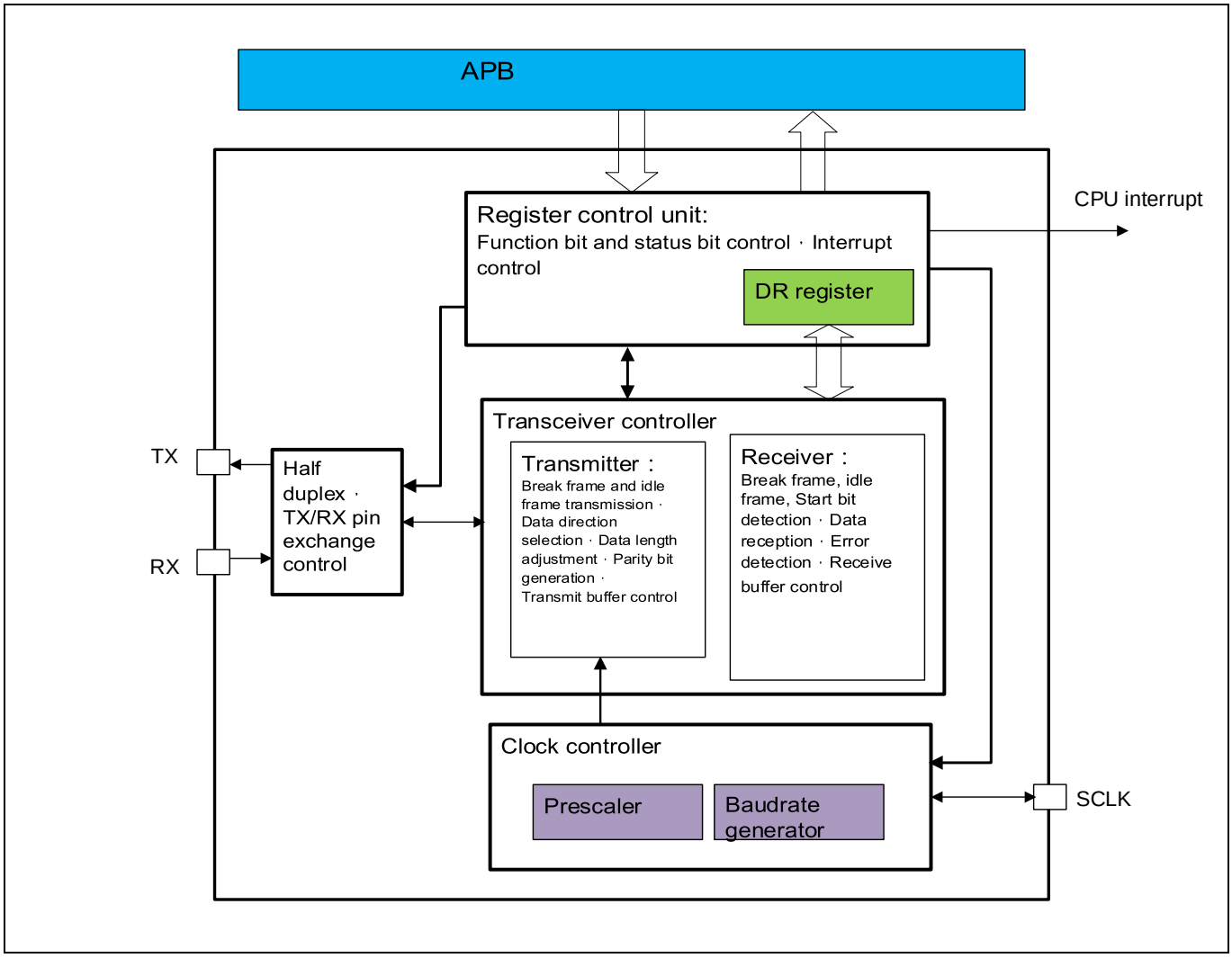
- Tx data register empty (TXE)
- Transmission complete (TC)
- Rx data valid (RXNE)
- Rx buffer overrun (OVR)
- Rx idle frame complete (IDLE)
- Parity error (PE)
- Noise flag (NF) and framing error (FE)

15.3 USART functional description

15.3.1 Functional block diagram

The functional block diagram of USART can be referred to as follows: USART can be divided into register-related control unit, receive/transmit data controller, clock controller, and pin control logic unit.

Figure 15-1 USART functional block diagram



15.3.2 Signal description

Signal Name	Type	Description
USART_SCLK	Output or input	Input or output clock pin in synchronous mode
USART_TX	Output or input	Transmit data pin, or half-duplex receive/transmit data pin
USART_RX	Input	Receive data pin (for full duplex)

15.3.3 Functional description

The full-duplex communication requires a minimum of two pins for USART: Receive Data In (RX) and Transmit Data Out (TX).

RX: External serial data is transmitted to the USART receiver through this pin. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise which is generated during the transmission.

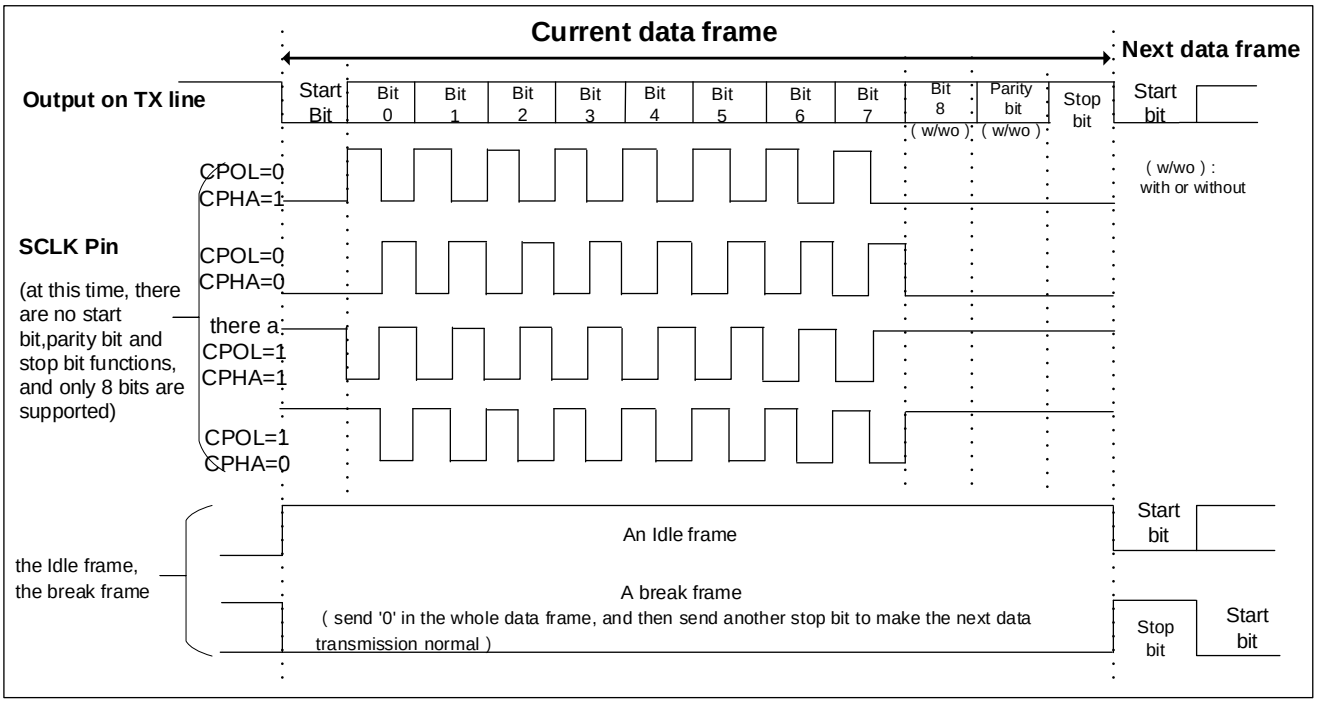
TX: The serial data generated internally in the USART transmitter is output through this pin. When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level.
The idle state is the initial state of the bus prior to transmission or reception.
There is one start bit represented by ‘0’.
In the USART communication process, a data frame (8 or 9 bits) can be transmitted and received in least significant bit first (LSB) or in most significant bit first (MSB).
The stop bit indicates the end of a frame with a ‘1’. The number of bits can be configured as 1 or 2.
The internal baud rate generator is controlled by programming the baud rate register USART_BRR to obtain the desired baud rate for data transmission (refer to the baud rate calculation equation in section 15.3.5).
By configuring the SWAP bit in the USART_CR2 register, the interface of Tx and Rx pins can be swapped.
By configuring the RXTOG/TXTOG bit in the USART_CR2 register, the level signal at the receiving / transmitting terminals can be reversed (including the start bit and stop bit).
In addition, the USART module supports synchronous mode (different from UART), and thus the following pins are required:

SCLK transmitter clock output or clock input: This pin is used in synchronous mode. In synchronous mode, the clock input and output functions are supported and the clock polarity and phase are configured via software.

15.3.4 Character description

Word length may be selected as being either 8 or 9 bits by programming the USART_CR1.DL bit. The TX pin is pulled down by the transmitter during the transmission start bit. It is pulled up during the transmission stop bit.
An idle frame is interpreted as an entire data frame of ‘1’s containing the stop bit. The start bit of the next data frame follows the idle frame.
A Break frame is interpreted as an entire data frame of ‘0’s containing the stop bit. At the end of the Break frame, the transmitter sends another '1' stop bit to acknowledge the start bit of the next frame (generating a falling edge to be detected by the Rx end).
The clock generated by the baud rate generator is provided for the transmitter or receiver, when the enable bit is set respectively for the transmitter and receiver.
The following figure shows a diagram of data frame format, the break frame, and the Idle frame.

Figure 15-2 UART data frame type diagram



15.3.5 Baud rate generator

For each communication mode, the baud rate can be configured according to the following equation.

For asynchronous mode (UART) and multiprocessor mode (including the multiprocessor under LIN protocol):

The communication baud rate:

$$f_{baudrate} = \frac{\text{PCLK}}{N\times\ (\text{MFD} + \text{FFD/N})}$$

Error E (%):

$$E(\%) = \{\frac{\text{PCLK}}{f_{baudrate} \times N\times\ (\text{MFD} + \text{FFD/N})} - 1\}\times 100$$

In the above equation, PCLK is the frequency of internal clock source; MFD and FFD are the integer and fractional frequency division that baud rate configuration of the USART_BRR; N=8× (2－OVER8).Select the oversampling mode by configuring the USART_CR1.OVER8; when OVER8=1 (8x oversampling), FFD [3:0] only use the lower 3 bits and user should configure the FFD [3] bit to 1'h0.

For synchronous mode:

The communication baud rate:

$$f_{baudrate} = \frac{\text{PCLK}}{4\times\ \text{MFD}}$$

In the above equation, PCLK is the frequency of the internal clock source; MFD is the integer baud rate configuration of the USART_BRR. In synchronous mode, fractional frequency division (FFD) is invalid, and user should configure FFD [3:0] to 4 'h0.

15.3.6 Sampling

The built-in detection circuit of UART detects the start of a data frame and the RX pin is sampled. The UART uses a clock of 8 or 16 times the data baud rate to sample the data on the RX pin.

The USART_CR1.OVER8 bit can be configured to select whether the USART uses a clock of 16 or 8 times the data baud rate to sample the data on the RX pin.

When 8x oversampling (OVER8=1) is selected, a higher speed (up to fPCLK/8) can be obtained, but the maximum tolerance of the receiver to clock deviation will be reduced.

15.3.7 Parity check control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the USART_CR1.PCE bit. The USART_CR1.PS bit is used to select odd parity or even parity for the data.

Even parity: the total number of ‘1s’ inside the data and the parity bit is an even number.

Odd parity: the total number of ‘1s’ inside the data and the parity bit is an odd number.

When parity control is effective：

The transmitter will automatically generate a parity bit and output it before the stop bit.

The receiver will detect the parity bit and judge whether it is wrong. If the parity bit is wrong, the hardware will automatically set USART_SR.PE flag, but the current received data will still be transferred from the shift register to USART_ DR register.

15.3.8 Transmitter

When the USART_CR1.TE bit is set, the transmitter is enabled and the serial data is output on the TX pin. The transmit data register USART_DR and the internal transmit shift register form a double buffer structure to transmit data continuously. As for UART mode, the data length (8 bits or 9 bits) can be selected by configuring the USART_CR1.DL bit.

15.3.8.1 Character transmission

During a USART transmission, data is written from the USART_DR register, and the data frame byte shifts out

least significant bit first (USART_CR1.MLS=0) or most significant bit first (USART_CR1.MLS=1) on the TX pin via the transmit shift register.
Data transmission order: a start bit, character, a parity bit (with or without), stop bits.
The number of stop bits can be configured in the register USART_CR2.STOP [1:0] as 1 or 2.
The USART_CR1.TE bit should not be reset before data transmission is complete. Otherwise, the baud rate generator will stop generating the clocks immediately. Subsequently, the latter part of the data being transmitted are lost.

15.3.8.2 Send break frame

Setting the USART_CR1.SBK bit to transmit a break frame. If the SBK bit is set to ‘1’ during the transmission, the break frame can't be sent on the TX pin until completing the current character transmission.
SBK bit is cleared automatically by hardware when the break frame is completed, and sending an additional high-level stop bit (to ensure that the start bit of the next frame data can be detected).
The length of break frames depends on the data frame length (CR2.DL), parity enable bit (CR1. PS) and stop bit (CR2. STOP). For example, when there is no parity bit and the stop bit is 1 bit, the break frame is 10 consecutive '0' bits if CR2.DL=0, while it is 11 consecutive '0' bits if CR2.DL=1.

15.3.8.3 Transmission configuration procedure

Refer to the following procedures to configure the USART to transmit data frames.
1. Configure the pin functions required by USART.
2. Enable the USART (USART_CR1.UE=1).
3. Configure the USART_BRR.
4. Configure the USART_CR1, USART_CR2, and USART_CR3 registers according to the requirements such as data frame transmission.
5. Enable the transmitter (USART_CR1.TE=1); set USART_CR1.TXEIEN=1 if the transmit data register empty interrupt is needed.
6. Wait until the transmit data register is empty. Write the communication data to USART_DR, and the data is transferred to the transmit shift register. Then the transmission starts.
7. Repeat step 6 if the data should be transmitted continuously.
8. Confirm that the transmission is complete by checking the USART_SR.TC bit. If TCIEN=1 is configured, a transition complete interrupt is generated after the transmission of the last frame is complete.

Note: The USART's transmitter supports two types of interrupts, namely the transmit data register empty interrupt TXE and the transition complete interrupt TC, which can be queried by the status bits in the USART_SR register. If TXEIEN=1, a TXE interrupt is generated when the value of USART_DR register is transferred to the transmit shift register. If TCIEN=1, a TC interrupt is generated when the last bit of data is transmitted and no further data is written to the USART_DR register.

15.3.9 Receiver

The data register USART_DR and the internal receive shift register form a double buffer structure to receive data continuously.
As for UART mode, the data length (8 bits or 9 bits) can be selected by configuring the USART_CR1.DL bit.
After the receiver enable bit USART_CR.RE is set to '1' and a start bit is detected, the data on the RX pin is received into the receive shift register. When a data frame is received, the data is transferred from the receive shift register to the data register USART_DR. Meanwhile, the status flag RXNE will be set to '1'. If RXNEIEN=1, the RXNE interrupt request is allowed. When the CPU uses this request to read the received data, the data can be read only once per request.
Data reception order: start bit -> data bit (MSB/LSB) -> parity bit (with or without) -> stop bits.

15.3.9.1 Receive break frame

When the USART receiver recognizes a break frame, it sets the USART_SR.FE flag (equivalent to receiving a '0' in the stop bit).

15.3.9.2 Receive idle character

When UART works normally, the receiver will set the USART_SR.IDLE flag when it receives an idle frame.

Configure IDLEIEN=1 to allow the IDLE interrupt request.

15.3.9.3 Reception configuration procedure

Refer to the following procedures to configure the USART to receive data frames.
1. Configure the functional pins required by USART.
2. Enable the USART (USART_CR1.UE=1).
3. Configure the USART_BRR.
4. Configure the USART_CR1, USART_CR2, and USART_CR3 registers according to the requirements such as data frame.
5. Configure the USART_BRR to set the communication baud rate (not necessary if the clock source is external).
6. Enable the receiver (USART_CR1.RE=1), set USART_CR1.RXNEIEN=1 if the receive interrupt is needed.
7. When a start bit is detected, the receiver receives the data into the receive shift register and checks the parity and stop bits. There are three error flags in total: PE, FE, and ORE. When no error occurs, the received data is transferred from the receive shift register to the USART_DR register, and the RXNE flag is set to '1'.
8. The received data can be read through the RXNE interrupt. Repeat step 7 if the data should be received continuously.
9. If any receive error has been detected during reception, the corresponding error flag can be set.

Note: To prevent overflow error, the RXNE bit must be cleared (software reads the data register USART_DR) before the end of the next character reception. Data reception can no longer be performed when any of the PE, FE, or ORE reception errors occur, but data reception can also be restarted by clearing all error flags.

When an overrun error occurs, the received data is lost and the ORE status bit is set to '1', but the RXNE interrupt is not generated.

When a parity error occurs, the received data is transferred to USART_DR and the PE status bit is set to '1', but the RXNE interrupt is not generated.

When a framing error occurs, the received data is transferred to USART_DR and the FE status bit is set to '1', but the RXNE interrupt is not generated.

15.3.10 Synchronous mode

By configuring USART_CR1.SAS bit to '1' to enable synchronous mode (Clock pin function will be valid at the same time).
In synchronous mode, user should configure USART_ CR2.HDSEL bit as '0'.
Synchronous mode supports master mode and slave mode: In master mode, the clock generated by the internal baud rate generator, it is used and output at the same time. In slave mode, the clock is input by SCLK pin. In the synchronous mode, USART can achieve data communication with SPI (At this time, user should configure the clock polarity and clock phase of SPI and USART to be consistent).

15.3.10.1 Clock description

Configure the USART_CR2.CLKEN bit to '1' to enable the clock pin function. In addition, select whether to use the internal baud rate clock or the input clock from the SCLK pin for data communication according to the USART_CR3.CKINE bit.
When the internal baud rate clock is selected, a synchronous clock can be output via the SCLK pin.
The transmission and reception of 1 data frame consist of 8 clock pulses.
When both RE and TE are '0', the clock output is stopped and fixed at the level configured by USART_CR2.CPOL.
The clock polarity is selected by configuring the USART_CR2.CPOL bit; the external clock phase is selected by configuring the USART_CR2.CPHA bit.

15.3.10.2 Clock synchronization description

When the SCLK pin is used as the clock output of the transmitter, the clock is output only in the data segment. 8 clock pulses are output for one frame of data. After the last bit is sent, the communication line holds the value of the last bit and the clock output is fixed at a high or low level (determined by the CPOL bit).
The USART receiver works differently in synchronous mode than in asynchronous mode. If RE=1, the data is sampled on the changing SCLK edge (rising or falling edge, depending on the CPOL and CPHA bit configuration) without any oversampling. Sufficient setup time and hold time must be ensured at this point to comply with the

timing requirements (similar to SPI protocol).
When the internal clock source is used, the baud rate generated by the internal baud rate generator is calculated in the following equation:

$$f_{baudrate} = \frac{\text{PCLK}}{4 \times \text{MFD}}$$

Where the communication baud rate is in MBps; PCLK is the frequency of the internal clock source; MFD is the integer baud rate configuration of the USART_BRR (Note that in synchronous mode MFD ≥ 2 should be configured). In synchronous mode, fractional frequency division (FFD) is invalid, and user should configure FFD [3:0] to 4 'h0.
Therefore, when the internal clock source is used and MFD=2, the maximum baud rate for synchronous mode is PCLK/8 (MBps).
When the external clock source is used, the maximum frequency required for the external input clock is PCLK/8 (MHz), at which time the maximum baud rate is also PCLK/8 (MBps).

15.3.11 Single-wire half-duplex communication

Configure the USART_CR3.HDSEL bit to ‘1’ to enable the single-wire half-duplex mode.
In single-wire half-duplex mode, the TX and RX pins are connected through the internal logic of the chip, meanwhile:

The RX pin is left suspended and not involved in the transmission. The TX of the USART is directly connected to the TX of another USART during transmission.

When data is being transmitted, the TX remains occupied until the stop bit is sent.

TX is always released when no data is transmitted. Thus, it acts as a standard IO in idle or in reception. It means that the IO corresponding to TX must be configured as floating input (or output high open-drain) when not driven by the USART.

Apart from the configuration of the single-wire pins, the rest configuration is similar to what is done in normal transmission.
Before the communication, both USARTs have RXEN enabled to stay in the wait-to-receive state. When the communication is required, both USARTs have to agree on which will send, with RE off and TE enabled in the USART_CR1 register of the sender. If both USARTs try to send data, a conflict will occur (the hardware will not block USARTs from sending: when the transmitters enable bit TE is set, TX will transmit data as long as USART_DR is written).

15.3.12 Interrupts

The USART module supports following interrupt sources:

Table 15-1 UART interrupt requests

Interrupt Event	Interrupt Status Bit	Enable Bit	UART	Synchronous mode
Transmit data register empty	TXE	TXEIEN	√	√
Transmission complete	TC	TCIEN	√	√
Receive data register full	RXNE	RXNEIEN	√	√
Idle line detected	IDLE	IDLEIEN	√	-
Parity error	PE	PEIEN	√	-
Noise flag	NF	ERRIEN	√	-
Overrun error	ORE	ERRIEN	√	√
Framing error	FE	ERRIEN	√	-

Note: "√" means the interrupt is used. "-" indicates that the interrupt is not used.

15.4 Register

15.4.1 Overview of registers

Table 15-2 Overview of USART registers

Offset	Acronym	Register Name	Reset
0x00	USART_SR	Status register	0x0000_00C0
0x04	USART_DR	Data register	0x0000_01FF
0x08	USART_BRR	Baud rate register	0x0000_0000
0x0C	USART_CR1	Control register 1	0x0000_0000
0x10	USART_CR2	Control register 2	0x0000_0000
0x14	USART_CR3	Control register 3	0x0000_6000

15.4.2 USART_SR status register

Address offset: 0x00

Reset value: 0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
								r	r	rc_w0	r	r	r	r	r

Bit	Field	Description
31:8	Reserved	Reserved, must be kept at reset value
7	TXE	Transmit data register Empty 0: Transmit data register is not empty (data is not transferred to the shift register) 1: Transmit data register is empty (data is transferred to the shift register) TXE bit is set and cleared automatically by hardware. The hardware will clear TXE when data is not transferred to the shift register (when written to DR register); the hardware will set this bit to '1' when data is transferred from DR to shift register.
6	TC	Transmission Complete 0: Transmission is not complete 1: Transmission is complete TC clearing condition: Write the transmit data to the data register when TE=1. TC setting condition: TE=0 or the transmit data register USART_DR is not updated when the last bit of the data frame is sent.
5	RXNE *Note 1	Receive data register not empty 0: Valid data is not received 1: Valid data is received Note: RXNE bit is set and cleared by hardware. It can also be cleared by writing '0' to it. RXNE bit is set automatically by hardware when the valid data is received. It is cleared by hardware after reading the received data.
4	IDLE *Note 2	IDLE frame detected 0: No Idle frame is detected 1: Idle frame is detected This bit is set automatically by hardware when an idle frame is detected.
3	ORE *Note 2	OverRun error 0: No overrun error 1: Overrun error is detected Note: This bit is set automatically by hardware when a new data frame is received while RXNE=1 (there already exists readable data).
2	NF *Note 2	Noise detected flag 0: No noise is detected

		1: Noise is detected Note: This bit is set automatically by hardware when noise is detected on a received signal line.
1	FE *Note 2	Framing error 0: No framing error is detected 1: Framing error occurred This bit is set automatically by hardware when: In asynchronous (UART) mode, the stop bit of the received data frame is low. Note: The received data will be transferred from the shift register to the data register when FE=1, but no RXNE interrupt request signal will be generated, and the subsequent data reception will be stopped.
0	PE *Note 2	Parity error 0: No parity error 1: Parity error This bit is set automatically by hardware when a parity error occurs during data reception. Note: The received data will be transferred from the shift register to the data register when PE=1, but no RXNE interrupt request signal will be generated, and the subsequent data reception will be stopped.

*Note 1: This bit can be cleared via software by writing '0' to it.
*Note 2: This bit can be cleared by software sequence (reading the status register and then performing a read access to the USART_DR data register).

15.4.3 USART_DR data register

Address offset: 0x04
Reset value: 0x0000 01FF

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

Bit	Field	Description
31:9	Reserved	Reserved, must be kept at reset value
8:0	DR[8:0]	Transmit /Receive data register Contains the Received or Transmitted data character, depending on whether it is read from or written to. When reading, it means receiving data; when transmitting, it means sending data. The most significant bit DR [8], is valid only in asynchronous (UART) mode when the data length is set to 9 bits (DL=1).

15.4.4 USART_BRR baud rate register

Address offset: 0x08
Reset value: 0x0000 0000

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

Bit	Field	Description
31:16	Reserved	Reserved, must be kept at reset value
19:4	MFD [15:0]	Mantissa frequency division of baudrate These 16 bits define the mantissa division of the USART baud rate.

		Before transmission or reception is enabled (TE or RE set to 1), this value must be changed according to the baud rate requirement.
3:0	FFD[3:0]	Fraction frequency division of baudrate These 4 bits define the fraction division of the USART baud rate. Before transmission or reception is enabled (TE or RE set to 1), this value must be configured according to the baud rate requirement. Note: When FFD [3:0] =4'h0, the fractional division function is disabled. In asynchronous (UART) mode and if USART_CR1.OVER8=1, FFD [3] is invalid so please configure FFD [3] =0. In synchronous mode, the fractional division is invalid, and should configure FFD [3:0] =4'h0.

15.4.5 USART_CR1 control register 1

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														SAS	MLS
														rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	Res.	UE	DL	Res.	PCE	PS	PEIEN	TXEIEN	TCIEN	RXNEIEN	IDLEIEN	TE	RE	Res.	SBK
rw		rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

Bit	Field	Description
31:18	Reserved	Reserved, must be kept at reset value
17	SAS	Synchronous/Asynchronous mode selection 0: UART mode (asynchronous) 1: Synchronous mode (synchronous) Note: User should configure this bit at TE=0 and RE=0.
16	MLS	MSB/LSB mode selection 0: LSB mode 1: MSB mode Note: User should configure this bit at TE=0 and RE=0.
15	OVER8	UART oversampling mode 0: 16x oversampling 1: 8x oversampling Note: User should configure this bit at TE=0 and RE=0.
14	Reserved	Reserved, must be kept at reset value
13	UE	USART enable 0: USART prescaler and pin outputs disabled 1: USART enabled When this bit is cleared, the USART prescaler and outputs are stopped and the current transition is stopped in order to reduce power consumption. Note: This bit is set and cleared by software.
12	DL	Data length 0: 8 bits 1: 9 bits Note: User should configure this bit at TE=0 and RE=0.
11	Reserved	Reserved, must be kept at reset value
10	PCE	Parity control enable 0: Parity control disabled 1: Parity control enabled Note: This bit is set or cleared by software. It must be kept at reset value in synchronous mode.
9	PS	Parity selection 0: Even parity 1: Odd parity Note: This bit is set and cleared by software. It is only valid when PCE=1.
8	PEIEN	PE interrupt enable 0: PE interrupt request disabled 1: PE interrupt request enabled Note: This bit is set and cleared by software.

7	TXEIEN	TXE interrupt enable 0: TXE interrupt request disabled 1: TXE interrupt request enabled Note: This bit is set and cleared by software.
6	TCIEN	Transmission complete interrupt enable 0: TC interrupt request disabled 1: TC interrupt request enabled Note: This bit is set and cleared by software.
5	RXNEIEN	RXNE interrupt enable 0: RXNE interrupt request disabled 1: RXNE interrupt request enabled Note: This bit is set and cleared by software.
4	IDLEIEN	IDLE interrupt enable 0: IDLE interrupt request disabled 1: IDLE interrupt request enabled Note: This bit is set and cleared by software.
3	TE	Transmitter enable 0: Transmitter disabled 1: Transmitter enabled Note: This bit is set and cleared by software. In synchronous mode, if simultaneous transmission and reception are required, user must configure both TE and RE bits at the same time to ensure proper timing sequence of clock and data.
2	RE	Receiver enable 0: Receiver disabled 1: Receiver enabled Note: This bit is set and cleared by software. In synchronous mode, if simultaneous transmission and reception are required, user must configure both RE and TE bits at the same time to ensure proper timing sequence of clock and data.
1	Reserved	Reserved, must be kept at reset value
0	SBK	Send break 0：No break frame is transmitted 1: Break frame will be transmitted This bit set is used to send break frame. It can be set by software and cleared automatically by hardware after sending the break frame.

15.4.6 USART_CR2 control register 2

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Field	Description
31:15	Reserved	Reserved, must be kept at reset value
15	SWAP	Swap pin between input and output 0: IO pin functions are not swapped 1: Input and output of IO pin functions are swapped Note: Once the SWAP bit is set, the MODE in the GPIOx_CRL register needs to be changed, e.g. the original input mode changes to the output mode.
14	Reserved	Reserved, must be kept at reset value
13:12	STOP[1:0]	STOP bit UART mode: 00: 1 stop bit 10: 2 stop bits 01: Reserved 11: Reserved

11	Reserved	Reserved, must be kept at reset value
10	CPOL	Clock polarity 0: The clock is at a low level when idle 1:The clock is at a high level when idle Note: This bit works in conjunction with the CPHA bit to produce the desired clock/data relationship (only valid in synchronous mode).
9	CPHA	Clock phase 0: The first clock transition is the first data capture edge 1: The second clock transition is the first data capture edge. Note: This bit works in conjunction with the CPOL bit to produce the desired clock/data relationship (only valid in synchronous mode).
8:0	Reserved	Reserved, must be kept at reset value

15.4.7 USART_CR3 control register 3

Address offset: 0x14

Reset value: 0x0000 6000

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

Bit	Field	Description
31:30	Reserved	Reserved, must be kept at reset value
29	TXTOG	Transmit toggle bit 0: Transmit toggle disabled 1: Transmit signal level's toggle enabled
28	RXTOG	Receive toggle bit 0: Receive toggle disabled 1: Receive signal level's toggle enabled
27:17	Reserved	Reserved, must be kept at reset value
16	CKINE	Clock input permission in synchronous mode This bit is used to control whether the clock is input externally (This bit is valid when CR1. SAS bit is' 1 '): 0: Clock is not input externally 1: Clock is input externally Note: User should configure this bit at TE=0 and RE=0.
15:12	Reserved	Reserved, must be kept at reset value
11	ONEBIT	UART one sample bit method enable 0: Three samples (majority decision) 1: One sample Note: User should configure this bit at TE=0 and RE=0. When one sample bit method is selected, the noise detected flag (USART_SR.NF) will be invalid.
10:4	Reserved	Reserved, must be kept at reset value
3	HDSEL	Single wire Half-duplex selection 0: Full-duplex mode 1: Half-duplex mode
2:1	Reserved	Reserved, must be kept at reset value
0	ERRIEN	Error interrupt enable 0: Error interrupt request disabled 1: Error interrupt request enabled Error interrupts include FE, ORE, and NF. Note: when reading or writing to DR, user can configure ERRIEN=1 to allow interrupt requests that sending to CPU for USART's abnormal communication.

16. SYSCFG System Controller

16.1 Introduction

The chip is composed of a set of system configuration registers. Main functions of these registers:

Manage external interrupt connected to GPIO port (pin configuration)

Remap memory to code initial area

System level configuration of some peripherals

16.2 Register

16.2.1 Register overview

Table 16-1 SYSCFG register overview

Offset	Acronym	Register Name	Reset
0x00	SYSCFG_CFGR	SYSCFG configuration register 1	0x00000000
0x08	SYSCFG_EXTICR1	SYSCFG external interrupt configuration register 1	0x00000000
0x0C	SYSCFG_EXTICR2	SYSCFG external interrupt configuration register 2	0x00000000
0x10	SYSCFG_EXTICR3	SYSCFG external interrupt configuration register 3	0x00000000
0x14	SYSCFG_EXTICR4	SYSCFG external interrupt configuration register 4	0x00000000
0x18	SYSCFG_PADHYS	SYSCFG PAD configuration register	0x00000000

16.2.2 SYSCFG_CFGR Configuration Register

This register has two control bits MEM_MODE, and is used to configure the initial address 0x00000000.
Address offset: 0x0
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.														MEM_MODE	
														rw	

Bit	Field	Description
31：2	Reserved	Reserved, must retain the reset value
1:0	MEM_MODE	Memory Mode Selection Bit) Control memory internal mapping to address 0x0000 0000. x0：Main Flash memory mapping to 0x0000 0000 01：System Flash mapping to 0x0000 0000 11：Built-in RAM mapping to 0x0000 0000

16.2.3 SYSCFG_EXTICR1 External Interrupt Configuration Register 1

Address offset: 0x08
Reset value: 0x0000 0000

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

EXTI3	EXTI2	EXTI1	EXTI0
rw	rw	rw	rw

Bit	Field	Description
31:16	Reserved	Reserved, must retain the reset value
15:0	EXTIx	EXTIx configuration（x=0 …3） Select the input source of EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin

16.2.4 SYSCFG_EXTICR2 External Interrupt Configuration Register 2

Address offset: 0x0C

Reset value: 0x0000 0000

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

Bit	Field	Description
31:16	Reserved	Reserved, must retain the reset value
15:0	EXTIx	EXTIx configuration（x=4…7） Select the input source of EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin

16.2.5 SYSCFG_EXTICR3 External Interrupt Configuration Register 3

Address offset: 0x10

Reset value: 0x0000 0000

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

Bit	Field	Description
31:16	Reserved	Reserved, must retain the reset value
15:0	EXTIx	EXTIx configuration（x=8…11） Select the input source of EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin

16.2.6 SYSCFG_EXTICR4 External Interrupt Configuration Register 4

Address offset: 0x014

Reset value: 0x0000 0000

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

Bit	Field	Description
31:16	Reserved	Reserved, must retain the reset value
15:0	EXTIx	EXTIx configuration（x=12…15） Select the input source of EXTIx external interrupt. 0000: PA[x] pin 0001: PB[x] pin

16.2.7 SYSCFG_PADHYS PAD Configuration Register

Address offset: 0x18

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.															I2C1_MODE_SEL
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.															

Bit	Field	Description
31:17	Reserved	Reserved, must retain the reset value
16	I2C1_MODE _SEL	I2C1 port mode selection bit 0：Open drain mode 1：Push pull mode Note: Push-pull mode is not recommended for I2C as slave; When the I2C is in push-pull mode as the host, the communicating slave cannot pull down the SCL
15:0	Reserved	Reserved, must retain the reset value

17. Device Electronic Signature

17.1 Overview

The Device electronic signature is stored in the system memory area in the Flash memory module, and can be read using the SWD or the CPU. It contains factory programd identification data that allow the user firmware or other external devices to automatically match its interface to the characteristics of the microcontroller.
The unique device identifier is ideally suited:

for use as serial numbers

for use as security keys in order to increase the security of code in Flash memory while using and combining this unique ID with software encryption-decryption algorithm before programming the internal Flash memory

to activate secure boot processes

The 96-bit unique device identifier provides a reference number which is unique for any microcontroller and in any context. These bits can never be altered by the user.

The 96-bit unique device identifier can also be read in single bytes (8 bits)/half-words (16 bits)/words (32 bits) in different ways.

17.2 Register description

Base address: 0x1FFF F7E8

Table 17-1 Device electronic signature register overview

Offset	Acronym	Register Name	Reset
0x00	UID1	Unique identifier	0xXXXXXXXX
0x04	UID2	Unique identifier	0xXXXXXXXX
0x08	UID3	Unique identifier	0xXXXXXXXX

17.2.1 UID1 Unique Identifier

Address offset：0x00
Reset value: This value is written by default.

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

Bit	Field	Description
31：0	U_ID（31：0）	U_ID：31：0 unique ID bits

17.2.2 UID2 Unique Identifier

Address offset：0x04

Reset value: This value is written by default.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U_ID （63：48）															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U_ID （47：32）															
r															

Bit	Field	Description
31：0	U_ID （63：32）	U_ID：63：32 unique ID bits

17.2.3 UID3 Unique Identifier

Address offset：0x08

Reset value: This value is written by default.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U_ID （95：80）															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U_ID （79：64）															
r															

Bit	Field	Description
31：0	U_ID（95：64）	U_ID：95：64 unique ID bits

18. DBG Debug Support

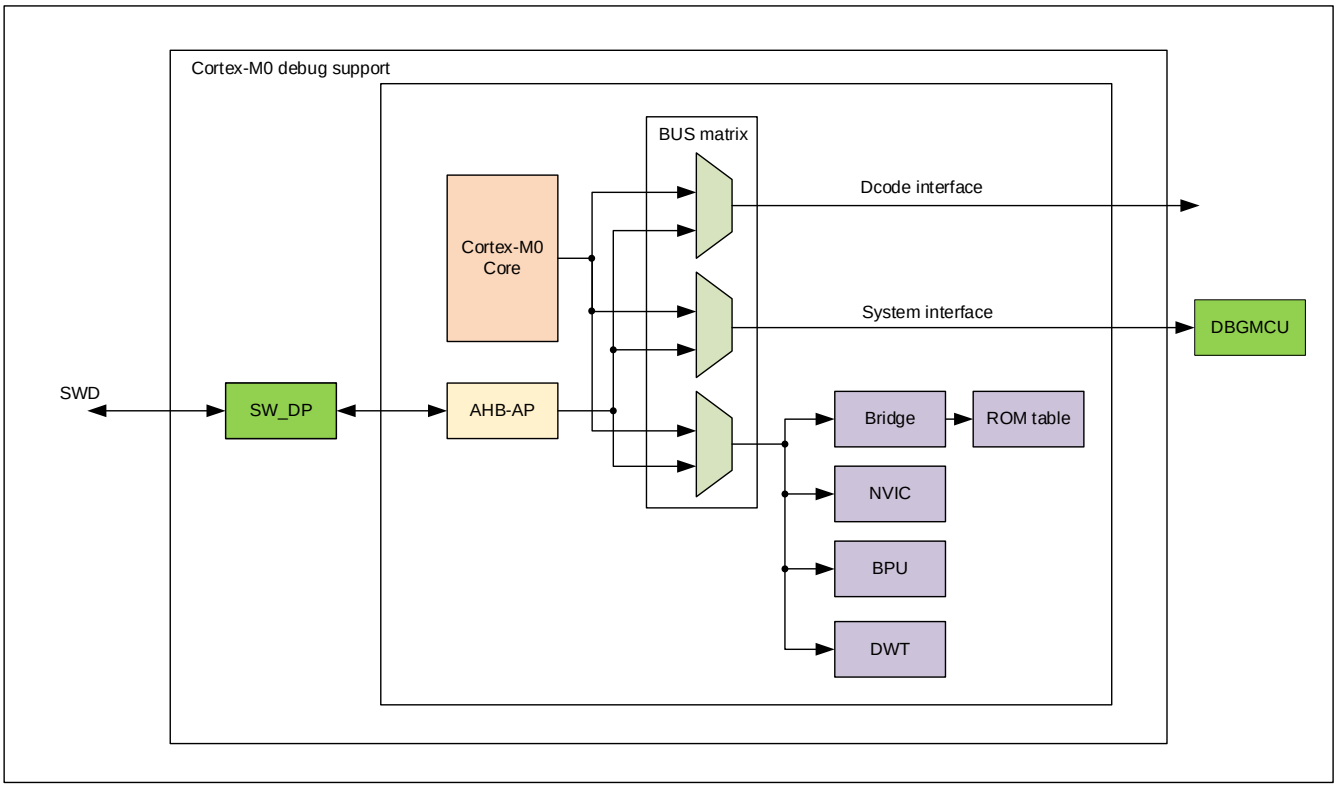
18.1 Introduction

The MCU core includes the debug module mainly used for the function commissioning. When the core gets address (address breakpoint) or accesses data (data breakpoint), the hardware debug module may control the core to stop. The user can enquire the core internal status and system status. After enquiry, the core may continue to execute the current program.
When the chip and debugger connect and begin debugging, the debugger auto calls the core debug module for debug operation.

18.2 Function descriptions

18.2.1 Function block diagram

Figure 18-1 Debug function block diagram



Cortex-M0 core includes the debug unit, and it's composed of:
SWDP: SW debug port
BPU:Breakpoint debug unit
DWT: Data watch point and trace

18.2.2 SWD internal pull up and down

SWD pin input directly controls the debug mode, and cannot be floated. To guarantee that I/O level is controllable, SWD pin has the built in pull up and down resistance.

SWDIO: Internal pull up

SWCLK: Internal pull down

The software can use these I/O ports as the ordinary I/O port. At this time, the pull up/down function closes by default. Refer to the chapter of General Purpose Input/output GPIO.

18.2.3 SWD debug port

The chip's two ordinary I/O ports are used as SWD-DP interface pin. These pins in different packages all support SWD debug port.

Table 18-1 SWD debug port pin

SWJ-DP port pin name	SW debug interface		Pin allocation
	Type	Debug function	
SWDIO	Input/output	Serial data input/output	PA13
SWCLK	Input	Serial clock	PA14

18.3 ID code

The chip has many ID codes, as seen below:

Table 18-2 ID code

ID Name	Chip
DEV_ID	0x4C505F00
CPU TAP SW ID	0x0BB11477

18.3.1 Microcontroller device ID code

The microcontroller contains the device ID code. This ID defines MCU chip version, and is mapped on the external APB bus. This code can be acquired by the user code and debug interface.

18.3.2 Cortex JEDEC-106 ID code

The microcontroller has a JEDEC-106ID code. It's located in the 4KB ROM table mapped to the internal PBB bus address 0xE00FF000_0xE00FFFFF

18.4 SW debug port

18.4.1 SW protocol introduction

This sync serial protocol uses 2 pins: Clock signal (SWCLK) and two-way data signal (SWDIO) from host to target.
As the two-way data line, SWDIO pin has the built in pull up resistor, and the extra external resistor is not required.
Data transmission begins from the low bit, and allow reading and writing the register DPACC and APACC.
According to the protocol, when SWDIO changes direction, insert one conversion time (one Bit time by default, adjustment by SWCLK). In this period, any device cannot drive the signal line.

18.4.2 SW protocol series

One series contains three phases:

The host sends 8 bit request packet;

The target sends 3 bit acknowledgement response;

According to the configuration direction, the host or target sends 33 bit (including one check bit) data;

Table 18-3 8bit request packet

Bit	Name	Description
0	Beginning	Must be 1
1	APnDP	0: Access DP 1: Access AP
2	RnW	0: Write request 1: Read request
4: 3	A[3 : 2]	DP or AP register address
5	Parity	Previous bit check bit
6	Stop	0
7	Park	Cannot be driven by host. Because of pull up, the target is always read as 1

Note: Each request packet is followed by 1 bit conversion time. For more information on DAPCC and APACC register, refer to the relevant CPU technology data sheet of ARM.

Table 18-4 3bit response package

Bit	Name	Description
2: 0	ACK	001: Fail 010: Wait 100: Success

Note：When the response (ACK) signal is one of scenarios above, the response bit is followed by one conversion time.

Table 18-5 33bit data packet

Bit	Name	Description
31: 0	WDATA/RDATA	Read or write data
32	Parity	32 bit data parity check bit

Note: Wait for one conversion time after read data bit.

18.4.3 SW-DP status unit (Reset, Idle states, ID code)

SW-DP status unit uses the internal ID code to identify SW_DP. Observe JEP-106 standard. For specific information, please refer to the relevant ARM manual.

SW-DP status unit doesn't work until before the debugger reads ID.

In case of power on reset, or DP switches from JTAG to SWD, or exceeds high level of 50 periods, SW-DP status unit is in the reset status;

If the low level of at least 2 periods appear after RESET status, the status unit will switch to the IDLE status;

When the status unit is in the reset status, first switch to the IDLE status, and read DP-SW ID register. Otherwise, the debugger cannot conduct other normal transmission, and ACK Fault will appear;

18.4.4 DP and AP read/write access

DP read operation has no delay: The debugger directly acquires data (in case of ACK return success status) or waits (in case of ACK return wait status);

AP read operation has latency. It means that the previous read operation result can only be acquired during the next operation. If the next operation is not access to AP, it's mandatory to read DP-RDBUFF register to acquire the previous read operation result;

DP-CTRL/STAT register READOK flag bit is refreshed after AP read operation and RDBUFF read operation to inform the debugger whether AP read operation succeeds;

SW-DP has the write buffer (DP and AP have the write buffer) so that the write operation can be accepted when other transmissions are ongoing. If the write buffer is full, the debugger will acquire one wait ACK response. Read IDCODE register, read CTRL/STAT register and write ABORT register

operation remain accepted when the write buffer is full;

Because SWCLK and HCLK are not in step, insert two extra SWCLK periods after the write operation (after parity check bit) to ensure that the internal write operation is correctly completed. These two extra clock periods should be inserted when the line is in IDLE status. This operation step is particularly important during writing CTRL/STAT register to propose one power on request. Otherwise, the next operation (valid operation after core power on) will execute immediately. It will lead to failure;

18.4.5 SW-DP register

In case of APnDP=0, access to these registers below.

Table 18-6 SW-DP register

A[3:2]	Read/write	SELECT register's CTRLSEL bit	Register	Description
00	Read		IDCODE	Fixed as 0x0BB11477（for identifying SW-DP）
00	Write		ABORT	
01	Read/Write	0	DP-CTRL/STAT	Request one system or debug power on operation; configure AP access operation mode; Control compare, check operation; Read some status bits (overflow, power on response).
01	Read/Write	1	WIRE CONTROL	Configure the serial communication physical layer protocol (such as conversion time length, etc.)
10	Read		READ RESEND	Allow to restore data from one error debug transmission rather than repeat the initial AP transmission.
10	Write		SELECT	Select current access port and valid 4 word register window.
11	Read/Write		READ BUFFER	This register will capture the data result of the previous read operation from AP. Therefore, the data can be acquired without again enabling the new AP transmission

18.4.6 SW-AP register

In case of APnDP=1, access and AP register access address is composed of two parts:

A[3：2] value

DP SELECT register's current value

18.5 MCU debug module (DBGMCU)

MCU debug module provides the debugger assist function below:

Support low power mode

Breakpoint timer and watchdog clock control

18.5.1 Debug support at low power mode

MCU has multiple low power modes. It can close CPU clock, reduce CPU power consumption, and enter the low power mode by executing WFE or WFI command. FCLK and AHB bus clock HCLK are mandatory for debug operation, and cannot be closed. MCU is equipped with some registers to change the low power mode characteristics, thus supporting the debug code in the low power mode. Specific configurations:

Enter the sleep mode. In order to make HCLK and FCLK have the same clock, the debugger must set the DBG_CR register DBG_SLEEP bit.

Enter the stop mode. It's mandatory to configure DBG_STOP bit. The operation will activate the internal oscillator HSI, thus providing the clock for FCLK and HCLK.

18.5.2 Support timer, watchdog

In case of the breakpoint, select the timer's work mode according to the application of timer and watchdog;

Counter count continues. The application is in the PWM wave control motor

Counter count stops. The application is in the watchdog counter.

18.6 Register

18.6.1 Register overview

Table 18-7 DBG register overview

Offset	Acronym	Register Name	Reset
0x00	DBG_IDCODE	DBG ID encode register	0x4C505F00
0x04	DBG_CR	DBG control register	0x00000000

18.6.2 DBG_IDCODE ID Encode Register

Address: 0x40013400 (only support 32-bit access, read only)
Reset value: 0x4C505F00

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

Bit	Field	Description
31:0	DEV_ID	Device Identifier Read only register, always read as reset value

18.6.3 DBG_CR Control Register

Address: 0x40013404 (only support 32 bit access)
Reset value: 0x0000 0000(POR reset only, not reset by system reset)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.													DBG_TIM14_STOP	Res.	
													rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG_TIM3_PWM_OFF	Res.	DBG_TIM1_PWM_OFF	DBG_TIM3_STOP	Res.	DBG_TIM1_STOP	Res.	DBG_IWDG_STOP	Res.				DBG_STOP_FOR_LDO	Res.	DBG_STOP	DBG_SLEEP
rw		rw	rw		rw		rw					rw		rw	rw

Bit	Field	Description
31:19	Reserved	Reserved, must retain the reset value
18	DBG_TIM14_STOP	TIM14 stops work when the core enters the debug mode 0: TIM14 still works normally when the timer is selected 1: TIM14 stops when the timer is selected
17:16	Reserved	Reserved, must retain the reset value
15	DBG_ TIM3_PWM_OFF	TIM3 PWM outputs 0 when the core enters the debug mode 0: TIM3 PWM normal outputs 1: TIM3 PWM outputs 0
14	Reserved	Reserved, must retain the reset value

13	DBG_TIM1_ PWM_OFF	TIM1 PWM outputs 0 when the core enters the debug mode 0: TIM1 PWM normal outputs 1: TIM1 PWM outputs 0
12	DBG_TIM3_STOP	TIM3 stops work when the core enters the debug mode 0: TIM3 still works normally when the timer is selected 1: TIM3 stops when the timer is selected
11	Reserved	Reserved, must retain the reset value
10	DBG_TIM1_STOP	TIM1 stops work when the core enters the debug mode 0: TIM1 still works normally when the timer is selected 1: TIM1 stops when the timer is selected
9	Reserved	Reserved, must retain the reset value
8	DBG_IWDG_STOP	Independent watchdog stops work This bit is not related to whether the core enters the debug status 0: The watchdog counter still works normally 1: The watchdog counter stops work
7:4	Reserved	Reserved, must retain the reset value
3	DBG_STOP_FOR_LDO	Debug stop mode LDO status 0: LDO enters the low power mode, and normally enters the STOP mode 1: LDO doesn't enter the low power mode, PLL sustains power on, and cannot truly enter STOP mode. CPU enters DEEPSLEEP, and HCLK closes
2	Reserved	Reserved, must retain the reset value
1	DBG_STOP	Debug stop mode 0: In the stop mode, the clock controller disables all clocks (including HCLK and FCLK). When exiting from STOP mode, the clock configuration is consistent with the configuration after reset. If it's required to again enable PLL, HSE clock, the software must reconfigure the clock control system. 1: In the stop mode, FCLK and HCLK open, and HSI provides the clock. When exiting from STOP mode, make reconfiguration by software if it's required to again enable PLL, HSE clock
0	DBG_SLEEP	Debug sleep mode 0: In the sleep mode, clock FCLK opens, FCLK keeps the configured system clock by default, and HCLK closes. The sleep mode won't reset the configured clock system. When exiting the sleep mode, the software doesn't need to reconfigure the system clock 1: In the sleep mode, FCLK and HCLK clock is provided by the formally configured system.

19. History Records

Table 19-1 History records

Date	Version	Content
2024/10/11	1.0	Initial Version
2025/01/03	1.1	Modified block diagrams