User Manual

MM32SPIN05x

32-bit Microcontroller Based on ARM Cortex M0 Core

Version: 1.19_q

Table of Contents

Figures

Table

1 Memory and bus architecture

Memory and bus architecture

1.1 System architecture

The main system consists of the following parts:

- Two drive units:
  - CPU system bus(S-bus)
  - Generic DMA
- Three passive units:
  - Internal SRAM
  - Internal flash memory
  - AHB to APB bridges(APBx), connecting all AHB devices

They are interconnected through a multi-stage AHB architecture, as shown in Figure 1.

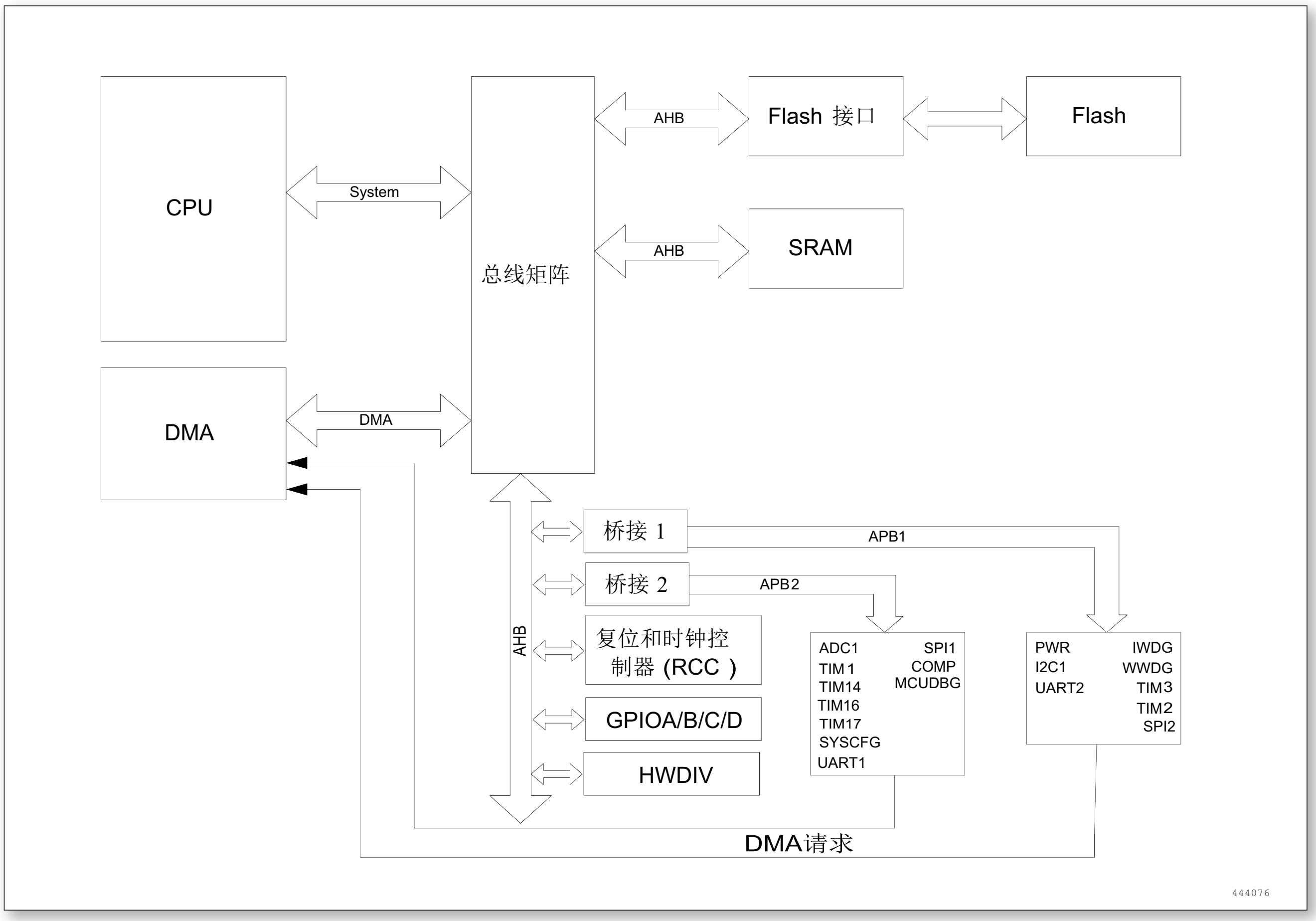


Figure 1. System architecture

System bus

The bus connects the system bus (peripheral bus) of the CPU core to the BusMatrix, which manages the access between the core and DMA.

DMA bus

The bus connects the AHB master interface of DMA with the BusMatrix, which manages the access of CPU and DMA to SRAM, flash memory and peripherals.

BusMatrix

The BusMatrix, composed of master module bus and slave module bus, manages the access arbitration between the core system bus and the DMA bus.

AHB peripherals are connected on system bus through a BusMatrix to allow DMA access.

AHB2APB bridges - APB

The AHB-APB bridges provide the synchronous connections between the AHB and APB bus. After each device reset, all peripheral clocks are disabled (except for the SRAM and Flash). Before using a peripheral, you have to enable its clock in the RCC_AHBENR, RCC_APB2ENR or RCC_APB1ENR register.

note:When a 16- or 8-bit access is performed on an APB register, the access is transformed into a 32-bit access: the bridge duplicates the 16- or 8-bit data to feed the 32-bit vector.

1.2 Memory organization

1.2.1 Introduction

Program memory, data memory, registers and I/O ports are organized within the same linear 4-Gbyte address space.

The bytes are coded in memory in Little Endian format. The lowest numbered byte in a word is considered as the word's least significant byte and the highest numbered byte the most significant.

The addressable memory space is divided into 8 main blocks, each of 512 MB. All the memory areas that are not allocated to on-chip memories and peripherals are considered "Reserved". Refer to the section 1.2.2 and Peripherals.

1.2.2 Memory map and register addressing

See the memory map in Peripherals chapters.The following table gives the start addresses of the embedded peripherals.

Table 1. Memory Map

Bus	Addressing range	Size	Peripheral	Remarks
	0x0000 0000 - 0x0001 FFFF	128 KB	Main flash memory, system memory or SRAM, depending on BOOT configuration	
	0x0002 0000 - 0x07FF FFFF	~ 128 MB	Reserved	
Flash	0x0800 0000 - 0x0800 7FFF	32 KB	Main Flash memory	

Bus	Addressing range	Size	Peripheral	Remarks
	0x0800 8000 - 0x1FFD FFFF	~ 256 MB	Reserved	
	0x1FFE 0000 - 0x1FFE 01FF	0.5 KB	Reserved	
	0x1FFE 0200 - 0x1FFE 0FFF	3 KB	Reserved	
	0x1FFE 1000 - 0x1FFE 1BFF	3 KB	Reserved	
	0x1FFE 1C00 - 0x1FFF F3FF	~ 256 MB	Reserved	
Flash	0x1FFF F400 - 0x1FFF F7FF	1 KB	System memory	
	0x1FFF F800 - 0x1FFF F80F	16 B	Option bytes	
	0x1FFF F810 - 0x1FFF FFFF	~ 2 KB	Reserved	
SRAM	0x2000 0000 - 0x2000 0FFF	4 KB	SRAM	
	0x2000 1000 - 0x2FFF FFFF	~ 512 MB	Reserved	
APB1	0x4000 0000 - 0x4000 03FF	1 KB	TIM2	
	0x4000 0400 - 0x4000 07FF	1 KB	TIM3	
	0x4000 0800 - 0x4000 27FF	8 KB	Reserved	
	0x4000 2800 - 0x4000 2BFF	1 KB	Reserved	
	0x4000 2C00 - 0x4000 2FFF	1 KB	WWDG	
	0x4000 3000 - 0x4000 33FF	1 KB	IWDG	
	0x4000 3400 - 0x4000 37FF	1 KB	Reserved	
	0x4000 3800 - 0x4000 3BFF	1 KB	SPI2	
	0x4000 4000 - 0x4000 43FF	1 KB	Reserved	
	0x4000 4400 - 0x4000 47FF	1 KB	UART2	
	0x4000 4800 - 0x4000 4BFF	3 KB	Reserved	
	0x4000 5400 - 0x4000 57FF	1 KB	I2C1	
	0x4000 5800 - 0x4000 5BFF	1 KB	Reserved	
	0x4000 5C00 - 0x4000 5FFF	1 KB	Reserved	
	0x4000 6000 - 0x4000 63FF	1 KB	Reserved	
	0x4000 6400 - 0x4000 67FF	1 KB	Reserved	
	0x4000 6800 - 0x4000 6BFF	1 KB	Reserved	
	0x4000 6C00 - 0x4000 6FFF	1 KB	Reserved	
	0x4000 7000 - 0x4000 73FF	1 KB	PWR	
	0x4000 7400 - 0x4000 FFFF	35 KB	Reserved	
APB2	0x4001 0000 - 0x4001 03FF	1 KB	SYSCFG	
	0x4001 0400 - 0x4001 07FF	1 KB	EXTI	
	0x4001 0800 - 0x4001 23FF	7 KB	Reserved	
	0x4001 2400 - 0x4001 27FF	1 KB	ADC1	
	0x4001 2800 - 0x4001 2BFF	1 KB	Reserved	
	0x4001 2C00 - 0x4001 2FFF	1 KB	TIM1	
	0x4001 3000 - 0x4001 33FF	1 KB	SPI1	
	0x4001 3400 - 0x4001 37FF	1 KB	DBGMCU	
	0x4001 3800 - 0x4001 3BFF	1 KB	UART1	
	0x4001 3C00 - 0x4001 3FFF	1 KB	COMP	
	0x4001 4000 - 0x4001 43FF	1 KB	TIM14	

Bus	Addressing range	Size	Peripheral	Remarks
	0x4001 4400 - 0x4001 47FF	1 KB	TIM16	
	0x4001 4800 - 0x4001 4BFF	1 KB	TIM17	
	0x4001 4C00 - 0x4001 7FFF	13 KB	Reserved	
AHB	0x4002 0000 - 0x4002 03FF	1 KB	DMA	
	0x4002 0400 - 0x4002 0FFF	3 KB	Reserved	
AHB	0x4002 1000 - 0x4002 13FF	1 KB	RCC	
	0x4002 1400 - 0x4002 1FFF	3 KB	Reserved	
	0x4002 2000 - 0x4002 23FF	1 KB	Flash Interface	
	0x4002 2400 - 0x4002 5FFF	15 KB	Reserved	
	0x4002 6000 - 0x4002 63FF	1 KB	Reserved	
	0x4002 6400 - 0x4002 FFFF	39 KB	Reserved	
	0x4003 0000 - 0x4003 03FF	1 KB	HWDIV	
	0x4003 0400 - 0x47FF FFFF	~ 128 MB	Reserved	
	0x4800 0000 - 0x4800 03FF	1 KB	GPIOA	
	0x4800 0400 - 0x4800 07FF	1 KB	GPIOB	
	0x4800 0800 - 0x4800 0BFF	1 KB	GPIOC	
	0x4800 0C00 - 0x4800 0FFF	1 KB	GPIOD	
	0x4800 1000 - 0x5FFF FFFF	~ 384 MB	Reserved	

1.3 Embedded SRAM

It features up to 2 K bytes of static SRAM.

It features up to 4K bytes of static SRAM.

It can be accessed as bytes, half-words (16 bits) or full words (32 bits). The SRAM start address is 0x2000 0000.

- SRAM up to 2K bytes on the data bus. Can be used by the CPU or DMA with the fastest system clock and without any waiting to access.

1.4 Overview of FLASH memory

The flash memory includes two different storage areas:

- The main block includes the program data area and the user data area (if necessary)
- The information block includes four parts:
  - Option bytes - Containing hardware and user storage protection configuration options.
  - System memory - Containing boot loader code. See Section ”Embedded Flash Memory”.

The flash memory interface, based on AHB protocol, executes instructions and accesses data. With the prefetch buffer function, it accelerates the code execution speed of CPU.

1.5 Boot configuration

3 different boot modes can be selected through BOOT0 pins and nBOOT1 bit, as shown in the following table.

Table 2. Boot Modes

Boot mode selection		**Boot mode**	**Aliasing**
nBOOT1	**BOOT0**		
x	0	Main flash memory	Main flash memory is selected as boot space
0	1	System memory	System memory is selected as boot space
1	1	Embedded SRAM	Embedded SRAM is selected as boot space

The values on the BOOT pins are latched on the 4th rising edge of SYSCLK after a reset. It is up to the user to set the BOOT1 and BOOT0 pins after Reset to select the required boot mode.

The BOOT pins are also re-sampled when waking up from Standby mode. Consequently, they must be kept in the required Boot mode configuration in Standby mode.

After this startup delay has elapsed, the CPU fetches the top-of-stack value from address 0x00000000, then starts code execution from the boot memory starting from 0x00000004.

Depending on the selected boot mode, main Flash memory, system memory or SRAM is accessible as follows:

- Boot from main Flash memory: the main Flash memory is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x0800 0000). In other words, the Flash memory contents can be accessed starting from address 0x0000 0000 or 0x0800 0000.
- Boot from system memory: the system memory is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x1FFF F400).
- Boot from the embedded SRAM: SRAM is aliased in the boot memory space (0x0000 0000), but still accessible from its original memory space (0x2000 0000).

Embedded boot loader

The embedded boot loader is located in the System memory, programmed by the manufacturer during production. It is used to reprogram the Flash memory with UART1.

2 Embedded flash(FLASH)

Embedded flash(FLASH)

2.1 Main features

- Up to 32K bytes of flash memory

The flash memory interface features:

- Data interface with prefetch buffer (2x64-bit)
- Option byte Loader
- Flash program/erase operation
- Read / write protection
- Low power mode

2.2 Functional description

2.2.1 Structure

The flash space consists of 64-bit memory cells, in which both code and data can be saved. The main block is divided into 32 pages (1 K bytes per page) or 8 sectors (4 K bytes per sector), and the write protection is set in sectors (see related content of ”Storage Protection”).

Table 3. Flash Module Structure

Module	Name	Address	Size(bytes)
Main memory	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 28	0x0800 7000 - 0x0800 73FF	1K
	Page 29	0x0800 7400 - 0x0800 77FF	1K
	Page 30	0x0800 7800 - 0x0800 7BFF	1K
	Page 31	0x0800 7C00 - 0x0800 7FFF	1K
Information block	Guard bytes	0x1FFE 0000 - 0x1FFE 01FF	0.5K
	Secrecy space	0x1FFE 1000 - 0x1FFE 1BFF	3K
	System memory	0x1FFF F400 - 0x1FFF F7FF	1K
	Option bytes	0x1FFF F800 - 0x1FFF F80F	16

Module	Name	Address	Size(bytes)
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
	FLASH_WRPR	0x4002 2020 - 0x4002 2023	4

2.2.2 Reading flash

Embedded flash modules, like common storage space, can be directly addressed and accessed. The operation of reading flash module shall undergo a special judgment process.

Both instruction fetch and data fetch are performed through AHB bus and can be executed in the manner specified by the option in the flash access control register (FLASH_ACR):

- Instruction fetch: CPU running speed can be increased after the prefetch buffer is enabled
- Latency: number of wait states for a correct read operation

Instruction fetch

Instructions are fetched by CPU through AHB. With the prefetch module, the instruction fetch efficiency is improved.

Prefetch buffer

Prefetch buffer (2x64-bit): It is automatically opened after reset. Since the size of each buffer (64-bit) is the same as the bandwidth of the flash, the contents of the entire buffer can be updated only by reading flash memory once. Due to the existence of the prefetch buffer, the CPU can run at a higher dominant frequency. In each time, the CPU fetches a word up to 32 bits; the next instruction is waiting in the buffer while one instruction is being fetched.

Prefetch controller

The prefetch controller will timely access the flash according to the available space in the prefetch buffer. In case of at least one available space in the prefetch buffer, the prefetch controller will initiate a read request. After reset, the prefetch buffer is opened by default, and can only be enabled/disabled if SYSCLK is lower than 24 MHz and the AHB clock has not undergone any frequency division (SYSCLK must be equal to HCLK). Usually, prefetch buffer has already determines its on/off state during the initialization process. At this time, MCU is running under the 8 MHz oscillator.

Note: When the prescaler of AHB clock is not equal to 1, the prefetch buffer shall initiate the access latency.

Access latency

In order to ensure the correct flash reading, the speed ratio of prefetch controller shall be specified in LATENCY 2: 0 in the flash access control register, and it is equal to the number of latent periods to be inserted between each flash accessing to the next access. After reset, this value defaults to zero, namely, there is no latent period to be inserted.

2.2.3 Programming and erasing flash

The embedded flash supports online programming and in-application programming.

ICP refers to rewriting flash online through SWD and burning the user code into the MCU. ICP provides a simple and efficient method, not requiring chip clamping during its writing.

Unlike ICP, IAP (in-application programming) enables downloading programs or data through any communication interface (I/Os, USB, UART, I2C, SPI, etc.) supported by MCU. IAP allows users to rewrite applications while running them, provided that some applications must be burned in advance by ICP/ISP.

The burn-in and erase operations can be completed within the whole operating voltage range of the product, and they are achieved with the following 7 registers:

- Key register (FLASH_KEYR)
- Option-byte key register (FLASH_OPRKEYR)
- 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)

As long as the CPU does not access the Flash space, the ongoing Flash programming will not hinder the operation of the CPU. That is to say, in the process of programming/erasing Flash, any access to Flash will disable the bus and it will not resume until the programming/erasing operation is completed, which means that instruction fetch and data access cannot be performed in case of Flash programming/erasing.

In the process of programming/erasing Flash space, the internal oscillator (HSI) must be on.

Unlocking Flash space

After reset, Flash memory is protected by default, preventing accidental erasing. The FLASH_CR is not allowed to be rewritten and the access to the FLASH_CR will not be authorized unless a series of unlocking operations for the FLASH_KEYR are performed. These operations include the following 2 write operations:

- Write key 1 = 0x45670123
- Write key 2 = 0xCDEF89AB

Any wrong sequence will lock FLASH_CR until the next reset.

In case of invalid keyword, a bus error will cause a hardware error interrupt; in case of KEY1 error, it will immediately interrupt, while a correct KEY1 will also lead to an interrupt when KEY2 error occurs.

Main flash programming

The 16-bit main flash can be programmed at a time. When PG bit in FLASH_CR is 1, writing a half words (16 bits) corresponding to the address is a programming operation. If you try to write another length instead of half words, the operation will cause a hardware error interrupt.

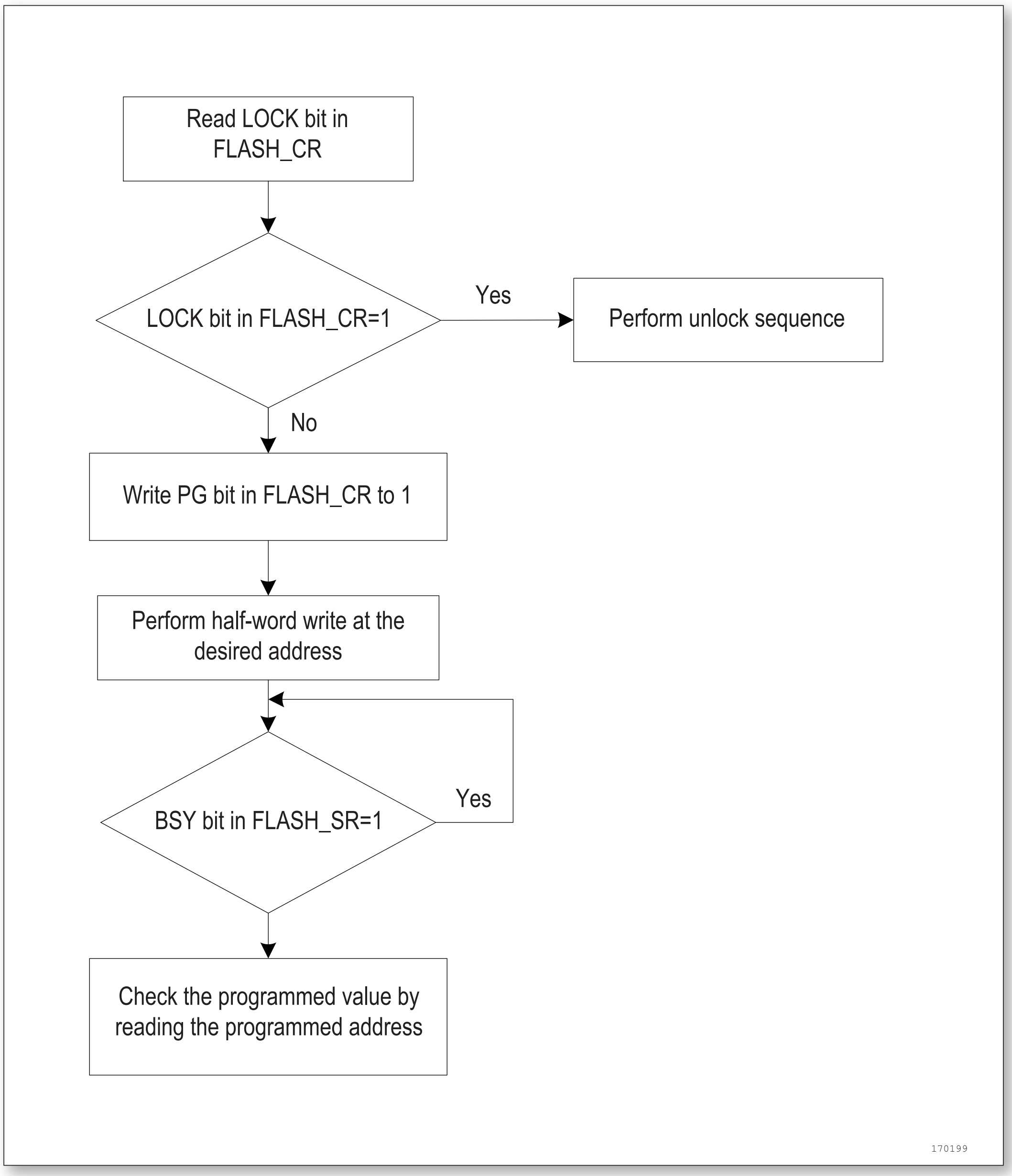

Read LOCK bit in FLASH_CR
LOCK bit in FLASH_CR=1
Yes
Perform unlock sequence
No
Write PG bit in FLASH_CR to 1
Perform half-word write at the desired address
BSY bit in FLASH_SR=1
Yes
Check the programmed value by reading the programmed address
170199

Figure 2. Programming Flow

The Flash memory interface will pre-read and judge whether the bytes behind those to be programmed are all 1s. If not, the programming operation will be automatically cancelled and an error warning will be prompted on the PGERR bit of the FLASH_SR.

If the write protection bit in FLASH_WRPR, corresponding to the address to be programmed, is valid, the programming will be disabled, and an error warning will be also generated. In addition, a prompt will be given at the EOP bit in FLASH_SR after the programming.

The programming process in the standard mode of the main Flash memory is as follows:

- Check BSY bit in FLASH_SR, to confirm that the previous operation has ended
- Set PG bit in FLASH_CR
- Write data to the target address in half words
- Wait for BSY in FLASH_SR to return to zero
- Read data for validation

Note: these registers cannot be written when the BSY bit in FLASH_SR is 1.

Erasing Flash memory

Flash memory can be erased by pages or whole pieces.

Page erasing

The specific procedures are as follows:

- Check BSY bit in FLASH_SR, to confirm that the previous operation has ended
- Set the PER bit in the FLASH_CR to 1
- Write the FLASH_AR, to select the page to be erased
- Set the STRT bit in the FLASH_CR to 1
- Wait for BSY in FLASH_SR to return to zero
- Read the erased pages for validation

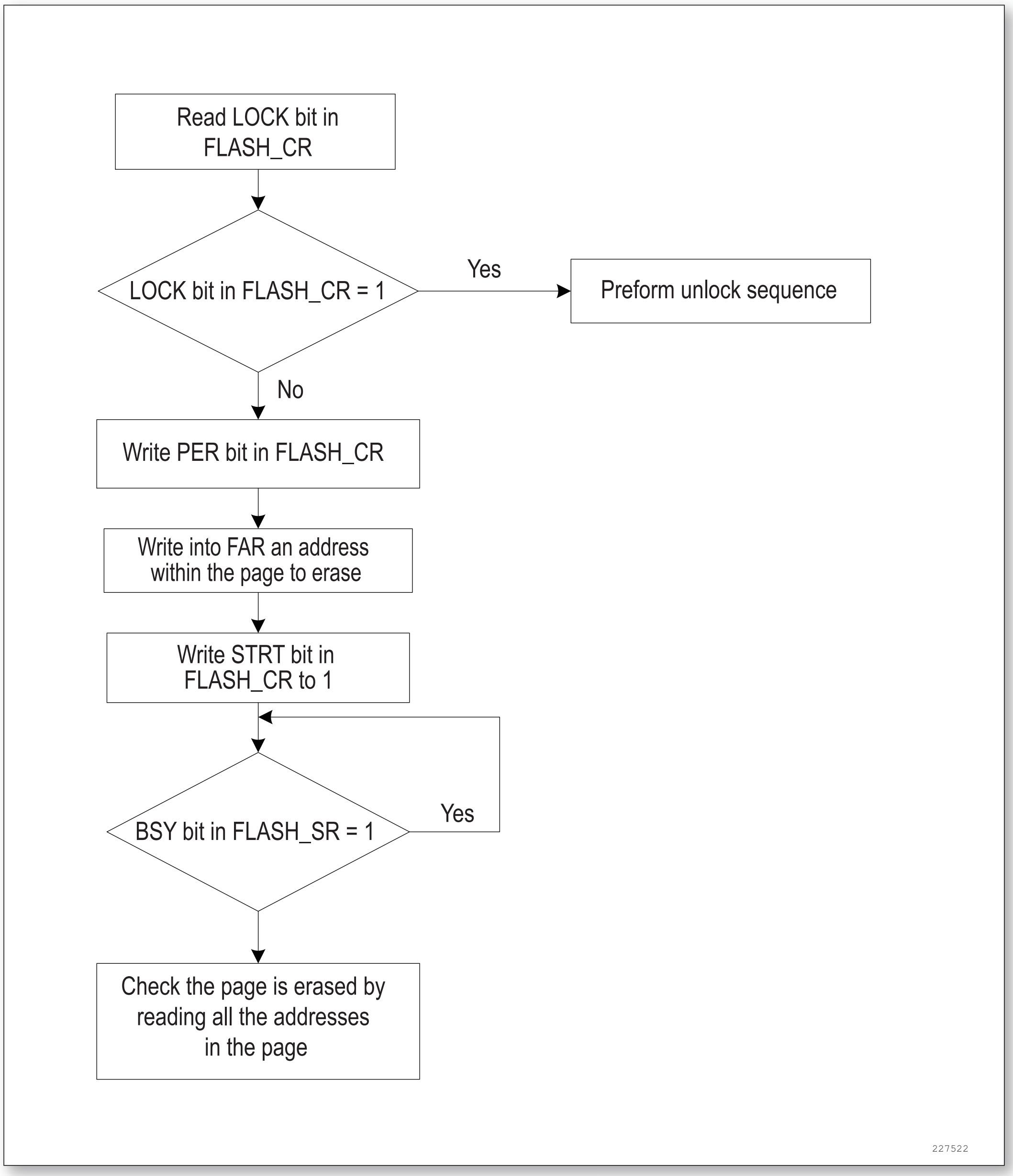

Read LOCK bit in FLASH_CR
LOCK bit in FLASH_CR = 1
Yes
Preform unlock sequence
No
Write PER bit in FLASH_CR
Write into FAR an address within the page to erase
Write STRT bit in FLASH_CR to 1
BSY bit in FLASH_SR = 1
Yes
Check the page is erased by reading all the addresses in the page
227522

Figure 3. Page Erasing Process of Flash Register

Whole erasing

The entire Flash user area can be erased at one time by using the whole erasing command, and the information block will not be affected by this command. The specific steps are as follows:

- Check BSY bit in FLASH_SR, to confirm that the previous operation has ended
- Set the MER bit in the FLASH_CR to 1
- Set the STRT bit in the FLASH_CR to 1
- Wait for BSY bit to return to zero
- Read and validate all pages

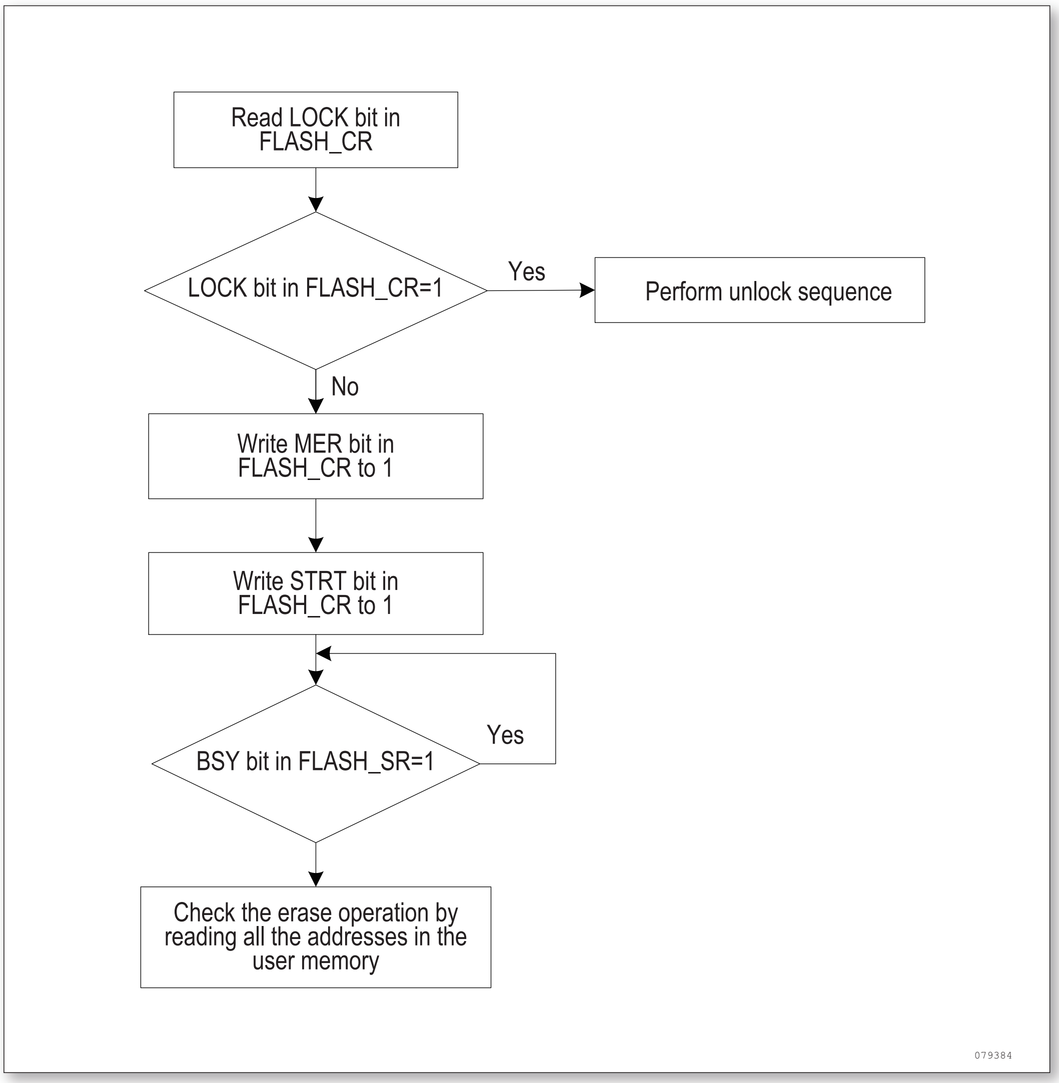

Read LOCK bit in FLASH_CR
LOCK bit in FLASH_CR=1
Yes
Perform unlock sequence
No
Write MER bit in FLASH_CR to 1
Write STRT bit in FLASH_CR to 1
BSY bit in FLASH_SR=1
Yes
Check the erase operation by reading all the addresses in the user memory
079384

Figure 4. Whole Erasing Process of Flash Register

Option bytes programming

The programming of option bytes is different from that for the conventional user address, including 2 write protections and 1 hardware configuration. After the Flash access restriction is lifted, the FLASH_OPTKEYR needs to be written with keywords. After that, the OPTWRE bit in the FLASH_CR will be set to ’1’, then the OPTPG bit in the FLASH_CR can be set first, and then the target address can be written in half words. Similarly, it will automatically check if the option byte is 1. If not, the relevant operation will be cancelled and an error will be prompted at the WRPRTERR bit in FLASH_SR. After the programming, a prompt will be given at the EOP bit of FLASH_SR.

The option byte is 16-bit data, the valid data is the lower-8-bit, and the upper 8 bits are the inverse of the lower 8 bits. In the programming process, the hardware will automatically set the upper 8 bits to the inverse code of the lower 8 bits, to ensure that the write value of the option byte is always correct. The steps are as follows:

- Check BSY bit in FLASH_SR, to confirm that the previous operation has ended
- Unlock OPTWRE bit in FLASH_CR

- Set the OPTPG bit in the FLASH_CR to 1
- Write data (half word) to the target address
- Wait for BSY bit to return to zero
- Read and validate that a whole erasing will be automatically triggered when the protection option byte is changed from the protected state to the unprotected state. If the user only wants to rewrite other bytes, the whole erasing will not be activated. This mechanism is used to protect the Flash.

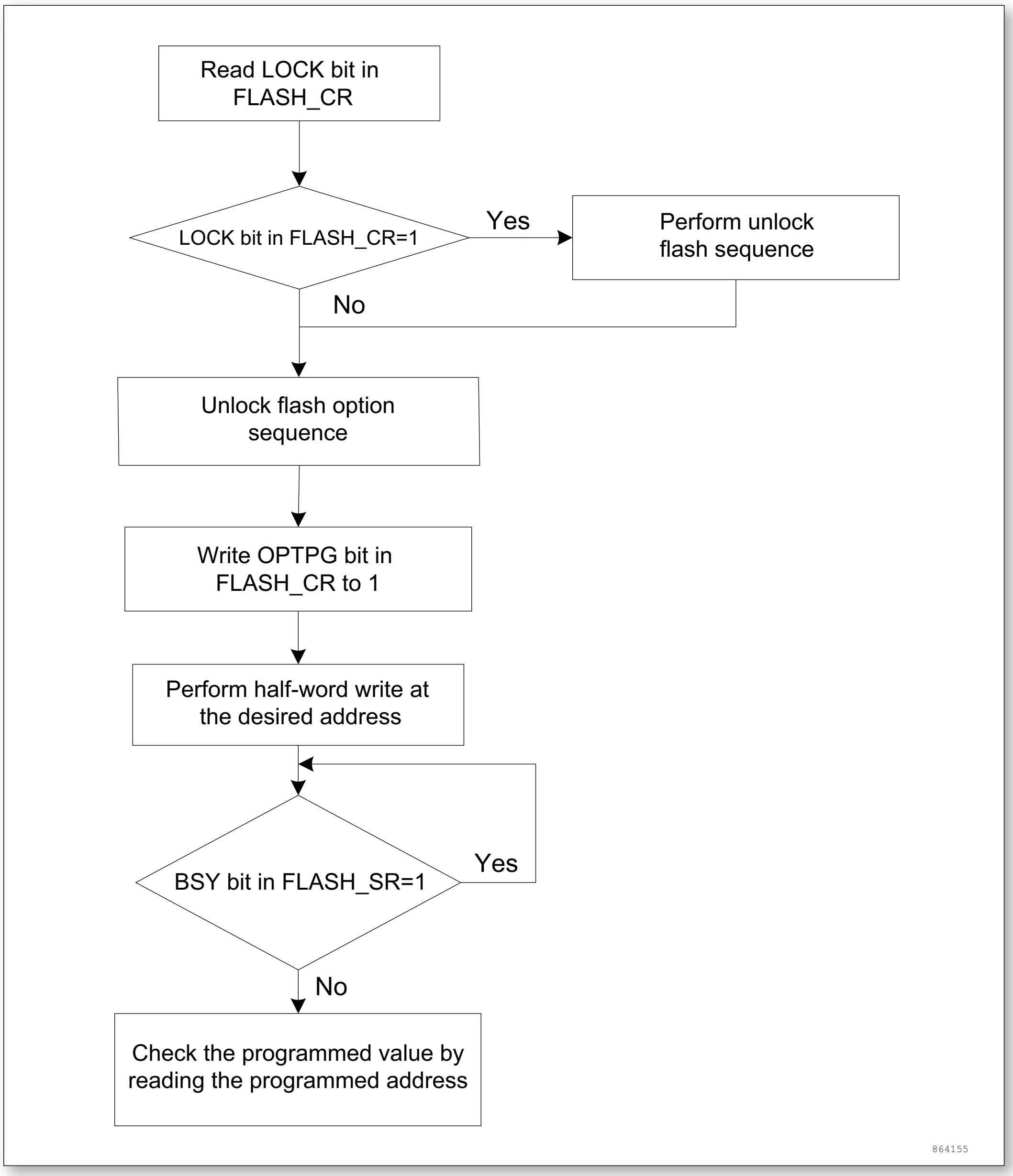

Read LOCK bit in FLASH_CR
LOCK bit in FLASH_CR=1
Yes
Perform unlock flash sequence
No
Unlock flash option sequence
Write OPTPG bit in FLASH_CR to 1
Perform half-word write at the desired address
BSY bit in FLASH_SR=1
Yes
No
Check the programmed value by reading the programmed address
864155

Figure 5. Option Byte Programming Process

Erasing process

The erasing process of option bytes is as follows:

- Check BSY bit in FLASH_SR, to confirm that the previous operation has ended
- Unlock OPTWRE bit in FLASH_CR

- Set OPTER bit in FLASH_CR to 1
- Set the STRT bit in the FLASH_CR to 1
- Wait for BSY bit to return to zero
- Read and validate

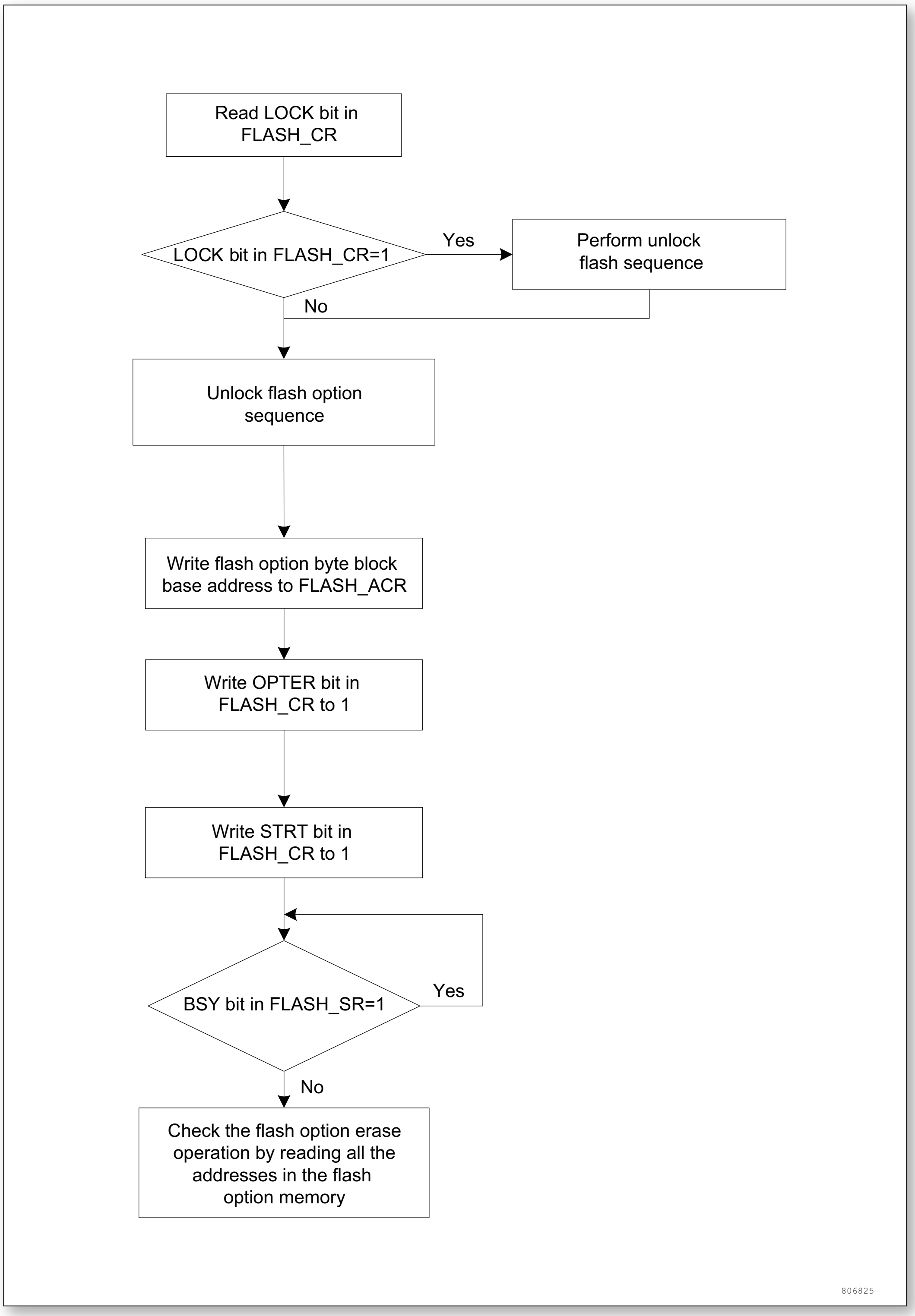

Read LOCK bit in FLASH_CR
LOCK bit in FLASH_CR=1
Yes
Perform unlock flash sequence
No
Unlock flash option sequence
Write flash option byte block base address to FLASH_ACR
Write OPTER bit in FLASH_CR to 1
Write STRT bit in FLASH_CR to 1
BSY bit in FLASH_SR=1
Yes
No
Check the flash option erase operation by reading all the addresses in the flash option memory
806825

Figure 6. Option Byte Erasing Process

2.3 Storage protection

It is used to prevent the codes in the Flash area of the user area from being read by untrusted codes, and to prevent accidental erasure of the Flash in case of running-out. The minimum unit of write protection is one sector (4 pages).

2.3.1 Write protection of main space

The write protection is controlled by one sector (4 pages), to configure the WRP bit in the option byte, and the subsequent system reset will load the new option byte, to enable the protection. If an attempt is made to write or erase a protected sector, the WRPRTERR flag bit in FLASH_SR will be set.

Unclocking

It is applicable to the startup program realized and programmed by the user:

- Erase the entire option byte area by using the OPTER bit of the flash control register (FLASH_CR);
- Reset the system and reload option bytes (including new WRP bytes), to unlock the write protection.

With this method, unlock the write protection of the entire main flash module, excepting Pages 0-3 which are still protected.

2.3.2 Write protection of option bytes

By default, the option byte block is always readable and write protected. To write (programming/erasing) an option byte block, write the correct key sequence in the OPTKEYR (the same as the locking operation), then enable the write operation to the option byte block. Note that the OPTWRE bit in the FLASH_CR indicates the write permission, and disable the write operation by clearing this bit.

2.4 Flash interrupt

Table 4. Flash Interrupt Request

Interrupt event	Event flag	Enable control bit
End of operation	EOP	EOPIE
Write protection error	WRPRTERR	ERRIE
Programming error	PGERR	ERRIE

2.5 Description of option bytes

Option bytes are configured by the user according to the application requirements, for example, a hardware-based watchdog or a software-based watchdog can be selected.

Each 32-bit word in the option byte is classified into the following formats:

Table 5. Option Byte Format

Bits 31 ~ 24	**Bits 23 ~ 16**	**Bits 15 ~ 8**	**Bits 7 ~ 0**
Inversion of option byte 1	Option byte 1	Inversion of option byte 0	Option byte 0

Note: The inversed code is automatically achieved by hardware, and cannot be written through the software.

The organization of option bytes in the option byte block is as shown in the following table.

Option bytes can be read from the memory addresses listed in the following table or from the option byte register (FLASH_OBR).

Note: The newly-written option byte (user's or read/write-protected) will not take effect until the system is reset.

Table 6. Structure of Option Bytes

Address	**[31: 24]**	**[23: 16]**	**[15: 8]**	**[7: 0]**
0x1FFF F800	nUSER	USER		
0x1FFF F804	nData1	Data1	nData0	Data0
0x1FFF F808	nWRP1	WRP1	nWRP0	WRP0
0x1FFF F80C	nWRP3	WRP3	nWRP2	WRP2

Table 7. Description of Option Bytes

Memory address	**Option bytes**
0x1FFF F800	Bit[31: 24] nUSER Bit[23: 16] USER: user option byte (saved in FLASH_OBR[9: 2]). It is used to configure the following functions: Select watchdog event: hardware or software Note: Only use Bits [20] and [18: 16], and do not use Bits [23: 21] and [19]. Bit 20: nBOOT1 Bit 18: nRST_STDBY 0: Reset when entering standby mode 1: Do no reset when entering standby mode Bit 17: nRST_STOP 0: Reset when entering STOP mode 1: Do not reset when entering STOP mode Bit 16: WDG_SW 0: Hardware watchdog 1: Software watchdog

Memory address	Option bytes
0x1FFF F804	Datax: 2-byte user data This address can be programmed through the programming methods for option bytes. Bits [31: 24]: nData1 Bits [23: 16]: Data1(saved in FLASH_OBR[25: 18]) Bits [15: 8]: nData0 Bits [7: 0]: Data0 (saved in FLASH_OBR[17: 10])
0x1FFF F808	WRPx: Flash write protection for option bytes Bits [31: 24]: nWRP1 Bits [23: 16]: WRP1(saved in FLASH_WRPR[15: 8]) Bits [15: 8]: nWRP0 Bits [7: 0]: WRP0(saved in FLASH_WRPR[7: 0])
0x1FFF F80C	WRPx: Flash write protection for option bytes Bits [31: 24]: nWRP3 Bits [23: 16]: WRP3(saved in FLASH_WRPR[31: 24]) Bits [15: 8]: nWRP2 Bits [7: 0]: WRP2(saved in FLASH_WRPR[23: 16]) Each bit in the option byte WRPx is used to protect 4 memory pages in the main memory: 0: Write protection is enabled 1: Write protection is disabled Four user option bytes are used to protect the main memory of 128 K bytes. WRP0: write protection on Pages 0 ~ 31 WRP1: write protection on Pages 32 ~ 63 WRP2: write protection on Pages 64 ~ 95 WRP3: write protection on Pages 96 ~ 127

After each system reset, the option byte loader (OBL) reads the data of the information block and saves it in the option byte register (FLASH-OBR); each select bit has its inverse code bit in the information block. When the select bit is loaded, the inverse code bit is used to validate whether the select bit is correct. In case of any difference, an option byte error flag (OPTERR) will be generated. When an option byte error occurs, the corresponding option byte is set to 0xFF. When the option byte and its inverse code are 0xFF (erased state), the above verification function is disabled.

All the select bits (excluding their inversed code bits) are used to configure the microcontroller, and the option byte register is read by CPU.

2.6 Description of Flash register

Table 8. Overview of Flash Register

Offset	Acronym	Register Name	Reset	Section
0x00	FLASH_ACR	Flash access control register	0x00000018	section 2.6.1
0x04	FLASH_KEYR	FPEC key register	0xXXXXXXXX	section 2.6.2
0x08	FLASH_OPTKEYR	Flash OPTKEY register	0xXXXXXXXX	section 2.6.3
0x0C	FLASH_SR	Flash status register	0x00000000	section 2.6.4
0x10	FLASH_CR	Flash control register	0x00000080	section 2.6.5
0x14	FLASH_AR	Flash address register	0x00000000	section 2.6.6
0x1C	FLASH_OBR	Option byte register	0x03FFFC1C	section 2.6.7
0x20	FLASH_WRPR	Write protection register	0xFFFFFFFF	section 2.6.8

2.6.1 Flash access control register(FLASH_ACR)

Address offset: 0x00

Reset value: 0x0000 0018

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											PRFTBE	HLFCYA	LATENCY		
											rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 5	Reserved			Reserved, always read as 0.
4	PRFTBE	rw	0x01	Prefetch buffer enable 0: Prefetch buffer is disabled 1: Prefetch buffer is enabled
3	HLFCYA	rw	0x01	Flash half cycle access enable 0: Half cycle is disabled 1: Half cycle is enabled
2 : 0	LATENCY	rw	0x00	Latency These bits represent the ratio of SYSCLK (system clock) period to Flash access time. 000: Zero wait state, if 0 < SYSCLK≤ 24 MHz 001: One wait state, if 24 MHz < SYSCLK ≤ 48 MHz 010: Two wait state, if 48 MHz < SYSCLK ≤ 72 MHz

2.6.2 Flash access control register(FLASH_KEYR)

Address offest: 0x04

Reset value: 0xXXXX XXXX

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

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

Bit	Field	Type	Reset	Description
31 : 0	FKEYR	w	0xXXXX XXXX	FPEC(Flash key) These bits are used to enter the unlock key of FPEC.

Note: All these bits are written only and 0 is returned when being read.

2.6.3 Flash OPTKEY register(FLASH_OPTKEYR)

Address offset: 0x08

Reset value: 0xXXXX XXXX

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

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

Bit	Field	Type	Reset	Description
31 : 0	OPTKEYR	w	0xXXXX XXXX	Option byte key These bits are used to enter the key of the option byte, to disable OPTWRE.

Note: All these bits are written only and 0 is returned when being read.

2.6.4 Flash status register(FLASH_SR)

Address offset: 0x0C

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										EOP	WRPRTERR	Res.	PGERR	Res.	BSY
										rc_w1	rc_w1		rc_w1		r

Bit	Field	Type	Reset	Description
31 : 6	Reserved			Reserved, always read as 0.
5	EOP	rc_w1	0x00	End of operation When the flash operation (programming/erasing) is completed, the hardware sets this bit to ’1’, and eliminate this state by writing ’1’. Note: EOP status will be set for every successful programming or erasing.
4	WRPRTERR	rc_w1	0x00	Write protection error When the flash address for write protection is programmed, the hardware sets this bit to ’1’, and eliminate this state by writing ’1’.
3	Reserved			Reserved, always read as 0.
2	PGERR	rc_w1	0x00	Programming error Attempting to program an address that is not ’0xFFFF’, the hardware sets this bit to ’1’, and eliminate this state by writing ’1’. Note: The STRT bit in the FLASH_CR shall be cleared before the programming.
1	Reserved			Reserved, always read as 0.
0	BSY	r	0x00	Busy This bit indicates that the flash operation is in progress. This bit is set to ’1’ when the flash operation begins and written to ’0’ at the end of operation or in case of an error.

2.6.5 Flash control register(FLASH_CR)

Address offset: 0x10

Reset value: 0x0000 0080

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			EOPIE	Res.	ERRIE	OPTWRE	Res.	LOCK	STRT	OPTER	OPTPG	Res.	MER	PER	PG
			rw		rw	rc_w0		rw	rw	rw	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 13	Reserved			Reserved, always read as 0.
12	EOPIE	rw	0x00	End of operation interrupt enable This bit leads to an interrupt when the EOP bit in the FLASH_SR register changes to ’1’. 0: Interrupt is disabled 1: Interrupt is enabled

Bit	Field	Type	Reset	Description
11	Reserved			Reserved, always read as 0.
10	ERRIE	rw	0x00	Error interrupt enable This bit enables an interrupt in case of an FPEC error (when PGERR/WRPRTERR in the Flash-SR is set to '1'). 0: Interrupt is disabled 1: Interrupt is enabled
9	OPTWRE	rc_w0	0x00	Option byte write enable When this bit is '1', the option byte is allowed to be programmed. This bit is set to '1' when the correct key sequence is written in the FLASH_OPTKEYR. This bit can be cleared by writing 0 through the software.
8	Reserved			Reserved, always read as 0.
7	LOCK	rw	0x01	Lock Only '1' can be written. When this bit is '1', FPEC and FLASH_CR are locked. After detecting the correct unlocking sequence, the hardware automatically clears and sets this bit to '0'. After an unsuccessful unlock operation, this bit cannot be changed again until the next system reset.
6	STRT	rw	0x00	Start When this bit is '1', an erasing operation will be enabled. This bit can only be set to '1' by software and cleared automatically when BSY changes to '1'.
5	OPTER	rw	0x00	Option byte erase Option bytes are erased.
4	OPTPG	rw	0x00	Option byte programming Option bytes are programmed
3	Reserved			Reserved, always read as 0.
2	MER	rw	0x00	Mass erase Select to erase all user pages.
1	PER	rw	0x00	Page erase Select to erase pages.
0	PG	rw	0x00	Programming Select to program.

2.6.6 Flash address register(FLASH_AR)

Address offset: 0x014

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 0	FAR	w	0x0000 0000	Flash Address Select the address to be programmed before programming, and the page to be erased when erasing. Note: it is not allowed to write the register when the BSY bit in FLASH_SR is 1.

Change to current/last used address from hardware. During the page erasing, modify the register, to specify the page to be erased.

2.6.7 Option byte register(FLASH_OBR)

Address offset: 0x1C

Reset value: 0x03FF FC1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						Data1								Data0	
						r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data0						Reserved			nBOOT1	Res.	nRST_ STDBY	nRST_ STOP	WDG _SW	Res.	OPTERR
r	r	r	r	r	r				r		r	r	r		r

Bit	Field	Type	Reset	Description
31 : 26	Reserved			Reserved, always read as 0.
25 : 18	Data1	r	0xFF	Data1
17 : 10	Data0	r	0xFF	Data0
9 : 7	Reserved			Reserved, always read as 0.
6	nBOOT1	r	0x00	nBOOT1
5	Reserved			Reserved, always read as 0.
4	nRST_STDBY	r	0x01	Reset event when entering standby mode 0: Reset when entering standby mode 1: Do no reset when entering standby mode
3	nRST_STOP	r	0x01	Reset event when entering stop mode 0: Reset when entering STOP mode 1: Do not reset when entering STOP mode

Bit	Field	Type	Reset	Description
2	WDG_SW	r	0x01	Select watchdog event 0: Hardware watchdog 1: Software watchdog
1	Reserved			Reserved, always read as 0.
0	OPTERR	r	0x00	Option byte error When this bit is ’1’, it means that the option byte does not match its inverse code. Note: This bit is read-only.

The reset value of this register is related to the value written in the option byte, and the reset value of the OPTERR bit is related to the result of comparing the option byte with its inverse code when the option bytes are loaded.

2.6.8 Write protection register(FLASH_WRPR)

Address offset: 0x20

Reset value: 0xFFFF FFFF

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

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

Bit	Field	Type	Reset	Description
31 : 0	WRP	r	0xFFFF FFFF	Write protect This register contains write protection option bytes loaded by OBL. 0: The write protection is enabled 1: The write protection is disabled Note: These bits are read-only.

3 Cyclic redundancy check calculation unit(CRC)

Cyclic redundancy check calculation unit(CRC)

3.1 CRC introduction

The CRC (cyclic redundancy check) calculation unit is used to get a 32-bit CRC result from a fixed generator polynomial. Among other applications, CRC-based techniques are used to verify data transmission or storage integrity. In the scope of the EN/IEC60335-1 standard, they offer a means of verifying the Flash memory integrity. The CRC calculation unit helps compute a signature of the software during runtime, which is compared with a reference signature generated at link time and then stored in a given memory space.

3.2 CRC main features

- Uses CRC-32 (Ethernet) polynomial: 0x4C11DB7
  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$
- Single input/output 32-bit data register
- CRC computation completed in 4 AHB clock cycles (HCLK)
- General-purpose 8-bit register (can be used for temporary storage)

The CRC block diagram is shown in the following figure

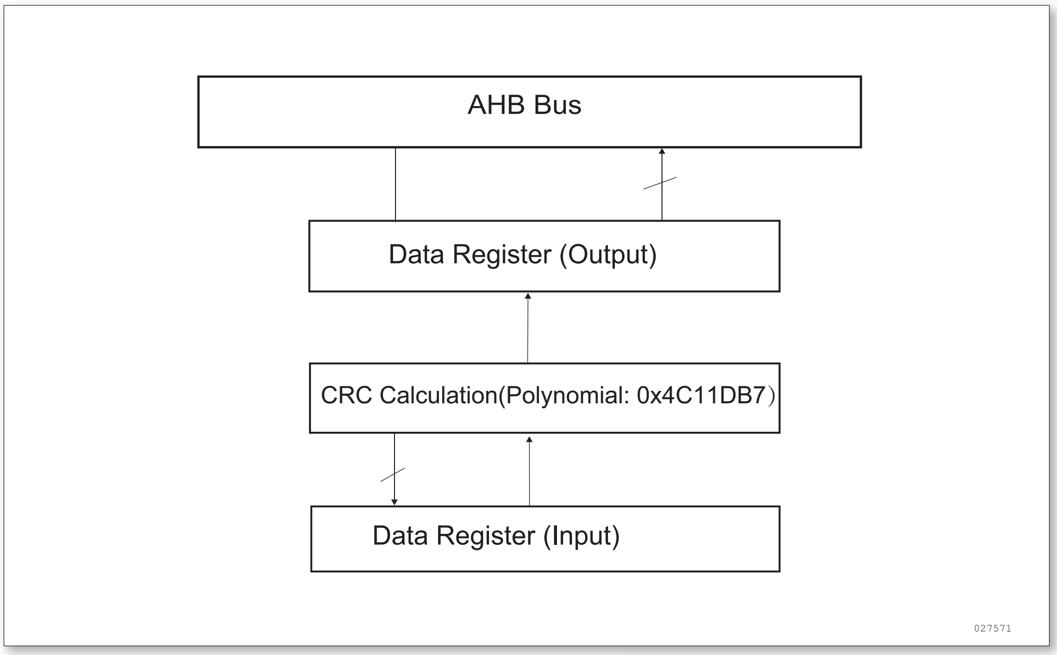


Figure 7. CRC Calculation Unit Block Diagram

3.3 CRC Functional description

The CRC calculation unit mainly consists of a single 32-bit data register, which:

- is used as an input register to enter new data in the CRC calculator (when writing into the register)
- holds the result of the previous CRC calculation (when reading the register)

Each write operation into the data register creates a combination of the previous CRC value and the new one (CRC computation is done on the whole 32-bit data word, and not byte per byte).

The write operation is stalled until the end of the CRC computation, thus allowing back-to-back write accesses or consecutive write and read accesses.

The CRC calculator can be reset to 0xFFFF FFFF with the RESET control bit in the CRC_CR register. This operation does not affect the contents of the CRC_IDR register.

3.4 CRC register

The CRC calculation unit contains two data registers and a control register.

Table 9. CRC Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	CRC_DR	CRC data register	0xFFFFFFFF	section 3.4.1
0x04	CRC_IDR	CRC independent data register	0x00000000	section 3.4.2
0x08	CRC_CTRL	CRC control register	0x00000000	section 3.4.3

3.4.1 CRC data register(CRC_DR)

Address offset: 0x00

Reset value: 0xFFFF FFFF

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

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

Bit	Field	Type	Reset	Description
31 : 0	DR	rw	0xFFFF FFFF	DR: Data register bits Used as an input register when writing new data into the CRC calculator. The CRC results are returned when being read.

3.4.2 CRC independent data register(CRC_IDR)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Always read as 0.
7 : 0	IDR	rw	0x00	IDR: General-purpose 8-bit data register bits Can be used as a temporary storage location for one byte. This register is not affected by CRC resets generated by the RESET bit in the CRC_CTRL register.

Note: This register is not involved in the CRC calculation and enables storing any data.

3.4.3 CRC control register(CRC_CTRL)

Address offset: 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															RESET
															w

Bit	Field	Type	Reset	Description
31 : 1	Reserved			Always read as 0.
0	RESET	w	0x00	RESET: CRC reset Sets the data register to 0xFFFF FFFF. This bit can only be set, it is automatically cleared by hard-ware.

4 Power control (PWR)

Power control(PWR)

4.1 Power supply

The chip requires a 2.0-to-5.5 V operating voltage supply (V_{DD}). An embedded regulator is used to supply the internal 1.5 V power.

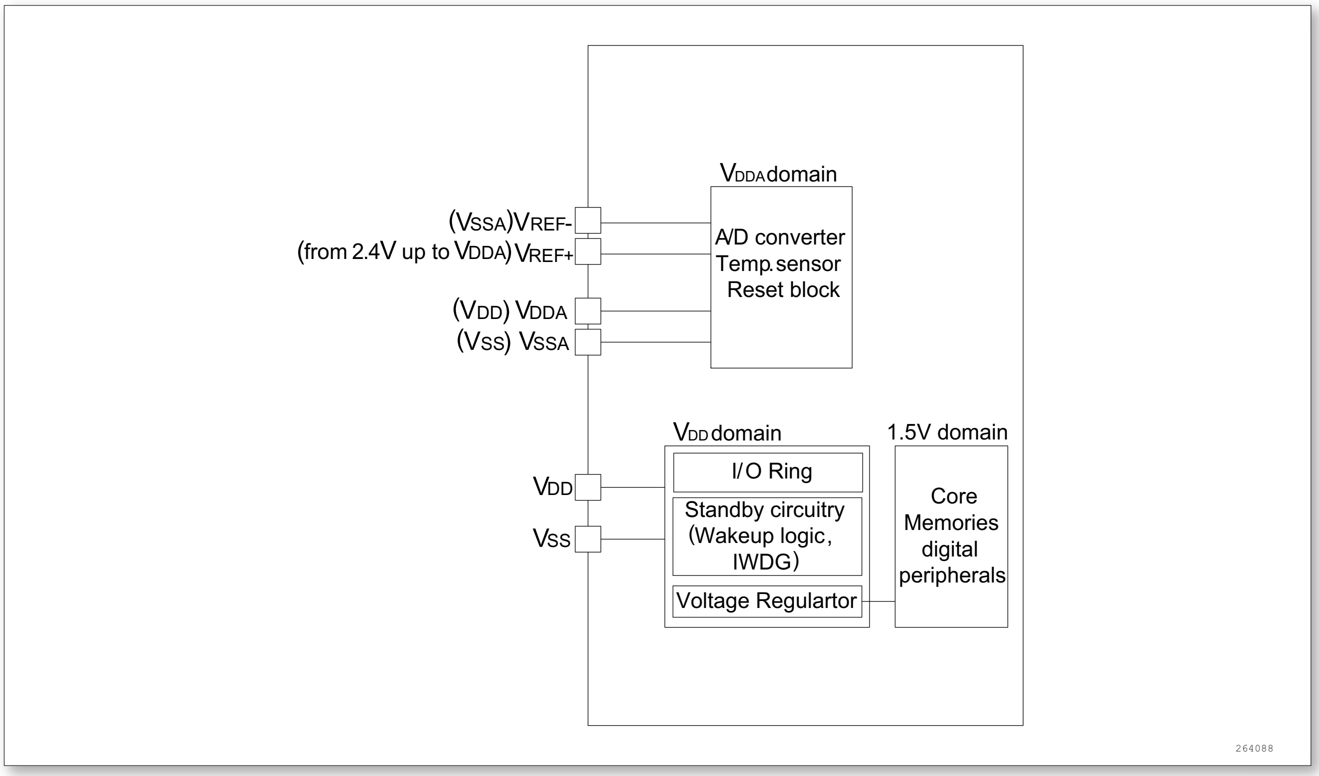


Figure 8. Power Supply Overview

Note: V_{DDA} and V_{SSA} must be connected to V_{DD} and V_{SS}.

4.1.1 Independent A/D converter supply and reference voltage

To improve conversion accuracy, the ADC has an independent power supply which can be separately filtered and shielded from noise on the PCB.

- The ADC supply input is available on a V_{DDA} pin
- An isolated supply ground connection is provided on pin V_{SSA}

When available (according to package), pin V_{REF-} must be tied to V_{SSA}.

4.1.2 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 full power to the 1.5 V domain (core, memories and digital peripherals).
- In Stop mode, the regulator supplies low-power to the 1.5 V domain, preserving contents of registers and SRAM.
- In Standby Mode, the regulator is powered off. The contents of the registers and SRAM are lost.

4.2 Power supply supervisor

4.2.1 Power on reset (POR)/power down reset (PDR)

The device has an integrated POR/PDR circuitry that allows proper operation starting from/down to 1.5 V.

The device remains in Reset mode when V_{DD}/V_{DDA} is below a specified threshold, V_{POR}/V_{PDR}, without the need for an external reset circuit. For more details concerning the power on/power down reset threshold, refer to the electrical characteristics of the datasheet.

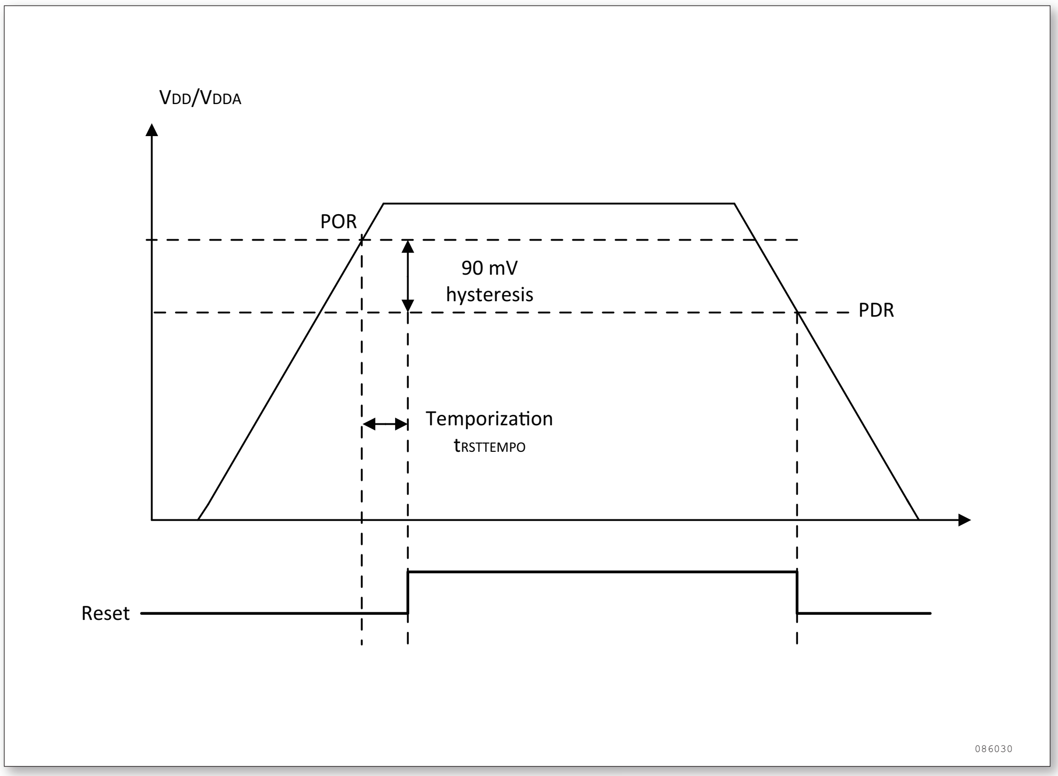


Figure 9. Power on Reset/Power Down Reset Waveform

4.2.2 Programmable voltage detector (PVD)

The PVD can be used to monitor the V_{DD} power supply by comparing it to a threshold selected by the PLS bits in the Power control register (PWR_CR).

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 threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if enabled through the EXTI registers. The PVD output interrupt can be generated when VDD drops below the PVD threshold and/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.

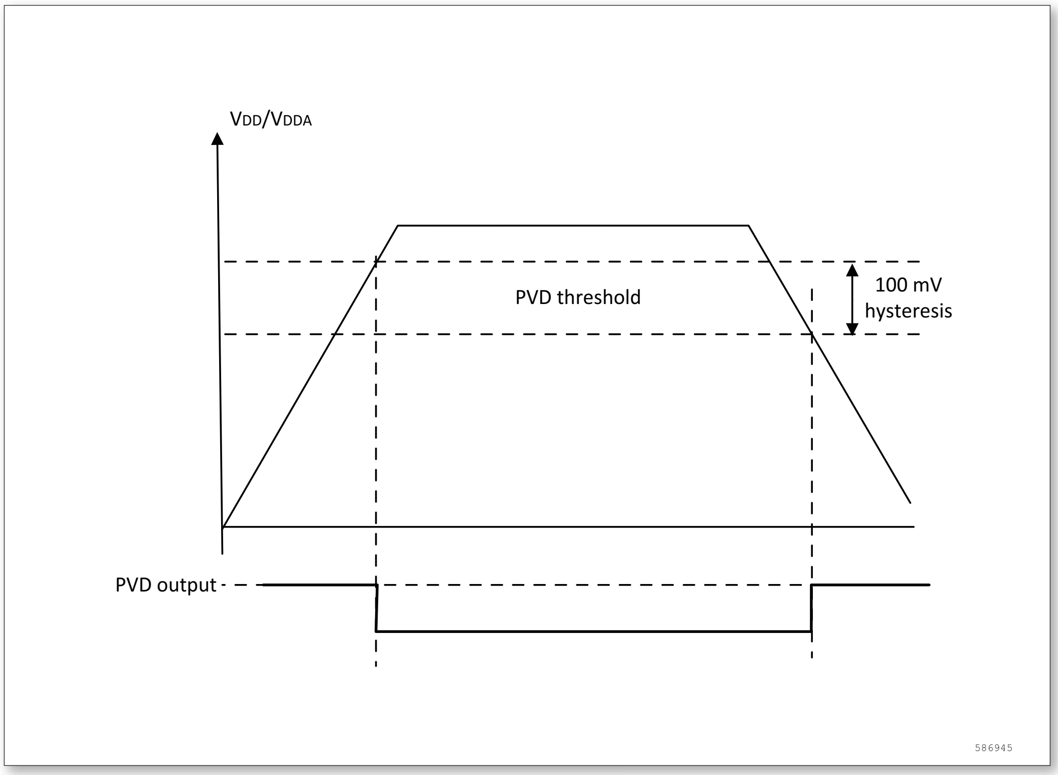


Figure 10. PVD Thresholds

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 device features three low-power modes:

- Sleep mode (CPU clock off, all peripherals including CPU peripherals like NVIC, SysTick, etc. are kept running)
- Stop mode (all clocks are stopped, and contents of register and SRAM are retained)
- Standby mode (1.5V domain is powered off, and the contents of the registers and SRAM are lost except)

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

- Slowing down the system clocks
- Gating the clocks to the APB and AHB peripherals when they are unused.

Table 10. Low-power Mode Summary

Mode	Entry	Wakeup	Effect on 1.5V domain clocks	Effect on V_{DD} domain clocks	Voltage regulator
Sleep (Sleep now or Sleep-on-exit)	WFI (Wait for Interrupt)	Any interrupt	CPU clock OFF, no effect on other clocks or ADC clock	None	ON
	WFE (Wait for Event)	Wakeup event			
Stop	PDDS bits + SLEEPDEEP bit + WFI or WFE	Any EXTI line (configured in the EXTI registers)	All 1.5V domain clocks OFF	HSI and HSE oscillators OFF	ON
Standby	PDDS bit + SLEEPDEEP bit + WFI or WFE	WKUP pin rising edge, external reset in NRST pin, and IWDG reset			OFF

4.3.1 Slowing down system clocks

In Run mode the speed of the system clocks (SYSCLK, HCLK, PCLK1, PCLK2) can be reduced by programming the prescaler registers. These prescalers can also be used to slow down peripherals before entering Sleep mode.

For more details refer to Section ”Clock Configuration Register (RCC_CFGR)

4.3.2 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), APB1 peripheral clock enable register (RCC_APB1ENR) and APB2 peripheral clock enable register (RCC_APB2ENR).

4.3.3 Sleep Mode

Entering Sleep mode

The Sleep mode is entered by executing the WFI (Wait For Interrupt) or WFE (Wait for Event) instructions. Two options are available to select the Sleep mode entry mechanism, depending on the SLEEPONEXIT bit in the system control register of CPU:

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

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

Refer to Table 11 and Table 12 for details on how to enter Sleep mode.

Table 11. Sleep-now

Sleep-now mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 - SLEEPONEXIT = 0 Refer to the CPU System Control register
Mode exit	If WFI was used for entry: Interrupt: Refer to Section ”Interrupt and Exception Vectors” If WFE was used for entry: Wakeup event: Refer to Section ”Wakeup Event Management”
Wakeup latency	None

Table 12. Sleep-on-exit

Sleep-on-exit mode	Description
Description	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - SLEEPDEEP = 0 - SLEEPONEXIT = 1 Refer to the CPU System Control register
Mode exit	If WFE was used for entry:Interrupt or clear Bit 1 of CPU control register If WFE was used for entry: Wakeup event: Refer to Section ”Wakeup Event Management”
Wakeup latency	None

4.3.4 Stop mode

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

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

Entering Stop mode

Refer to Table 13 for details on how to enter the Stop mode.

In 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): this is configured by the LSION bit in the Control/status register (RCC_CSR).

The ADC can also consume power in the Stop mode, unless they are disabled before entering it. To disable them, the ADON bit in the ADC_CR2 register shall be written to 0.

In addition, other GPIOs not used shall be configured with analog input, to prevent current consumption.

Exiting Stop mode

Refer to Table 13 for details on how to exit the Stop mode.

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

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

Table 13. Stop Mode

Stop Mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Set SLEEPDEEP bit in CPU System Control register - Clear PDDS bit in Power Control register (PWR_CR) Note: To enter Stop mode, all EXTI Line pending bits (in Pending register (EXTI_PR)), all peripheral interrupt pending bits must be reset. Otherwise, the Stop mode entry procedure is ignored and program execution continues.
must be reset	WFI (Wait for Interrupt) is executed in case of: Any EXTI Line configured in Interrupt mode (the corresponding EXTI Interrupt vector must be enabled in the NVIC). Refer to Section ”Interrupt and Exception Vectors”. If WFE (Wait for Event) is executed in case of: Any EXTI Line configured in event mode. Refer to Section ”Wakeup Event Management”.
Wakeup latency	HSI wakeup time

4.3.5 Standby mode

The Standby mode allows to achieve the lowest power consumption. It is based on the CPU deepsleep mode, with the voltage regulator disabled. The 1.5 V domain is consequently powered off. The PLL, the HSI oscillator and the HSE oscillator are also switched off. SRAM and register contents are lost.

Entering Standby mode

Refer to Table 14 for details on how to enter the Standby mode.

In Standby 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): this is configured by the LSION bit in the Control/status register (RCC_CSR).

Exiting Standby mode

The microcontroller exits the Standby mode when an external reset (NRST pin), an IWDG reset, a rising edge on the WKUP pin or the rising edge of an RTC alarm occurs. All registers are reset after wakeup from Standby except for Power control/status register (PWR_CSR).

After waking up from Standby mode, program execution restarts in the same way as after a Reset (boot pins sampling, vector reset is fetched, etc.). The SBF status flag in the Power control/status register (PWR_CSR) indicates that the core exits from the Standby mode.

Refer to Table 14 for details on how to exit the Standby mode.

Table 14. Standby Mode

Standby Mode	Description
Entry	WFI (Wait for Interrupt) or WFE (Wait for Event) while: - Set SLEEPDEEP bit in CPU System Control register - Set PDDS bit in Power Control register (PWR_CR) - Clear WUF bit in Power Control/Status register (PWR_CSR)
Exit	WKUP pin rising edge, external reset in NRST pin, and IWDG reset
Wakeup latency	Activating power regulator in the reset stage.

I/O states in Standby mode

In Standby mode, all I/O pins are high impedance except:

- Reset pin (still available)
- TAMPER pin if configured for tamper or calibration out
- WKUP pin, if enabled

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 Section ”Debug Support for Low-power Modes”.

4.4 Power control registers

Table 15. Overview of Power Control Registers

Offset	Acronym	Register Name	Reset	Section
0x00	PWR_CR	Power control register	0x00000000	section 4.4.1
0x04	PWR_CSR	Power control/status register	0x00000000	section 4.4.2

4.4.1 Power control registers(PWR_CR)

Address offset: 0x00

Reset value: 0x0000 0000(reset by wakeup from Standby mode)

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			PLS				Reserved				PVDE	CSBF	CWUF	PDDS	Res.
			rw	rw	rw	rw					rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31 : 13	Reserved			always read as 0.
12 : 9	PLS	rw	0x00	PVD level selection These bits are used to select the voltage threshold detected by 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 Other:reserved Note: Refer to the electrical characteristics of the datasheet for more details.
8:5	Reserved			always read as 0.
4	PVDE	rw	0x00	Power voltage detector enable 1: PVD enabled 0: PVD disabled
3	CSBF	rw	0x00	Clear standby flag Always read as 0 1: Clear the SBF Standby Flag (write). 0: No effect
2	CWUF	rw	0x00	Clear wakeup flag Always read as 0 1: Clear the WUF Wakeup Flag after 2 System clock cycles (write) 0: No effect
1	PDDS	rw	0x00	Power down deepsleep 1: Enter Standby mode when the CPU enters Deepsleep. 0: Enter Stop mode when the CPU enters Deepsleep.
0	Reserved			always read as 0.

4.4.2 Power control/status register(PWR_CSR)

Address offset: 0x04

Reset value: 0x0000 0000(not reset by wakeup from Standby mode)

Additional APB cycles are needed to read this register versus a standard APB read.

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							EWUP	Reserved					PVDO	SBF	WUF
							rw						rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 9	Reserved			always read as 0.
8	EWUP	rw	0x00	Enable WKUP pin 1: WKUP pin is used for wakeup from Standby mode and forced in input pull down configuration (rising edge on WKUP pin wakes up the system from Standby mode). 0: WKUP pin is used for general purpose I/O. An event on the WKUP pin does not wakeup CPU from Standby mode. Note: This bit is reset by a system reset.
7:3	Reserved			always read as 0.
2	PVDO	rw	0x00	PVD output It is valid only if PVD is enabled by the PVDE bit. 1: V_{DD}/V_{DDA} is lower than the PVD threshold selected with the PLS bits. 0: V_{DD}/V_{DDA} is higher than the PVD threshold selected with the PLS bits. Note: The PVD is stopped by Standby mode. For this reason, this bit is equal to 0 after Standby or reset until the PVDE bit is set.
1	SBF	rw	0x00	Standby flag This bit is set by hardware and cleared only by a POR/PDR (power on reset/power down reset) or by setting the CSBF bit in the Power control register (PWR_CR). 1: System has been in Standby mode 0: System has not been in Standby mode

Bit	Field	Type	Reset	Description
0	WUF	rw	0x00	Wakeup flag This bit is set by hardware and cleared only by a POR/PDR (power on reset/power down reset) or by setting the CWUF bit in the Power control register (PWR_CR). 1: A wakeup event was received from the WKUP pin 0: No wakeup event occurred Note: An additional wakeup event is detected if the WKUP pin is enabled (by setting the EWUP bit) when the WKUP pin level is already high.

5 Reset and clock control (RCC)

Reset and clock control (RCC)

5.1 Reset

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

5.1.1 System reset

A system reset sets all registers to their reset values except the reset flags in the clock controller CSR register.

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

1. A low level on the NRST pin (external reset)
2. Window watchdog end of count condition (WWDG reset)
3. Independent watchdog end of count condition (IWDG reset
4. A software reset (SW reset)

The reset source can be identified by checking the reset flags in the Control/Status register (RCC_CSR).

Software reset

The SYSRESETREQ bit in CPU Application Interrupt and Reset Control Register must be Set to ”1” to force a software reset.

5.1.2 Power reset

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

1. Power-on/power-down reset (POR/PDR reset)
2. When exiting Standby mode

A power reset sets all registers to their reset values.

These sources in Figure 11 act on the NRST pin and it is always kept low during Reset. The RESET service routine vector is fixed at address 0x0000_0004.

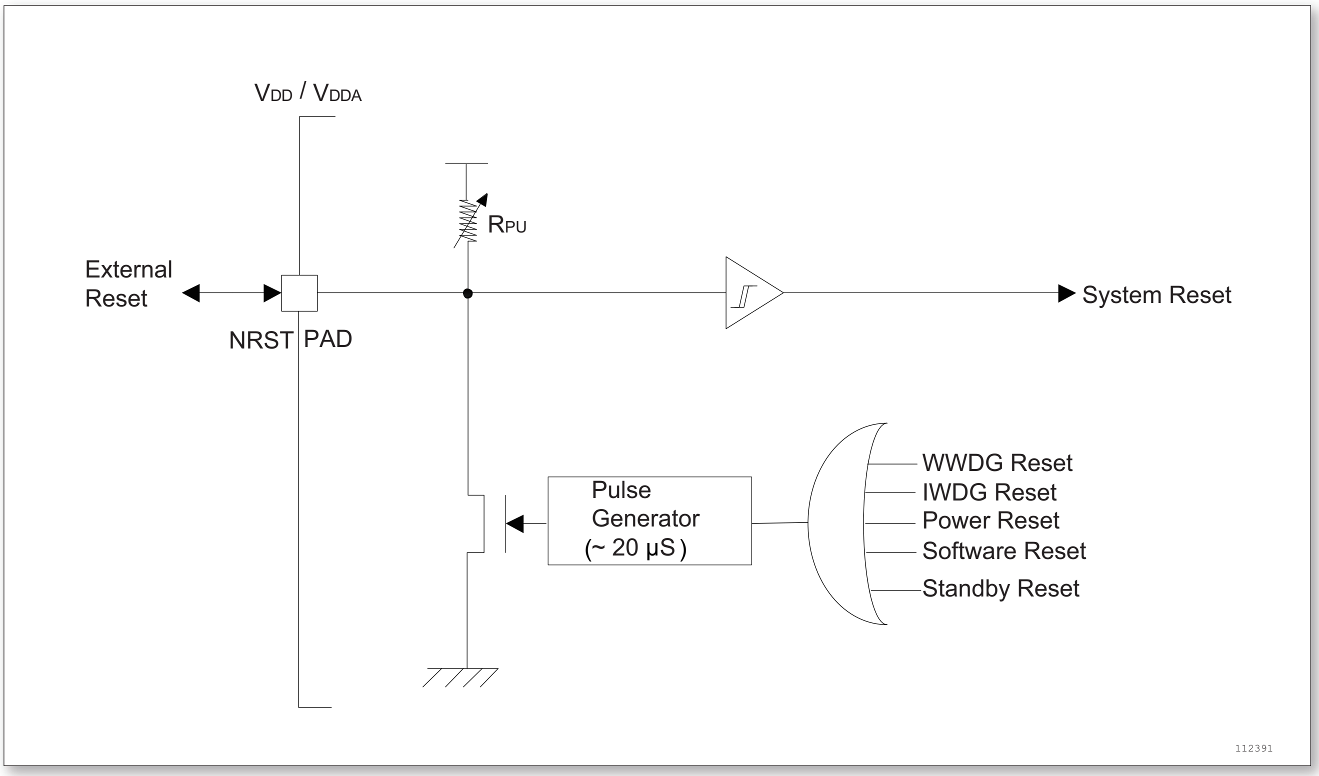


Figure 11. Reset Circuit

5.2 Clocks

Four different clock sources can be used to drive the system clock (SYSCLK):

- HSI oscillator clock divided by 2
- HSI oscillator clock
- HSE oscillator clock
- LSI clock

The devices have the following two secondary clock sources:

- 40 kHz low speed internal RC (LSI RC), which drives the independent watchdog.

Each clock source can be switched on or off independently when it is not used, to optimize power consumption.

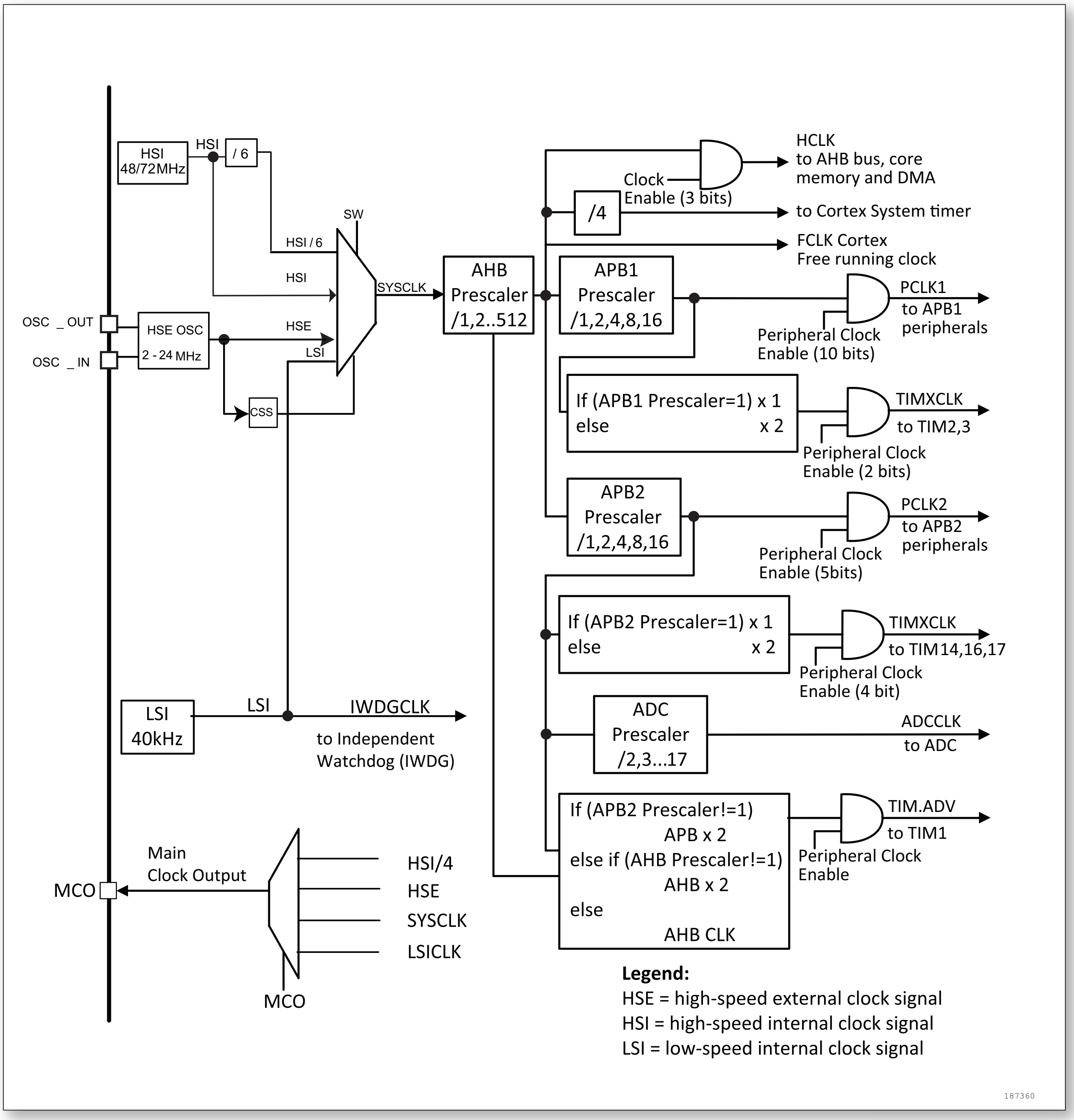

HSI 48/72MHz
HSI
/ 6
HSI / 6
HSI
SW
SYSCLK
OSC _ OUT
OSC _ IN
HSE OSC 2 - 24 MHz
HSE
LSI
CSS
AHB Prescaler /1,2..512
Clock Enable (3 bits)
/4
HCLK to AHB bus, core memory and DMA
to Cortex System timer
FCLK Cortex Free running clock
APB1 Prescaler /1,2,4,8,16
PCLK1 to APB1 peripherals
Peripheral Clock Enable (10 bits)
If (APB1 Prescaler=1) x 1 else x 2
TIMXCLK to TIM2,3
Peripheral Clock Enable (2 bits)
APB2 Prescaler /1,2,4,8,16
PCLK2 to APB2 peripherals
Peripheral Clock Enable (5bits)
If (APB2 Prescaler=1) x 1 else x 2
TIMXCLK to TIM14,16,17
Peripheral Clock Enable (4 bit)
ADC Prescaler /2,3...17
ADCCLK to ADC
If (APB2 Prescaler!=1) APB x 2 else if (AHB Prescaler!=1) AHB x 2 else AHB CLK
TIM.ADV to TIM1
Peripheral Clock Enable
LSI 40kHz
LSI
IWDGCLK to Independent Watchdog (IWDG)
Main Clock Output
MCO
HSI/4
HSE
SYSCLK
LSICLK
MCO
Legend:
HSE = high-speed external clock signal
HSI = high-speed internal clock signal
LSI = low-speed internal clock signal
187360

Figure 12. Clock Tree

Several prescaler factors allow the configuration of the frequency of AHB, the high-speed APB (APB2) and the low-speed APB (APB1) domains. The maximum frequency of the AHB, APB1 and the APB2 domains is 72MHzMHz.

The RCC feeds the CPU System Timer (SysTick) external clock with the AHB clock divided by 8. This clock can be used or AHB clock used as SysTick through the SysTick Control and by configuring Status Register. The ADC clock is generated by dividing the high-speed APB2 clock.

The timer clock frequencies are automatically fixed by hardware. There are two cases:

1. If the APB prescaler factor is 1, the timer clock frequencies are Set to the same frequency as that of the APB domain to which the timers are connected.
2. Otherwise, they are Set to twice (×2) the frequency of the APB domain to which the timers are connected.

FCLK is the free running clock.

5.2.1 HSE clock

The high speed external clock signal (HSE) can be generated from two possible clock sources:

- HSE external crystal/ceramic resonator
- HSE user external clock

The resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and startup stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

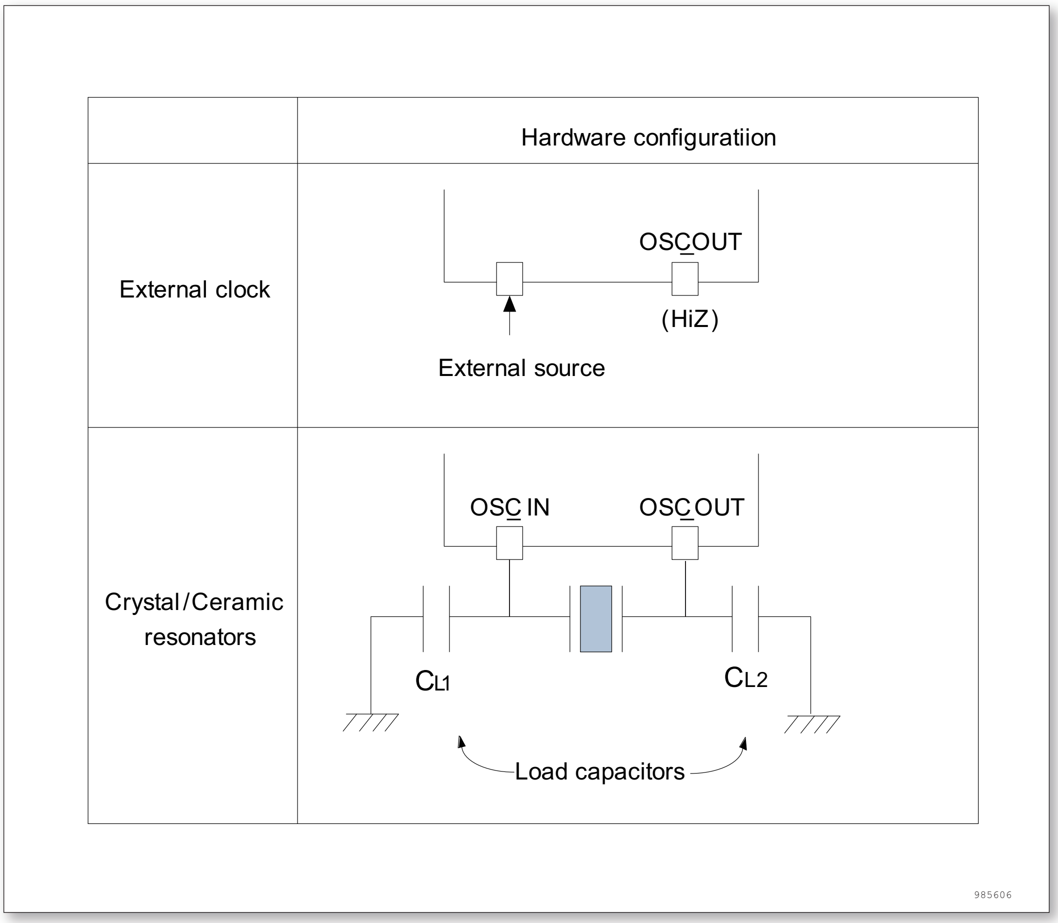


Figure 13. Clock Sources

External clock source (HSE bypass)

In this mode, an external clock source must be provided. It can have a frequency of up to 24 MHz. You can select this mode by setting the HSEBYP and HSEON bits in the Clock control register (RCC_CR). The external clock signal (square, sinus or triangle) with 50%

dutycycle has to drive the OSC_IN pin while the OSC_OUT pin should be left hi-Z.

External crystal/ceramic resonator (HSE crystal)

The external oscillator has the advantage of producing a very accurate rate on the main clock. The associated hardware configuration is shown in Figure 13. Refer to the electrical characteristics section of the datasheet for more details.

The HSERDY flag in the Clock control register (RCC_CR) indicates if the high-speed external oscillator is stable or not. At startup, the clock is not released until this bit is Set to ’1’ by hardware. An interrupt can be generated if enabled in the Clock interrupt register (RCC_CIR).

The HSE Crystal can be switched on and off using the HSEON bit in the Clock control register (RCC_CR).

5.2.2 HSI clock

The HSI clock signal is generated from an internal HSI Oscillator The HSI oscillator has the advantage of providing a clock source at low cost (no external components). It also has a faster startup time than the HSE crystal oscillator, however, even with calibration the frequency is less accurate.

Calibration

The oscillator frequencies can vary from one chip to another due to manufacturing process variations, this is why each device is factory calibrated by ST for 1%(25°C). After reset, the factory calibration value is loaded in the HSICAL bits in the Clock control register (RCC_CR).

The HSIRDY flag in the Clock control register (RCC_CR) indicates if the HSI RC is stable or not. At startup, the HSI RC output clock is not released until this bit is Set to ’1’ by hardware. The HSI RC can be switched on and off using the HSION bit in the Clock control register (RCC_CR).

The HSI signal can also be used as a backup source (Auxiliary clock) if the HSE crystal oscillator fails. Refer to Section section 5.2.5。

5.2.3 LSI clock

The LSI RC acts as an low-power clock source that can be kept running in Stop and Standby mode for the independent watchdog (IWDG) and Auto-wakeup unit (AWU). The clock frequency is around 40 kHz (between 30 kHz and 60 kHz). For more details, refer to the electrical characteristics section of the datasheets.

The LSI RC can be switched on and off using the LSION bit in the Control/status register (RCC_CSR). The LSIRDY flag in the Control/status register (RCC_CSR) indicates if the low-speed internal oscillator is stable or not. At startup, the clock is not released until this bit is Set to ’1’ by hardware. An LSI interrupt request can be generated if enabled in the Clock interrupt register (RCC_CIR).

5.2.4 System clock (SYSCLK) selection

After a system reset, the HSI oscillator is selected as system clock. A switch from one clock source to another occurs only if the target clock source is ready. The switch will occur only when the clock source is ready. Status bits in the Clock configuration register (RCC_CFGR). indicate which clock(s) is (are) ready and which clock is currently used as system clock.

5.2.5 Clock security system (CSS)

Clock Security System can be activated by software. In this case, the clock detector is enabled after the HSE oscillator startup delay, and disabled when this oscillator is stopped.

If a failure is detected on the HSE clock, the HSE oscillator is automatically disabled, a clock failure event is sent to the break input of the advanced-control timers (TIM1) and an interrupt is generated to inform the software about the failure (Clock Security SystemInter-rupt CSSI), allowing the software to perform rescue operations. The CSSI is linked to the CPU NMI (Non-Maskable Interrupt) exception vector.

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

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

5.2.6 Watchdog clock

If the Independent watchdog (IWDG) is started by either hardware option or software access, the LSI oscillator is forced ON and cannot be disabled. After the LSI oscillator stabilization, the clock is provided to the IWDG.

5.2.7 Clock-out capability

The microcontroller clock output (MCO) capability allows the clock to be output onto the external MCO pin.

The registers of the corresponding GPIO port shall be configured with functions.One of clock signals can be selected as the MCO clock.

- SYSCLK
- HSI/4
- HSE
- LSI
  - The selection is controlled by the MCO [2:0] bits of the Clock configuration register (RCC_CFGR).

5.3 RCC Register file and memory mapping description

Table 16. Overview of RCC Registers

Offset	Acronym	Register Name	Reset	Section
0x00	RCC_CR	Clock control register	0x0000FF01	section 5.3.1
0x04	RCC_CFGR	Clock configuration register	0x00000000	section 5.3.2
0x08	RCC_CIR	Clock interrupt register	0x00000000	section 5.3.3
0x0C	RCC_APB2RSTR	APB2 peripheral reset register	0x00000000	section 5.3.4
0x10	RCC_APB1RSTR	APB1 peripheral reset register	0x00000000	section 5.3.5
0x14	RCC_AHBENR	AHB peripheral clock enable register	0x00000014	section 5.3.6
0x18	RCC_APB2ENR	APB2 peripheral clock enable register	0x00000000	section 5.3.7
0x1C	RCC_APB1ENR	APB1 peripheral clock enable register	0x00000000	section 5.3.8
0x24	RCC_CSR	Control status register	0x0C000000	section 5.3.9
0x28	RCC_AHBRSTR	AHB peripheral clock reset register	0x00000000	section 5.3.10
0x40	RCC_SYSCFG	System configuration register	0x00001400	section 5.3.11

5.3.1 Clock control register(RCC_CR)

Address offset: 0x00

Reset value: 0x0000 FF01

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											HSI_72 M_EN	CSSON	HSEBYP	HSERDY	HSEON
											rw	rw	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSICAL								HSICALSEL					Res.	HSIRDY	HSION
rw	rw	rw	rw	rw	rw	rw	rw	w	w	w	w	w		r	rw

Bit	Field	Type	Reset	Description
31 : 21	Reserved			Always read as 0.
20	HSI_72M_EN	rw	0x00	HSI_72M_EN: Internal high speed clock output selection 0: Internal high-speed clock output 48MHZ clock 1: Internal high-speed clock output 72MHZ clock Default is 0
19	CSSON	rw	0x00	CSSON: Clock security system enable Set to '1' or cleared by software to enable the clock detector . 0: Clock detector OFF 1: Clock detector ON if the external oscillator is ready

Bit	Field	Type	Reset	Description
18	HSEBYP	rw	0x00	HSEBYP: External high-speed clock bypass Set to ’1’ or cleared by software to bypass the oscillator with an external clock. The HSEBYP bit can be written only if the external oscillator is disabled. 0: external oscillator not bypassed 1: external oscillator bypassed with external clock
17	HSERDY	r	0x00	HSERDY: External high-speed clock ready flag Set to ’1’ by hardware to indicate that the external clock is stable. 0: External clock not ready 1: External clock ready
16	HSEON	rw	0x00	HSEON: External high-speed clock enable Set to ’1’ or cleared by software. Cleared by hardware to stop the external clock when entering Stop or Standby mode. This bit cannot be reset if the external clock is used directly or indirectly as the system clock. 0: HSE oscillator OFF 1: HSE oscillator ON
15: 8	HSICAL	rw	0xFF	HSICAL: Internal high-speed clock calibration These bits are initialized automatically at startup.For factory calibration values, the user can write other calibration values, but the readout is always the factory calibration value. If HSICALSEL = 0x1F, the value written can recalibrate the HSI frequency, otherwise the write will only have no effect.
7: 3	HSICALSEL	w	0x00	HSICALSEL：The initial value of the internal high-speed clock calibration value is 0, and it is still read as 0 after writing 1F. 1F: Select the value of the register HSICAL Other: Select factory calibration value
2	Reserved			Always read as 0.
1	HSIRDY	r	0x00	HSIRDY: Internal high-speed clock ready flag A ’1’ is set by hardware to indicate that the internal clock has stabilized. After the HSION bit is cleared, this bit requires six internal clock cycles to be cleared. 0: Internal clock is not ready 1: Internal clock ready

Bit	Field	Type	Reset	Description
0	HSION	rw	0x01	HSION: Internal high-speed clock enable Set to ’1’ or cleared by software. Set to ’1’ by hardware to force the internal 8 MHz RC oscillator ON when leaving Stop or Standby mode or in case of failure of the external oscillator used as system clock. This bit cannot be reset if the internal 8 MHz RC is used directly or indirectly as system clock or is selected to become the system clock. 0: internal high speed clock disabled 1: internal high speed clock enabled

5.3.2 Clock configuration register(RCC_CFGR)

Address offset：0x04

Reset value：0x0000 0000

Access: no wait state, word, half-word and byte access

1 or 2 wait states inserted only if the access occurs during clock source switch.

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PPRE2			PPRE1			HPRE				SWS		SW	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

Bit	Field	Type	Reset	Description
31 : 27	Reserved			Always read as 0.
26 : 24	MCO	rw	0x00	MCO: Microcontroller clock output Set to ’1’ or cleared by software. 00x: No clock; 010: LSI clock selected; 011：Reserved 100: System clock (SYSCLK) selected; 101：HSI clock divided by 4 selected； 110: HSE clock selected; 111：Reserved Note: 1. This clock output may have some truncated cycles at startup or during MCO clock source switching. 2. When the System Clock is selected to output to the MCO pin, make sure that this clock does not exceed 50 MHz (the maximum I/O speed).

Bit	Field	Type	Reset	Description
23：14	Reserved			Always read as 0.
13: 11	PPRE2	rw	0x00	PPRE2: APB high-speed prescaler(APB2) Set to '1' and cleared by software to control the prescaler factor of the APB high-speed clock (PCLK2). 0xx: HCLK not divided 100: HCLK divided by 2 101: HCLK divided by 4 110: HCLK divided by 8 111: HCLK divided by 16
10: 8	PPRE1	rw	0x00	PPRE1: APB low-speed prescaler(APB1) Set to '1' and cleared by software to control the prescaler factor of the APB1 low-speed clock (PCLK1). 0xx: HCLK not divided 100: HCLK divided by 2 101: HCLK divided by 4 110: HCLK divided by 8 111: HCLK divided by 16
7: 4	HPRE	rw	0x00	HPRE: AHB Prescaler Set to '1' and cleared by software to control the prescaler factor of the AHB clock. 0xxx: SYSCLK not divided 1000: SYSCLK divided by 2 1001: SYSCLK divided by 4 1010: SYSCLK divided by 8 1011: SYSCLK divided by 16 1100: SYSCLK divided by 64 1101: SYSCLK divided by 128 1110: SYSCLK divided by 256 1111: SYSCLK divided by 512 Note: 1. The prefetch buffer must be kept on when using a prescaler factor greater than 1 on theAHB clock. Refer to Section "Reading the Flash Memory" for more details.
3: 2	SWS	r	0x00	SWS: System clock switch status Set to '1' and cleared by hardware to indicate which clock source is used as system clock. 00: HSI oscillator divided by 6 and used as system clock 01: HSE oscillator used as system clock 10: HSI used as system clock 11: LSI used as system clock

Bit	Field	Type	Reset	Description
1: 0	SW	rw	0x00	SW: System clock switch Set to ’1’ and cleared by software to select SYSCLK source. Set by hardware to force HSI selection when leaving Stop and Standby mode or in case of failure of the HSE oscillator used directly or indirectly as system clock. 00: HSI oscillator divided by 6 and used as system clock 01: HSE oscillator selected as system clock 10：HSI is used as the system clock and is forced to 1 11: LSI used as system clock

5.3.3 Clock interrupt register(RCC_CIR)

Address offset：0x08

Reset value：0x0000 0000

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								CSSC	Reserved			HSE RDYC	HSI RDYC	Res.	LSI RDYC
								rc_w1				rc_w1	rc_w1		rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				HSE RDYIE	HSI RDYIE	Res.	LSI RDYIE	CSSF	Reserved			HSE RDYF	HSI RDYF	Res.	LSI RDYF
				rw	rw		rw	r				r	r		r

Bit	Field	Type	Reset	Description
31 : 24	Reserved			Always read as 0.
23	CSSC	rc_w1	0x00	CSSC: Clock security system interrupt clear This bit is set to ’1’ by software to clear the CSSF flag. 0: No effect 1: Clear CSSF flag
22 : 20	Reserved			Always read as 0.
19	HSERDYC	rc_w1	0x00	HSERDYC: HSE ready interrupt clear This bit is set to ’1’ by software to clear the HSERDYF flag. 0: No effect 1: HSERDYF cleared
18	HSIRDYC	rc_w1	0x00	HSIRDYC: HSI ready interrupt clear This bit is set to ’1’ by software, to clear the HSIRDYF flag. 0: No effect 1: HSIRDYF cleared
17	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
16	LSIRDYC	rc_w1	0x00	LSIRDYC: LSI ready interrupt clear This bit is set to ’1’ by software to clear the LSIRDYF flag. 0: No effect 1: LSIRDYF cleared
15: 12	Reserved			Always read as 0.
11	HSERDYIE	rw	0x00	HSERDYIE: HSE ready interrupt enable Set to ’1’ and cleared by software to enable/disable interrupt caused by the external oscillator stabilization. 0: HSE ready interrupt disabled 1: HSE ready interrupt enabled
10	HSIRDYIE	rw	0x00	HSIRDYIE: HSI ready interrupt enable Set to ’1’ and cleared by software to enable/disable interrupt caused by the internal 8 MHz RC oscillator stabilization. 0: HSI ready interrupt disabled 1: HSI ready interrupt enabled
9	Reserved			Always read as 0.
8	LSIRDYIE	rw	0x00	LSIRDYIE: LSI ready interrupt enable Set to ’1’ and cleared by software to enable/disable interrupt caused by the internal 40 kHz RC oscillator stabilization. 0: LSI ready interrupt disabled 1: LSI ready interrupt enabled
7	CSSF	r	0x00	CSSF: Clock security system interrupt flag Set to ’1’ by hardware when a failure is detected in the external oscillator. Cleared by software through setting the CSSC bit to ’1’. 0: No clock security interrupt caused by HSE clock failure 1: Clock security interrupt caused by HSE clock failure
6: 4	Reserved			Always read as 0.
3	HSERDYF	r	0x00	HSERDYF: HSE ready interrupt flag Set to ’1’ by hardware when External High Speed clock becomes stable and HSERDYDIE is set to ’1’. Cleared by software setting the HSERDYC bit. 0：No clock ready interrupt caused by the external oscillator 1: Clock ready interrupt caused by the external oscillator

Bit	Field	Type	Reset	Description
2	HSIRDYF	r	0x00	HSIRDYF: HSI ready interrupt flag Set to ’1’ by hardware when the Internal High Speed clock becomes stable and HSIRDYDIE is set. Cleared by software setting the HSIRDYC bit. 0: No clock ready interrupt caused by the internal RC oscillator 1: Clock ready interrupt caused by the internal RC oscillator
1	Reserved			Always read as 0.
0	LSIRDYF	r	0x00	LSIRDYF: LSI ready interrupt flag Set to ’1’ by hardware when the internal low speed clock becomes stable and LSIRDYDIE is set. Cleared by software setting the LSIRDYC bit. 0: No clock ready interrupt caused by the internal RC 40 kHz oscillator 1: Clock ready interrupt caused by the internal RC 40 kHz oscillator

5.3.4 APB2 peripheral reset register(RCC_APB2RSTR)

Address offset：0x0C

Reset value：0x0000 0000

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									DBGMCU	Reserved			TIM17	TIM16	TIM14
									rw				rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP	UART1	Res.	SPI1	TIM1	Res.	ADC1	Reserved								SYSCFG
rw	rw		rw	rw		rw									rw

Bit	Field	Type	Reset	Description
31: 23	Reserved			Always read as 0.
22	DBGMCU	rw	0x00	DBGMCU：DBGMCU reset
21: 19	Reserved			Always read as 0.
18	TIM17	rw	0x00	TIM17：TIM17 timer reset Set to ’1’ or cleared by software. 0: No effect 1: Reset TIM17 timer
17	TIM16	rw	0x00	TIM16: TIM16 timer reset Set to ’1’ or cleared by software. 0: No effect 1: Reset TIM16 timer

Bit	Field	Type	Reset	Description
16	TIM14	rw	0x00	TIM14: TIM14 timer reset Set to '1' or cleared by software. 0: No effect 1: Reset TIM14 timer
15	COMP	rw	0x00	COMPRST: Comparator reset Set to '1' or cleared by software. 0: No effect 1: Reset comparator interface
14	UART1	rw	0x00	UART1: UART1 reset Set to '1' or cleared by software. 0: No effect 1：Reset UART1
13	Reserved			Always read as 0.
12	SPI1	rw	0x00	SPI1: SPI1 reset Set to '1' or cleared by software. 0: No effect 1：Reset SPI1
11	TIM1	rw	0x00	TIM1: TIM1 timer reset Set to '1' or cleared by software. 0: No effect 1: Reset TIM1 timer
10	Reserved			Always read as 0.
9	ADC1	rw	0x00	ADC1: ADC1 interface reset Set to '1' or cleared by software. 0: No effect 1: Reset ADC1 interface
8: 1	Reserved			Always read as 0.
0	SYSCFG	rw	0x00	SYSCFG: System Configuration register reset Set to '1' or cleared by software. 0: No effect 1: Reset SYSCFG

5.3.5 APB1 peripheral reset register(RCC_APB1RSTR)

Address offset：0x10

Reset value：0x0000 0000

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			PWR	Reserved						I2C1	Reserved			UART2	Res.
			rw							rw				rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2	Reserved		WWDG	Reserved									TIM3	TIM2
	rw			rw										rw	rw

Bit	Field	Type	Reset	Description
31 : 29	Reserved			Always read as 0.
28	PWR	rw	0x00	PWR: Power interface reset Set to ’1’ or cleared by software. 0: No effect 1: Reset power interface
27 : 22	Reserved			Always read as 0.
21	I2C1	rw	0x00	I2C1: I2C1 reset Set to ’1’ or cleared by software. 0: No effect 1: Reset I2C1
20 : 18	Reserved			Always read as 0.
17	UART2	rw	0x00	UART2: UART2 reset Set to ’1’ or cleared by software. 0: No effect 1: Reset UART2
16 : 15	Reserved			Always read as 0.
14	SPI2	rw	0x00	SPI2: SPI2 reset Set to ’1’ or cleared by software. 0: No effect 1: Reset SPI2
13 : 12	Reserved			Always read as 0.
11	WWDG	rw	0x00	WWDG: Window watchdog reset Set to ’1’ or cleared by software. 0: No effect 1: Reset window watchdog
10 : 2	Reserved			Always read as 0.
1	TIM3	rw	0x00	TIM3: Timer3 reset Set to ’1’ or cleared by software. 0: No effect 1: Reset TIM3 timer
0	TIM2	rw	0x00	TIM2: Timer2 reset Set to ’1’ or cleared by software. 0: No effect 1: Reset TIM2 timer

5.3.6 AHB peripheral clock enable register(RCC_AHBENR)

Address offset：0x14

Reset value：0x0000 0014

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					HWDIV	Reserved					GPIOD	GPIOC	GPIOB	GPIOA	Res.
					rw						rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CRC	Res.	FLASH	Res.	SRAM	Res.	DMA
									rw		rw		rw		rw

Bit	Field	Type	Reset	Description
31 : 27	Reserved			Always read as 0.
26	HWDIV	rw	0x00	HWDIV: HWDIV clock enable Set to ’1’ or cleared by software. 0：HWDIV clock disabled 1：HWDIV clock enabled
25 : 21	Reserved			Always read as 0.
20	GPIOD	rw	0x00	GPIOD: GPIOD clock enable 0: GPIOD clock disabled 1: GPIOD clock enabled
19	GPIOC	rw	0x00	GPIOC: GPIOC clock enable 0: GPIOC clock disabled 1: GPIOC clock enabled
18	GPIOB	rw	0x00	GPIOB: GPIOB clock enable 0: GPIOB clock disabled 1: GPIOB clock enabled
17	GPIOA	rw	0x00	GPIOA: GPIOA clock enable 0: GPIOA clock disabled 1: GPIOA clock enabled
16 : 7	Reserved			Always read as 0.
6	CRC	rw	0x00	CRC: CRC clock enable Set to ’1’ or cleared by software. 0: CRC clock disabled 1: CRC clock enabled
5	Reserved			Always read as 0.
4	FLASH	rw	0x01	FLASH: FLASH clock enable 0：FLASH clock disabled 1：FLASH clock enabled
3	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
2	SRAM	rw	0x01	SRAM: SRAM interface clock enable Set to ’1’ or clear by software, to enable/disable the SRAM clock in Sleep mode. 0: SRAM clock disabled in Sleep mode. 1: SRAM clock enabled in Sleep mode.
1	Reserved			Always read as 0.
0	DMA	rw	0x00	DMA: DMA clock enable Set to ’1’ or cleared by software. 0：DMA clock disabled 1：DMA clock enabled

5.3.7 APB2 peripheral clock enable register(RCC_APB2ENR)

Address offset：0x18

Reset value：0x0000 0000

Access: no wait state, word, half-word and byte access

Note: When the peripheral clock is not active, the peripheral registervalues may not be read by software.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									DBGMCU	Reserved			TIM17	TIM16	TIM14
									rw				rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP	UART1	Res.	SPI1	TIM1	Res.	ADC1	Reserved								SYSCFG
rw	rw		rw	rw		rw									rw

Bit	Field	Type	Reset	Description
31: 23	Reserved			Always read as 0.
22	DBGMCU	rw	0x00	DBGMCU: DBGMCU enable
21: 19	Reserved			Always read as 0.
18	TIM17	rw	0x00	TIM17: TIM17 timer enable Set to ’1’ or cleared by software. 0: TIM17 clock disabled 1: TIM17 clock enabled
17	TIM16	rw	0x00	TIM16: TIM16 timer enable Set to ’1’ or cleared by software. 0: TIM16 clock disabled 1: TIM16 clock enabled
16	TIM14	rw	0x00	TIM14: TIM14 timer enable Set to ’1’ or cleared by software. 0: TIM14 clock disabled 1: TIM14 clock enabled

Bit	Field	Type	Reset	Description
15	COMP	rw	0x00	COMP: Comparator enable Set to ’1’ or cleared by software. 0: Compartator interface clock disabled 1: Compartator interface clock enabled
14	UART1	rw	0x00	UART1: UART1 clock enable Set to ’1’ or cleared by software. 0：UART1 clock disabled 1：UART1 clock enabled
13	Reserved			Always read as 0.
12	SPI1	rw	0x00	SPI1: SPI1 clock enable Set to ’1’ or cleared by software. 0：SPI1 clock disabled 1：SPI1 clock enabled
11	TIM1	rw	0x00	TIM1: TIM1 Timer clock enable Set to ’1’ or cleared by software. 0: TIM1 clock disabled 1: TIM1 clock enabled
10	Reserved			Always read as 0.
9	ADC1	rw	0x00	ADC1: ADC1 interface clock enable Set to ’1’ or cleared by software. 0: ADC1 interface clock disabled 1: ADC1 interface clock enabled
8: 1	Reserved			Always read as 0.
0	SYSCFG	rw	0x00	SYSCFGEN: System configuration register enable Set to ’1’ or cleared by software. 0: System configuration register clock disabled 1: System configuration register clock enabled

5.3.8 APB1 peripheral clock enable register (RCC_APB1ENR)

Address offset：0x1C

Reset value：0x0000 0000

Access: no wait state, word, half-word and byte access

Note: When the peripheral clock is not active, the peripheral registervalues may not be read by software and the returned value is always 0x0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PWR	Reserved						I2C1	Reserved			UART2	Res.
			rw							rw				rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2	Reserved		WWDG	Reserved									TIM3	TIM2
	rw			rw										rw	rw

Bit	Field	Type	Reset	Description
31 : 29	Reserved			Always read as 0.
28	PWR	rw	0x00	PWR: Power interface clock enable Set to ’1’ or cleared by software. 0: Power interface clock disabled 1: Power interface clock enabled
27: 22	Reserved			Always read as 0.
21	I2C1	rw	0x00	I2C1: I2C1 clock enable Set to ’1’ or cleared by software. 0：I2C1 clock disabled 1：I2C1 clock enabled
20：18	Reserved			Always read as 0.
17	UART2	rw	0x00	UART2: UART2 clock enable Set to ’1’ or cleared by software. 0：UART2 clock disabled 1：UART2 clock enabled
16：15	Reserved			Always read as 0.
14	SPI2	rw	0x00	SPI2: SPI2 clock enable Set to ’1’ or cleared by software. 0：SPI2 clock disabled 1：SPI2 clock enabled
13：12	Reserved			Always read as 0.
11	WWDG	rw	0x00	WWDG: Window watchdog clock enable Set to ’1’ or cleared by software. 0: Window watchdog clock disabled 1: Window watchdog clock enabled
10：2	Reserved			Always read as 0.
1	TIM3	rw	0x00	TIM3: TIM3 clock enable Set to ’1’ or cleared by software. 0: TIM3 clock disabled 1: TIM3 clock enabled
0	TIM2	rw	0x00	TIM2: TIM2 clock enable Set to ’1’ or cleared by software. 0: TIM2 clock disabled 1: TIM2 clock enabled

5.3.9 Control status register(RCC_CSR)

Address offset：0x24

Reset value：0xXC00 0000

Access: 0 ≤ wait state ≤ 3, word, half-word and byte access

Wait states are inserted in case of successive accesses to this register.

Reset by system Reset, except reset flags by power Reset only.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWR RSTF	WWDG RSTF	IWDG RSTF	SFT RSTF	POR RSTF	PIN RSTF	Reserved									
r	r	r	r	r	r										

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

Bit	Field	Type	Reset	Description
31	LPWRRSTF	r	0x0x	LPWRRSTF: Low power reset flag Set by hardware when a Low-power management reset occurs. Cleared by writing to the RMVF bit. 0: No Low-power management reset occurred 1: Low-power management reset occurred
30	WWDGRSTF	r	0x0x	WDGRSTF: Window watchdog reset flag Set to '1' by hardware when a window watchdog reset occurs. Cleared by writing to the RMVF bit. 0: No window watchdog reset occurred 1: Window watchdog reset occurred
29	IWDGRSTF	r	0x0x	IWDGRSTF: Independent watchdog reset flag Set to '1' by hardware when an independent watchdog reset from VDD domain occurs. Cleared by writing to the RMVF bit. 0: No watchdog reset occurred 1: Watchdog reset occurred
28	SFTRSTF	r	0x0x	SFTRSTF: Software reset flag Set to '1' by hardware when a software reset occurs. Cleared by writing to the RMVF bit. Cleared by writing to the RMVF bit. 0: No software reset occurred 1: Software reset occurred
27	PORRSTF	r	0x01	PORRSTF: POR/PDR reset flag Set to '1' by hardware when a POR/PDR reset occurs. Cleared by writing to the RMVF bit. 0: No POR/PDR reset occurred 1: POR/PDR reset occurred

Bit	Field	Type	Reset	Description
26	PINRSTF	r	0x01	PINRSTF: PIN reset flag Set to '1' by hardware when a reset from the NRST pin occurs. Cleared by writing to the RMVF bit. 0: No reset from NRST pin occurred 1: Reset from NRST pin occurred
25 : 2	Reserved			Always read as 0.
1	LSIRDY	r	0x00	LSIRDY: Internal low-speed oscillator ready Set to '1' and cleared by software to indicate when the internal RC 40 kHz oscillator is ready. After the LSION bit is cleared, LSIRDY goes low after 3 internal RC 40 kHz oscillator clock cycles. 0: Internal RC 40 kHz oscillator not ready 1: Internal RC 40 kHz oscillator ready
0	LSION	rw	0x00	LSION: Internal low-speed oscillator enable Set to '1' or cleared by software. 0: Internal RC 40 kHz oscillator disabled 1: Internal RC 40 kHz oscillator enabled

5.3.10 AHB peripheral clock reset register(RCC_AHBRSTR)

Address offset：0x28

Reset value：0x0000 0000

Access: 0 ≤ wait state ≤ 3, word, half-word and byte access

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					HWDIV	Reserved					GPIOD	GPIOC	GPIOB	GPIOA	Res.
					rw						rw	rw	rw	rw	

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

Bit	Field	Type	Reset	Description
30: 27	Reserved			Always read as 0.
26	HWDIV	rw	0x00	HWDIV: HWDIV reset Set to '1' or cleared by software. 1：No effect 0：Reset HWDIV
25: 21	Reserved			Always read as 0.
20	GPIOD	rw	0x00	GPIOD: GPIOD reset Set to '1' or cleared by software. 0: No effect 1：Reset GPIOD

Bit	Field	Type	Reset	Description
19	GPIOC	rw	0x00	GPIOC: GPIOC reset Set to ’1’ or cleared by software. 0: No effect 1：Reset GPIOC
18	GPIOB	rw	0x00	GPIOB: GPIOB reset Set to ’1’ or cleared by software. 0: No effect 1：Reset GPIOB
17	GPIOA	rw	0x00	GPIOA：GPIOA reset Set to ’1’ or cleared by software. 0: No effect 1：Reset GPIOA
16: 0	Reserved			Always read as 0.

5.3.11 System configuration register(RCC_SYSCFG)

Address offset：0x40

Reset value：0x0000 1400

Access: 0 ≤ wait state ≤ 3, word, half-word and byte access

Reset by system Reset, except reset flags by power Reset only.

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.	OSC_LPFEN	Res.	OSC_ITRIM		OSC_RTRIM			Reserved							
	rw		rw	rw	rw	rw	rw								

Bit	Field	Type	Reset	Description
31: 15	Reserved			Always read as 0.
14	OSC_LPFEN	rw	0x00	OSC_LPFEN: External crystal low-pass filter enable 0: Disable 1: Enable
13	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
12: 11	OSC_ITRIM	rw	0x01	OSC_ITRIM: External crystal drive current calibration value If the crystal oscillator is abnormal, adjust the drive current to match the crystal oscillator 00：2mA 01：4mA 10：6mA 11：8mA
10: 8	OSC_RTRIM	rw	0x04	OSC_RTRIM: External crystal feedback resistance calibration value If the crystal oscillator is abnormal, adjust the drive current to match the crystal oscillator 000：100KΩ 001：200KΩ 010：500KΩ 011：700KΩ 100：1MΩ 101：2MΩ 110：4MΩ 111：8MΩ
7: 0	Reserved			Always read as 0.

6 General-purpose I/O(GPIO)

General-purpose I/O(GPIO)

6.1 GPIO functional description

Each of the general-purpose I/O ports has two 32-bit configuration registers (GPIOx_CRL, GPIOx_CRH), two 32-bit data registers (GPIOx_IDR and GPIOx_ODR), a 32-bit set/reset register (GPIOx_BSRR), a 16-bit reset register (GPIOx_BRR),a 32-bit locking register (GPIOx_LCKR) and two alternate-function select registers (GPIOx_AFRH) and (GPIOx_A-FRL).

Each port bits of the General Purpose IO (GPIO) Ports, can be individually configured by software in several modes:

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

Each I/O port bits is freely programmable, however, the I/O port registers have to be accessed as 32-bit words (half-word or byte accesses are not allowed).

The purpose of the GPIOx_BSRR and GPIOx_BRR registers is to allow atomic read/modify accesses to any of the GPIO registers. In this way, there is no risk that an IRQ occurs between the read and the modify access.

The following figure shows the basic structure of an I/O port bits.

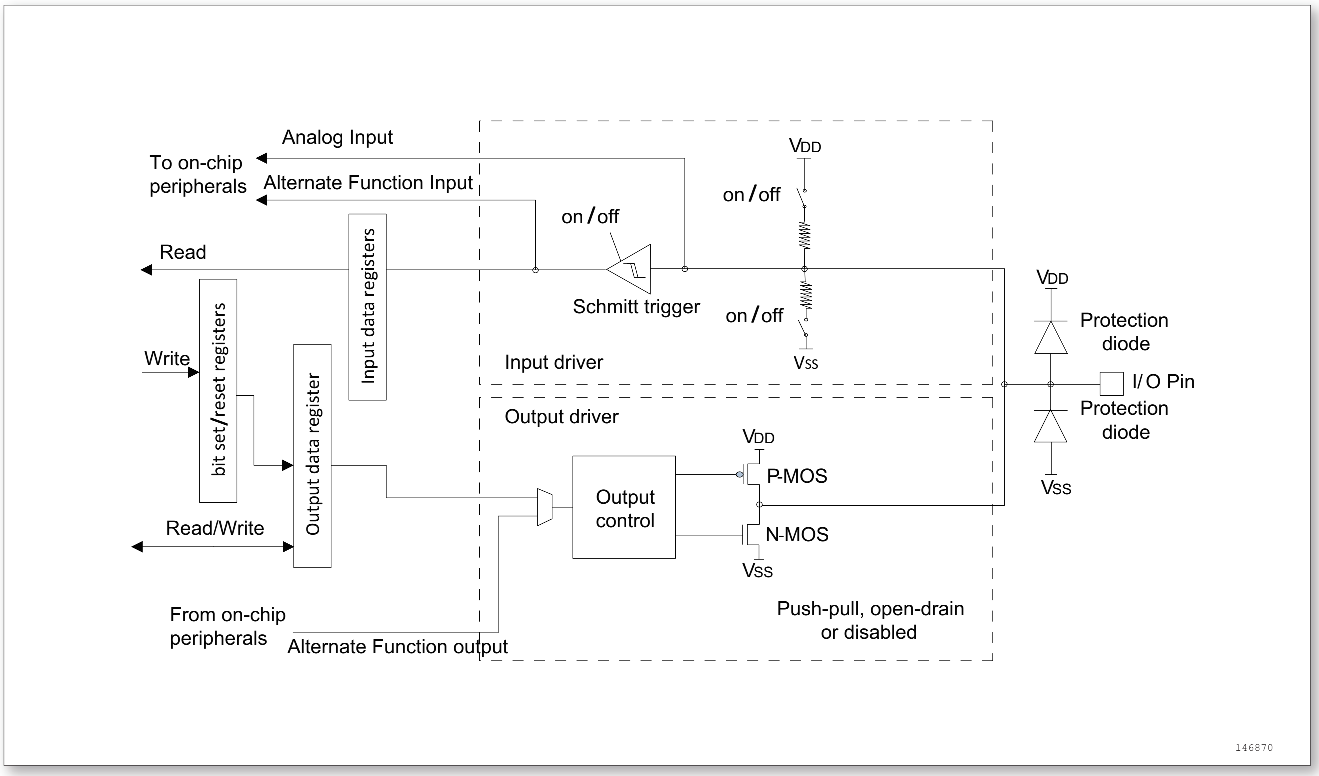


Figure 14. Basic Structure of I/O Port bits

Table 17. Port bits Configuration Table

Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-Pull	0	0	01		0 or 1
	Open-Drain		1			0 or 1
Alternate function output	Push-Pull	1	0			Not used
	Open-Drain		1			Not used
Input	Analog input	0	0	00		Not used
	Input floating		1			Not used
	Input pull-down	1	0			0
	Input pull-up					1

Table 18. Output MODE bits

MODE[1: 0]	Meaning
00	Reserved
01	Output

6.1.1 General-purpose I/O(GPIO)

During and just after reset, the alternate functions are not active and the I/O ports are configured in Input Floating mode (CNFx[1: 0] = 01, MODEx[1: 0] = 00).

The SWD pins are in input PU/PD after reset:

- PA14: SWCLK in PD
- PA13: SWDIO in PU

When configured as output, the value written to the Output Data Register (GPIOx_ODR) is output to the I/O pin. It is possible to use the output driver in Push-Pull mode or Open-Drain mode (only the N-MOS is activated when outputting 0).

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

All GPIO pins have an internal weak pull-up and weak pull-down that can be activated or not when configured as input.

6.1.2 Atomic bits set or reset

There is no need for the software to disable interrupts when programming the GPIOx_ODR at bits level:

it is possible to modify only one or several bits in a single AHB write access.

This is achieved by programming to '1' the bits Set/Reset Register (GPIOx_BSRR, or for reset only GPIOx_BRR) to select the bits to modify. The unselected bits will not be modified.

6.1.3 External interrupt/wakeup lines

All ports have external interrupt capability. To use external interrupt lines, the port must be configured in input mode. For more information on external interrupts, refer to 7.2 External interrupt/event controller (EXTI).

6.1.4 Alternate functions

It is necessary to program the Port bits Configuration Register before using a default alternate function.

- For alternate function inputs, the port must be configured in Input mode (floating, pullup or pull-down) and the input pin must be driven externally.

Note: It is also possible to emulate the AFI input pin by software by programming the GPIO controller. In this case, the port should be configured in Alternate Function Output mode. And obviously, the corresponding port should not be driven externally as it will be driven by the software using the GPIO controller.

- For alternate function outputs, the port must be configured in Alternate Function Output mode (Push-Pull or Open-Drain).
- For bidirectional Alternate Functions, the port bits must be configured in Alternate Function Output mode (Push-Pull or Open-Drain). In this case the input driver is configured in input floating mode.

If a port bits is configured as Alternate Function Output, this disconnects the output register and connects the pin to the output signal of an on-chip peripheral. If software configures a GPIO pin as Alternate Function Output, but peripheral is not activated, its output is not specified.

6.1.5 Software remapping of I/O alternate functions

To optimize the number of peripheral I/O functions for different device packages, it is possible to remap some alternate functions to some other pins. This is achieved by software, by programming the corresponding registers (refer to AFR registers).

In that case, the alternate functions are no longer mapped to their original assignations.

6.1.6 GPIO locking mechanism

The locking mechanism allows the IO configuration to be frozen. When the LOCK sequence has been applied on a port bits, it is no longer possible to modify the value of the port bits until the next reset.

6.1.7 Input configuration

When the I/O Port is programmed as Input:

- The Output Buffer is disabled
- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors are activated or not depending on input configuration (pull-up, pull-down or floating);
- 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 obtains the I/O State

The following figure shows the Input Configuration of the I/O Port bits:

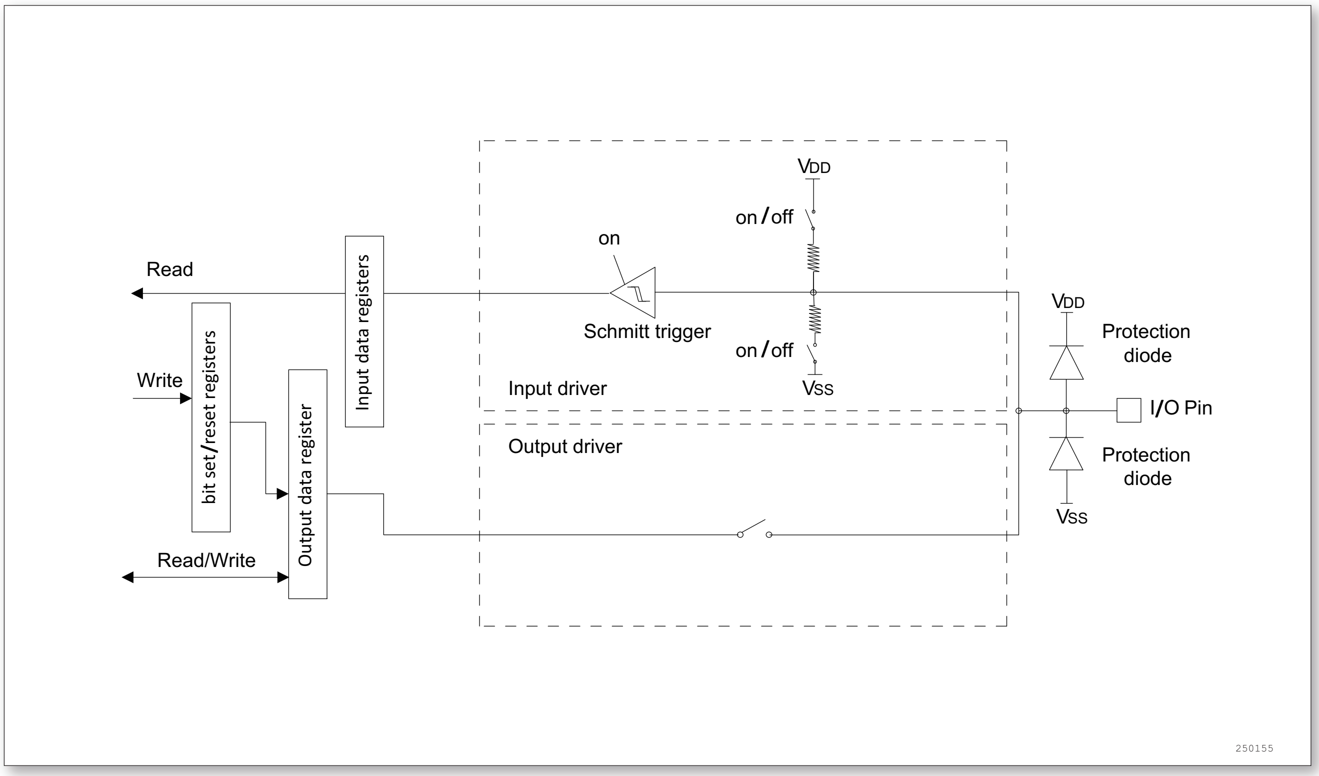


Figure 15. Input Floating/Pull Up/Pull Down Configurations

6.1.8 Output configuration

When the I/O Port is programmed as Output:

- The Output Buffer is enabled
  - Open drain mode: ’0’ on the output register activates N-MOS, and ’1’ on the output register places the port in a high-impedance state (P-MOS is never activated)
  - Push-pull mode: ’0’ on the output register activates N-MOS, and ’1’ on the output register activates P-MOS
- The Schmitt Trigger Input is activated.
- The weak pull-up and pull-down resistors are disabled.
- 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 in open drain mode
- A read access to the Output Data register gets the last written value in Push-Pull mode

The following figure shows the Output configuration of the I/O Port bits:

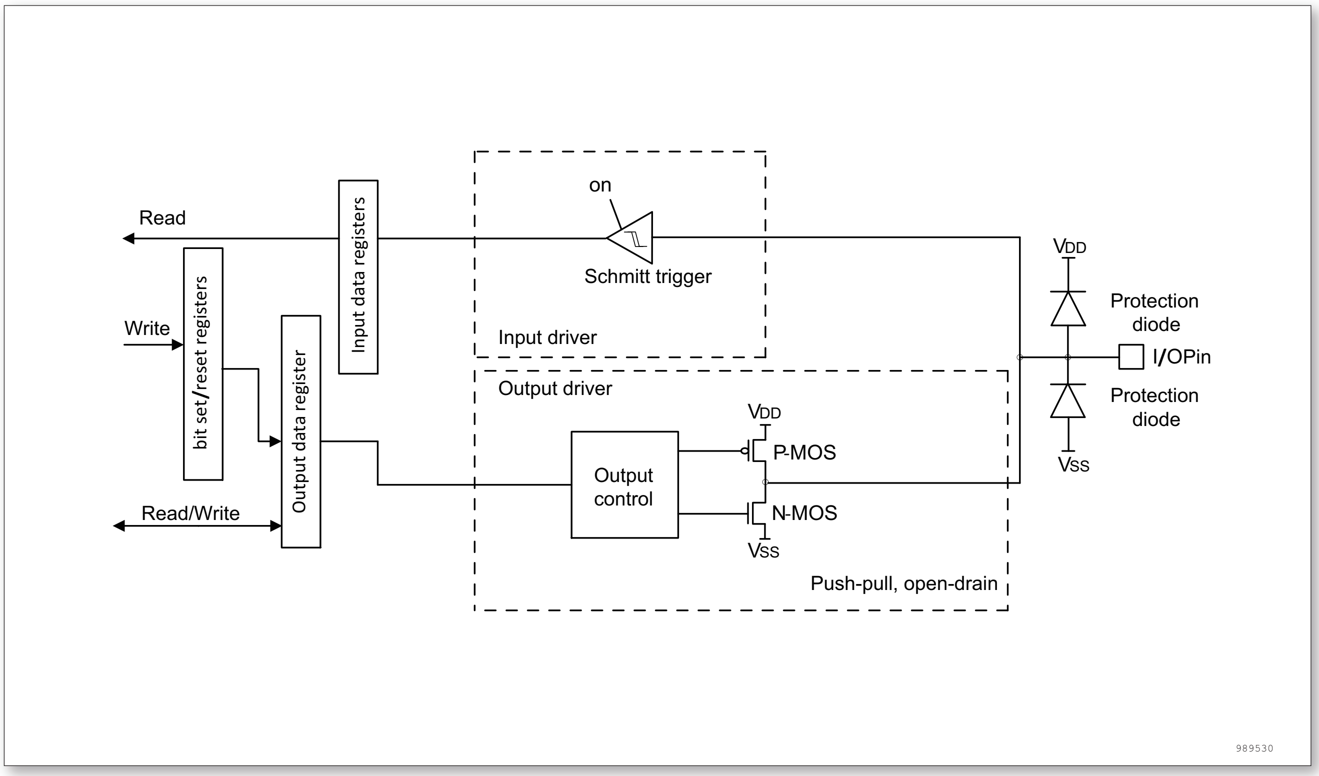


Figure 16. Output Configuration

6.1.9 Alternate functions configuration

When the I/O Port is programmed as Alternate Function:

- The Output Buffer is turned on in Open Drain or Push-Pull configuration
- The Output Buffer is driven by the signal coming from the peripheral (alternate function output)
- The Schmitt Trigger Input is activated
- The weak pull-up and pull-down resistors are disabled.

- 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 in open drain mode
- A read access to the Output Data register gets the last written value in Push-Pull mode

The following figure shows the Alternate Function Configuration of the I/O Port bits. Also, refer to AFIO registers for further information.

A set of Alternate Function I/O registers allow the user to remap some alternate functions to different pins.

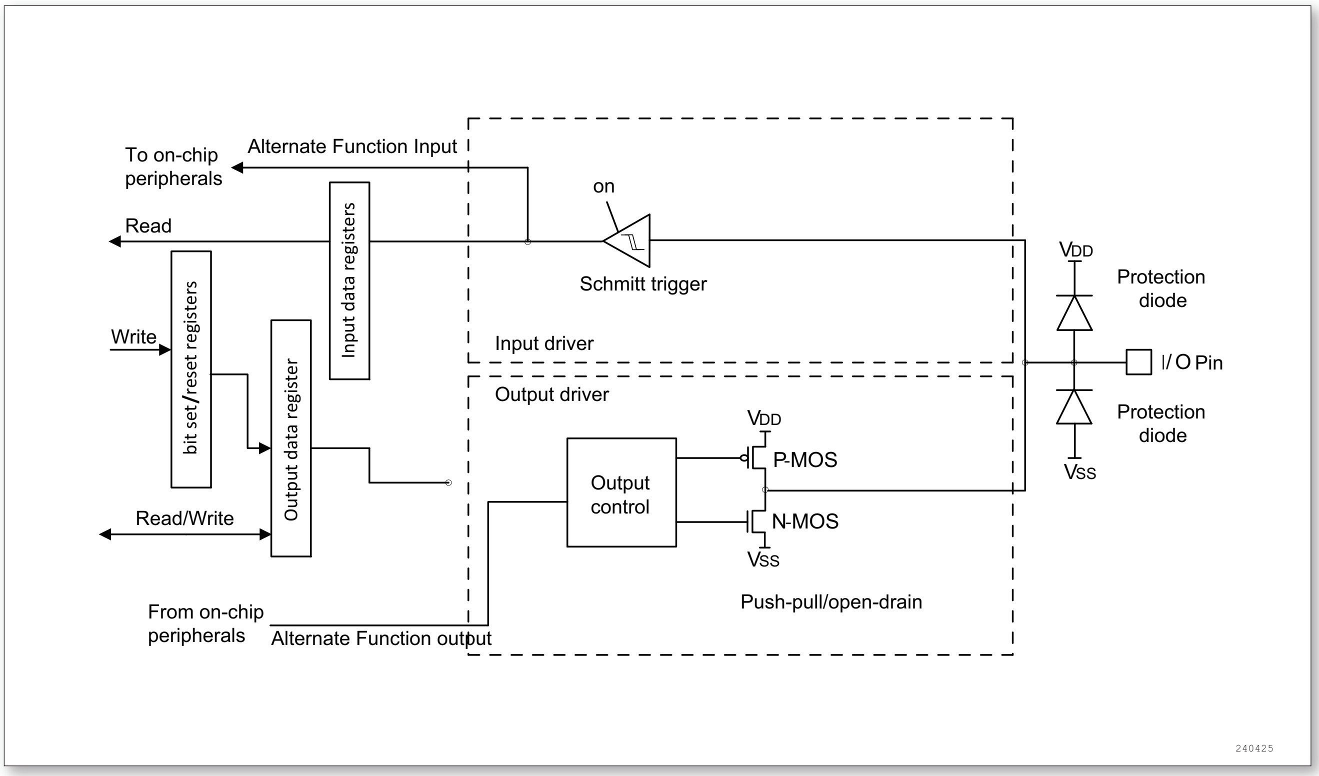


Figure 17. Alternate functions configuration

6.1.10 Analog configuration

When the I/O Port is programmed as Analog configuration:

- The Output Buffer is disabled.
- The Schmitt Trigger Input is de-activated providing zero consumption for every analog value of the I/O pin. The output of the Schmitt Trigger is forced to a constant value ’0’.
- The weak pull-up and pull-down resistors are disabled.
- Read access to the Input Data Register gets the value ’0’.

The following figure shows the high impedance-analog configuration of the I/O Port bits.

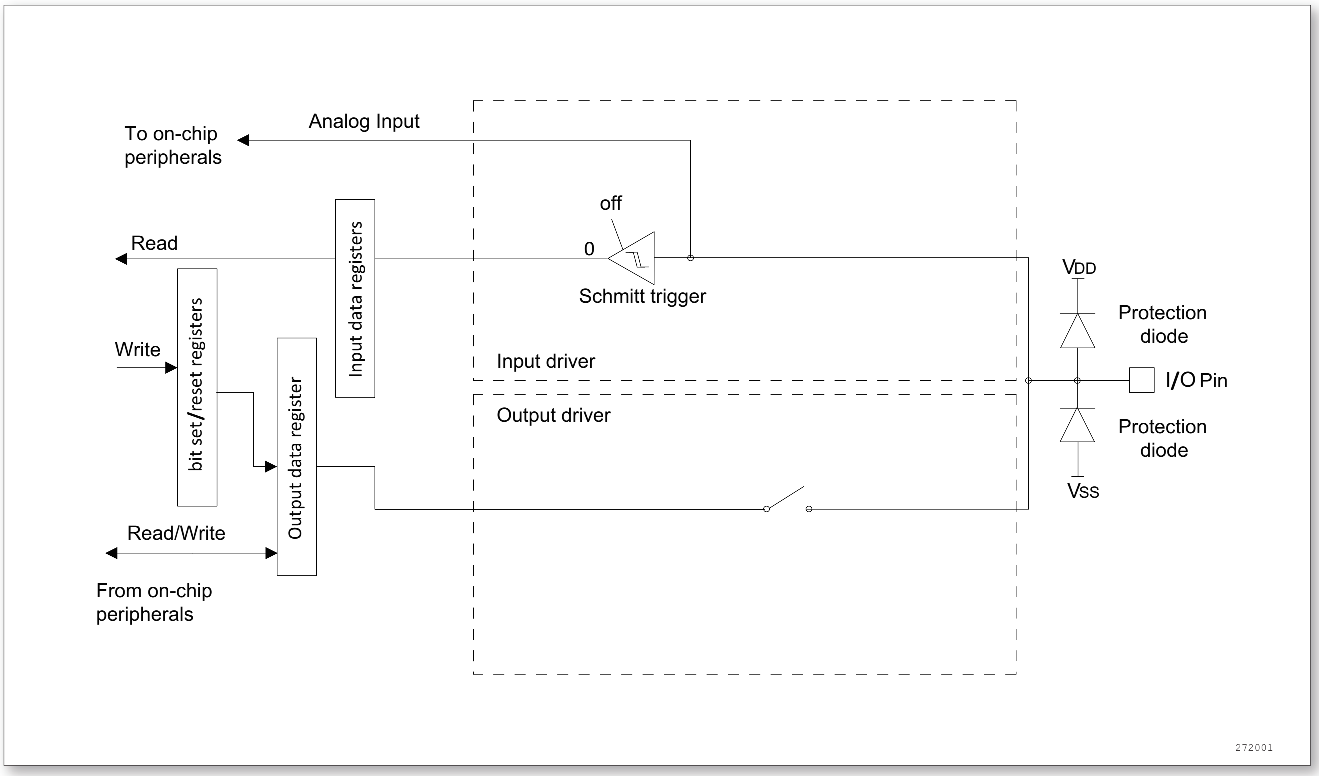


Figure 18. High Impedance-analog Configuration

6.1.11 GPIO configurations for device peripherals

The following tables give the GPIO configurations of the device peripherals:

Table 19. Advanced Timers TIM1

TIM1 pin	**configuration**	**GPIO configuration**
TIM1_CHx	Input capture channel x	Input floating
	Output compare channel x	Alternate function push-pull
TIM1_CHxN	Complementary output channel x	Alternate function push-pull
TIM1_BKIN	Break input	Input floating
TIM1_ETR	External trigger timer input	Input floating

Table 20. General-purpose timers TIM2/3/14/16/17

TIM2/3/14/16/17 pin	**configuration**	**GPIO configuration**
TIM2/3/14/16/17_CHx	Input capture channel x	Input floating
	Output compare channel x	Alternate function push-pull
TIM2/3_ETR	External trigger timer inputx	Input floating

Table 21. UART

UART pin	configuration	GPIO configuration
UARTx_TX	Serial-port sending	Alternate function push-pull
UARTx_RX	Serial-port receiving	Input floating/Input pull-up
UARTx_RTS	Hardware flow control	Alternate function push-pull
UARTx_CTS	Hardware flow control	Input floating/Input pull-up

Table 22. SPI

SPI pin	configuration	GPIO configuration
SPIx_SCK	Master	Alternate function push-pull
	Slave	Input floating
SPIx_MOSI	Full duplex/Master	Alternate function push-pull
	Full duplex/Slave	Input floating/Input pull-up
	Simplex bidirectional data wire/Master	Alternate function push-pull
	Simplex bidirectional data wire/Slave	Not used. Can be used as GPIO
SPIx_MISO	Full duplex/Master	Input floating/Input pull-up
	Full duplex/Slave	Alternate function push-pull
	Simplex bidirectional data wire/Master	Not used. Can be used as GPIO
	Simplex bidirectional data wire/Slave	Alternate function push-pull
SPIx_NSS	Hardware master/Slave	Input floating/Input pull-up/Input pull-down
	Hardware master /NSS output enabled	Alternate function push-pull
	Software	Not used. Can be used as GPIO

Table 23. I2C

I2C pin	configuration	GPIO configuration
I2Cx_SCL	I2C clock	Alternate function open drain
I2Cx_SDA	I2C data	Alternate function open drain

Table 24. ADC

ADC pin	GPIO configuration
ADC	Analog input

Table 25. Other I/Os

pin	configuration	GPIO configuration
MCO	Clock output	Alternate function push-pull
EXTI input lines	External input interrupts	Input floating/Input pull-up/ input pull-down

6.2 Alternate function I/O and debug configuration (AFIO)

To optimize the number of peripherals, it is possible to remap some alternate functions to some other pins. This is achieved by software, by programming the alternate function register (AFR). In this case, the alternate functions are no longer mapped to their original assignations.

6.2.1 Using OSC_IN/OSC_OUT pins as GPIO ports PD0/PD1

The external oscillator pin OSC_IN/OSC_OUT can be used as the PD0/PD1 of GPIO by disabling the internal high-speed clock and setting the alternate function register (AFR).

Note: The external interrupt/event function is not remapped.

6.2.2 SWD alternate function remapping

The debug interface signals are mapped on the GPIO ports as shown in the following table.

Table 26. Debug Interface Signals

Alternate functions	GPIO port
SWDIO	PA13
SWCLK	PA14

To optimize the number of free GPIOs during debugging, this mapping can be configured by setting the AF remap and alternate function register, so as to change the above-mentioned remapping configuration.

6.3 GPIO register description

Table 27. Overview of GPIO Registers

Offset	Acronym	Register Name	Reset	Section
0x00	GPIOx_CRL	Port configuration low register	0x44444444	section 6.3.1
0x04	GPIOx_CRH	Port configuration high register	0x44444444	section 6.3.2
0x08	GPIOx_IDR	Port input data register	0x0000XXXX	section 6.3.3
0x0C	GPIOx_ODR	Port output data register	0x00000000	section 6.3.4
0x10	GPIOx_BSRR	Port set/reset register	0x00000000	section 6.3.5

Offset	Acronym	Register Name	Reset	Section
0x14	GPIOx_BRR	Port bits reset register	0x00000000	section 6.3.6
0x18	GPIOx_LCKR	Port configuration lock register	0x00000000	section 6.3.7
0x20	GPIOx_AFRL	Port alternate-function register low	0x00000000	section 6.3.8
0x24	GPIOx_AFRH	Port alternate-function register high	0x00000000	section 6.3.9

6.3.1 Port configuration low register(GPIOx_CRL)(x = A..D)

Offset address: 0x00

Reset value:0x4444 4444

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	Type	Reset	Description
31: 30 27: 26 23: 22 19: 18 15: 14 11: 10 7: 6 3: 2	CNFy	rw	0x01	Port x configuration bits(0⋯7) These bits are written by software to configure the corresponding I/O port. Refer to Table 17 : Port pin configuration In input mode (MODE = 00): 00: Analog mode 01: Floating input 10: Input with pull-up / pull-down 11: Reserved In output mode (MODE > 00): 00: General-purpose output push-pull 01: General-purpose output Open-drain 10: Alternate functionsoutput push-pull 11: Alternate functionsoutput Open-drain
29: 28 25: 24 21: 20 17: 16 13: 12 9: 8 5: 4 1: 0	MODEy	rw	0x00	Port x mode bits(y = 0⋯7) These bits are written by software to configure the corresponding I/O port. Refer to Table 17 : Port pin configuration 00: Input mode (reset state) 01: Output mode 10: Reserved 11: Reserved

6.3.2 Port configuration high register(GPIOx_CRH)(x = A..D)

Offset address: 0x04

Reset value: 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	Type	Reset	Description
31: 30 27: 26 23: 22 19: 18 15: 14 11: 10 7: 6 3: 2	CNFy	rw	0x01	Port x configuration bits(8⋯15) These bits are written by software to configure the corresponding I/O port. Refer to Table 17 : Port pin configuration In input mode (MODE = 00): 00: Analog mode 01: Floating input 10: Input with pull-up / pull-down 11: Reserved In output mode (MODE[1: 0] > 00): 00: General-purpose output push-pull 01: General-purpose output Open-drain 10: Alternate functionsoutput push-pull 11: Alternate functionsoutput Open-drain
29: 28 25: 24 21: 20 17: 16 13: 12 9: 8 5: 4 1: 0	MODEy	rw	0x00	Port x mode bits(y = 8⋯15) These bits are written by software to configure the corresponding I/O port. Refer to Table 17 : Port pin configuration 00: Input mode (reset state) 01: Output mode 10: Reserved 11: Reserved

6.3.3 Port input data register(GPIOx_IDR)(x = A..D)

Offset address: 0x08

Reset value: 0x0000 XXXX

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

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			Always read as 0.
15: 0	IDRy	r	0xXXXX	Port input data(y = 0..15) These bits are read only and can be accessed in Word (16 bits) mode only. They contain the input value of the corresponding I/O port.

6.3.4 Port output data register(GPIOx_ODR)(x = A..D)

Offset address: 0x0C

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
ODR															
rw															

Bit	Field	Type	Reset	Description
31: 16	Reserved			Always read as 0.
15: 0	ODRy	rw	0x0000	Port output data(y = 0..15) These bits can be read and written by software and can be accessed in Word (16 bits) mode only. Note: For GPIOx_BSRR (x = A-E), the ODR bits can be individually set and cleared.

6.3.5 Port set/reset register(GPIOx_BSRR)(x = A..D)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31: 16	BRy	w	0x0000	Port x Reset bit y (y =0-15) These bits are write-only and can be accessed in Word (16 bits) mode only. 0: No action on the corresponding ODRy bit 1: Reset the corresponding ODRy bit
15: 0	BSy	w	0x0000	Port x Set bit y (y =0-15) These bits are write-only and can be accessed in Word (16 bits) mode only. 0: No action on the corresponding ODRy bit 1: Set the corresponding ODRy bit to ’1’

6.3.6 Port bits reset register(GPIOx_BRR)(x = A..D)

Offset address: 0x14

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			Always read as 0
15: 0	BRy	w	0x0000	Port x Reset bit y (y =0-15) These bits are write-only and can be accessed in Word (16 bits) mode only. 0: No action on the corresponding ODRy bit 1: Reset the corresponding ODRy bit

6.3.7 Port configuration lock register(GPIOx_LCKR)(x = A..D)

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

GPIO. During the write sequence, the value of LCKR[15:0] must not change. When the LOCK sequence has been applied on a port bit it is no longer possible to modify the value of the port bit until the next reset. Each lock bit freezes the corresponding 4 bits of the control register (CRL, CRH).

Address offset: 0x18

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31: 17	Reserved			Always read as 0
16	LCKK	rw	0x00	Lock key This bit can be read anytime. It can only be modified using the Lock Key Writing Sequence. 0: Port configuration ock key not active 1: Port configuration ock key active, GPIOx_LCKR register is locked until the next reset. LOCK key writing sequence: Write 1->Write 0->Write 1->Read 0->Read 1 The last read is optional but confirms that the lock is active. Note: During the LOCK Key Writing sequence, the value of LCK[15:0] must not change. Any error in the lock sequence will abort the lock.
15: 0	LCKy	rw	0x00	Port x Lock bits y(y = 0···15) These bits are read and written but can only be written when the LCKK bit is 0. 0: Port configuration not locked 1: Port configuration locked

6.3.8 Port alternate-function register low(GPIOx_AFRL)(x = A..D)

Offset address: 0x20

Reset value: 0x0000 0000

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	Type	Reset	Description
31: 0	AFRy	rw	0x0000 0000	Port x alternate-function Select bit y (y =0-7) These bits can be written by software to configuration IOAl-ternate functions. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7

6.3.9 Port alternate-function register high(GPIOx_AFRH)(x = A..D)

Offset address: 0x24

Reset value: 0x0000 0000

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	Type	Reset	Description
31: 0	AFRy	rw	0x0000 0000	Port x alternate-function Select bit y (y =8-15) These bits can be written by software to configuration IOAl-ternate functions. 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7

7 Interrupts and events(EXTI)

Interrupts and events(EXTI)

7.1 Nested Vectored Interrupt Controller

Features

- Interrupts can be masked (except NMI)
- 16 programmable priority levels (4 bits of interrupt priority are used)
- Low-latency exception and interrupt handling
- Power management control
- Implementation of System Control Registers

The NVIC and the processor core interface are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts.

All interrupts including the core exceptions are managed by the NVIC. For more information on exceptions and NVIC programming, refer to CPU programming manual.

7.1.1 SysTick calibration value register

The SysTick calibration value is set to 9000, which gives a reference time base of 1 ms with the SysTick clock set to 3 MHz (HCLK/8, HCLK = 24 MHz).

7.1.2 Interrupt and exception vectors

The following tables are the vector tables for the product series.

Table 28. Vectors for the Product Series

Position	Priority	Type of priority	Acronym	Description	Address
	-	-	-	Reserved	0x0000_0000
	-3	Fixed	Reset	Reset	0x0000_0004
	-2	Fixed	NMI	Non maskable interrupt. The RCC Clock Security System (CSS) is linked to the NMI vector.	0x0000_0008
	-1	Fixed	HardFault	All class of fault	0x0000_000C
	0	Settable	MemManage	Memory management	0x0000_0010
	1	Settable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014

Position	Priority	Type of priority	Acronym	Description	Address
	2	Settable	UsageFault	Undefined instruction or illegal state	0x0000_0018
	-	-	-	Reserved	0x0000_001C ~ 0x0000_002B
	3	Settable	SVCall	System service call via SWI instruction	0x0000_002C
	4	Settable	DebugMonitor	Debug Monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	Settable	PendSV	Pendable request for system service	0x0000_0038
	6	Settable	SysTick	System tick timer	0x0000_003C
0	7	Settable	WWDG/IWDG	Watchdog interrupt	0x0000_0040
1	8	Settable	PVD	PVD through EXTI 16 Line detection interrupt	0x0000_0044
2	9	-	-	Reserved	0x0000_0048
3	10	Settable	Flash	Flash global interrupt	0x0000_004C
4	11	Settable	RCC	RCC global interrupt	0x0000_0050
5	12	Settable	EXTI0_1	EXTI line [1:0] interrupt	0x0000_0054
6	13	Settable	EXTI2_3	EXTI line [3:2] interrupt	0x0000_0058
7	14	Settable	EXTI4_15	EXTI line [15:4] interrupt	0x0000_005C
8	15	Settable	HWDIV	HWDIV global interrupt	0x0000_0060
9	16	Settable	DMA1 Channel 1	DMA1 Channel 1 global interrupt	0x0000_0064
10	17	Settable	DMA1 Channel 2_3	DMA1 Channel 2_3 global interrupt	0x0000_0068
11	18	Settable	DMA1 Channel 4_5	DMA1 Channel 4_5 global interrupt	0x0000_006C
12	19	Settable	ADC_COMP	ADC and COMP interrupt (EXTI19)	0x0000_0070
13	20	Settable	TIM1_BRK_UP_TRG_COM	TIM1 Break, Update, Trigger and Commutation interrupt	0x0000_0074
14	21	Settable	TIM1_CC	TIM1 Capture and Compare interrupt	0x0000_0078
15	22	Settable	TIM2	TIM2 global interrupt	0x0000_007C
16	23	Settable	TIM3	TIM3 global interrupt	0x0000_0080
17	-	-	-	Reserved	0x0000_0084
18	-	-	-	Reserved	0x0000_0088
19	26	Settable	TIM14	TIM14 global interrupt	0x0000_008C
20	-	-	-	Reserved	0x0000_0090

Position	Priority	Type of priority	Acronym	Description	Address
21	28	Settable	TIM16	TIM16 global interrupt	0x0000_0094
22	29	Settable	TIM17	TIM17 global interrupt	0x0000_0098
23	30	Settable	I2C1	I2C1 global interrupt	0x0000_009C
24	-	-	-	Reserved	0x0000_00A0
25	32	Settable	SPI1	SPI1 global interrupt	0x0000_00A4
26	33	Settable	SPI2	SPI2 global interrupt	0x0000_00A8
27	34	Settable	UART1	UART1 global interrupt	0x0000_00AC
28	35	Settable	UART2	UART2 global interrupt	0x0000_00B0
29	-	-	-	Reserved	0x0000_00B4
30	-	-	-	Reserved	0x0000_00B8
31	-	-	-	Reserved	0x0000_00BC

7.2 External interrupt/event controller (EXTI)

The external interrupt/event controller, consisting of edge detectors, is used for managing external and internal asynchronous events/interrupts, and for corresponding event requests sent to the CPU/ interrupt controller, and a wake-up request transmitted to the power manager.

The event/interrupt requests can be generated by the edge detector. Each input line can be independently configured to select the type (pulse or pending) and the corresponding trigger event (rising or falling or both). Each line can also be masked independently. A pending register maintains the status line of the interrupt requests.

7.2.1 Main features

The main features of EXTI controller are as follows:

- Independent trigger and mask on each interrupt/event line
- Dedicated status bit for each interrupt line
- Generation of software event/interrupt requests
- Detection of external signal with pulse width lower than APB2 clock period. Refer to the electrical characteristics section of the datasheet for details on this parameter.

7.2.2 Block diagram

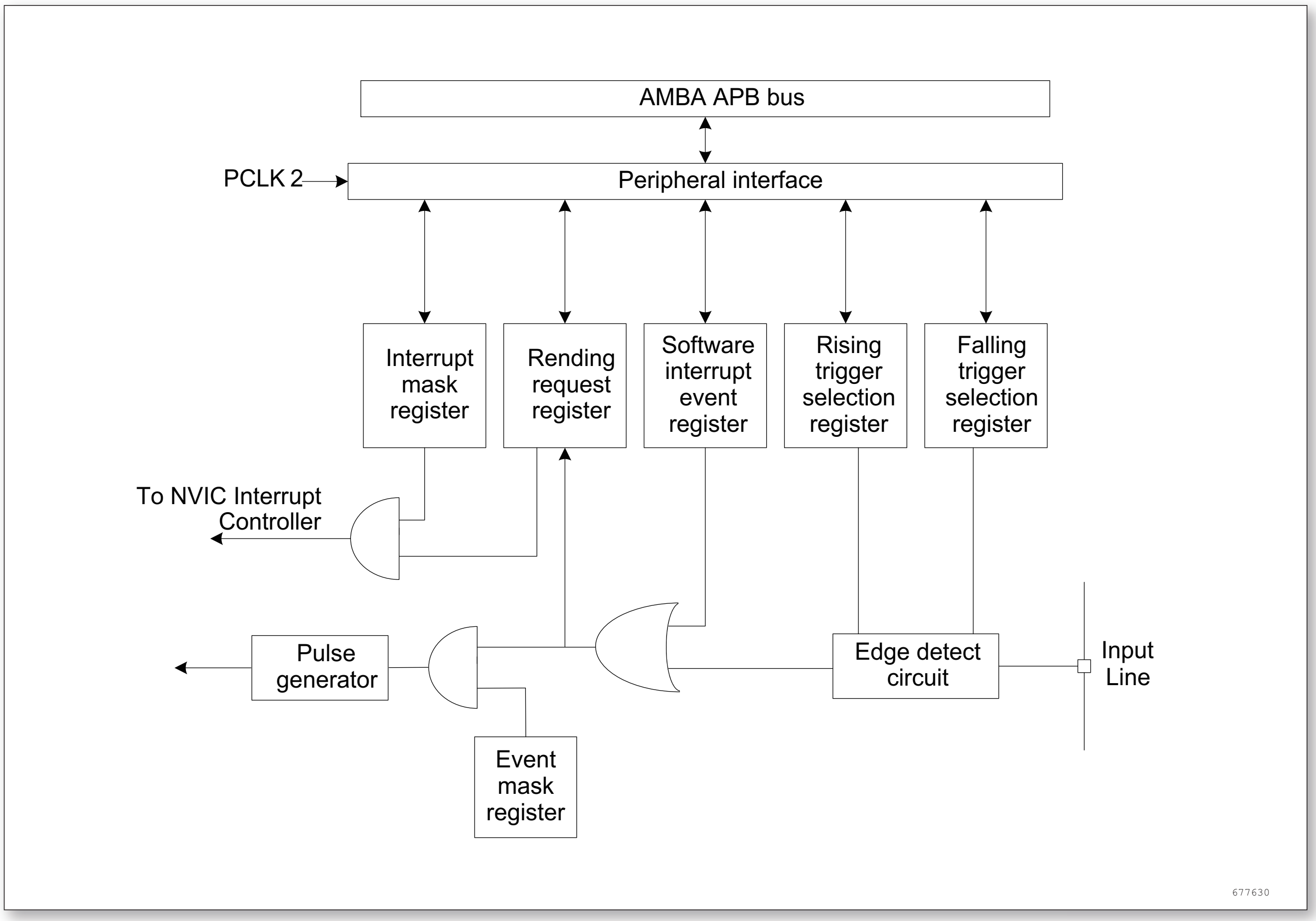


Figure 19. External Interrupt/Event Controller Block Diagram

7.2.3 Wakeup event management

The device(WFE) is able to handle external or internal events (WFE). The wakeup event can be generated either by:

- Enabling an interrupt in the peripheral control register but not in the NVIC, and enabling the SEVONPEND bit in the CPU's Control register.
  When the CPU resumes from WFE, the peripheral interrupt pending bit and the peripheral NVIC IRQ channel pending bit (in the NVIC interrupt clear pending register) have to be cleared.
- Or configuring an external or internal EXTI line in event mode. When the CPU resumes from WFE, it is not necessary to clear the peripheral interrupt pending bit or the NVIC IRQ channel pending bit as the pending bit corresponding to the event line is not set.

To use an external line as a wakeup event, refer to Section ”Functional Description”.

7.2.4 Functional description

To generate an interrupt, an interrupt line should be configured and enabled. This is done by programming the two trigger registers with the desired edge detection and by enabling the interrupt request by writing a ‘1’ to the corresponding bit in the interrupt mask register. When the selected edge occurs on the external interrupt line, an interrupt request is generated. The pending bit corresponding to the interrupt line is also set. This request is reset by writing a ‘1’ in the pending register.

To generate an event, an event line should be configured and enabled. This is done by programming the two trigger registers with the desired edge detection and by enabling the event request by writing a ‘1’ to the corresponding bit in the event mask register. When the selected edge occurs on the event line, an event pulse is generated. The pending bit corresponding to the event line is not set.

An interrupt/event request can also be generated by software by writing a ‘1’ in the software interrupt/event register.

Hardware interrupt selection

To configure the several lines as interrupt sources, use the following procedure:

- Configure the mask bits of the Interrupt lines (EXTI_IMR)
- Configure the Trigger Selection bits of the Interrupt lines (EXTI_RTSR and EXTI_FTSR)
- Configure the enable and mask bits that control the NVIC IRQ channel mapped to the External Interrupt Controller (EXTI) so that an interrupt coming from one of the lines can be correctly acknowledged.

Hardware event selection

To configure the several lines as event sources, use the following procedure:

- Configure the mask bits of the Event lines (EXTI_EMR)
- Configure the Trigger Selection bits of the Event lines (EXTI_RTSR and EXTI_FTSR)

Software interrupt/event selection

The several lines can be configured as software interrupt/event lines. The following is the procedure to generate a software interrupt.

- Configure the mask bits of the Interrupt/Event lines (EXTI_IMR, EXTI_EMR)
- Set the required bit of the software interrupt register (EXTI_SWIER)

7.2.5 External interrupt/event line mapping

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

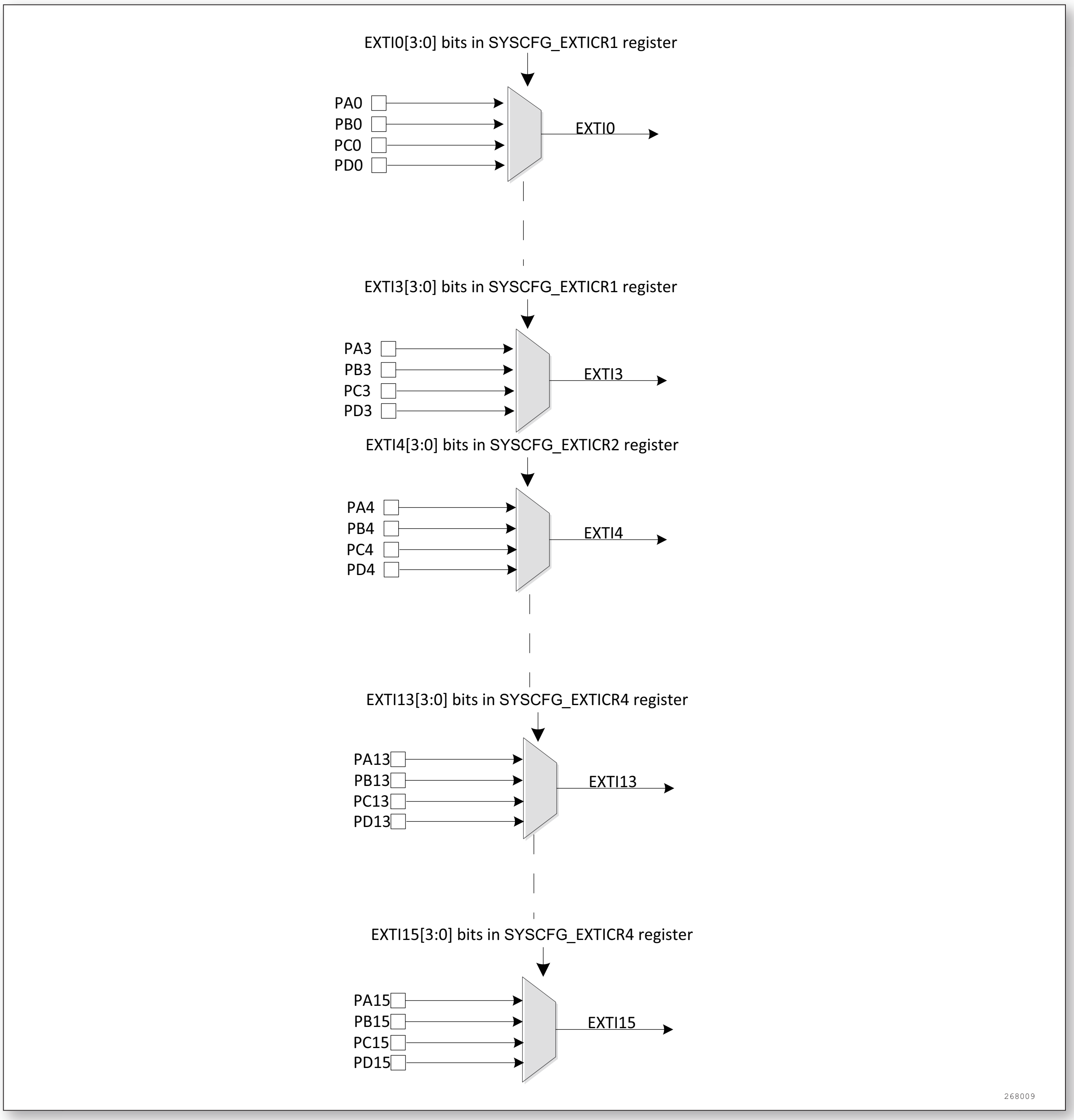

EXTI0[3:0] bits in SYSCFG_EXTICR1 register
PA0
PB0
PC0
PD0
EXTI0
EXTI3[3:0] bits in SYSCFG_EXTICR1 register
PA3
PB3
PC3
PD3
EXTI3
EXTI4[3:0] bits in SYSCFG_EXTICR2 register
PA4
PB4
PC4
PD4
EXTI4
EXTI13[3:0] bits in SYSCFG_EXTICR4 register
PA13
PB13
PC13
PD13
EXTI13
EXTI15[3:0] bits in SYSCFG_EXTICR4 register
PA15
PB15
PC15
PD15
EXTI15
268009

Figure 20. External Interrupt/Event GPIO Mapping

Note: GPIO corresponding to the above figure may differ due to actual chip package, and the actual package shall prevail.

The other EXTI lines are connected as follows:

- EXTI line 16 is connected to the PVD output
- EXTI line 19 is connected to Comparator 1 output
- EXTI line 24 is connected to IWDG interrupt

7.3 EXTI register description

Table 29. EXTI Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	EXTI_IMR	Interrupt Mask Register	0x00000000	section 7.3.1
0x04	EXTI_EMR	Event Mask Register	0x00000000	section 7.3.2
0x08	EXTI_RTSR	Rising Trigger Selection Register	0x00000000	section 7.3.3
0x0C	EXTI_FTSR	Falling Trigger Selection Register	0x00000000	section 7.3.4
0x10	EXTI_SWIER	Software Interrupt Event Register	0x00000000	section 7.3.5
0x14	EXTI_PR	Pending Register	0x00000000	section 7.3.6

7.3.1 Interrupt Mask Register(EXTI_IMR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							IMR24	Reserved				IMR19	Reserved		IMR16
							rw					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	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	IMRx	rw	0x00	Interrupt Mask on line x 1 = Interrupt request from Line x is not masked 0 = Interrupt request from Line x is masked
23 : 20	Reserved			Always read as 0.
19	IMRx	rw	0x00	Interrupt Mask on line x 1 = Interrupt request from Line x is not masked 0 = Interrupt request from Line x is masked
18 : 17	Reserved			Always read as 0.
16 : 0	IMRx	rw	0x00	Interrupt Mask on line x 1 = Interrupt request from Line x is not masked 0 = Interrupt request from Line x is masked

7.3.2 Event Mask Register(EXTI_EMR)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							EMR24	Reserved				EMR19	Reserved		EMR16
							rw					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	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	EMRx	rw	0x00	Event Mask on line x 1 = Event request from Line x is not masked 0 = Event request from Line x is masked
23 : 20	Reserved			Always read as 0.
19	EMRx	rw	0x00	Event Mask on line x 1 = Event request from Line x is not masked 0 = Event request from Line x is masked
18 : 17	Reserved			Always read as 0.
16 : 0	EMRx	rw	0x00	Event Mask on line x 1 = Event request from Line x is not masked 0 = Event request from Line x is masked

7.3.3 Rising Trigger Selection Register(EXTI_RTSR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TR24	Reserved				TR19	Reserved		TR16
							rw					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	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	TRx	rw	0x00	Rising trigger event configuration bit of line x 1 = Rising trigger enabled (for Event and Interrupt) for input line 0 = Rising trigger disabled (for Event and Interrupt) for input line
23 : 20	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
19	TRx	rw	0x00	Rising trigger event configuration bit of line x 1 = Rising trigger enabled (for Event and Interrupt) for input line 0 = Rising trigger disabled (for Event and Interrupt) for input line
18 : 17	Reserved			Always read as 0.
16 : 0	TRx	rw	0x00	Rising trigger event configuration bit of line x 1 = Rising trigger enabled (for Event and Interrupt) for input line 0 = Rising trigger disabled (for Event and Interrupt) for input line

7.3.4 Falling Trigger Selection Register(EXTI_FTSR)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TR24	Reserved				TR19	Reserved		TR16
							rw					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	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	TRx	rw	0x00	Falling trigger event configuration bit of line x 1 = Falling trigger enabled (for Event and Interrupt) for input line 0 = Falling trigger disabled (for Event and Interrupt) for input line
23 : 20	Reserved			Always read as 0.
19	TRx	rw	0x00	Falling trigger event configuration bit of line x 1 = Falling trigger enabled (for Event and Interrupt) for input line 0 = Falling trigger disabled (for Event and Interrupt) for input line
18 : 17	Reserved			Always read as 0.
16 : 0	TRx	rw	0x00	Falling trigger event configuration bit of line x 1 = Falling trigger enabled (for Event and Interrupt) for input line 0 = Falling trigger disabled (for Event and Interrupt) for input line

7.3.5 Software Interrupt Event Register(EXTI_SWIER)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							SWIER24	Reserved				SWIER19	Reserved		SWIER16
							rw					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	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	SWIERx	rw	0x00	Software interrupt on line x If the interrupt is enabled on this line in EXTI_INTMASK and EXTI_EVNTMASK, write ’1’ to this bit when it is set to ’0’, so as to set the corresponding pending bit in EXTI_PR, and to generate an interrupt request. Note: This bit is cleared by clearing the corresponding bit of EXTI_PEND (by writing a ’1’ into the bit)
23 : 20	Reserved			Always read as 0.
19	SWIERx	rw	0x00	Software interrupt on line x If the interrupt is enabled on this line in EXTI_INTMASK and EXTI_EVNTMASK, write ’1’ to this bit when it is set to ’0’, so as to set the corresponding pending bit in EXTI_PR, and to generate an interrupt request. Note: This bit is cleared by clearing the corresponding bit of EXTI_PEND (by writing a ’1’ into the bit)
18 : 17	Reserved			Always read as 0.
16 : 0	SWIERx	rw	0x00	Software interrupt on line x If the interrupt is enabled on this line in EXTI_INTMASK and EXTI_EVNTMASK, write ’1’ to this bit when it is set to ’0’, so as to set the corresponding pending bit in EXTI_PR, and to generate an interrupt request. Note: This bit is cleared by clearing the corresponding bit of EXTI_PEND (by writing a ’1’ into the bit)

7.3.6 Pending register(EXTI_PR)

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							PR24	Reserved				PR19	Reserved		PR16
							rc_w1					rc_w1			rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit	Field	Type	Reset	Description
31 : 25	Reserved			Always read as 0.
24	PRx	rc_w1	0x00	Pending bit 1 = selected trigger request occurred 0 = No trigger request occurred This bit is set to ’1’ when the selected edge event arrives on the external interrupt line. This bit is cleared by writing a ‘1’into the bit or by changing the polarity of edge detection.
23 : 20	Reserved			Always read as 0.
19	PRx	rc_w1	0x00	Pending bit 1 = selected trigger request occurred 0 = No trigger request occurred This bit is set to ’1’ when the selected edge event arrives on the external interrupt line. This bit is cleared by writing a ‘1’into the bit or by changing the polarity of edge detection.
18 : 17	Reserved			Always read as 0.
16 : 0	PRx	rc_w1	0x00	Pending bit 1 = selected trigger request occurred 0 = No trigger request occurred This bit is set to ’1’ when the selected edge event arrives on the external interrupt line. This bit is cleared by writing a ‘1’into the bit or by changing the polarity of edge detection.

8 Direct memory access controller(DMA)

Direct memory access controller(DMA)

8.1 DMA introduction

Direct memory access (DMA) is used in order to provide high-speed data transfer between peripherals and memory as well as memory to memory. Data can be quickly moved by DMA without any CPU actions. This keeps CPU resources free for other operations.

The DMA controller has 5 channels, each dedicated to managing memory access requests from several peripherals.

8.2 DMA main features

- 5 independently configurable channels.
- Each channel is connected to dedicated hardware DMA requests, software trigger is also supported on each channel. This configuration is done by software.
- Priorities between requests from 5 channels are software-programmable (4 levels consisting of very high, high, medium, low) or hardware in case of equality (request 0 has priority over request 1, etc.)
- Independent source and destination transfer size (byte, half word, word), emulating packing and unpacking. Source/destination addresses must be aligned on the data size.
- Support for circular buffer management.
- 3 event flags (DMA Half Transfer, DMA Transfer complete and DMA Transfer Error) logically or together in a single interrupt request for each channel
- Memory-to-memory transfer
- Peripheral-to-memory and memory-to-peripheral transfers
- Access to Flash, SRAM, SRAM peripherals, APB1, APB2 and AHB peripherals as source and destination
- Programmable number of data to be transferred: up to 65536.

The block diagram is shown in the following figure:

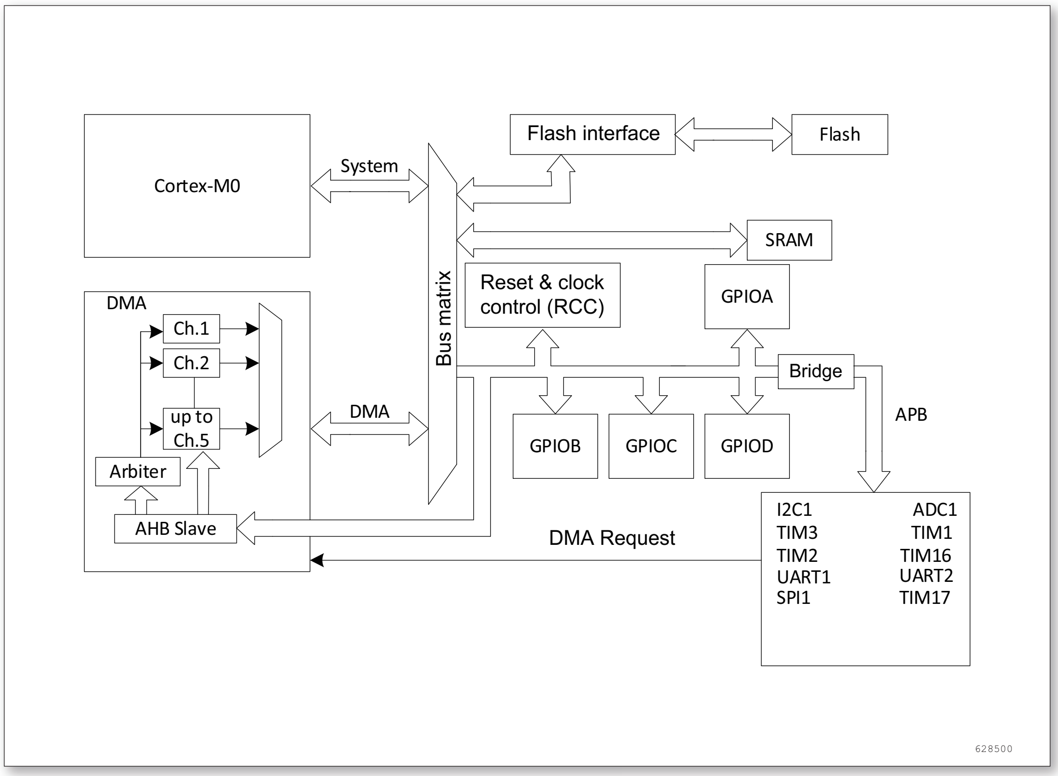


Figure 21. DMA Block Diagram

8.3 Functional description

The DMA controller performs direct memory transfer by sharing the system bus with the CPU. The DMA request may stop the CPU access to the system bus for some bus cycles, when the CPU and DMA are targeting the same destination (RAM or peripheral). The bus arbiter implements round-robin scheduling, thus ensuring at least half of the system bus bandwidth (both to memory and peripheral) for the CPU.

8.3.1 DMA transactions

After an event, the peripheral sends a request signal to the DMA Controller. The DMA controller serves the request depending on the channel priorities. As soon as the DMA Controller accesses the peripheral, an Acknowledge is sent to the peripheral by the DMA Controller. The peripheral releases its request as soon as it gets the Acknowledge from the DMA Controller. Once the request is deasserted by the peripheral, and the DMA Controller release the Acknowledge. If there are more requests, the peripheral can initiate the next transaction.

In summary, each DMA transfer consists of three operations:

1. The loading of data from the peripheral data register or a location in memory addressed through DMA_CMARx register.
2. The storage of the data loaded to the peripheral data register or a location in memory addressed through DMA_CMARx register.
3. The post-decrementing of the DMA_CNDTRx register, which contains the number of transactions that have still to be performed.

8.3.2 DMA arbiter

The arbiter manages the channel requests based on their priority and launches the peripheral/memory access sequences. The priorities are managed in two stages:

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

8.3.3 DMA channels

Each channel can handle DMA transfer between a peripheral register located at a fixed address and a memory address. The amount of data to be transferred (up to 65535) is programmable. The register which contains the amount of data items to be transferred is decremented after each transaction.

Programmable data sizes

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

Pointer incrementation

Peripheral and memory pointers can optionally be automatically post-incremented after each transfer depending on the PINC and MINC bits in the DMA_CCRx register. If the incremented mode is enabled, the address of the next transfer will be the address of the previous one incremented by 1, 2 or 4 depending on the chosen data size. The first transfer address is the one programmed in the DMA_CPARx/DMA_CMARx registers. If the channel is configured in noncircular mode, no DMA request is served after the last transfer (that is once the number of data items to be transferred has reached zero).

Channel configuration procedure

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

1. Set the peripheral register address in the DMA_CPARx register. The data will be moved from/ to this address to/ from the memory after the transfer request of peripheral data.
2. Set the memory address in the DMA_CMARx register. The data will be written to or read from this memory after the transfer request of peripheral data.
3. Configure the total number of data to be transferred in the DMA_CNDTRx register. After each data transmission, this value will be decremented.
4. Configure the channel priority using the PL [1:0] bits in the DMA_CCRx register.
5. Configure data transfer direction, circular mode, peripheral & memory incremented

mode, peripheral & memory data size, and interrupt after half and/or full transfer in the DMA_CCRx register.

6. Activate the channel by setting the ENABLE bit in the DMA_CCRx register. As soon as the channel is enabled, it can serve any DMA request from the peripheral connected on the channel.

Once half of the bytes are transferred, the half-transfer flag (HTIF) is set and an interrupt is generated if the Half-Transfer Interrupt Enable bit (HTIE) is set. At the end of the transfer, the Transfer Complete Flag (TCIF) is set to ’1’ and an interrupt is generated if the Transfer Complete Interrupt Enable bit (TCIE) is set.

Circular mode

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

Memory-to-memory mode

The DMA channels can also work without being triggered by a request from a peripheral. This mode is called Memory to Memory mode. If the MEM2MEM bit in the DMA_CCRx register is set, then the channel initiates transfers as soon as it is enabled by software by setting the Enable bit (EN) in the DMA_CCRx register. The transfer stops once the DMA_CNDTRx register reaches zero. Memory to Memory mode may not be used at the same time as Circular mode.

8.3.4 Programmable data width, data alignment and endians

When PSIZE and MSIZE are not equal, the DMA performs some data alignments as described in the following table.

Table 31. Programmable Data Width and Endian Behavior (When Bits PINC = MINC = 1)

Source port width	Destination port width	Number of data items to be transferred (NDT)	Source content (address / data)	Transfer operations	Destination content (address / data)
8	8	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: read B0[7: 0] @ 0x0, write B0[7: 0] @ 0x0 2: read B1[7: 0] @ 0x1, write B1[7: 0] @ 0x1 3: read B2[7: 0] @ 0x2, write B2[7: 0] @ 0x2 4: read B3[7: 0] @ 0x3, write B3[7: 0] @ 0x3	0x0/B0 0x1/B1 0x2/B2 0x3/B3

Source port width	Destination port width	Number of data items to be transferred (NDT)	Source content (address / data)	Transfer operations	Destination content (address / data)
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: read B0[7: 0] @ 0x0, write 00B0[15: 0] @ 0x0 2: read B1[7: 0] @ 0x1, write 00B1[15: 0] @ 0x2 3: read B2[7: 0] @ 0x2, write 00B2[15: 0] @ 0x4 4: read B3[7: 0] @ 0x3, write 00B3[15: 0] @ 0x6	0x0/00B0 0x2/00B1 0x4/00B2 0x6/00B3
8	32	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: read B0[7: 0] @ 0x0, write 000000B0[31: 0] @ 0x0 2: read B1[7: 0] @ 0x1, write 000000B1[31: 0] @ 0x4 3: read B2[7: 0] @ 0x2, write 000000B2[31: 0] @ 0x8 4: read B3[7: 0] @ 0x3, write 000000B3[31: 0] @ 0xC	0x0/000000B0 0x4/000000B1 0x8/000000B2 0xC/000000B3
16	8	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: read B1B0[15: 0] @ 0x0, write B0[7: 0] @ 0x0 2: read B3B2[15: 0] @ 0x2, write B2[7: 0] @ 0x1 3: read B5B4[15: 0] @ 0x4, write B4[7: 0] @ 0x2 4: read B7B6[15: 0] @ 0x6, write B6[7: 0] @ 0x3	0x0/B0 0x1/B2 0x2/B4 0x3/B6
16	16	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: read B1B0[15: 0] @ 0x0, write B1B0[15: 0] @ 0x0 2: read B3B2[15: 0] @ 0x2, write B3B2[15: 0] @ 0x2 3: read B5B4[15: 0] @ 0x4, write B5B4[15: 0] @ 0x4 4: read B7B6[15: 0] @ 0x6, write B7B6[15: 0] @ 0x6	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6

Source port width	Destination port width	Number of data items to be transferred (NDT)	Source content (address / data)	Transfer operations	Destination content (address / data)
16	32	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: read B1B0[15: 0] @ 0x0, write 0000B1B0[31: 0] @ 0x0 2: read B3B2[15: 0] @ 0x2, write 0000B3B2[31: 0] @ 0x4 3: read B5B4[15: 0] @ 0x4, write 0000B5B4[31: 0] @ 0x8 4: read B7B6[15: 0] @ 0x6, write 0000B7B6[31: 0] @ 0xC	0x0/0000B1B0 0x4/0000B3B2 0x8/0000B5B4 0xC/0000B7B6
32	8	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: read B3B2B1B0[31: 0] @ 0x0, write B0[7: 0] @ 0x0 2: read B7B6B5B4[31: 0] @ 0x4, write B4[7: 0] @ 0x1 3: read BBBAB9B8[31: 0] @ 0x8, write B8[7: 0] @ 0x2 4: read BFBEBDBC[31: 0] @ 0xC, write BC[7: 0] @ 0x3	0x0/B0 0x1/B4 0x2/B8 0x3/BC
32	16	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: read B3B2B1B0[31: 0] @ 0x0, write B1B0[15: 0] @ 0x0 2: read B7B6B5B4[31: 0] @ 0x4, write B5B4[15: 0] @ 0x2 3: read BBBAB9B8[31: 0] @ 0x8, write B8B8[15: 0] @ 0x4 4: read BFBEBDBC[31: 0] @ 0xC, write BDBC[15: 0] @ 0x6	0x0/B1B0 0x2/B5B4 0x4/B9B8 0x6/BDBC
32	32	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: read B3B2B1B0[31: 0] @ 0x0, write B3B2B1B0[31: 0] @ 0x0 2: read B7B6B5B4[31: 0] @ 0x4, write B7B6B5B4[31: 0] @ 0x4 3: read BBBAB9B8[31: 0] @ 0x8, write BBBAB8B8[31: 0] @ 0x8 4: read BFBEBDBC[31: 0] @ 0xC, write BFBEBDBC[31: 0] @ 0xC	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC

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

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

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

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

- an AHB byte write operation of the data “0xB0” to 0x0 (or to 0x1, 0x2 or 0x3) will be converted to an APB word write operation of the data “0xB0B0B0B0” to 0x0.
- an AHB halfword write operation of the data “0xB1B0” to 0x0 (or to 0x2) will be converted to an APB word write operation of the data “0xB1B0B1B0” to 0x0.

For instance, to write the APB backup registers (16-bit registers aligned to a 32-bit address boundary), the memory source size (MSIZE) must be configured to “16-bit” and the peripheral destination size (PSIZE) to “32-bit”.

8.3.5 Error management

A DMA transfer error can be generated by reading from or writing to a reserved address space. When a DMA transfer error occurs during a DMA read or a write access, the faulty channel is automatically disabled through a hardware clear of its EN bit in the corresponding Channel configuration register (DMA_CCRx). The channel’s transfer error interrupt flag (TEIF) in the DMA_IFR register is set and an interrupt is generated if the transfer error interrupt enable bit (TEIE) in the DMA_CCRx register is set.

8.3.6 Interrupts

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

Table 32. DMA Interrupt Requests

Interrupt event	Event flag	Enable Control bit
Half-transfer	HTIF	HTIE
Transfer complete	TCIF	TCIE
Transfer error	TEIF	TEIE

8.3.7 DMA request mapping

DMA controller

The 5 requests from the peripherals TIMx、ADC、SPI、I2C and UART are simply logically ORed before entering the DMA1, this means that only one request must be enabled at a time. Refer to the following figure.

The peripheral DMA requests can be independently activated/de-activated by programming the DMA control bit in the registers of the corresponding peripheral.

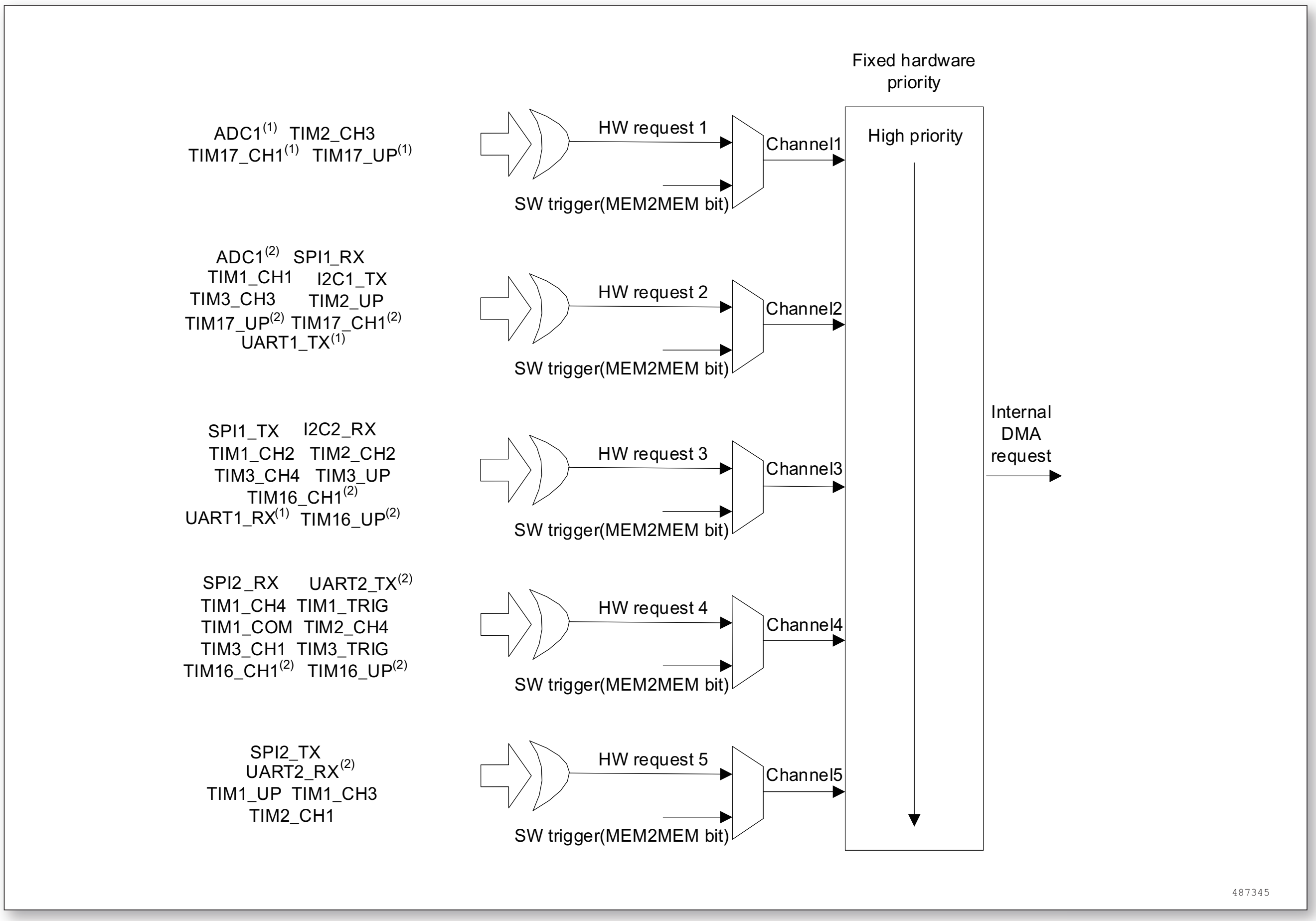

Fixed hardware priority
High priority
ADC1(1) TIM2_CH3 TIM17_CH1(1) TIM17_UP(1)
HW request 1
SW trigger(MEM2MEM bit)
Channel1
ADC1(2) SPI1_RX TIM1_CH1 I2C1_TX TIM3_CH3 TIM2_UP TIM17_UP(2) TIM17_CH1(2) UART1_TX(1)
HW request 2
SW trigger(MEM2MEM bit)
Channel2
SPI1_TX I2C2_RX TIM1_CH2 TIM2_CH2 TIM3_CH4 TIM3_UP TIM16_CH1(2) UART1_RX(1) TIM16_UP(2)
HW request 3
SW trigger(MEM2MEM bit)
Channel3
Internal DMA request
SPI2_RX UART2_TX(2) TIM1_CH4 TIM1_TRIG TIM1_COM TIM2_CH4 TIM3_CH1 TIM3_TRIG TIM16_CH1(2) TIM16_UP(2)
HW request 4
SW trigger(MEM2MEM bit)
Channel4
SPI2_TX UART2_RX(2) TIM1_UP TIM1_CH3 TIM2_CH1
HW request 5
SW trigger(MEM2MEM bit)
Channel5
487345

Figure 22. Peripheral DMA Request Mapping

Table 33. Summary of DMA Requests for Each Channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC	ADC1(1)	ADC1(2)			
SPI		SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX
UART		UART1_TX (1)	UART1_RX(1)	UART2_TX(2)	UART2_RX(2)
I2C		I2C1_TX	I2C1_RX		
TIM1		TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP TIM1_CH3
TIM2	TIM2_CH3	TIM2_UP	TIM2_CH2	TIM2_CH4	TIM2_CH1
TIM3		TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	

TIM16			TIM16_CH1(1)
			TIM16_UP(1)
				TIM16_CH1(2)
				TIM16_UP(2)	

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
TIM17	TIM17_CH1[(1)] TIM17_UP[(1)]	TIM17_CH1[(2)] TIM17_UP[(2)]			

1. If the mapping bit in SYSCFG_CFGR register is cleared, the DMA request is mapped on the DMA channel.
2. If the mapping bit in SYSCFG_CFGR register is set, the DMA request is mapped on the DMA channel.

8.4 DMA register description

Table 34. Summary of DMA Registers

Offset	Acronym	Register Name	Reset	Section
0x00	DMA_ISR	DMA interrupt status register	0x00000000	section 8.4.1
0x04	DMA_IFCR	DMA interrupt flag clear register	0x00000000	section 8.4.2
0x08 + 20 × (n - 1)	DMA_CCRx	DMA channel x configuration register	0x00000000	section 8.4.3
0x0C + 20 × (n - 1)	DMA_CNDTRx	DMA channel x number of data register	0x00000000	section 8.4.4
0x10 + 20 × (n - 1)	DMA_CPARx	DMA channel x peripheral address register	0x00000000	section 8.4.5
0x14 + 20 × (n - 1)	DMA_CMARx	DMA channel x memory address register	0x00000000	section 8.4.6

8.4.1 DMA interrupt status register(DMA_ISR)

Offset address: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												TEIF5	HTIF5	TCIF5	GIF5
												r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31 : 20	Reserved			Reserved, always read as 0.
19,15,11, 7, 3	TEIFx	r	0x00	Channel x transfer error flag(x = 1 ... 5) This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register. 0: No transfer error (TE) on channel x 1: A transfer error (TE) occurred on channel x

Bit	Field	Type	Reset	Description
18,14,10, 6, 2	HTIFx	r	0x00	Channel x half transfer flag(x = 1 ... 5) This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register. 0: No half transfer (HT) event on channel x 1: A half transfer (HT) event occurred on channel x
17,13,9, 5, 1	TCIFx	r	0x00	Channel x transfer complete flag(x = 1 ... 5) This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register. 0: No transfer complete (TC) event on channel x 1: A transfer complete (TC) event occurred on channel x
16,12,8, 4, 0	GIFx	r	0x00	Channel x global interrupt flag(x = 1 ... 5) This bit is set by hardware. It is cleared by software writing 1 to the corresponding bit in the DMA_IFCR register. 0: No TE, HT or TC event on channel x 1: A TE, HT or TC event occurred on channel x

8.4.2 DMA interrupt flag clear register(DMA_IFCR)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CTE IF5	CHT IF5	CTC IF5	CG IF5
												w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTE IF4	CHT IF4	CTC IF4	CG IF4	CTE IF3	CHT IF3	CTC IF3	CG IF3	CTE IF2	CHT IF2	CTC IF2	CG IF2	CTE IF1	CHT IF1	CTC IF1	CG IF1
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31 : 20	Reserved			Reserved, always read as 0.
19,15,11, 7, 3	CTEIFx	w	0x00	Channel x transfer error clear(x = 1 ... 5) This bit is set and cleared by software. 0: No effect 1: Clear the corresponding TEIF flag in the DMA_ISR register
18,14,10, 6, 2	CHTIFx	w	0x00	Channel x half transfer clear(x = 1 ... 5) This bit is set and cleared by software. 0: No effect 1: Clear the corresponding HTIF flag in the DMA_ISR register

Bit	Field	Type	Reset	Description
17,13,9, 5, 1	CTCIFx	w	0x00	Channel x transfer complete clear(x = 1 ... 5) This bit is set and cleared by software. 0: No effect 1: Clear the corresponding TCIF flag in the DMA_ISR register
16,12,8, 4, 0	CGIFx	w	0x00	Channel x global interrupt clear(x = 1 ... 5) This bit is set and cleared by software. 0: No effect 1: Clear the GIF, TEIF, HTIF and TCIF flags in the DMA_ISR register

8.4.3 DMA channel x configuration register(DMA_CCRx) (x = 1···5)

Offset address: 0x08 + 20 x (channel number - 1)

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
ARE	MEM2 MEM	PL		MSIZE		PSIZE		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Reserved, always read as 0.
15	ARE	rw	0x00	Auto-Reload Enable This bit is set and cleared by software. After aborting the transfer, whether the NDT, PADDR, MADDR registers of each channel return to the initial value set: 1: Initial value of auto-reload setting after aborting transfer 0: Disable automatic reload function
14	MEM2MEM	rw	0x00	Memory to memory mode This bit is set and cleared by software. 0: Memory to memory mode disabled 1: Memory to memory mode enabled
13 : 12	PL	rw	0x00	Channel priority level This bit is set and cleared by software. 00: Low 01: Medium 10: High 11: Very high

Bit	Field	Type	Reset	Description
11 : 10	MSIZE	rw	0x00	Memory size This bit is set and cleared by software. 00: 8-bits 01: 16-bits 10: 32-bits 11: Reserved
9 : 8	PSIZE	rw	0x00	Peripheral size This bit is set and cleared by software. 00: 8-bits 01: 16-bits 10: 32-bits 11: Reserved
7	MINC	rw	0x00	Memory increment mode This bit is set and cleared by software. 0: Memory increment mode disabled 1: Memory increment mode enabled
6	PINC	rw	0x00	Peripheral increment mode This bit is set and cleared by software. 0: Peripheral increment mode disabled 1: Peripheral increment mode enabled
5	CIRC	rw	0x00	Circular mode This bit is set and cleared by software. 0: Circular mode disabled 1: Circular mode enabled
4	DIR	rw	0x00	Data transfer direction This bit is set and cleared by software. 0: Read from peripheral 1: Read from memory
3	TEIE	rw	0x00	Transfer error interrupt enable This bit is set and cleared by software. 0: TE interrupt disabled 1: TE interrupt enabled
2	HTIE	rw	0x00	Half transfer interrupt enable This bit is set and cleared by software. 0: HT interrupt disabled 1: HT interrupt enabled
1	TCIE	rw	0x00	Transfer complete interrupt enable This bit is set and cleared by software. 0: TC interrupt disabled 1: TC interrupt enabled

Bit	Field	Type	Reset	Description
0	EN	rw	0x00	Channel enable This bit is set and cleared by software. 0: Channel disabled 1: Channel enabled

8.4.4 DMA channel x number of data register(DMA_CNDTRx) (x = 1···5)

Offset address: 0x0C + 20 x (channel number - 1)

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
NDT															
rw															

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Reserved, always read as 0.
15 : 0	NDT	rw	0x0000	Number of data to transfer Number of data to be transferred (0 to 65535). This register can only be written when the channel is disabled (EN bit of DMA_CCRx is 0). Once the channel is enabled, this register is read-only, indicating the remaining bytes to be transmitted. This register decrements after each DMA transfer. Once the transfer is completed, this register can either stay at zero or be reloaded automatically by the value previously programmed if the channel is configured in auto-reload mode. If this register is zero, no transaction can be served no matter whether the channel is enabled or not.

8.4.5 DMA channel x peripheral address register(DMA_CPARx) (x = 1···5)

Offset address: 0x10 + 20 x (channel number - 1)

Reset value: 0x0000 0000

This register must not be written when the channel (EN bit of DMA_CCRx is 0) is enabled.

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

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

Bit	Field	Type	Reset	Description
31 : 0	PA	rw	0x0000 0000	Peripheral address Base address of the peripheral data register is used as a source or target of data transfer. When PSIZE is 01 (16-bit), the PA[0] bit is ignored. Access is automatically aligned to a half word address. When PSIZE is 10 (32-bit), PA[1:0] are ignored. Access is automatically aligned to a word address.

8.4.6 DMA channel x memory address register(DMA_CMARx) (x = 1···5)

Offset address: 0x14 + 20 x (channel number - 1)

Reset value: 0x0000 0000

This register must not be written when the channel (EN bit of DMA_CCRx is 0) is enabled.

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

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

Bit	Field	Type	Reset	Description
31 : 0	MA	rw	0x0000 0000	Memory address The memory address serves as the source or destination of data transmission. When MSIZE is 01 (16-bit), the MA[0] bit is ignored. Access is automatically aligned to a half-word address. When MSIZE is 10 (32-bit), MA [1:0] are ignored. Access is automatically aligned to a word address.

9 Analog-to-digital converter(ADC)

Analog-to-digital converter(ADC)

9.1 ADC introduction

The 12-bit ADC is a successive approximation analog-to-digital converter.

A/D conversion of the various channels can be performed in single, continuous, scan mode, and you can choose automatic channel scanning.

Its start-up mode includes the software setting, external pin triggering and activated by 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 PCLK2 clock divided by a prescaler and it must not exceed 15 MHz.

9.2 ADC main features

- 12-bit-resolution SAR ADC, up to 13 external input channels and 2 internal input channels
- Up to 1 Msps conversion rate
- Supporting multiple operation modes:
  - Single conversion mode: one A/D conversion in specified channel
  - Single-cycle scanning mode: one A/D conversion cycle (from low-number channel to high-number channel) completed in all designated channels
  - Continuous scan mode: A/D converter continuously performs single-cycle scanning until the converter is disabled by software
- Channel sampling time and resolution can be configured by software
- Support DMA transfer
- Conditions of A/D conversion:
  - By software
  - External triggering
  - Timer matching
- In terms of analog watchdog, the conversion result can be compared with the specified value; the user can set whether to generate an interrupt request or not when the conversion value matches the set value.

9.3 ADC functional description

The ADC block diagram is as shown below.

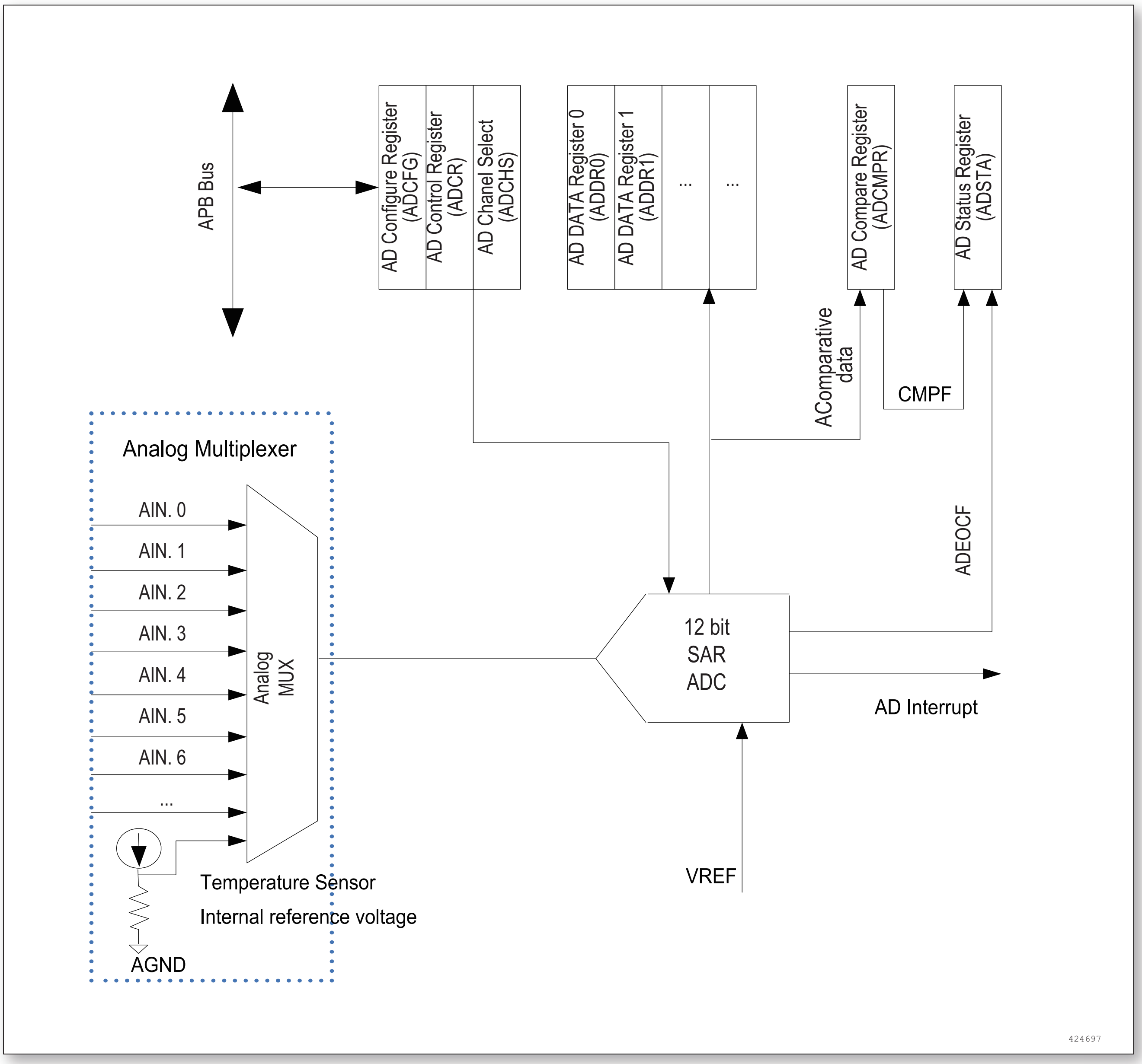


Figure 23. ADC block diagram

9.3.1 ADC on-off control

The ADC can be powered-on by setting the ADEN 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 of ADCR register is set after ADC power-up time.

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

9.3.2 Channel selection

There are several external input channels, internal temperature sensor channel and internal 1.2 V reference voltage channel. Among them, each external input channel has independent enabling bit, which can be configured by setting bits concerned of the AD-

CHS register.

9.4 ADC operating mode

9.4.1 Single conversion mode

In the single conversion mode, the A/D conversion is only performed once on the corresponding channel, and the specific process is as follows:

- The A/D conversion is activated by software, external trigger input, and the ADST bit of the timer overflow setting ADCR register.
- After the A/D conversion, the data value concerned will be saved in the ADDATA and ADDRn data registers of A/D converter.
- ADIF bit of status register ADSTA is set to '1' after the A/D conversion. If ADIE bit of control register ADCR is set to '1' at this time, an AD conversion end interrupt request will be generated.
- During the A/D conversion, the ADST bit remains 1. After that, the ADST bit is cleared automatically and the idle mode is enabled.

Note: If the software, in the single conversion mode, enables more than one channel, the channel with the smallest number will be converted and other channels will be ignored.

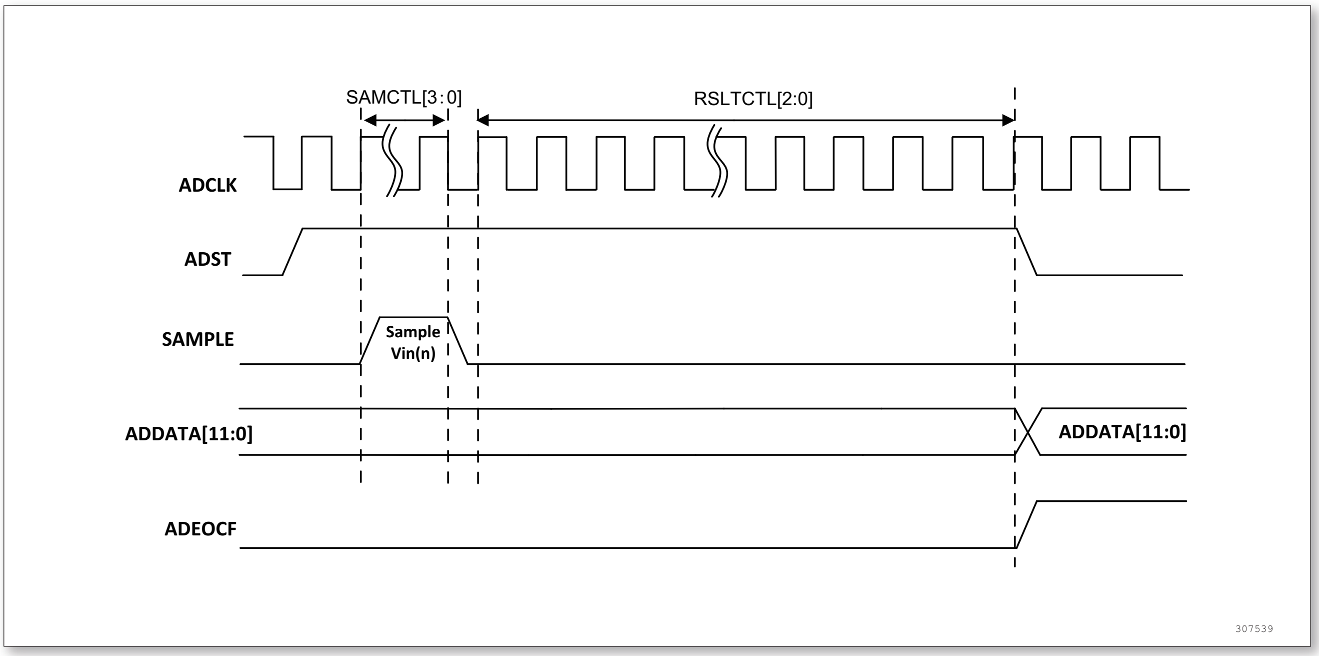


Figure 24. Timing Diagram of Single Conversion Mode

9.4.2 Single-cycle scan mode

In the single-cycle scan mode, the A/D conversion is performed in the order of the channels that can be enabled (the scan channel direction can be selected by the configuration register bit SCAN_DIR). The operation steps are as follows:

- A/D conversion starts in software or external trigger setting ADST. The direction setting defaults from the minimum serial number channel to the maximum serial number channel. It can also be set according to the program, from the largest serial number channel to the smallest serial number channel.

- After A/D conversion in each channel, the A/D conversion values will be loaded into the data registers in the corresponding channels in an orderly manner, and the ADIF conversion end flag will be set. If the conversion end interrupt flag is set, an interrupt request will be generated after the conversion in all channels.
- After the conversion, ADST bit is cleared automatically, so that A/D converter enters idle state.

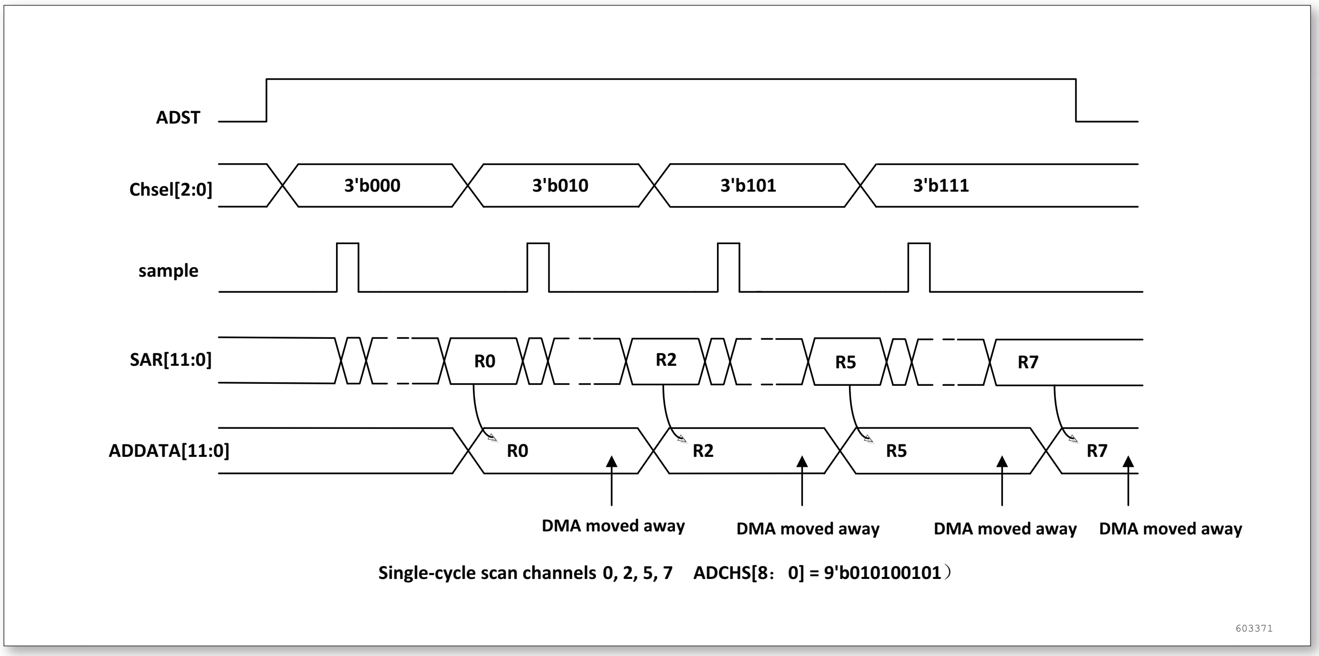


Figure 25. Timing Diagram of Enabled Channel During Conversion in Single-cycle Scan Mode(channel direction from low to high)

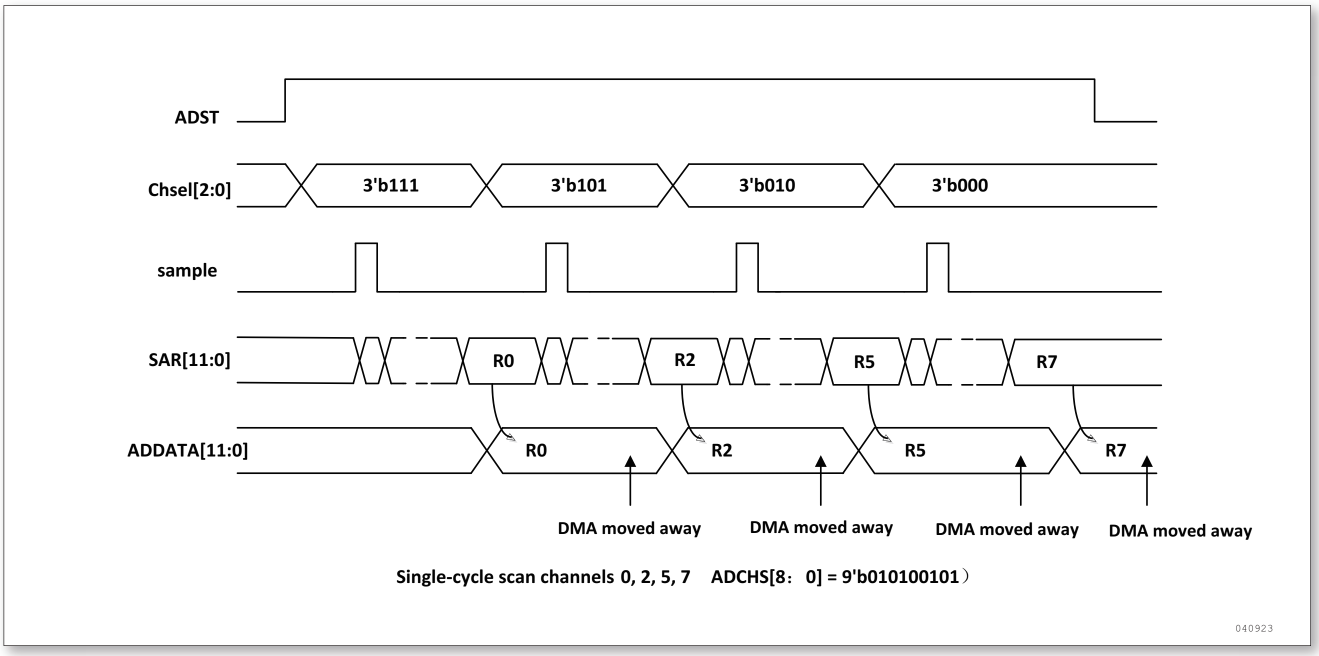


Figure 26. Timing Diagram of Enabled Channel During Conversion in Single-cycle Scan Mode(channel direction from high to low)

9.4.3 Continuous scan mode

In the continuous scan mode, the A/D conversion is performed on the enabled CHENn bit in the ADCHS register(the scan channel direction can be selected by the configuration register bit SCAN_DIR). The operation steps are as follows:

- A/D conversion starts in software or external trigger setting ADST. External trigger can be configured by software. The direction setting defaults from the minimum serial number channel to the maximum serial number channel. It can also be set according to the program, from the largest serial number channel to the smallest serial number channel.
- After A/D conversion in all channels, the A/D conversion values will be loaded into the data registers concerned in an orderly manner, and the ADIF conversion end flag will be set. If the conversion end interrupt flag is set, an interrupt request will be generated after the conversion in all channels.
- As long as the ADST bit remains 1, the A/D conversion continues. When ADST bit is cleared and A/D conversion is completed, A/D converter enters the idle mode. When ADST is cleared, the current A/D conversion will be completed.

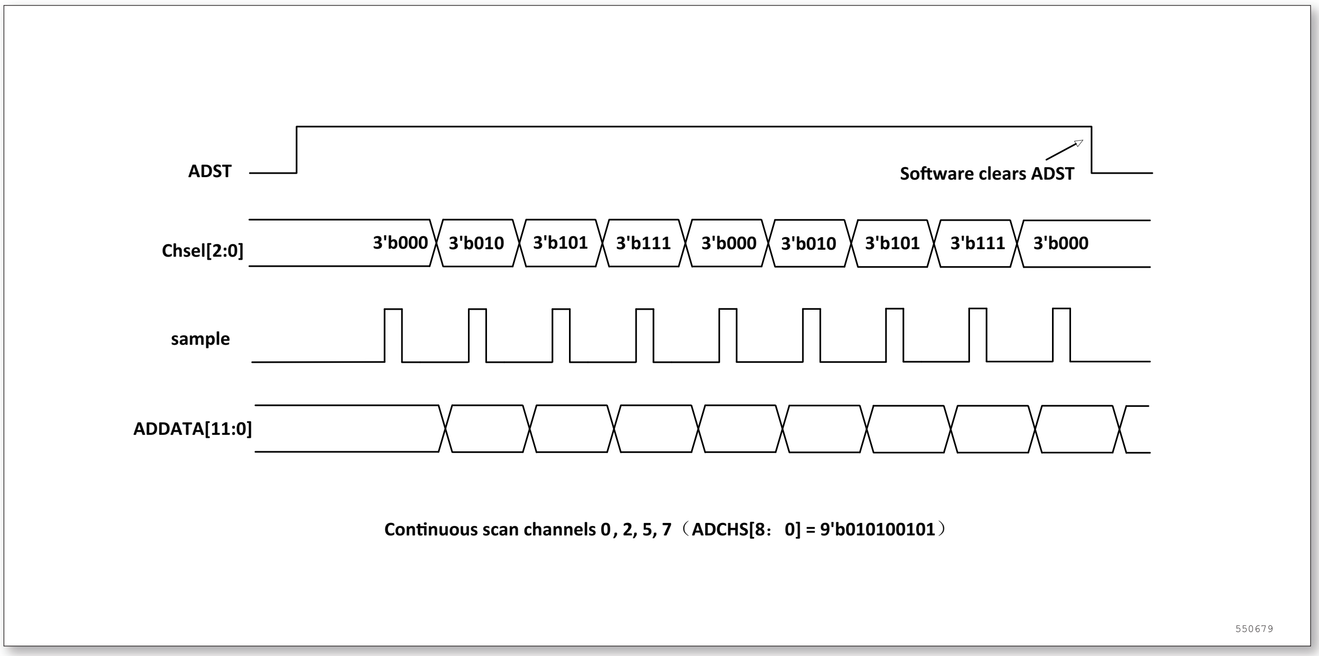


Figure 27. Timing Diagram of Enabled Channel During Conversion in Continuous Scan Mode(channel direction from low to high)

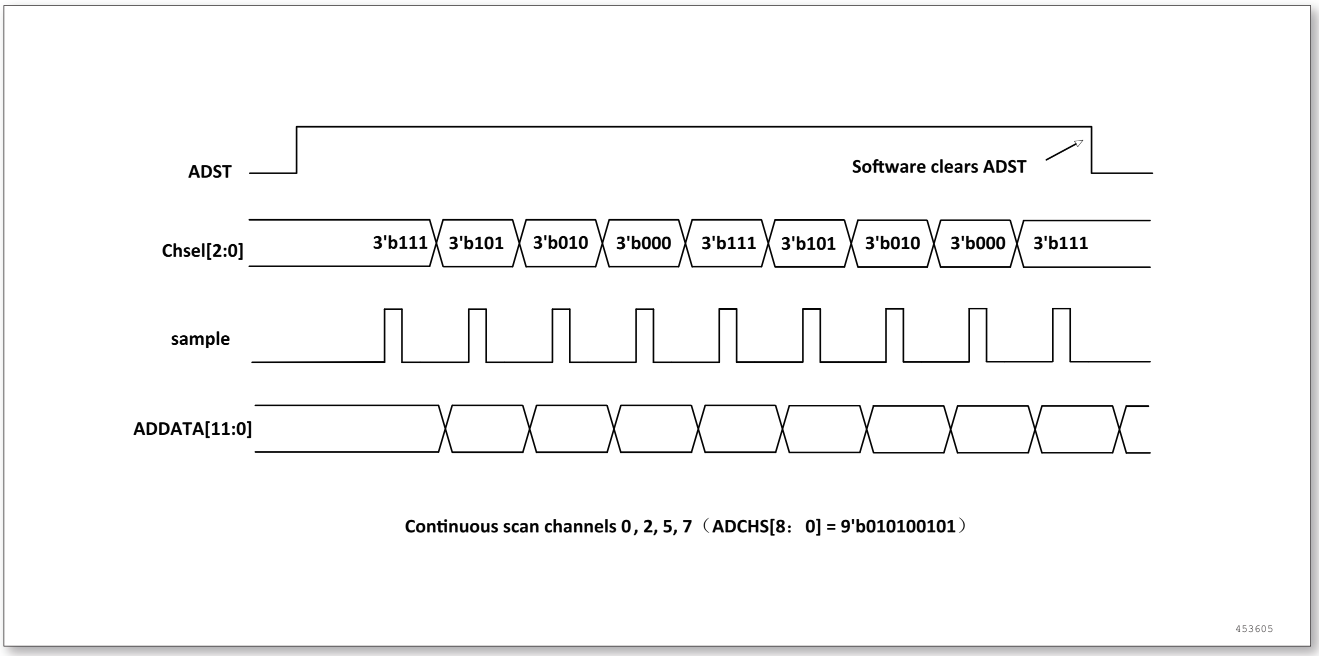


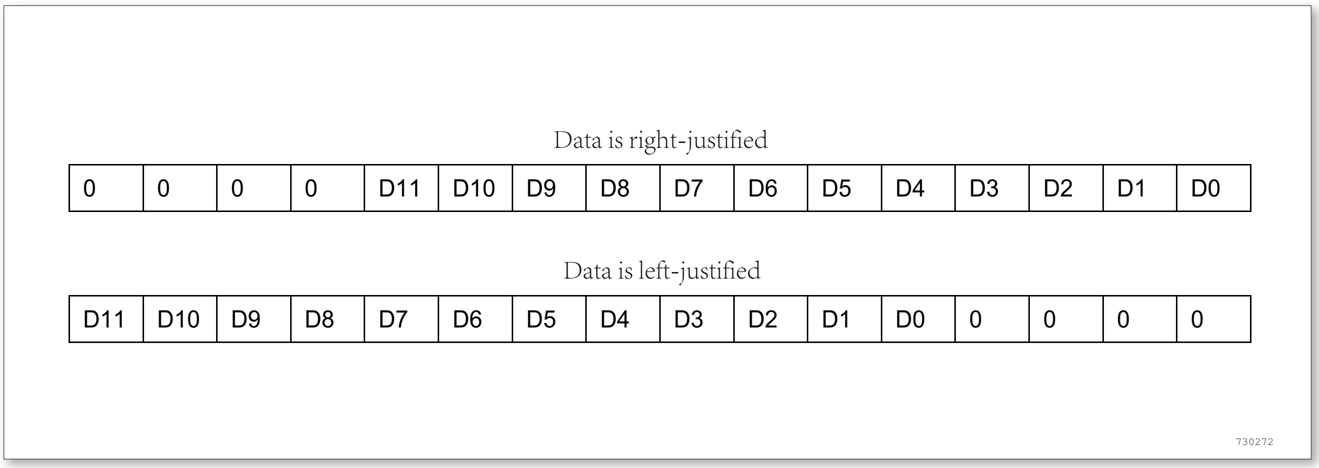
Figure 28. Timing Diagram of Enabled Channel During Conversion in Continuous Scan Mode(channel direction from high to low)

9.4.4 DMA request

In the single-cycle scan and continuous scan modes, the value of channel conversion is saved in the data registers (ADDRn) in respective channel, and the result of the latest conversion is also stored in the ADDATA register. During DMA transmission, you can choose to transfer data in a specific channel or transfer the results of all scanning channels.

9.5 Data alignment

ALIGN bit in the ADCR register selects the alignment of data stored after conversion. Data can be left or right aligned as shown in the following figure .


Data is right-justified

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Data is left-justified

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

Figure 29. Data Alignment Modes

9.5.1 Programmable resolution

The effective ADC conversion bits can be changed by modifying RSLT CTL [2: 0] bits in ADC_CFG register, to improve the data conversion rate. The effective data bits are aligned at the high bits of 12-bit data.

9.5.2 Programmable sample time

ADCLK, the ADC clock, is generated by dividing PCLK2, and its division factor can be determined by setting the AD-CPRE bit in ADCFG bit, namely, the PCLK2 / (N + 1) / 2 is used as the ADC clock.

The ADC resolution is n (n=8, 9, 10, 11 and 12), the sampling period in each channel is m, and the number of sampling periods can be modified through the SAMCTL bits in the ADC_CFG registers.

The sampling frequency and sampling time are calculated as follows:

$F_{sample} = F_{ADCLK} / (m + n + 1.5)$.

Example:

With an ADCCLK = 15MHz, the resolution of 12 bits and a sampling time of 1.5 cycles

$F_{sample} = F_{ADCLK} / 15$.

$T_{CONV} = 1.5 + 13.5 = 15$ cycles = 1μs

9.6 Conversion on external trigger

Conversion can be triggered by an external event (e.g. timer capture, EXTI line). If the TRGEN bit of ADCR register is set, then external events are able to trigger a conversion. By setting the TRGSEL bits, the external trigger sources can be selected.

For the selection of specific external trigger sources, please refer to the description of relevant bits in AD control register.

The external trigger can set the delay control, refer to the description of TRGSHIFT of ADCR[21:19].

The sampling is initiated after the generation of trigger signal and N PCLK2 cycles. In the trigger scan mode, only the sampling of the first channel is delayed, and the rest channels is sampled immediately after the end of the previous operation.

9.7 Temperature sensor

The temperature sensor can be used to measure the ambient temperature (T_A) of the device.

The temperature sensor is internally connected to the ADC input channel which is used to convert the sensor output voltage into a digital value. When not in use, this sensor can be disabled separately by setting relevant bits of the register.

The temperature sensor output voltage changes linearly with temperature. The offset of this line varies from chip to chip due to process variation.

The internal temperature sensor is more suited to applications that detect temperature variations instead of absolute temperatures. If accurate temperature readings are needed, an external temperature sensor part should be used.

The temperature is calculated as follows:

$T(^\circ C) = (V_{SENSE} - V_{25}) / Avg\_Slope + 25$

V_{25}: V_{SENSE} value for 25°C

V_{SENSE}: the current output voltage of temperature sensor

$V_{SENSE} = Value * V_{dd} / 4096$ (Value is the conversion result of ADC)

Avg_Slope: Average Slope for curve between Temperature vs. V_{SENSE}(given in mV/°C or μV/°C)

Refer to the temperature sensor section for the actual values of V_{25} and Avg_Slope.

9.8 Internal reference voltage

The input channel of ADC is loaded with an internal reference voltage (1.2V), converting the reference voltage output of 1.2V into a digital value.

The internal reference voltage has a separate enable bit, which can be enabled or disabled by setting the corresponding bit in the register.

9.9 Monitoring of AD conversion results in window comparator mode

The upper limit and lower limit compare registers are enabled in the comparison mode, and the CMPCH bit can be set by software, to select the monitoring channel.

If CPMHDATA is ≥ CPMLDATA, and the comparison result is greater than or equal to the specified value of CMPHDATA in the ADCMPR register or less than the specified value of CMPLDATA, the ADWIF bit of the status register ADSTA is set to 1.

If CPMHDATA is < CPMLDATA and the comparison result is equal to the specified value of CMPHDATA or between the two specified value, the ADWIF bit of the status register ADSTA is set to 1. An interrupt request will be generated if ADWIE bit of the control register ADCR is set. An interrupt request will be generated if ADWIE bit of the control register ADCR is set.

9.10 ADC register description

Table 35. Summary of ADC Registers

Offset	Acronym	Register Name	Reset	Section
0x00	ADC_ADDATA	A/D data register	0x00000000	section 9.10.1
0x04	ADC_ADCFG	A/D configuration register	0x00000000	section 9.10.2
0x08	ADC_ADCR	A/D control register	0x00000000	section 9.10.3
0x0C	ADC_ADCHS	A/D channel select register	0x00000000	section 9.10.4

Offset	Acronym	Register Name	Reset	Section
0x10	ADC_ADCMPR	A/D window compare register	0x00000000	section 9.10.5
0x14	ADC_ADSTA	A/D status register	0x00000000	section 9.10.6
0x18~ 0x48	ADC_ADDR0 ~ 12	A/D data register	0x00000000	section 9.10.7
0x50~ 0x54	ADC_ADDR14 ~ 15	A/D data register	0x00000000	section 9.10.7
0x58	ADC_ADSTA_EXT	A/D extended status register	0x00000000	section 9.10.8

9.10.1 A/D data register(ADC_ADDATA)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										VAILD	OVER RUN	CHANNELSEL			
										r	r	r	r	r	r

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

Bit	Field	Type	Reset	Description
31 : 22	Reserved			Reserved, always read as 0.
21	VALID	r	0x00	Valid flag (read-only) 1 = DATA[11: 0] bits are valid 0 = DATA[11: 0] bits are invalid After the conversion in the corresponding analog channel, this bit is set and this bit is cleared by hardware after the ADDATA register is read.
20	OVERRUN	r	0x00	Overrun flag (read-only) 1 = DATA[11: 0] data overwritten 0 = The last conversion result of DATA[11: 0] data Before the new conversion result is loaded into the register, if the data of DATA[11: 0] is not read, OVERRUN will be set to 1. This bit is cleared by hardware after the ADDATA register is read.

Bit	Field	Type	Reset	Description
19 : 16	CHANNELSEL	r	0x00	Channel selection (the 4 bits show the channel corresponding to the current data) 0000 = convert data for Channel 0 0001 = convert data for Channel 1 0010 = convert data for Channel 2 0011 = convert data for Channel 3 0100 = convert data for Channel 4 0101 = convert data for Channel 5 0110 = convert data for Channel 6 0111 = convert data for Channel 7 1000 = convert data for Channel 8 1001 = convert data for Channel 9 1010 = convert data for Channel 10 1011 = convert data for Channel 11 1100 = convert data for Channel 12 1110 = convert data of temperature sensor 1111 = convert data of internal reference voltage Others: invalid
15 : 0	DATA	r	0x00	Transfer data (12-bit A/D conversion result) Left alignment or right alignment, depending on specific settings.

9.10.2 A/D configuration register(ADC_ADCFG)

Address offset: 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
Res.	ADCPRE	SAMCTL				RSLTCTL			ADCPRE			VSEN	TSEN	ADWEN	ADEN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 15	Reserved			Reserved, always read as 0.
14	ADCPRE	rw	0x00	ADC prescaler As the lowest bit of ADCPRE[3:0], combined with Bit[6:4]

Bit	Field	Type	Reset	Description
13 : 10	SAMCTL	rw	0x00	Channel x Sample time selection These bits are used to independently select the sampling time for each channel. The channel select bit must remain unchanged during the sampling period. 0000: 1.5 cycles 0100: 41.5 cycles 0001: 7.5 cycles 0101: 55.5 cycles 0010: 13.5 cycles 0110: 71.5 cycles 0011: 28.5 cycles 0111: 239.5 cycles 1000: 2.5 cycles 1001: 3.5 cycles 1010: 4.5 cycles 1011: 5.5 cycles 1100: 6.5 cycles Others: Reserved
9 : 7	RSLTCTL	rw	0x00	Resolution (select ADCx conversion data resolution) 000: valid in 12 bits 001: valid in 11 bits 010: valid in 10 bits 011: valid in 9 bits 100: valid in 8 bits
6 : 4	ADCPRE	rw	0x00	ADC prescaler Set to '1' or cleared by software to determine the ADC clock frequency. When Bit[14] is 0, the actual division factor is (2 * (ADCPRE + 1)) When Bit[14] is 1, the actual division factor is (2 * (ADCPRE + 1) + 1)
3	VSEN	rw	0x00	Voltage Sensor enable 1: Internal voltage sensor enabled 0: Internal voltage sensor disabled
2	TSEN	rw	0x00	Temperature sensor enable 1 = Temperature sensor enabled 0 = Temperature sensor disabled
1	ADWEN	rw	0x00	ADC window comparison enable 1 = A/D window comparator enabled 0 = A/D window comparator disabled
0	ADEN	rw	0x00	ADC enable 1 = Enabled 0 = Disabled

9.10.3 A/D control register(ADC_ADCR)

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										TRGSHIFT			TRGSEL		SCANDIR
										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPCH				ALIGN	ADMD		ADST	Res.	TRGSEL			DMAEN	TRGEN	ADWIE	ADIE
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 22	Reserved			Reserved, always read as 0.
21 : 19	TRGSHIFT	rw	0x00	External trigger shift sample After the trigger signal is generated, the clock period of N PCLK2 is delayed to start sampling again. If the scan mode is triggered, the other channels start immediately after the last sample is finished. 0: No delay 1: 4 cycles 2: 16 cycles 3: 32 cycles 4: 64 cycles 5: 128 cycles 6: 256 cycles 7: 512 cycles
18 :17	TRGSEL	rw	0x00	External trigger selection Used in conjunction with Bit[6:4].
16	SCANDIR	rw	0x00	ADC scan direction Set the order of the scan channels in single-cycle or continuous scan mode: 0: ADC channel select registers are scanned from low to high 1: ADC channel select registers are scanned from high to low

Bit	Field	Type	Reset	Description
15 : 12	CMPCH	rw	0x00	Window comparison channel selection 0000 = Conversion result of select comparison channel 0 0001 = Conversion result of select comparison channel 1 0010 = Conversion result of select comparison channel 2 0011 = Conversion result of select comparison channel 3 0100 = Conversion result of select comparison channel 4 0101 = Conversion result of select comparison channel 5 0110 = Conversion result of select comparison channel 6 0111 = Conversion result of select comparison channel 7 1000 = Conversion result of select comparison channel 8 1001 = Conversion result of select comparison channel 9 1010 = Conversion result of select comparison channel 10 1011 = Conversion result of select comparison channel 11 1100 = Conversion result of select comparison channel 12 1110 = Conversion result of select comparison temperature sensor 1111 = Conversion result of reference voltage on select comparison channel Other: invalid
11	ALIGN	rw	0x00	Data alignment 0: Right alignment 1: Left alignment
10 : 9	ADMD	rw	0x00	ADC mode 00: Single conversion 01: Single-cycle scan 10: Continuous scan When changing the conversion mode, disable the ADST bit by the software.
8	ADST	rw	0x00	ADC start 1 = Conversion starts 0 = Conversion ends or it enables idle mode ADST bit can be set in the following two ways: In single mode or single-cycle mode, ADST bit will be automatically cleared by hardware after the conversion. In the continuous scan mode, the A/D conversion continues until the software writes' 0' to this bit or the system resets.
7	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
6 : 4	TRGSEL	rw	0x00	External trigger selection, Bit[18:17,6:4] select external trigger source 00000: TIM1_CC1 00001: TIM1_CC2 00010: TIM1_CC3 00011: TIM2_CC2 00100: TIM3_TRGO 00101: TIM1_CC4 and CC5 00110: TIM3_CC1 00111: EXTI line 11 01000: TIM1_TRGO 01011: TIM2_CC1 01100: TIM3_CC4 01101: TIM2_TRGO 01111: EXTI line 15 10000: TIM1_CC4 10001: TIM1_CC5 Others: invalid
3	DMAEN	rw	0x00	Direct memory access enable 1 = DMA request enabled 0 = DMA disabled
2	TRGEN	rw	0x00	External trigger enable 1 = Start A/D conversion with external trigger signal 0 = Start A/D conversion without external trigger signal
1	ADWIE	rw	0x00	ADC window comparator interrupt enable 1 = A/D window comparator interrupt enabled 0 = A/D window comparator interrupt disabled
0	ADIE	rw	0x00	ADC interrupt enable 1 = A/D interrupt enabled 0 = A/D interrupt disabled If ADIF is set, an interrupt request is generated after the A/D conversion.

9.10.4 A/D channel select register(ADC_ADCHS)

Address offset: 0x0C

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
CHENVS	CHENTS	Res.	CHEN12	CHEN11	CHEN10	CHEN9	CHEN8	CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0
rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Reserved, always read as 0.
15	CHENVS	rw	0x00	Voltage Sensor enable 1 = Enabled 0 = Disabled
14	CHENTS	rw	0x00	Temperature Sensor enable 1 = Enabled 0 = Disabled
13	Reserved			Reserved, always read as 0.
12	CHEN12	rw	0x00	Analog input channel 12 enable 1 = Enabled 0 = Disabled
11	CHEN11	rw	0x00	Analog input channel 11 enable 1 = Enabled 0 = Disabled
10	CHEN10	rw	0x00	Analog input channel 10 enable 1 = Enabled 0 = Disabled
9	CHEN9	rw	0x00	Analog input channel 9 enable 1 = Enabled 0 = Disabled
8	CHEN8	rw	0x00	Analog input channel 8 enable 1 = Enabled 0 = Disabled
7	CHEN7	rw	0x00	Analog input channel 7 enable 1 = Enabled 0 = Disabled
6	CHEN6	rw	0x00	Analog input channel 6 enable 1 = Enabled 0 = Disabled
5	CHEN5	rw	0x00	Analog input channel 5 enable 1 = Enabled 0 = Disabled
4	CHEN4	rw	0x00	Analog input channel 4 enable 1 = Enabled 0 = Disabled

Bit	Field	Type	Reset	Description
3	CHEN3	rw	0x00	Analog input channel 3 enable 1 = Enabled 0 = Disabled
2	CHEN2	rw	0x00	Analog input channel 2 enable 1 = Enabled 0 = Disabled
1	CHEN1	rw	0x00	Analog input channel 1 enable 1 = Enabled 0 = Disabled
0	CHEN0	rw	0x00	Analog input channel 0 enable 1 = Enabled 0 = Disabled

Note: If channels enabled are all 0, Channel 0 is enabled.

9.10.5 A/D window compare register(ADC_ADCMPR)

Address offset: 0x10

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 28	Reserved			Reserved, always read as 0.
27 : 16	CMPHDATA	rw	0x00	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	rw	0x00	Compare data low limit The 12-bit value will be compared with the conversion result of the specified channel.

9.10.6 A/D status register(ADC_ADSTA)

Address offset: 0x14

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALID								CHANNEL				Res.	BUSY	ADWIF	ADIF
r	r	r	r	r	r	r	r	r	r	r	r		r	rc_w1	rc_w1

Bit	Field	Type	Reset	Description
31 : 20	OVERRUN	r	0x0000	Overrun flag (Channel 0 ~ 11) Read only.
19 : 8	VALID	r	0x0000	Valid flag (Channel 0 ~ 11) Read only.
7 : 4	CHANNEL	r	0x00	Current conversion channel In case of BUSY = 1, the 4 bits indicate the channel being converted. In case of BUSY = 0, they indicate the channel to be converted in the next time.
3	Reserved			Reserved, always read as 0.
2	BUSY	r	0x00	Busy/idle 1 = A/D converter is busy 0 = A/D converter is idle
1	ADWIF	rc_w1	0x00	ADC window comparator interrupt flag If the result of selected A/D conversion channel is greater than or equal to ADCMPHR or less than ADCMPLR, this bit is set to '1'. This flag bit is cleared by writing '1'.
0	ADIF	rc_w1	0x00	ADC interrupt flag This bit is set by hardware at the end of channel group conversion and cleared by software. 1 = A/D conversion completed 0 = A/D conversion not completed This flag bit is cleared by writing' 1'.

9.10.7 A/D data register(ADC_ADDR0 ~ 12, 14 ~ 15)

Address offset: 0x18 – 0x48, 0x50 – 0x54

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										VAILD	OVER RUN	Reserved			
										r	r				

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

Bit	Field	Type	Reset	Description
31 : 22	Reserved			Reserved, always read as 0.
21	VALID	r	0x00	Valid flag(read-only) 1 = DATA[11: 0] bits are valid 0 = DATA[11: 0]bits are invalid After the conversion in the corresponding analog channel, this bit is set and this bit is cleared by hardware after the ADDATA register is read.
20	OVERRUN	r	0x00	Overrun flag(read-only) 1 = DATA [11: 0]bits are overwritten 0 = The last conversion result of DATA[11: 0] data Before the new conversion result is loaded into the register, if the data of DATA[11: 0] is not read, OVERRUN will be set to 1. This bit is cleared by hardware after the ADDATA register is read.
19 : 16	Reserved			Reserved, always read as 0.
15 : 0	DATA	r	0x00	Transfer data(12-bit A/D conversion result on channel) Left alignment or right alignment, depending on specific settings.

9.10.8 A/D extended status register(ADC_ADSTA_EXT)

Address offset: 0x58

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								OVERRUN				VALID			
								r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
7 : 4	OVERRUN	r	0x00	Overrun flag 1000: channel 15(V_SENSOR) 0100: channel 14(T_SENSOR) 0001: channel 12
3 : 0	VALID	r	0x00	Valid flag 1000: channel 15(V_SENSOR) 0100: channel 14(T_SENSOR) 0001: channel 12

10 Comparator(COMP)

Comparator(COMP)

10.1 COMP introduction

Universal embedded chip comparator may be used independently (for all terminals on the I / O port), also be combined with the use of a timer which can be used for a variety of functions, including:. Par

- Trigger the wakeup event in the low power consumption mode through the analog signal
- Adjust analog signal
- Combined with the PWM of the timer output to form a cycle-by-cycle current control loop

10.2 Main features of comparator

- Rail-to-rail comparator
- Each comparator has optional thresholds
  - Reusable I/O pins
  - Alternatively CRV internal comparison voltage AVDD or the internal reference voltage value of divided voltage
- Programmable latency voltage
- Programmable rate and power consumption
- Filter function that supports comparison results
- The output can be redirected to one I/O port or several timer inputs, to trigger the following events:
  - Capture event
  - OCref_clr event (cycle-by-cycle current control)
  - Break event enabling fast PWM shutdown
- Two comparators can be integrated in one window comparator for operation.
- Each comparator can trigger interrupts and wake up the CPU from sleep and shutdown modes (via EXTI controller).
- COMP has 4 positive phase inputs and 4 inverting inputs with polling
  - Polling function for fixed cycle switching
  - Controllable polling channel 1/2/3 or 1/2
  - Optional fixed inverting input

10.3 Functional description of comparator

10.3.1 Introduction

The following figure is a block diagram of the comparator.

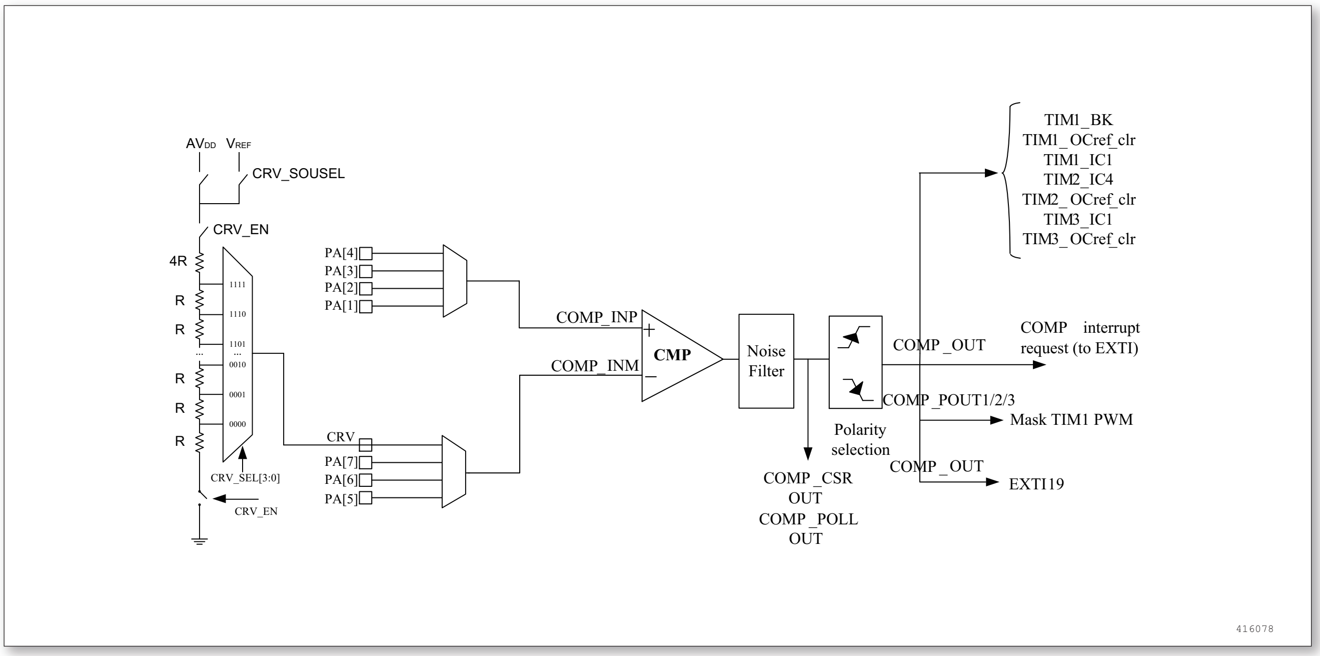


Figure 30. Comparator Block Diagram

10.3.2 Clock

The clock, provided by the COMP clock controller, is synchronized with PCLK (APB2 clock). Before using the comparator, enable the clock enable control bit in the RCC controller.

10.3.3 Comparator switch control

The COMP can be powered up by setting the EN bit of the COMPx_CSR register. When the EN bit is set, it wakes up the COMP from the power-down state, and clearing the EN bit stops the comparator operation.

10.3.4 Comparator input and output

The I/O pin as the comparator input shall be set to the analog mode in the GPIO register.

The comparator output can be internally redirected to various timer inputs:

- As break input, disabling the PWM signal in emergency mode
- As OCref_clr, enabling cycle-by-cycle current control
- As input capture, measuring time sequence

10.3.5 Comparator channel selection

The COMP has four positive-phase inputs and four inverting input channels. The positive-phase input can be selected from four external pins. The inverting input can be divided from three external pins or CRV voltage. The voltage of the CRV can be Select AVDD or internal reference 1.2V divider.

The input channel of COMP can be selected by software in normal working mode, or it can monitor the comparison result of multiple channels by hardware polling in polling mode. It is logically similar to multiple comparators working at the same time.

In normal mode, the comparator compares the signals on the selected INP and INM ports as follows:

- Configure the INP_SEL bit and the INM_SEL bit of the COMPx_CSR register to select the signal to be compared;
- Configure the EN bit of the COMPx_CSR register, and the comparator starts to power on;
- The result of the comparison is stored in the OUT bit of the COMPx_CSR register.

In addition, when COMM’s INM_SEL selects CRV, you need to configure the CRV_SEL bit in the COMP_CRV register, then set CRV_EN.

In the polling mode, the signal on the INP port of COMP4/5 will be periodically polled, and the signal of the INM port can be configured to match the FIXN bit of the COMPx_POLL register to follow the INP port change or by COMPx_CSR The INM_SEL bit is configured. It should be noted that the INP_SEL bit of COMPx_CSR will be disabled when the polling function is started. Similarly, if the FIXN bit of the COMPx_POLL register selects the INM port to follow the INP polling change, INN_SEL bit of COMPx_CSR will also lose its effect. The specific process is as follows:

- Configure the PERIOD bit of the COMPx_POLL register to select the desired polling wait period;
- Configure the FIXN bit of the COMPx_POLL register to determine if the signal on the INM port follows the INP port polling change;
- Configure the POLL_CH bit of the COMPx_POLL register to determine whether the channel to be polled is 1/2/3 or 1/2;
- Configure the POLL_EN bit in the COMPx_POLL register to start the polling function;
- Configure the EN bit of the COMPx_CSR register and the comparator starts to power up.
- The result of the polling comparison is stored in the POUT bit of the COMPx_POLL register, where the POUT[2], POUT[1], and POUT[0] bits respectively store the comparison result of the polling channel 3/2/1.

10.3.6 Interrupt and wakeup

The output of the comparator can be internally connected to an external interrupt and event controller. Each comparator has its own EXTI signal, enabling triggering interrupts or events. The same mechanism can be used to exit from the low power mode.

Refer to Interrupts and events section of the datasheet for details.

10.3.7 Power consumption mode

In specific applications, the optimal results can be obtained by adjusting the power consumption and response time of the comparator.

The MODE bit in the COMPx_CSR register is configured as follows:

- 00: High speed/high power consumption
- 01: Medium speed/medium power consumption
- 10: Low speed/low power consumption

- 11: Very low speed/very low power consumption

10.3.8 Comparator locking mechanism

Comparators can be used for safety purposes, such as overcurrent or overheat protection. In some applications with specific requirements, it is necessary to ensure that comparator settings will not be changed by invalid register access or failure of program counter.

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

Once being configured, the LOCK bit shall be set to 1, making the entire COMPx_CSR register read-only, including the LOCK bit. The write protection can only be cleared by resetting MCU.

10.3.9 Latency

The configurable latency voltage of the comparator can prevent noise signals generated by invalid output changes, and the latency can be disabled without latency voltage .

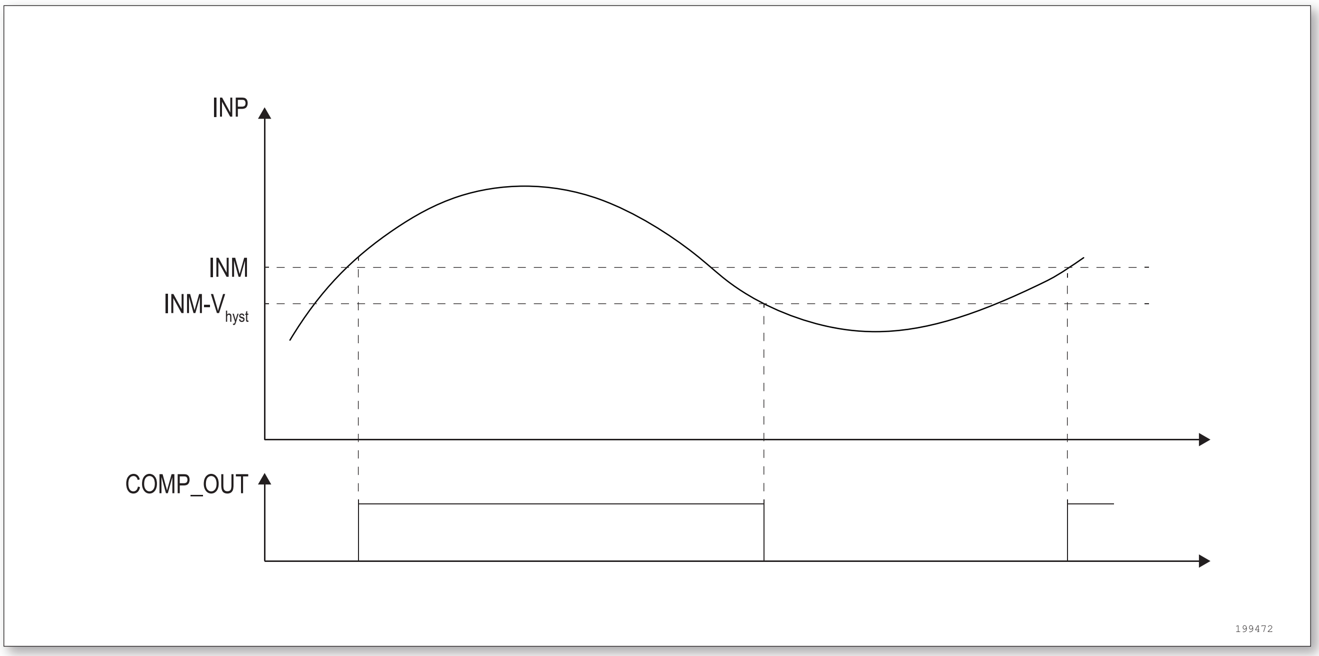


Figure 31. Comparator Latency

10.4 Description of comparator register

Table 36. Summary of Compare Register

Offset	Acronym	Register Name	Reset	Section
0x00	COMPx_CSR(x=1)	Comparator x(x=1)Control and Status Register	0x00000000	section 10.4.1
0x18	COMP_CRV	Comparator external reference voltage register	0x00000000	section 10.4.2
0x1C	COMPx_POLL(x=1)	Comparator x(x=1)polling register	0x00000000	section 10.4.3

10.4.1 Comparator control status register(COMPx_CSR)(x=1)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OUT	Reserved									OFLT			HYST	
rw	r										rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL	Res.	OUT_SEL				Res.	INP_SEL		Res.	INM_SEL		MODE		Res.	EN
rw		rw	rw	rw	rw		rw	rw		rw	rw	rw	rw		rw

Bit	Field	Type	Reset	Description
31	LOCK	rw	0x00	Comparator lock These bit can only be written once, set by software to '1', cleared by system reset. It makes all control bits of comparator x read-only. 1: COMPx_CSR read-only. 0: COMPx_CSR is readable and writable
30	OUT	r	0x00	Comparator x lock Read-only, reflecting the output state of the comparator x. 1: High output (non-inverting input is higher than inverting input) 0: Low output (non-inverting input is lower than inverting input)
29 : 21	Reserved			Always read as 0.
20 : 18	OFLT	rw	0x00	Comparator output filter These bits control the output filtering of comparator x, and the continuous PCLK2 clock comparator output is considered to be a valid result, otherwise it remains unchanged. 111: 512 clock cycles 110: 256 clock cycles 101: 128 clock cycles 100: 64 clock cycles 011: 32 clock cycles 010: 16 clock cycles 001: 4 clock cycles 000: 1 clock cycle, no filtering
17 : 16	HYST	rw	0x00	Comparator x hysteresis These bits control the hysteresis voltage of comparator x. 11: 90mV 10: 30mV 01: 15mV 00: 0mV

Bit	Field	Type	Reset	Description
15	POL	rw	0x00	Comparator x output polarity This bit is used to switch the comparator x output polarity. 1: inverted output 0: Non-inverting output
14	Reserved			Always read as 0.
13 : 10	OUT_SEL	rw	0x00	Comparator x output selection These bits are used to select the x output direction. 0010: Timer 1 brake input 0110: Timer 1 Ocrefclear input 0111: Timer 1 input capture 1 1000: Timer 2 input capture 4 1001: Timer 2 OCrefclear input 1010: Timer 3 input capture 1 1011: Timer 3 Ocrefclear input Other: No choice
9	Reserved			Always read as 0.
8 : 7	INP_SEL	rw	0x00	Comparator x normal phase input selection These bits are used to select the source connected to the non-inverting input of comparator x. 00: COMP1_INP0(PA1) 01: COMP1_INP1(PA2) 10: COMP1_INP2(PA3) 11: COMP1_INP3(PA4)
6	Reserved			Always read as 0.
5 : 4	INM_SEL	rw	0x00	Comparator x inverting input selection These bits are used to select the source of the inverting input connected to comparator x. 00: COMP1_INM0(PA5) 01: COMP1_INM1(PA6) 10: COMP1_INM2(PA7) 11: COMP1_INM3(CRV)
3 : 2	MODE	rw	0x00	Comparator x mode Comparator x operating mode control bit, allowing adjustment of rate and loss. 11: Very low power 10: Low power 01: medium rate 00: High rate
1	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
0	EN	rw	0x00	Comparator xEnable This bit is the Comparator switch control bit. 1: Comparator x opens 0: Comparator x closes

10.4.2 Comparator external reference voltage register(COMP_CRV)

Address offset: 0x18

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										CRV_ SRC	CRV_EN	CRV_SEL			
										rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 6	Reserved			Always read as 0.
5	CRV_SRC	rw	0x00	Comparator external reference voltage source select 0: AVDD 1: VREF
4	CRV_EN	rw	0x00	Comparator external reference voltage enable 1: Comparator External reference voltage enable. 0: Comparator External reference voltage is prohibited.
3 : 0	CRV_SEL	rw	0x00	Comparator external reference voltage select Select Comparator external reference voltage. 0000: 1/20AVDD / VREF 0001: 2/20AVDD / VREF 0010: 3/20AVDD / VREF 0011: 4/20AVDD / VREF 0100: 5/20AVDD / VREF 0101: 6/20AVDD / VREF 0110: 7/20AVDD / VREF 0111: 8/20AVDD / VREF 1000: 9/20AVDD / VREF 1001: 10/20AVDD / VREF 1010: 11/20AVDD / VREF 1011: 12/20AVDD / VREF 1100: 13/20AVDD / VREF 1101: 14/20AVDD / VREF 1110: 15/20AVDD / VREF 1111: 16/20AVDD / VREF

10.4.3 Comparator polling register(COMPx_POLL)(x=1)

Address offset: 0x1c

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					POUT			Res.	PERIOD			Res.	FIXN	POLL_CH	POLL_EN
					r	r	r		rw	rw	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 11	Reserved			Always read as 0.
10 : 8	POUT	r	0x00	Polling output Read-only, reflecting the polling channel output status. POUT[0] corresponds to channel 1, POUT[1] corresponds to channel 2, and POUT[2] corresponds to channel 3. 1: High output (non-inverting input is higher than inverting input) 0: low output (non-inverting input is lower than inverting input)
7	Reserved			Always read as 0.
6 : 4	PERIOD	rw	0x00	Polling wait cycle Switch to the next polling channel every n PCLK2 cycles. 111: 128 clock cycles 110: 64 clock cycles 101: 32 clock cycles 100: 16 clock cycles 011: 8 clock cycles 010: 4 clock cycles 001: 2 clock cycles 000: 1 clock cycle
3	Reserved			Always read as 0.
2	FIXN	rw	0x00	Polling inverting input fix 1: Polling channel inverting input fixed. Determined by CSR register INM_SEL. 0: The polling channel inverting input is not fixed. It changes simultaneously with the INP channel, and INM_SEL is invalid.

Bit	Field	Type	Reset	Description
1	POLL_CH	rw	0x00	Comparator Polling Channel 1: Polling channel 1/2/3. 0: polling channel 1/2. Note: INP_SEL is invalid at this time.
0	POLL_EN	rw	0x00	Comparator Polling mode enable (Comparator polling enable) 1: Comparator polling mode enable. 0: Comparator polling mode is disabled.

11 Advanced-control timer(TIM1)

Advanced-control timer(TIM1)

11.1 TIM1 introduction

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 Section Timer Synchronization.

11.2 Main features

TIM1 functions include:

- 16-bit up, down, up/down auto-reload register
- 16-bit programmable prescaler allowing dividing (modifing in real time) the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- outputs with programmable dead-time
- circuit to control the timer with external signals and to interconnect several timers together.
- counter to update the timer registers only after a given number of cycles of the counter
- input to put the timer’s output signals in reset state or in a known state
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture

  - Output compare
  - Break input
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes
- Trigger input for external clock or cycle-by-cycle current management

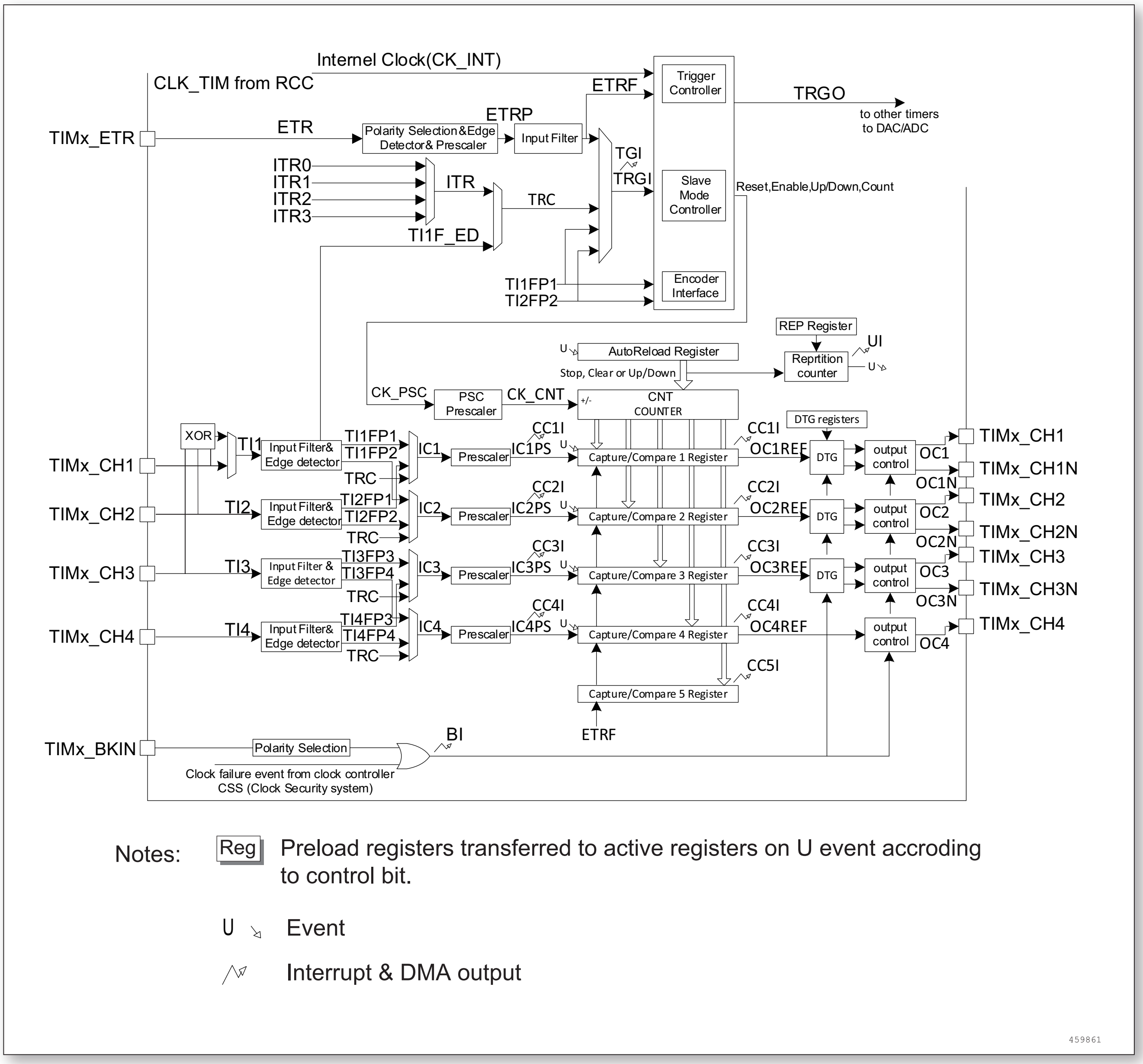


Figure 32. Block Diagram of Advanced-control Timer

11.3 Functional description

11.3.1 Time-base unit

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

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

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in details 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: The counter starts counting 1 clock cycle after setting the CEN bit in the TIMx_CR register.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler factor is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler factor is changed on the fly:

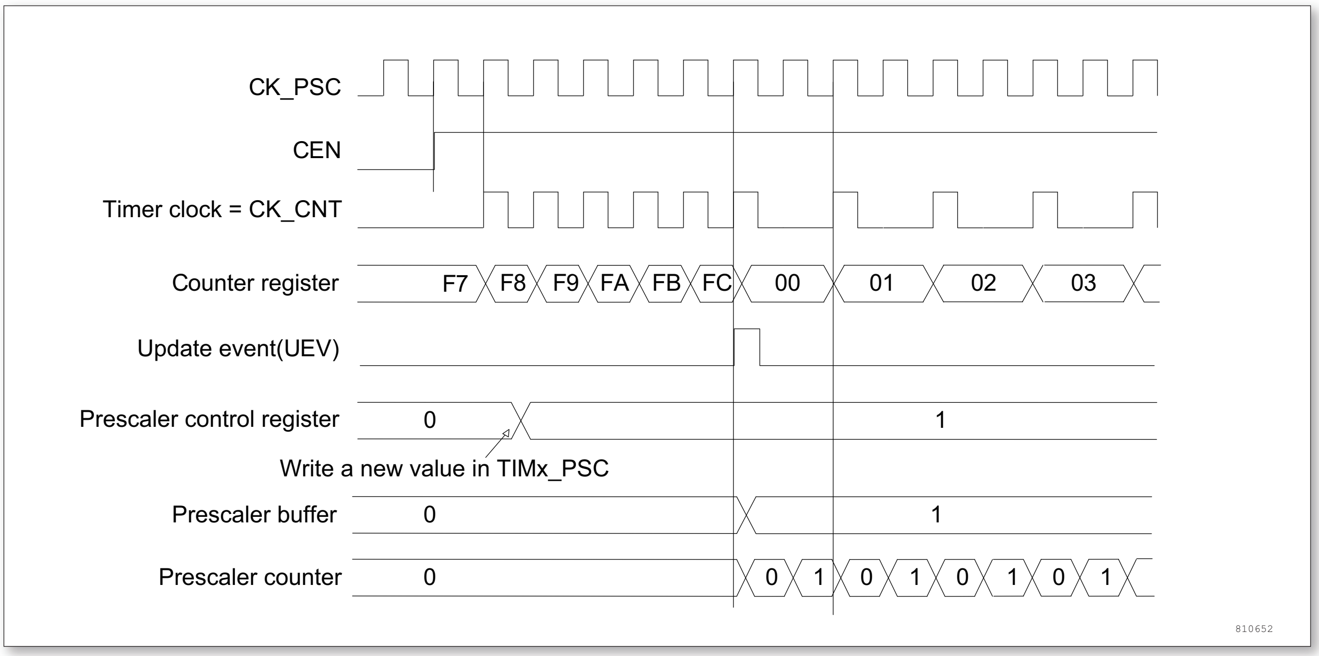


Figure 33. Counter Timing Diagram with Prescaler Division Change from 1 to 2

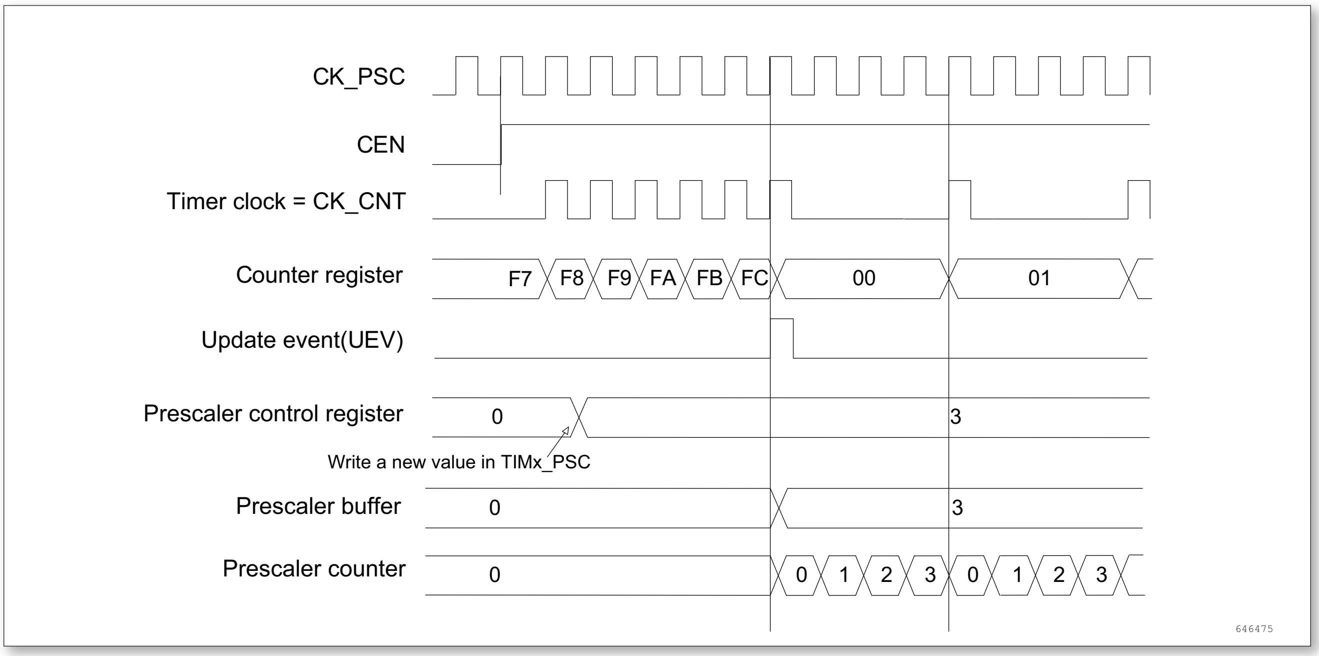


Figure 34. Counter Timing Diagram with Prescaler Division Change from 1 to 4

11.3.2 Counter modes

Upcounting mode

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

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

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

The UEV event can be disabled by software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

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

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

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

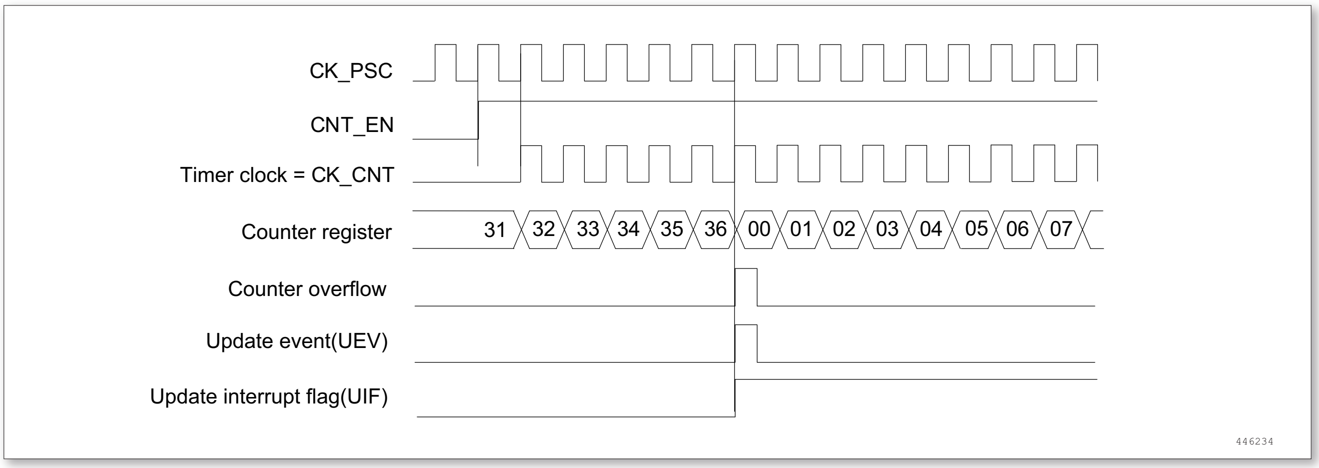


Figure 35. Counter Timing Diagram, Internal Clock Divided by 1

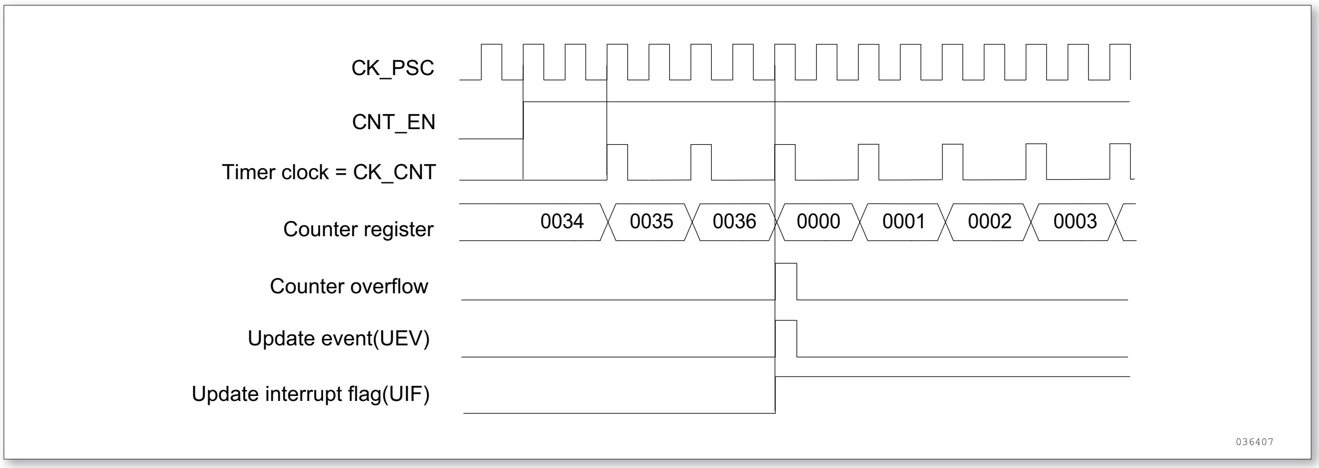


Figure 36. Counter Timing Diagram, Internal Clock Divided by 2

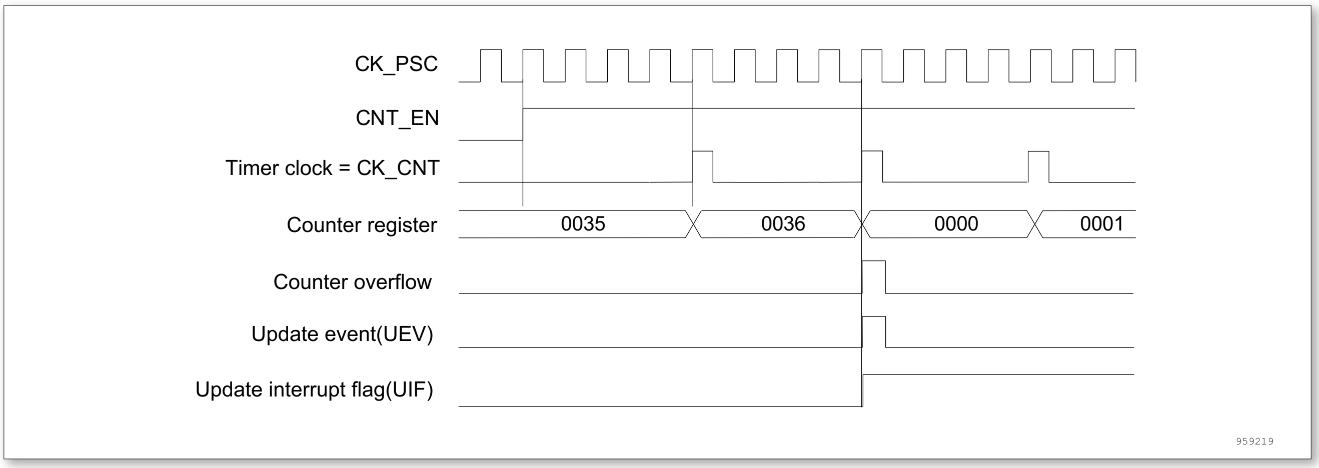


Figure 37. Counter Timing Diagram, Internal Clock Divided by 4

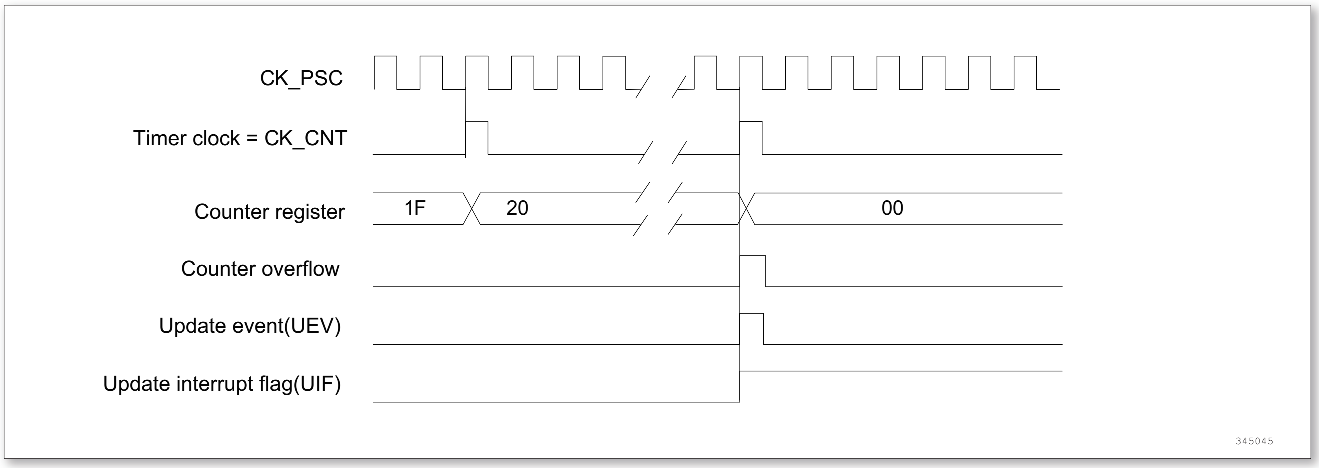


Figure 38. Counter Timing Diagram, Internal Clock Divided by N

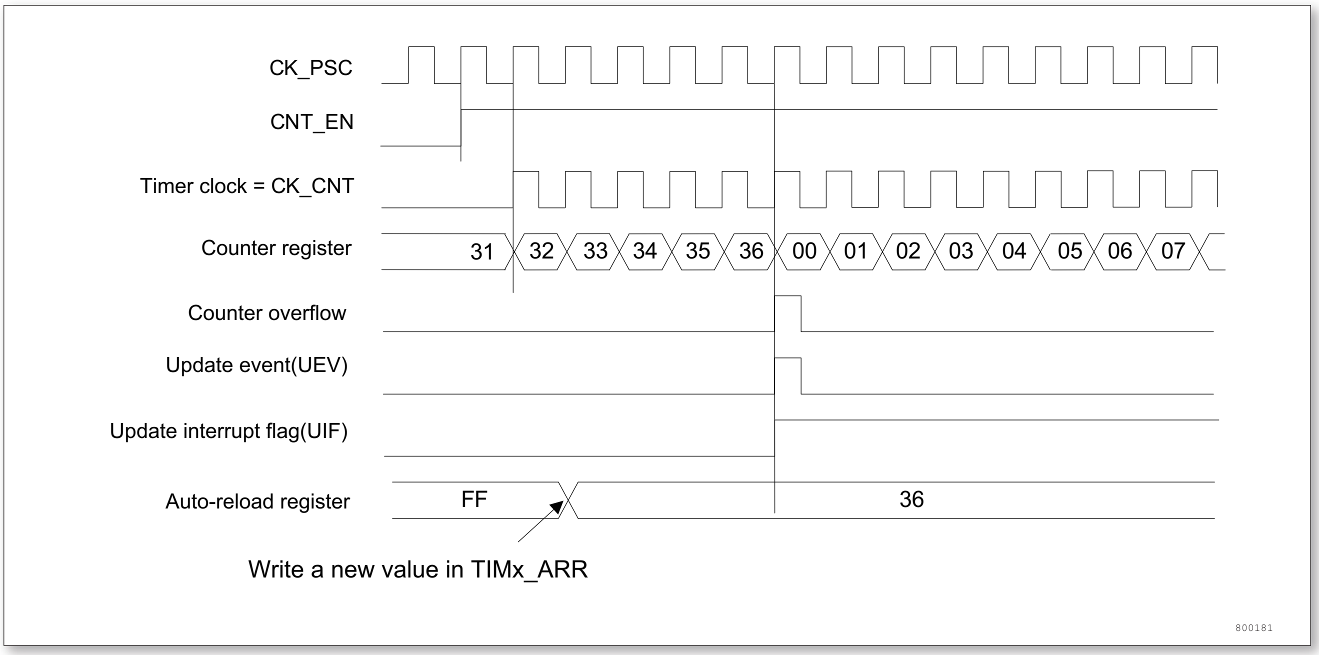


Figure 39. Counter Timing Diagram, Update Event When ARPE = 0 (TiMx_ARR Not Preloaded)

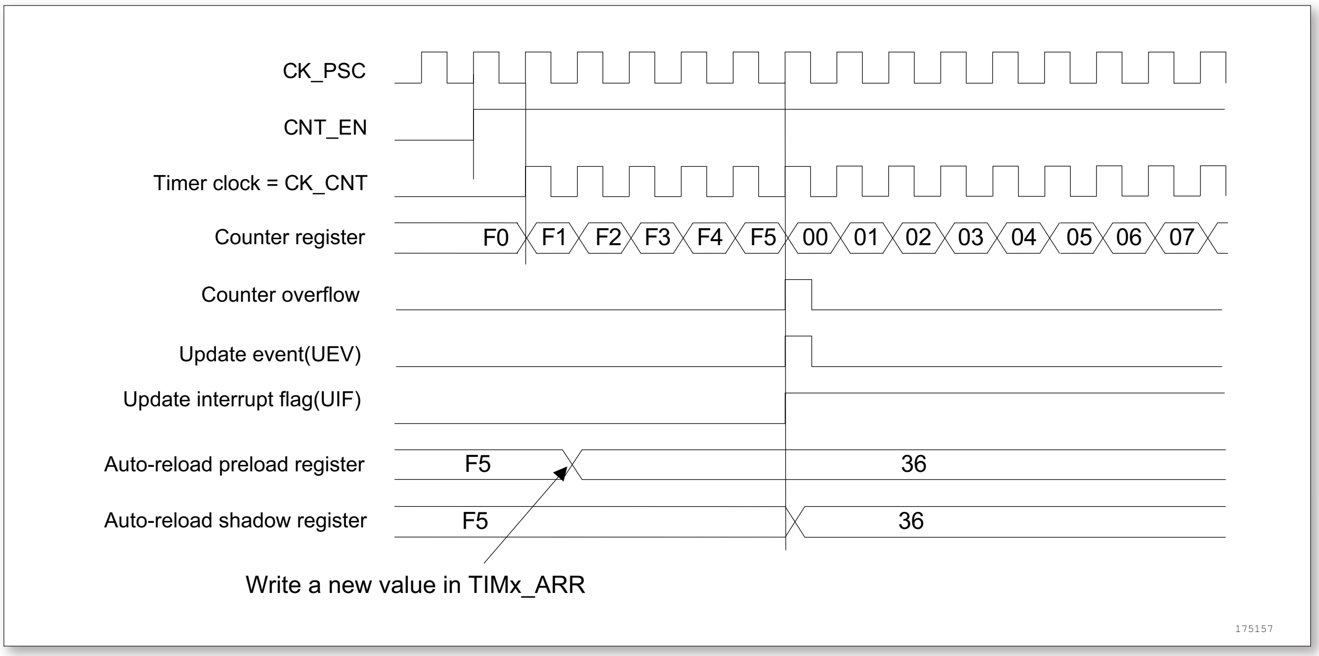


Figure 40. Counter Timing Diagram, Update Event When ARPE = 1 (TiMx_ARR Preloaded)

Downcounting mode

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

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

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag bit(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.

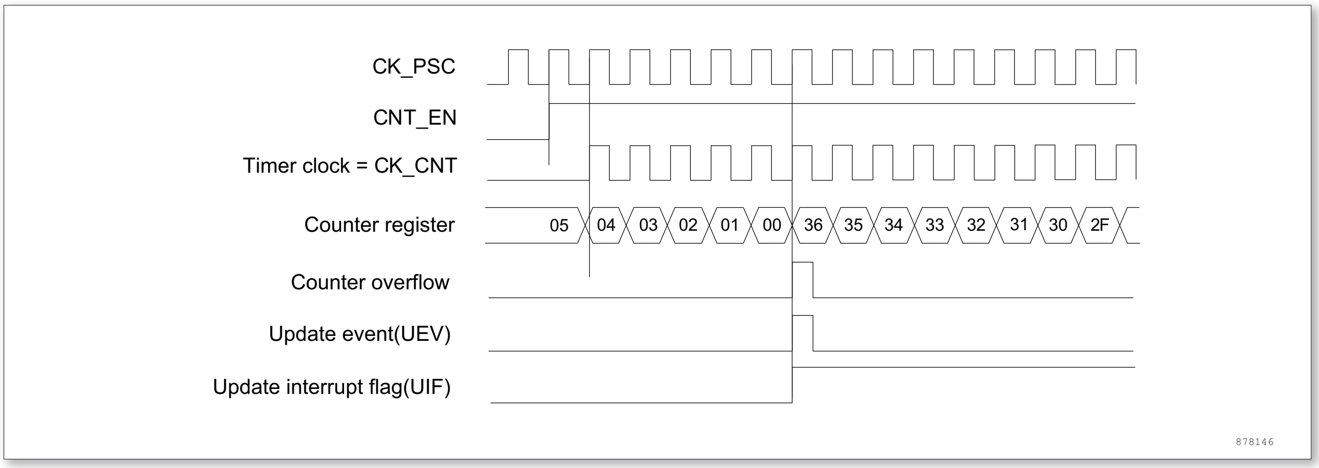


Figure 41. Counter Timing Diagram, Internal Clock Divided by 1

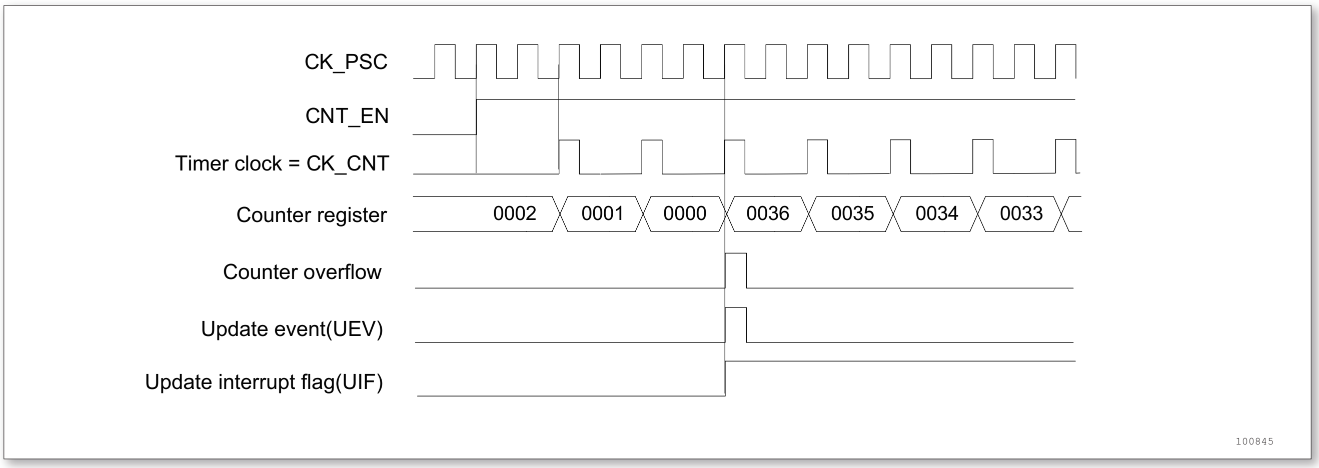


Figure 42. Counter Timing Diagram, Internal Clock Divided by 2

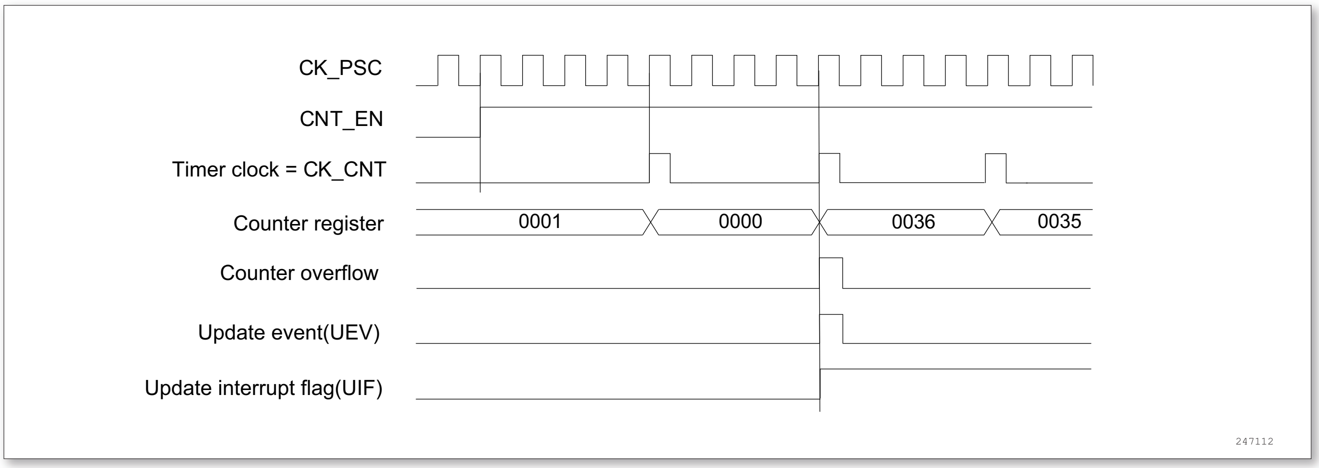


Figure 43. Counter Timing Diagram, Internal Clock Divided by 4

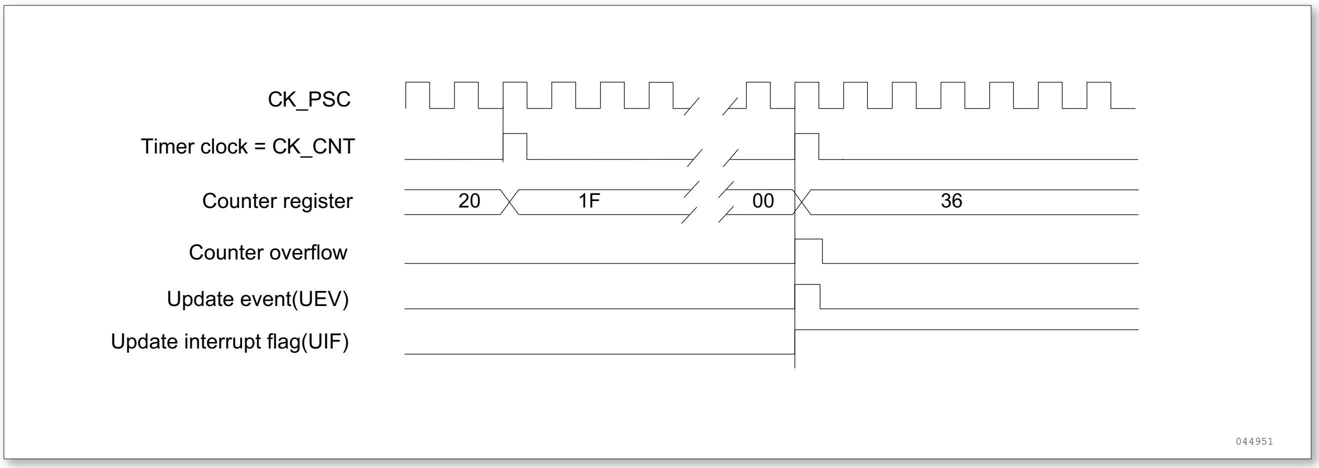


Figure 44. Counter Timing Diagram, Internal Clock Divided by N

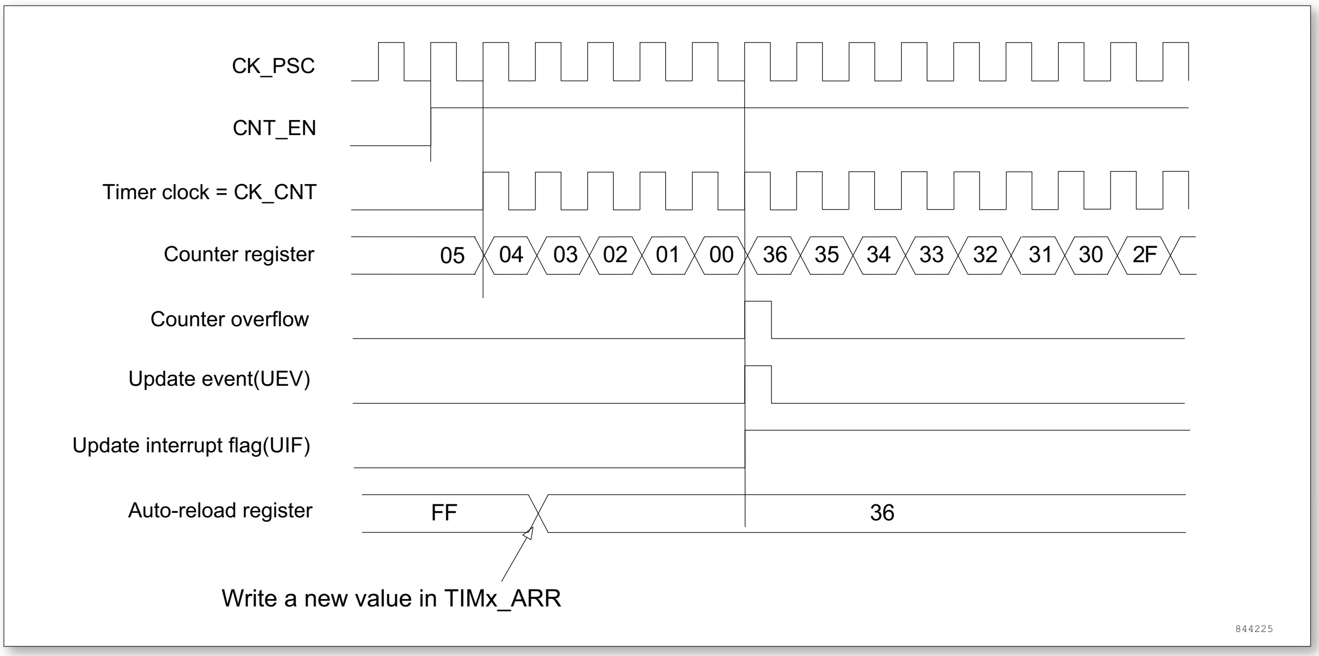


Figure 45. Counter Timing Diagram, Update Event when Repetition Counter is Not Used

Center-aligned mode (Upcounting/Downcounting))

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

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, so does the counter of the prescaler.

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

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

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

Note: If the update source is a counter overflow, the 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.

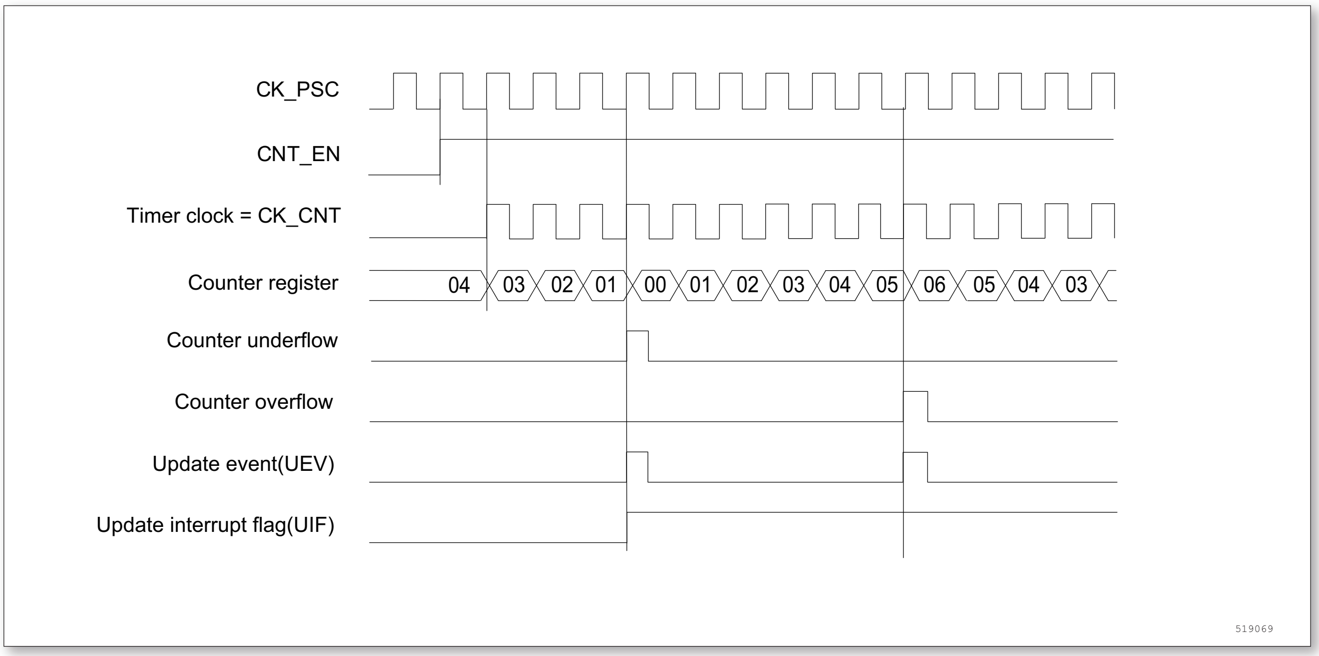


Figure 46. Counter Timing Diagram, Internal Clock Divided by 1, TIMx_ARR = 6

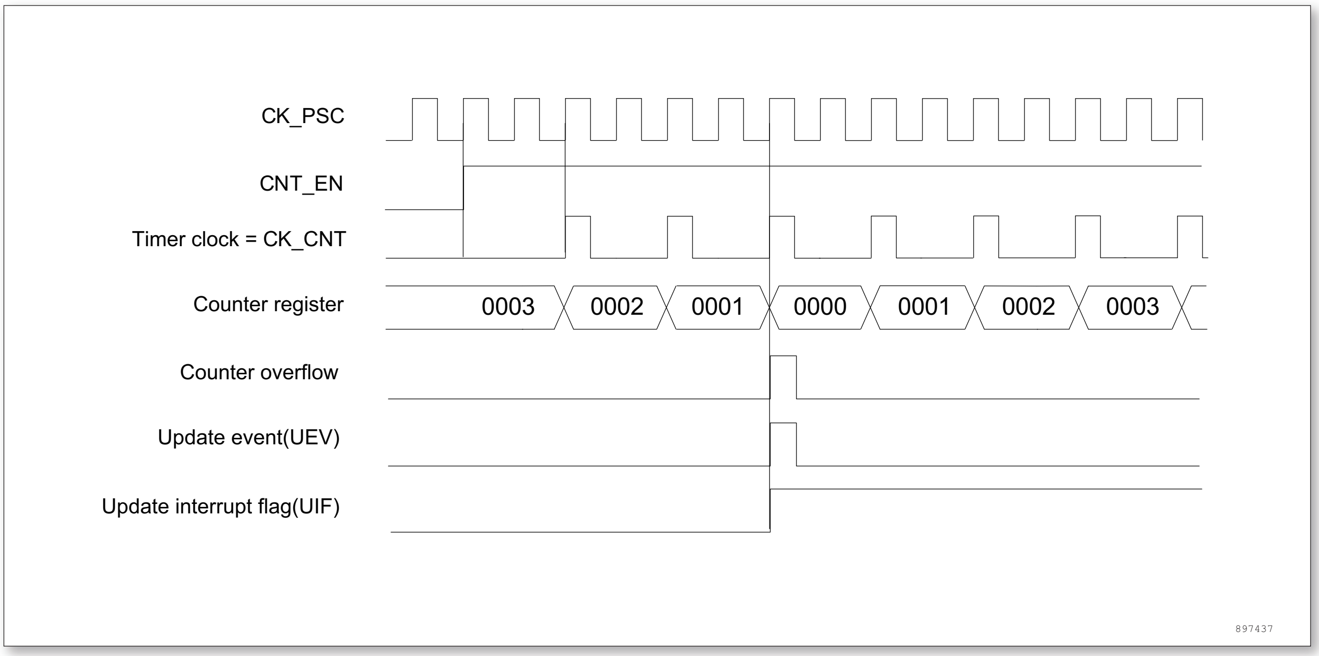


Figure 47. Counter Timing Diagram, Internal Clock Divided by 2

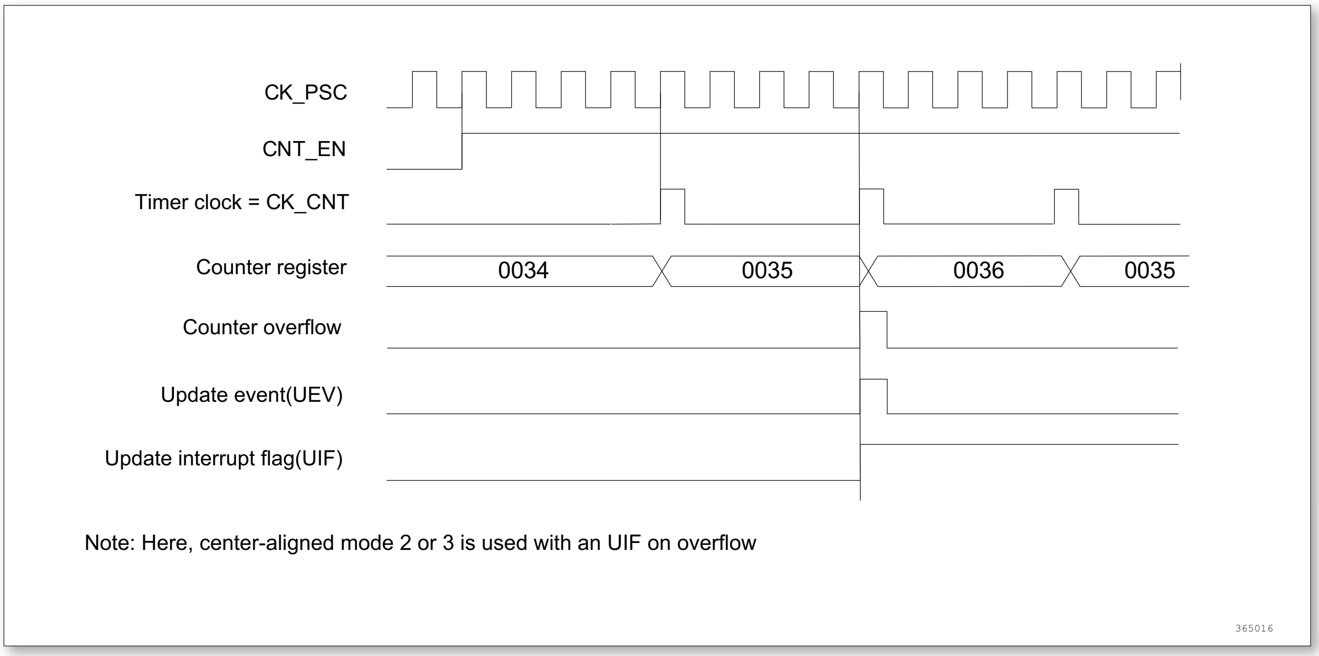


Figure 48. Counter Timing Diagram, Internal Clock Divided by 4, TIMx_ARR = 0×36

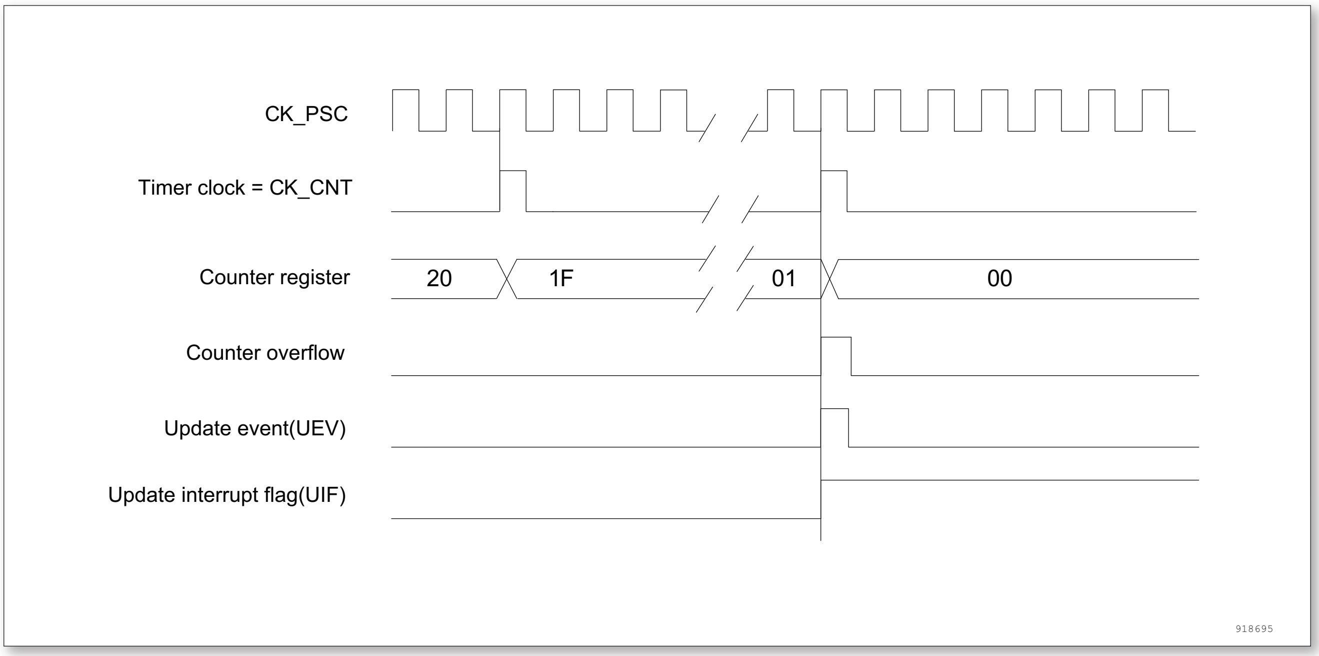


Figure 49. Counter Timing Diagram, Internal Clock Divided by N

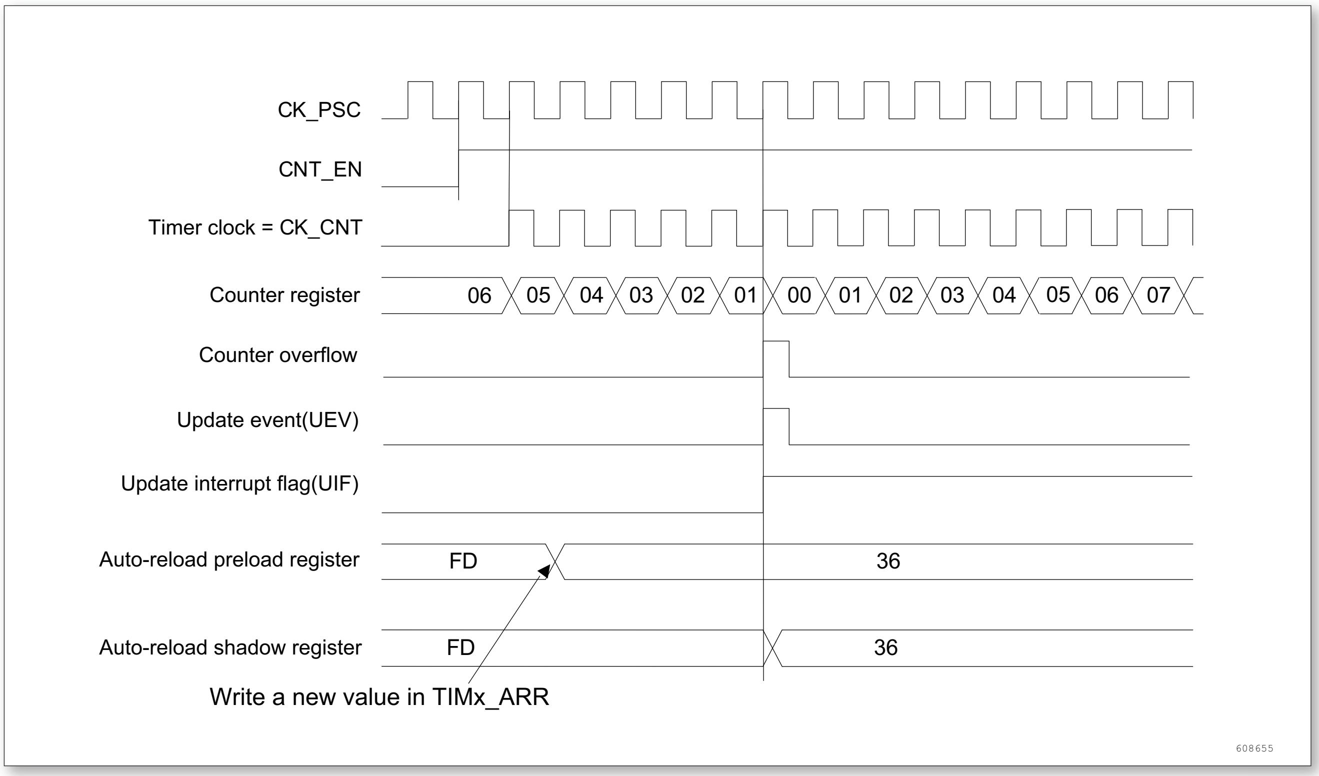


Figure 50. Counter Timing Diagram, Update Event with ARPE = 1(Counter Underflow)

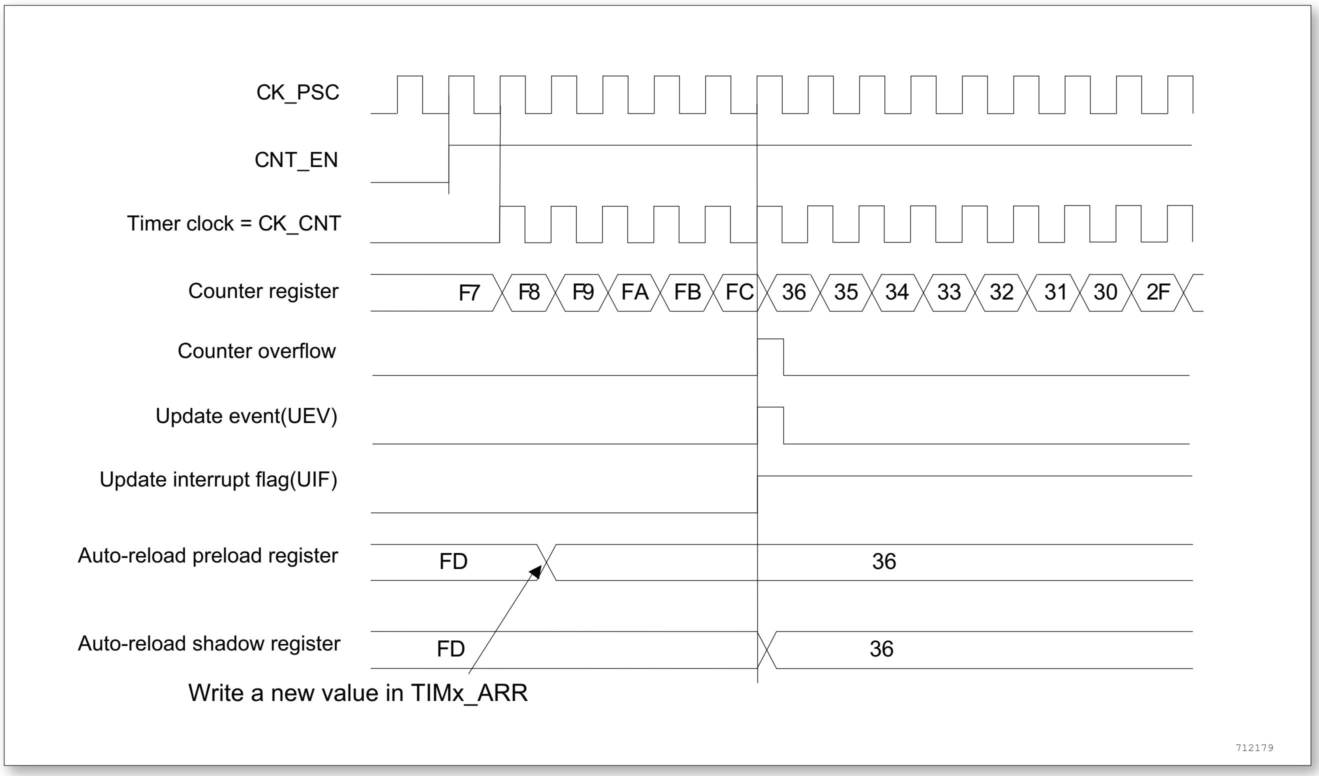


Figure 51. Counter Timing Diagram, Update Event with ARPE = 1(Counter Overflow))

11.3.3 Repetition counter

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

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N 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 49). 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.

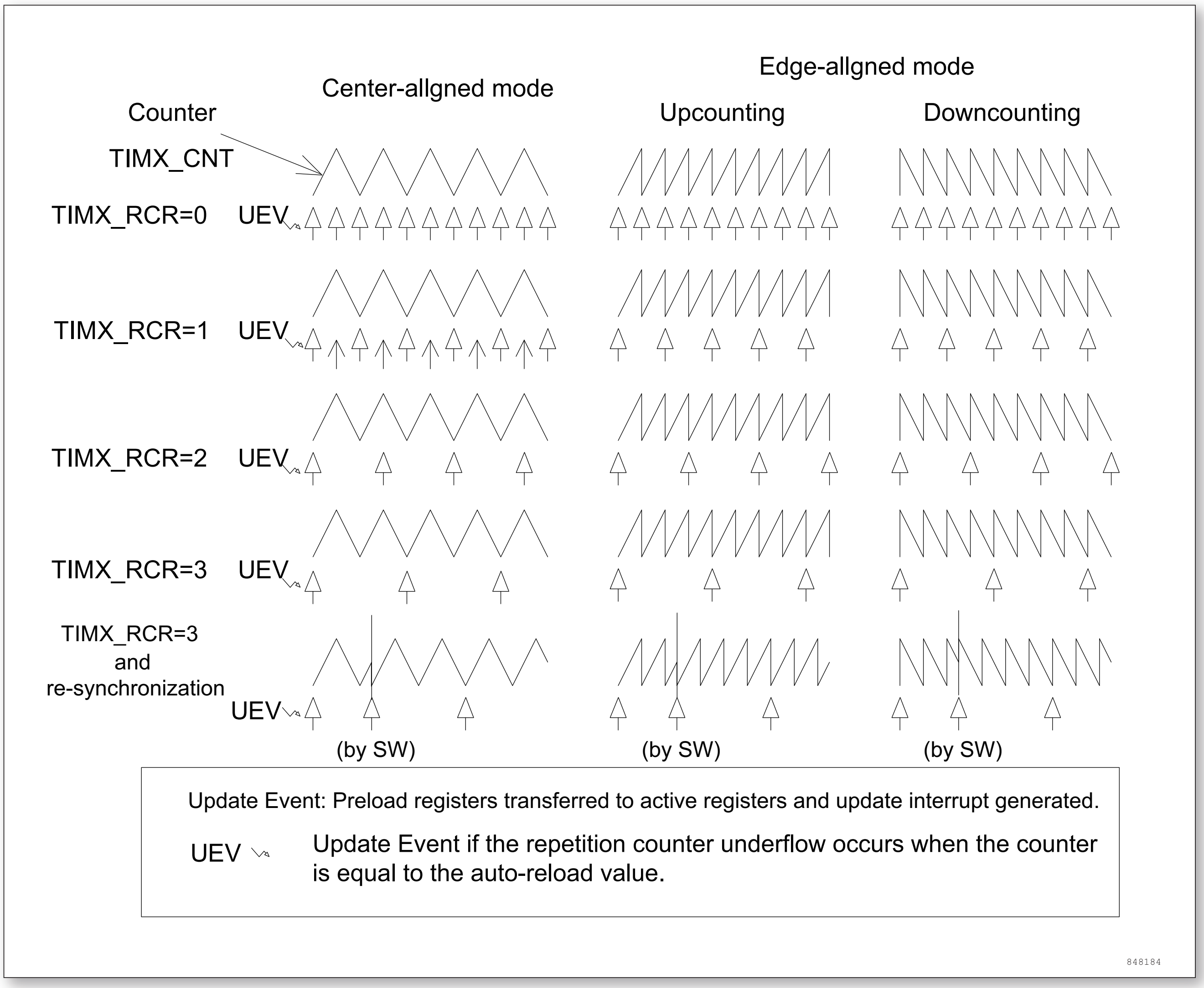

Edge-allgned mode
Center-allgned mode
Counter
Upcounting
Downcounting
TIMX_CNT
TIMX_RCR=0
UEV
TIMX_RCR=1
UEV
TIMX_RCR=2
UEV
TIMX_RCR=3
UEV
TIMX_RCR=3
and
re-synchronization
UEV
(by SW)
(by SW)
(by SW)
Update Event: Preload registers transferred to active registers and update interrupt generated.
UEV
Update Event if the repetition counter underflow occurs when the counter is equal to the auto-reload value.
848184

Figure 52. Update Rate Examples Depending on Modes and TIMx_RCR Register Settings

11.3.4 Clock selection

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

- Internal clock (CK_INT).
- External clock mode1: external input pin (TIx).
- External clock mode2: external trigger input (ETR).
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 2.

Internal clock (CK_INT)

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

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

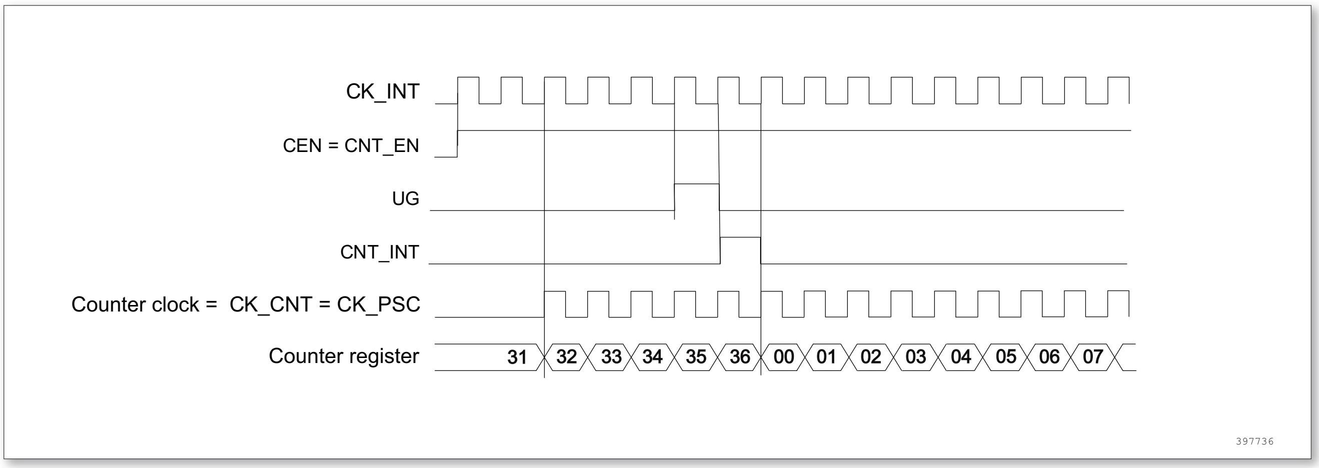


Figure 53. Control Circuit in Normal Mode, Internal Clock Divided By 1

External clock source mode 1

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

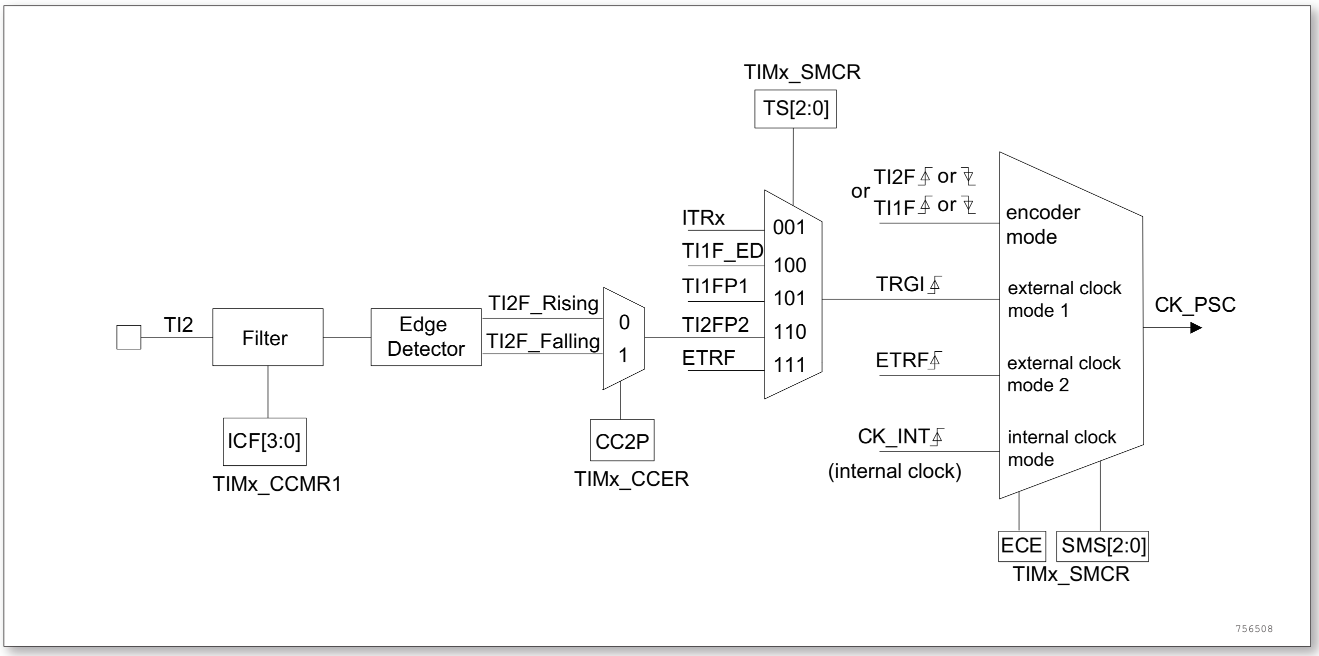


Figure 54. TI2 External Clock Connection Example

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

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = ‘01’ in the TIMx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F=0000).
3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.
4. Select the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.
5. Select TI2 as the trigger input source by writing TS=110 in the TIMx_SMCR register.

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

Note: The capture prescaler is not used for triggering, so the user does not need to configure it. 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.

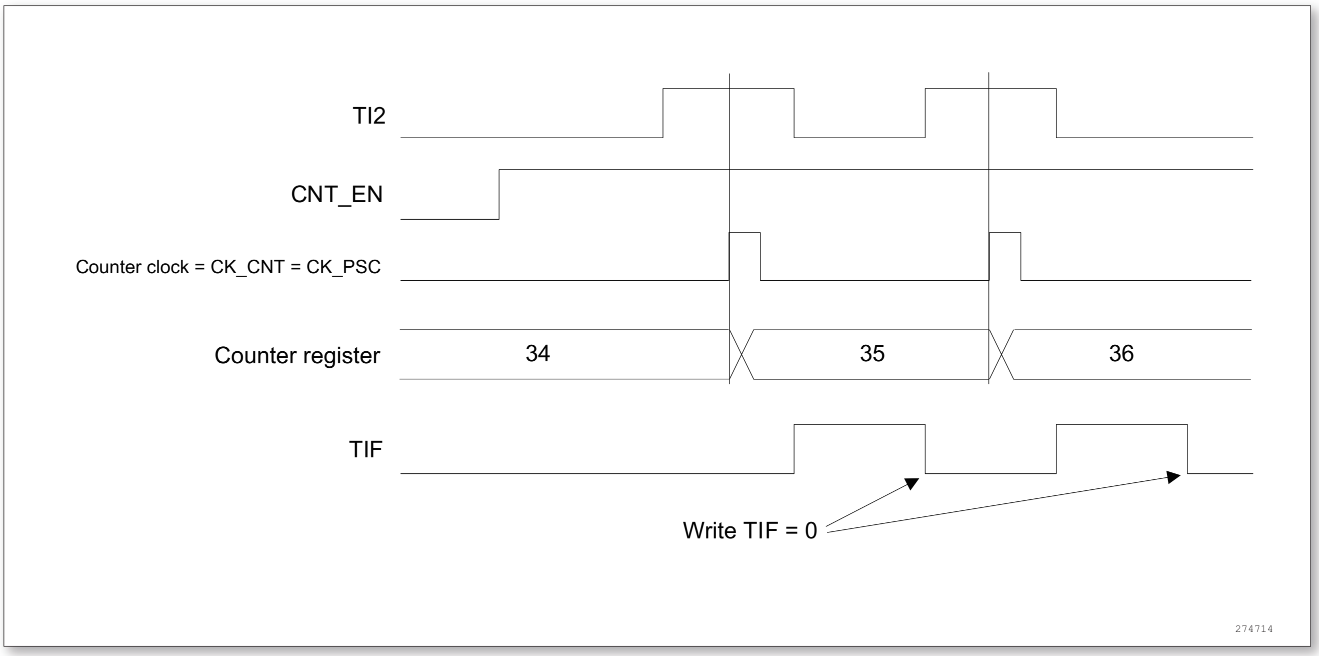


Figure 55. Control Circuit in External Clock Mode 1

External clock source mode 2

This mode is selected by writing ECE = 1 in the TIMx_SMCR register.

The counter can count at each rising or falling edge on the external trigger input ETR.

The following figure gives an overview of the external trigger input block.

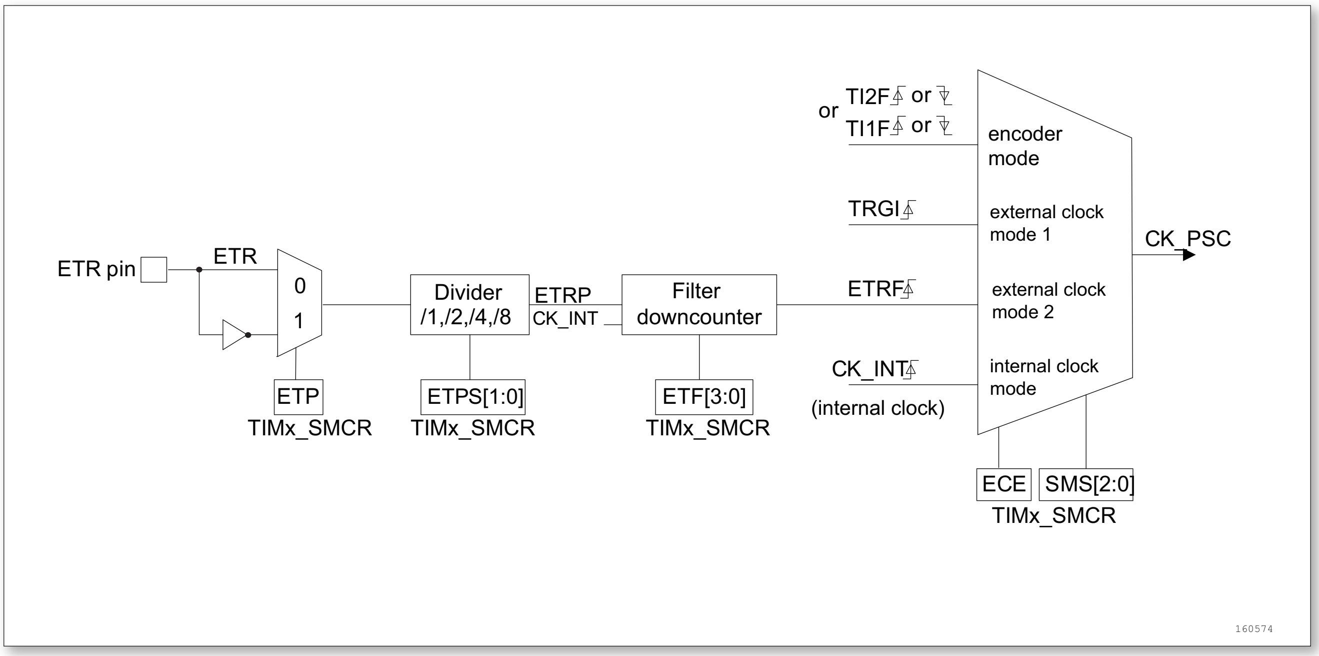


Figure 56. External Trigger Input Block

For example, to configure the upcounter to count once each 2 rising edges on ETR, use the following procedure:

- As no filter is needed in this example, write ETF [3:0]=0000 in the TIMx_SMCR register.
- Set the prescaler by writing ETPS [1:0]=01 in the TIMx_SMCR register.
- Select rising edge detection on the ETR pin by writing ETP=0 in the TIMx_SMCR register.
- Enable external clock mode 2 by writing ECE=1 in the TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter counts once each 2 ETR rising edges.

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

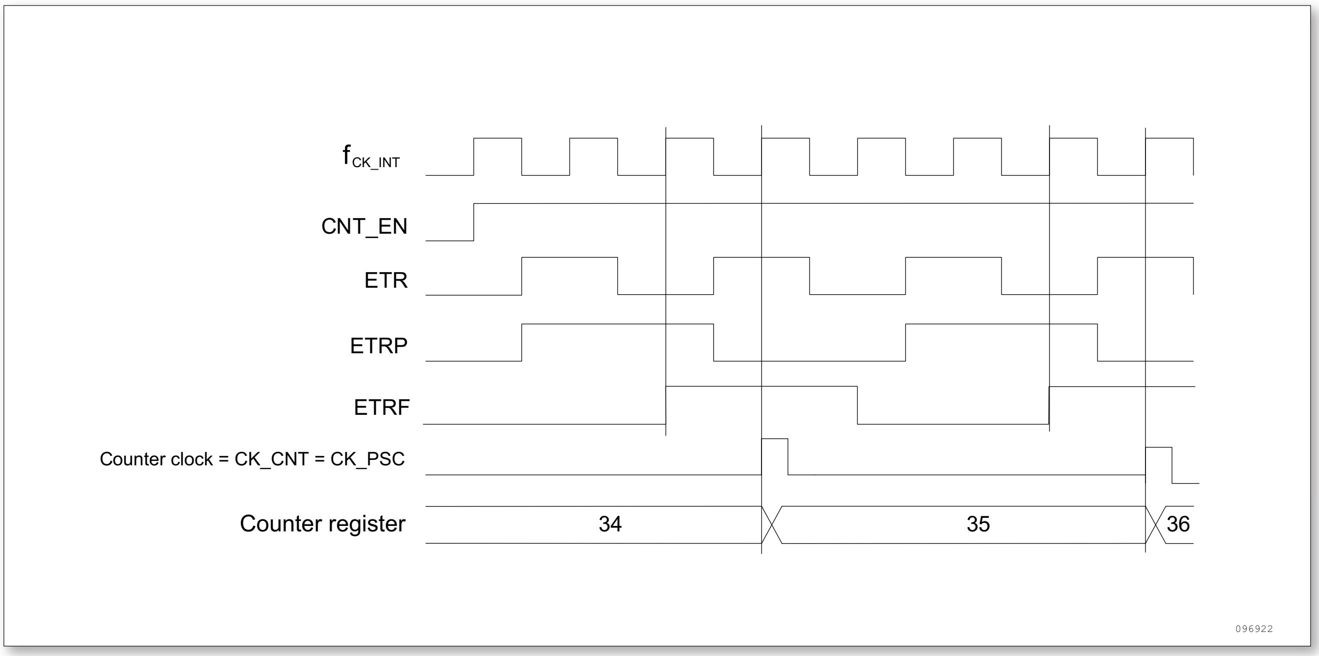


Figure 57. Control Circuit in External Clock Mode 2

11.3.5 Capture/compare channels

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

Figure 58 to Figure 61 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).

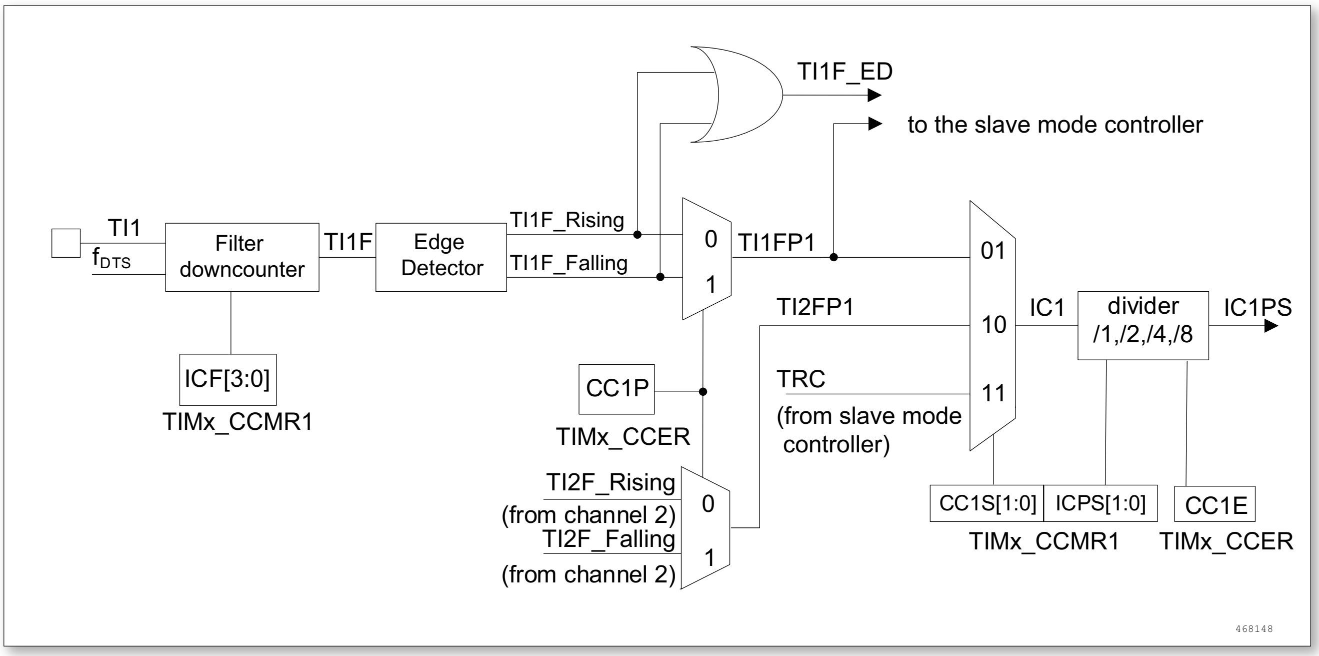


Figure 58. Capture/Compare Channel (Example: Channel 1 Input Stage)

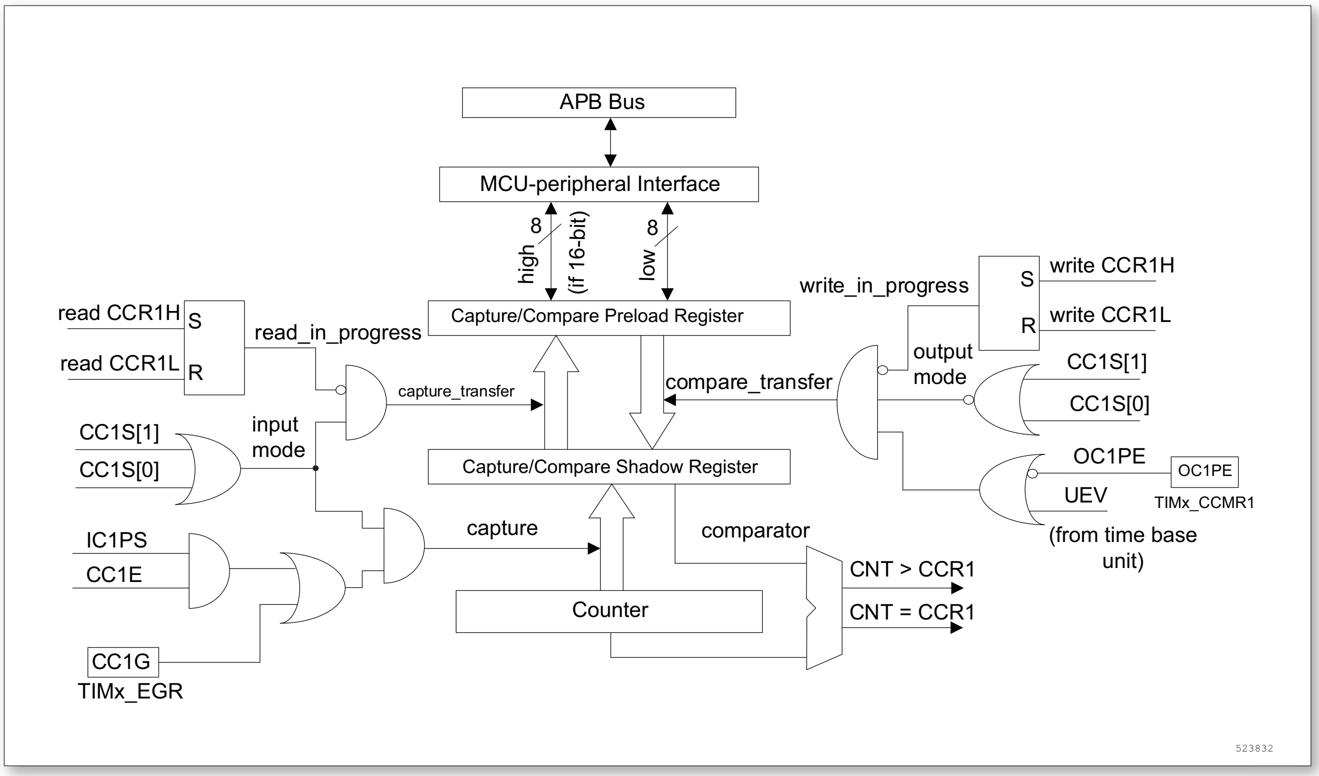


Figure 59. Capture/Compare Channel 1 Main Circuit

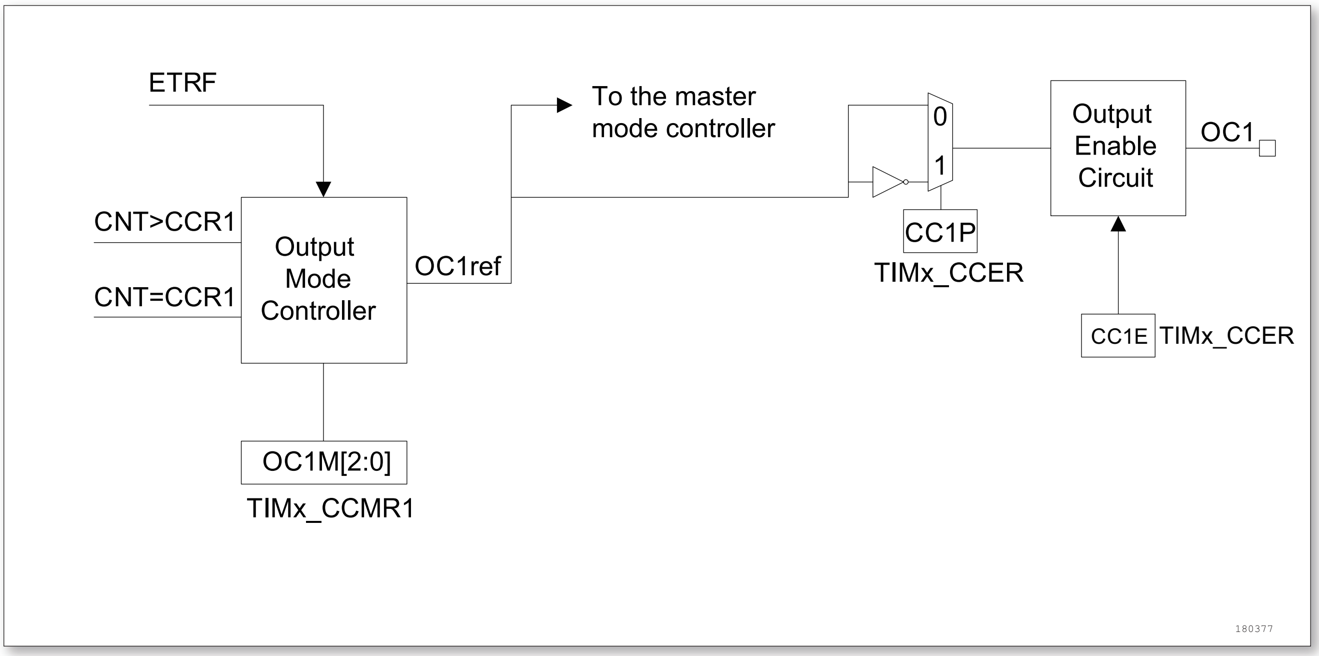


Figure 60. Output stage of Capture/Compare Channel (Channels 1 to 3)

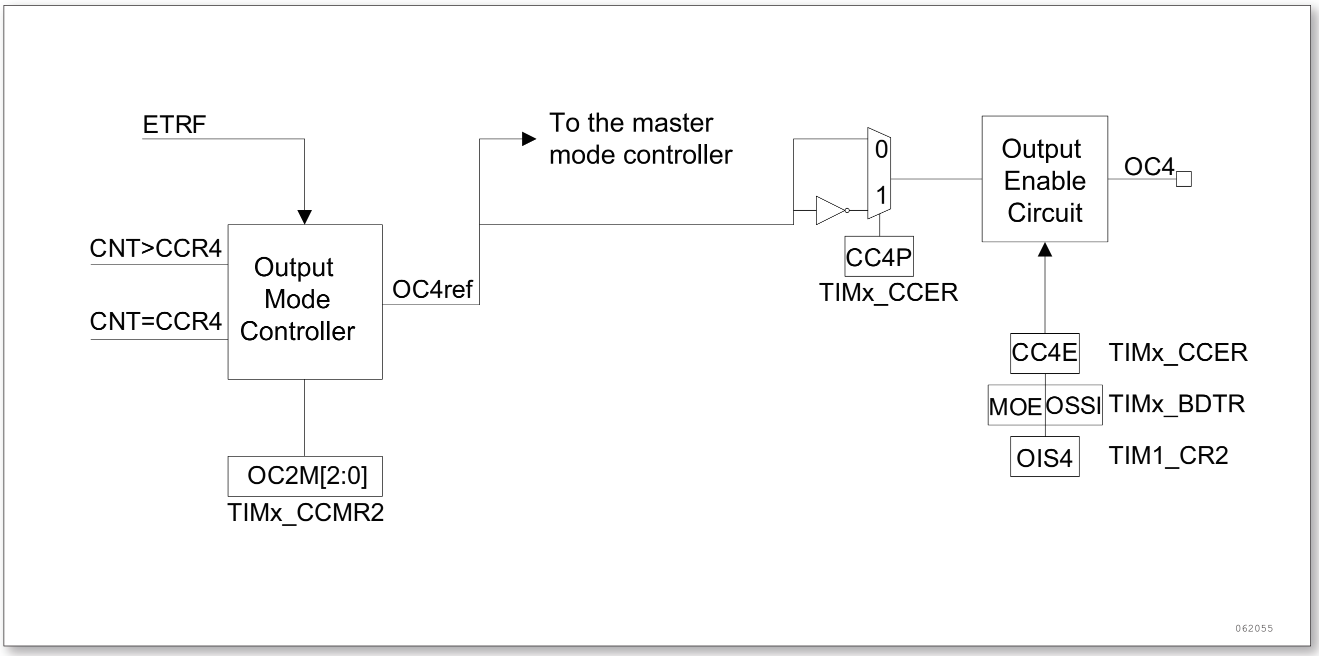


Figure 61. Output stage of Capture/Compare Channel (Channel 4)

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

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

11.3.6 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to '0'.

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

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let' s imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Configure the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register to '1'.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

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

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

11.3.7 PWM input mode

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

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.

- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode. For example, user can measure the period (in TIMx_CCR1 register) and the duty cycle (in TIMx_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):
- Select the active input for TIMx_CCR1: write the CC1S bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMx_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMx_CCER register.

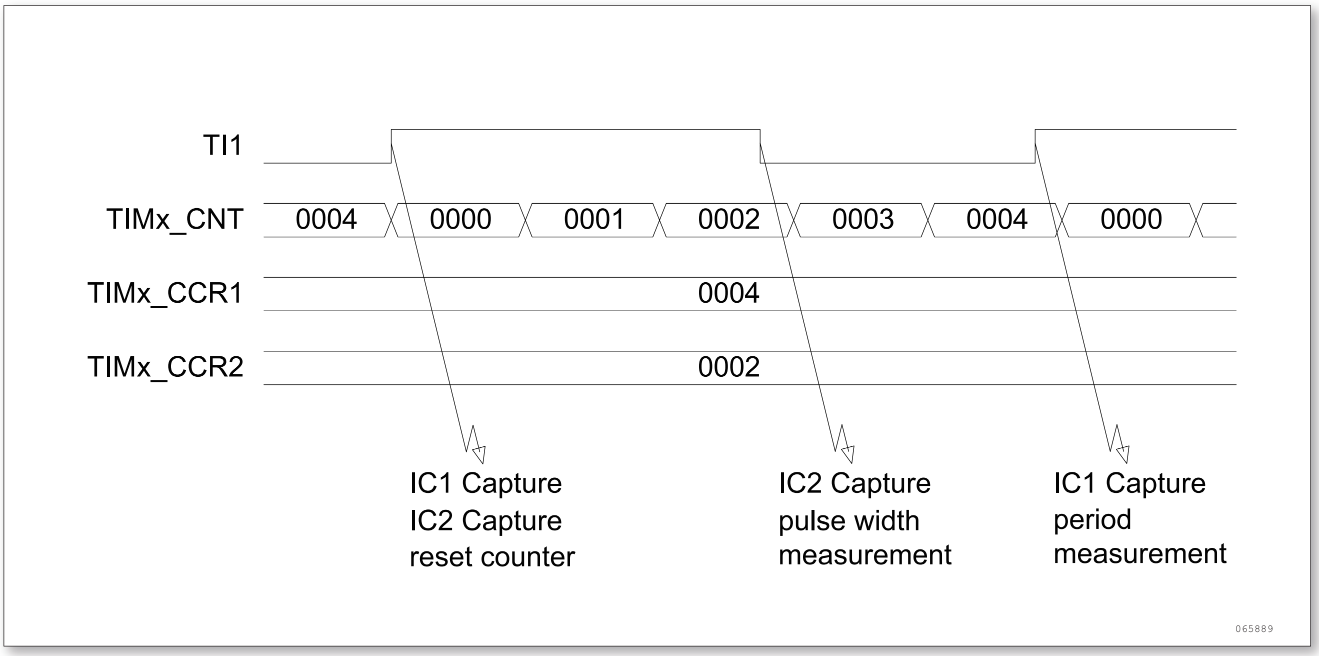


Figure 62. PWM Input Mode Timing

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.

11.3.8 Forced output mode

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

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, in this mode, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

11.3.9 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag bit in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

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

Procedure:

- Select the counter clock (internal, external, 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 with 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’, otherwise

TIMx_CCRx shadow register is updated only at the next update event UEV). An example is given in the following figure.

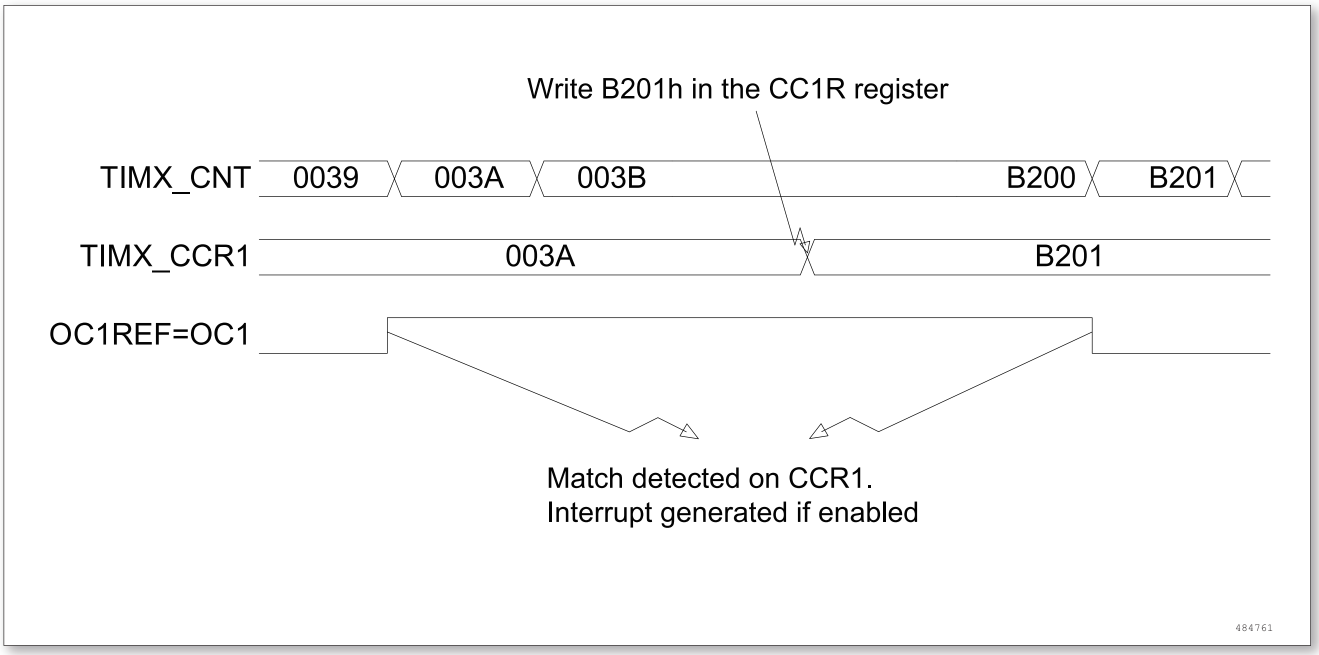


Figure 63. Output Compare Mode, Toggle on OC1

11.3.10 PWM mode

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

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

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

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

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

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

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to section 11.3.2.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx, otherwise it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’. Figure 61 shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

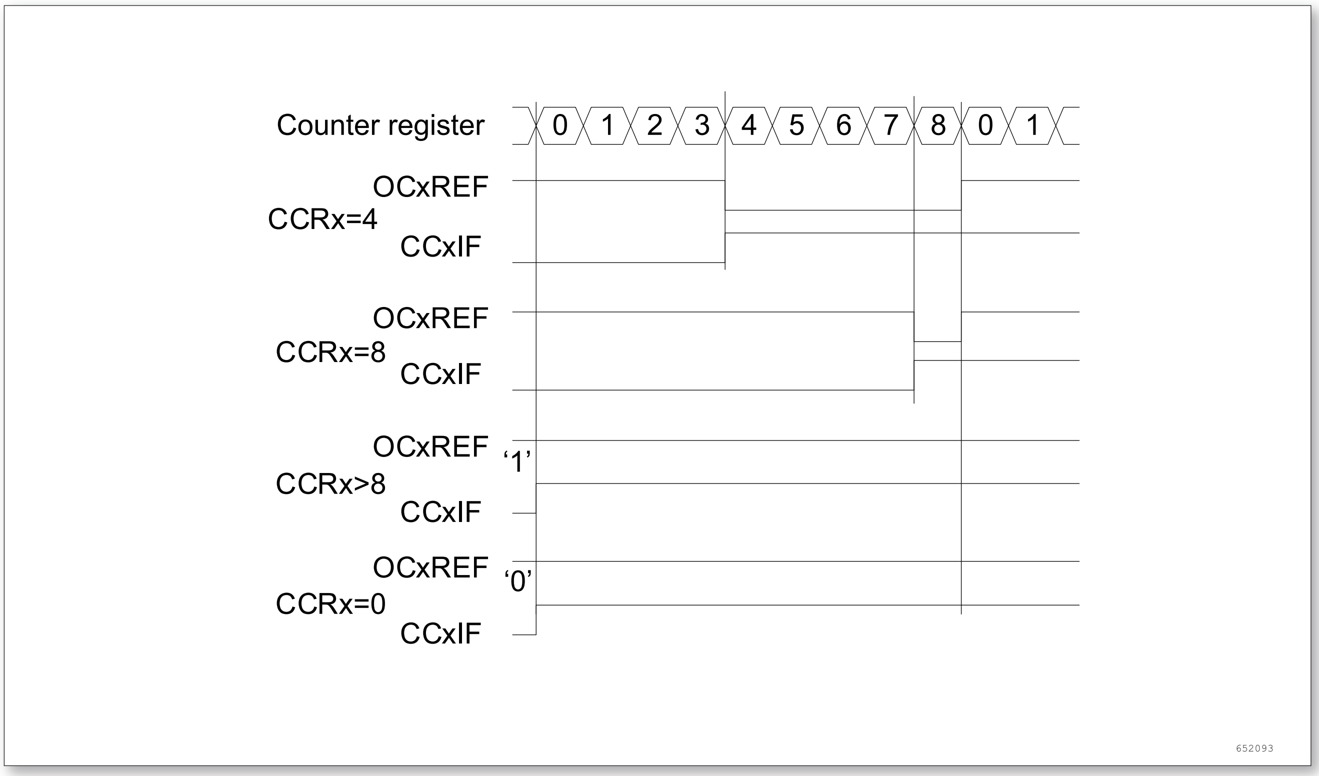


Figure 64. Edge-aligned PWM Waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high. Refer to section 11.3.2.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx, otherwise it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from ‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals). The compare flag is set when the counter counts up, when it counts down or 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 section 11.3.2 Center-aligned Mode.

Figure 65 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.

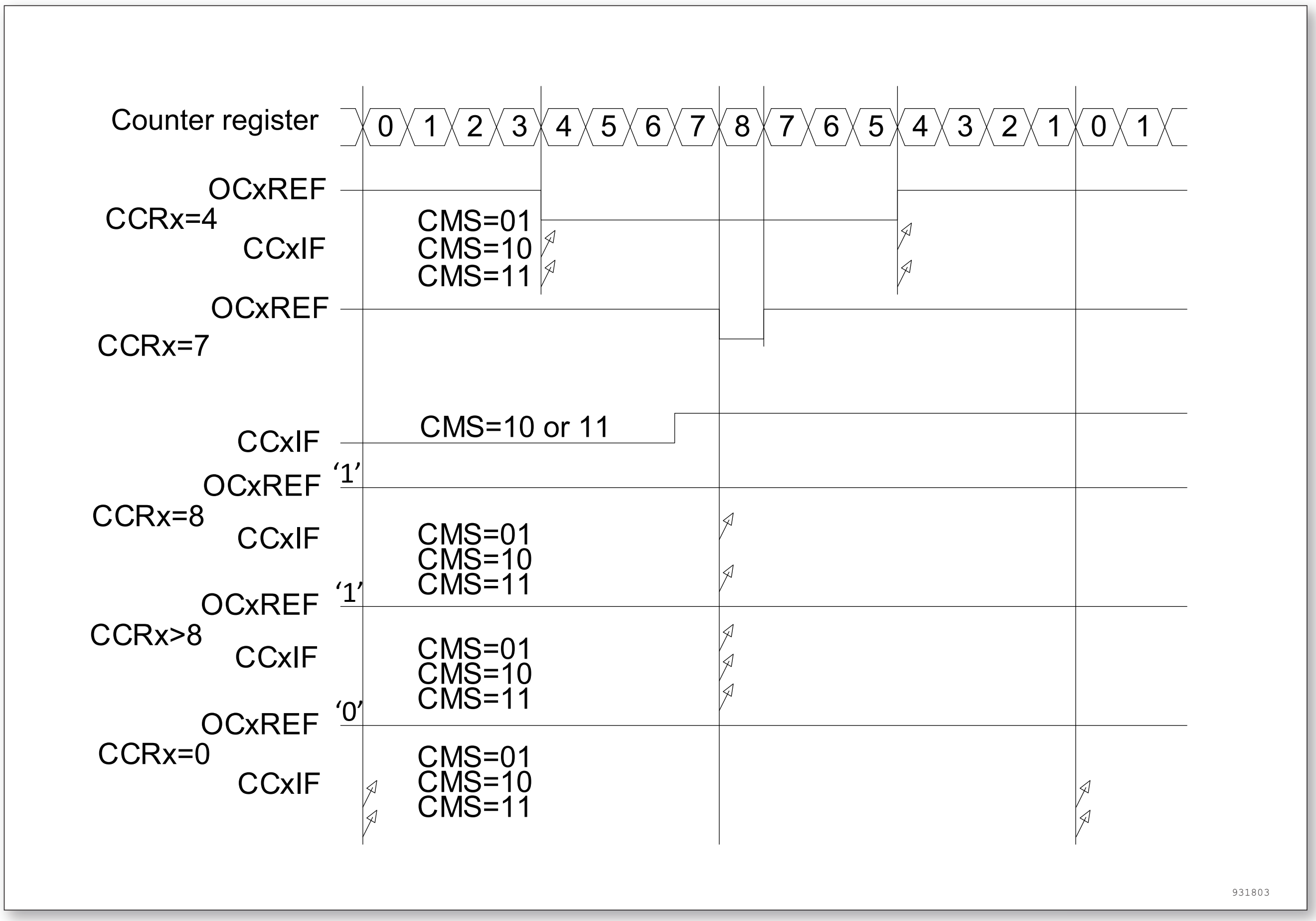


Figure 65. Center-aligned PWM Waveforms (ARR = 8)

Hints in center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT>TIMx_ARR).
  - For example, if the counter was counting up, it will continue to count up.
  - The direction is updated if the user writes 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated
- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write

the counter while it is running.

11.3.11 Complementary outputs and dead-time insertion

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

User can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. Refer to Table 40 Output Control Bits for Complementary OCx and OCxN Channels with Break Feature for more details. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. DTG[7:0] bits of the TIMx_BDTR register are used to control the dead-time generation for all channels. From a reference waveform OCxREF, it generates 2 outputs OCx and OCxN. If OCx and OCxN are active high:

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

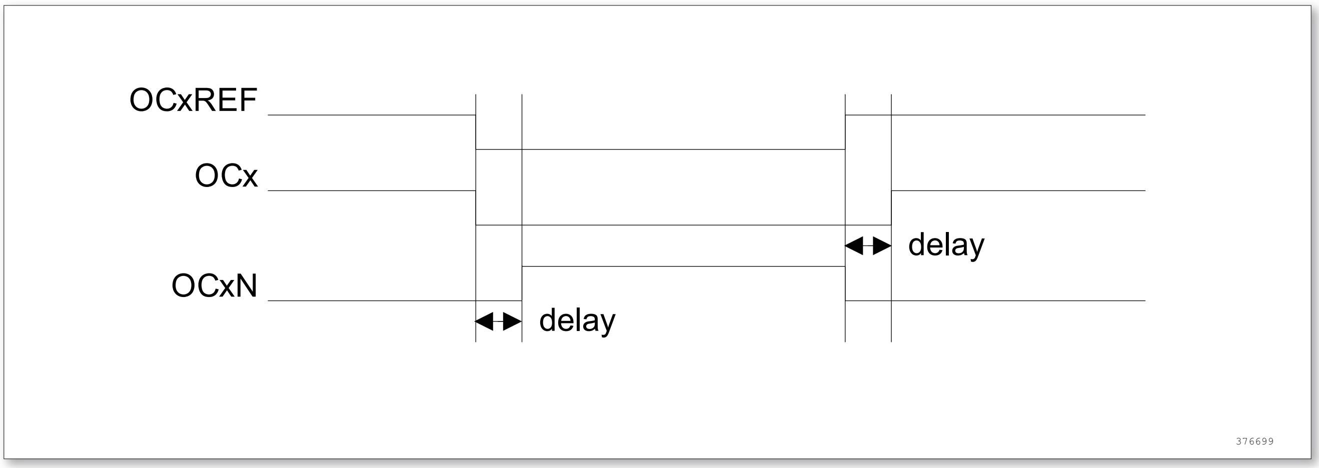


Figure 66. Complementary Output with Dead-time Insertion

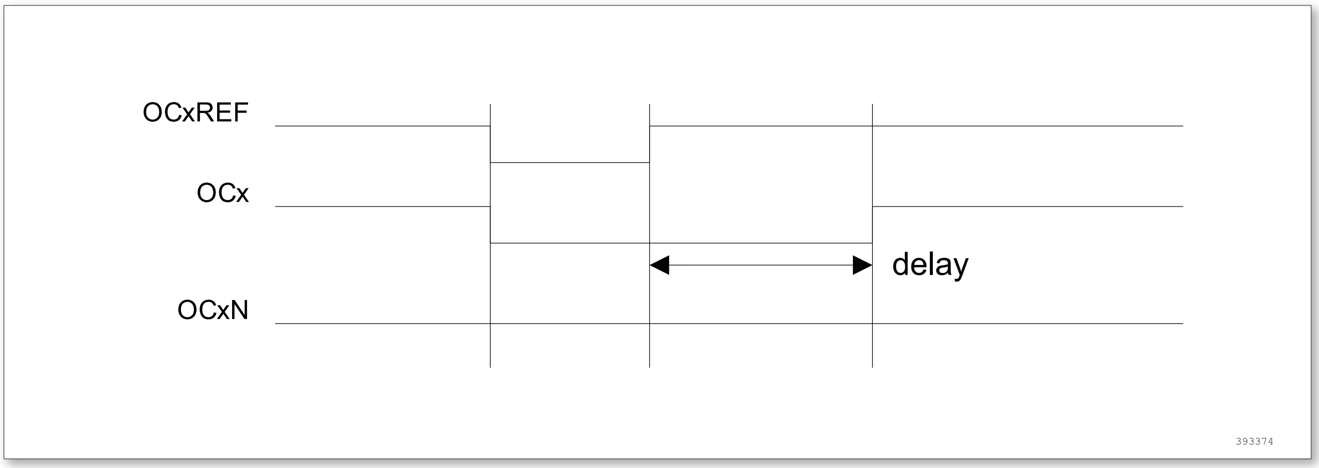


Figure 67. Dead-time Waveforms with Delay Greater Than the Negative Pulse

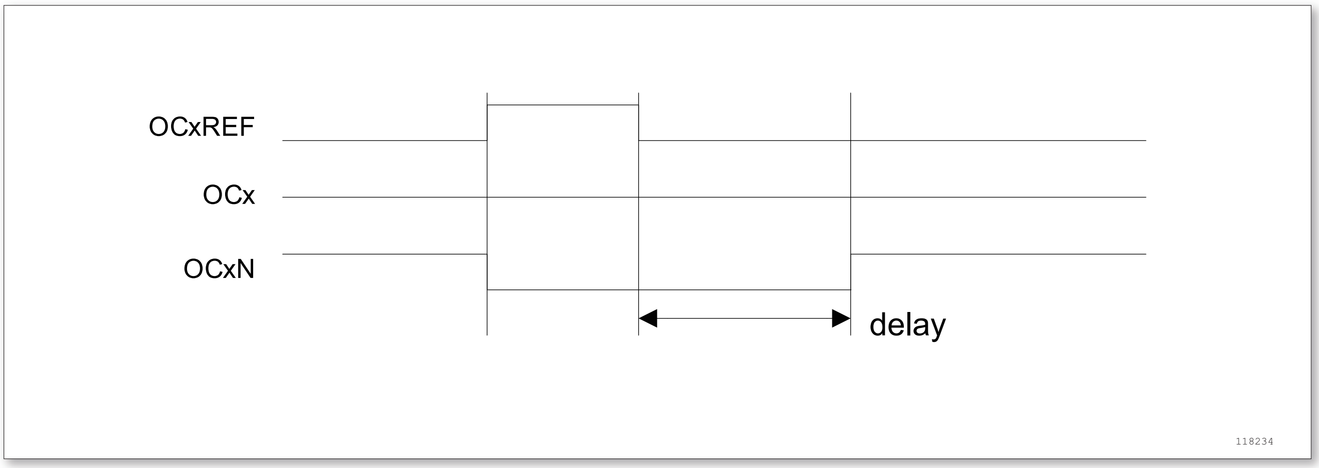


Figure 68. Dead-time Waveforms with Delay Greater than the Positive Pulse

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 section 11.4.18 for delay calculation.

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.

11.3.12 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 40 Output Control Bits for Complementary OCx and OCxN Channels with Break Feature for more details.

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.

When exiting from 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 programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSSI=0 then the timer releases the enable output otherwise the enable output remains high.
- In case of complementary output:
  - The outputs are first put in reset state i.e. 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, the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 CK_TIM clock cycles).
  - If OSSI = 0 then the timer releases the enable outputs otherwise the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set. A DMA request can be sent if the BDE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Otherwise, MOE remains low until it is written to ‘1’ again. In this case, it can be used for security and the break input can be connected to an alarm from power drivers, thermal sensors or any security components.

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

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

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 Section 11.4.8. The LOCK bits can be written only once after an MCU reset.

The following figure shows an example of behavior of the outputs in response to a break:

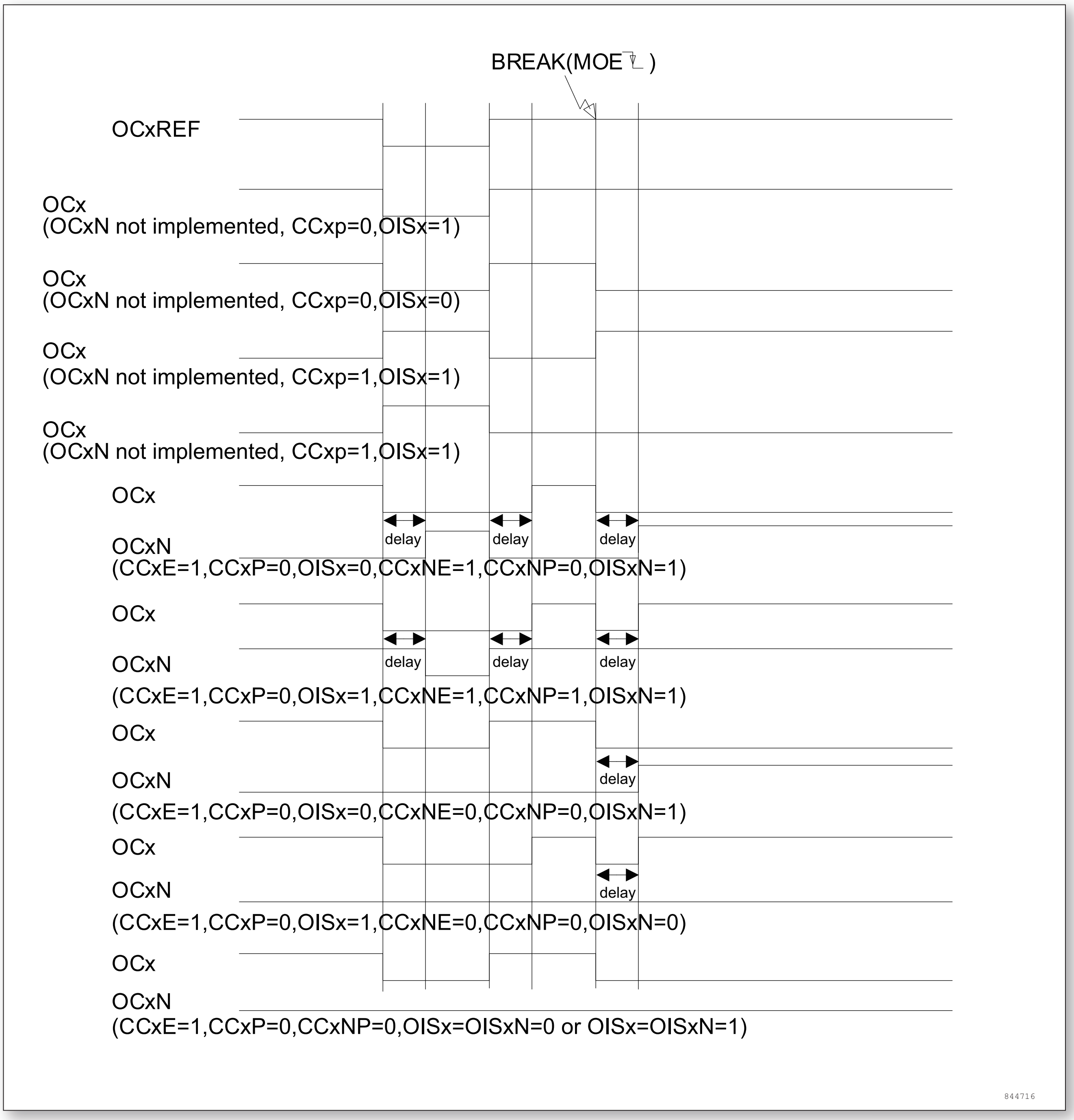

BREAK(MOE)
OCxREF
OCx
(OCxN not implemented, CCxp=0,OISx=1)
OCx
(OCxN not implemented, CCxp=0,OISx=0)
OCx
(OCxN not implemented, CCxp=1,OISx=1)
OCx
(OCxN not implemented, CCxp=1,OISx=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=0,CCxNE=1,CCxNP=0,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=1,CCxNE=1,CCxNP=1,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=0,CCxNE=0,CCxNP=0,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=1,CCxNE=0,CCxNP=0,OISxN=0)
OCx
OCxN
(CCxE=1,CCxP=0,CCxNP=0,OISx=OISxN=0 or OISx=OISxN=1)
delay
844716

Figure 69. Output Behavior in Response to A Break

11.3.13 Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMRx register set to ‘1’). The OCxREF signal remains Low until the next update event, UEV, occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the ETR signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to ‘00’.
- The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
- The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

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

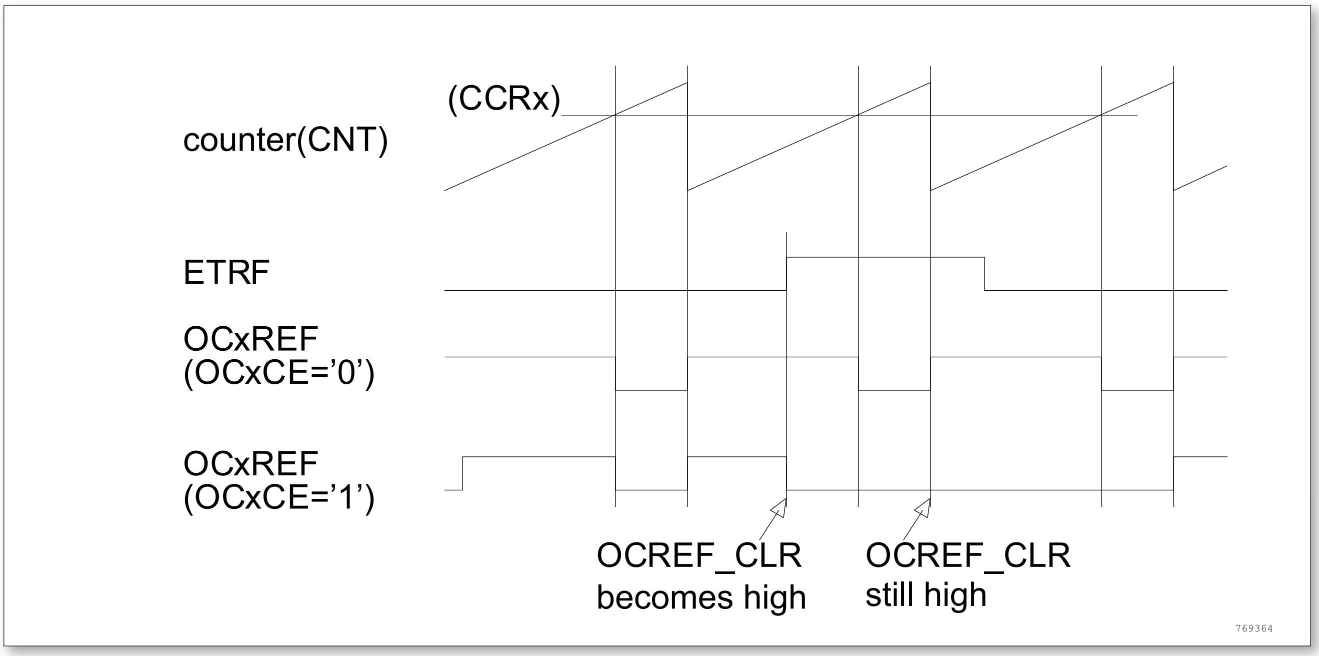


Figure 70. Clearing TIMx OCxREF

11.3.14 Six-step PWM generation

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

A flag is set when the COM event occurs (COMIF bit in the TIMx_SR register), which can generate an interrupt (if the COMIE bit is set in the TIMx_DIER register) or a DMA request (if the COMDE bit is set in the TIMx_DIER register).

The following figure describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

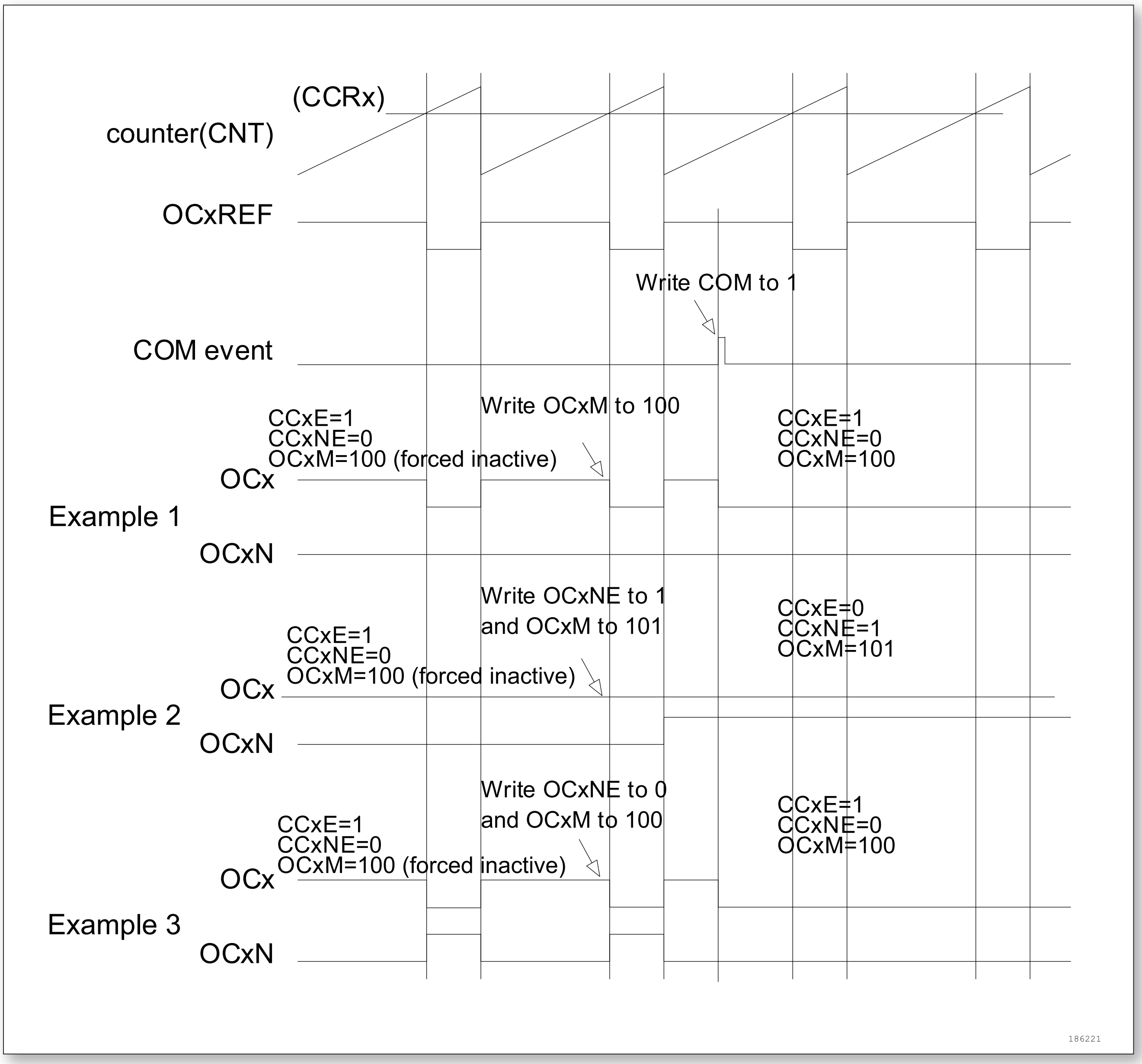


Figure 71. Six-step PWM, COM Example (OSSR = 1)

11.3.15 One-pulse mode

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

Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. 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

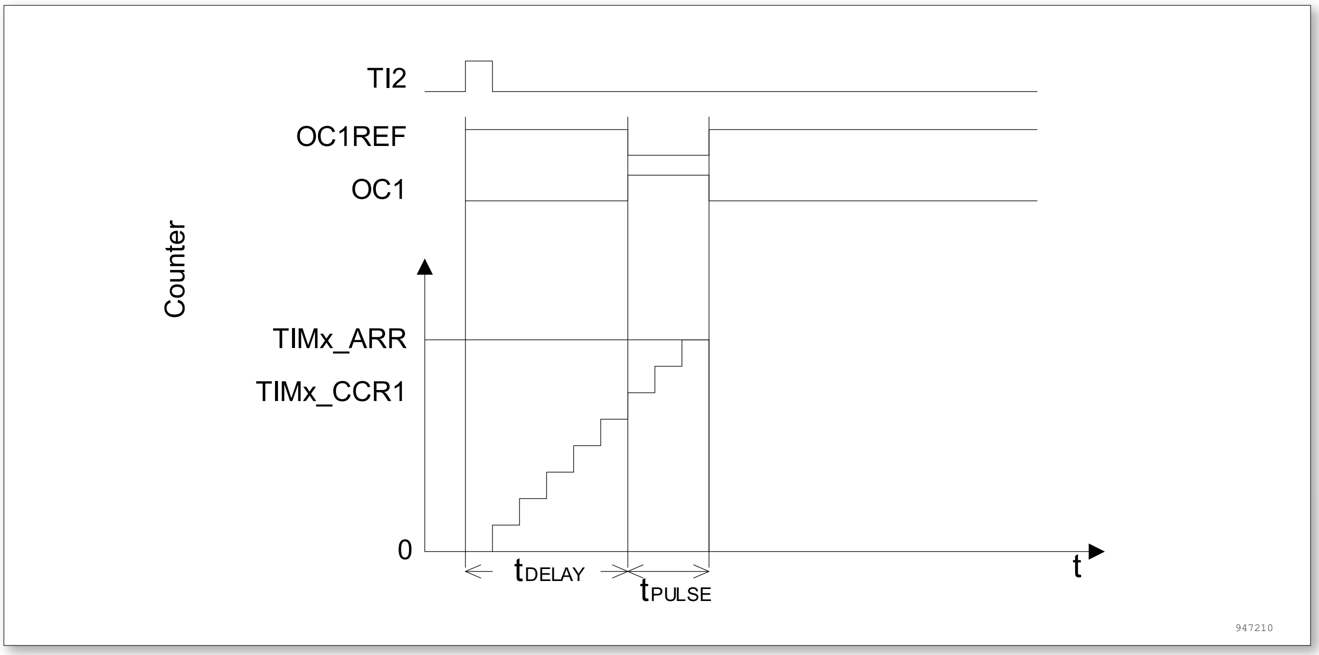


Figure 72. Example of One Pulse Mode

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

Let’s use TI2FP2 as trigger 1:

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

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

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let 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 ‘0’ 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 (Single mode), so '1' must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

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 t_{DELAY} 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.

11.3.16 Encoder interface mode

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

Select the TI1 and TI2 polarity by programming the CC1P and CC2P bits in the TIMx_CCER register. When needed, the user can program the input filter as well. The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 37. The counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted) assuming that it is enabled (CEN bit in TIMx_CR1 register written to '1'). The sequence of transitions of the two inputs is evaluated and generates count pulses as well as the direction signal. Depending on the sequence the counter counts up or down, the DIR bit in the TIMx_CR1 register is modified by hardware accordingly. The DIR bit is calculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1 only, TI2 only or both TI1 and TI2.

Encoder interface mode acts simply as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIMx_ARR register (0 to ARR or ARR down to 0 depending on the direction). So user must configure TIMx_ARR before starting. In the same way, the capture, compare, prescaler, repetition counter, trigger output features continue to work as normal. Encoder mode and External clock mode 2 are not compatible and must not be selected together.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction corresponds 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 37. 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 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 restrained where both edges are selected. This might occur if the sensor is positioned near to one of the switching points.

For this example we assume that the configuration is the following:

- CC1S=’01’(TIMx_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S=’01’(TIMx_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P=’0’, (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1=TI1).
- C2P= ’0’ (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2=TI2).
- SMS= ’011’ (TIMx_SMCR register, all inputs are active on both rising and falling edges).
- CEN= ’1’ (TIMx_CR1 register, Counter enabled).

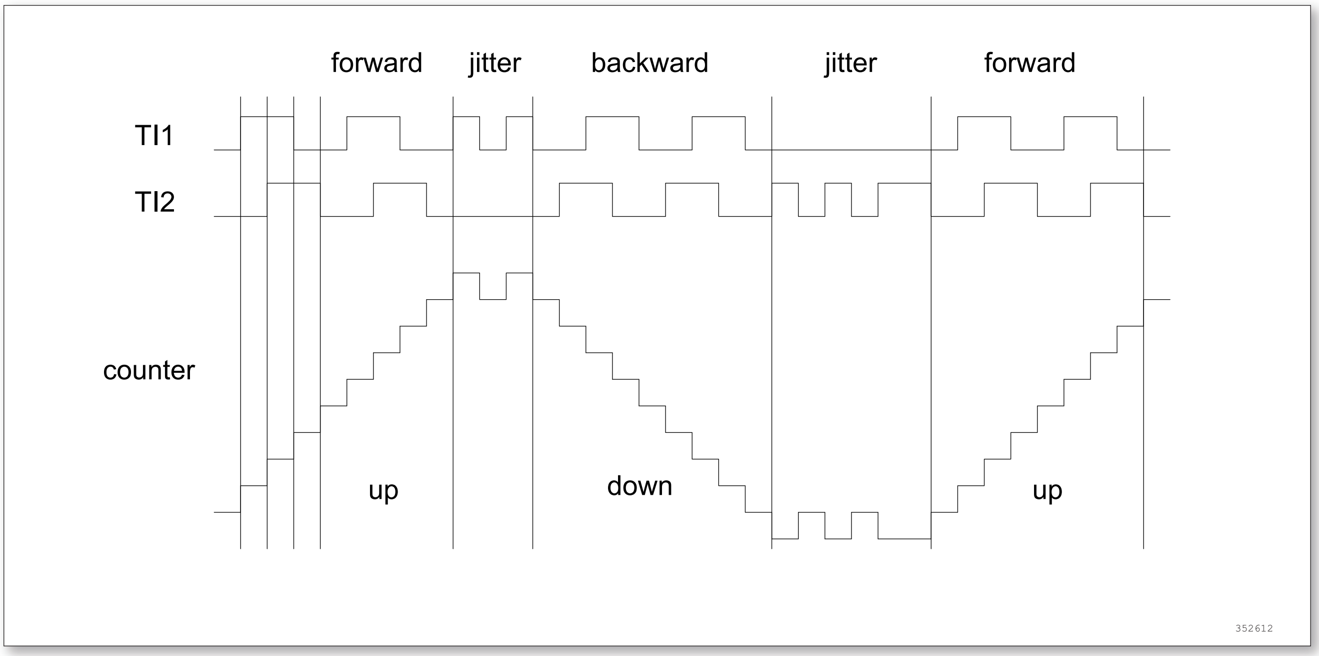


Figure 73. Example of Counter Operation in Encoder Mode

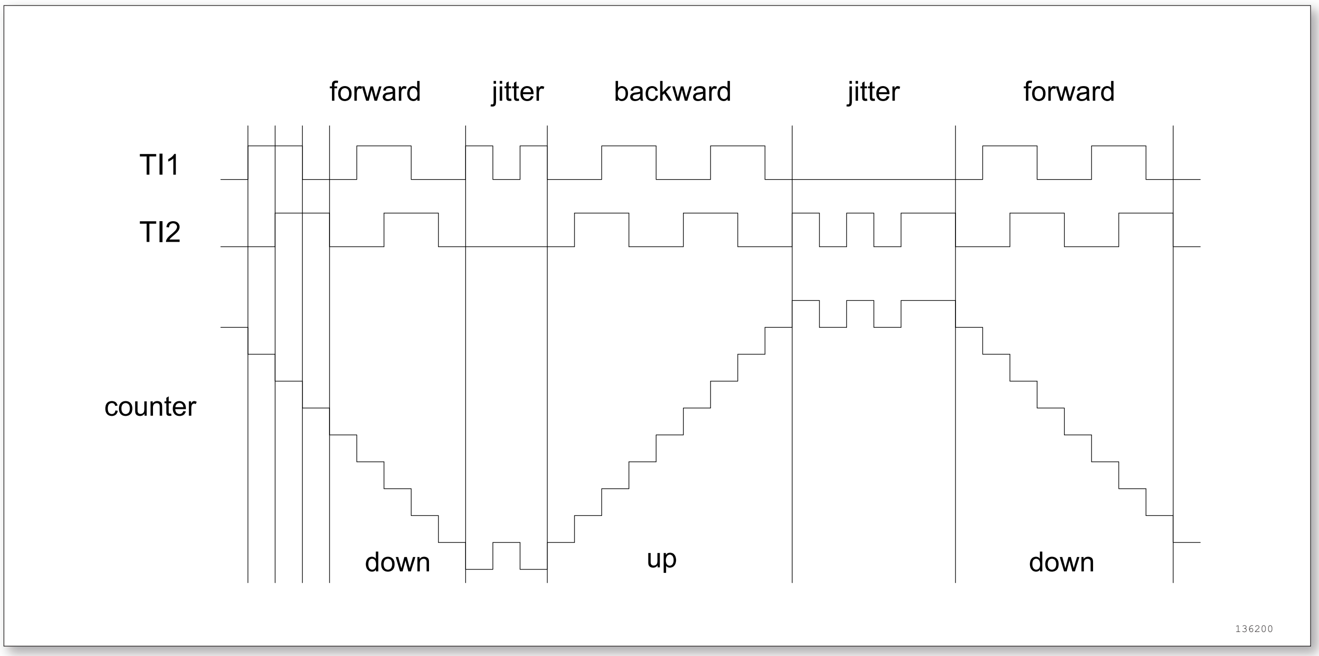


Figure 74. Example of Encoder Interface Mode with Inverted IC1FP1

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). When available, it is also possible to read its value through a DMA request generated by a real-time clock.

11.3.17 Timer input XOR function

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

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in section 11.3.18.

11.3.18 Interfacing with Hall sensors

This is done using the advanced-control timers (TIM1) to generate PWM signals to drive the motor and another timer TIMx (TIM2 or TIM3) referred to as ”interfacing timer” in Figure 75. The ”interfacing timer” captures the 3 timer input pins (CC1, CC2, CC3) connected through an XOR to the TI1 input channel (selected by setting the TI1S bit in the TIMx_CR2 register).

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

On the “interfacing timer”, capture/compare channel 1 is configured in capture mode, capture signal is TRC (see Figure 58). The captured value, which corresponds to the time elapsed between 2 changes on the inputs, gives information about motor speed.

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

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

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

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

The following figure describes this example.

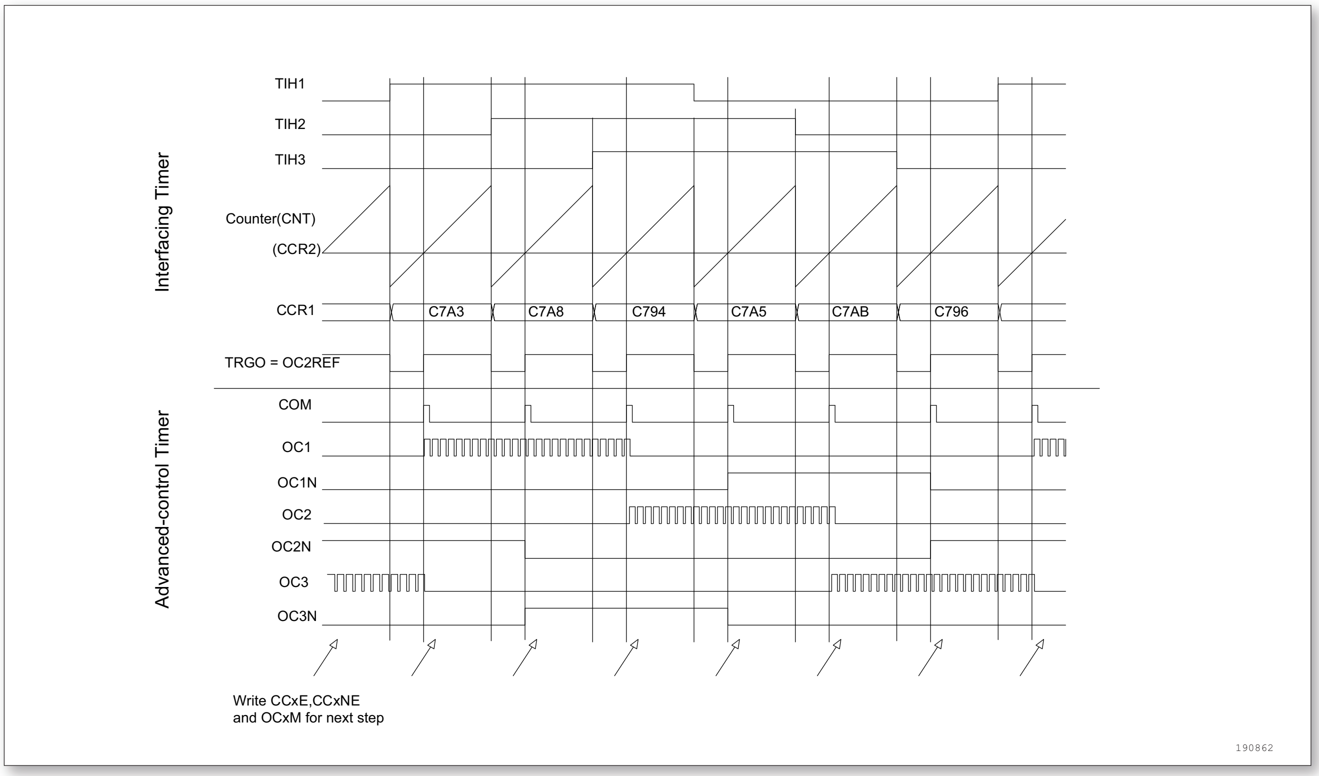


Figure 75. Example of Hall Sensor Interface

11.3.19 TIMx and external trigger synchronization

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

Slave mode: Reset mode

The counter and its prescaler can be reinitialized in response to an event on a trigger input. Moreover, if the 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, CC1S = 01 in the TIMx_CCMR1 register. Write CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Start the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request, or a DMA request can be sent if enabled (depending on the TIE (interrupt enable) and TDE (DMA enable) bits in TIMx_DIER register).

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

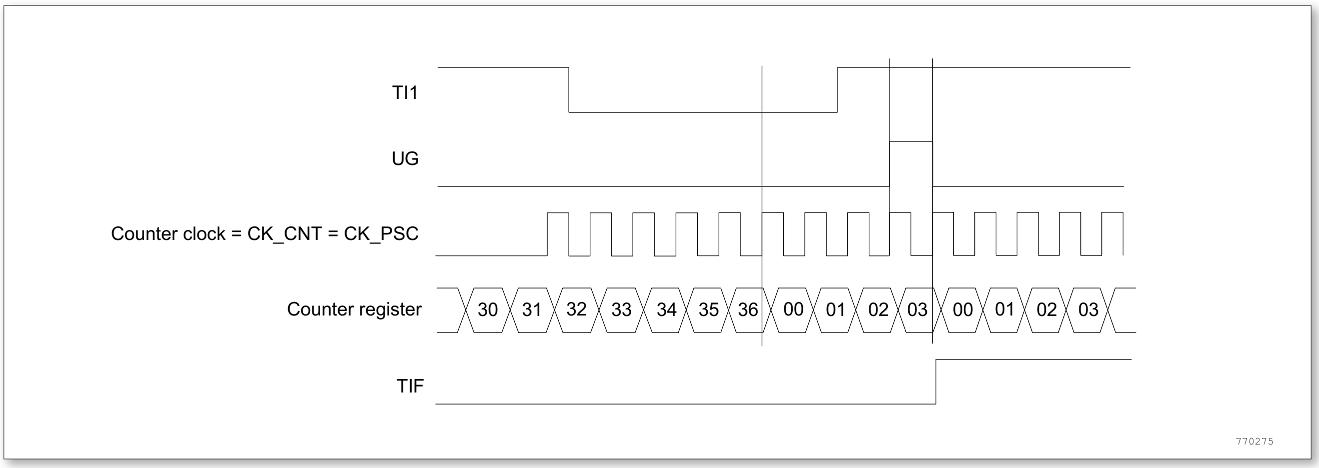


Figure 76. Control Circuit in Reset Mode

Slave mode: Gated mode

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

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

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don’t need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S=01 in TIMx_CCMR1 register. Write CC1P=1 in TIMx_CCER register to validate the polarity (and detect low level only)
- Configure the timer in gated mode by writing SMS=101 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register (in gated mode, the counter doesn’ t start if CEN=0, whatever the trigger input level is)

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

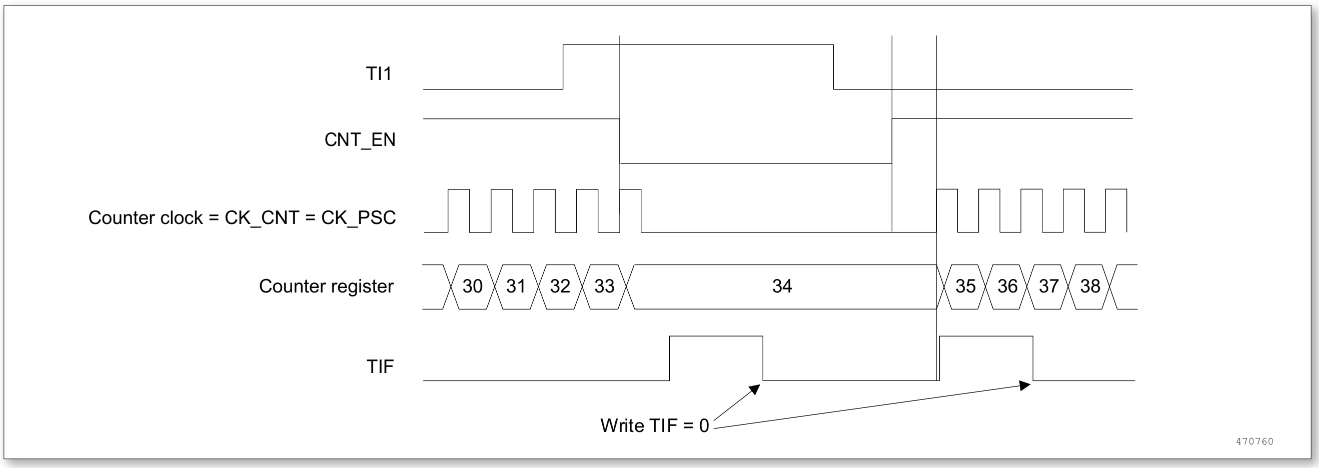


Figure 77. Control Circuit in Gated Mode

Slave mode: Trigger mode

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

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

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are only configured to select CC2P=1 in TIMx_CCER register, so as 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.

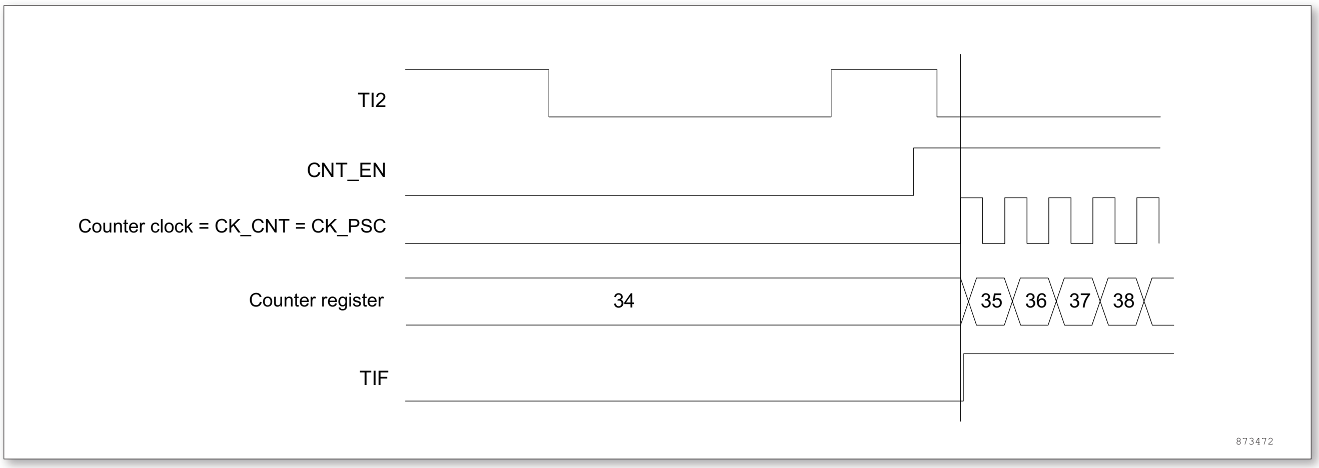


Figure 78. Control Circuit in Trigger Mode

Slave mode: external clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

- Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
  - ETF = 0000: no filter
  - ETPS = 00: prescaler disabled
  - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.
- Configure the channel 1 as follows, to detect rising edges on T1:
  - IC1F=0000: no filter.
  - The capture prescaler is not used for triggering and does not need to be configured.
  - CC1S=01 in TIMx_CCMR1 register to select only the input capture source.
  - CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edge only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

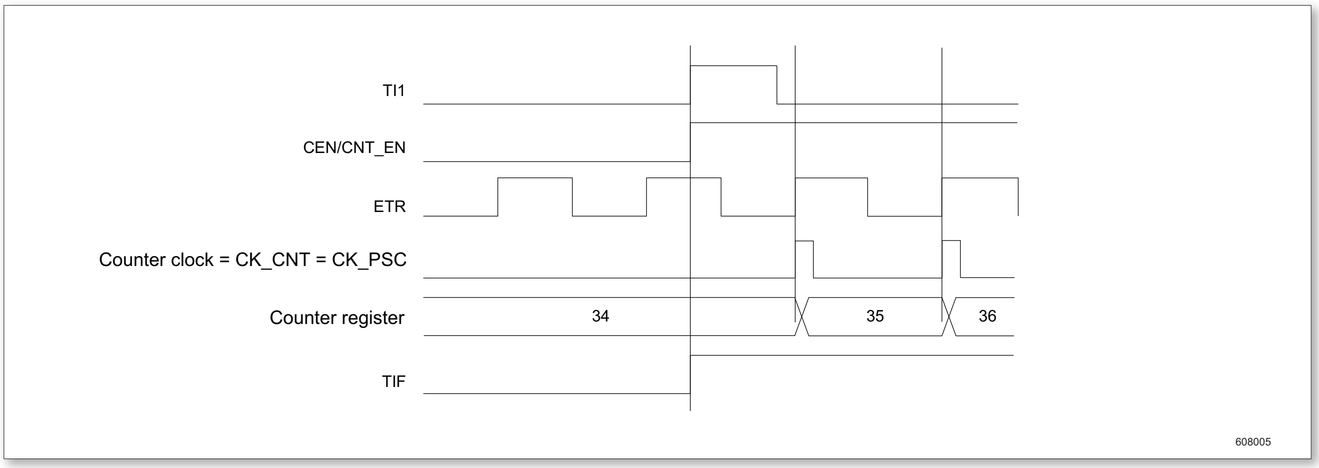


Figure 79. Control Circuit in External Clock Mode 2 + Trigger Mode

11.3.20 Timer synchronization

The TIM timers are linked together internally for timer synchronization or chaining. Refer to Section TIM2/3/4 for details.

11.3.21 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 the following debug sections.

11.4 Register description

Table 38. Summary of TIM1 Register

Offset	Acronym	Register Name	Reset	Section
0x00	TIMx_CR1	Control register 1	0x00000000	section 11.4.1
0x04	TIMx_CR2	Control register 2	0x00000000	section 11.4.2
0x08	TIMx_SMCR	Slave mode control register	0x00000000	section 11.4.3
0x0C	TIMX_DIER	DMA /interrupt enable register	0x00000000	section 11.4.4
0x10	TIMx_SR	Status register	0x00000000	section 11.4.5
0x14	TIMx_EGR	Event generation register	0x00000000	section 11.4.6
0x18	TIMx_CCMR1	Capture/compare mode register 1	0x00000000	section 11.4.7
0x1C	TIMx_CCMR2	Capture/compare mode register 2	0x00000000	section 11.4.8
0x20	TIMx_CCER	Capture/compare enable register	0x00000000	section 11.4.9
0x24	TIMx_CNT	Counter	0x00000000	section 11.4.10
0x28	TIMx_PSC	Prescaler	0x00000000	section 11.4.11
0x2C	TIMx_ARR	Auto-reload register	0x00000000	section 11.4.12
0x30	TIMx_RCR	Repetition counter register	0x00000000	section 11.4.13
0x34	TIMx_CCR1	Capture/compare register 1	0x00000000	section 11.4.14
0x38	TIMx_CCR2	Capture/compare register 2	0x00000000	section 11.4.15
0x3C	TIMx_CCR3	Capture/compare register 3	0x00000000	section 11.4.16
0x40	TIMx_CCR4	Capture/compare register 4	0x00000000	section 11.4.17
0x44	TIMx_BDTR	Break and dead-time register	0x00000000	section 11.4.18
0x48	TIMx_DCR	DMA control register	0x00000000	section 11.4.19
0x4C	TIMx_DMAR	DMA address in continuous mode	0x00000000	section 11.4.20
0x54	TIMx_CCMR3	Capture/compare mode register 3	0x00000000	section 11.4.21
0x58	TIMx_CCR5	Capture/compare register 5	0x00000000	section 11.4.22

11.4.1 Control register 1(TIMx_CR1)

Offset address: 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	rw	rw

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9: 8	CKD	rw	0x00	Clock division The 2 bits indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock (tDTS) used by the dead-time generators and the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6: 5	CMS	rw	0x00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode when the counter is enabled (CEN=1).
4	DIR	rw	0x00	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Bit	Field	Type	Reset	Description
2	URS	rw	0x00	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generates an update interrupt or DMA request if enabled.These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	rw	0x00	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller, buffered registers are then loaded with their preload values 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	rw	0x00	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.

11.4.2 Control register 2(TIMx_CR2)

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															OIS5
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS			CCDS	CCUS	Res.	CCPC
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

Bit	Field	Type	Reset	Description
31: 17	Reserved			Reserved, always read as 0.
16	OIS5	rw	0x00	Output Idle state 5 (OC5 output).Refer to OIS1 bit.
15	Reserved			Reserved, always read as 0.
14	OIS4	rw	0x00	Output Idle state 4 (OC4 output).Refer to OIS1 bit.
13	OIS3N	rw	0x00	Output Idle state 3(OC3N output). Refer to OIS1N bit.
12	OIS3	rw	0x00	Output Idle state 3 (OC3 output).Refer to OIS1 bit.
11	OIS2N	rw	0x00	Output Idle state 2(OC2N output). Refer to OIS1N bit.
10	OIS2	rw	0x00	Output Idle state 2 (OC2 output).Refer to OIS1 bit.
9	OIS1N	rw	0x00	Output Idle state 1 (OC1N output). 0: OC1N = 0 after a dead-time when MOE = 0 1: OC1N = 1 after a dead-time when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BKR register).
8	OIS1	rw	0x00	Output Idle state 1 (OC1 output). 0: OC1 = 0 (after a dead-time if OC1N is implemented) when MOE = 0 1: OC1 = 1 (after a dead-time if OC1N is implemented) when MOE = 0 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BKR register).
7	TI1S	rw	0x00	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

Bit	Field	Type	Reset	Description
6: 4	MMS	rw	0x00	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance, a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	rw	0x00	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs

Bit	Field	Type	Reset	Description
2	CCUS	rw	0x00	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit only 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when an rising edge occurs on TRGI Note: This bit acts only on channels that have a complementary output.
1	Reserved			Reserved, always read as 0.
0	CCPC	rw	0x00	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs (COMG bit set). Note: This bit acts only on channels that have a complementary output.

11.4.3 Slave mode control register(TIMx_SMCR)

Offset address: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS		ETF				MSM	TS			OCCS	SMS		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	ETP	rw	0x00	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.

Bit	Field	Type	Reset	Description
14	ECE	rw	0x00	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13: 12	ETPS	rw	0x00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8

Bit	Field	Type	Reset	Description
11: 8	ETF	rw	0x00	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS}. 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, 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
7	MSM	rw	0x00	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.

Bit	Field	Type	Reset	Description
6: 4	TS	rw	0x00	Trigger selection This bit-field selects the trigger input to be used to synchro-nize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1(ITR1) 010: Internal Trigger 2(ITR2) 011: Internal Trigger 3(ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2(TI2FP2) 111: External Trigger input (ETRF) See the following table for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	OCCS	rw	0x00	Output compare clear selection In PWM mode, clear the comparator output 1: Comparator output as clear signal 0: External trigger signal as clear signal

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x00	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 (refer to Input Control register and Control Register description). 000: Slave mode disabled - if CEN = ‘1’then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP1 edge depending on TI1FP2 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP2 edge depending on TI2FP1 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger input becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger input (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS= ’100’). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 39. TIMx Internal Trigger Connection

Slave timer	ITR0	ITR1	ITR2	ITR3
TIM1	x	TIM2	TIM3	x
TIM2	TIM1	x	TIM3	x
TIM3	TIM1	TIM2	x	x

11.4.4 DMA/interrupt enable register (TIMX_DIER)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CC5 DE	CC5 IE
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	COM DE	CC4 DE	CC3 DE	CC2 DE	CC1 DE	UDE	BIE	TIE	COM IE	CC4 IE	CC3 IE	CC2 IE	CC1 IE	UIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:18	Reserved			Reserved, always read as 0.
17	CC5DE	rw	0x00	Capture/Compare 5 DMA request enable 0: CC5 DMA request disabled 1: CC5 DMA request enabled
16	CC5IE	rw	0x00	Capture/Compare 5 interrupt enable 0: CC5 interrupt disabled 1: CC5 interrupt enabled
15	Reserved			Reserved, always read as 0.
14	TDE	rw	0x00	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	COMDE	rw	0x00	COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled
12	CC4DE	rw	0x00	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	rw	0x00	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	rw	0x00	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	rw	0x00	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	rw	0x00	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	rw	0x00	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	TIE	rw	0x00	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled

Bit	Field	Type	Reset	Description
5	COMIE	rw	0x00	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	rw	0x00	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	rw	0x00	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	rw	0x00	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

11.4.5 Status register(TIMx_SR)

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CC5 IF
															rc_w0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4 OF	CC3 OF	CC2 OF	CC1 OF	Res.	BIF	TIF	COM IF	CC4 IF	CC3 IF	CC2 IF	CC1 IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit	Field	Type	Reset	Description
31: 17	Reserved			Reserved, always read as 0.
16	CC5IF	rc_w0	0x00	Capture/Compare 5 interrupt flag Refer to CC1IF description.
15: 13	Reserved			Reserved, always read as 0.
12	CC4OF	rc_w0	0x00	Capture/Compare 4 overcapture flag Refer to CC1OF description.
11	CC3OF	rc_w0	0x00	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	rc_w0	0x00	Capture/Compare 2 overcapture flag Refer to CC1OF description.

Bit	Field	Type	Reset	Description
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	Reserved			Reserved, always read as 0.
7	BIF	rc_w0	0x00	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	COMIF	rc_w0	0x00	COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 0: No COM event occurred. 1: COM interrupt pending.
4	CC4IF	rc_w0	0x00	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	rc_w0	0x00	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	rc_w0	0x00	Capture/Compare 2 interrupt flag Refer to CC1IF description.

Bit	Field	Type	Reset	Description
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if repetition counter= 0) and if the UDIS=0 in the TIMx_CR1 register. – When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register. –When CNT is reinitialized by a trigger event (refer to the description of synchronization control register), if URS=0 and UDIS=0 in the TIMx_CR1 register.

11.4.6 Event generation register (TIMx_EGR)

Offset address: 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	Type	Reset	Description
31: 17	Reserved			Reserved, always read as 0.
16	CC5G	w	0x00	Capture/Compare 5 generation Refer to CC1G description.
15: 8	Reserved			Reserved, always read as 0.
7	BG	w	0x00	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	COMG	w	0x00	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels having a complementary output.
4	CC4G	w	0x00	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	w	0x00	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	w	0x00	Capture/Compare 2 generation Refer to CC1G description.

Bit	Field	Type	Reset	Description
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1 If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler factor is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), otherwise, it takes the auto-reload value (TIMx_ARR) if DIR=1 (downcounting).

11.4.7 Capture/compare mode register 1 (TIMx_CCMR1)

Offset address: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M			OC2PE	OC2FE	CC2S		OC1CE	OC1M			OC1PE	OC1FE	CC1S	
IC2F				IC2PSC				IC1F				IC1PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x00	Output compare 2 clear enable
14: 12	OC2M	rw	0x00	Output compare 2 mode
11	OC2PE	rw	0x00	Output compare 2 preload enable
10	OC2FE	rw	0x00	Output compare 4 fast enable
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7	OC1CE	rw	0x00	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF Input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input

Bit	Field	Type	Reset	Description
6: 4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT<TIMx_CCR1 otherwise inactive. In downcounting, channel 1 is inactive (OC1REF=‘0’) as long as TIMx_CNT>TIMx_CCR1 otherwise active (OC1REF=’1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1otherwise active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 otherwise inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=’00’(the channel is configured in output). Note 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=’00’(the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC2F	rw	0x00	Input capture 2 filter
11: 10	IC2PSC	rw	0x00	Input capture 2 prescaler
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7: 4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 6 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 8 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 5 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 6 0100: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 6 1100: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 8 0101: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 8 1101: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 5 0110: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 6 1110: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 6 0111: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 8 1111: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 8

Bit	Field	Type	Reset	Description
3: 2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the factor of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=’0’(TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input. 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	rw	0x00	Capture/compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

11.4.8 Capture/compare mode register 2(TIMx_CCMR2)

Offset address: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M			OC4PE	OC4FE	CC4S		OC3CE	OC3M			OC3PE	OC3FE	CC3S	
IC4F				IC4PSC				IC3F				IC3PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x00	Output compare 4 clear enable
14: 12	OC4M	rw	0x00	Output compare 4 mode
11	OC4PE	rw	0x00	Output compare 4 preload enable
10	OC4FE	rw	0x00	Output compare 4 fast enable

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER)
7	OC3CE	rw	0x00	Output compare 3 clear enable
6: 4	OC3M	rw	0x00	Output compare 3 mode
3	OC3PE	rw	0x00	Output compare 3 preload enable
2	OC3FE	rw	0x00	Output compare 3 fast enable
1: 0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER)

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC4F	rw	0x00	Input capture 4 filter
11: 10	IC4PSC	rw	0x00	Input capture 4 prescaler

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER).
7: 4	IC3F	rw	0x00	Input capture 3 filter
3: 2	IC3PSC	rw	0x00	Input capture 3 prescaler
1: 0	CC3S	rw	0x00	Capture/compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI4 10: CC3 channel is configured as input, IC3 is mapped on TI3 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER).

11.4.9 Capture/compare enable register(TIMx_CCER)

Offset address: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CC5P	CC5E
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CC4P	CC4E	CC3 NP	CC3 NE	CC3P	CC3E	CC2 NP	CC2 NE	CC2P	CC2E	CC1 NP	CC1 NE	CC1P	CC1E
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 18	Reserved			Reserved, always read as 0.
17	CC5P	rw	0x00	Capture/Compare 5 output polarity Refer to CC1P description.
16	CC5E	rw	0x00	Capture/Compare 5 output enable Refer to CC1E description.
15: 14	Reserved			Reserved, always read as 0.
13	CC4P	rw	0x00	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	rw	0x00	Capture/Compare 4 output enable Refer to CC1E description.
11	CC3NP	rw	0x00	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
10	CC3NE	rw	0x00	Capture/Compare 3 complementary output enable Refer to CC1NE description.
9	CC3P	rw	0x00	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	rw	0x00	Capture/Compare 3 output enable Refer to CC1E description.
7	CC2NP	rw	0x00	Capture/Compare 2 complementary output polarity Refer to CC1NP description.
6	CC2NE	rw	0x00	Capture/Compare 2 complementary output enable Refer to CC1NE description.
5	CC2P	rw	0x00	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	rw	0x00	Capture/Compare 2 output enable Refer to CC1E description.
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output polarity 0: OC1N active high 1: OC1N active low Note: This bit is not modified as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=” 00” (the channel is configured in output).
2	CC1NE	rw	0x00	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.

Bit	Field	Type	Reset	Description
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel is configured as output: 0: OC1 active high 1: OC1 active low CC1 channel is configured as input: This bit selects whether IC1 or inverted IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel is configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. CC1 channel is configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table 40. Output Control Bits for Complementary OCx and OCxN Channels with Break Feature

Control bits					Output states(1)	
MOE bit	**OSSI bit**	**OSSR bit**	**CCxE bit**	**CCxNE bit**	**OCx output state**	**OCxN output state**
1	X	0	0	0	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = 0, OCxN_EN = 0
		0	0	1	Output Disabled (not driven by the timer), OCx = 0, OCx_EN = 0	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Output Disabled (not driven by the timer), OCxN = 0, OCxN_EN = 0
		0	1	1	OCREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + Polarity + dead-time, OCxN_EN = 1
		1	0	0	Output Disabled (not driven by the timer), OCx = CCxP, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = CCxNP, OCxN_EN = 0
		1	0	1	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Off-State (output enabled with in-active state), OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + Polarity + dead-time, OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer) Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN = 0; Then if the clock is present:after a dead-time OCx = OISx , OCxN = OISxN, Assuming that OISx and OISxN do not correspond to OCX and OCxN in active state.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1; Then 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 in active state.	
	1		0	1		
	1		1	0		
	1		1	1		

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

Note 1: 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.

2: In case of CCxE=0 and CCxNE=0, OCx and OCxN are in high impedance state after output is disabled.

11.4.10 Counter(TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CNT	rw	0x0000	Counter value

11.4.11 Prescaler(TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} /(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in "reset mode")

11.4.12 Auto-reload register(TIMx_ARR)

Offset address: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	ARR	rw	0x0000	Prescaler value ARR is the value to be loaded in the actual auto-reload register. Refer to section 11.3.1 for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

11.4.13 Repetition counter register(TIMx_RCR)

Offset address: 0x30

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7: 0	REP	rw	0x00	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodically transfers from preload to active registers) when preload registers are enabled, as well as the update interrupt generation rate, if this interrupt is enabled. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP + 1) corresponds to: - the number of PWM periods in edge-aligned mode - number of half PWM period in center-aligned mode

11.4.14 Capture/compare register 1(TIMx_CCR1)

Offset address: 0x34

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR1	rw	0x0000	Capture/Compare 1 value If CC1 channel is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Otherwise the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

11.4.15 Capture/compare register2(TIMx_CCR2)

Offset address: 0x38

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR2	rw	0x0000	Capture/Compare 2 value If CC2 channel is configured as output: CCR2 contains the value to be loaded in the actual capture/compare 2 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Otherwise the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If CC2 channel is configured as input: CCR2 contains the counter value transferred by the last input capture 2 event (IC2).

11.4.16 Capture/compare register 3(TIMx_CCR3)

Offset address: 0x3C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR3	rw	0x0000	Capture/Compare 3 value If CC3 channel is configured as output: CCR3 contains the value to be loaded in the actual capture/compare 3 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Otherwise the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. If CC3 channel is configured as input: CCR3 contains the counter value transferred by the last input capture 3 event (IC3).

11.4.17 Capture/compare register 4(TIMx_CCR4)

Offset address: 0x40

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR4	rw	0x0000	Capture/Compare 4 value If CC4 channel is configured as output: CCR4 contains the value to be loaded in the actual capture/compare 4 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Otherwise the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If CC4 channel is configured as input: CCR4 contains the counter value transferred by the last input capture 4 event (IC4).

11.4.18 Break and dead-time register(TIMx_BDTR)

Offset address: 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	rw	rw	rw	rw	rw	rw	rw	rw

Note: As the bits AOE, BKP, BKE, OSSI, OSSR and DTG can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIMx_BDTR register.

Bit	Field	Type	Reset	Description
31:17	Reserved			Reserved, always read as 0.
16	DOE	rw	0x00	Direct output enable When the brake is valid and the MOE is set to zero, it is valid. 1: Immediately output the idle state, no longer waiting for the dead time to output. 0: After the brake input, wait for a dead time and output the idle state.

Bit	Field	Type	Reset	Description
15	MOE	rw	0x00	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is cleared by software or set automatically, depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIMx_CCER register). See OC/OCN enable description for more details (section 11.4.9: capture/compare enable register).
14	AOE	rw	0x00	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active) Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	rw	0x00	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
12	BKE	rw	0x00	Break enable 0: Break inputs (BRK and BRK_ACTH) disabled 1: Break inputs (BRK and BRK_ACTH) enabled Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
11	OSSR	rw	0x00	Off-state selection for Run mode This bit is used when MOE=1 on channels configured as complementary outputs. OSSR is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details (section 11.4.9: capture/compare enable register (TIMx_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE=1 or CCxNE=1. Then, set OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).

Bit	Field	Type	Reset	Description
10	OSSI	rw	0x00	Off-state selection for Idle mode This bit is used when MOE=0 on channels configured as outputs. See OC/OCN enable description for more details (section 11.4.9: capture/compare enable register (TIMx_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are forced first with their idle level as soon as CCxE=1 or CCxNE=1. OC/OCN enable output signal=1 Note: This bit can not be modified as long as LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
9: 8	LOCK	rw	0x00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = DTG, BKE, BKP, AOE bits in TIMx_BDTR register, and OISx/OISxN bits in TIMx_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.

Bit	Field	Type	Reset	Description

7: 0	DTG	rw	0x00	Dead-time generator setup
				This bit-field defines the duration of the dead-time inserted between the complementary outputs. It is assumed that DT correspond to this duration.
				DTG[7: 5] = 0xx:
				DT = (DTG[7: 0] + 1) × t_{dtg}, t_{dtg} = t_{DTS};
				DTG[7: 5] = 10x:
				DT = (DTG[5: 0] + 1 + 64) × t_{dtg}, t_{dtg} = 2 × t_{DTS};
				DTG[7: 5] = 110:
				DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 8 × t_{DTS};
				DTG[7: 5] = 111:
				DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 16 × t_{DTS};
				Example: if t_{DTS} = 125ns(8MHz), dead-time possible values are:
				125ns to 15875ns by 125 nS steps;
				16µs to 31750ns by 250 nS steps;
				32µs to 63µs by 1 µs steps;
				64µs to 126µs by 2 µs steps;
				Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

11.4.19 DMA control register(TIMx_DCR)

Offset address: 0x48

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			DBL					Res.			DBA				
			w	w	w	w	w				w	w	w	w	w

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
12: 8	DBL	w	0x00	DMA burst length This bit field defines the burst transfer in the continuous mode (the timer detects a burst transfer when a write access to the TIMx_DMAR register address is performed), namely, the number of transfers, in half-word (double bytes) or bytes. 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers .... 10001: 18 transfers Example: Let us consider the following transfer: DBL = 7 and DBA = TIM2_CR1. - If DBL =7 and DBA = TIM2_CR1 represent the address of data to be transferred, the transfer address is given by: (Address of TIMx_CR1) + DBA + (DMA index), where, DMA index = DBL TIMx_CR1 address + DBA + 7 is the address of data to be written or read, so that the transfer is completed to/from 7 registers starting from the TIMx_CR1 address + DBA. According to the setting of DMA data length, the following case may occur: -If the data is set to half word (16 bits), the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, the user must specify the data width of DMA transfer for the timer.
7: 5	Reserved			Reserved, always read as 0.
4: 0	DBA	w	0x00	DMA base address These bits define the base-address for DMA transfers in the continuous mode (when write access is done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ......

11.4.20 DMA address for full transfer(TIMx_DMAR)

Offset address: 0x4C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	DMAB	w	0x0000	DMA register for burst accesses A write operation to the TIMx_DMAR register will access the register located at the following address: TIMx_CR1 address + DBA + DMA index, Where: ‘TIMx_CR1 address’ is the address of the control register 1; ‘DBA’ is the DMA base address configured in TIMx_DCR register; ‘DMA index’ is the offset automatically controlled by the DMA transfer, depending on DBL configured in TIMx_DCR.

11.4.21 Capture/compare mode register 3(TIMx_CCMR3)

Offset address: 0x54

Reset value: 0x0000

The channel can only be used in output (compare mode).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								OC5 CE	OC5M			OC5 PE	OC5 FE	Reserved	
								rw	rw			rw	rw		

Output compare mode:

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7	OC5CE	rw	0x00	Output compare 5 clear enable
6: 4	OC5M	rw	0x00	Output compare 5 mode
3	OC5PE	rw	0x00	Output compare 5 preload enable
2	OC5FE	rw	0x00	Output compare 5 fast enable
1: 0	Reserved			Reserved, always read as 0.

11.4.22 Capture/compare register 5(TIMx_CCR5)

Offset address: 0x58

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR5	rw	0x0000	Capture/Compare 5 value The CC5 channel can only be configured as an output: CCR5 is the value to be loaded in the actual capture/compare 5 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC5PE). Otherwise the preload value is copied in the active capture/compare 5 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC5 output.

12 16-bit general-purpose timers (TIMx16 Bit)

16-bit general-purpose timers (TIMx16 Bit)

12.1 TIMx introduction

General-purpose timers consist of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare 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.

TIMx are completely independent, and do not share any resources. They can be synchronized together as described in Section Timer Synchronization.

12.2 TIMx Main features

TIM3 functions include:

- 16-bit up, down, up/down auto-reload register
- 16-bit programmable prescaler allowing dividing (modifing in real time) the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- circuit to control the timer with external signals and to interconnect several timers together.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes

- Trigger input for external clock or cycle-by-cycle current management

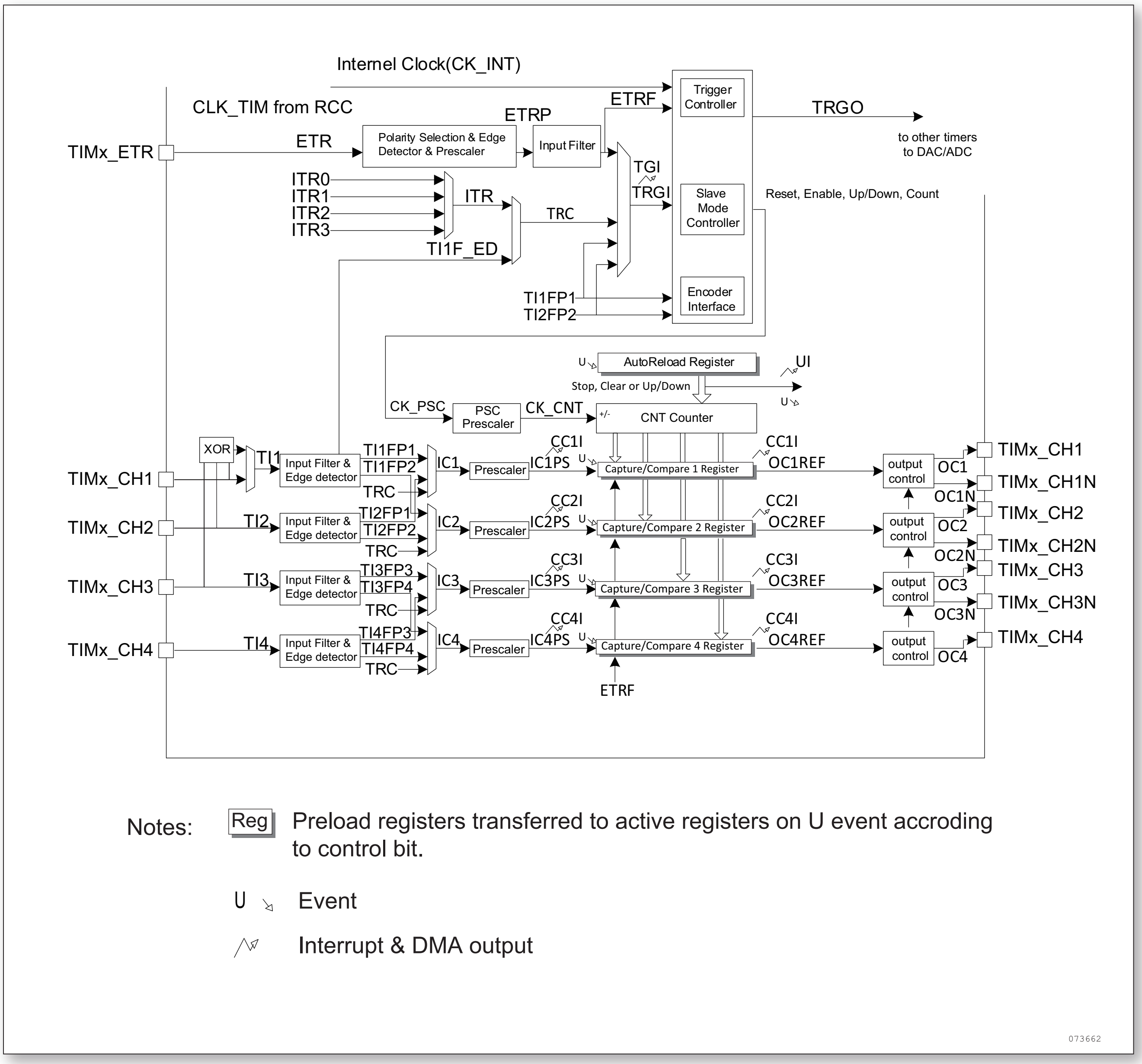


Figure 80. Block Diagram of general-purpose timer

12.3 TIMx Functional description

12.3.1 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 auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in details 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).

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 factor is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler factor is changed on the fly:

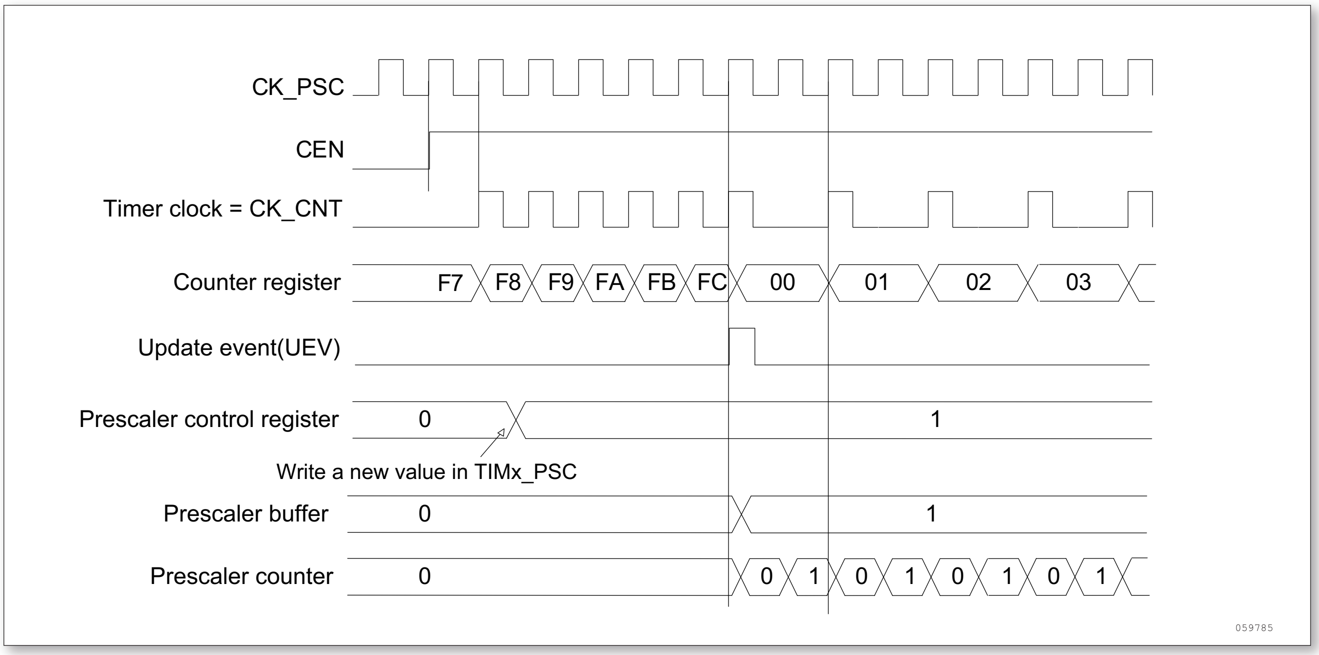


Figure 81. Counter Timing Diagram with Prescaler Division Change from 1 to 2

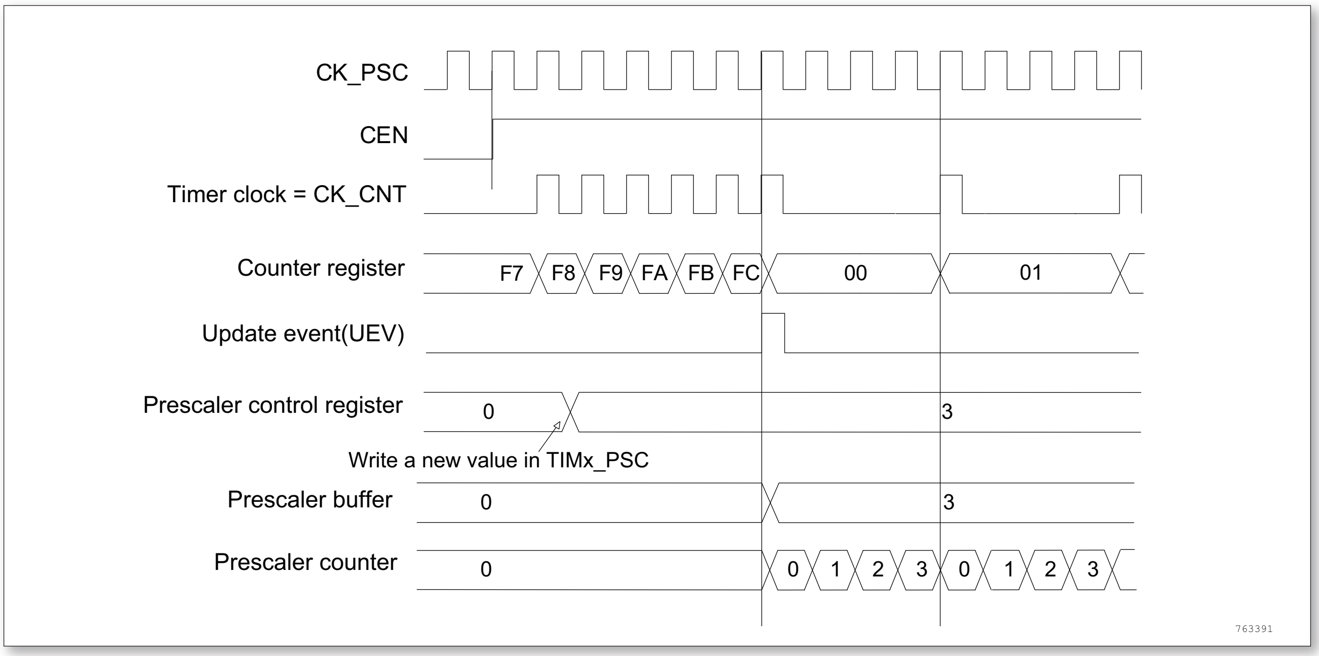


Figure 82. Counter Timing Diagram with Prescaler Division Change from 1 to 4

12.3.2 Counter modes

Upcounting mode

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

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 software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag bit(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 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.

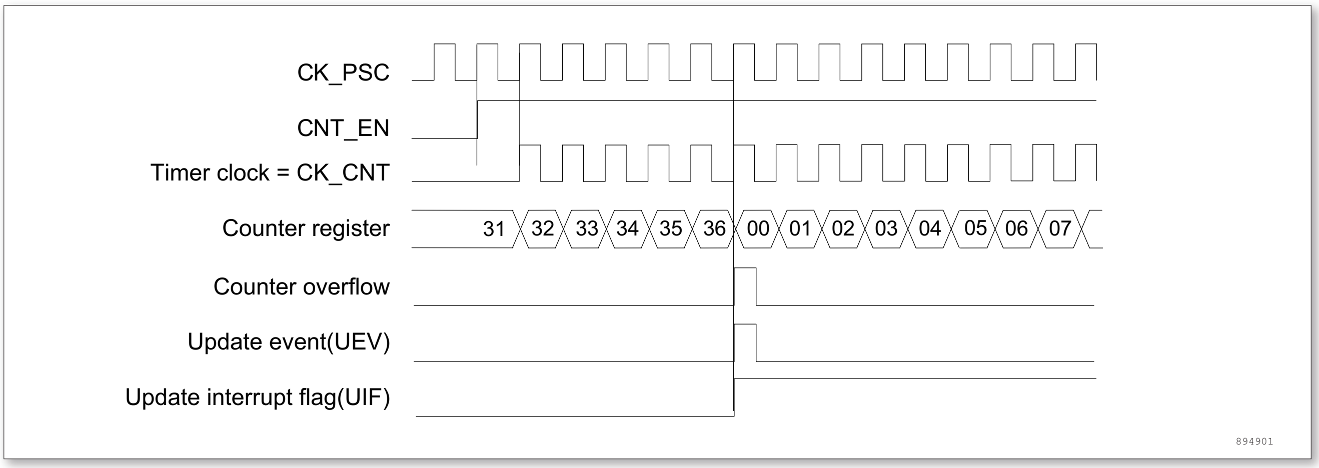


Figure 83. Counter Timing Diagram, Internal Clock Divided by 1

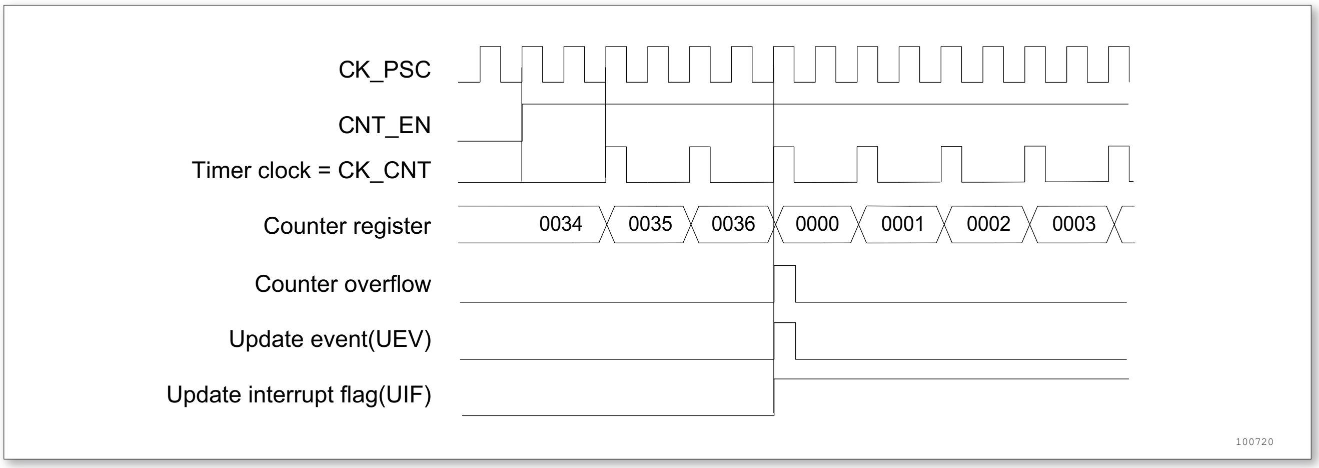


Figure 84. Counter Timing Diagram, Internal Clock Divided by 2

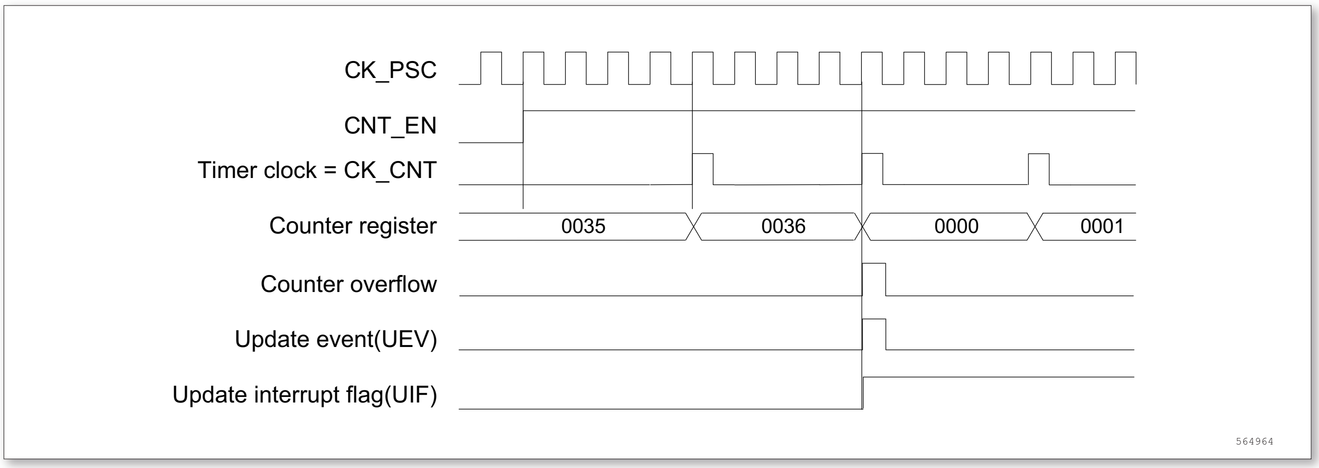


Figure 85. Counter Timing Diagram, Internal Clock Divided by 4

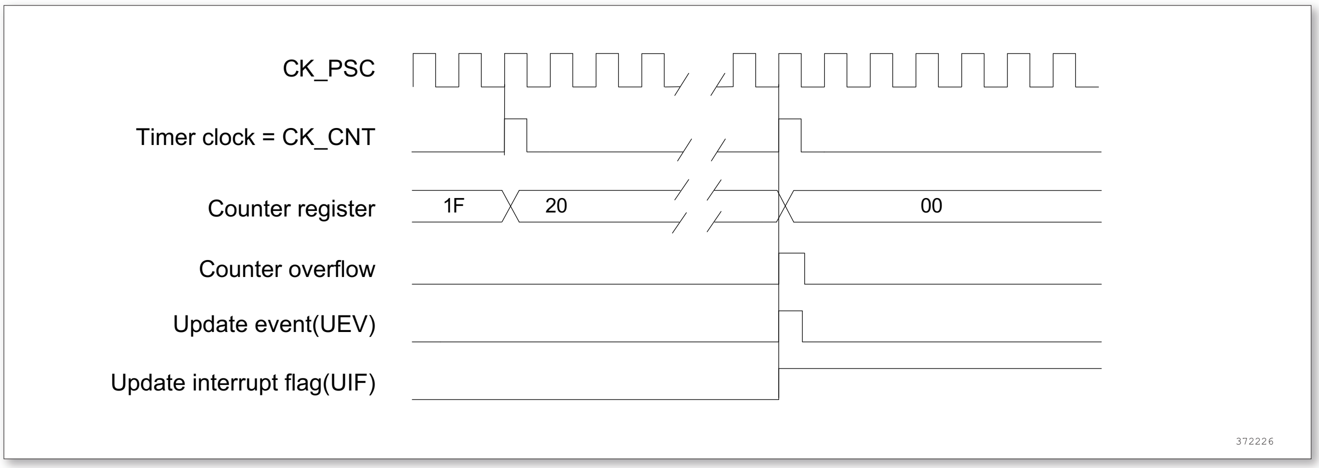


Figure 86. Counter Timing Diagram, Internal Clock Divided by N

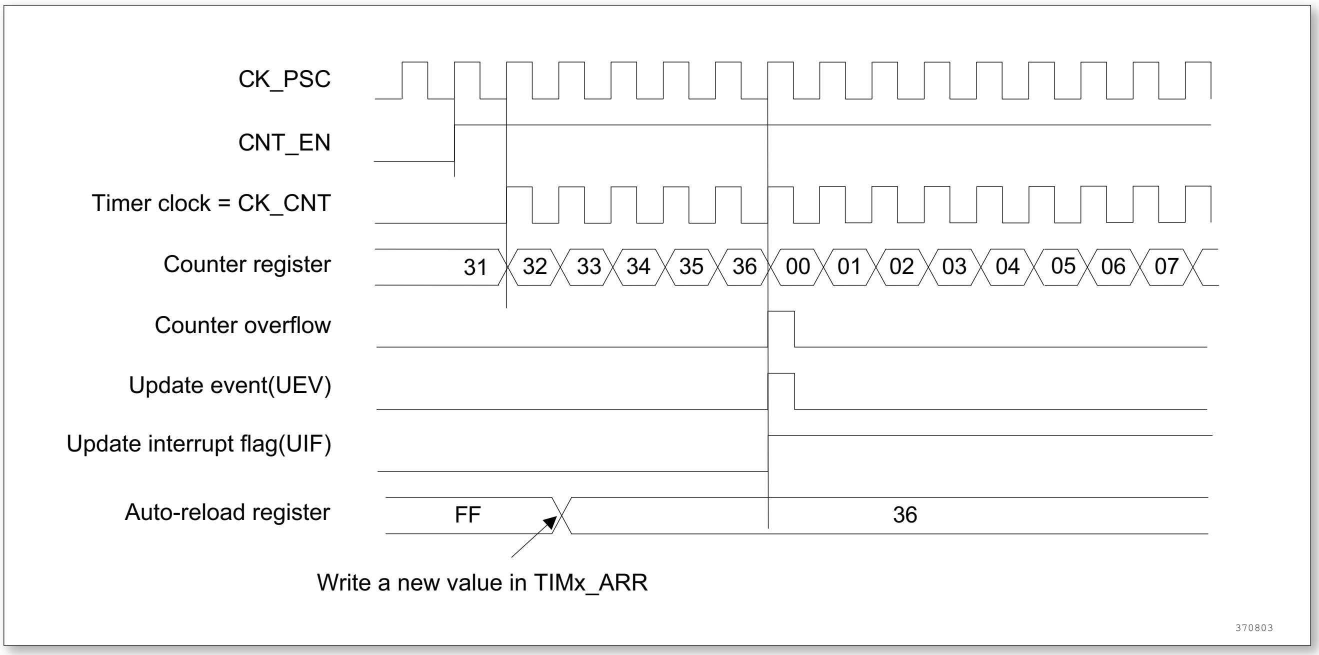


Figure 87. Counter Timing Diagram, Update Event When ARPE = 0 (TiMx_ARR Not Preloaded)

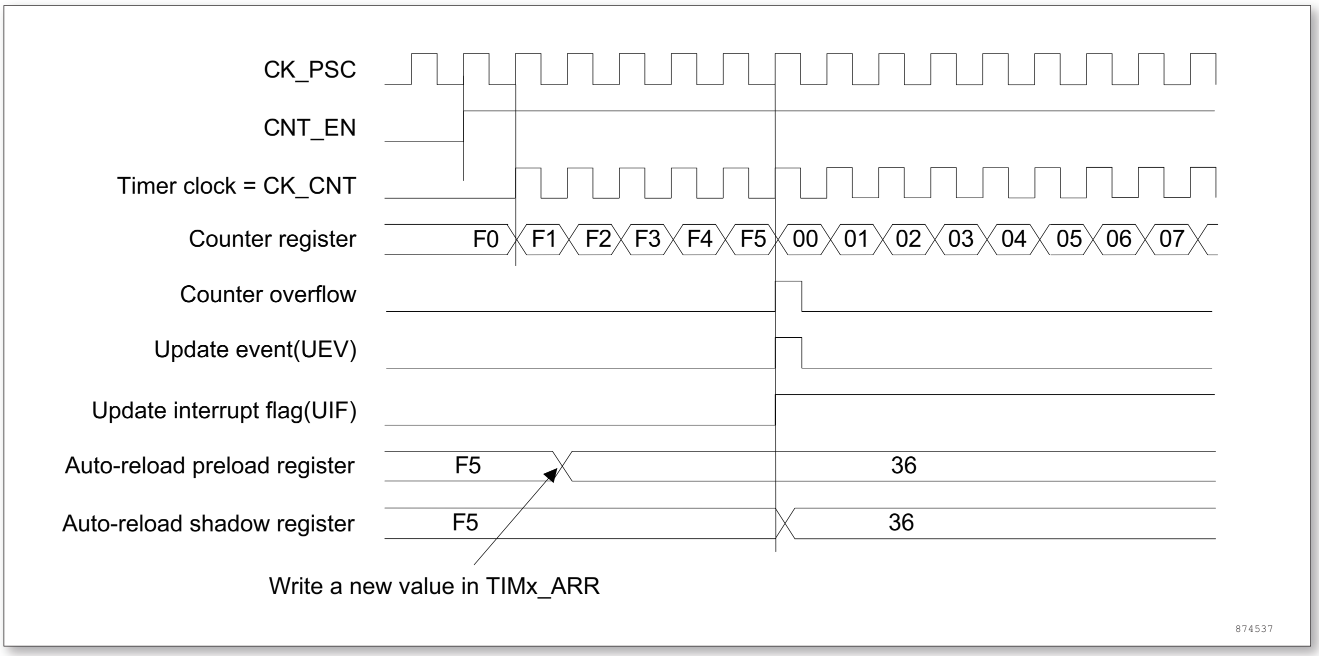

CK_PSC
CNT_EN
Timer clock = CK_CNT
Counter register
F0 F1 F2 F3 F4 F5 00 01 02 03 04 05 06 07
Counter overflow
Update event(UEV)
Update interrupt flag(UIF)
Auto-reload preload register
F5
36
Auto-reload shadow register
F5
36
Write a new value in TIMx_ARR
874537

Figure 88. Counter Timing Diagram, Update Event When ARPE = 1 (TiMx_ARR Preloaded)

Downcounting mode

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

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

- The 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.

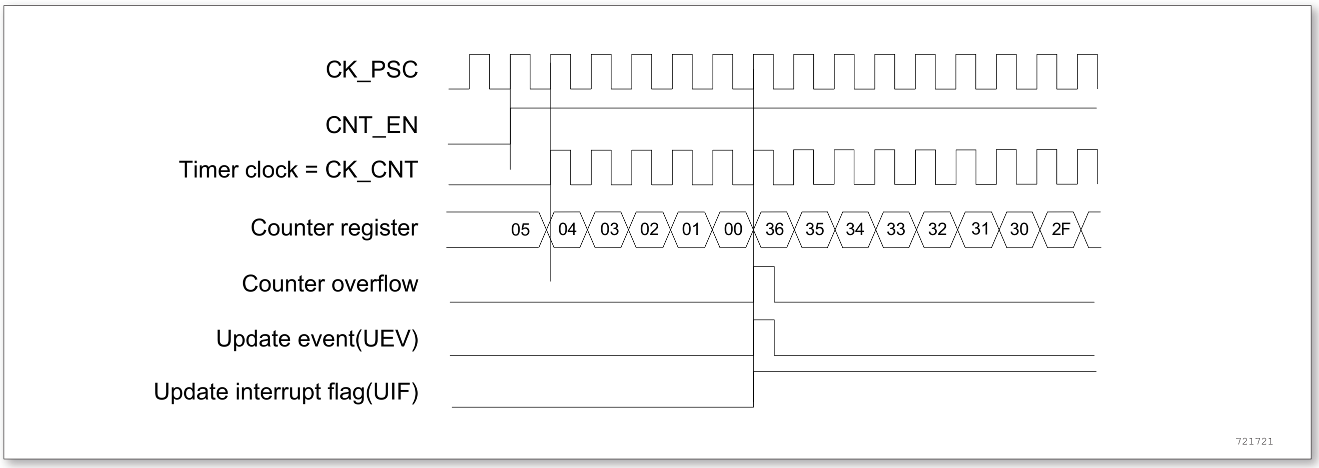


Figure 89. Counter Timing Diagram, Internal Clock Divided by 1

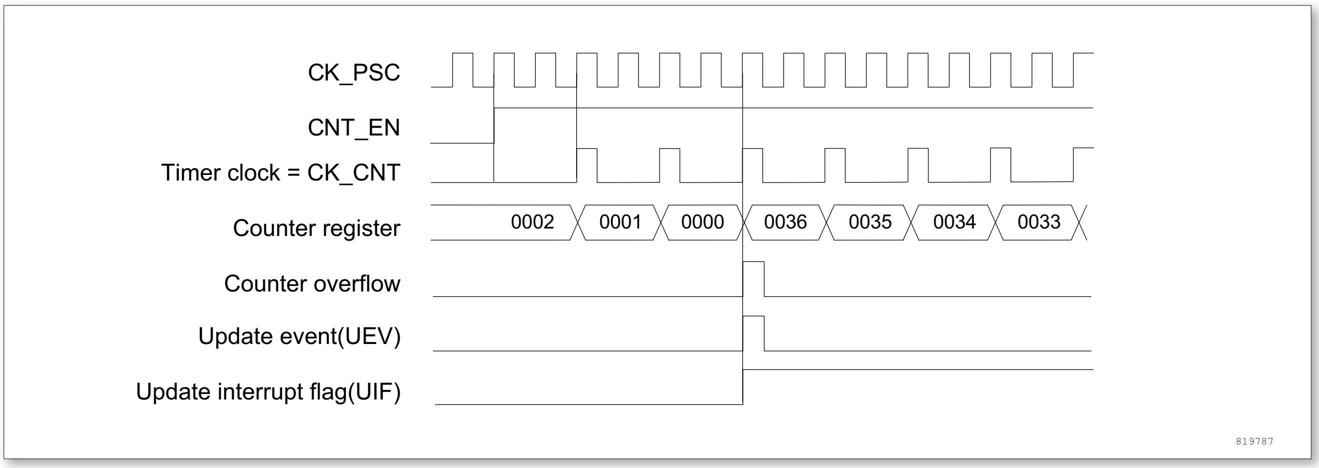


Figure 90. Counter Timing Diagram, Internal Clock Divided by 2

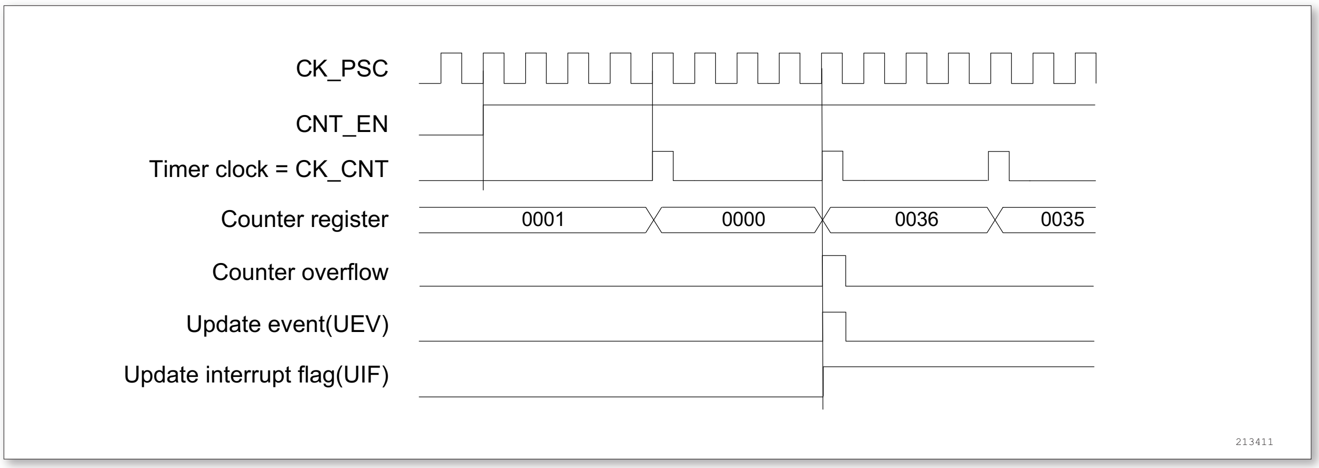


Figure 91. Counter Timing Diagram, Internal Clock Divided by 4

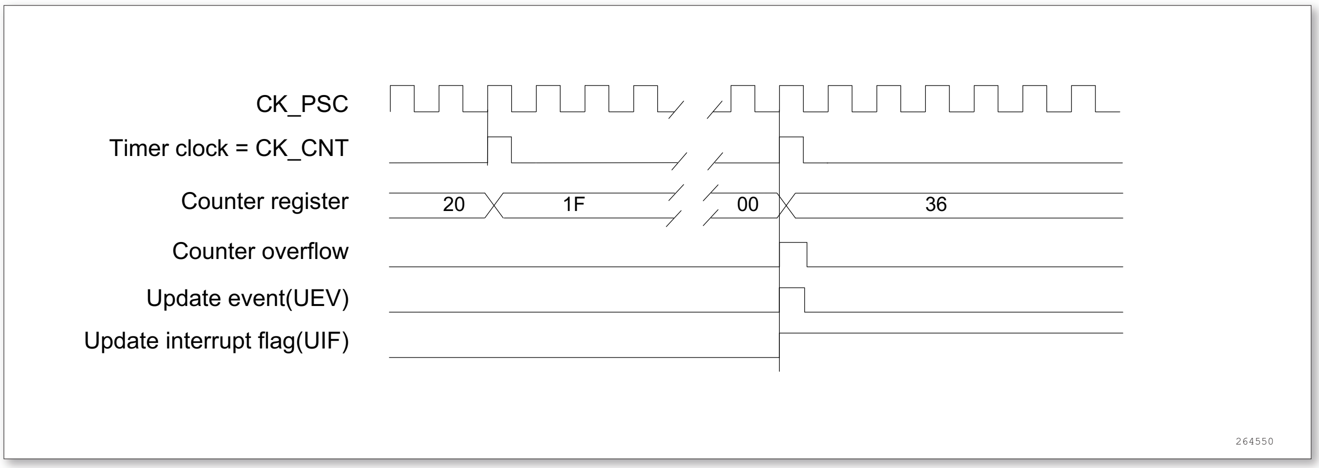


Figure 92. Counter Timing Diagram, Internal Clock Divided by N

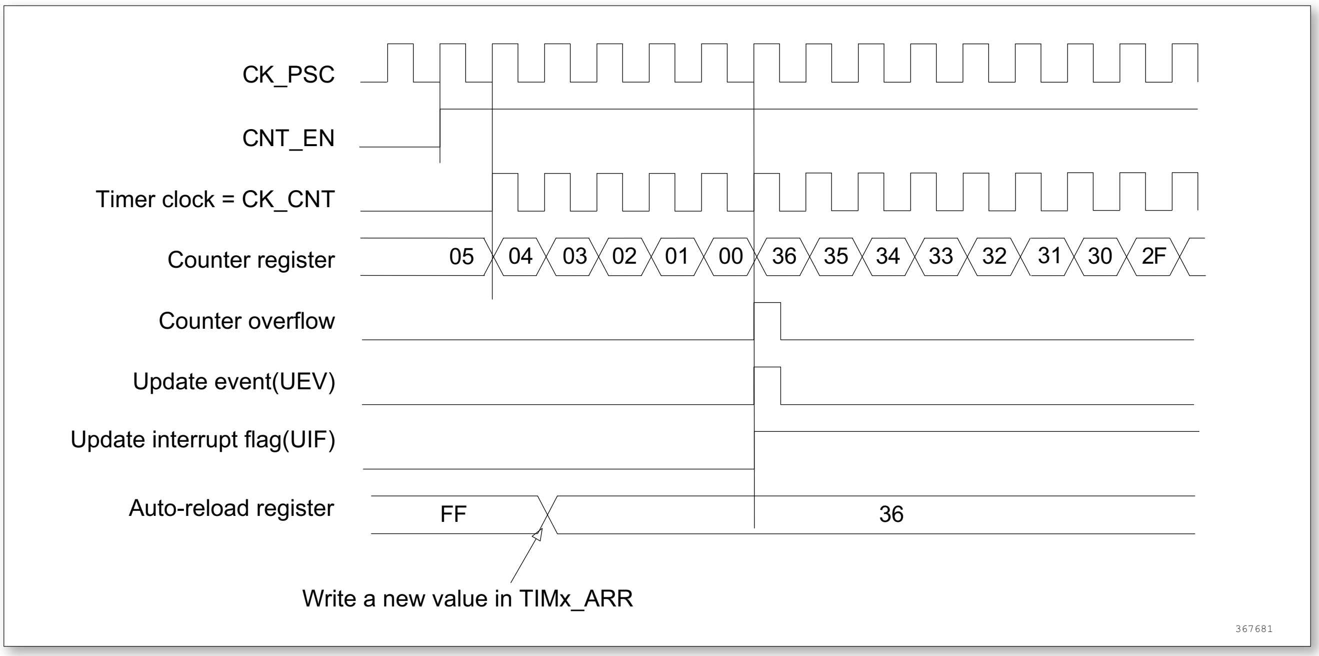


Figure 93. Counter Timing Diagram, Update Event when Repetition Counter is Not Used

Center-aligned mode (Upcounting/Downcounting))

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

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, so does the counter of the prescaler.

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag bit(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.

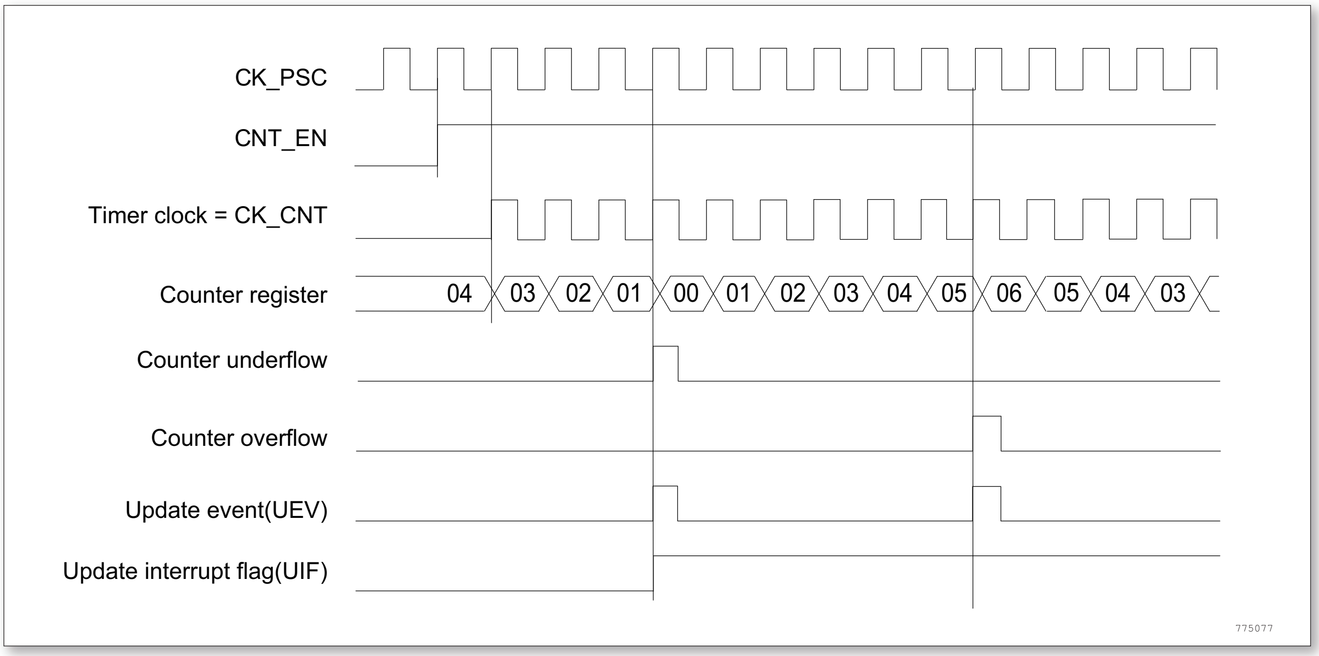


Figure 94. Counter Timing Diagram, Internal Clock Divided by 1, TIMx_ARR = 6

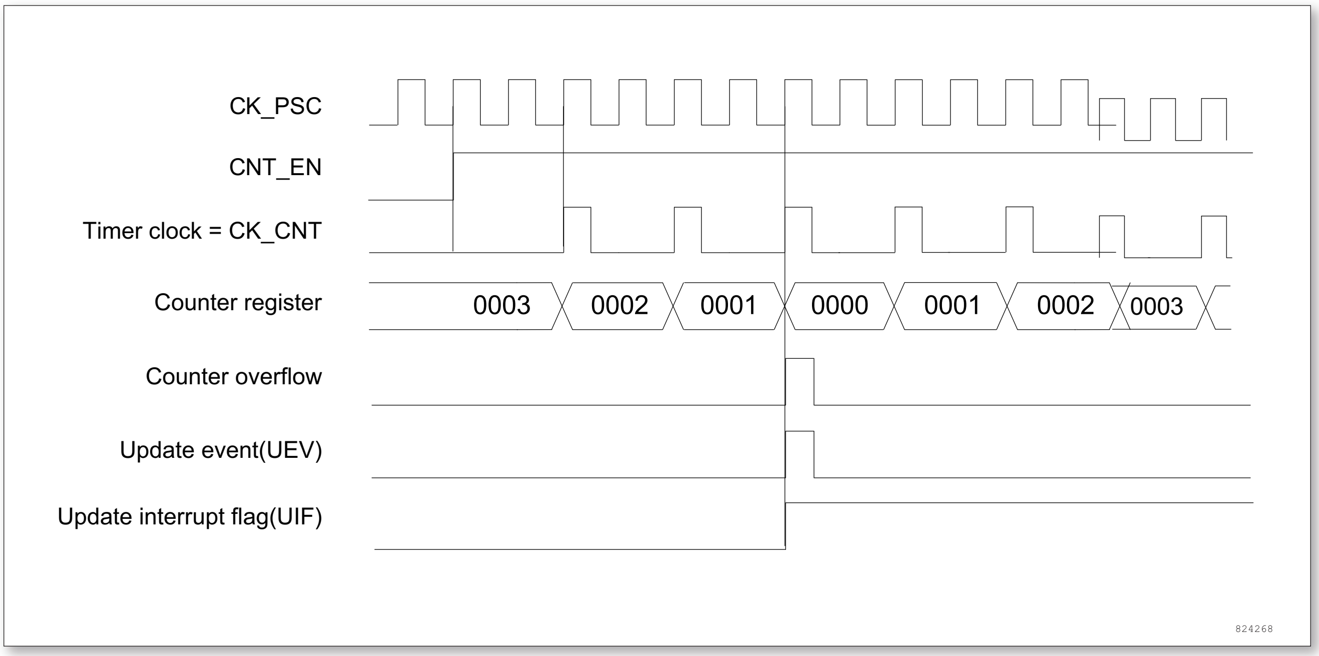


Figure 95. Counter Timing Diagram, Internal Clock Divided by 2

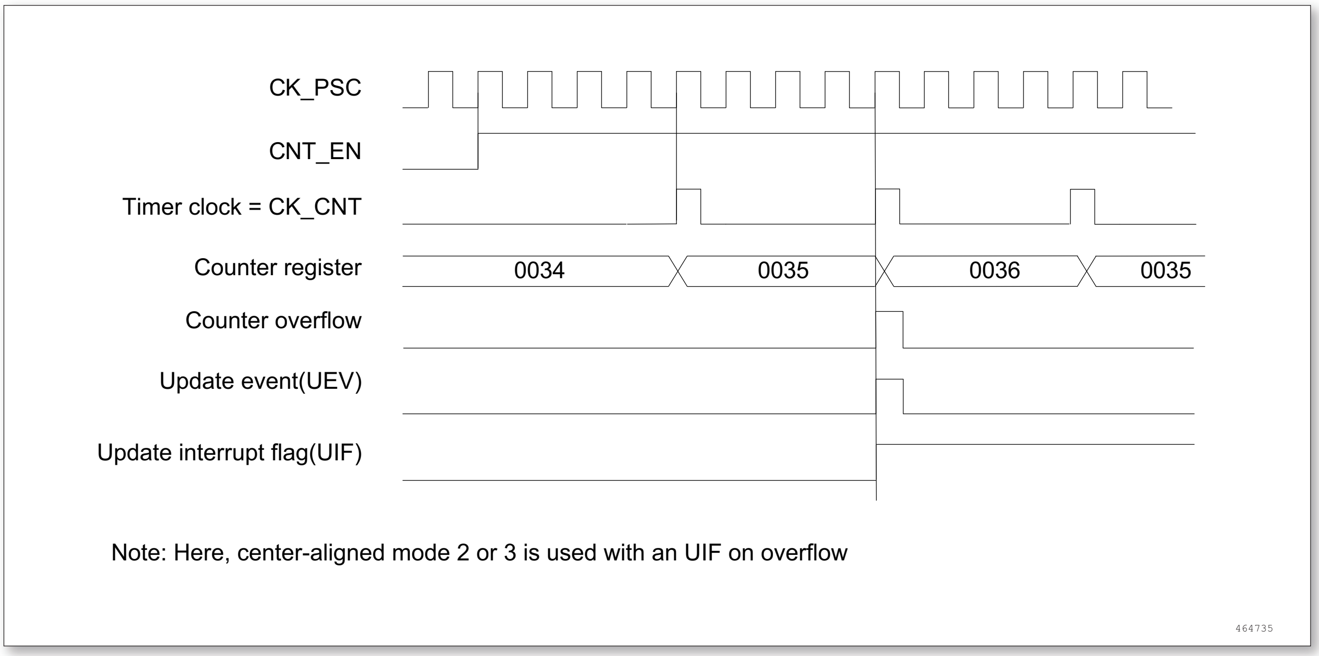


Figure 96. Counter Timing Diagram, Internal Clock Divided by 4, TIMx_ARR = 0×36

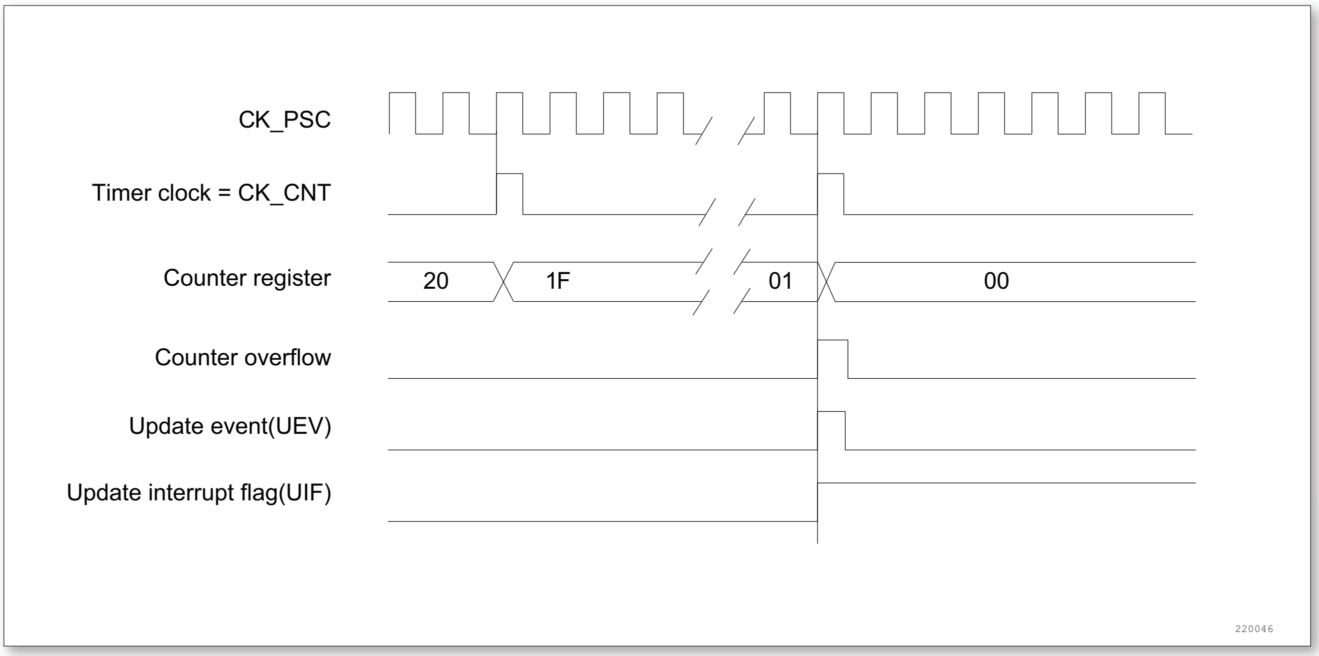


Figure 97. Counter Timing Diagram, Internal Clock Divided by N

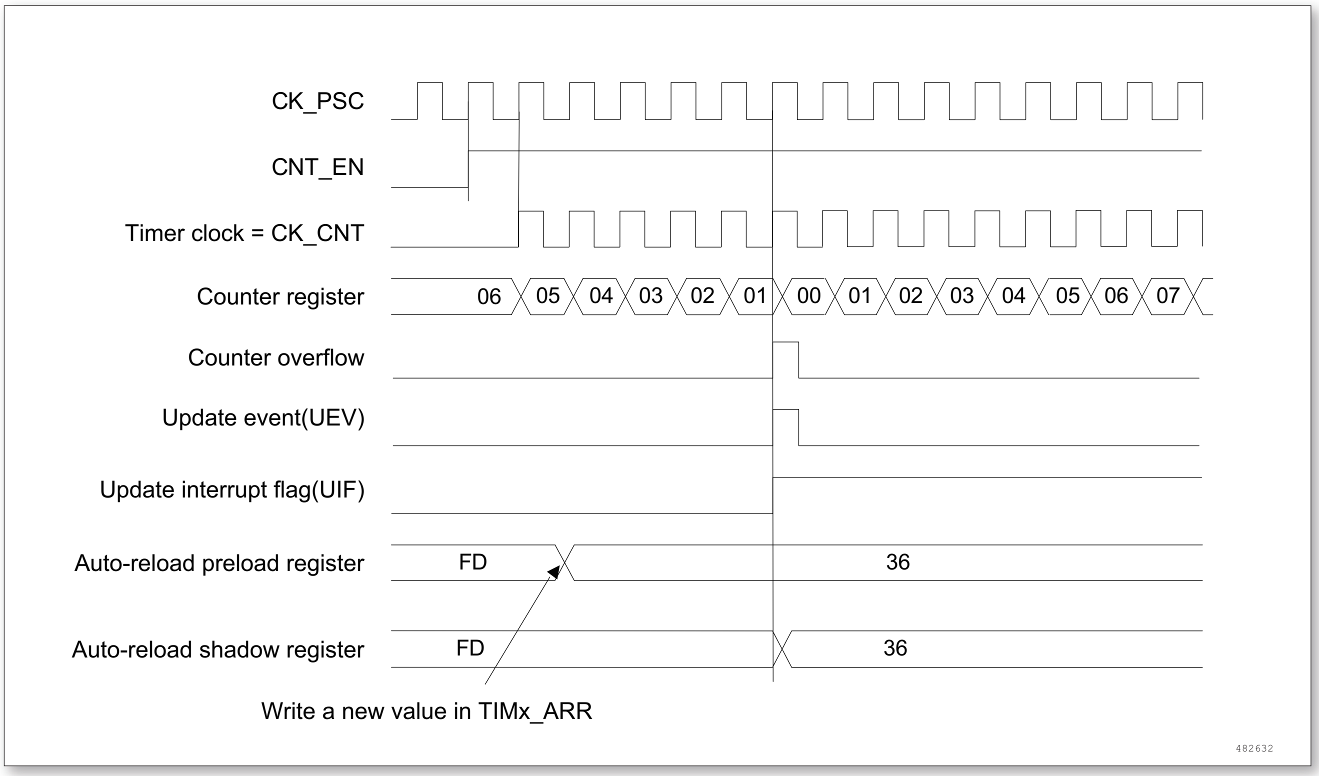


Figure 98. Counter Timing Diagram, Update Event with ARPE = 1(Counter Underflow)

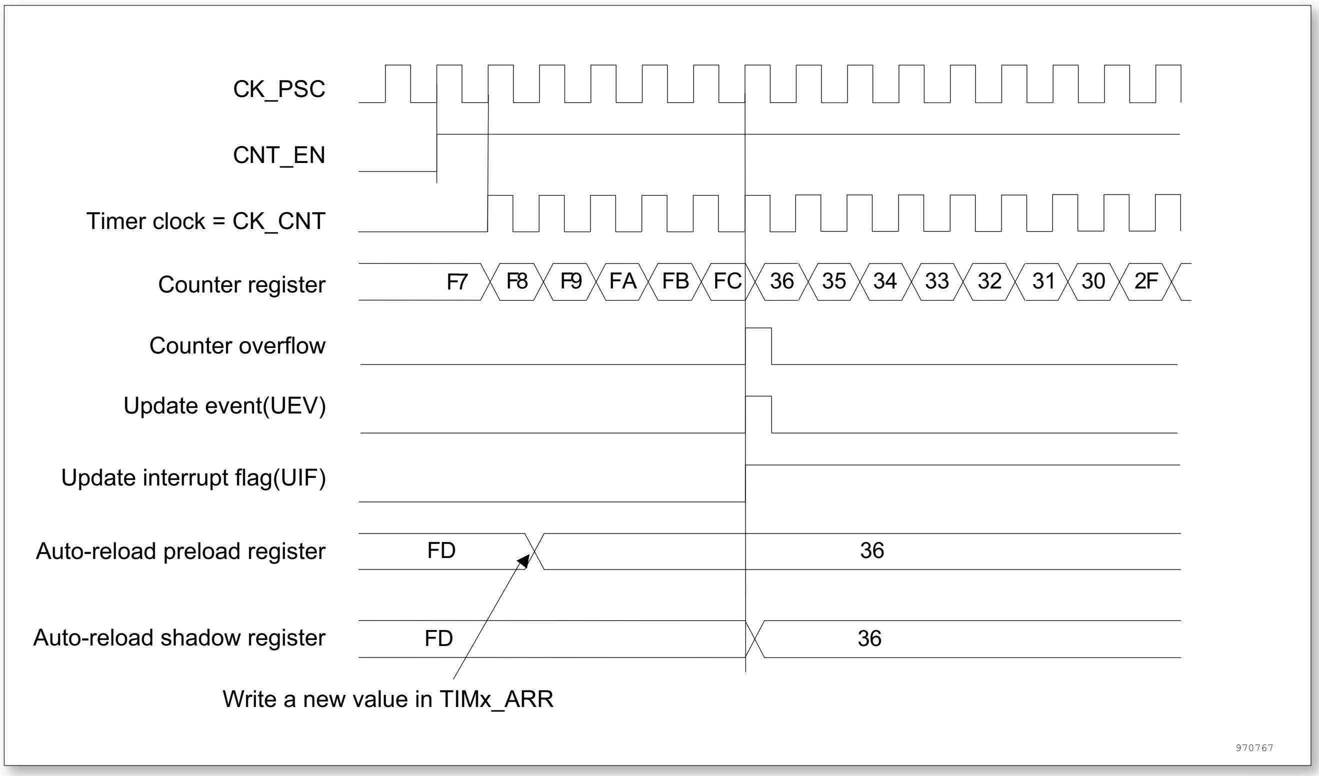


Figure 99. Counter Timing Diagram, Update Event with ARPE = 1(Counter Overflow))

12.3.3 Clock selection

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

- Internal clock (CK_INT).
- External clock mode1: external input pin (TIx).
- External clock mode2: external trigger input (ETR).
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 2.

Internal clock (CK_INT)

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

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

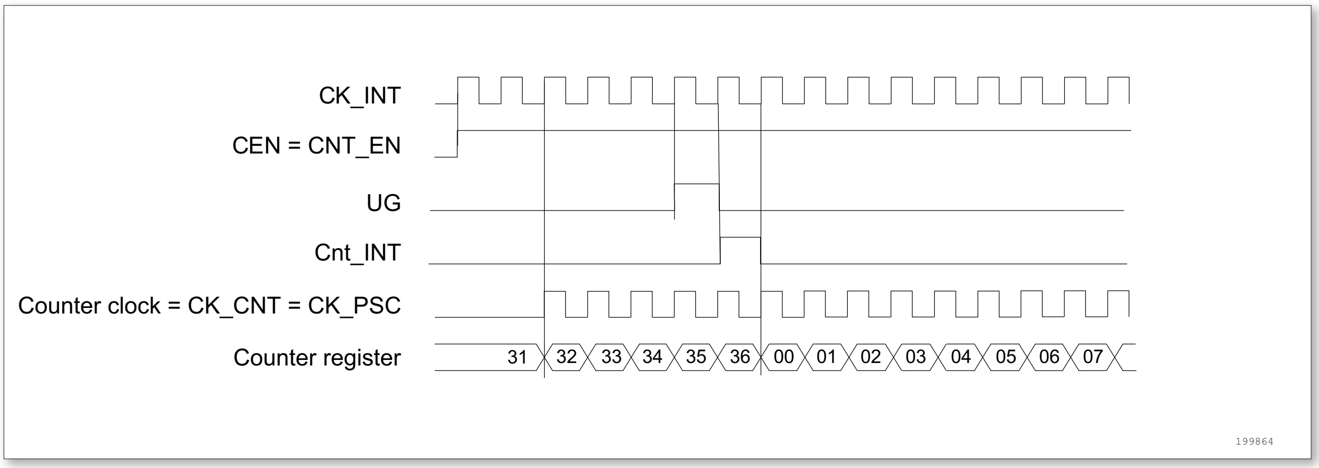


Figure 100. Control Circuit in Normal Mode, Internal Clock Divided By 1

External clock source mode 1

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

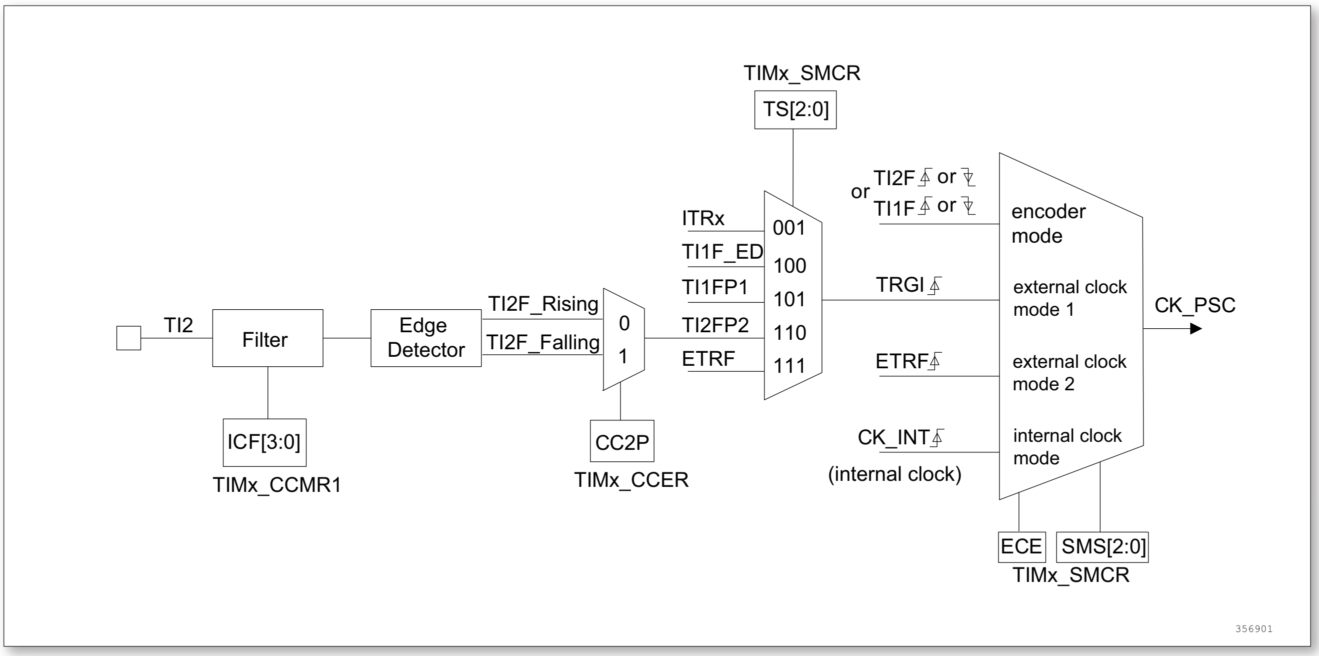


Figure 101. TI2 External Clock Connection Example

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

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = ‘01’ in the TIMx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F=0000). Note: The capture prescaler is not used for triggering, so there: Tno need to configure it.
3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.
4. Select the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=110 in the TIMx_SMCR register.
6. 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.

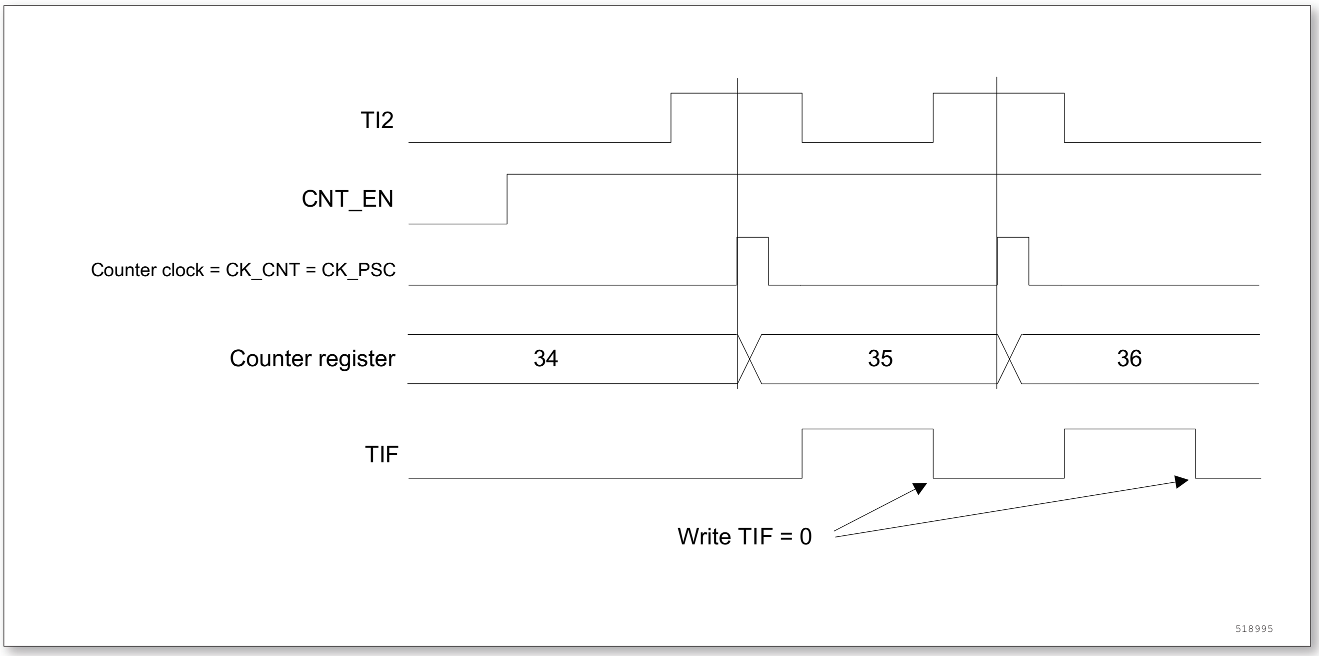


Figure 102. Control Circuit in External Clock Mode 1

External clock source mode 2

This mode is selected by writing ECE = 1 in the TIMx_SMCR register.

The counter can count at each rising or falling edge on the external trigger input ETR.

The following figure gives an overview of the external trigger input block.

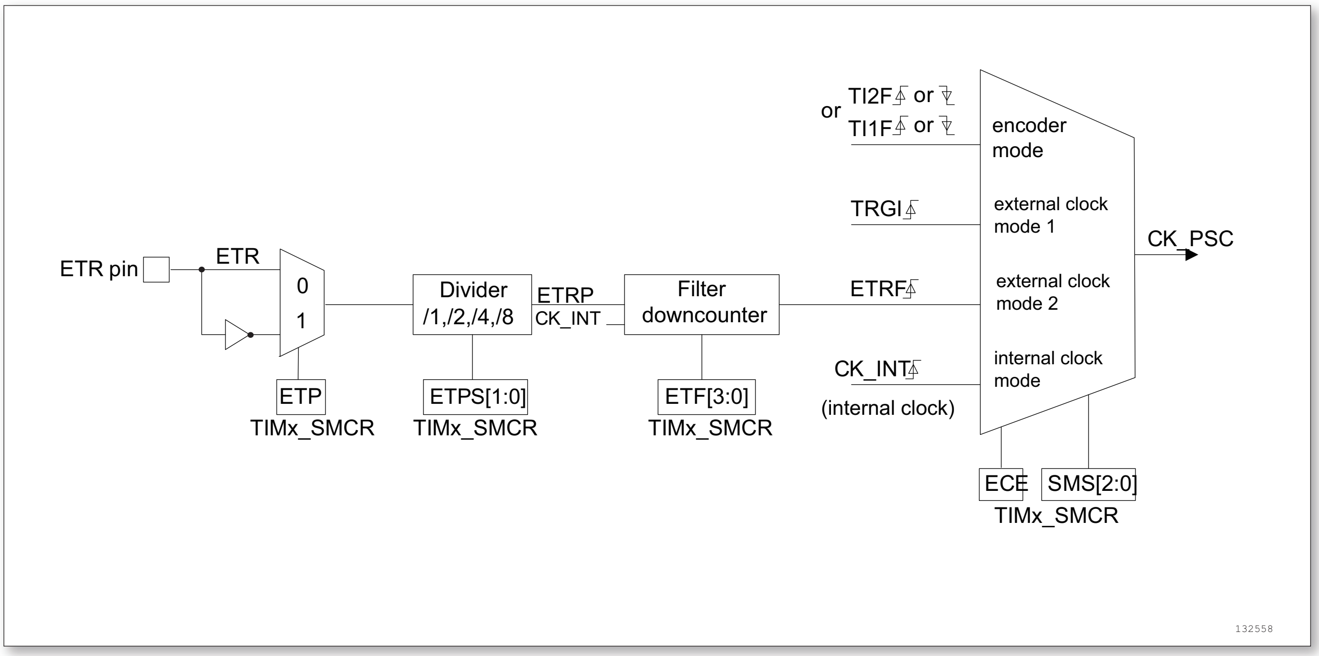


Figure 103. External Trigger Input Block

For example, to configure the upcounter to count once each 2 rising edges on ETR, use the following procedure:

- As no filter is needed in this example, write ETF [3:0]=0000 in the TIMx_SMCR register.
- Set the prescaler by writing ETPS [1:0]=01 in the TIMx_SMCR register.
- Select rising edge detection on the ETR pin by writing ETP=0 in the TIMx_SMCR register.
- Enable external clock mode 2 by writing ECE=1 in the TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter counts once each 2 ETR rising edges.

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

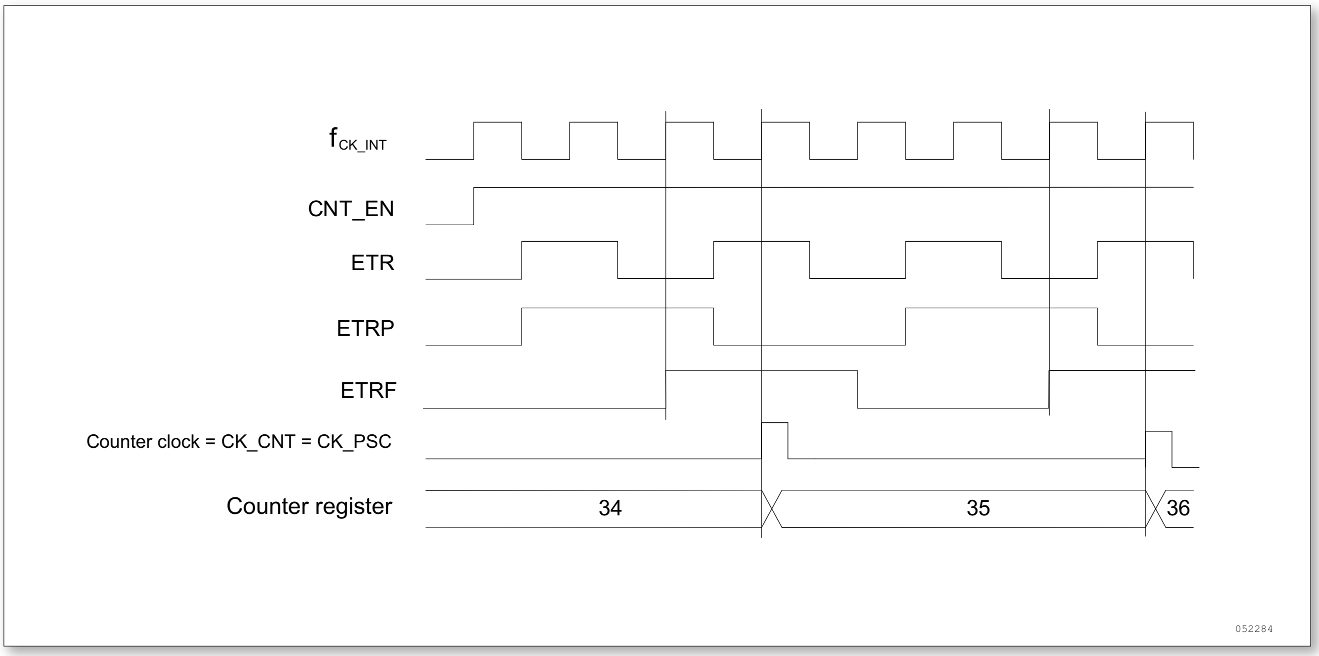


Figure 104. Control Circuit in External Clock Mode 2

12.3.4 Capture/compare channels

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

The following figures show a 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).

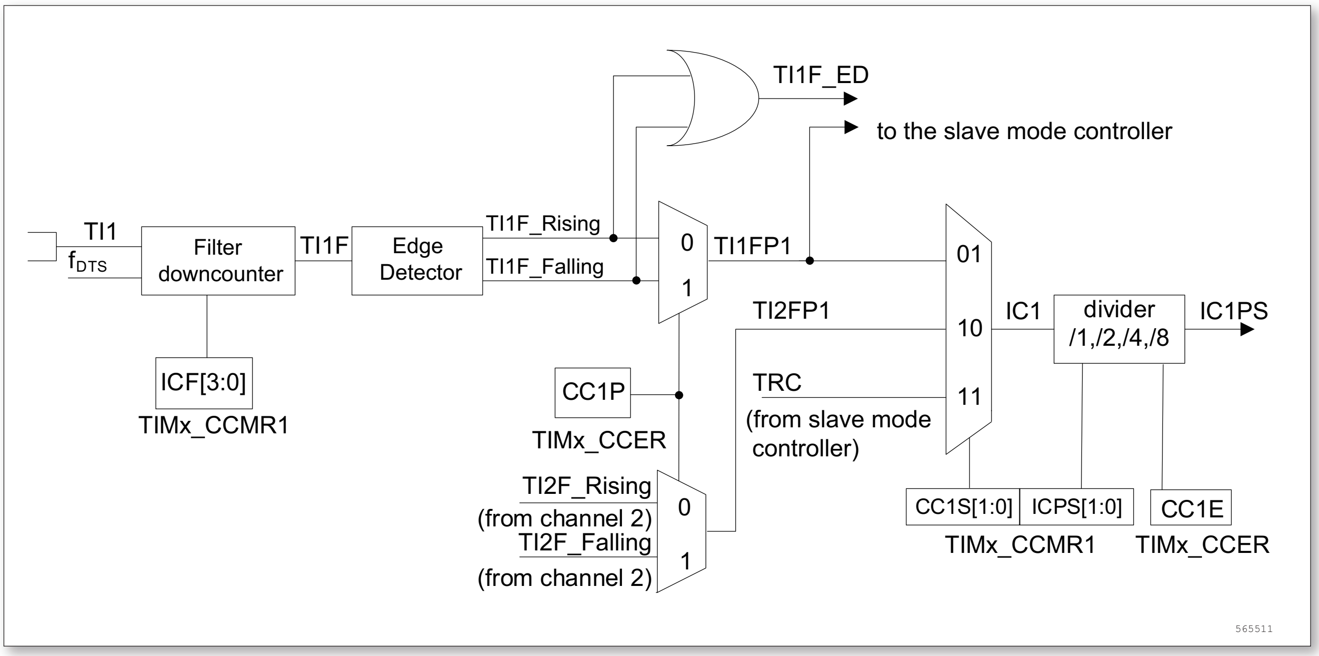


Figure 105. Capture/Compare Channel (Example: Channel 1 Input Stage)

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.

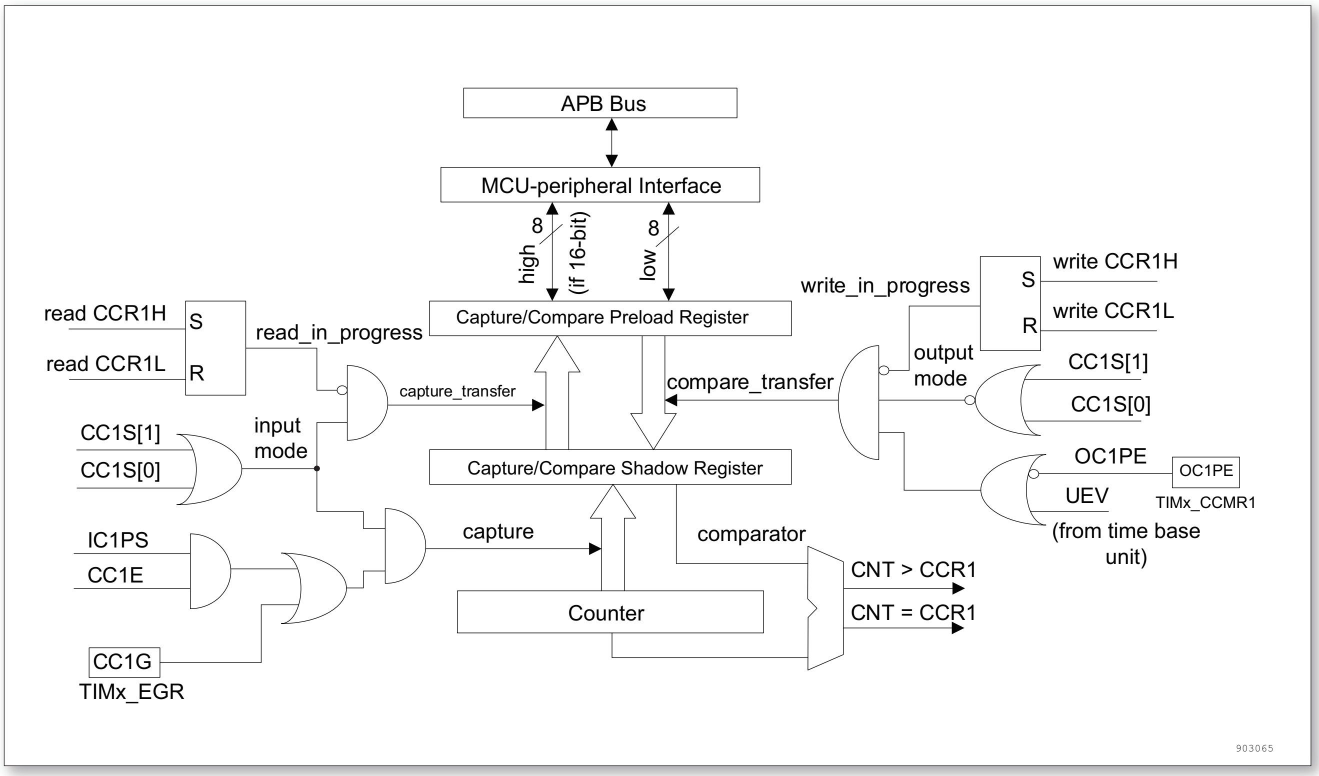


Figure 106. Capture/Compare Channel 1 Main Circuit

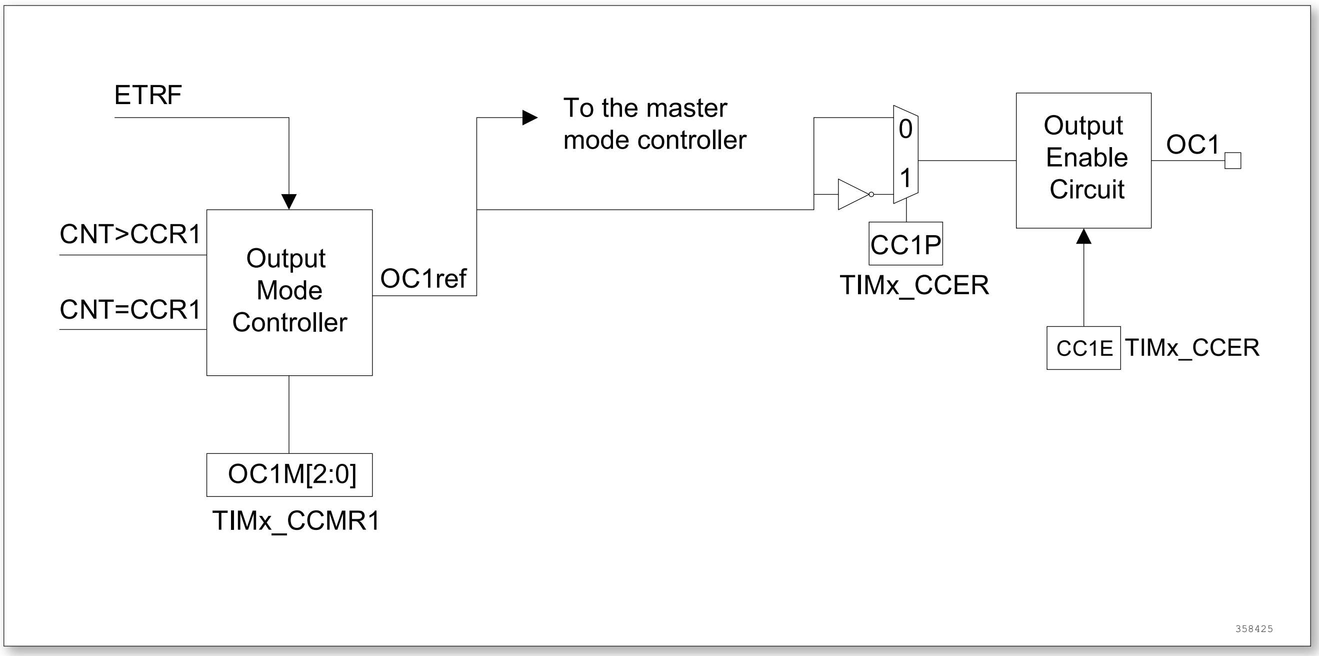


Figure 107. Output Stage of Capture/Compare Channel (Channel 1)

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.

12.3.5 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to ’0’ or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to ’0’.

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

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let’ s imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.

- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Configure the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to ’00’ in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register to ’1’.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- CC1OF is also set to 1.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

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

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

12.3.6 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 bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to ’0’ (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to ’1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the

TIMx_SMCR register.

- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

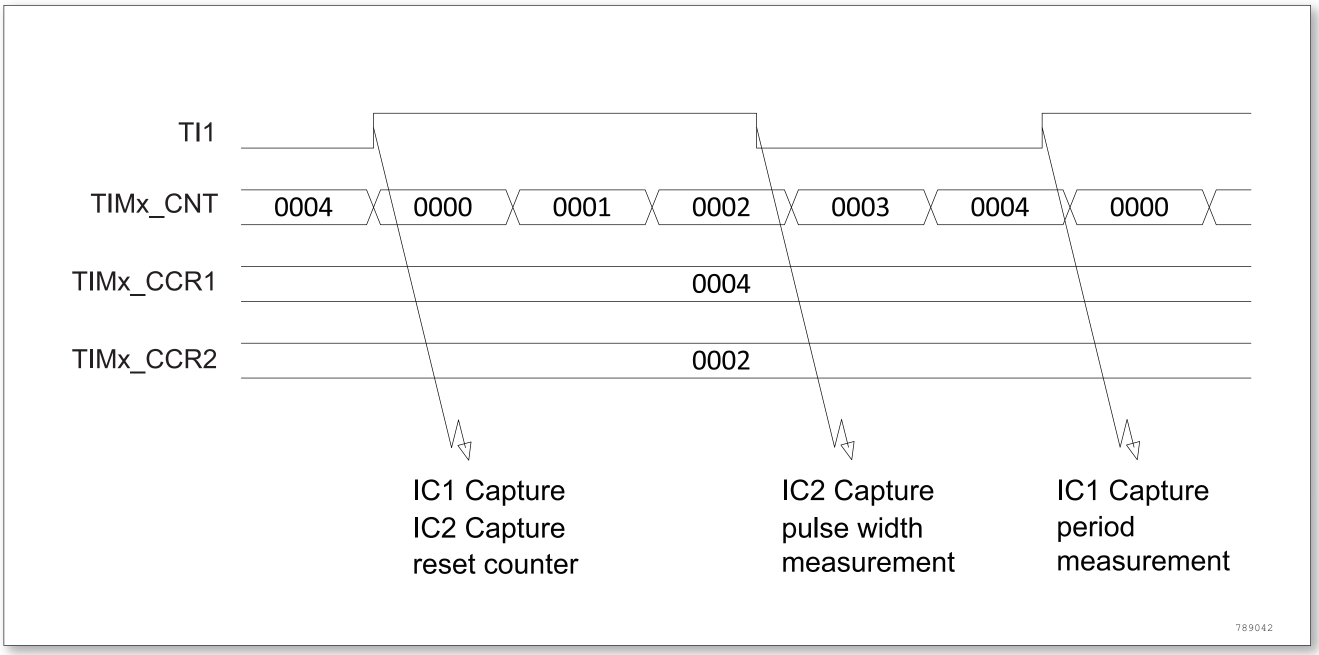


Figure 108. Output Stage of Capture/Compare Channel (Channel 1)

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.

12.3.7 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, being independent of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, in this mode, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

12.3.8 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag bit in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

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

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE and/or CCxDE bits if an interrupt and/or a DMA request is to be generated.
4. Select the output mode. For example, the user must write OCxM=011, OCxPE=0, CCxP=0 and CCxE=1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.
5. Enable the counter by setting the CEN bit in the 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.

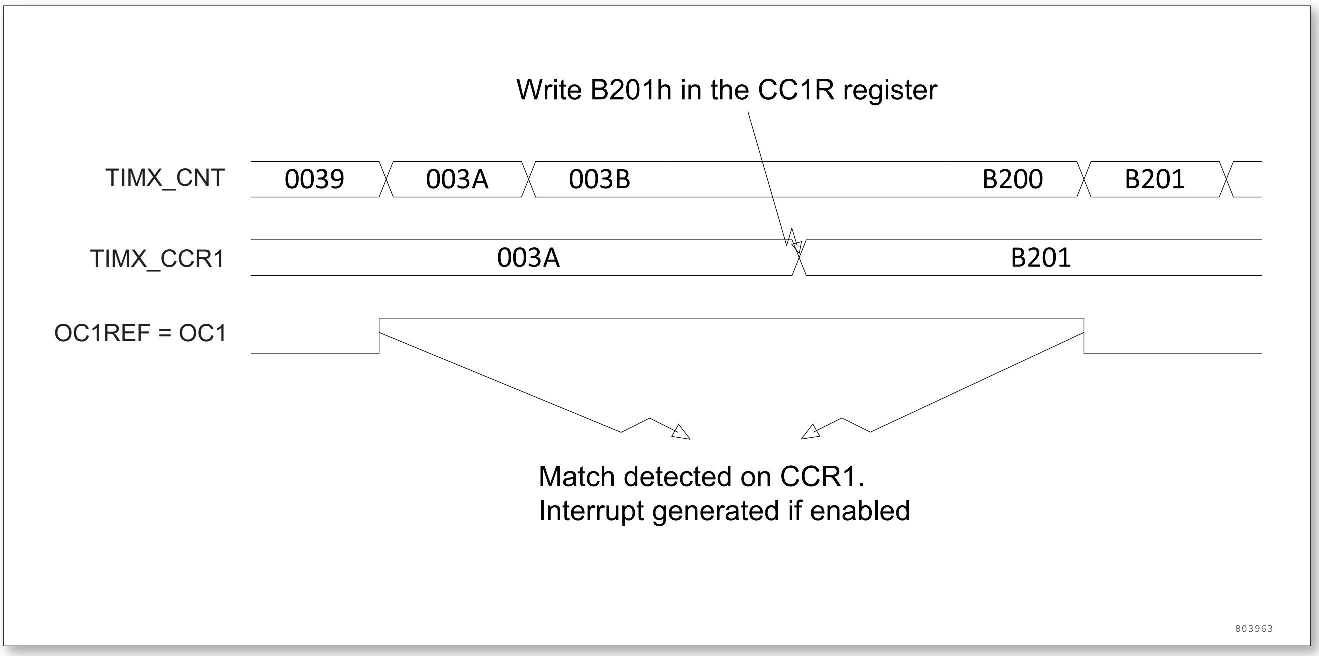


Figure 109. Output Compare Mode (Toggle OC1)

12.3.9 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 bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

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

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by the CCxE bit in the TIMx_CCER register. Refer to the TIMx_CCERx register description for more details.

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). However, to comply with the OCREF_CLR (OCxREF can be cleared by an external event through the ETR signal until the next PWM period), the OCxREF signal is asserted only:

- When the result of the comparison changes
- When the output compare mode (OCxM bits in TIMx_CCMRx register) switches from the period), the OC enabled by the CCxE bit in the TIMx_CCER register. Refer to the

TIMx_CCERx register

This forces the PWM by software while the timer is running. The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx, otherwise it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’. Figure 61 shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

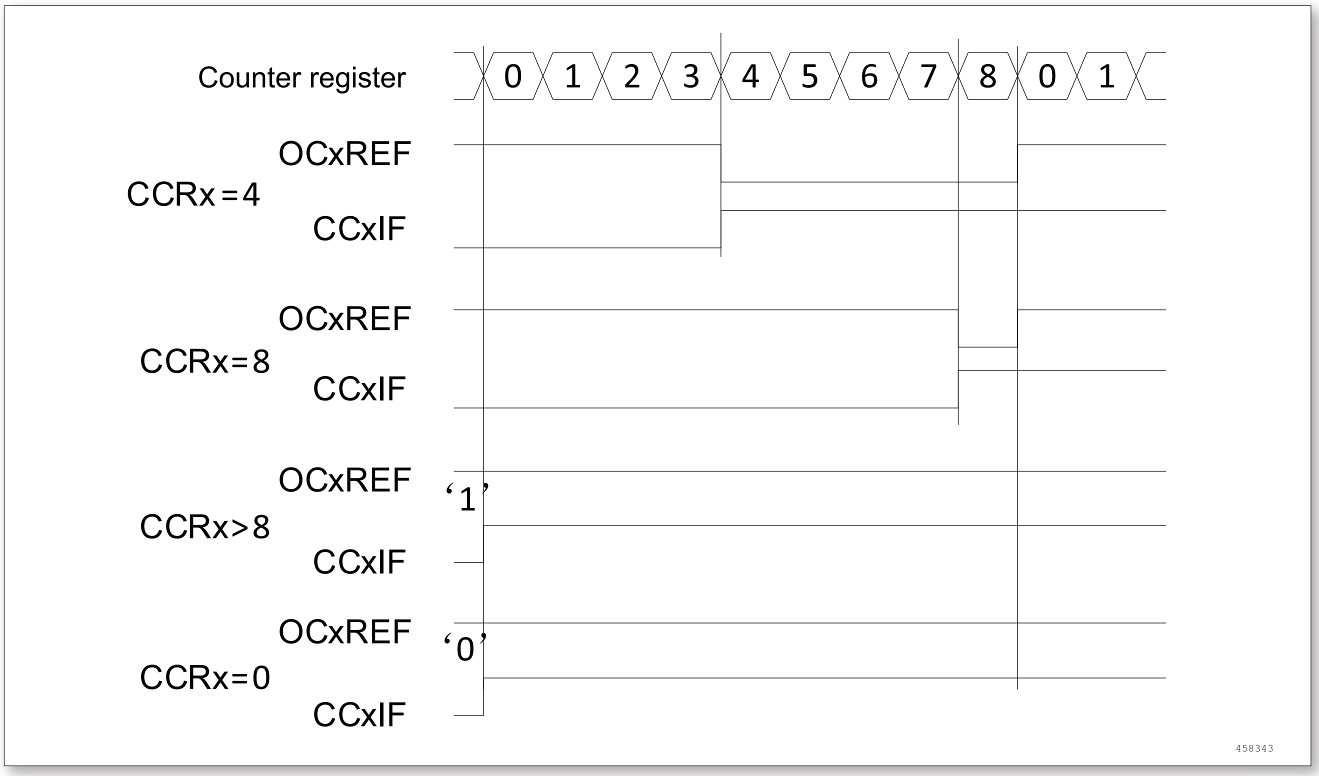


Figure 110. Edge-aligned PWM Waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx, otherwise it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from ‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals).

The compare flag is set when the counter counts up, when it counts down or 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 section 11.3.2 Center-aligned Mode.

The following figure 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.

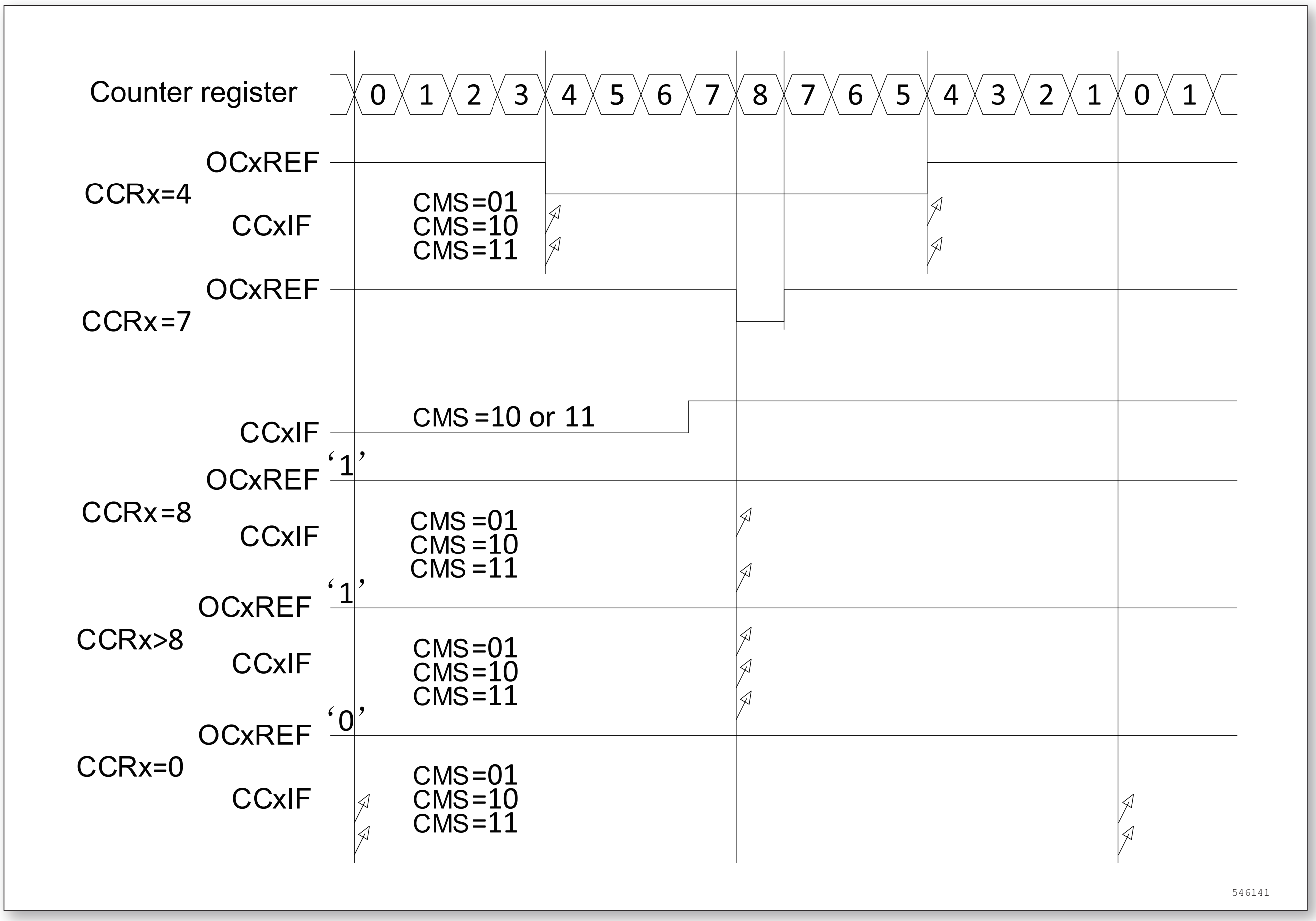


Figure 111. Center-aligned PWM Waveforms (ARR = 8)

Hints in center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT>TIMx_ARR). For example, if the counter was counting up, it will continue to count up.

– The direction is updated if the user writes 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated

- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

12.3.10 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 $\leq$ ARR (in particular, 0 $<$ CCRx)
- In downcounting: CNT $>$ CCRx

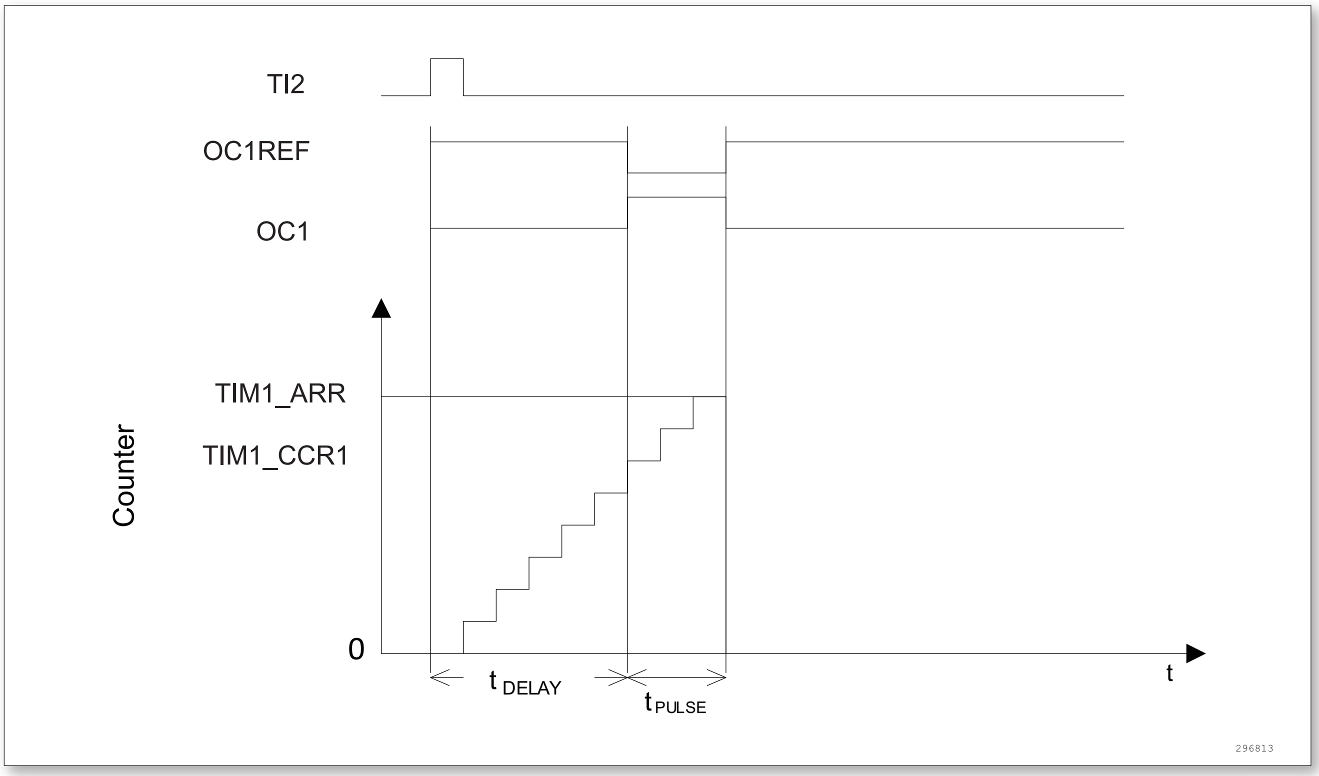


Figure 112. Example of One Pulse Mode

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

Let’s use TI2FP2 as trigger 1:

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

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

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let 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 ‘0’ 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 (Single mode), so ’1’ must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

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 t_{DELAY} 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.

12.3.11 Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMRx register set to ‘1’). The OCxREF signal remains Low until the next update event, UEV, occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to ‘00’.
- The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
- The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

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

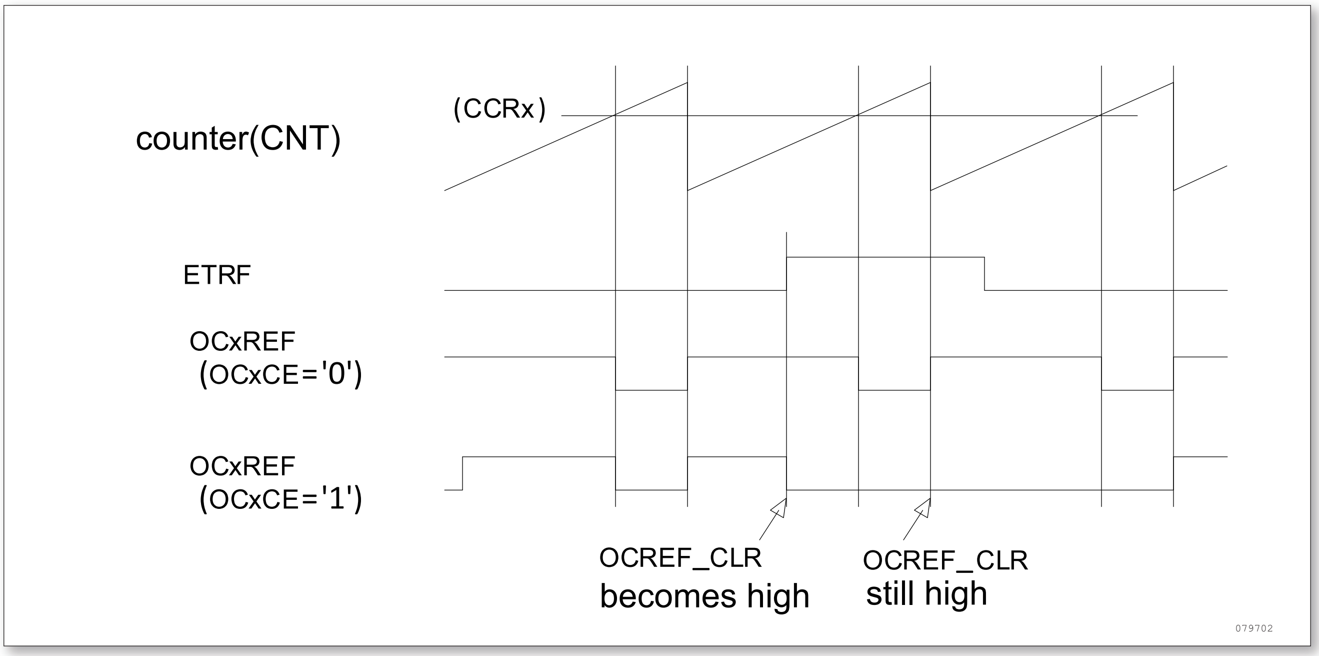


Figure 113. Clearing TIMx OCxREF

12.3.12 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. Assuming that the counter is enabled (CEN bit in TIMx_CR1 register written to ‘1) in the following table, it 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). 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 corresponds 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 41. 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 indicate 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:

- CC1S=' 01' (TIMx_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S=' 01' (TIMx_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P=' 0' , (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1=TI1).
- C2P= '0' (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2=TI2).
- SMS= '011' (TIMx_SMCR register, all inputs are active on both rising and falling edges).
- CEN= '1' (TIMx_CR1 register, Counter enabled).

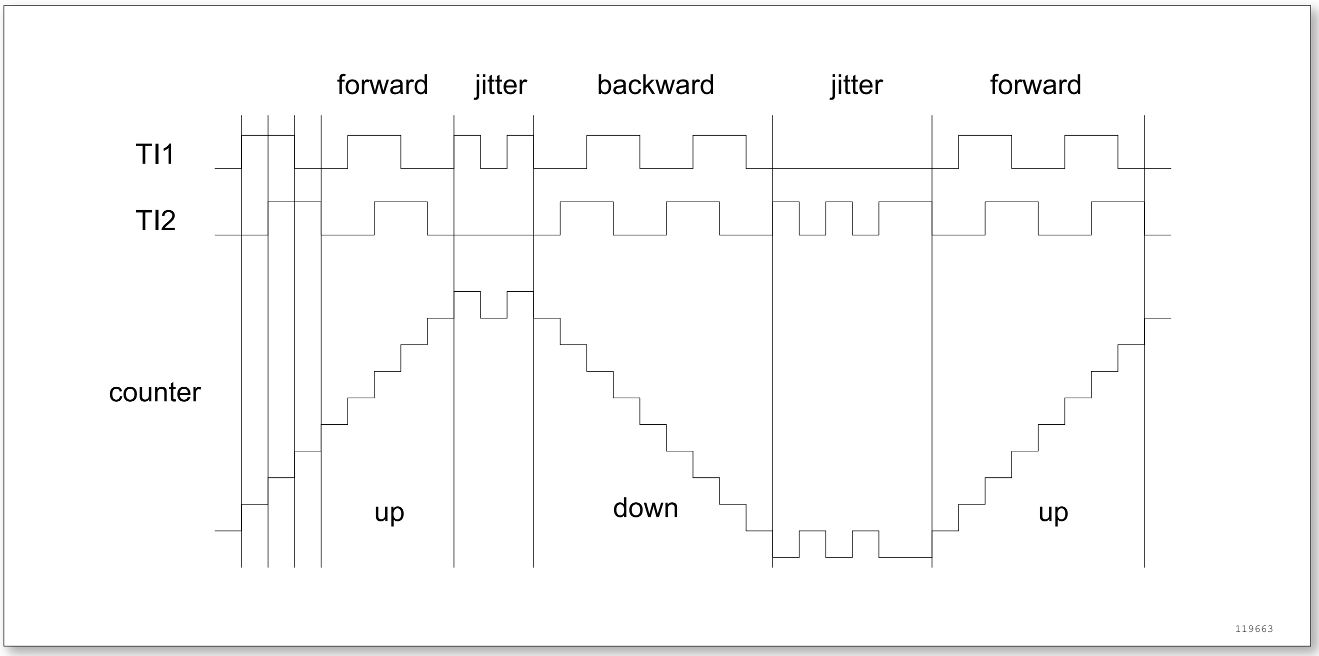


Figure 114. Example of Counter Operation in Encoder Mode

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

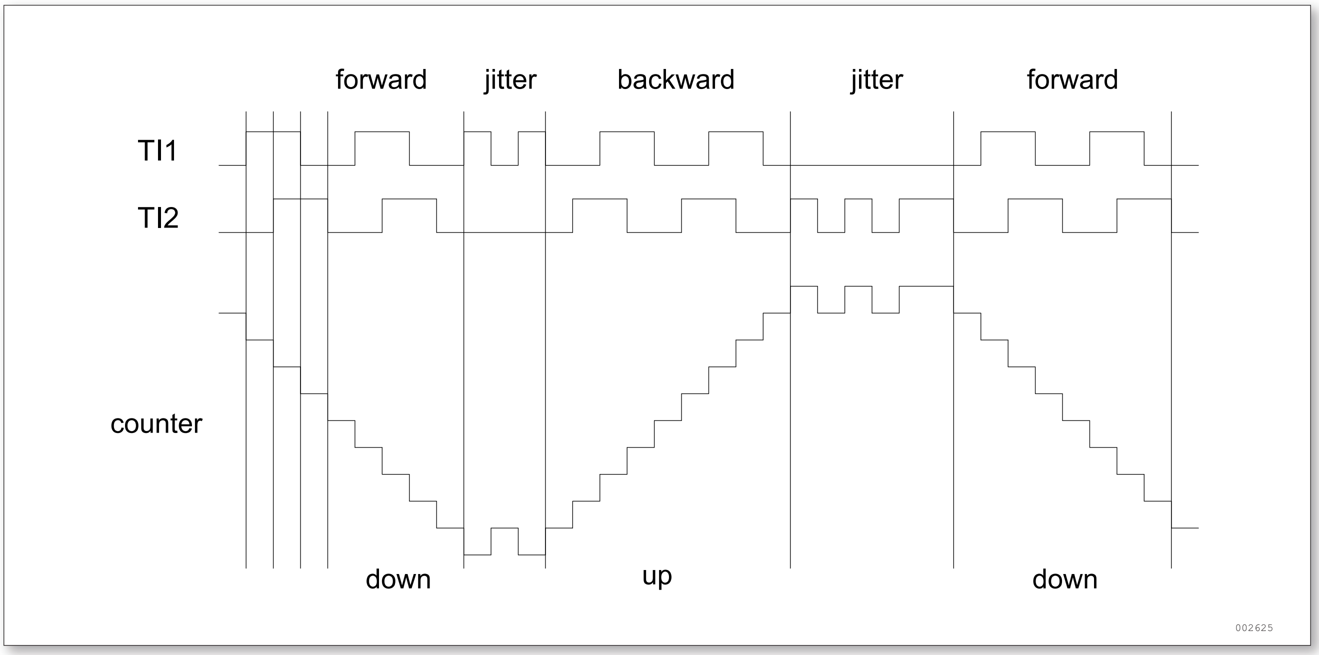


Figure 115. Example of Encoder Interface Mode with Inverted Polarity IC1FP1

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). When available, it is also possible to read its value through a DMA request generated by a real-time clock.

12.3.13 Timer input XOR function

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

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in section 11.3.18.

12.3.14 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.

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 the user does not 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.
- Start the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request, or a DMA request can be sent if enabled depending on the TIE (interrupt enable) and TDE (DMA enable) bits in TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

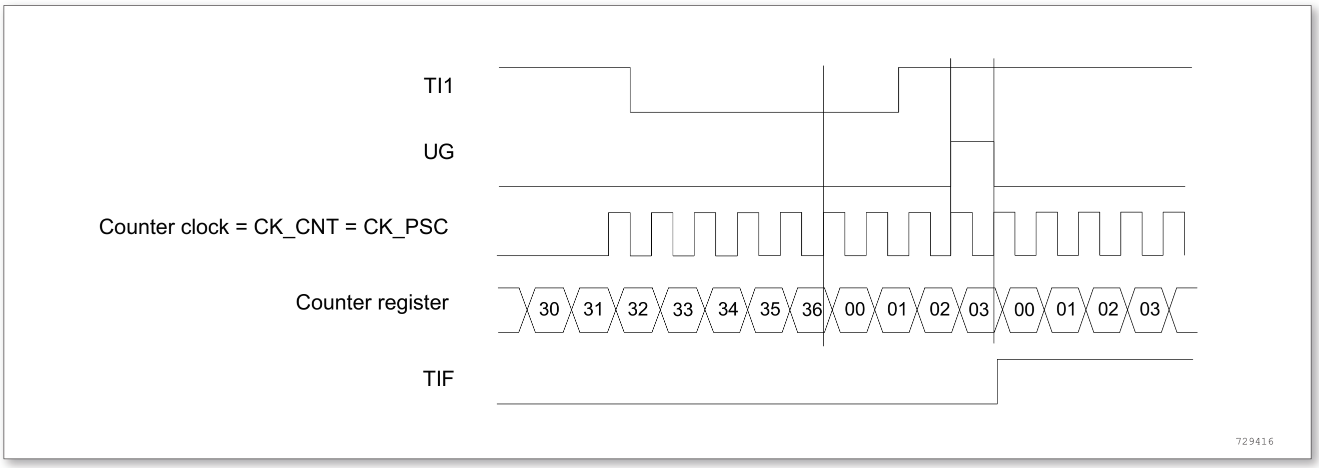


Figure 116. Control Circuit in Reset Mode

Slave mode: Gated mode

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

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

- Configure the channel 1 to detect low level on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the 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.
- Start the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn't start if CEN=0, whatever is the trigger input level

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

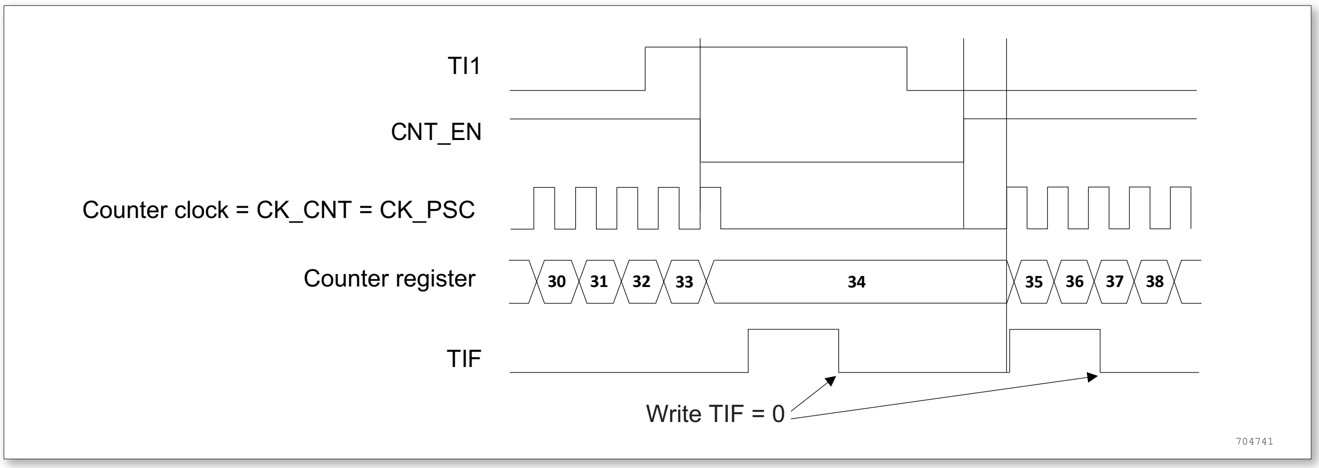


Figure 117. Control Circuit in Gated Mode

Slave mode: Trigger mode

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

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

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there's no need to configure it. The CC2S bits are only configured to select CC2P=1 in TIMx_CCER register, so as 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 stop of the counter is due to the resynchronization circuit on TI2 input.

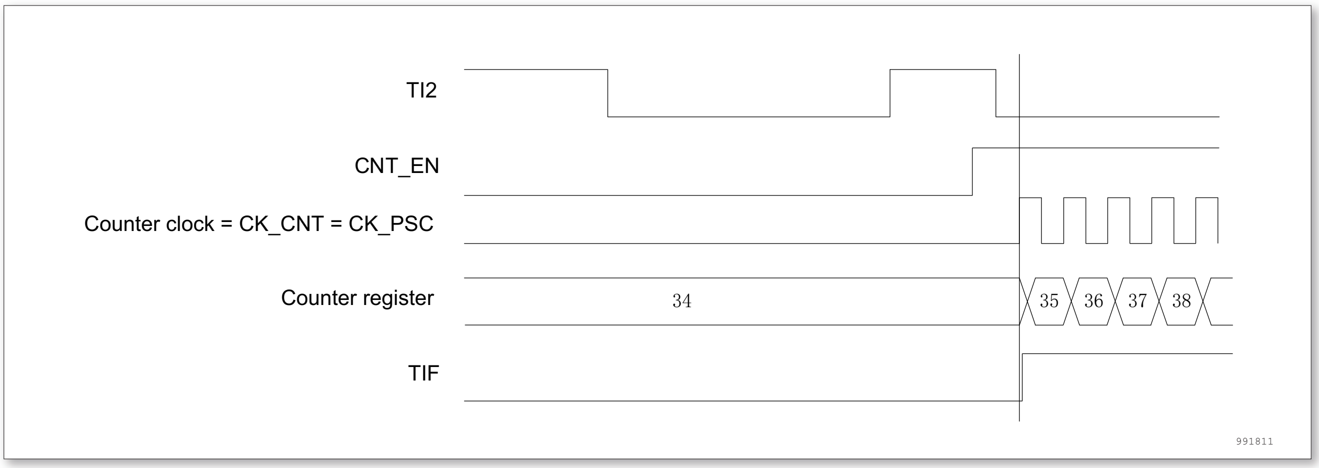


Figure 118. Control Circuit in Trigger Mode

Slave mode: External clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

- Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
  - ETF = 0000: no filter
  - ETPS = 00: prescaler disabled
  - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.

- Configure the channel 1 as follows, to detect rising edges on T1:
  - IC1F=0000: no filter.
  - The capture prescaler is not used for triggering and does not need to be configured.
  - CC1S=01 in TIMx_CCMR1 register to select only the input capture source.
  - CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edge only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

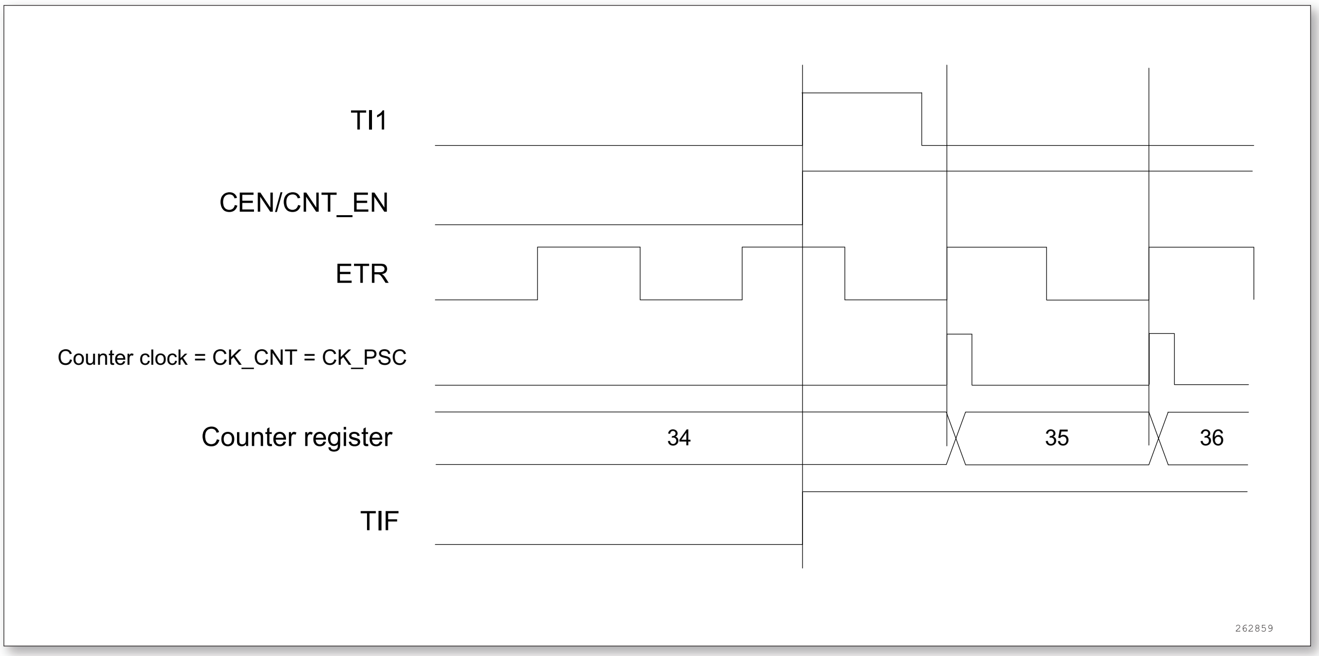


Figure 119. Control Circuit in External Clock Mode 2 + Trigger Mode

12.3.15 Timer synchronization

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

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

Using one timer as prescaler for another timer

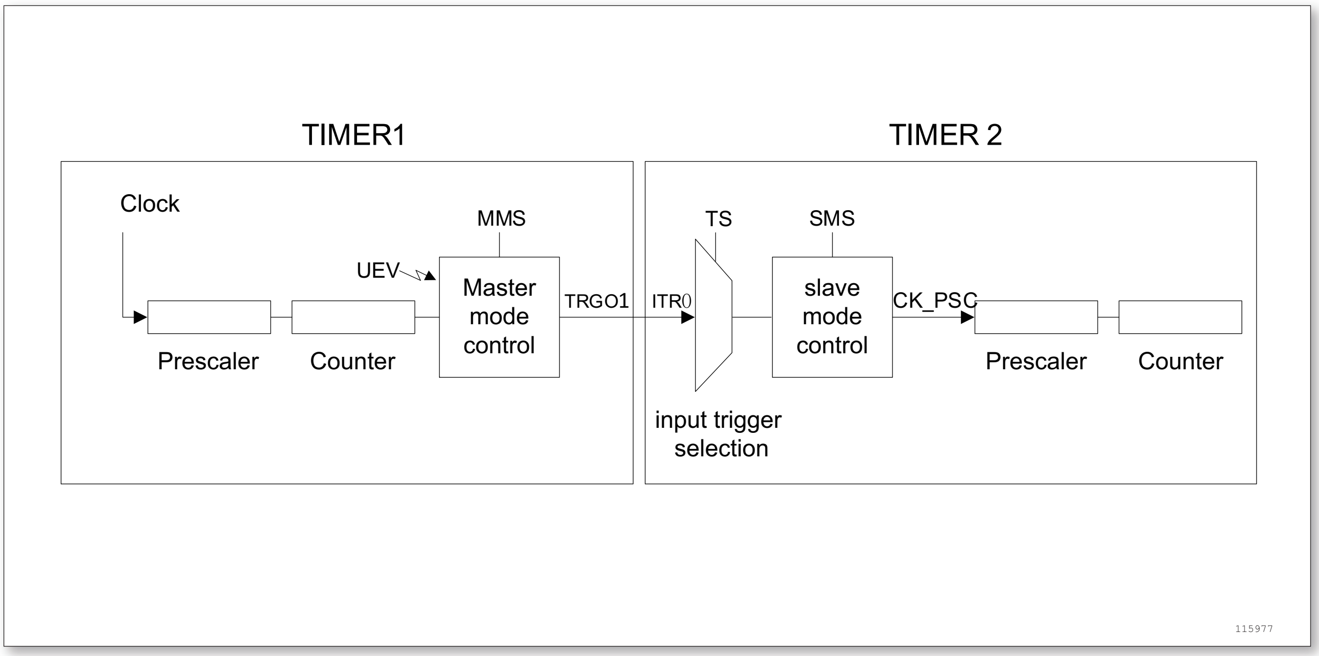


Figure 120. Master/Slave Timer Example

For example, the user can configure Timer 1 to act as a prescaler for Timer 2 (see the above figure). To do this:

- 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 2, Timer 2 must be configured in slave mode using ITR1 as internal trigger. You select this through the TS bits in the TIM2_SMCR register (writing TS=000).
- Then you put the slave mode controller in external clock mode 1 (write SMS=111 in the TIM2_SMCR register). This causes Timer 2 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 2.

Using one timer to enable another timer

In this example, we control the enable of Timer 2 with the output compare 1 of Timer 1. Refer to the following figure for connections. Timer 2 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to 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 2 to get the input trigger from Timer 1 (TS=001 in the TIM2_SMCR

register).

- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Enable Timer 2 by writing ‘1 in the CEN bit (TIM2_CR1 register).
- Enable Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).

Note: The counter 2 clock is not synchronized with counter 1, this mode only affects the Timer 2 counter enable signal.

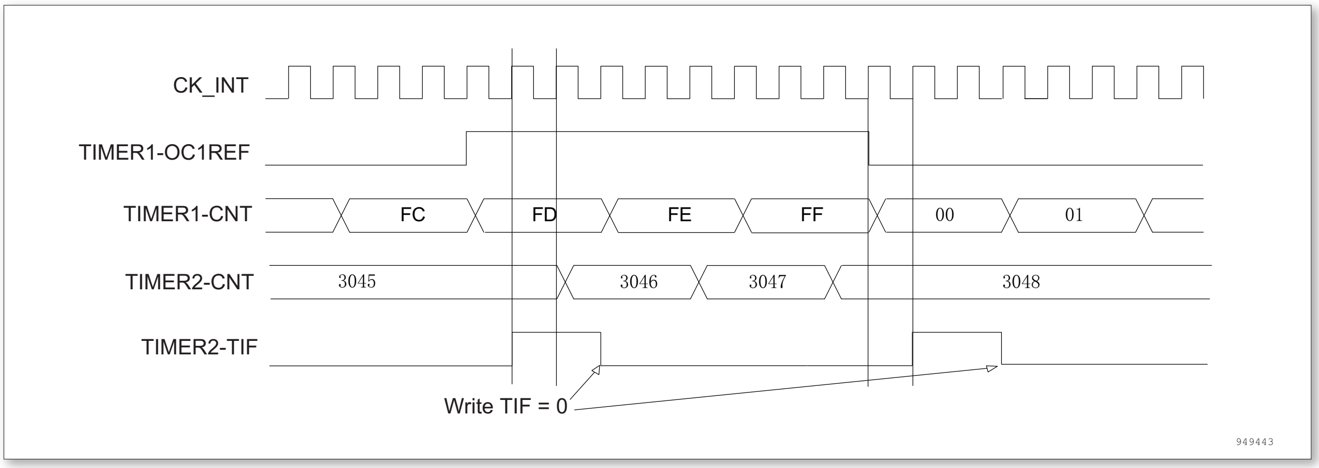


Figure 121. Gating Timer 2 with OC1REF of Timer 1

In the example in the above figure, the Timer 2 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 software using the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 2. Timer 1 is the master and starts from 0. Timer 2 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 2 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 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Reset Timer 1 by writing ‘1 in UG bit (TIM1_EGR register).
- Reset Timer 2 by writing ‘1 in UG bit (TIM2_EGR register).
- Initialize Timer 2 to 0xE7 by writing ‘0xE7’ in the Timer 2 counter (TIM2_CNT).
- Enable Timer 2 by writing ‘1 in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing ‘0 in the CEN bit (TIM1_CR1 register).

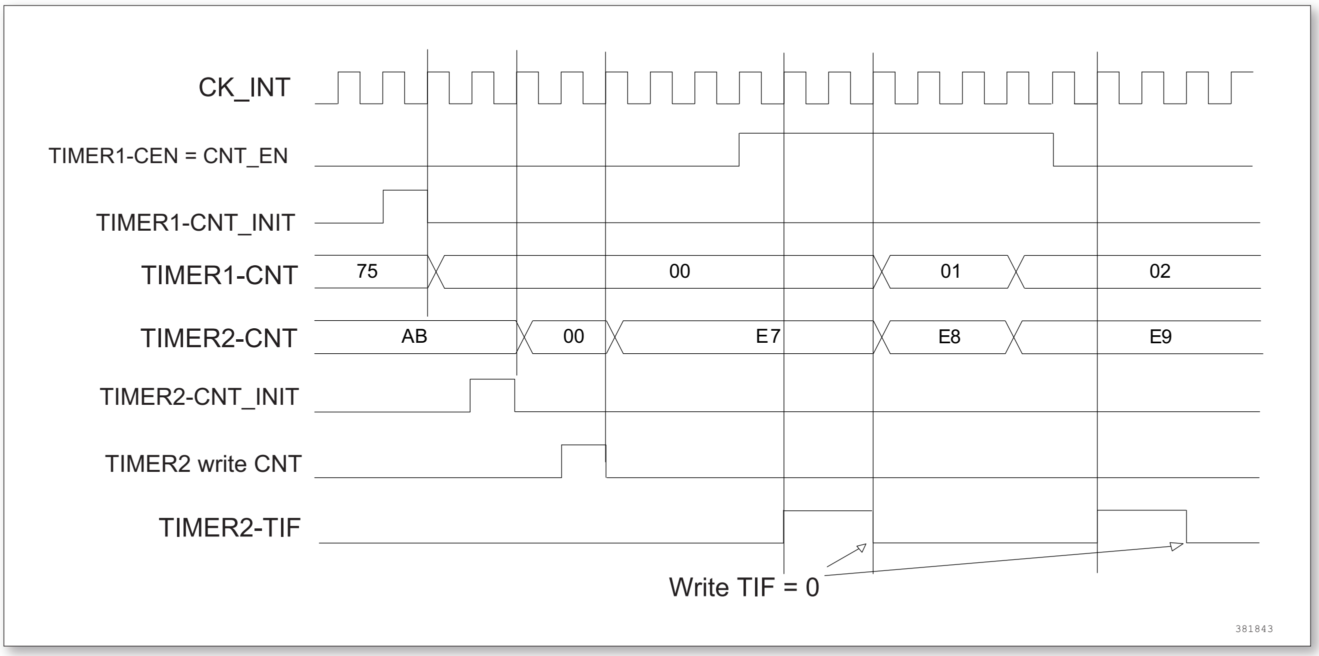


Figure 122. Gating Timer 2 with Enable of Timer 1

Using one timer to start another timer

In this example, we set the enable of Timer 2 with the update event of Timer 1. Refer to the following figure for connections. Timer 2 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 2 receives the trigger signal its CEN bit is automatically set and the counter counts until we write ’0’ to the CEN bit in the TIM2_CR1 register. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS=010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in TIM2_SMCR register).
- Start Timer 1 by writing ’1’ in the CEN bit (TIM1_CR1 register)

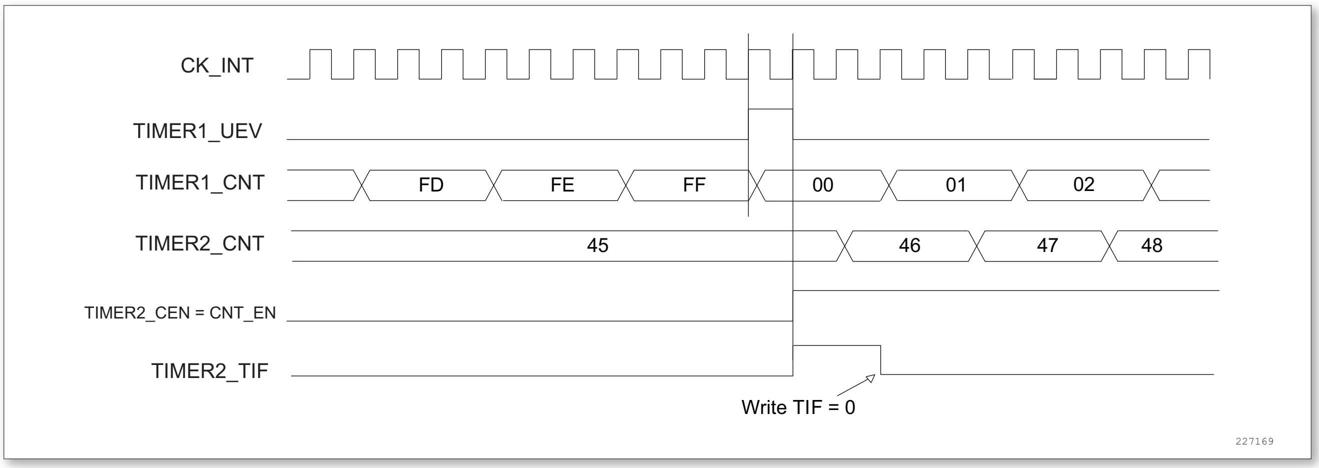


Figure 123. Triggering Timer 2 with Update of Timer 1

As in the previous example, the user can initialize both counters before starting counting. The following figure shows the behavior with the same configuration as ’0’ but in trigger mode instead of gated mode (SMS=110 in the TIM2_SMCR register).

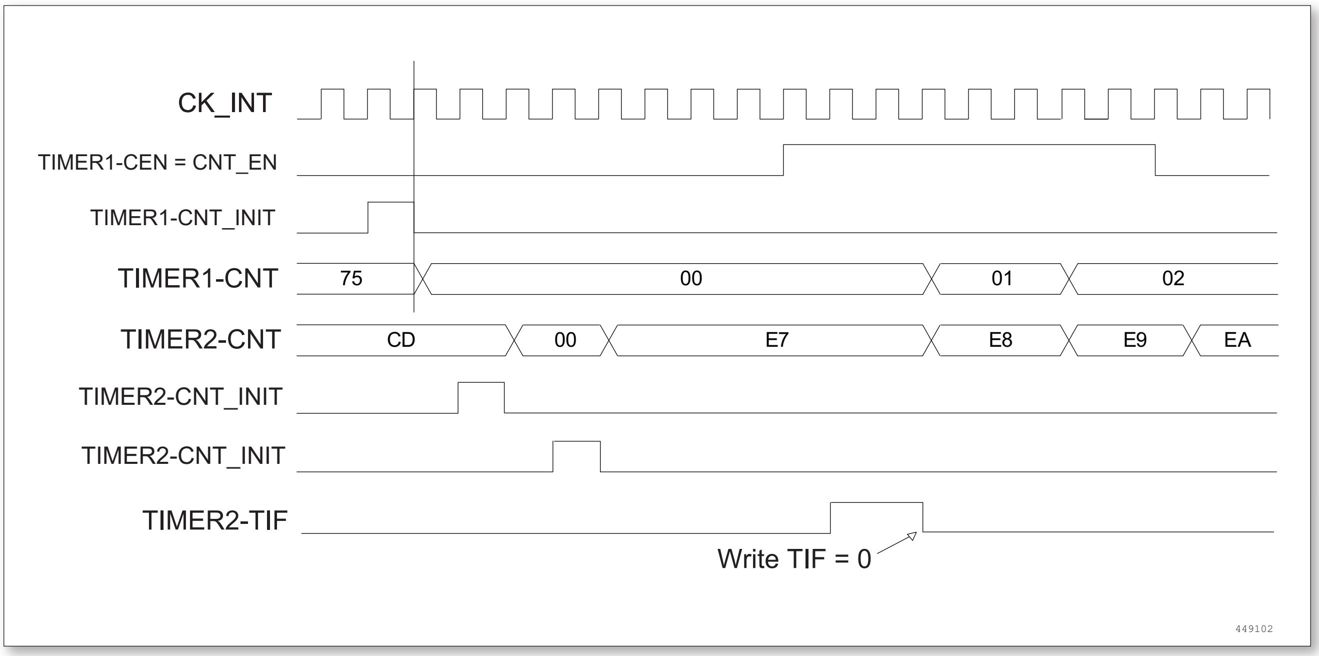


Figure 124. Triggering Timer 2 with Enable of Timer 1

Using one additional timer as prescaler for another timer

In this example, we use Timer 1 as the prescaler for Timer 2. The configuration is as follows:

- Configure Timer 1 in master mode, togenerate the update event (UEV) as the trigger output (MMS=010 in the TIM1_CR2 register). Then, output a periodic signal in case of each counter overflow.
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in the external clock mode (write SMS=111 in the TIM2_SMCR reg-

ister).

- Start Timer 2 by writing ’1’ in the CEN bit (TIM1_CR2 register)
- Start Timer 1 by writing ’1’ in the CEN bit (TIM1_CR1 register).

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 2 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 2):

- 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 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in the TIM2_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 that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer 1.

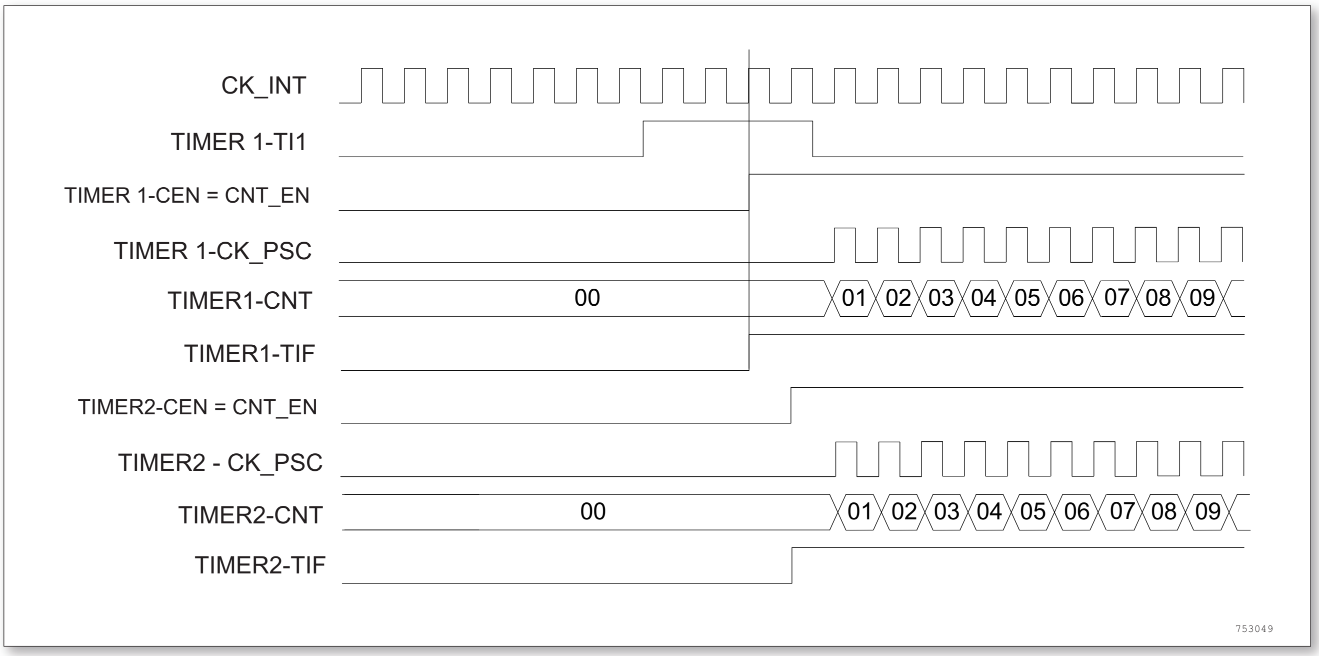


Figure 125. Triggering Timer 1 and 2 with Timer 1 TI1 input

12.3.16 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 ”Debug” sections.

12.4 TIMx register description

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

Table 42. Summary of TIMx Register

Offset	Acronym	Register Name	Reset	Section
0x00	TIMx_CR1	Control register 1	0x00000000	section 12.4.1
0x04	TIMx_CR2	Control register 2	0x00000000	section 12.4.2
0x08	TIMx_SMCR	Slave mode control register	0x00000000	section 12.4.3
0x0C	TIMx_DIER	DMA /interrupt enable register	0x00000000	section 12.4.4
0x10	TIMx_SR	Status register	0x00000000	section 12.4.5
0x14	TIMx_EGR	Event generation register	0x00000000	section 12.4.6
0x18	TIMx_CCMR1	Capture/compare mode register 1	0x00000000	section 12.4.7
0x1C	TIMx_CCMR2	Capture/compare mode register 2	0x00000000	section 12.4.8
0x20	TIMx_CCER	Capture/compare enable register	0x00000000	section 12.4.9
0x24	TIMx_CNT	Counter	0x00000000	section 12.4.10
0x28	TIMx_PSC	Prescaler	0x00000000	section 12.4.11
0x2C	TIMx_ARR	Auto-reload register	0x00000000	section 12.4.12
0x34	TIMx_CCR1	Capture/compare register 1	0x00000000	section 12.4.13
0x38	TIMx_CCR2	Capture/compare register 2	0x00000000	section 12.4.14
0x3C	TIMx_CCR3	Capture/compare register 3	0x00000000	section 12.4.15
0x40	TIMx_CCR4	Capture/compare register 4	0x00000000	section 12.4.16
0x48	TIMx_DCR	DMA control register	0x00000000	section 12.4.17
0x4C	TIMx_DMAR	DMA address in continuous mode	0x00000000	section 12.4.18

12.4.1 Control register 1(TIMx_CR1)

Offset address: 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	rw	rw

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
9: 8	CKD	rw	0x00	Clock division The 2 bits indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock used by the dead-time generators and the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6: 5	CMS	rw	0x00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
4	DIR	rw	0x00	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Bit	Field	Type	Reset	Description
2	URS	rw	0x00	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generates an update interrupt or DMA request if enabled.These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	rw	0x00	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller, buffered registers are then loaded with their preload values 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	rw	0x00	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.

12.4.2 Control register 2(TIMx_CR2)

Offset address: 0x04

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7	TI1S	rw	0x00	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6: 4	MMS	rw	0x00	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance, a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	rw	0x00	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs

Bit	Field	Type	Reset	Description
2: 0	Reserved			Reserved, always read as 0.

12.4.3 Slave mode control register(TIMx_SMCR)

Offset address: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS		ETF				MSM	TS			OCCS	SMS		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	ETP	rw	0x00	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
14	ECE	rw	0x00	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13: 12	ETPS	rw	0x00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8

Bit	Field	Type	Reset	Description
11: 8	ETF	rw	0x00	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS}. 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, 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
7	MSM	rw	0x00	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.

Bit	Field	Type	Reset	Description
6: 4	TS	rw	0x00	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1(ITR1) 010: Internal Trigger 2(ITR2) 011: Internal Trigger 3(ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2(TI2FP2) 111: External Trigger input (ETRF) See the following table for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	OCCS	rw	0x00	Output compare clear selection In PWM mode, clear the comparator output 1: Comparator output as clear signal 0: External trigger signal as clear signal

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x00	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 (refer to Input Control register and Control Register description). 000: Slave mode disabled - if CEN = ‘1’ then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP1 edge depending on TI1FP2 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP2 edge depending on TI2FP1 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger input becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger input (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS= ’100’). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 43. TIMx Internal Trigger Connection

Slave timer	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM3	TIM1	TIM2	x	x

12.4.4 DMA/interrupt enable register(TIMx_DIER)

Offset address: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	Reserved			Reserved, always read as 0.
14	TDE	rw	0x00	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	Reserved			Reserved, always read as 0.
12	CC4DE	rw	0x00	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	rw	0x00	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	rw	0x00	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	rw	0x00	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	rw	0x00	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	Reserved			Reserved, always read as 0.
6	TIE	rw	0x00	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	Reserved			Reserved, always read as 0.
4	CC4IE	rw	0x00	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	rw	0x00	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	rw	0x00	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled

Bit	Field	Type	Reset	Description
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

12.4.5 Status register(TIMx_SR)

Offset address: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			CC4OF	CC3OF	CC2OF	CC1OF	Res.		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.
12	CC4OF	rc_w0	0x00	Capture/Compare 4 overcapture flag Refer to CC1OF description.
11	CC3OF	rc_w0	0x00	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	rc_w0	0x00	Capture/Compare 2 overcapture flag Refer to CC1OF description.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8: 7	Reserved			Reserved, always read as 0.
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved			Reserved, always read as 0.
4	CC4IF	rc_w0	0x00	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	rc_w0	0x00	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	rc_w0	0x00	Capture/Compare 2 interrupt flag Refer to CC1IF description.

Bit	Field	Type	Reset	Description
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if repetition counter= 0) and if the UDIS=0 in the TIMx_CR1 register. – When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register. –When CNT is reinitialized by a trigger event (refer to the description of synchronization control register), if URS=0 and UDIS=0 in the TIMx_CR1 register.

12.4.6 Event generation register (TIMx_EGR)

Offset address: 0x14

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 7	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	Reserved			Reserved, always read as 0.
4	CC4G	w	0x00	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	w	0x00	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	w	0x00	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler factor is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), otherwise, it takes the auto-reload value (TIMx_ARR) if DIR=1 (downcounting).

12.4.7 Capture/compare mode register 1(TIMx_CCMR1)

Offset address: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M			OC2PE	OC2FE	CC2S		OC1CE	OC1M			OC1PE	OC1FE	CC1S	
IC2F				IC2PSC				IC1F				IC1PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x00	Output compare 2 clear enable
14: 12	OC2M	rw	0x00	Output compare 2 mode
11	OC2PE	rw	0x00	Output compare 2 preload enable
10	OC2FE	rw	0x00	Output compare 4 fast enable
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7	OC1CE	rw	0x00	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF Input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input

Bit	Field	Type	Reset	Description
6: 4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT<TIMx_CCR1 otherwise inactive. In downcounting, channel 1 is inactive (OC1REF= ‘0’) as long as TIMx_CNT>TIMx_CCR1 otherwise active (OC1REF=’ 1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1otherwise active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 otherwise inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=’ 00’ (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=’00’(the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC2F	rw	0x00	Input capture 2 filter
11: 10	IC2PSC	rw	0x00	Input capture 2 prescaler
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7: 4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 6 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 8 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 5 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 6 0100: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 6 1100: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 8 0101: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 8 1101: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 5 0110: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 6 1110: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 6 0111: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 8 1111: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 8

Bit	Field	Type	Reset	Description
3: 2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the factor of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=’0’(TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input. 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	rw	0x00	Capture/compare 1 selection This bit-field defines the direction of the channel (in-put/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ‘0’ in TIMx_CCER).

12.4.8 Capture/compare mode register 2(TIMx_CCMR2)

Offset address: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M			OC4PE	OC4FE	CC4S		OC3CE	OC3M			OC3PE	OC3FE	CC3S	
IC4F				IC4PSC				IC3F				IC3PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x00	Output compare 4 clear enable
14: 12	OC4M	rw	0x00	Output compare 4 mode
11	OC4PE	rw	0x00	Output compare 4 preload enable
10	OC4FE	rw	0x00	Output compare 4 fast enable

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER)
7	OC3CE	rw	0x00	Output compare 3 clear enable
6: 4	OC3M	rw	0x00	Output compare 3 mode
3	OC3PE	rw	0x00	Output compare 3 preload enable
2	OC3FE	rw	0x00	Output compare 3 fast enable
1: 0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER)

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC4F	rw	0x00	Input capture 4 filter
11: 10	IC4PSC	rw	0x00	Input capture 4 prescaler

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER).
7: 4	IC3F	rw	0x00	Input capture 3 filter
3: 2	IC3PSC	rw	0x00	Input capture 3 prescaler
1: 0	CC3S	rw	0x00	Capture/compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI4 10: CC3 channel is configured as input, IC3 is mapped on TI3 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER).

12.4.9 Capture/compare enable register(TIMx_CCER)

Offset address: 0x20

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 14	Reserved			Reserved, always read as 0.
13	CC4P	rw	0x00	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	rw	0x00	Capture/Compare 4 output enable Refer to CC1E description.

Bit	Field	Type	Reset	Description
11: 10	Reserved			Reserved, always read as 0.
9	CC3P	rw	0x00	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	rw	0x00	Capture/Compare 3 output enable Refer to CC1E description.
7: 6	Reserved			Reserved, always read as 0.
5	CC2P	rw	0x00	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	rw	0x00	Capture/Compare 2 output enable Refer to CC1E description.
3: 2	Reserved			Reserved, always read as 0.
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel is configured as output: 0: OC1 active high 1: OC1 active low CC1 channel is configured as input: This bit selects whether IC1 or inverted IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted Note: This bit can not be modified as long as LOCK level 3 or 2 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel is configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel is configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table 44. Output Control Bit for Standard OCx Channels

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

Note: The state of the external I/O pins connected to the standard OCx channels depends on the OCx channel state and the GPIO and AFIO registers.

12.4.10 Counter(TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CNT	rw	0x0000	Counter value

12.4.11 Prescaler(TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} /(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in ”reset mode”)

12.4.12 Auto-reload register(TIMx_ARR)

Offset address: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	ARR	rw	0x0000	Prescaler value ARR is the value to be loaded in the actual auto-reload register. Refer to section 13.3.1 for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

12.4.13 Capture/compare register 1(TIMx_CCR1)

Offset address: 0x34

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR1	rw	0x0000	Capture/Compare 1 value If CC1 channel is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Otherwise the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

12.4.14 Capture/compare register2(TIMx_CCR2)

Offset address: 0x38

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR2	rw	0x0000	Capture/Compare 2 value If CC2 channel is configured as output: CCR2 contains the value to be loaded in the actual capture/compare 2 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Otherwise the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If CC2 channel is configured as input: CCR2 contains the counter value transferred by the last input capture 2 event (IC2).

12.4.15 Capture/compare register 3(TIMx_CCR3)

Offset address: 0x3C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR3	rw	0x0000	Capture/Compare 3 value If CC3 channel is configured as output: CCR3 contains the value to be loaded in the actual capture/compare 3 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Otherwise the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. If CC3 channel is configured as input: CCR3 contains the counter value transferred by the last input capture 3 event (IC3).

12.4.16 Capture/compare register 4(TIMx_CCR4)

Offset address: 0x40

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR4	rw	0x0000	Capture/Compare 4 value If CC4 channel is configured as output: CCR4 contains the value to be loaded in the actual capture/compare 4 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Otherwise the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If CC4 channel is configured as input: CCR4 contains the counter value transferred by the last input capture 4 event (IC4).

12.4.17 DMA control register(TIMx_DCR)

Offset address: 0x48

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			DBL					Res.			DBA				
			rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
12: 8	DBL	w	0x00	DMA burst length This bit field defines the burst transfer in the continuous mode (the timer detects a burst transfer when a write access to the TIMx_DMAR register address is performed), namely, the number of transfers, in half-word (double bytes) or bytes. 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers .... 10001: 18 transfers Example: Let us consider the following transfer: DBL = 7 and DBA = TIM2_CR1. - If DBL =7 and DBA = TIM2_CR1 represent the address of data to be transferred, the transfer address is given by: (Address of TIMx_CR1) + DBA + (DMA index), where, DMA index = DBL TIMx_CR1 address + DBA + 7 is the address of data to be written or read, so that the transfer is completed to/from 7 registers starting from the TIMx_CR1 address + DBA. According to the setting of DMA data length, the following case may occur: -If the data is set to half word (16 bits), the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, the user must specify the data width of DMA transfer for the timer.
7: 5	Reserved			Reserved, always read as 0.
4: 0	DBA	w	0x00	DMA base address These bits define the base-address for DMA transfers in the continuous mode (when write access is done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ......

12.4.18 DMA address for full transfer(TIMx_DMAR)

Offset address: 0x4C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	DMAB	w	0x0000	DMA register for burst accesses A write operation to the TIMx_DMAR register will access the register located at the following address: TIMx_CR1 address + DBA + DMA index, Where: ‘TIMx_CR1 address’ is the address of the control register 1; ‘DBA’ is the DMA base address configured in TIMx_DCR register; ‘DMA index’ is the offset automatically controlled by the DMA transfer, depending on DBL configured in TIMx_DCR.

13 32-bit general-purpose timers (TIMx32 Bit)

32-bit general-purpose timers (TIMx32 Bit)

13.1 TIMx introduction

General-purpose timers consist of a 32-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

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

TIMx are completely independent, and do not share any resources. They can be synchronized together as described in Section Timer Synchronization.

13.2 TIMx Main features

TIM2 functions include:

- 32-bit up, down, up/down auto-reload register
- 16-bit programmable prescaler allowing dividing (modifing in real time) the counter clock frequency either by any factor between 1 and 65536.
- Up to 4 independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge and Center-aligned Mode)
  - One-pulse mode output
- circuit to control the timer with external signals and to interconnect several timers together.
- Interrupt/DMA generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes

• Trigger input for external clock or cycle-by-cycle current management

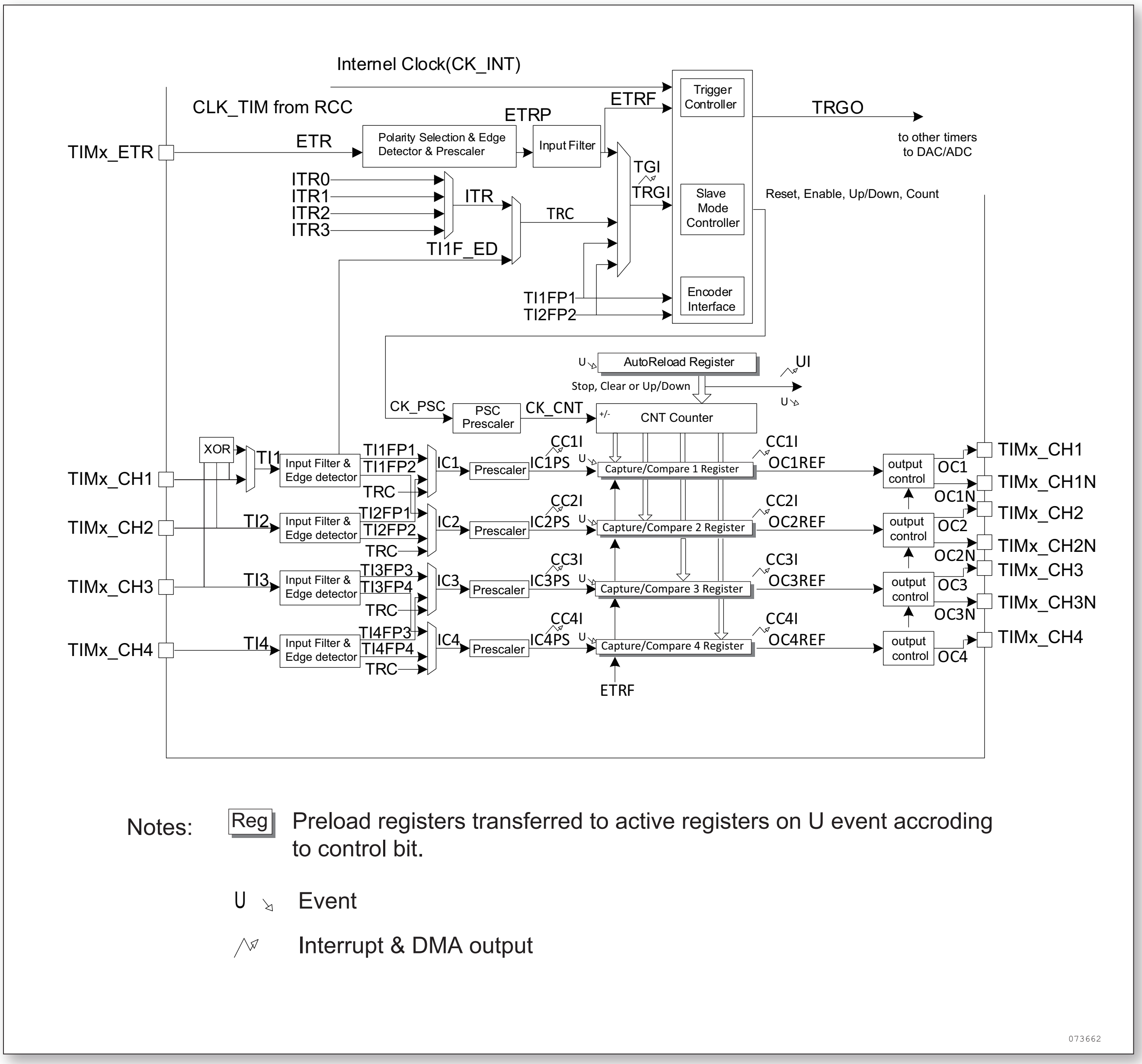


Figure 126. Block Diagram of general-purpose timer

13.3 TIMx Functional description

13.3.1 Time-base unit

The main block of the programmable general-purpose timer is a 32-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 auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in details 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).

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 32-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 factor is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler factor is changed on the fly:

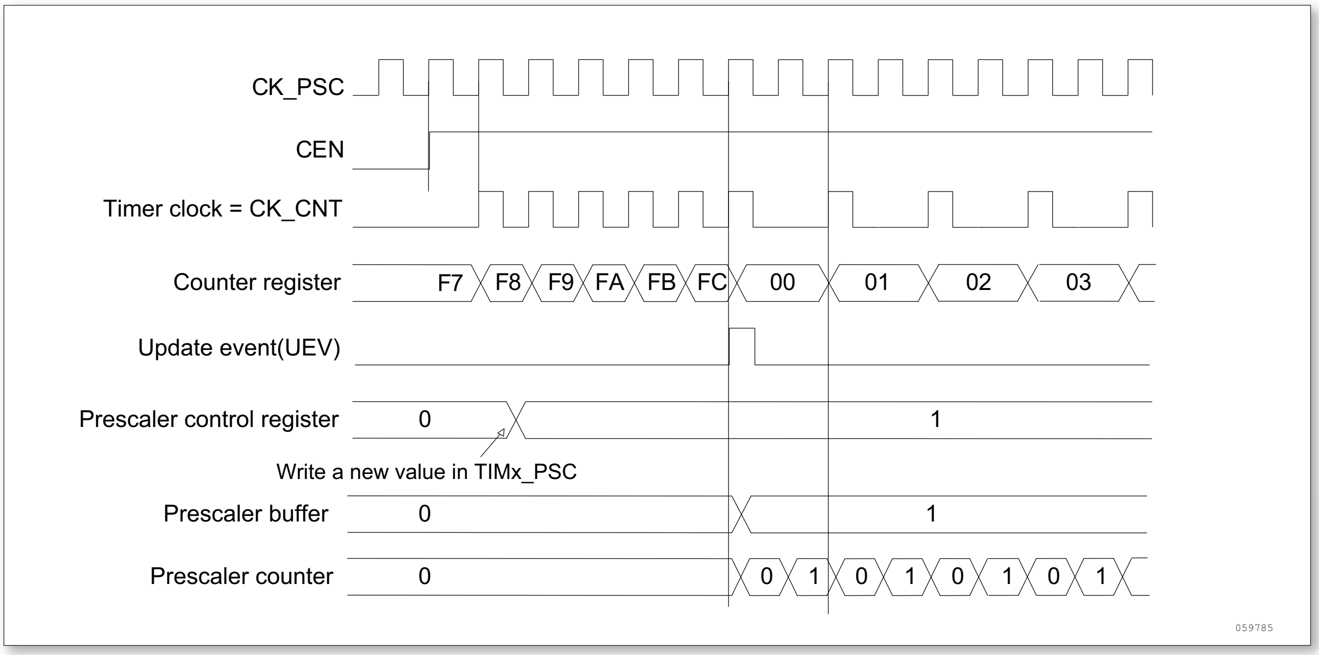


Figure 127. Counter Timing Diagram with Prescaler Division Change from 1 to 2

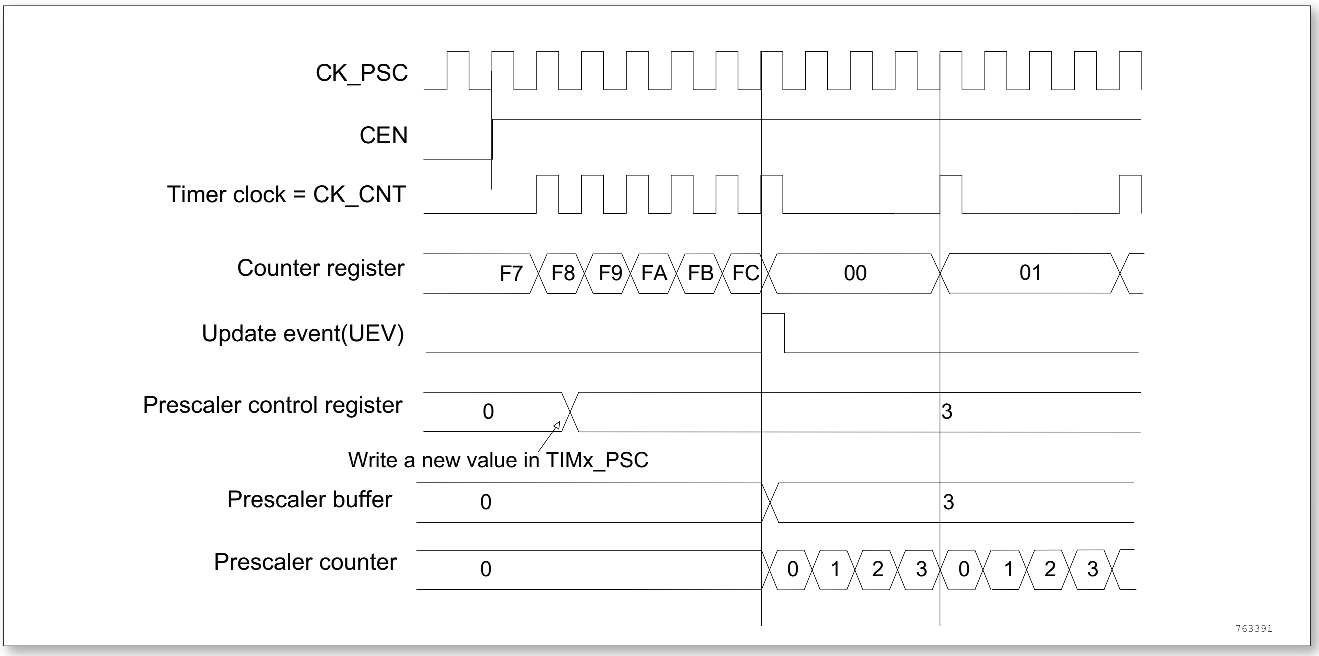


Figure 128. Counter Timing Diagram with Prescaler Division Change from 1 to 4

13.3.2 Counter modes

Upcounting mode

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

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 software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change).

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag bit(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 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.

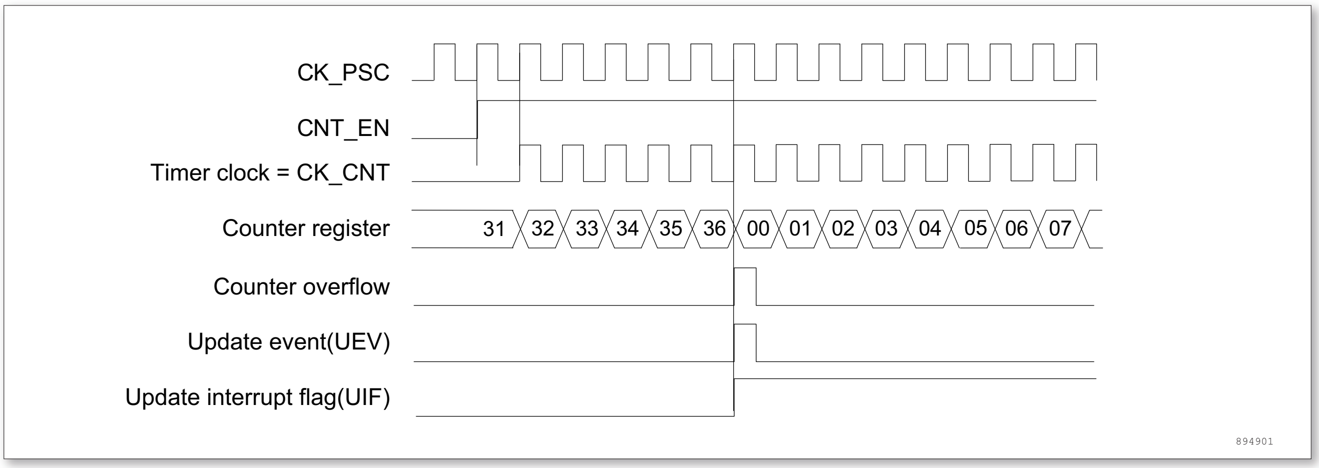


Figure 129. Counter Timing Diagram, Internal Clock Divided by 1

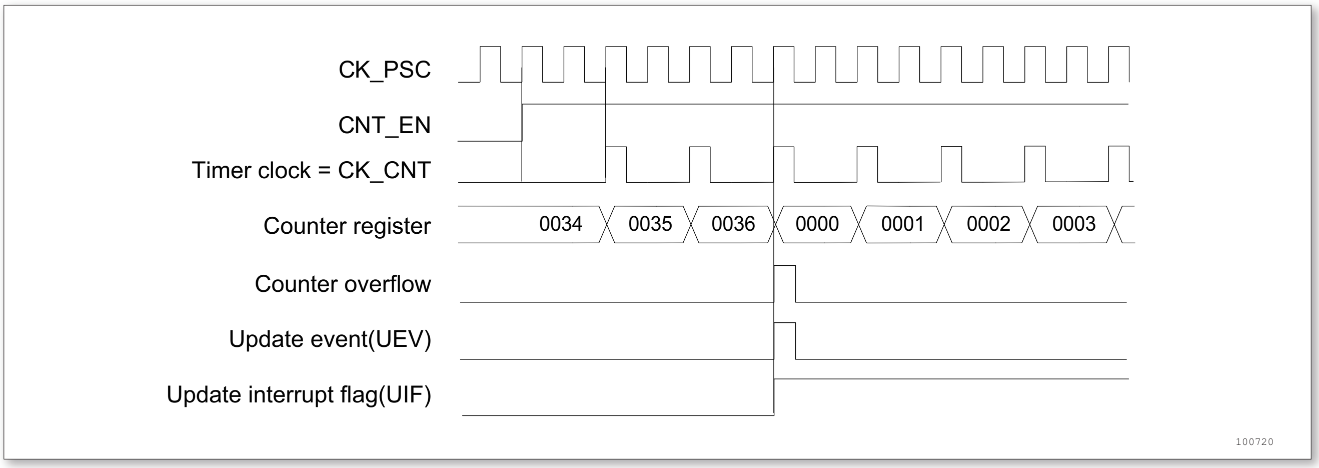


Figure 130. Counter Timing Diagram, Internal Clock Divided by 2

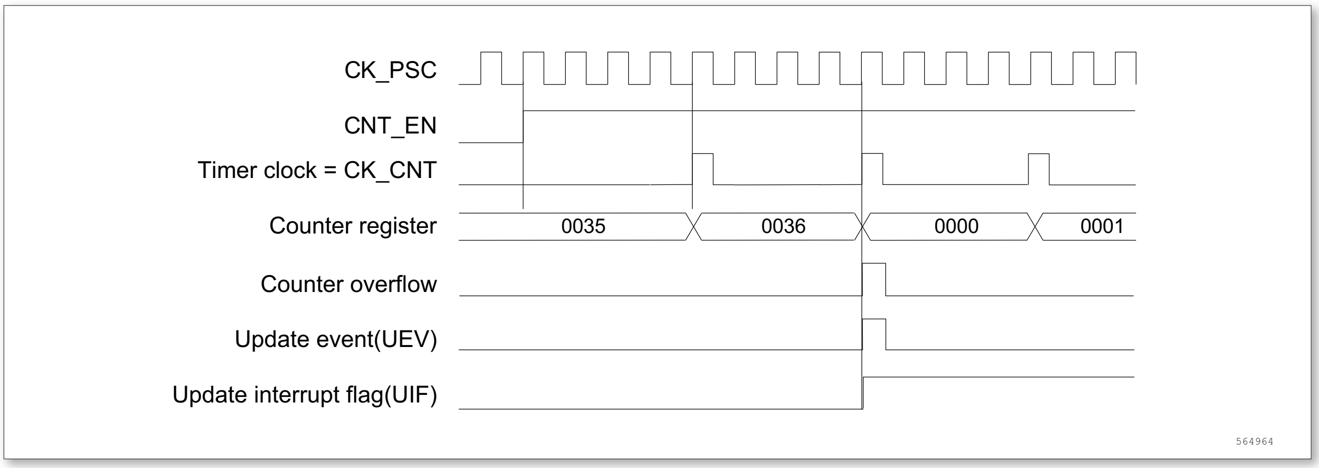


Figure 131. Counter Timing Diagram, Internal Clock Divided by 4

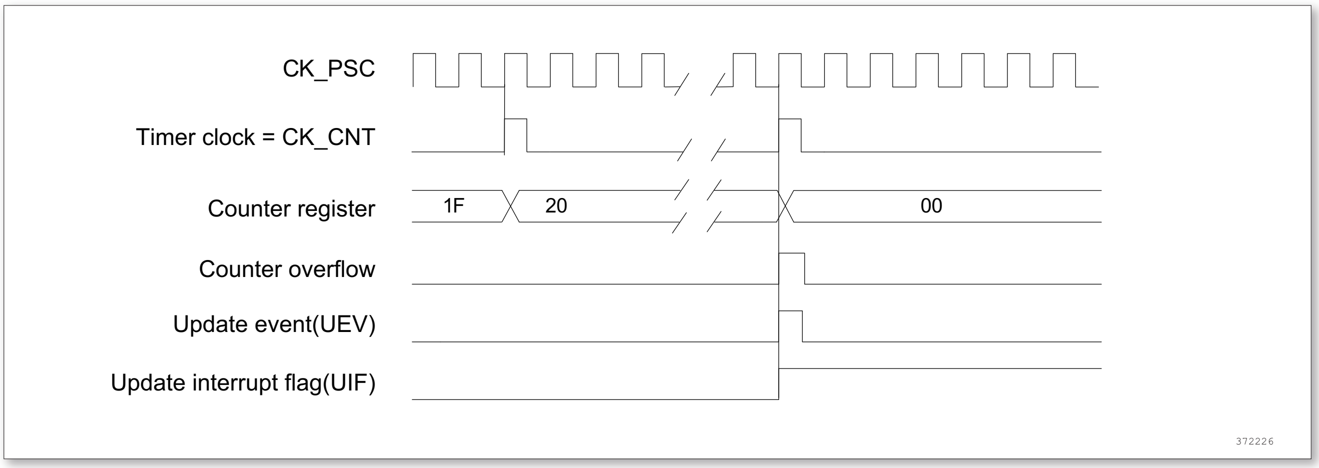


Figure 132. Counter Timing Diagram, Internal Clock Divided by N

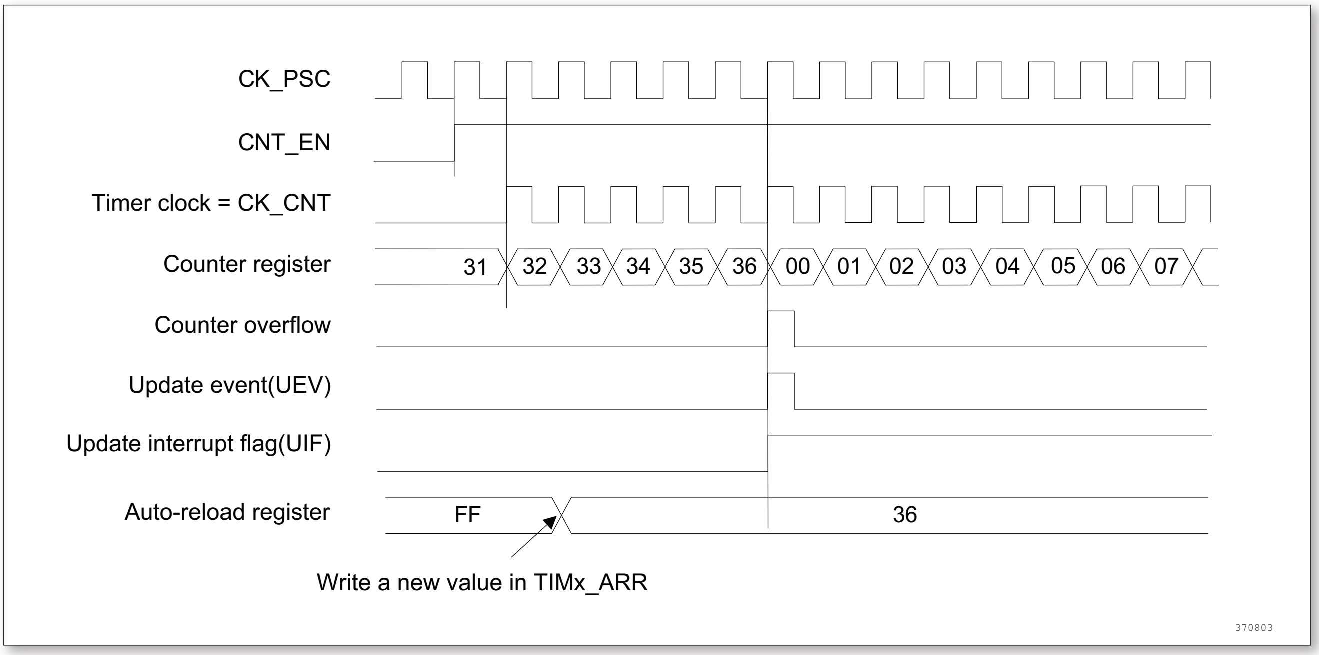


Figure 133. Counter Timing Diagram, Update Event When ARPE = 0 (TiMx_ARR Not Preloaded)

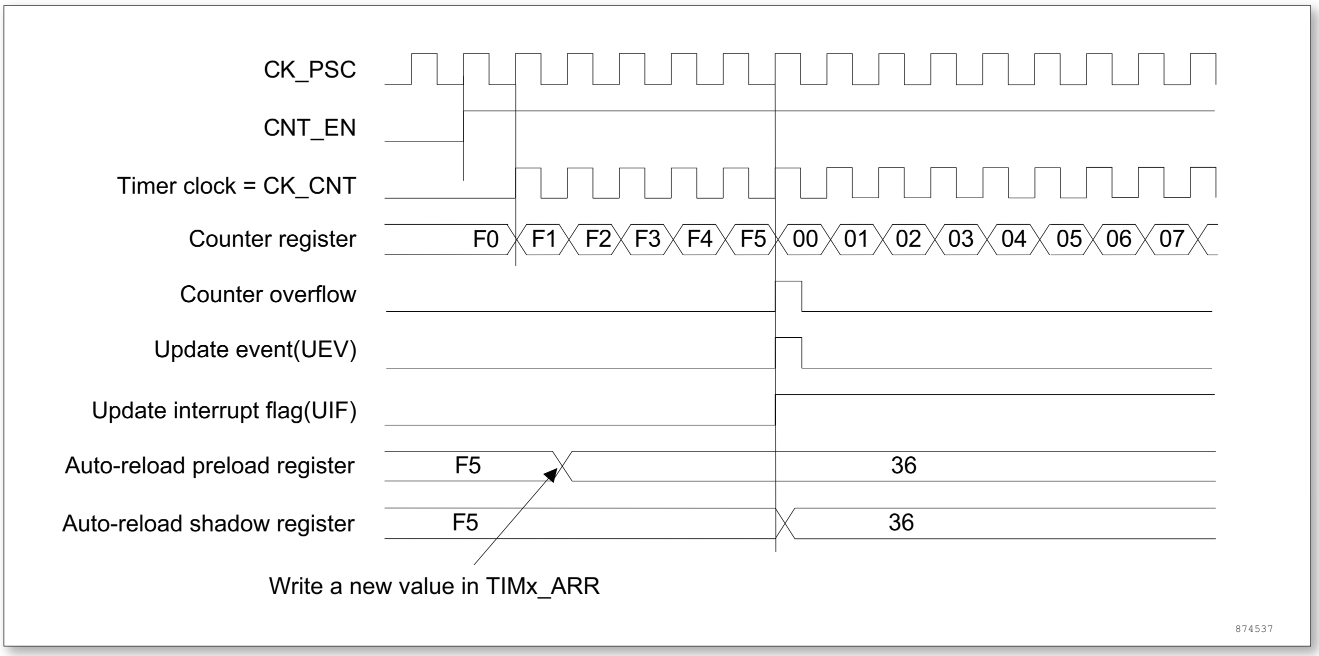

CK_PSC
CNT_EN
Timer clock = CK_CNT
Counter register
F0 F1 F2 F3 F4 F5 00 01 02 03 04 05 06 07
Counter overflow
Update event(UEV)
Update interrupt flag(UIF)
Auto-reload preload register
F5
36
Auto-reload shadow register
F5
36
Write a new value in TIMx_ARR
874537

Figure 134. Counter Timing Diagram, Update Event When ARPE = 1 (TiMx_ARR Preloaded)

Downcounting mode

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

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

- The 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.

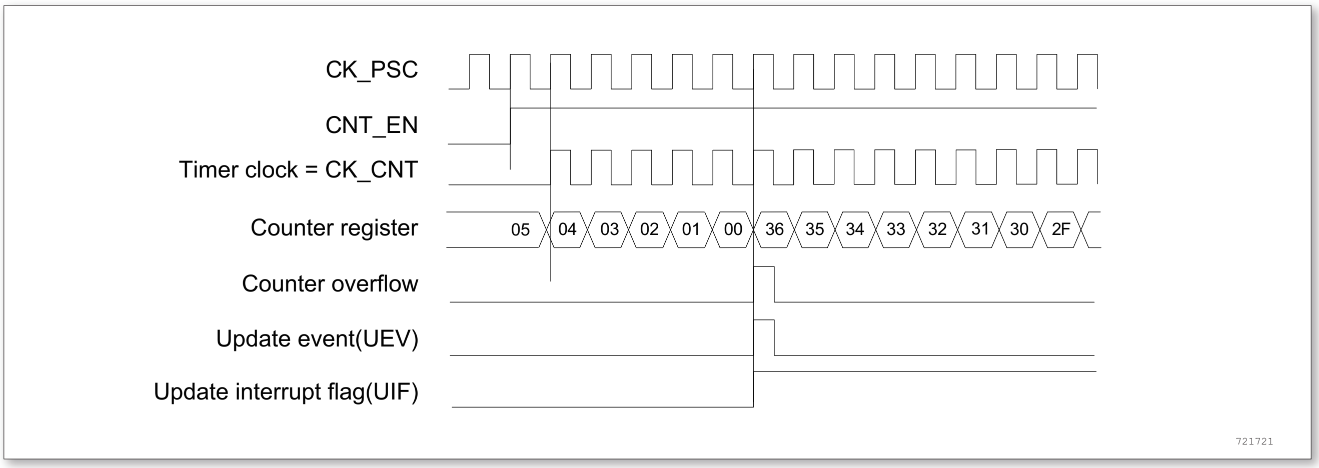


Figure 135. Counter Timing Diagram, Internal Clock Divided by 1

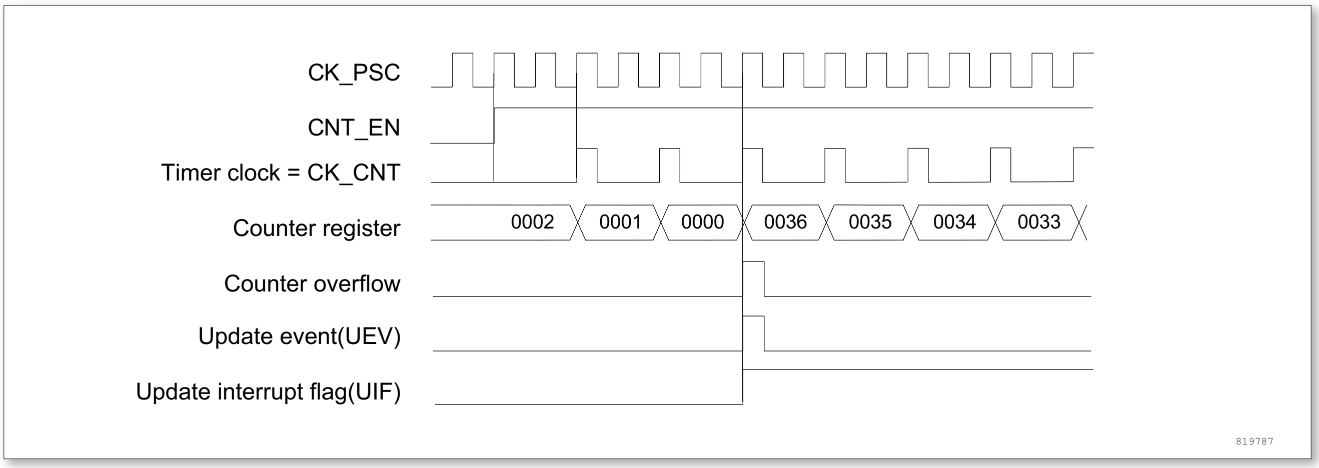


Figure 136. Counter Timing Diagram, Internal Clock Divided by 2

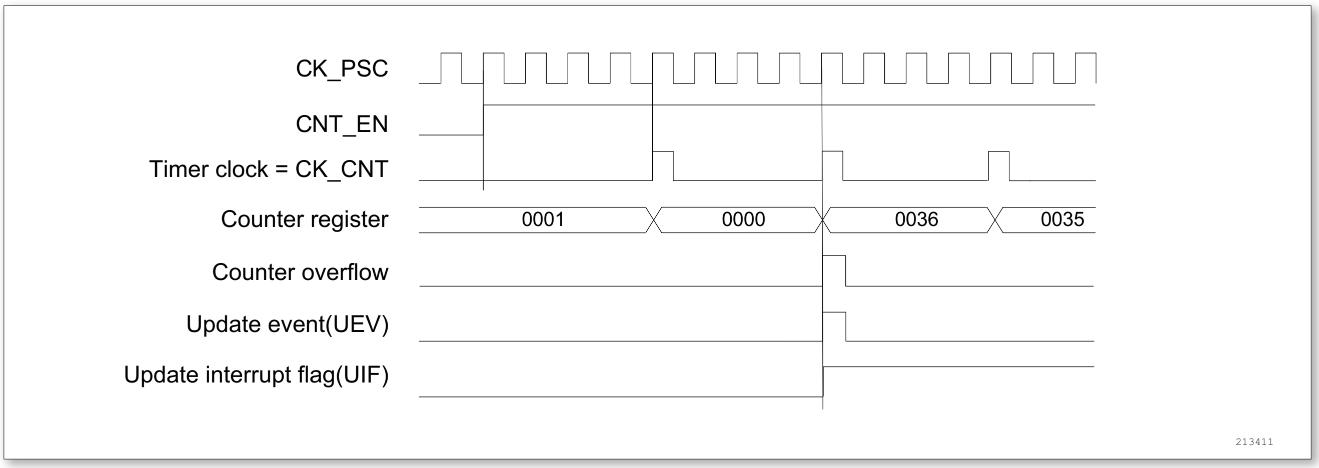


Figure 137. Counter Timing Diagram, Internal Clock Divided by 4

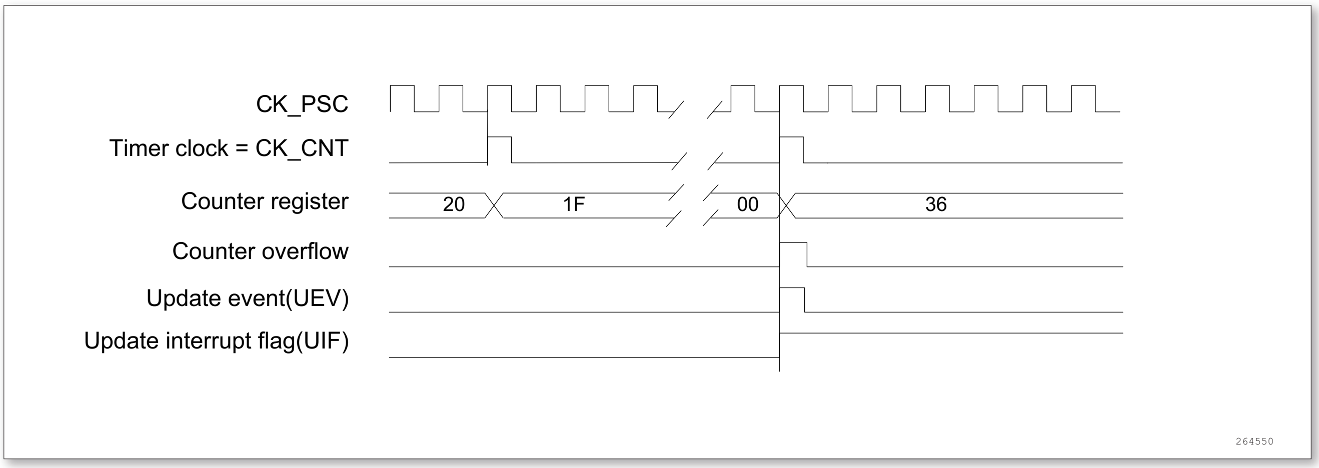


Figure 138. Counter Timing Diagram, Internal Clock Divided by N

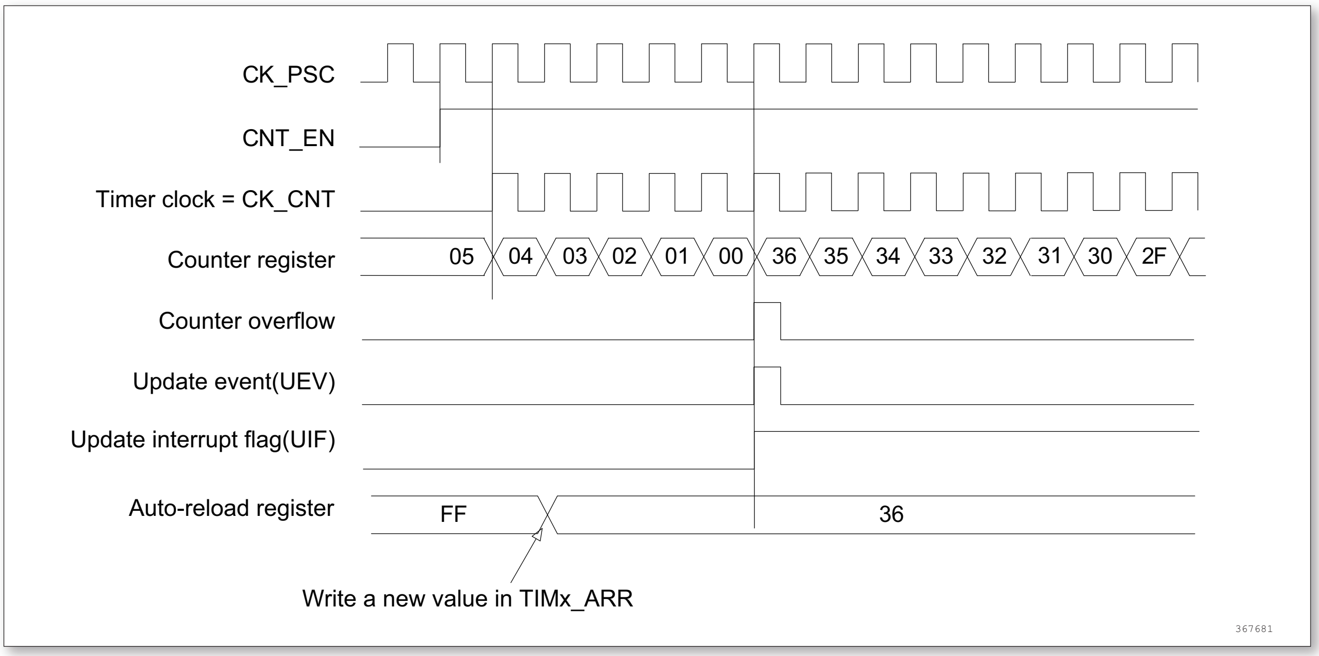


Figure 139. Counter Timing Diagram, Update Event when Repetition Counter is Not Used

Center-aligned mode (Upcounting/Downcounting))

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

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, so does the counter of the prescaler.

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

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

In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag bit(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.

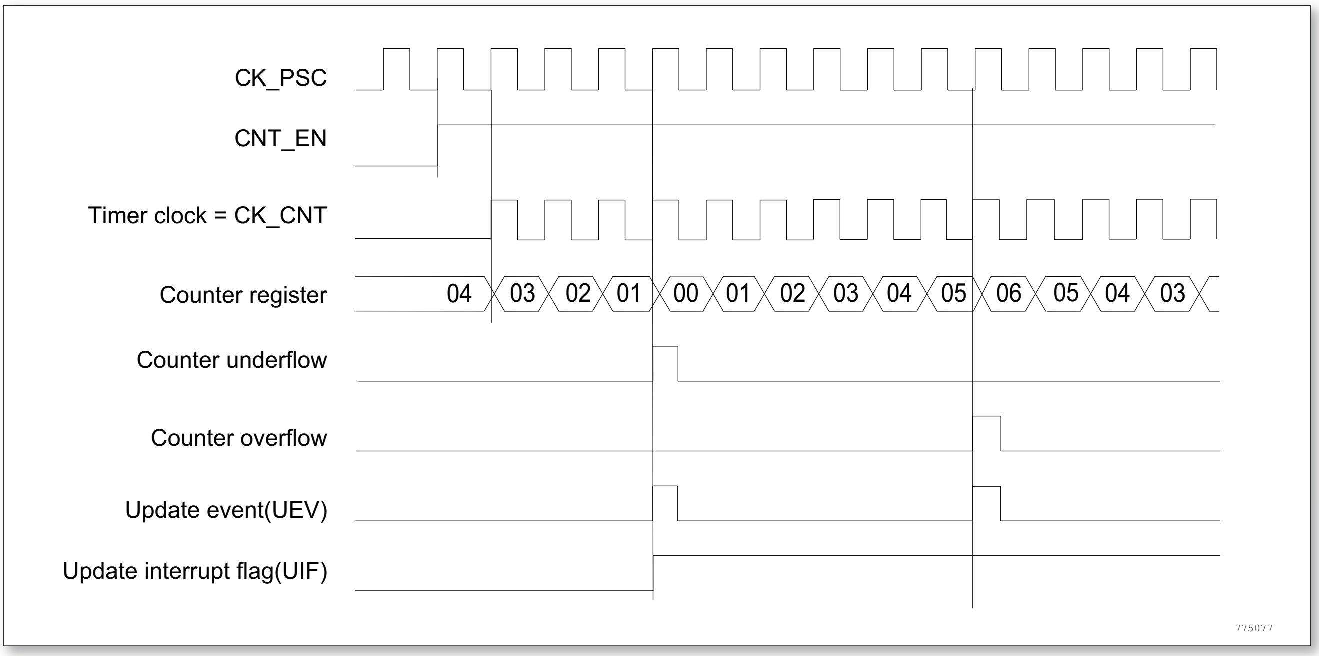


Figure 140. Counter Timing Diagram, Internal Clock Divided by 1, TIMx_ARR = 6

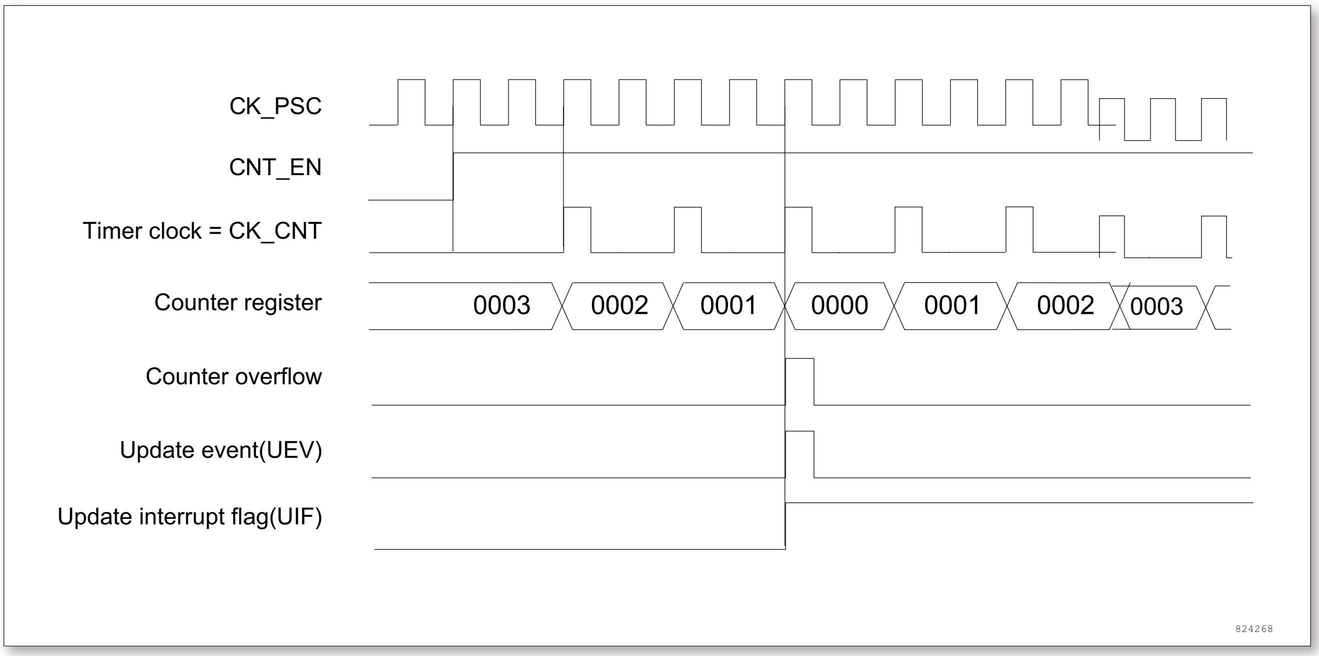


Figure 141. Counter Timing Diagram, Internal Clock Divided by 2

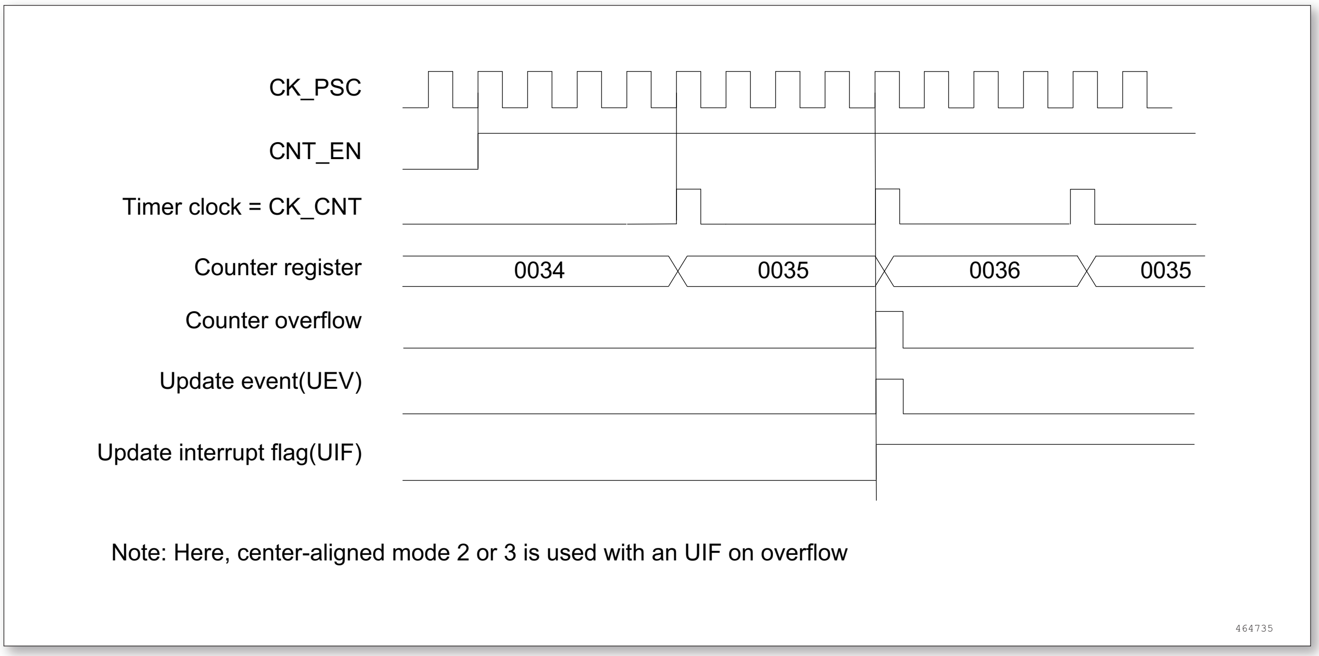


Figure 142. Counter Timing Diagram, Internal Clock Divided by 4, TIMx_ARR = 0x36

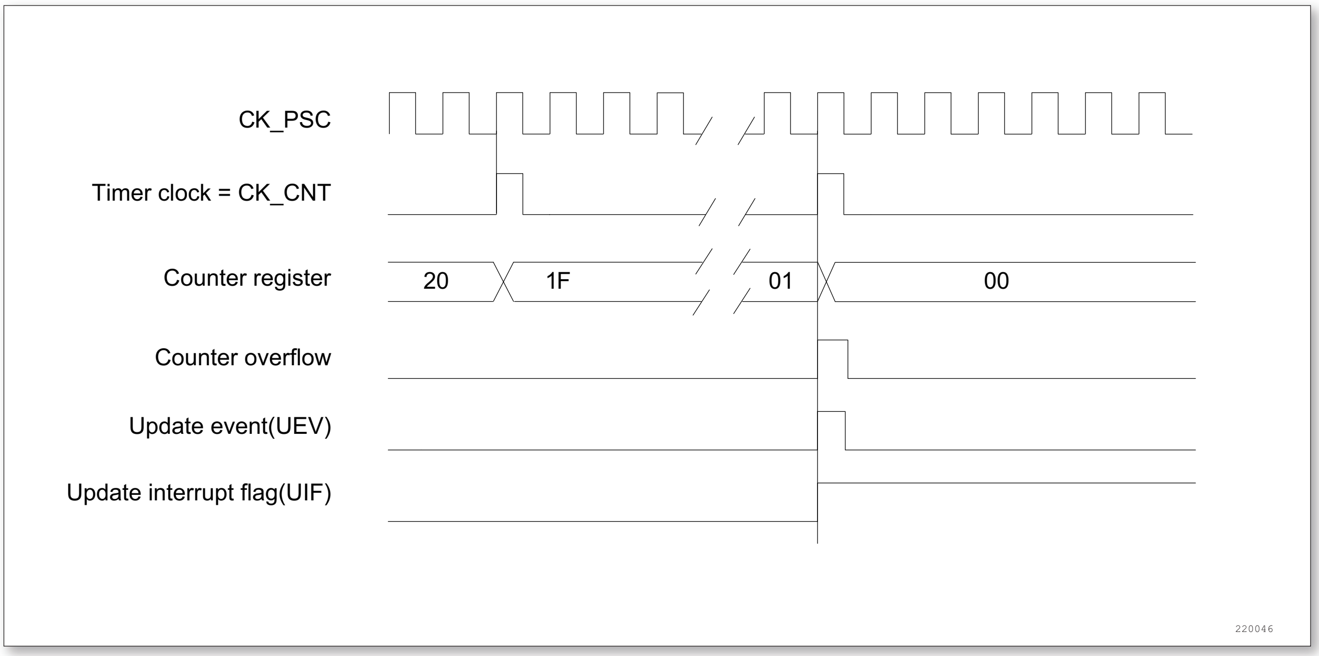


Figure 143. Counter Timing Diagram, Internal Clock Divided by N

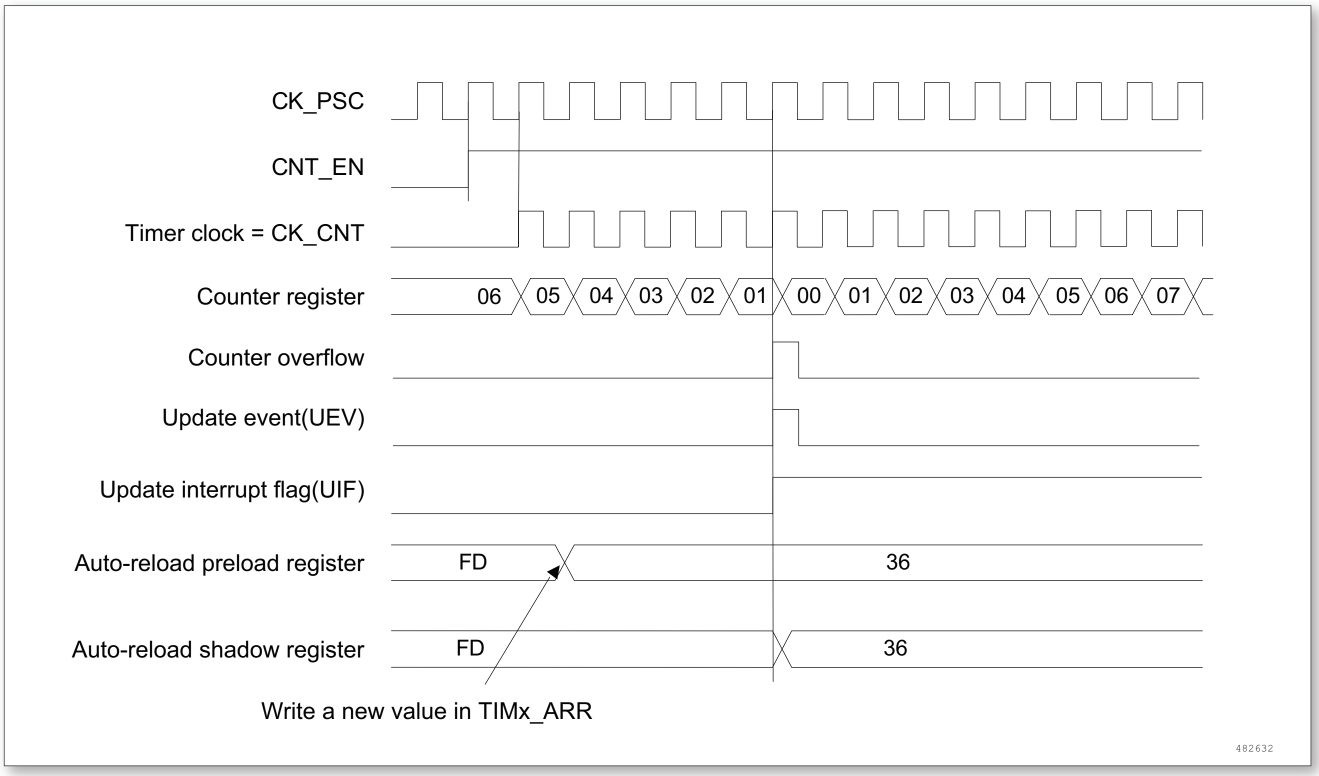


Figure 144. Counter Timing Diagram, Update Event with ARPE = 1(Counter Underflow)

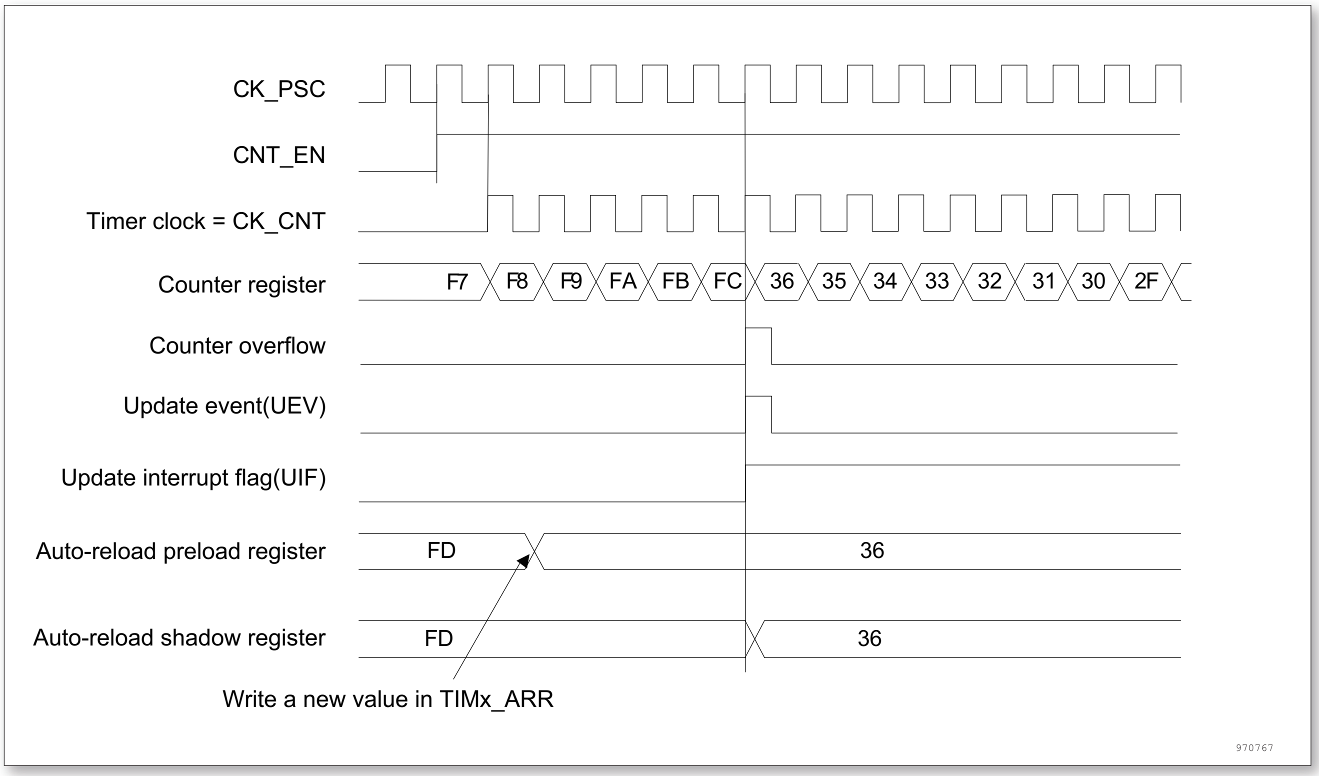


Figure 145. Counter Timing Diagram, Update Event with ARPE = 1(Counter Overflow))

13.3.3 Clock selection

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

- Internal clock (CK_INT).
- External clock mode1: external input pin (TIx).
- External clock mode2: external trigger input (ETR).
- Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, the user can configure Timer 1 to act as a prescaler for Timer 2.

Internal clock (CK_INT)

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

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

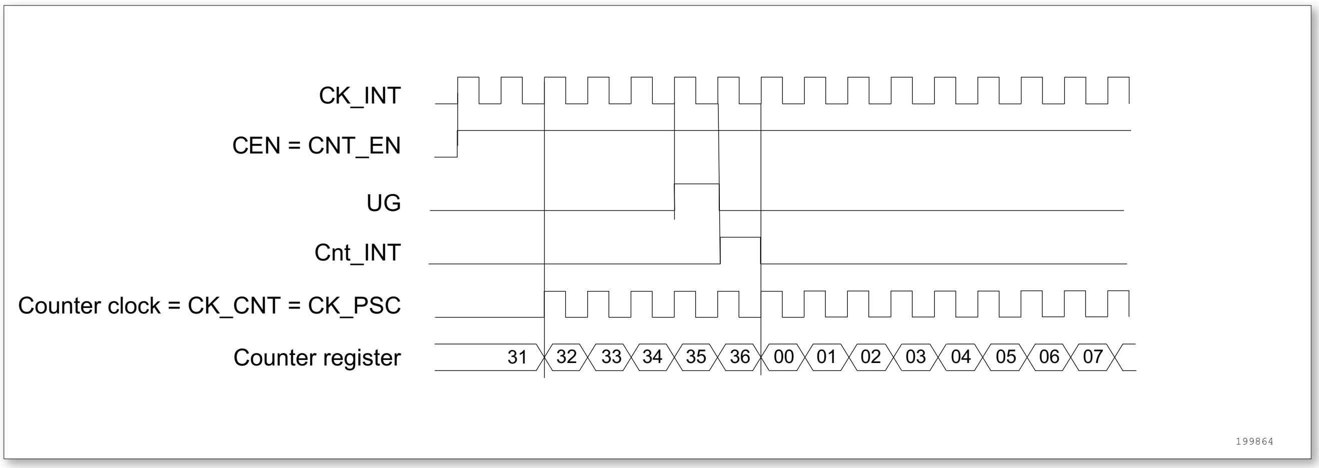


Figure 146. Control Circuit in Normal Mode, Internal Clock Divided By 1

External clock source mode 1

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

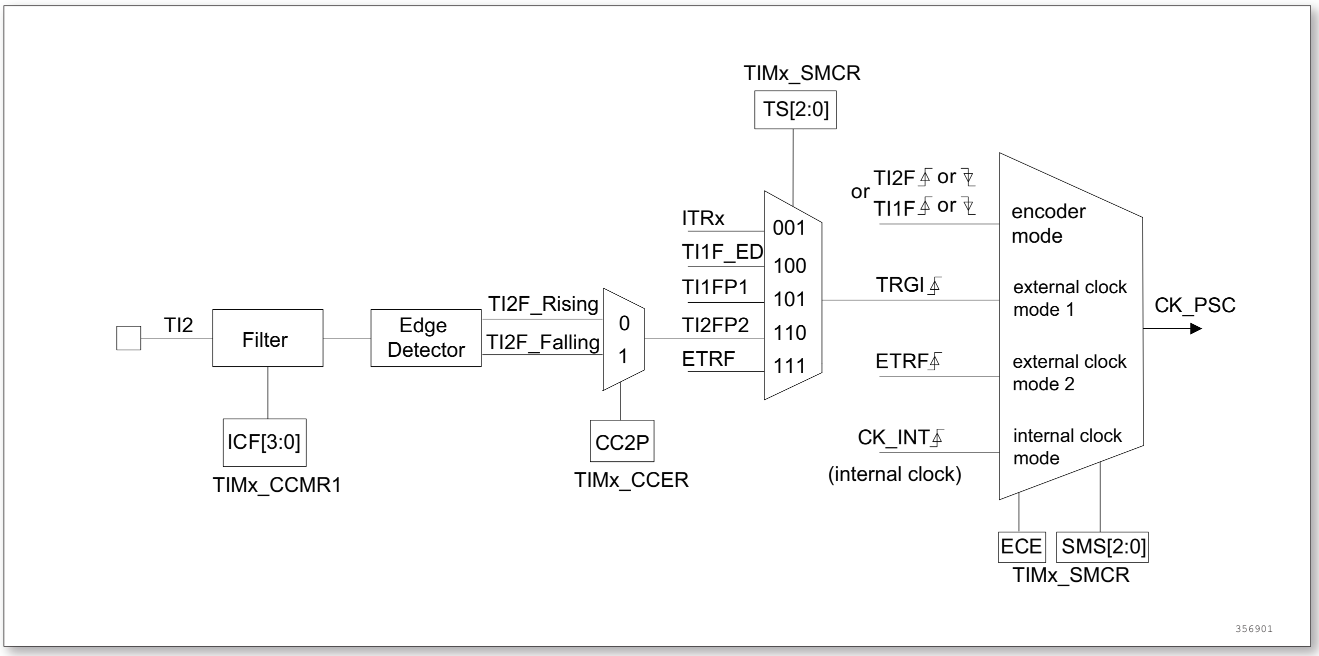


Figure 147. TI2 External Clock Connection Example

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

1. Configure channel 2 to detect rising edges on the TI2 input by writing CC2S = ‘01’ in the TIMx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMx_CCMR1 register (if no filter is needed, keep IC2F=0000). Note: The capture prescaler is not used for triggering, so there: Tno need to configure it.
3. Select rising edge polarity by writing CC2P=0 in the TIMx_CCER register.
4. Select the timer in external clock mode 1 by writing SMS=111 in the TIMx_SMCR register.

5. Select TI2 as the trigger input source by writing TS=110 in the TIMx_SMCR register.
6. 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.

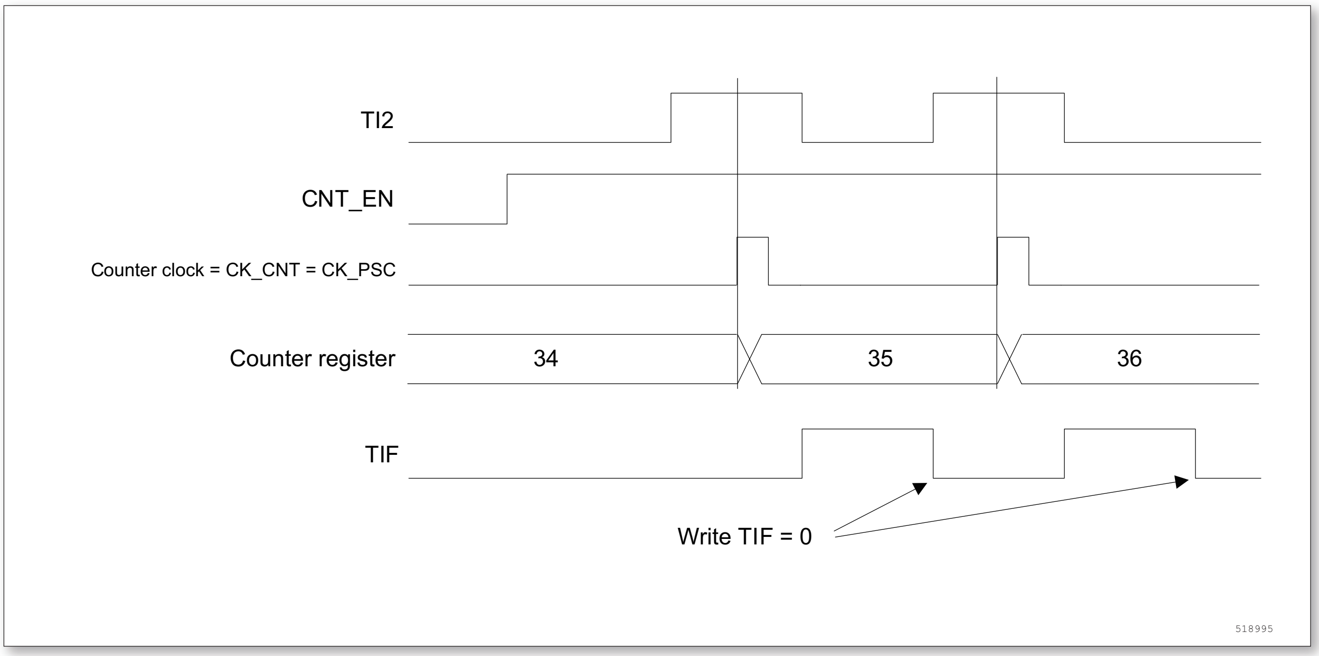


Figure 148. Control Circuit in External Clock Mode 1

External clock source mode 2

This mode is selected by writing ECE = 1 in the TIMx_SMCR register.

The counter can count at each rising or falling edge on the external trigger input ETR.

The following figure gives an overview of the external trigger input block.

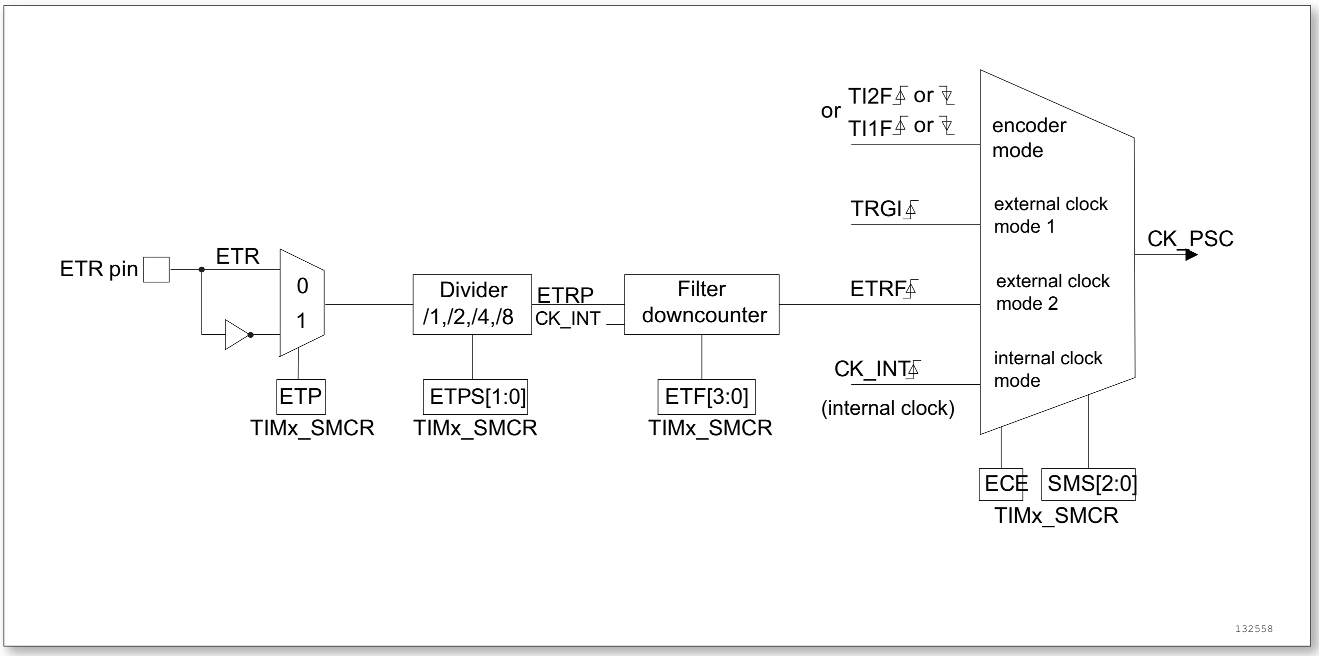


Figure 149. External Trigger Input Block

For example, to configure the upcounter to count once each 2 rising edges on ETR, use the following procedure:

- As no filter is needed in this example, write ETF [3:0]=0000 in the TIMx_SMCR register.
- Set the prescaler by writing ETPS [1:0]=01 in the TIMx_SMCR register.
- Select rising edge detection on the ETR pin by writing ETP=0 in the TIMx_SMCR register.
- Enable external clock mode 2 by writing ECE=1 in the TIMx_SMCR register.
- Enable the counter by writing CEN=1 in the TIMx_CR1 register.

The counter counts once each 2 ETR rising edges.

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

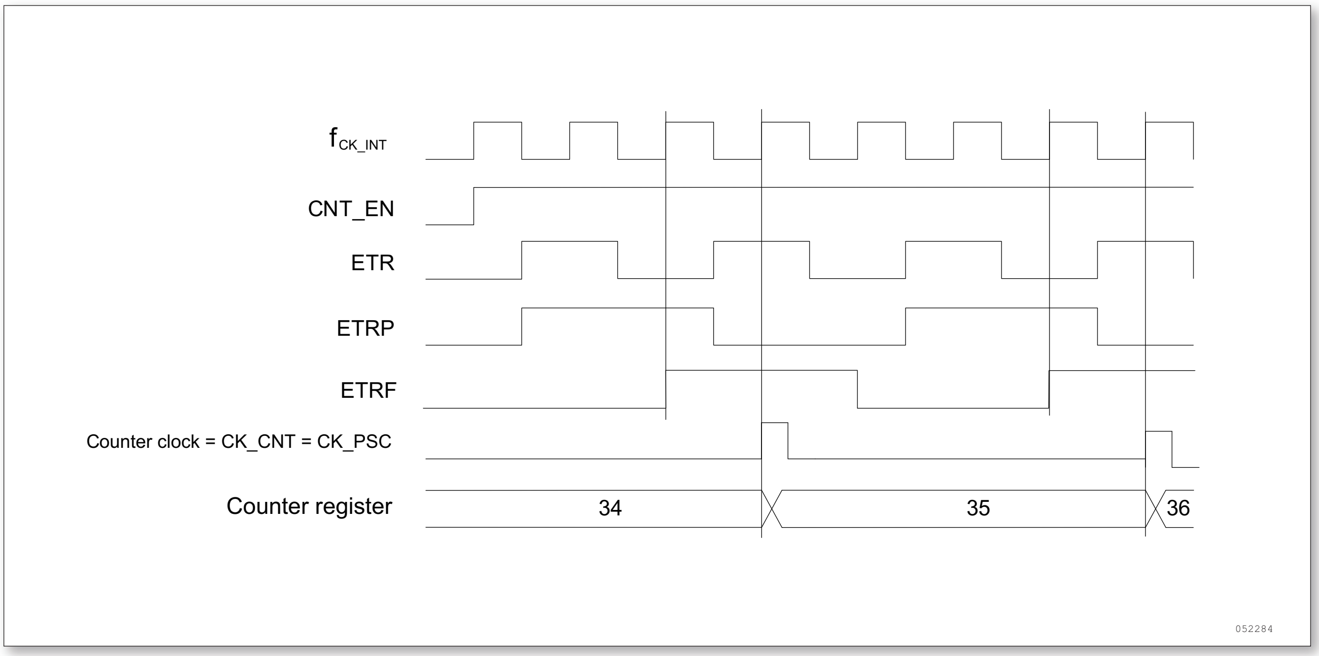


Figure 150. Control Circuit in External Clock Mode 2

13.3.4 Capture/compare channels

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

The following figures show a 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).

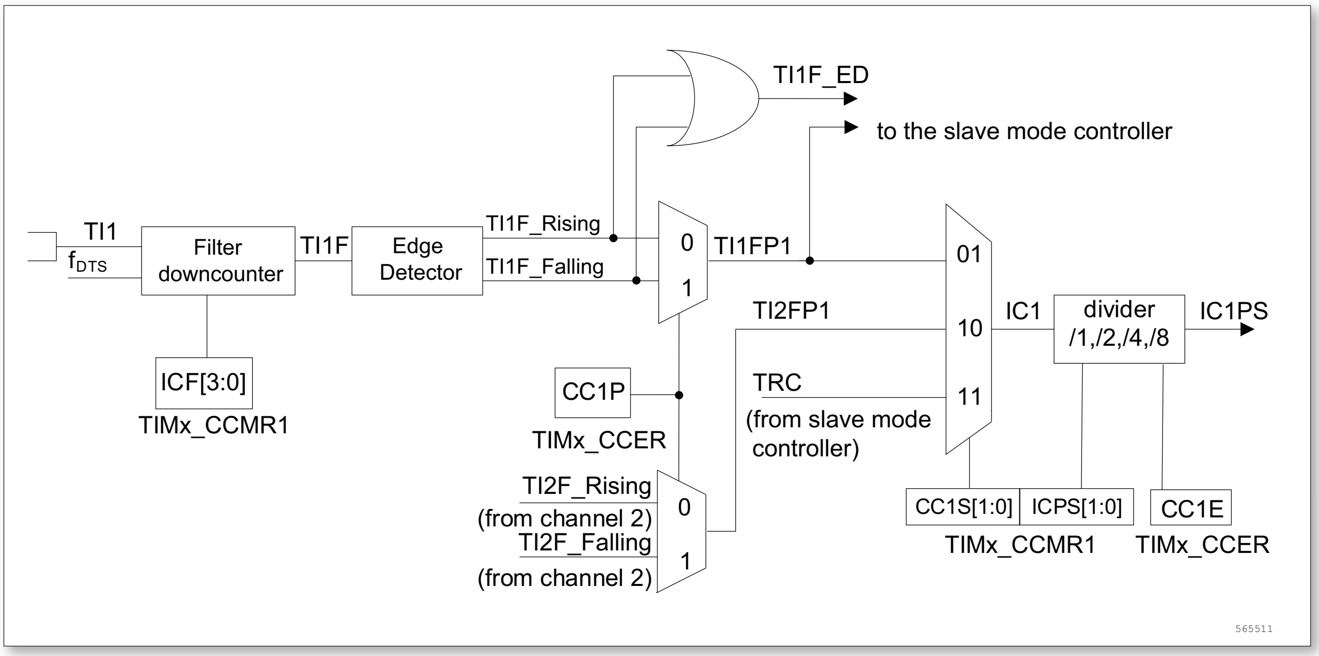


Figure 151. Capture/Compare Channel (Example: Channel 1 Input Stage)

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.

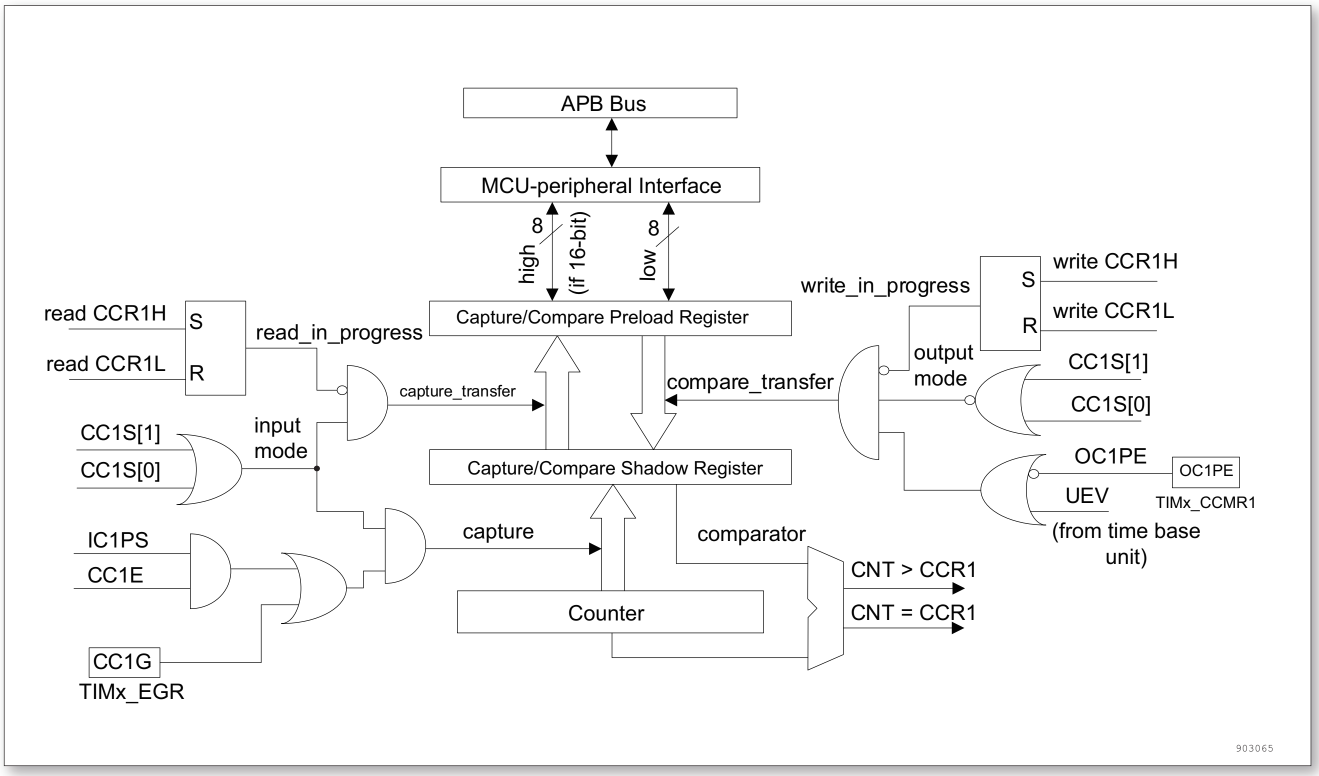


Figure 152. Capture/Compare Channel 1 Main Circuit

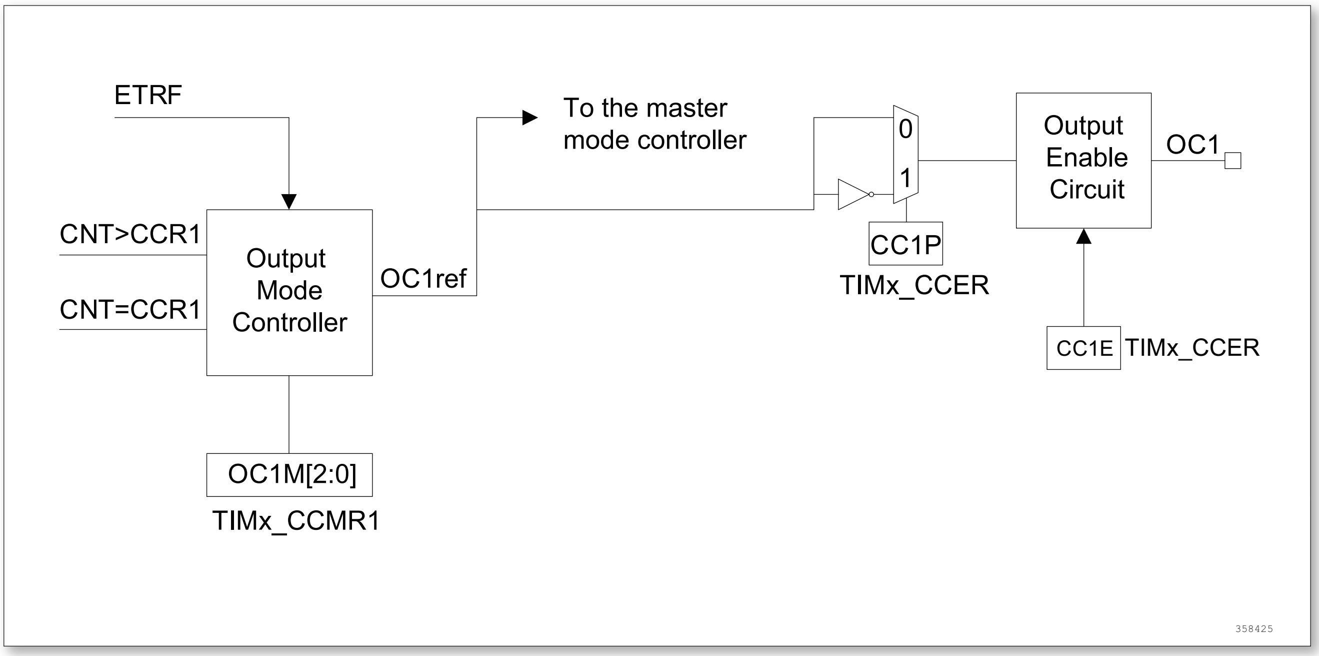


Figure 153. Output Stage of Capture/Compare Channel (Channel 1)

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.

13.3.5 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set. CCxIF can be cleared by software by writing it to ’0’ or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to ’0’.

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

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMx_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let’ s imagine that, when toggling, the input signal is not stable during at most five internal clock cycles. We must program a filter duration longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.

- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Configure the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to ’00’ in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register to ’1’.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- TIMx_CCR1 register gets the value of the counter on the active transition.
- CC1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the flag was not cleared.
- CC1OF is also set to 1.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

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

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

13.3.6 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 bits to 01 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used both for capture in TIMx_CCR1 and counter clear): write the CC1P bit to ’0’ (active on rising edge).
- Select the active input for TIMx_CCR2: write the CC2S bits to 10 in the TIMx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capture in TIMx_CCR2): write the CC2P bit to ’1’ (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIMx_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the

TIMx_SMCR register.

- Enable the captures: write the CC1E and CC2E bits to ‘1’ in the TIMx_CCER register.

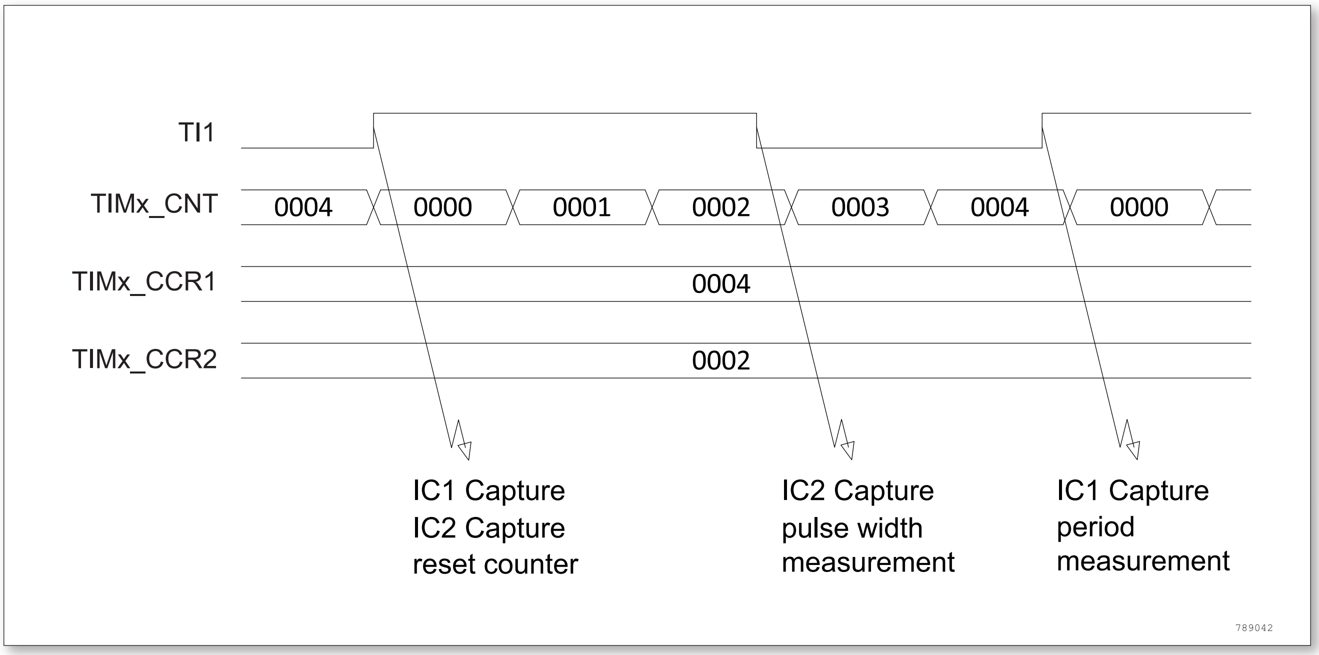


Figure 154. Output Stage of Capture/Compare Channel (Channel 1)

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.

13.3.7 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, being independent of any comparison between the output compare register and the counter.

To force an output compare signal (OCXREF/OCx) to its active level, the user just needs to write 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCXREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, in this mode, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

13.3.8 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCXM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag bit in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIMx_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register.

In output compare mode, the update event UEV has no effect on OCxREF and OCx output.

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

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIMx_ARR and TIMx_CCRx registers.
3. Set the CCxIE and/or CCxDE bits if an interrupt and/or a DMA request is to be generated.
4. Select the output mode. For example, the user must write OCxM=011, OCxPE=0, CCxP=0 and CCxE=1 to toggle OCx output pin when CNT matches CCRx, CCRx preload is not used, OCx is enabled and active high.
5. Enable the counter by setting the CEN bit in the 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.

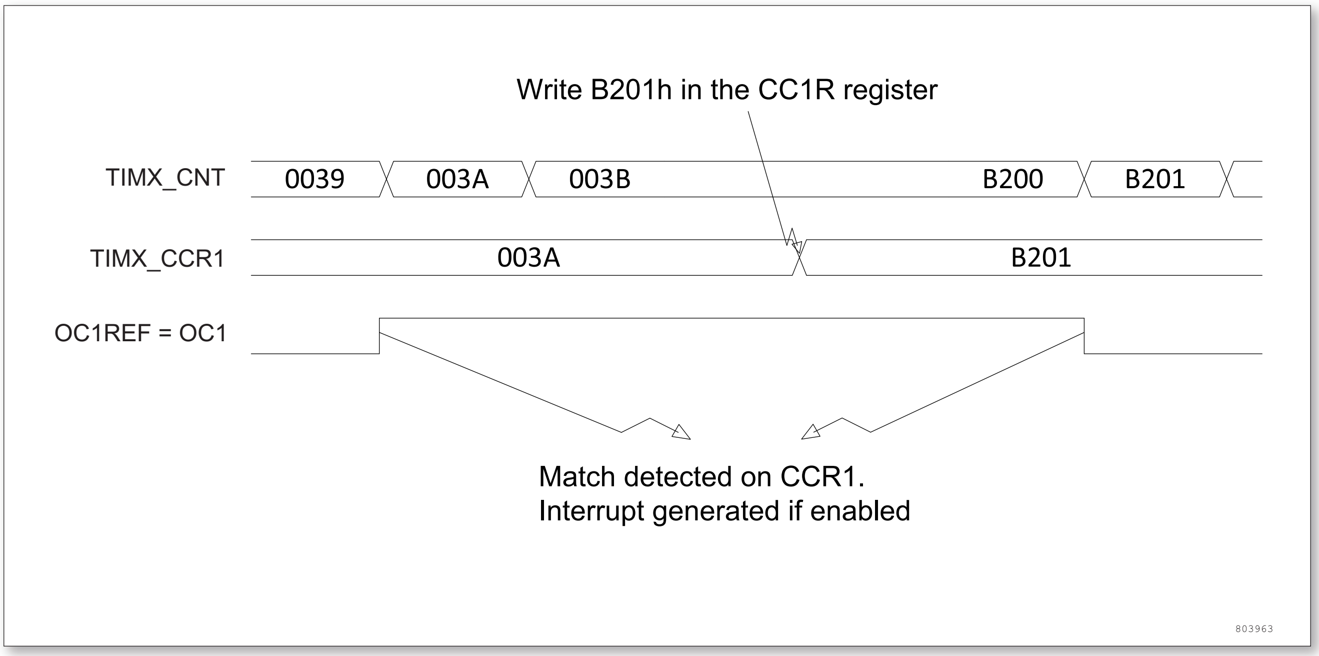


Figure 155. Output Compare Mode (Toggle OC1)

13.3.9 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 bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMx_CR1 register.

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

OCx polarity is software programmable using the CCxP bit in the TIMx_CCER register. It can be programmed as active high or active low. OCx output is enabled by the CCxE bit in the TIMx_CCER register. Refer to the TIMx_CCERx register description for more details.

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). However, to comply with the OCREF_CLR (OCxREF can be cleared by an external event through the ETR signal until the next PWM period), the OCxREF signal is asserted only:

- When the result of the comparison changes
- When the output compare mode (OCxM bits in TIMx_CCMRx register) switches from the period), the OC enabled by the CCxE bit in the TIMx_CCER register. Refer to the

TIMx_CCERx register

This forces the PWM by software while the timer is running. The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low.

In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT < TIMx_CCRx, otherwise it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxRef is held at ‘0’. Figure 61 shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

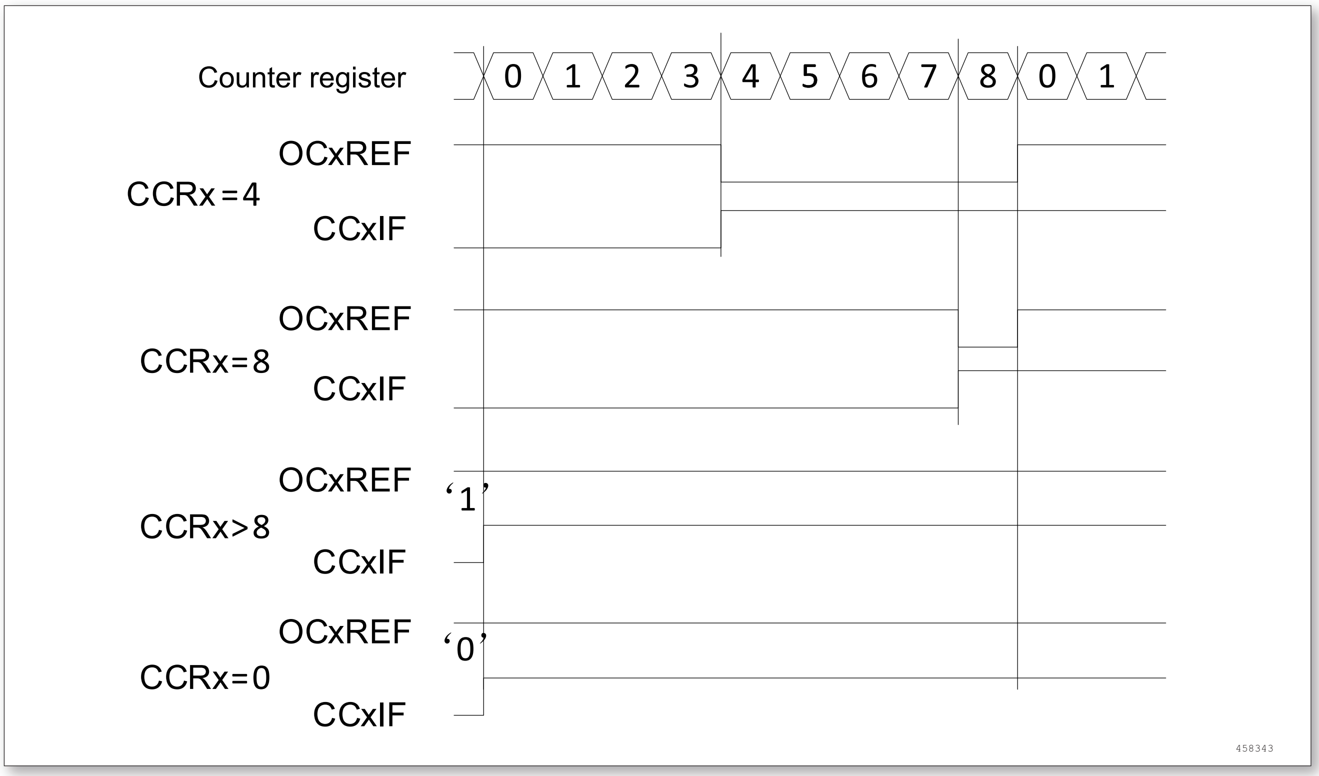


Figure 156. Edge-aligned PWM Waveforms (ARR = 8)

Downcounting configuration

Downcounting is active when DIR bit in TIMx_CR1 register is high.

In PWM mode 1, the reference signal OCxRef is low as long as TIMx_CNT > TIMx_CCRx, otherwise it becomes high. If the compare value in TIMx_CCRx is greater than the auto-reload value in TIMx_ARR, then OCxREF is held at ‘1’. 0% PWM is not possible in this mode.

PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIMx_CR1 register are different from ‘00’ (all the remaining configurations having the same effect on the OCxRef/OCx signals).

The compare flag is set when the counter counts up, when it counts down or 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 section 11.3.2 Center-aligned Mode.

The following figure 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.

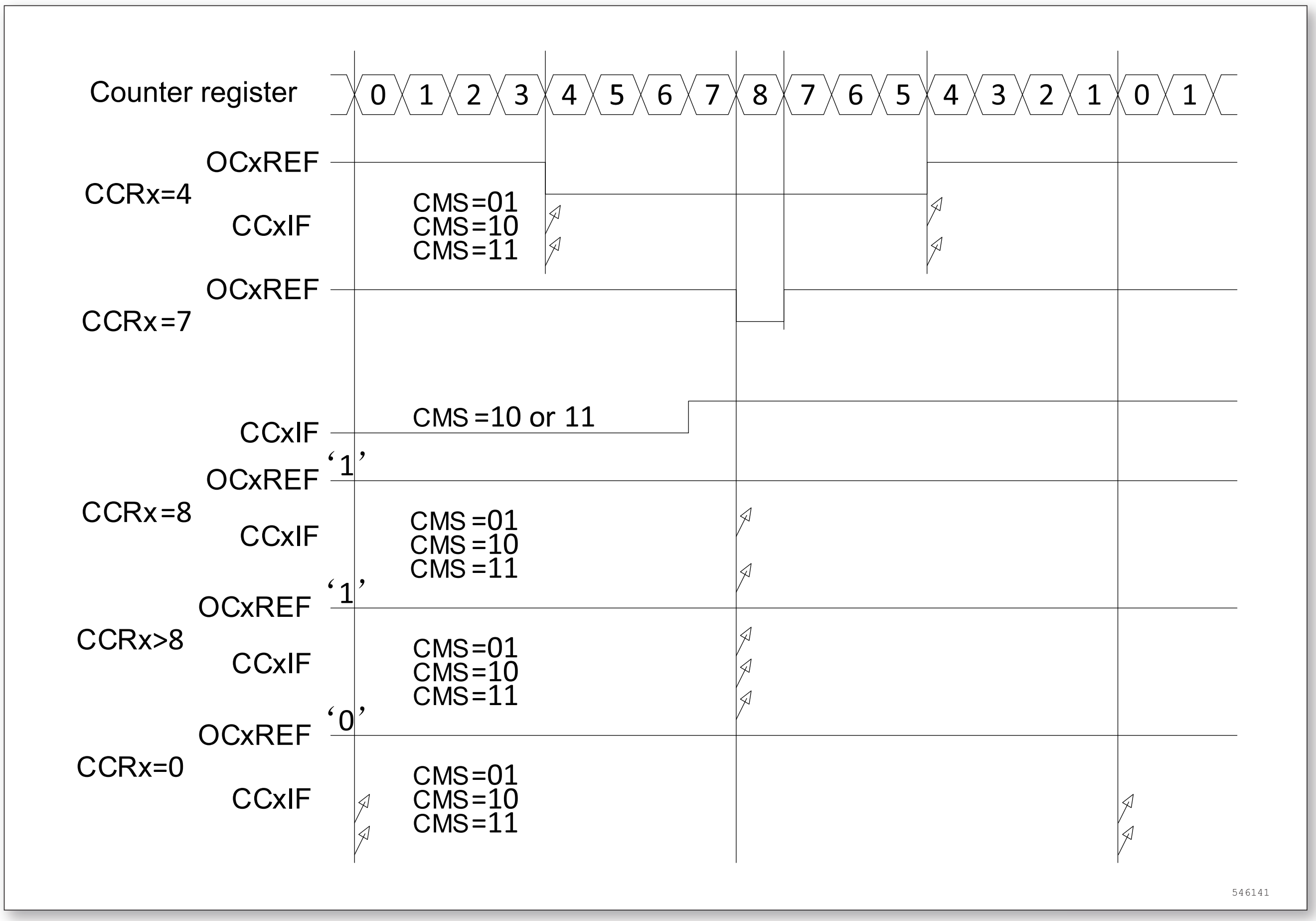


Figure 157. Center-aligned PWM Waveforms (ARR = 8)

Hints in center-aligned mode:

- When starting in center-aligned mode, the current up-down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIMx_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
- Writing to the counter while running in center-aligned mode is not recommended as it can lead to unexpected results. In particular:
  - The direction is not updated if the user writes a value in the counter greater than the auto-reload value (TIMx_CNT>TIMx_ARR). For example, if the counter was counting up, it will continue to count up.

   – The direction is updated if the user writes 0 or write the TIMx_ARR value in the counter but no Update Event UEV is generated

- The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIMx_EGR register) just before starting the counter and not to write the counter while it is running.

13.3.10 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

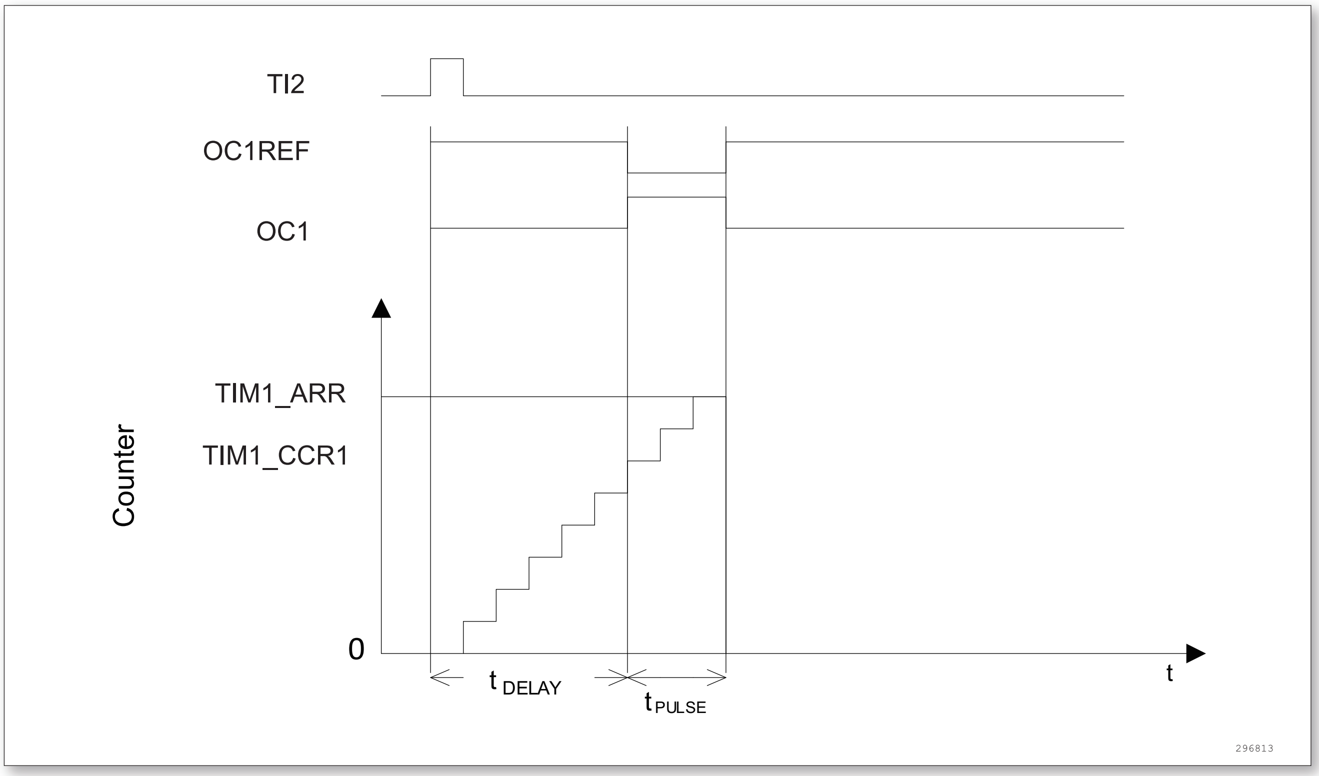


Figure 158. Example of One Pulse Mode

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

Let’s use TI2FP2 as trigger 1:

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

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

- The t_{DELAY} is defined by the value written in the TIMx_CCR1 register.
- The t_{PULSE} is defined by the difference between the auto-reload value and the compare value (TIMx_ARR - TIMx_CCR1).
- Let 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 ‘0’ 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 (Single mode), so ’1’ must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0).

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 t_{DELAY} 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.

13.3.11 Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven Low by applying a High level to the ETRF input (OCxCE enable bit of the corresponding TIMx_CCMRx register set to ‘1’). The OCxREF signal remains Low until the next update event, UEV, occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follow:

- The External Trigger Prescaler should be kept off: bits ETPS[1:0] of the TIMx_SMCR register set to ‘00’.
- The external clock mode 2 must be disabled: bit ECE of the TIMx_SMCR register set to ‘0’.
- The External Trigger Polarity (ETP) and the External Trigger Filter (ETF) can be configured according to the user needs.

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

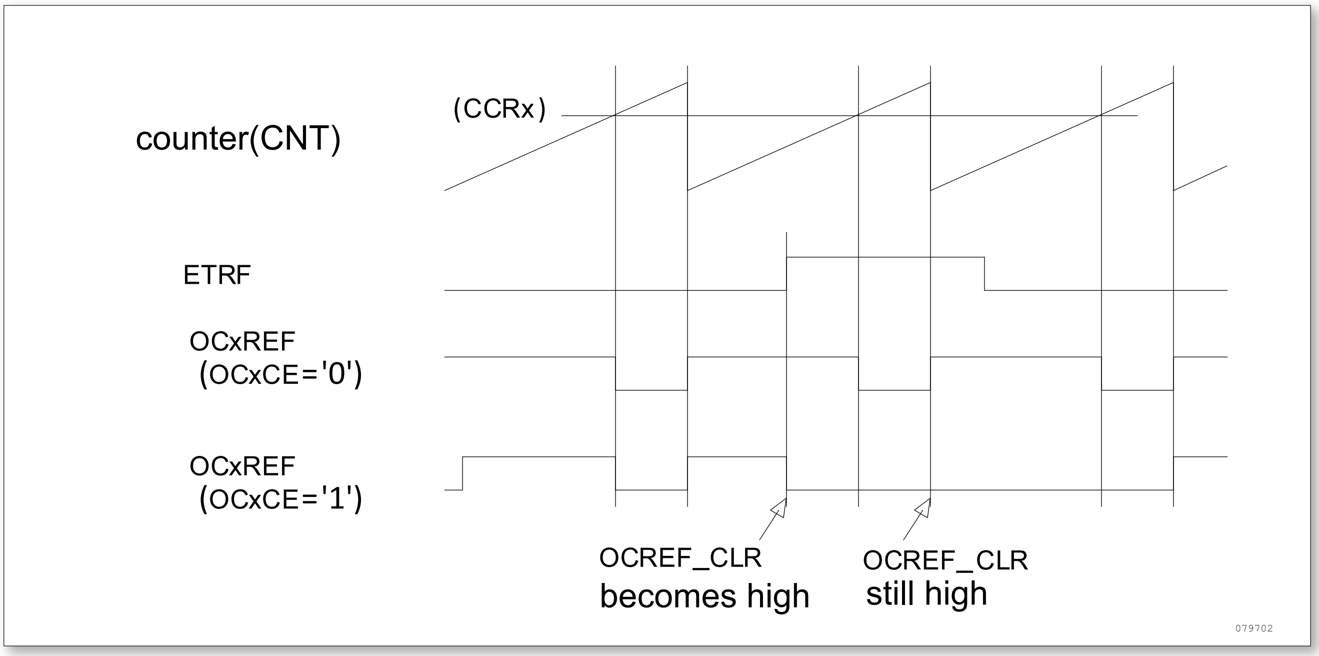


Figure 159. Clearing TIMx OCxREF

13.3.12 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. Assuming that the counter is enabled (CEN bit in TIMx_CR1 register written to ‘1) in the following table, it 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). 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 corresponds 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 45. 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 indicate 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:

- CC1S=’01’ (TIMx_CCMR1 register, TI1FP1 mapped on TI1).
- CC2S=’01’ (TIMx_CCMR2 register, TI1FP2 mapped on TI2).
- CC1P=’0’, (TIMx_CCER register, IC1FP1 non-inverted, IC1FP1=TI1).
- C2P= ’0’ (TIMx_CCER register, IC2FP2 non-inverted, IC2FP2=TI2).
- SMS= ’011’ (TIMx_SMCR register, all inputs are active on both rising and falling edges).
- CEN= ’1’ (TIMx_CR1 register, Counter enabled).

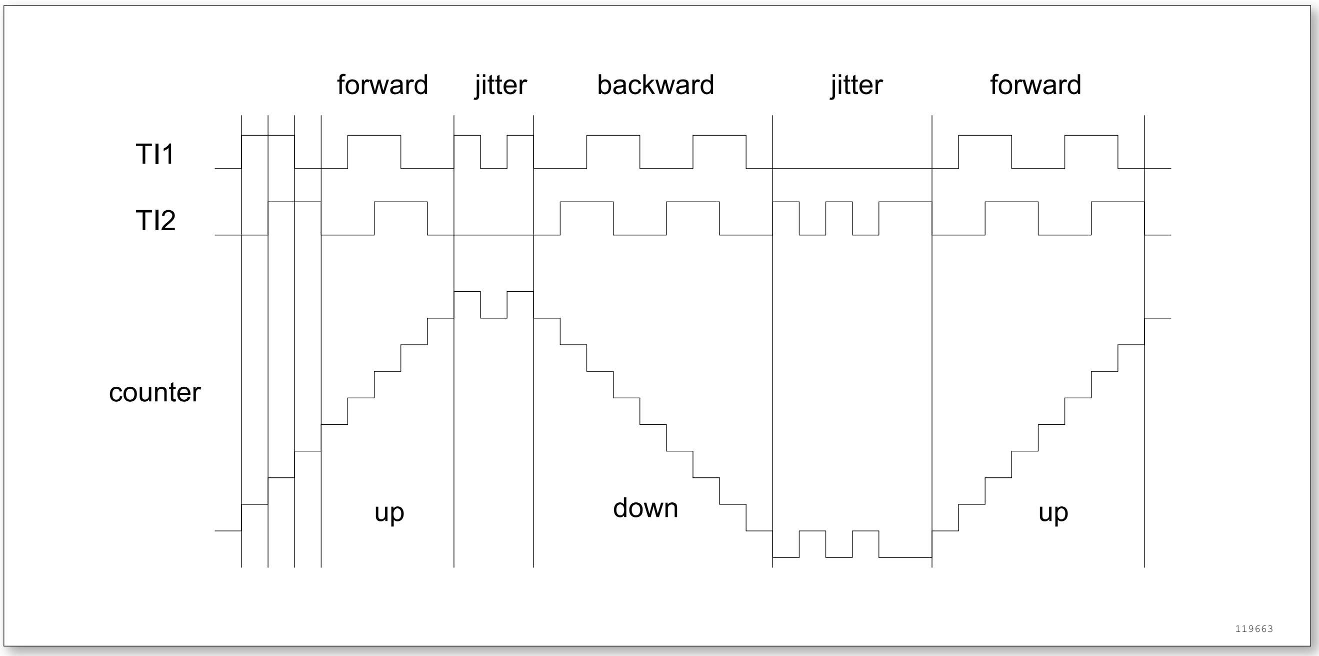


Figure 160. Example of Counter Operation in Encoder Mode

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

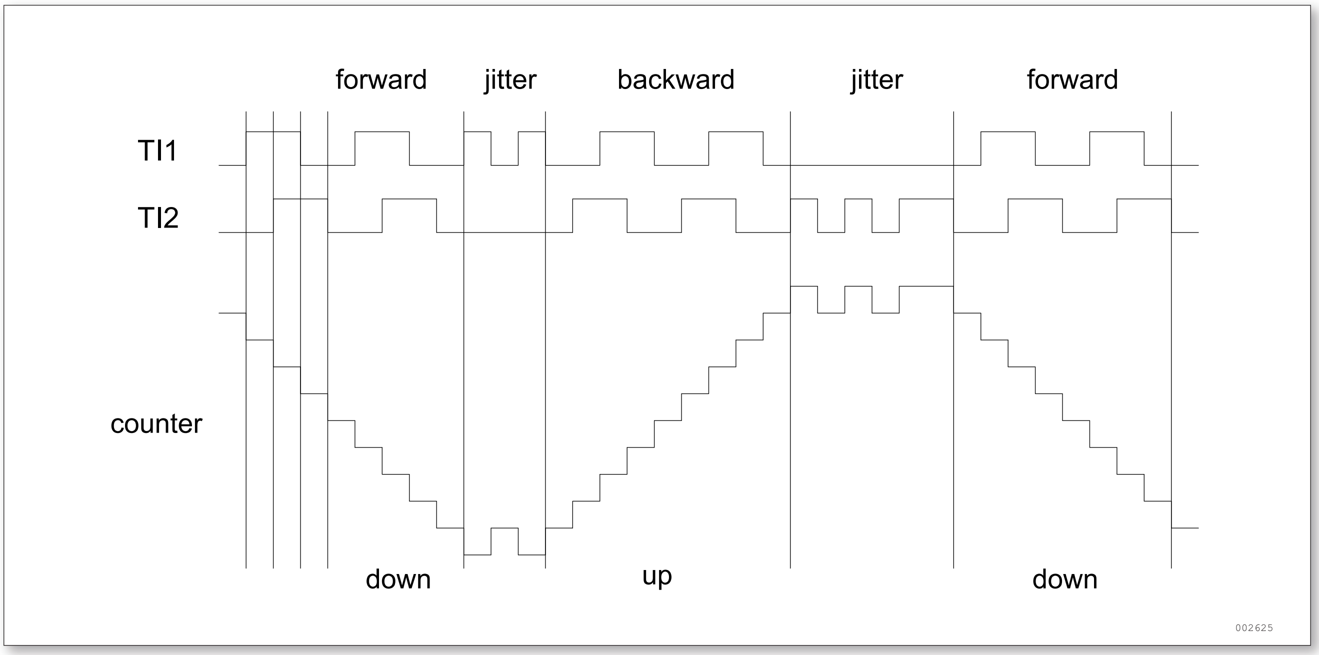


Figure 161. Example of Encoder Interface Mode with Inverted Polarity IC1FP1

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). When available, it is also possible to read its value through a DMA request generated by a real-time clock.

13.3.13 Timer input XOR function

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

The XOR output can be used with all the timer input functions such as trigger or input capture. An example of this feature used to interface Hall sensors is given in section 11.3.18.

13.3.14 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.

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 the user does not 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.
- Start the counter by writing CEN=1 in the TIMx_CR1 register.

The counter starts counting on the internal clock, then behaves normally until TI1 rising edge. When TI1 rises, the counter is cleared and restarts from 0. In the meantime, the trigger flag is set (TIF bit in the TIMx_SR register) and an interrupt request, or a DMA request can be sent if enabled depending on the TIE (interrupt enable) and TDE (DMA enable) bits in TIMx_DIER register.

The following figure shows this behavior when the auto-reload register TIMx_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter is due to the resynchronization circuit on TI1 input.

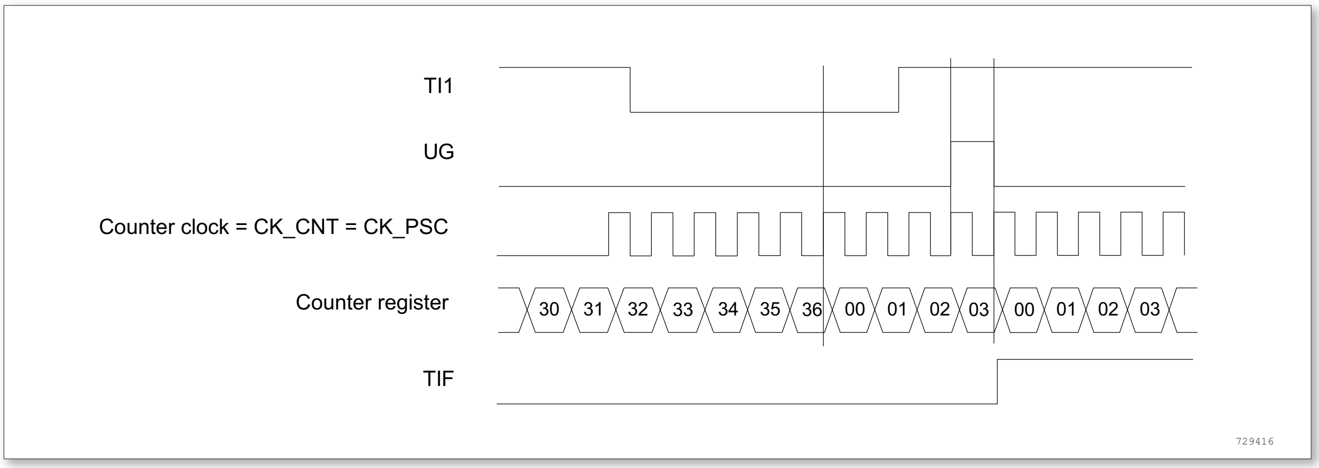


Figure 162. Control Circuit in Reset Mode

Slave mode: Gated mode

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

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

- Configure the channel 1 to detect low level on TI1. Configure the input filter duration (in this example, we don’t need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S = 01 in the 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.
- Start the counter by writing CEN=1 in the TIMx_CR1 register. In gated mode, the counter doesn’t start if CEN=0, whatever is the trigger input level

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIMx_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter is due to the resynchronization circuit on TI1 input.

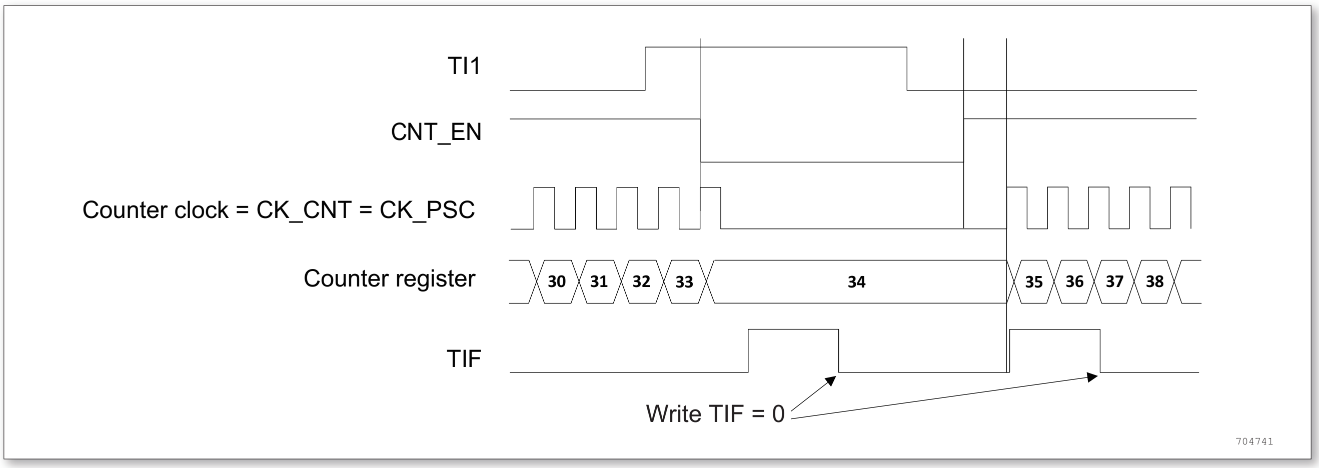


Figure 163. Control Circuit in Gated Mode

Slave mode: Trigger mode

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

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

- Configure the channel 2 to detect rising edges on TI2. Configure the input filter duration (in this example, we don’t need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there’s no need to configure it. The CC2S bits are only configured to select CC2P=1 in TIMx_CCER register, so as 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 stop of the counter is due to the resynchronization circuit on TI2 input.

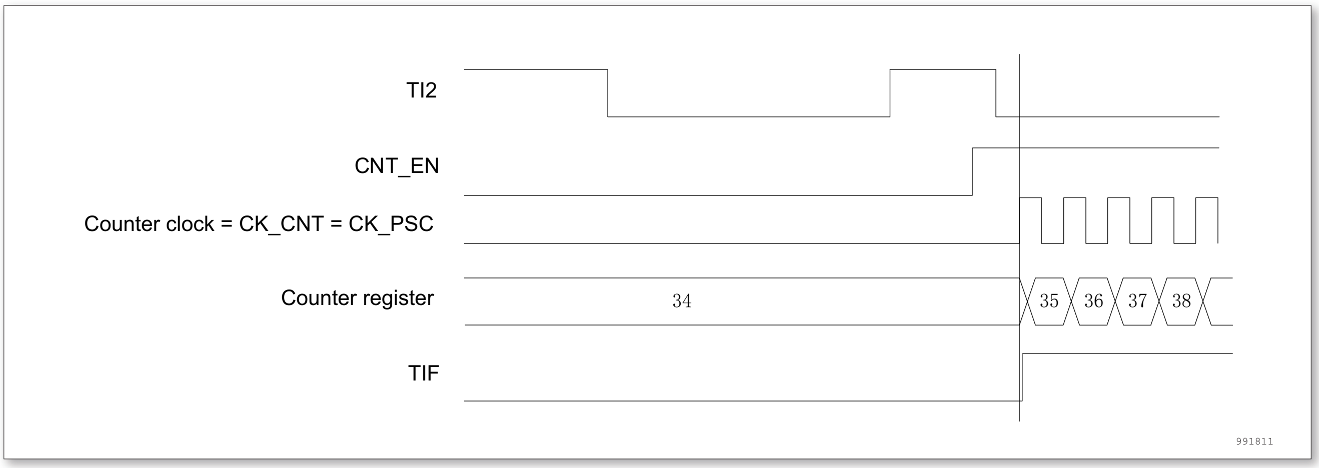


Figure 164. Control Circuit in Trigger Mode

Slave mode: External clock mode 2 + trigger mode

The external clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input (in reset mode, gated mode or trigger mode). It is recommended not to select ETR as TRGI through the TS bits of TIMx_SMCR register.

In the following example, the upcounter is incremented at each rising edge of the ETR signal as soon as a rising edge of TI1 occurs:

- Configure the external trigger input circuit by programming the TIMx_SMCR register as follows:
  - ETF = 0000: no filter
  - ETPS = 00: prescaler disabled
  - ETP = 0: detection of rising edges on ETR and ECE=1 to enable the external clock mode 2.

- Configure the channel 1 as follows, to detect rising edges on T1:
  - IC1F=0000: no filter.
  - The capture prescaler is not used for triggering and does not need to be configured.
  - CC1S=01 in TIMx_CCMR1 register to select only the input capture source.
  - CC1P=0 in TIMx_CCER register to validate the polarity (and detect rising edge only).
- Configure the timer in trigger mode by writing SMS=110 in TIMx_SMCR register. Select TI1 as the input source by writing TS=101 in TIMx_SMCR register

A rising edge on TI1 enables the counter and sets the TIF flag. The counter then counts on ETR rising edges.

The delay between the rising edge of the ETR signal and the actual reset of the counter is due to the resynchronization circuit on ETRP input.

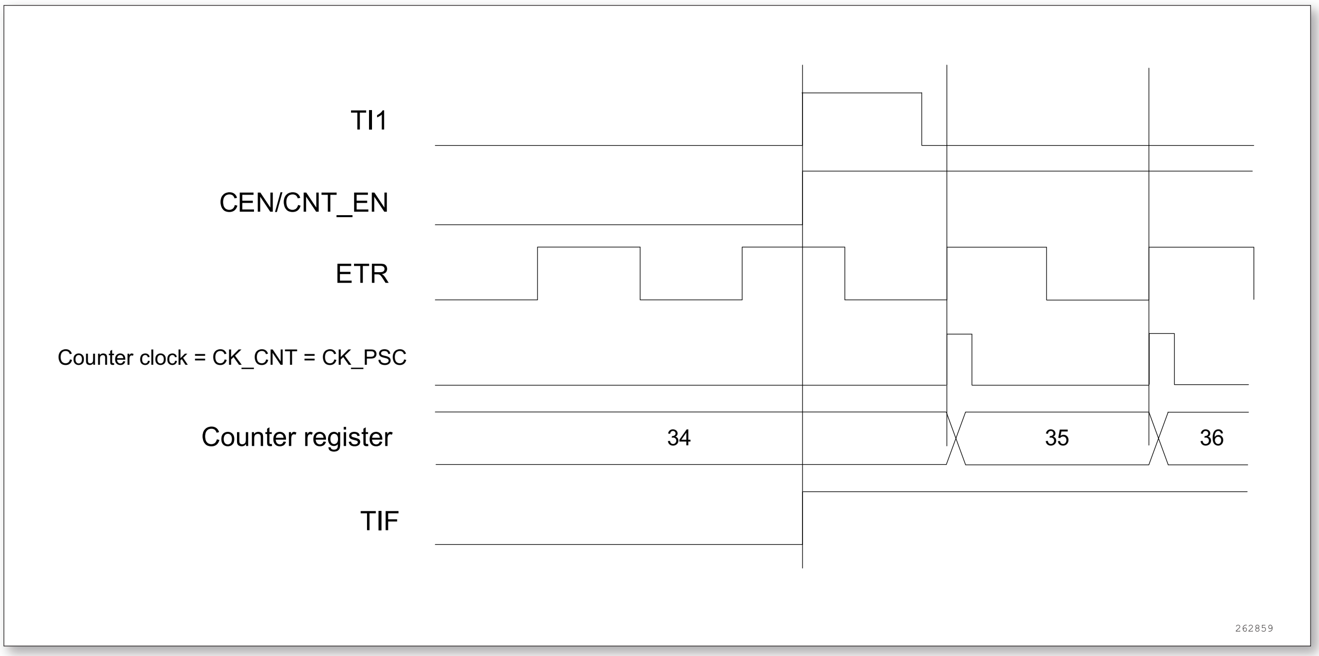


Figure 165. Control Circuit in External Clock Mode 2 + Trigger Mode

13.3.15 Timer synchronization

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

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

Using one timer as prescaler for another timer

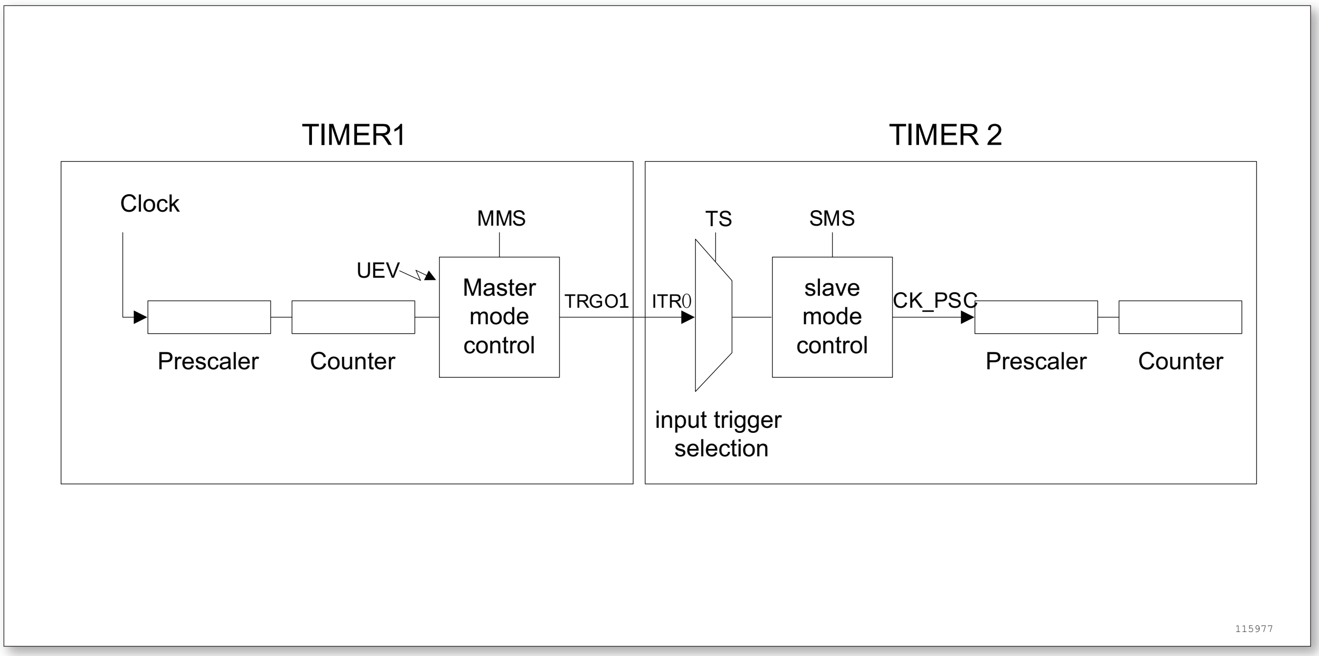


Figure 166. Master/Slave Timer Example

For example, the user can configure Timer 1 to act as a prescaler for Timer 2 (see the above figure). To do this:

- 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 2, Timer 2 must be configured in slave mode using ITR1 as internal trigger. You select this through the TS bits in the TIM2_SMCR register (writing TS=000).
- Then you put the slave mode controller in external clock mode 1 (write SMS=111 in the TIM2_SMCR register). This causes Timer 2 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 2.

Using one timer to enable another timer

In this example, we control the enable of Timer 2 with the output compare 1 of Timer 1. Refer to Figure 167 for connections. Timer 2 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to 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 2 to get the input trigger from Timer 1 (TS=001 in the TIM2_SMCR

register).

- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Enable Timer 2 by writing ‘1 in the CEN bit (TIM2_CR1 register).
- Enable Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).

Note: The counter 2 clock is not synchronized with counter 1, this mode only affects the Timer 2 counter enable signal.

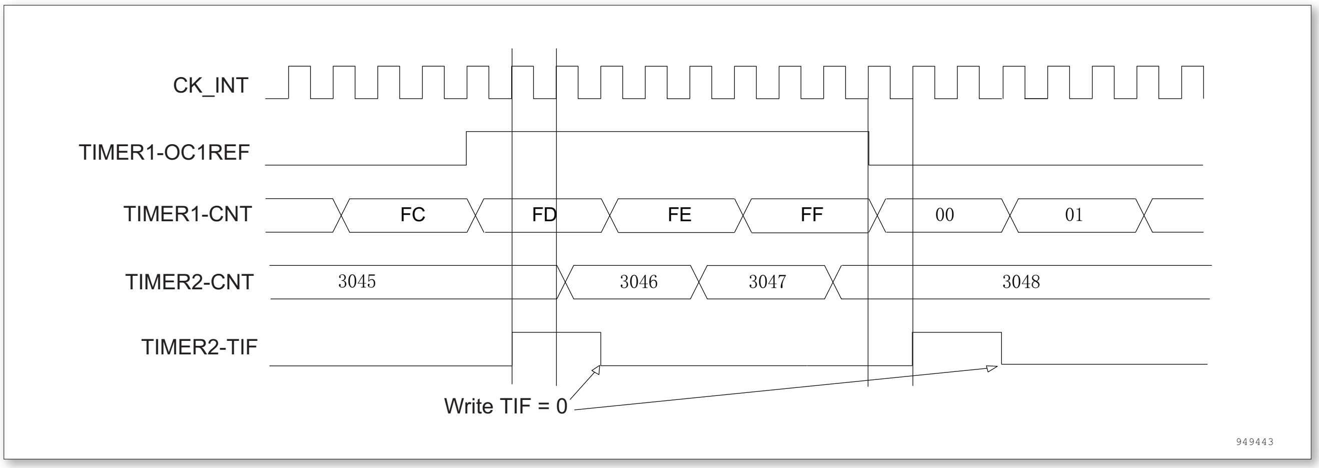


Figure 167. Gating Timer 2 with OC1REF of Timer 1

In the example in the above figure, the Timer 2 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 software using the UG bit in the TIMx_EGR registers.

In the next example, we synchronize Timer 1 and Timer 2. Timer 1 is the master and starts from 0. Timer 2 is the slave and starts from 0xE7. The prescaler ratio is the same for both timers. Timer 2 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 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Reset Timer 1 by writing ‘1 in UG bit (TIM1_EGR register).
- Reset Timer 2 by writing ‘1 in UG bit (TIM2_EGR register).
- Initialize Timer 2 to 0xE7 by writing ‘0xE7’ in the Timer 2 counter (TIM2_CNT).
- Enable Timer 2 by writing ‘1 in the CEN bit (TIM2_CR1 register).
- Start Timer 1 by writing ‘1 in the CEN bit (TIM1_CR1 register).
- Stop Timer 1 by writing ‘0 in the CEN bit (TIM1_CR1 register).

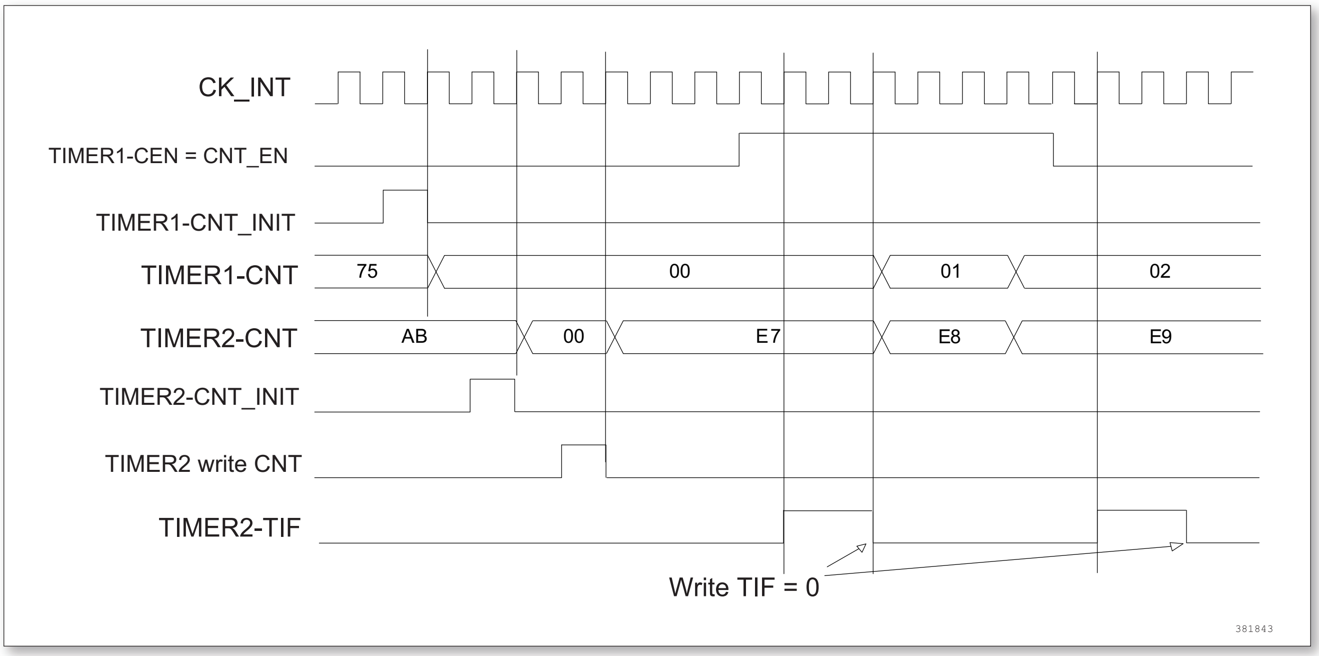


Figure 168. Gating Timer 2 with Enable of Timer 1

Using one timer to start another timer

In this example, we set the enable of Timer 2 with the update event of Timer 1. Refer to the following figure for connections. Timer 2 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 2 receives the trigger signal its CEN bit is automatically set and the counter counts until we write '0' to the CEN bit in the TIM2_CR1 register. Both counter clock frequencies are divided by 3 by the prescaler compared to CK_INT ($f_{CK\_CNT} = f_{CK\_INT/3}$).

- Configure Timer 1 master mode to send its Update Event (UEV) as trigger output (MMS=010 in the TIM1_CR2 register).
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in TIM2_SMCR register).
- Start Timer 1 by writing '1' in the CEN bit (TIM1_CR1 register)

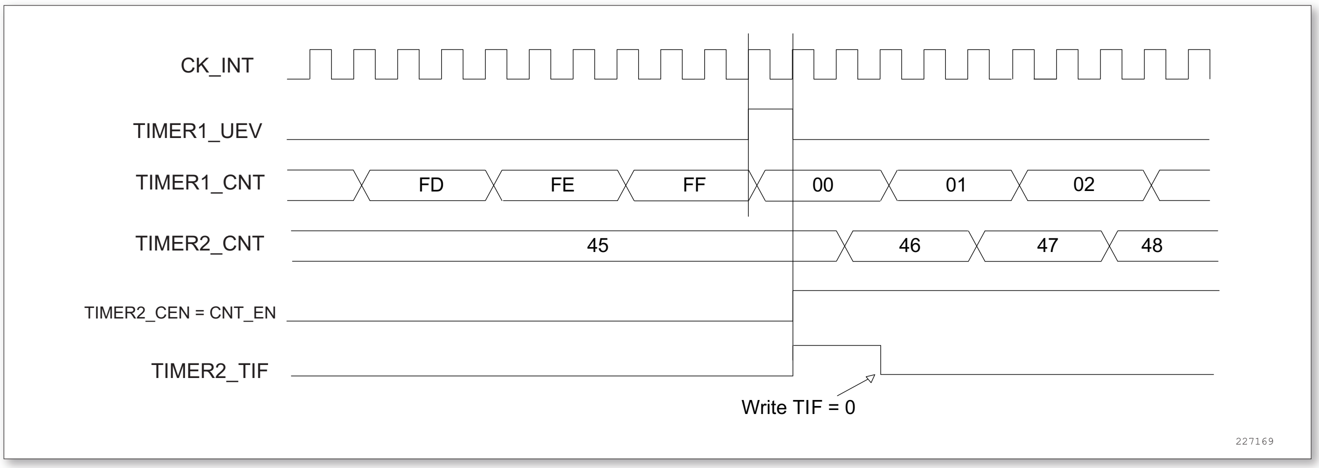


Figure 169. Triggering Timer 2 with Update of Timer 1

As in the previous example, the user can initialize both counters before starting counting. The following figure shows the behavior with the same configuration as '0' but in trigger mode instead of gated mode (SMS=110 in the TIM2_SMCR register).

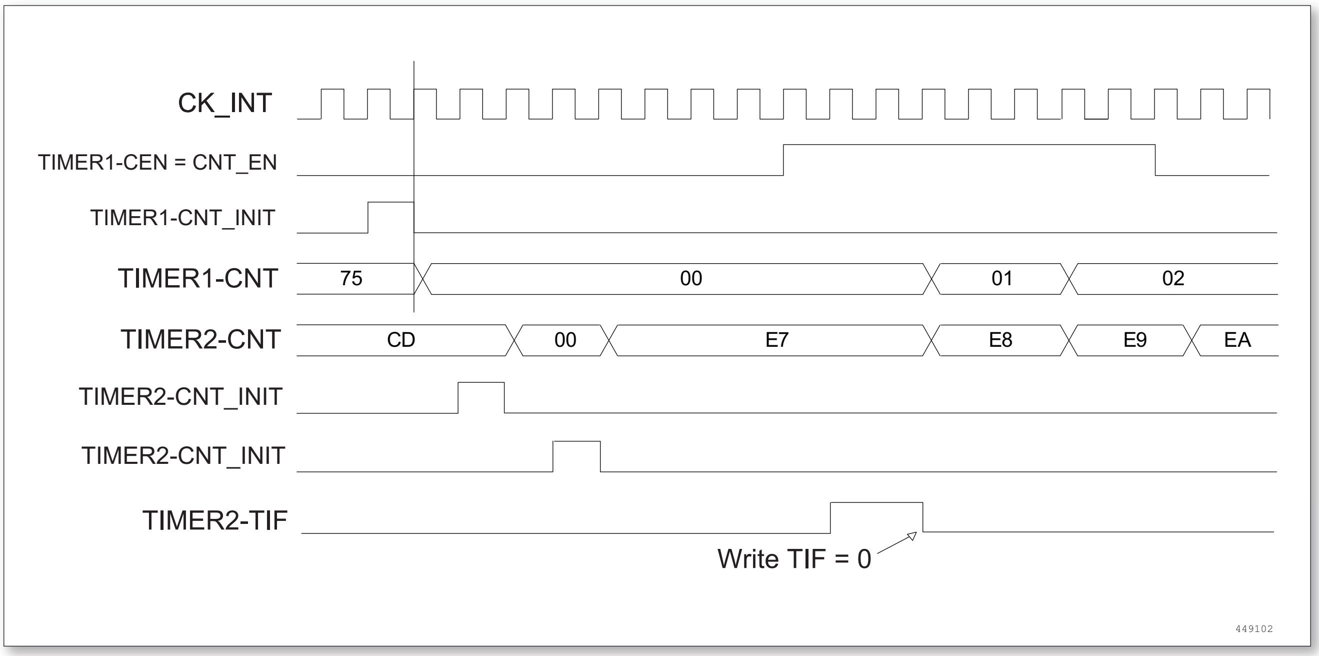


Figure 170. Triggering Timer 2 with Enable of Timer 1

Using one additional timer as prescaler for another timer

In this example, we use Timer 1 as the prescaler for Timer 2. The configuration is as follows:

- Configure Timer 1 in master mode, togenerate the update event (UEV) as the trigger output (MMS=010 in the TIM1_CR2 register). Then, output a periodic signal in case of each counter overflow.
- Configure the Timer 1 period (TIM1_ARR registers).
- Configure Timer 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in the external clock mode (write SMS=111 in the TIM2_SMCR reg-

ister).

- Start Timer 2 by writing ’1’ in the CEN bit (TIM1_CR2 register)
- Start Timer 1 by writing ’1’ in the CEN bit (TIM1_CR1 register).

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 2 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 2):

- 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 2 to get the input trigger from Timer 1 (TS=000 in the TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in the TIM2_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 that the master/slave mode insert a delay between CNT_EN and CK_PSC on timer 1.

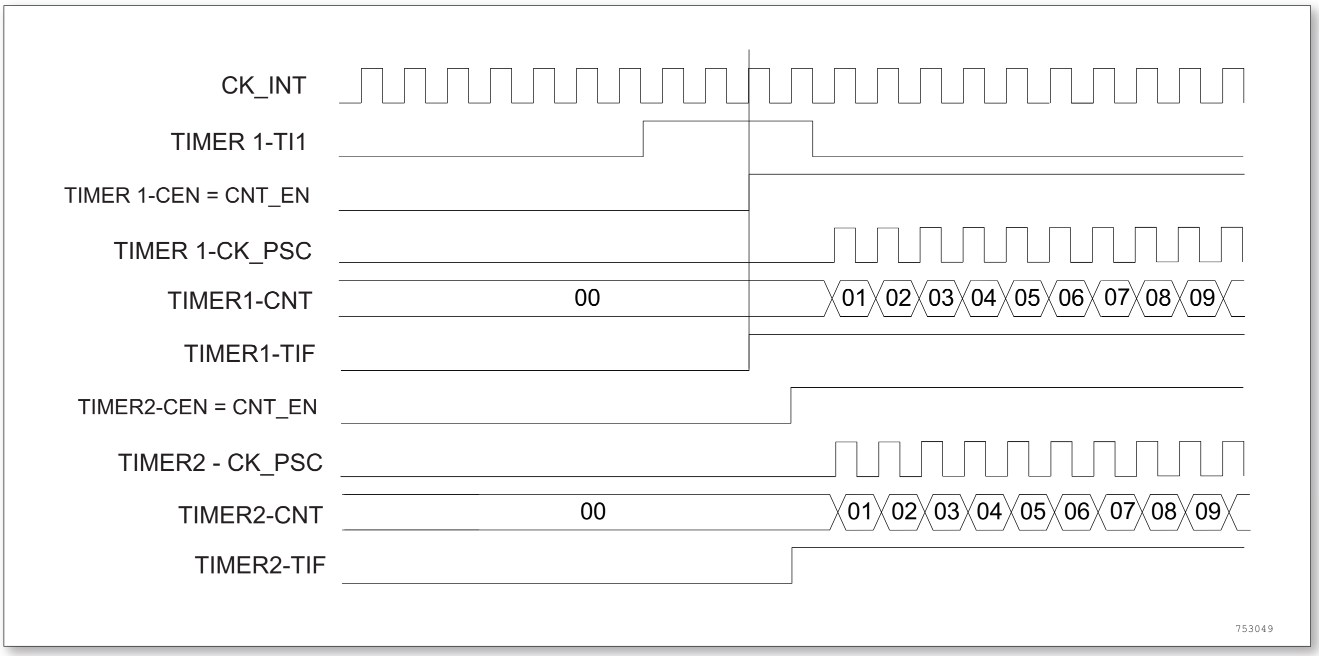


Figure 171. Triggering Timer 1 and 2 with Timer 1 TI1 input

13.3.16 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 ”Debug” sections.

13.4 TIMx register description

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

Table 46. Summary of TIMx Register

Offset	Acronym	Register Name	Reset	Section
0x00	TIMx_CR1	Control register 1	0x00000000	section 13.4.1
0x04	TIMx_CR2	Control register 2	0x00000000	section 13.4.2
0x08	TIMx_SMCR	Slave mode control register	0x00000000	section 13.4.3
0x0C	TIMx_DIER	DMA /interrupt enable register	0x00000000	section 13.4.4
0x10	TIMx_SR 32	Status register	0x00000000	section 13.4.5
0x14	TIMx_EGR	Event generation register	0x00000000	section 13.4.6
0x18	TIMx_CCMR1	Capture/compare mode register 1	0x00000000	section 13.4.7
0x1C	TIMx_CCMR2	Capture/compare mode register 2	0x00000000	section 13.4.8
0x20	TIMx_CCER	Capture/compare enable register	0x00000000	section 13.4.9
0x24	TIMx_CNT	Counter	0x00000000	section 13.4.10
0x28	TIMx_PSC	Prescaler	0x00000000	section 13.4.11
0x2C	TIMx_ARR	Auto-reload register	0x00000000	section 13.4.12
0x34	TIMx_CCR1	Capture/compare register 1	0x00000000	section 13.4.13
0x38	TIMx_CCR2	Capture/compare register 2	0x00000000	section 13.4.14
0x3C	TIMx_CCR3	Capture/compare register 3	0x00000000	section 13.4.15
0x40	TIMx_CCR4	Capture/compare register 4	0x00000000	section 13.4.16
0x48	TIMx_DCR	DMA control register	0x00000000	section 13.4.17
0x4C	TIMx_DMAR	DMA address in continuous mode	0x00000000	section 13.4.18

13.4.1 Control register 1(TIMx_CR1)

Offset address: 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	rw	rw

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
9: 8	CKD	rw	0x00	Clock division The 2 bits indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling clock used by the dead-time generators and the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6: 5	CMS	rw	0x00	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMx_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).
4	DIR	rw	0x00	Direction 0: Counter used as upcounter 1: Counter used as downcounter Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Bit	Field	Type	Reset	Description
2	URS	rw	0x00	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generates an update interrupt or DMA request if enabled.These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	rw	0x00	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller, buffered registers are then loaded with their preload values 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	rw	0x00	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.

13.4.2 Control register 2(TIMx_CR2)

Offset address: 0x04

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7	TI1S	rw	0x00	TI1 selection 0: The TIMx_CH1 pin is connected to TI1 input 1: The TIMx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
6: 4	MMS	rw	0x00	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the TIMx_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enable. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance, a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
3	CCDS	rw	0x00	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs

Bit	Field	Type	Reset	Description
2: 0	Reserved			Reserved, always read as 0.

13.4.3 Slave mode control register(TIMx_SMCR)

Offset address: 0x08

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS		ETF				MSM	TS			OCCS	SMS		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	ETP	rw	0x00	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
14	ECE	rw	0x00	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: It is possible to simultaneously use external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13: 12	ETPS	rw	0x00	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of TIMxCLK frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8

Bit	Field	Type	Reset	Description
11: 8	ETF	rw	0x00	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS}. 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, 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
7	MSM	rw	0x00	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.

Bit	Field	Type	Reset	Description
6: 4	TS	rw	0x00	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1(ITR1) 010: Internal Trigger 2(ITR2) 011: Internal Trigger 3(ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2(TI2FP2) 111: External Trigger input (ETRF) See the following table for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	OCCS	rw	0x00	Output compare clear selection In PWM mode, clear the comparator output 1: Comparator output as clear signal 0: External trigger signal as clear signal

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x00	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 (refer to Input Control register and Control Register description). 000: Slave mode disabled - if CEN = ‘1’then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP1 edge depending on TI1FP2 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP2 edge depending on TI2FP1 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger input becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger input (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS= ’100’). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Table 47. TIMx Internal Trigger Connection

Slave timer	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM2	TIM1	x	TIM3	x

13.4.4 DMA/interrupt enable register(TIMx_DIER)

Offset address: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	Reserved			Reserved, always read as 0.
14	TDE	rw	0x00	Trigger DMA request enable 0: Trigger DMA request disabled 1: Trigger DMA request enabled
13	Reserved			Reserved, always read as 0.
12	CC4DE	rw	0x00	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled 1: CC4 DMA request enabled
11	CC3DE	rw	0x00	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled 1: CC3 DMA request enabled
10	CC2DE	rw	0x00	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled 1: CC2 DMA request enabled
9	CC1DE	rw	0x00	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	rw	0x00	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	Reserved			Reserved, always read as 0.
6	TIE	rw	0x00	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	Reserved			Reserved, always read as 0.
4	CC4IE	rw	0x00	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	rw	0x00	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	rw	0x00	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled

Bit	Field	Type	Reset	Description
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

13.4.5 Status register(TIMx_SR)

Offset address: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			CC4OF	CC3OF	CC2OF	CC1OF	Res.		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.
12	CC4OF	rc_w0	0x00	Capture/Compare 4 overcapture flag Refer to CC1OF description.
11	CC3OF	rc_w0	0x00	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	rc_w0	0x00	Capture/Compare 2 overcapture flag Refer to CC1OF description.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8: 7	Reserved			Reserved, always read as 0.
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
5	Reserved			Reserved, always read as 0.
4	CC4IF	rc_w0	0x00	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	rc_w0	0x00	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	rc_w0	0x00	Capture/Compare 2 interrupt flag Refer to CC1IF description.

Bit	Field	Type	Reset	Description
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value, with some exception in center-aligned mode (refer to the CMS bits in the TIMx_CR1 register description). It is cleared by software. 0: No match 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if repetition counter= 0) and if the UDIS=0 in the TIMx_CR1 register. – When CNT is reinitialized by software using the UG bit in TIMx_EGR register, if URS=0 and UDIS=0 in the TIMx_CR1 register. –When CNT is reinitialized by a trigger event (refer to the description of synchronization control register), if URS=0 and UDIS=0 in the TIMx_CR1 register.

13.4.6 Event generation register(TIMx_EGR)

Offset address: 0x14

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 7	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	Reserved			Reserved, always read as 0.
4	CC4G	w	0x00	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	w	0x00	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	w	0x00	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler factor is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), otherwise, it takes the auto-reload value (TIMx_ARR) if DIR=1 (downcounting).

13.4.7 Capture/compare mode register 1(TIMx_CCMR1)

Offset address: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M			OC2PE	OC2FE	CC2S		OC1CE	OC1M			OC1PE	OC1FE	CC1S	
IC2F				IC2PSC				IC1F				IC1PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x00	Output compare 2 clear enable
14: 12	OC2M	rw	0x00	Output compare 2 mode
11	OC2PE	rw	0x00	Output compare 2 preload enable
10	OC2FE	rw	0x00	Output compare 4 fast enable
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (in-put/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7	OC1CE	rw	0x00	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF Input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input

Bit	Field	Type	Reset	Description
6: 4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT<TIMx_CCR1 otherwise inactive. In downcounting, channel 1 is inactive (OC1REF= ‘0’) as long as TIMx_CNT>TIMx_CCR1 otherwise active (OC1REF=’ 1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1otherwise active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 otherwise inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=’ 00’ (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S='00' (the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC2F	rw	0x00	Input capture 2 filter
11: 10	IC2PSC	rw	0x00	Input capture 2 prescaler
9: 8	CC2S	rw	0x00	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC2S bits are writable only when the channel is OFF (CC2E = ‘0’ in TIMx_CCER).
7: 4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 6 0001: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 2 1001: sampling frequency $f_{SAMPLING}=f_{DTS}/8$, N = 8 0010: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 4 1010: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 5 0011: sampling frequency $f_{SAMPLING}=f_{CK\_INT}$, N = 8 1011: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 6 0100: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 6 1100: sampling frequency $f_{SAMPLING}=f_{DTS}/16$, N = 8 0101: sampling frequency $f_{SAMPLING}=f_{DTS}/2$, N = 8 1101: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 5 0110: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 6 1110: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 6 0111: sampling frequency $f_{SAMPLING}=f_{DTS}/4$, N = 8 1111: sampling frequency $f_{SAMPLING}=f_{DTS}/32$, N = 8

Bit	Field	Type	Reset	Description
3: 2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the factor of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E='0'(TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input. 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	rw	0x00	Capture/compare 1 selection This bit-field defines the direction of the channel (in-put/output) as well as the input pin. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIMx_CCER).

13.4.8 Capture/compare mode register 2(TIMx_CCMR2)

Offset address: 0x1C

Reset value: 0x0000

Refer to the above CCMR1 register description.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M			OC4PE	OC4FE	CC4S		OC3CE	OC3M			OC3PE	OC3FE	CC3S	
IC4F				IC4PSC				IC3F				IC3PSC			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x00	Output compare 4 clear enable
14: 12	OC4M	rw	0x00	Output compare 4 mode
11	OC4PE	rw	0x00	Output compare 4 preload enable
10	OC4FE	rw	0x00	Output compare 4 fast enable

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER)
7	OC3CE	rw	0x00	Output compare 3 clear enable
6: 4	OC3M	rw	0x00	Output compare 3 mode
3	OC3PE	rw	0x00	Output compare 3 preload enable
2	OC3FE	rw	0x00	Output compare 3 fast enable
1: 0	CC3S	rw	0x00	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER)

Input capture mode:

Bit	Field	Type	Reset	Description
15: 12	IC4F	rw	0x00	Input capture 4 filter
11: 10	IC4PSC	rw	0x00	Input capture 4 prescaler

Bit	Field	Type	Reset	Description
9: 8	CC4S	rw	0x00	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMx_CCER).
7: 4	IC3F	rw	0x00	Input capture 3 filter
3: 2	IC3PSC	rw	0x00	Input capture 3 prescaler
1: 0	CC3S	rw	0x00	Capture/compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the input pin. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI4 10: CC3 channel is configured as input, IC3 is mapped on TI3 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (TIMx_SMCR register) Note: CC3S bits are writable only when the channel is OFF (CC3E = ‘0’ in TIMx_CCER).

13.4.9 Capture/compare enable register(TIMx_CCER)

Offset address: 0x20

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 14	Reserved			Reserved, always read as 0.
13	CC4P	rw	0x00	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	rw	0x00	Capture/Compare 4 output enable Refer to CC1E description.

Bit	Field	Type	Reset	Description
11: 10	Reserved			Reserved, always read as 0.
9	CC3P	rw	0x00	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	rw	0x00	Capture/Compare 3 output enable Refer to CC1E description.
7: 6	Reserved			Reserved, always read as 0.
5	CC2P	rw	0x00	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	rw	0x00	Capture/Compare 2 output enable Refer to CC1E description.
3: 2	Reserved			Reserved, always read as 0.
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel is configured as output: 0: OC1 active high 1: OC1 active low CC1 channel is configured as input: This bit selects whether IC1 or inverted IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted Note: This bit can not be modified as long as LOCK level 3 or 2 has been programmed (LOCK bits in TIMx_BDTR register).
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel is configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel is configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table 48. Output Control Bit for Standard OCx Channels

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

Note: The state of the external I/O pins connected to the standard OCx channels depends on the OCx channel state and the GPIO and AFIO registers.

13.4.10 Counter(TIMx_CNT)

Offset address: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
31: 16	CNT	rw	0x0000	High counter value
15: 0	CNT	rw	0x0000	Low counter value

13.4.11 Prescaler(TIMx_PSC)

Offset address: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} /(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in ”reset mode”)

13.4.12 Auto-reload register(TIMx_ARR)

Offset address: 0x2C

Reset value: 0x0000

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

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

Bit	Field	Type	Reset	Description
31: 16	ARR	rw	0x0000	High auto-reload value
15: 0	ARR	rw	0x0000	Low auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to section 13.3.1 for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

13.4.13 Capture/compare register 1(TIMx_CCR1)

Offset address: 0x34

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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

Bit	Field	Type	Reset	Description
31: 16	CCR1	rw	0x0000	High Capture/Compare 1 value
15: 0	CCR1	rw	0x0000	Low Capture/Compare 1 value If CC1 channel is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Otherwise the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

13.4.14 Capture/compare register2(TIMx_CCR2)

Offset address: 0x38

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 16	CCR2	rw	0x0000	High Capture/Compare 2 value
15: 0	CCR2	rw	0x0000	Low Capture/Compare 2 value If CC2 channel is configured as output: CCR2 contains the value to be loaded in the actual capture/compare 2 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR2 register (bit OC2PE). Otherwise the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC2 output. If CC2 channel is configured as input: CCR2 contains the counter value transferred by the last input capture 2 event (IC2).

13.4.15 Capture/compare register 3(TIMx_CCR3)

Offset address: 0x3C

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 16	CCR3	rw	0x0000	High Capture/Compare 3 value

Bit	Field	Type	Reset	Description
15: 0	CCR3	rw	0x0000	Low Capture/Compare 3 value If CC3 channel is configured as output: CCR3 contains the value to be loaded in the actual capture/compare 3 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR3 register (bit OC3PE). Otherwise the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC3 output. If CC3 channel is configured as input: CCR3 contains the counter value transferred by the last input capture 3 event (IC3).

13.4.16 Capture/compare register 4(TIMx_CCR4)

Offset address: 0x40

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31: 16	CCR4	rw	0x0000	High Capture/Compare 4 value
15: 0	CCR4	rw	0x0000	Low Capture/Compare 4 value If CC4 channel is configured as output: CCR4 contains the value to be loaded in the actual capture/compare 4 register (preload value). The written value is transferred to the current register immediately if the preload feature is not selected in the TIMx_CCMR4 register (bit OC4PE). Otherwise the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC4 output. If CC4 channel is configured as input: CCR4 contains the counter value transferred by the last input capture 4 event (IC4).

13.4.17 DMA control register(TIMx_DCR)

Offset address: 0x48

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.			DBL					Res.			DBA				
			w	w	w	w	w				w	w	w	w	w

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.
12: 8	DBL	w	0x00	DMA burst length This bit field defines the burst transfer in the continuous mode (the timer detects a burst transfer when a write access to the TIMx_DMAR register address is performed), namely, the number of transfers, in half-word (double bytes) or bytes. 00000: 1 transfer 00001: 2 transfers 00010: 3 transfers .... 10001: 18 transfers Example: Let us consider the following transfer: DBL = 7 and DBA = TIM2_CR1. - If DBL =7 and DBA = TIM2_CR1 represent the address of data to be transferred, the transfer address is given by: (Address of TIMx_CR1) + DBA + (DMA index), where, DMA index = DBL TIMx_CR1 address + DBA + 7 is the address of data to be written or read, so that the transfer is completed to/from 7 registers starting from the TIMx_CR1 address + DBA. According to the setting of DMA data length, the following case may occur: -If the data is set to half word (16 bits), the data will be transferred to all 7 registers. -If the data is set to bytes, the data will still be transferred to all 7 registers: the first register contains the first MSB byte, the second register contains the first LSB byte, and so on. Therefore, the user must specify the data width of DMA transfer for the timer.
7: 5	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
4: 0	DBA	w	0x00	DMA base address These bits define the base-address for DMA transfers in the continuous mode (when write access is done through the TIMx_DMAR address). DBA is defined as an offset starting from the address of the TIMx_CR1 register. 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ......

13.4.18 DMA address for full transfer(TIMx_DMAR)

Offset address: 0x4C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	DMAB	w	0x0000	DMA register for burst accesses A write operation to the TIMx_DMAR register will access the register located at the following address: TIMx_CR1 address + DBA + DMA index, Where: ‘TIMx_CR1 address’ is the address of the control register 1; ‘DBA’ is the DMA base address configured in TIMx_DCR register; ‘DMA index’ is the offset automatically controlled by the DMA transfer, depending on DBL configured in TIMx_DCR.

14 Basic timer(TIM14)

Basic timer(TIM14)

14.1 TIM14 introduction

Basic timer TIM14 consist of a 16-bit auto-reload counter driven by a programmable prescaler.

It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare 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.

TIM14 are completely independent, and do not share any resources.

14.2 TIM14 Main features

- 16-bit auto-reload register
- 16-bit programmable prescaler allowing dividing (modifing in real time) the counter clock frequency either by any factor between 1 and 65536.
- Independent channels for:
  - Input capture
  - Output compare
  - PWM generation (Edge Mode)
- Interrupt/DMA generation on the following events:
  - Update: counter overflow, counter initialization (by software)
  - Input capture
  - Output compare

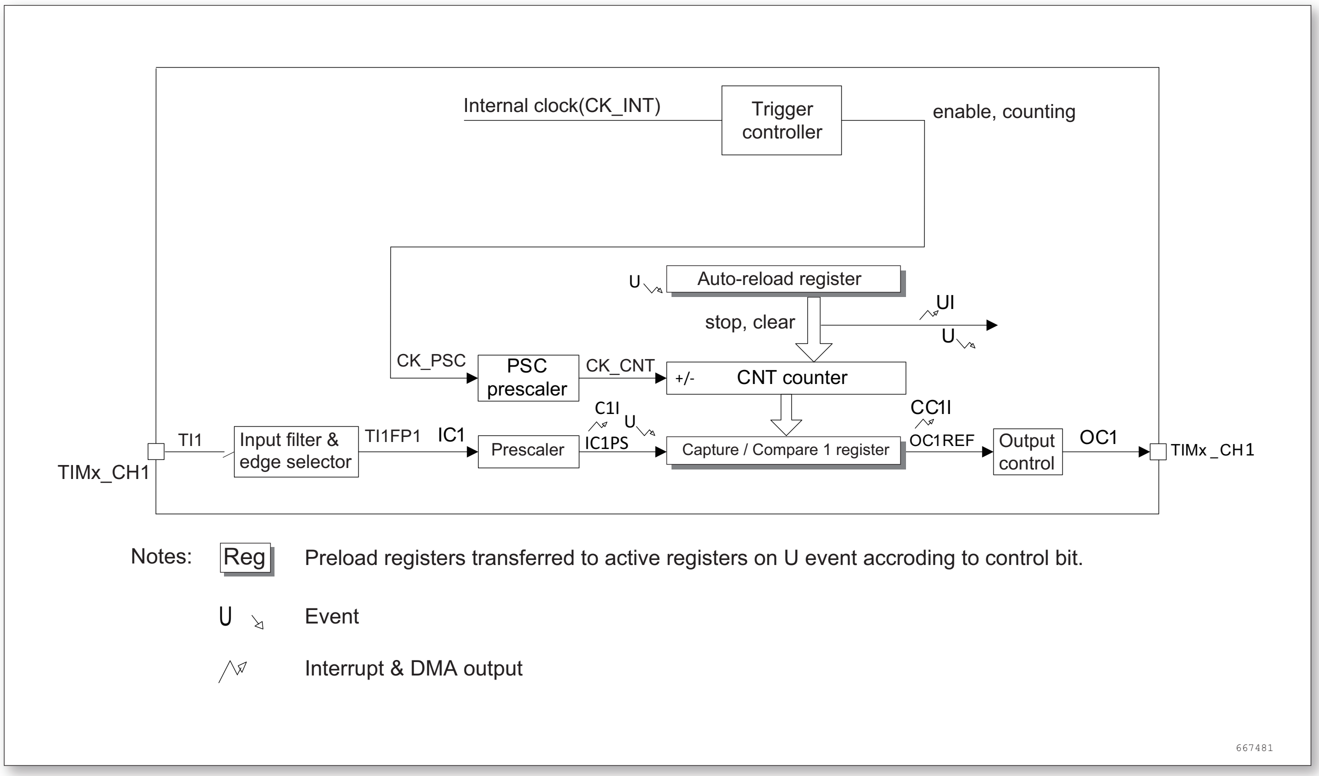


Figure 172. Block Diagram of basic timer

14.3 TIM14 Functional description

14.3.1 Time-base unit

The main block of the programmable basic timer is a 16-bit counter with its related auto-reload 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 (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. The preload register is accessed each time an attempt is made to write or read the auto-reload register. The contents of the preload register are transferred into the shadow register permanently or at each update event UEV, depending on the auto-reload preload enable bit (ARPE) in the TIM14_CR1 register. The update event is sent when the counter reaches the overflow value and if the UDIS bit equals 0 in the TIMx_CR1 register. It can also be generated by software. The generation of the update event is described in detail 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: The actual counter enable signal CNT_EN is set 1 clock cycle after CEN.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 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 the TIMx_PSC control register is buffered. The new prescaler ratio is taken into account at the next update event.

The following figures give some examples of the counter behavior when the prescaler ratio is changed on the fly.

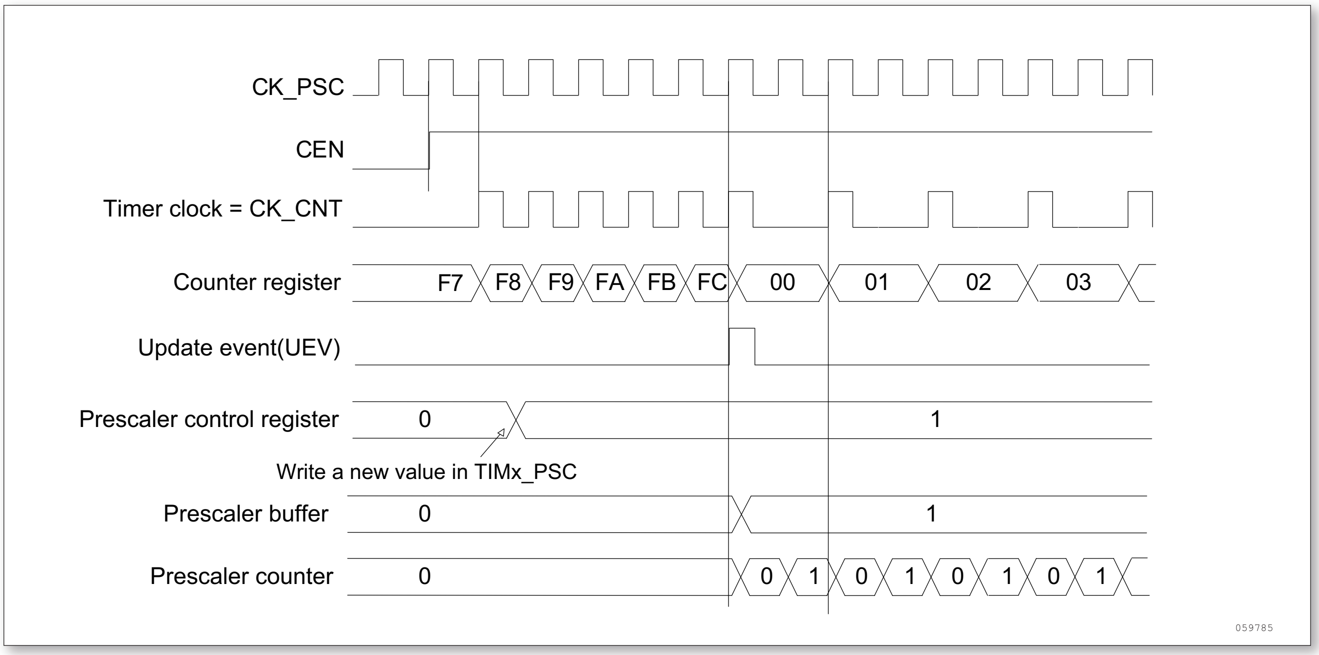


Figure 173. Counter Timing Diagram with Prescaler Division Change from 1 to 2

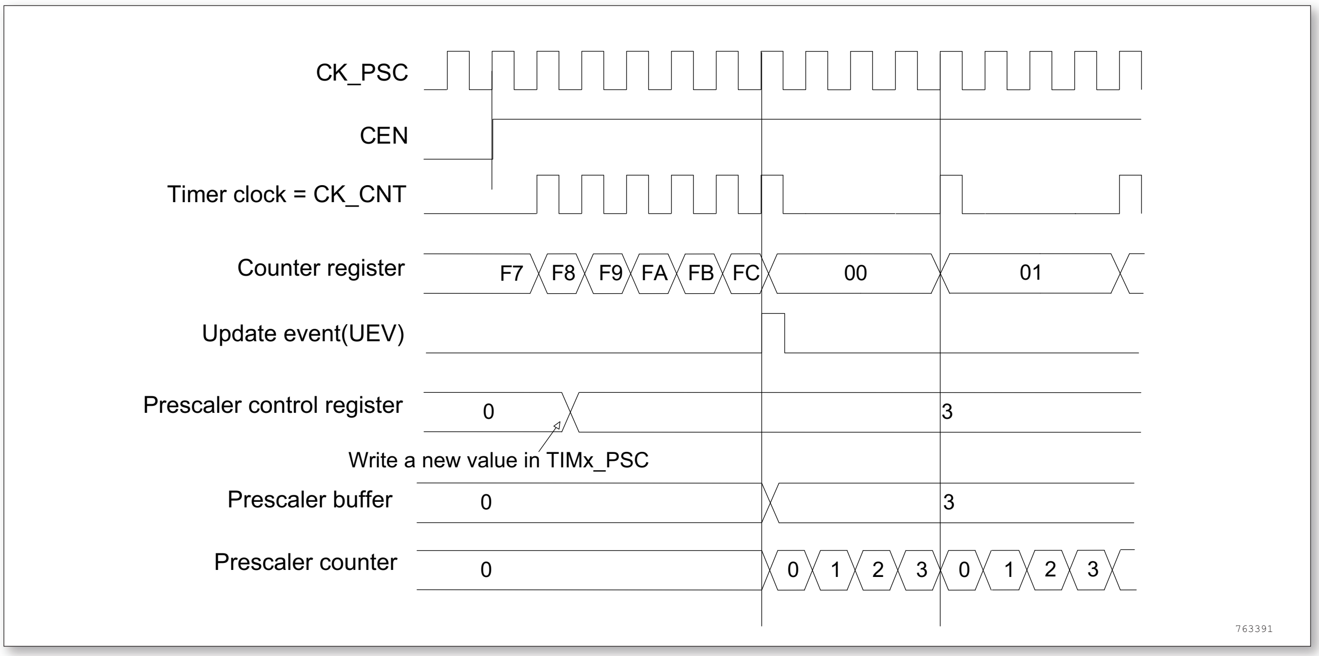


Figure 174. Counter Timing Diagram with Prescaler Division Change from 1 to 4

14.3.2 Counter modes

Upcounting mode

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

Setting the UG bit in the TIM14_EGR register also generates an updateevent.

The UEV event can be disabled by software by setting the UDIS bit in the TIM14_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIM14_CR1 register is set, setting the UG bit generates an update event UEV but withoutsetting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM14_SR register) is set (depending on the URS bit):

- The auto-reload shadow register is updated with the preload value (TIM14_ARR).
- The buffer of the prescaler is reloaded with the preload value (content of the TIM14_PSC register).

The following figures show some examples of the counter behavior for different clock frequencies when TIM14_ARR=0x36.

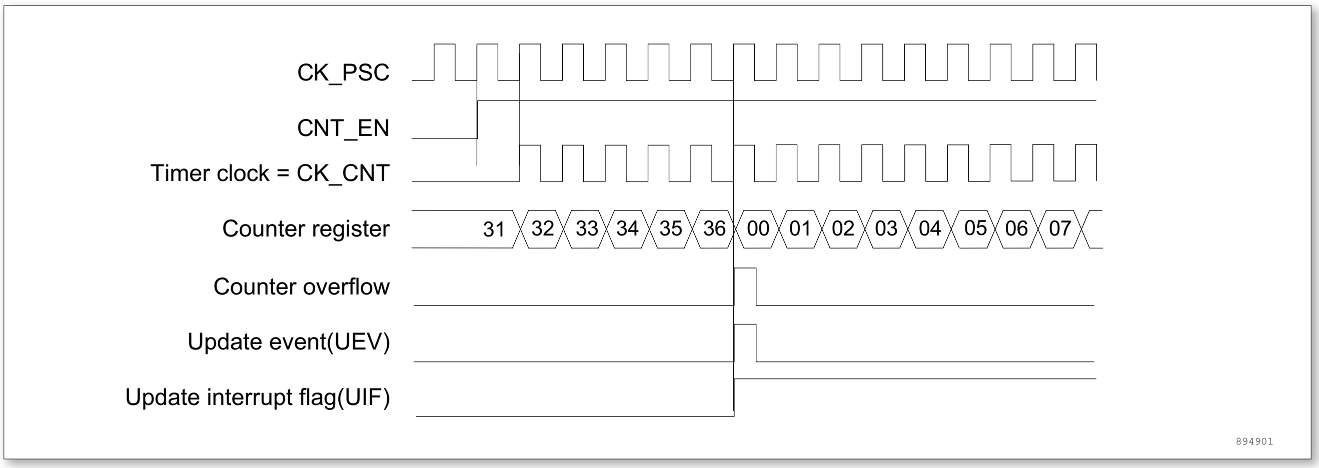


Figure 175. Counter Timing Diagram, Internal Clock Divided by 1

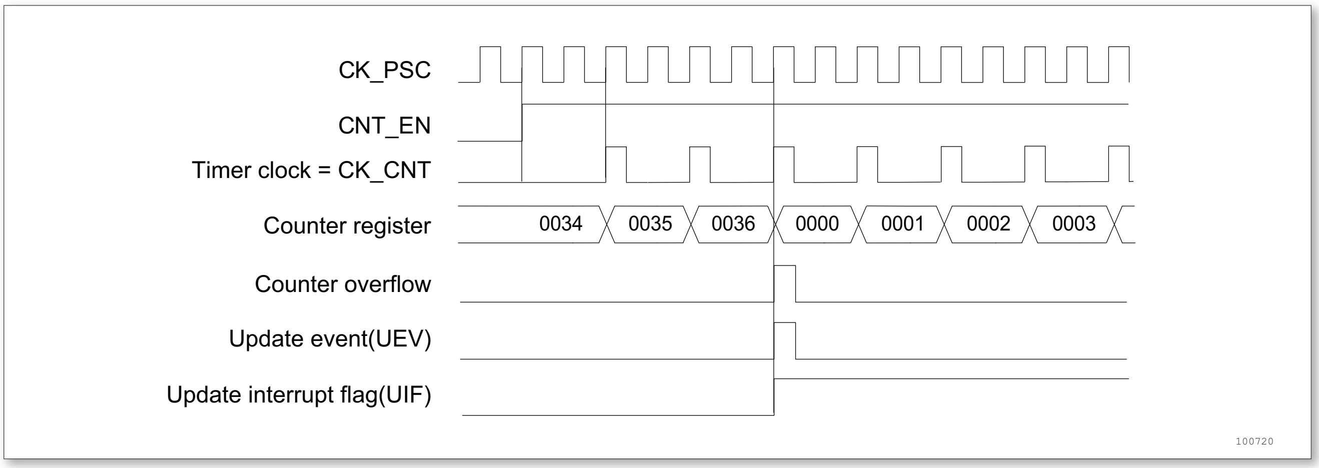


Figure 176. Counter Timing Diagram, Internal Clock Divided by 2

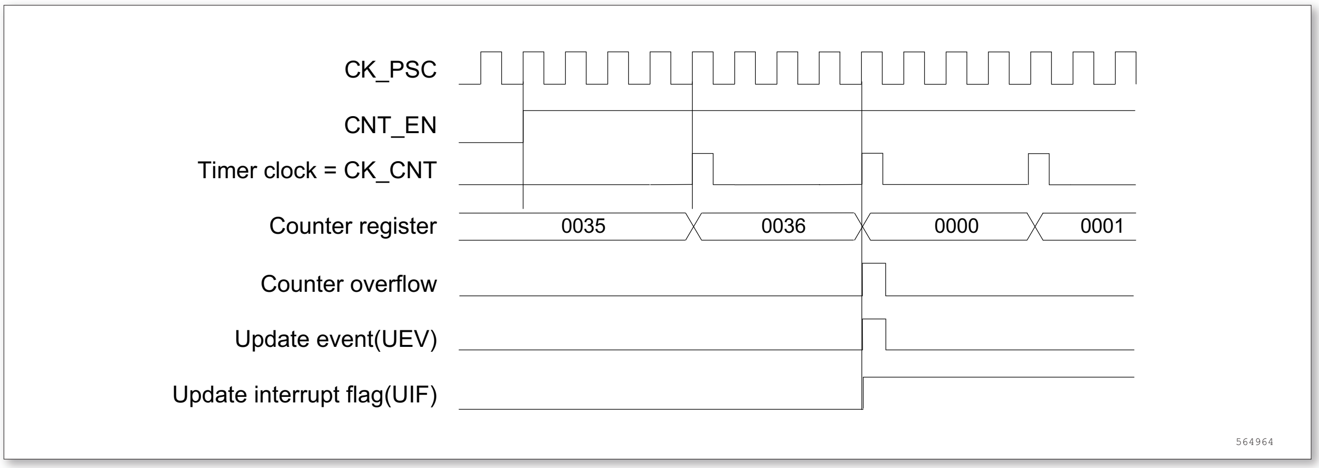


Figure 177. Counter Timing Diagram, Internal Clock Divided by 4

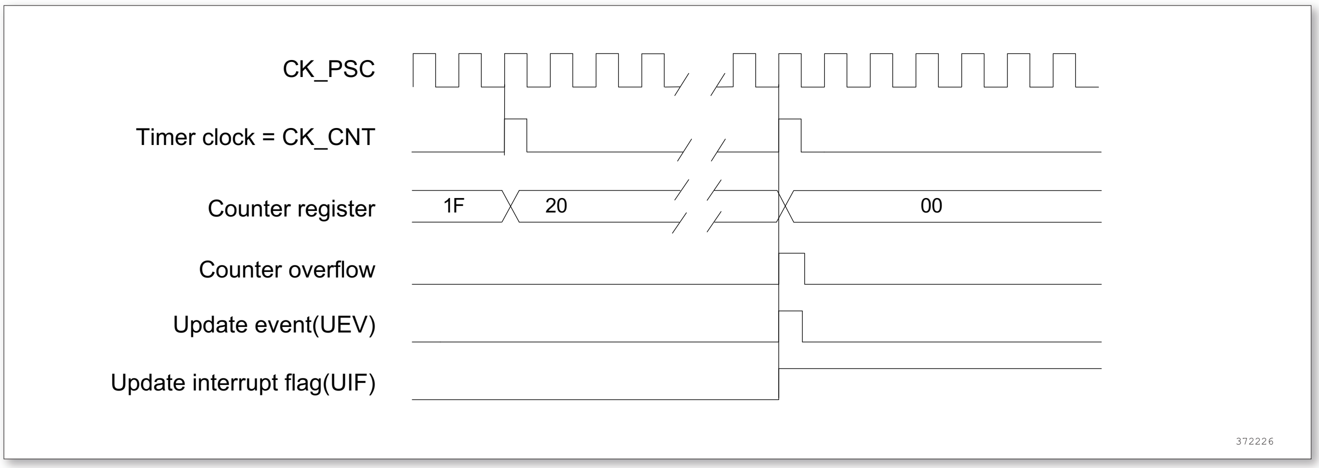


Figure 178. Counter Timing Diagram, Internal Clock Divided by N

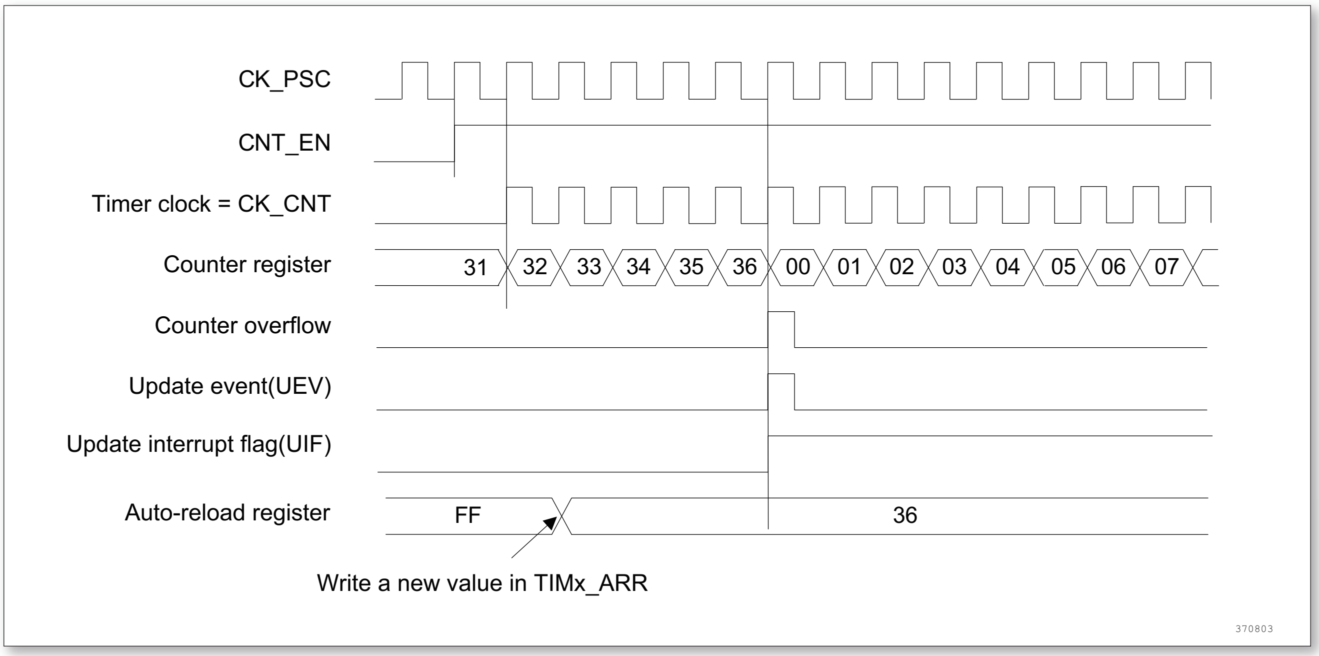


Figure 179. Counter Timing Diagram, Update Event When ARPE=0 (TIM14_ARR Not Preloaded)

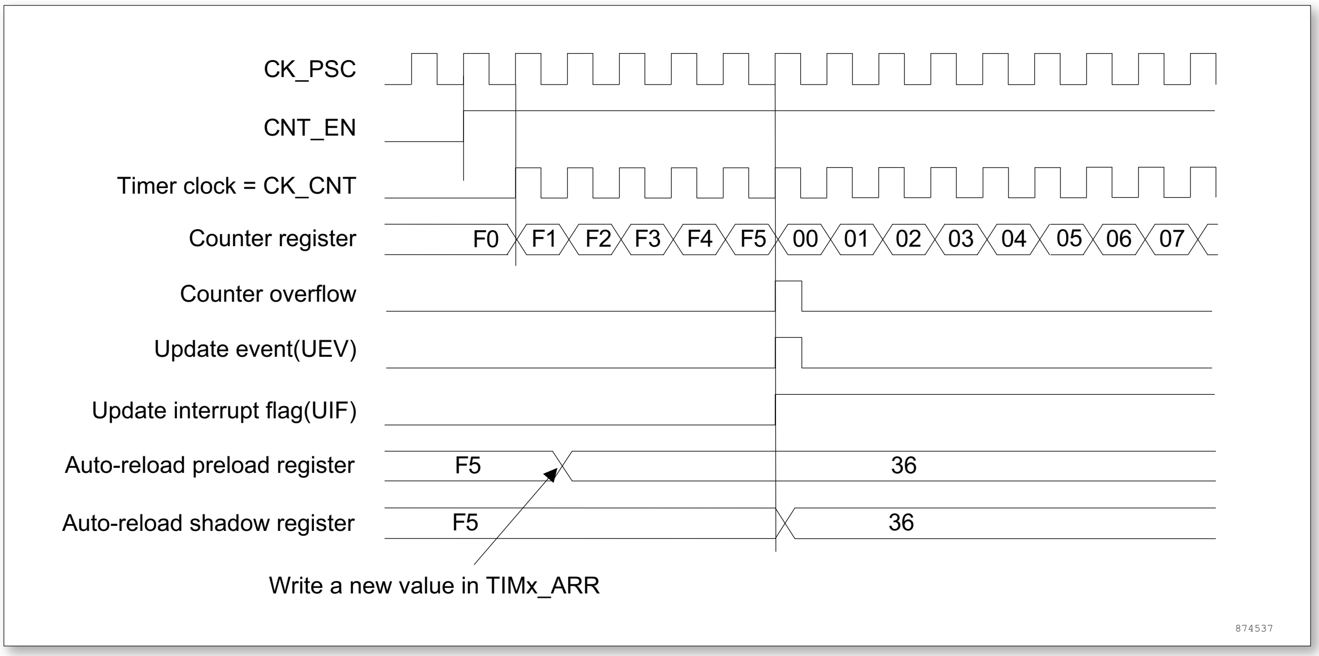


Figure 180. Counter Timing Diagram, Update Event When ARPE=1 (TIM14_ARR Preloaded)

14.3.3 Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows. It is actually generated only when the repetition counter has reached zero, which is very useful to generate PWM signal.

This means that data are transferred from the preload registers to the shadow registers (TIMx_ARR auto-reload register, TIMx_PSC prescaler register, but also TIMx_CCRx capture/compare registers in compare mode) every N counter overflows. N represents the value in TIMx_RCR repetition counter register, which diminishes in case of any following condition:

- At each counter overflow in upcounting mode

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

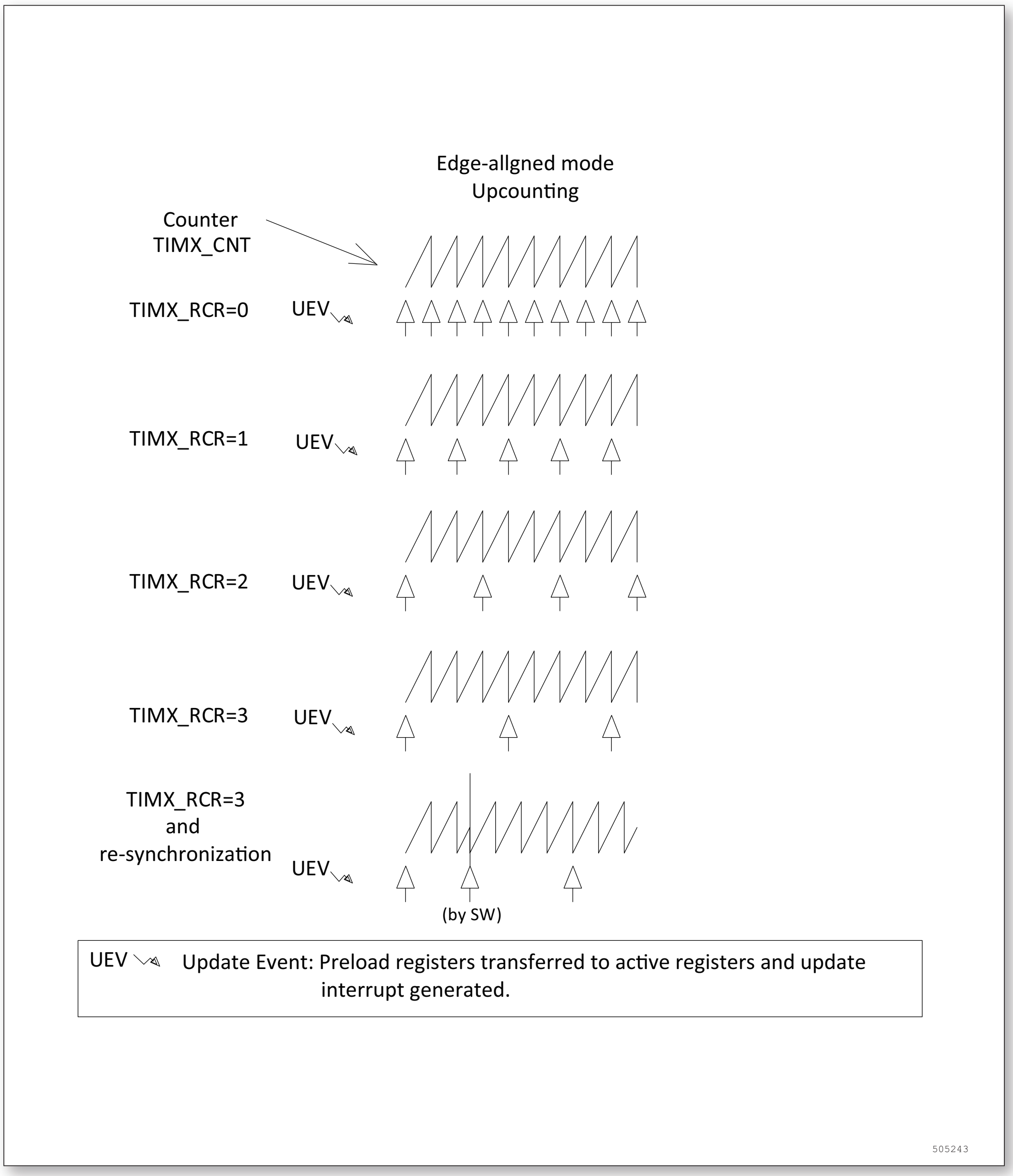


Figure 181. Example of Update Rates in Different Modes and Different TIMx_PCR Register Settings

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

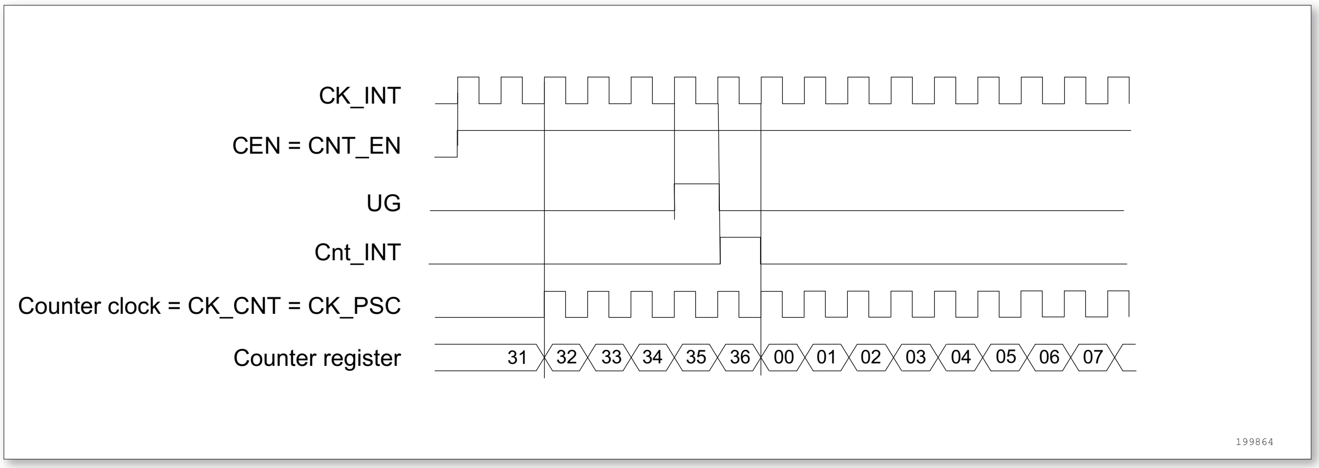


Figure 182. Control Circuit in Normal Mode, Internal Clock Divided By 1

14.3.5 Capture/compare channels

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

The following figures show a 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).

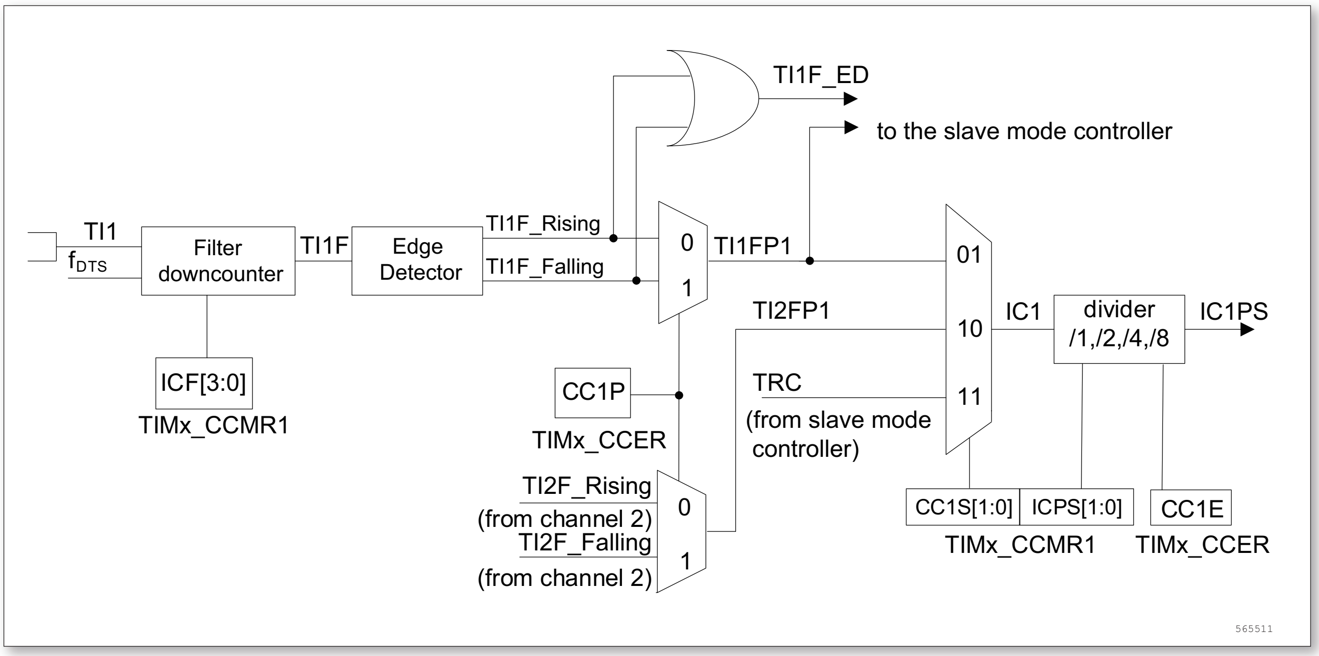


Figure 183. Capture/Compare Channel (Example: Channel 1 Input Stage)

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.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.

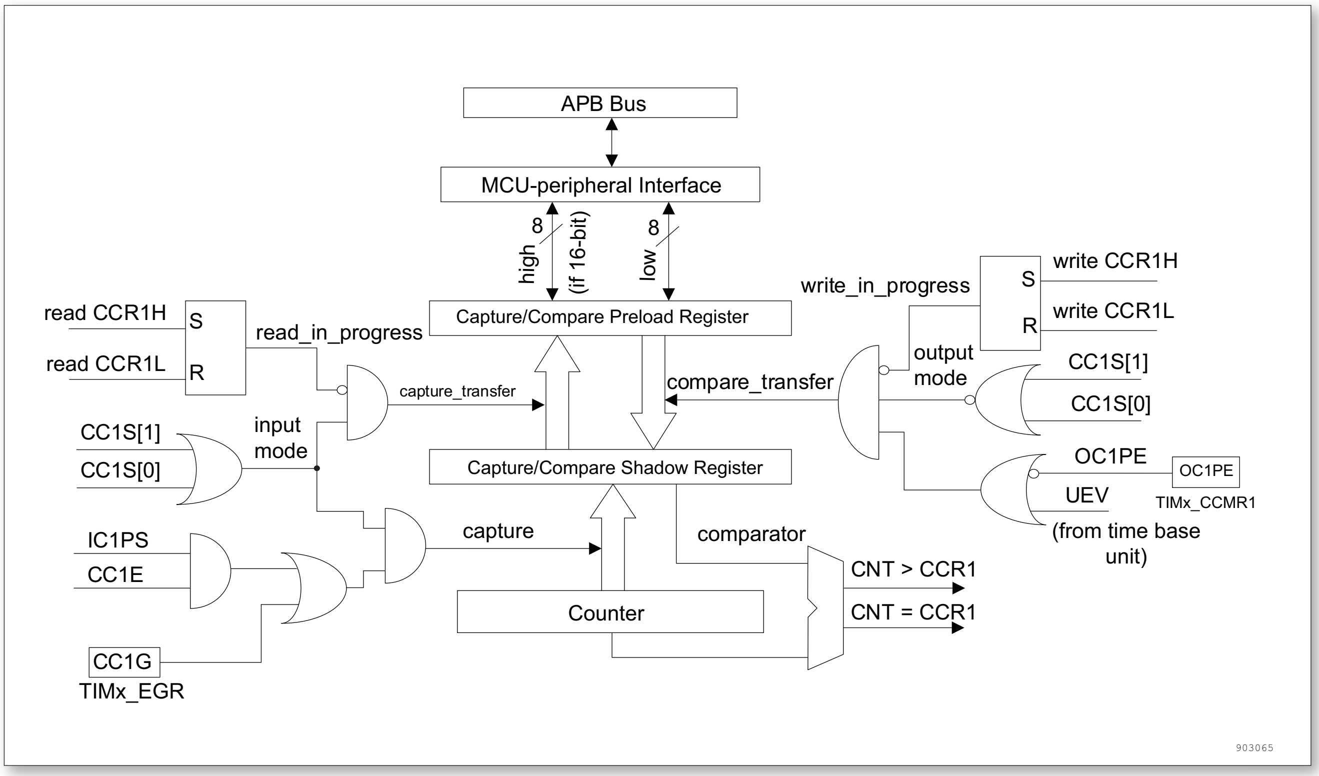


Figure 184. Capture/Compare Channel 1 Main Circuit

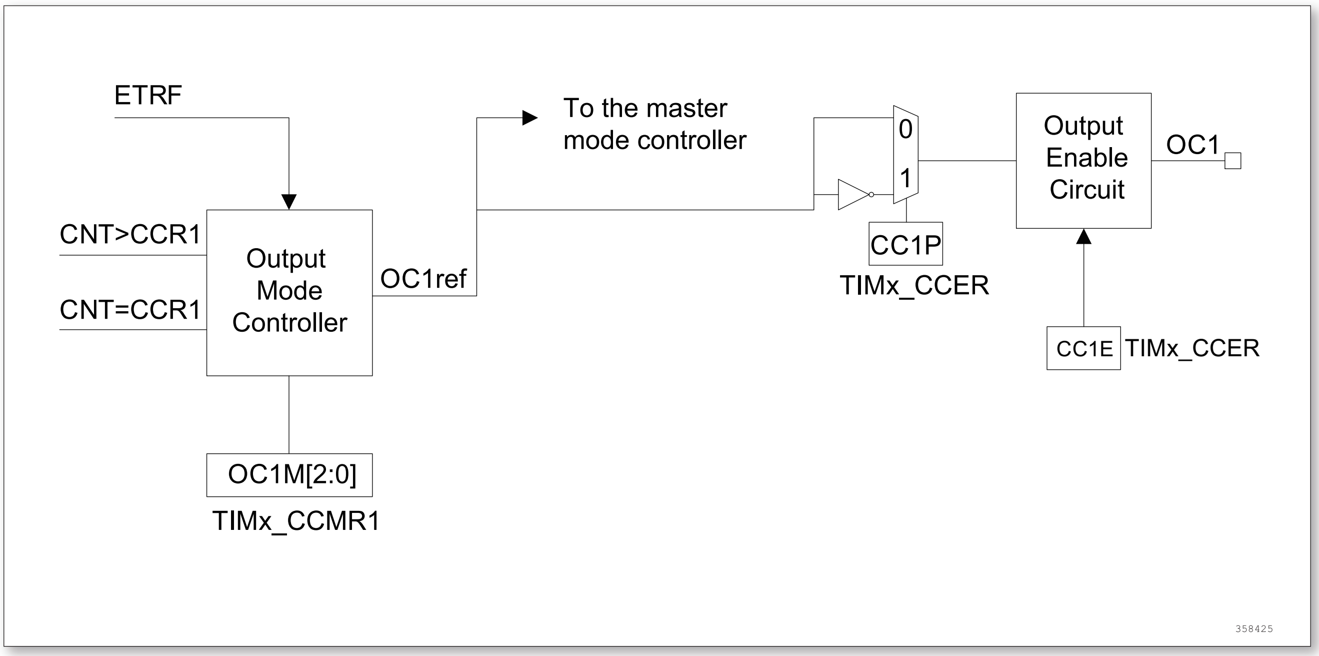


Figure 185. Output Stage of Capture/Compare Channel (Channel 1)

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.

14.3.6 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIM14_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCXIF flag (TIM14_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (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:

1. Select the active input: TIM14_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIM14_CCR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIM14_CCR1 register becomes read-only.
2. Program the needed input filter duration with respect to the signal connected to the timer (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 must five internal clock cycles. We must program a filter bandwidth longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIM14_CCMR1 register.
3. Select the edge of the active transition on the TI1 channel by writing CC1P bit and CC1NP bit to 0 in the TIM14_CCER register (rising edge).
4. Configure 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).
5. Enable capture from the counter into the capture register by setting the CC1E bit in the TIM14_CCER register to ’1’.
6. If needed, enable the related interrupt request by setting the CC1IE bit in the TIM14_DIER register, and/or the DMA request by setting the CC1DE bit in the TIM14_DIER register.

When an input capture occurs:

- The TIM14_CCR1 register gets the value of the counter on the active transition. CC1IF flag is set (interrupt flag). CC1IF is 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: IC (input compare) interrupt DMA request can be generated by software by setting the corresponding CCxG bit in the TIM14_EGR register.

14.3.7 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 get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level. OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM14_CCMRx register.

In this mode, 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.

14.3.8 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

1. Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIM14_CCMRx register) and the output polarity (CCxP bit in the 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.
2. Sets a flag in the interrupt status register (CCxIF bit in the TIM14_SR register).
3. Generates an interrupt if the corresponding interrupt mask is set (CCXIE bit in the TIM14_DIER register).

The TIM14_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIM14_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing precision is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

Procedure:

- Select the counter clock (internal, external, prescaler).
- Write the desired data in the TIM14_ARR and TIM14_CCRx registers.
- Set the CCxIE and/or CCxDE bits if an interrupt and/or a DMA 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 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 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.

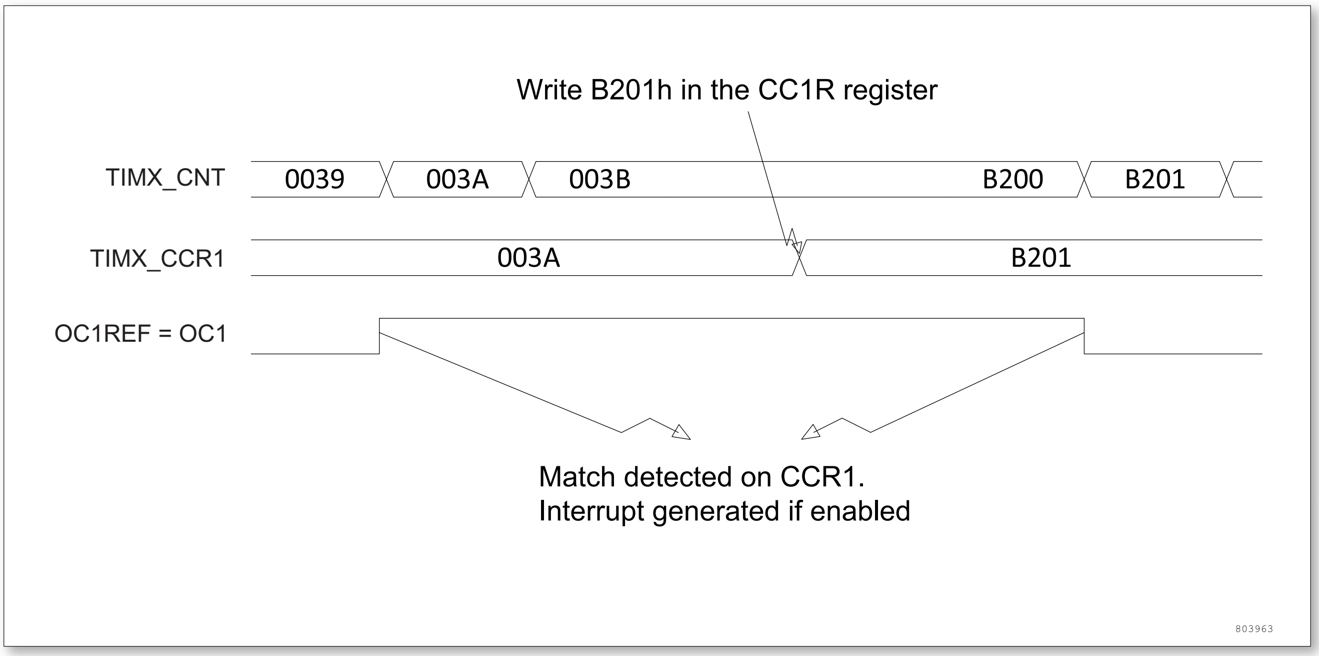


Figure 186. Output Compare Mode, Toggle on OC1

14.3.9 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 bits in the TIM14_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIM14_CCMRx register, and eventually the auto-reload preload register (in upcounting mode) 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 programmed as high or low. OCx output is enabled by the CCxE bit in the TIM14_CCER register. Refer to the TIM14_CCERx register description for more details.

In PWM mode (1 or 2), TIM14_CNT and TIM14_CCRx are always compared to determine whether TIM14_CNT ≤ TIMx_CCRx.

The upcounting timer is only able to generate PWM in edge-aligned mode.

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.

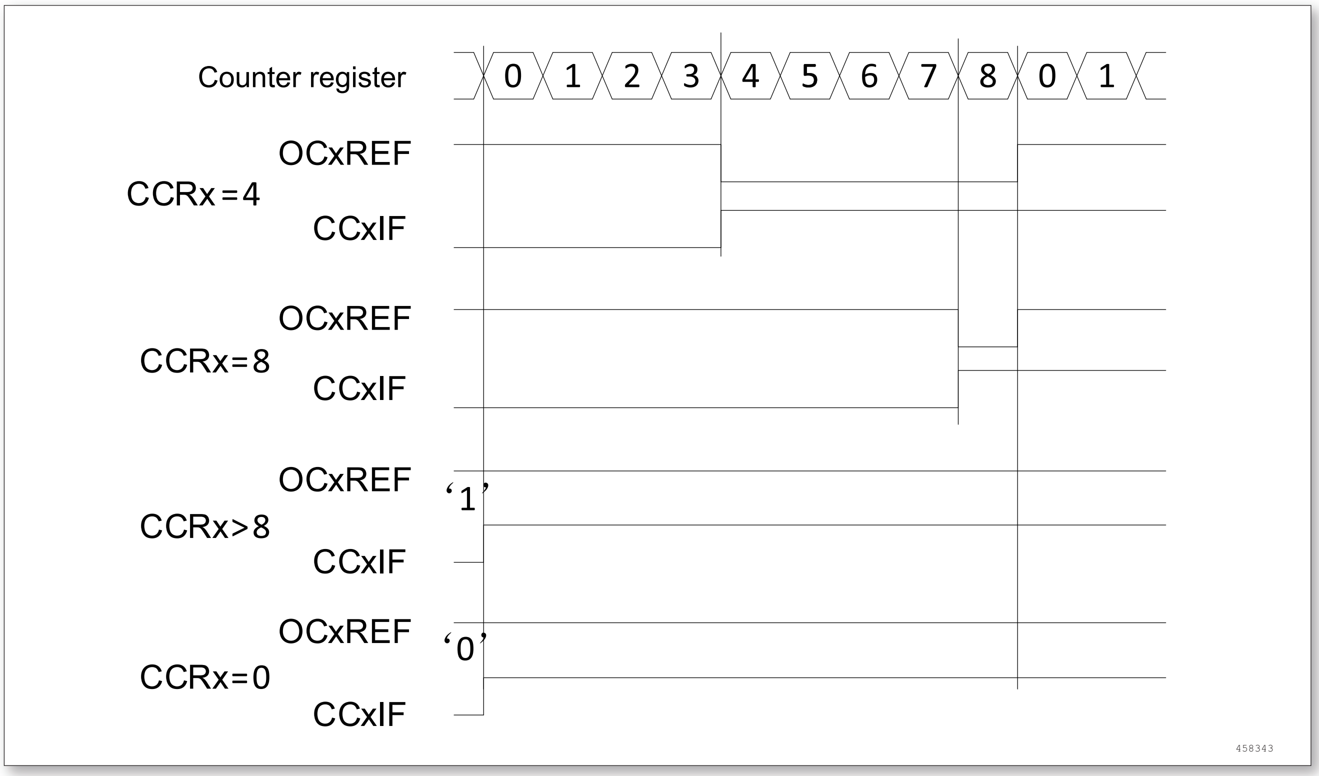


Figure 187. Edge-aligned PWM Waveforms (ARR=8)

14.3.10 Debug mode

When the microcontroller enters debug mode (Cortex^{TM}-M0 halted), the TIM14 counter either continues to work normally or stops, depending on DBG_TIMx_STOP configuration bit in DBG module.

14.4 TIM14 register description

Table 49. Summary of TIM14 Register

Offset	Acronym	Register Name	Reset	Section
0x00	TIM14_CR1	Control register 1	0x00000000	section 14.4.1
0x0C	TIM14_DIER	Interrupt enable register	0x00000000	section 14.4.2
0x10	TIM14_SR	Status register	0x00000000	section 14.4.3
0x14	TIM14_EGR	Event generation register	0x00000000	section 14.4.4
0x18	TIM14_CCMR1	Capture/compare mode register 1	0x00000000	section 14.4.5

Offset	Acronym	Register Name	Reset	Section
0x20	TIM14_CCER	Capture/compare enable register	0x00000000	section 14.4.6
0x24	TIM14_CNT	Counter	0x00000000	section 14.4.7
0x28	TIM14_PSC	Prescaler	0x00000000	section 14.4.8
0x2C	TIM14_ARR	Auto-reload register	0x00000000	section 14.4.9
0x30	TIM14_RCR	Repetition counter register	0x00000000	section 14.4.10
0x34	TIM14_CCR1	Capture/compare register 1	0x00000000	section 14.4.11

14.4.1 Control register 1(TIM14_CR1)

Offset address: 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	rw

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9: 8	CKD	rw	0x00	Clock division The 2 bits indicates the division ratio between the timer clock (CK_INT) frequency and the dead-time and sampling frequency used by the dead-time generators and the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved, do not program this value
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIM14_ARR register is not buffered 1: TIM14_ARR register is buffered
6: 3	Reserved			Reserved, always read as 0.
2	URS	rw	0x00	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generates an update interrupt (UEV) if enabled. - Counter overflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow generates an update interrupt (UEV) if enabled.

Bit	Field	Type	Reset	Description
1	UDIS	rw	0x00	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow - Setting the UG bit 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set.
0	CEN	rw	0x00	Counter enable 0: Counter disabled. 1: Counter enabled.

14.4.2 Interrupt enable register(TIM14_DIER)

Offset address: 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	Type	Reset	Description
15:2	Reserved			Reserved, always read as 0.
1	CC1IE	rw	0x00	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

14.4.3 Status register(TIM14_SR)

Offset address: 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
						rc_w0								rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode 1. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIM14_CCR1 register while CC1IF flag was already set.
8: 2	Reserved			Reserved, always read as 0.
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match 1: The content of the counter TIM14_CNT matches the content of the TIM14_CCR1 register. If the content of TIM14_CCR1 is greater than that of TIM14_ARR, CC1IF flag becomes high in case of counter overflow. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIM14_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIM14_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow regarding the counter value and if the UDIS=0 in the TIM14_CR1 register. -When timer is reinitialized by software using the UG bit in TIM14_EGR register, and if URS=0 and UDIS=0 in the TIM14_CR1 register.

14.4.4 Event generation register(TIM14_EGR)

Offset address: 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	Type	Reset	Description
15: 2	Reserved			Reserved, always read as 0.
1	CC1G	w	0x00	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM14_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update event. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared.

14.4.5 Capture/compare mode register 1(TIM14_CCMR1)

Offset address: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Res.	OC1M			OC1PE	OC1FE	CC1S	
								IC1F				IC1PSC			
								rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15:7	Reserved			Reserved, always read as 0.
6: 4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 are derived. OC1REF is active high whereas OC1 active level depends on CC1P bit. 000: Frozen - The comparison between the output compare register TIM14_CCR1 and the counter TIM14_CNT has no effect on OC1REF 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIM14_CNT matches the capture/compare register 1 (TIM14_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIM14_CNT matches the capture/compare register 1 (TIM14_CCR1). 011: Toggle - OC1REF toggles when TIM14_CNT=TIM14_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - Channel 1 is active as long as TIM14_CNT < TIM14_CCR1 else inactive. 111: PWM mode 2 - Channel 1 is inactive as long as TIM14_CNT < TIM14_CCR1 else active. In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload register on TIM14_CCR1 disabled. TIM14_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM14_CCR1 enabled. Read/Write operations access the preload register. TIM14_CCR1 preload value is loaded in the active register at each update event. Note: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIM14_CR1 register). Else the behavior is not guaranteed.

Bit	Field	Type	Reset	Description
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: Reserved 11: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIM14_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15:8	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
7: 4	IC1F	rw	0x00	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 1000: Sampling frequency $f_{SAMPLING}$=f_{DTS}/8, N = 6 0001: Sampling frequency $f_{SAMPLING}$=f_{CK_INT}, N = 2 1001: Sampling frequency $f_{SAMPLING}$=f_{DTS}/8, N = 8 0010: Sampling frequency $f_{SAMPLING}$=f_{CK_INT}, N = 4 1010: Sampling frequency $f_{SAMPLING}$=f_{DTS}/16, N = 5 0011: Sampling frequency $f_{SAMPLING}$=f_{CK_INT}, N = 8 1011: Sampling frequency $f_{SAMPLING}$=f_{DTS}/16, N = 6 0100: Sampling frequency $f_{SAMPLING}$=f_{DTS}/2, N = 6 1100: Sampling frequency $f_{SAMPLING}$=f_{DTS}/16, N = 8 0101: Sampling frequency $f_{SAMPLING}$=f_{DTS}/2, N = 8 1101: Sampling frequency $f_{SAMPLING}$=f_{DTS}/32, N = 5 0110: Sampling frequency $f_{SAMPLING}$=fDTS/4, N = 6 1110: Sampling frequency $f_{SAMPLING}$=f_{DTS}/32, N = 6 0111: Sampling frequency $f_{SAMPLING}$=f_{DTS}/4, N = 8 1111: Sampling frequency $f_{SAMPLING}$=f_{DTS}/32, N = 8
3: 2	IC1PSC	rw	0x00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E= ’0’ (TIM14_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input. 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1: 0	CC1S	rw	0x00	Capture/compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: Reserved 11: Reserved Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIM14_CCER)

14.4.6 Capture/compare enable register(TIM14_CCER)

Offset address: 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	Type	Reset	Description
15: 4	Reserved			Reserved, always read as 0.
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output Polarity If CC1 is configured as an output, CC1NP shall be cleared, namely, CC1NP= 0; If channel CC1 is configured as an input, the polarity of TI1FP1 is jointly controlled by CC1NP and CC1P. See CC1P description for details.
2	Reserved			Reserved, always read as 0.
1	CC1P	rw	0x00	Capture/Compare 1 output polarity CC1 channel is configured as output: 0: OC1 active high 1: OC1 active low CC1 channel is configured as input: CC1P/CC1NP bit (IC1 or inverted IC1) is used to select the polarity of TI1FP1 and TI2FP1 as trigger or capture. 00: non-inverted/rising edge: capture is done on a rising edge of TIxFP1 (capture mode), and TIxFP1 is non-inverted; 01: inverted/falling edge: capture is done on a falling edge of TIxFP1 (capture mode), and TIxFP1 is inverted; 10: Reserved, this configuration is not used 11: non-inverted/rising and falling edges: capture is done on rising and falling edges of TIxFP1 (capture mode), and TIxFP1 is non-inverted; Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S= '00' (the channel is configured in output).

Bit	Field	Type	Reset	Description
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel is configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin CC1 channel is configured as input: This bit determines if a capture of the counter value can actually be done into the TIM14_CCR1 register or not. 0: Capture disabled. 1: Capture enabled.

Table 50. Output Control Bit for Standard OCx Channels

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

Note: The state of the external I/O pins connected to the standard OCx channels depends on the OCx channel state and the GPIO register.

14.4.7 Counter(TIM14_CNT)

Offset address: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CNT	rw	0x0000	counter value

14.4.8 Prescaler(TIM14_PSC)

Offset address: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} / (PSC + 1). PSC contains the value to be loaded in the current prescaler register at each update event.

14.4.9 Auto-reload register(TIM14_ARR)

Offset address: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	ARR	rw	0x0000	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to time-base unit sections for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

14.4.10 Repetition counter register(TIM14_RCR)

Offset address: 0x30

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
7: 0	REP	rw	0x00	Repetition counter value These bits allow the user to set up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related upcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM edge-aligned mode (REP + 1) corresponds to the number of PWM periods.

14.4.11 Capture/compare register 1(TIM14_CCR1)

Offset address: 0x34

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR1	rw	0x0000	Capture/Compare 1 value If CC1 channel is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIM14_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM14_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

15 Basic timer(TIM16/17)

Basic timer(TIM16/17)

15.1 TIM16/17 introduction

The basic timer TIM16/17 consists of a 16-bit auto-reload counter driven by a programmable prescaler. It has multiple purposes, including measuring pulse width (input capture) of input signal or generating output waveform (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 (TIM16/17) is completely independent, sharing no resource.

15.2 Main features

- 16-bit up auto-reload register
- 16-bit programmable prescaler used to divide (also “on the fly”) the counter clock frequency by any factor between 1 and 65536.
- 1 independent channel for:
  - Input capture
  - Output compare
  - PWM generation (edge-aligned mode)
  - One-pulse mode output
- Complementary output of programmable dead time
- Repetition counter to update the timer registers only after a given number of cycles of the counter
- Break input to put the timer’s output signals in reset state or in a known state
- Interrupt/DMA generation on the following events:
  - Update: counter overflow
  - Input capture
  - Output compare
  - Break signal input

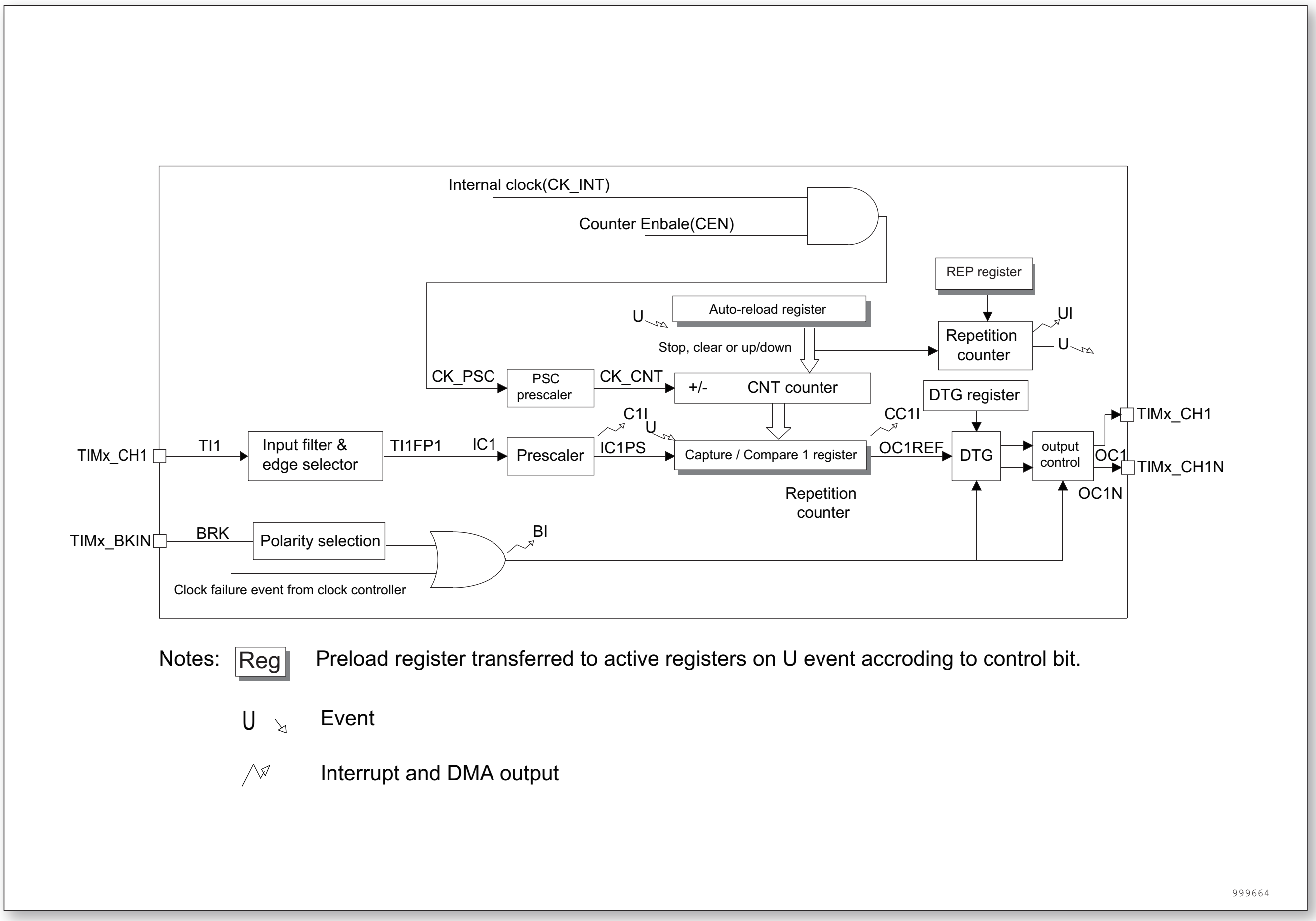


Figure 188. Basic Timers TIM16 and TIM17 Block Diagram

15.3 Functional description

15.3.1 Time-base unit

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

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)
- Repetition counter register (TIMx_RCR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register are transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIMx_CR1 register. The update event is sent when the counter 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 details 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: The actual counter enable signal is set 1 clock cycle after CEN.

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 figure gives some examples of the counter behavior when the prescaler ratio is changed on the fly:

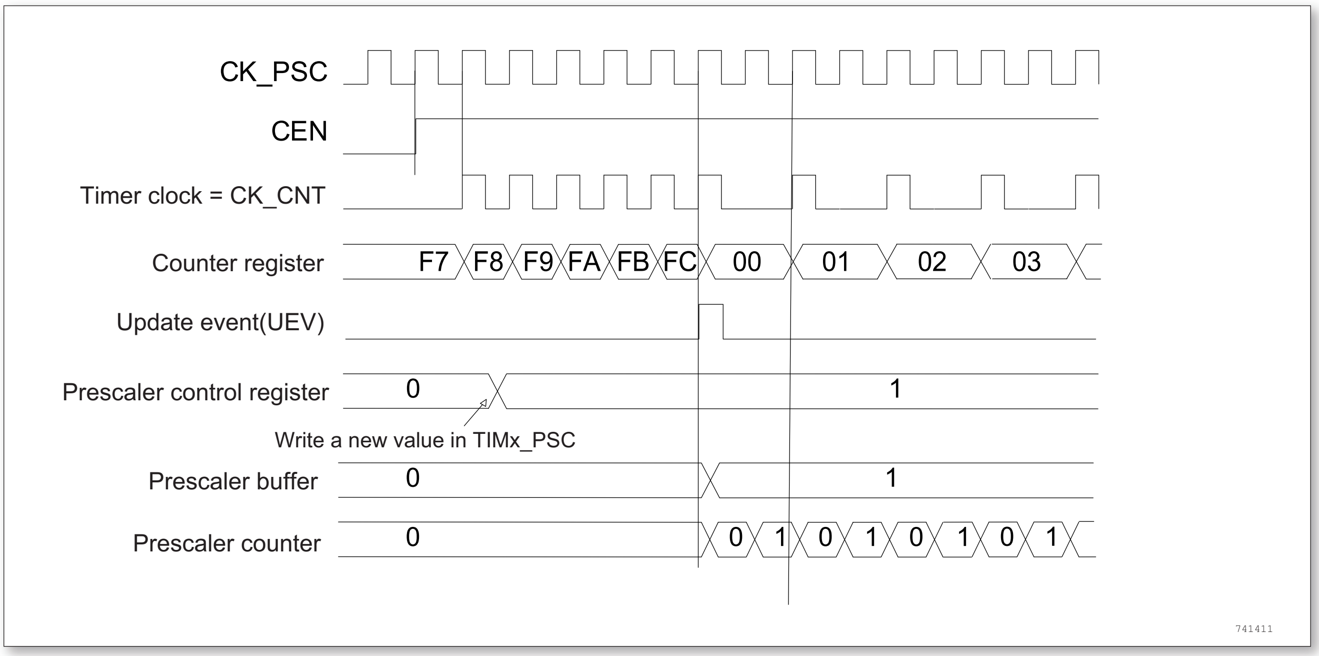


Figure 189. Counter Timing Diagram with Prescaler Division Change from 1 to 2

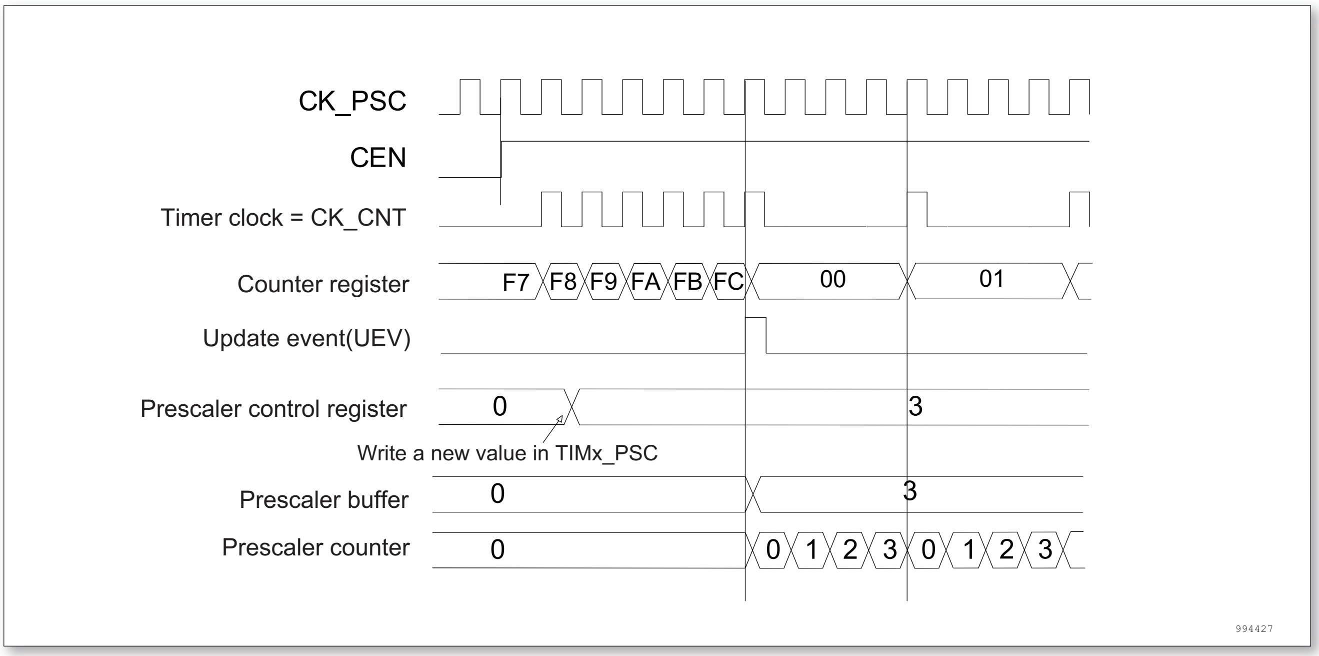


Figure 190. Counter Timing Diagram with Prescaler Division Change from 1 to 4

15.3.2 Counting unit

Upcounting mode

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

If the repetition counter is used, the update event (UEV) is generated after upcounting is repeated for the number of times programmed in the repetition counter register (TIMx_RCR). Else, the update event is generated at each counter overflow. 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 software by setting the UDIS bit in the TIMx_CR1 register. This is to avoid updating the shadow registers while writing new values in the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler (but the prescale rate does not change). In addition, if the URS bit (update request selection) in TIMx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt or DMA request is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event. When an update event occurs, all the registers are updated and the update flag (UIF bit in TIMx_SR register) is set (depending on the URS bit).

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

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

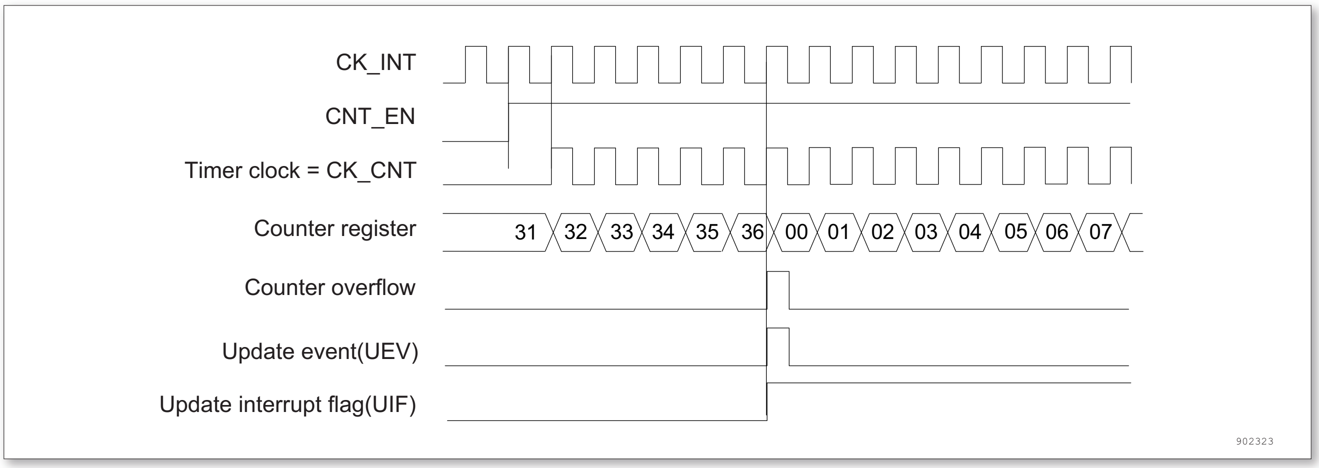


Figure 191. Counter Timing Diagram, Internal Clock Divided by 1

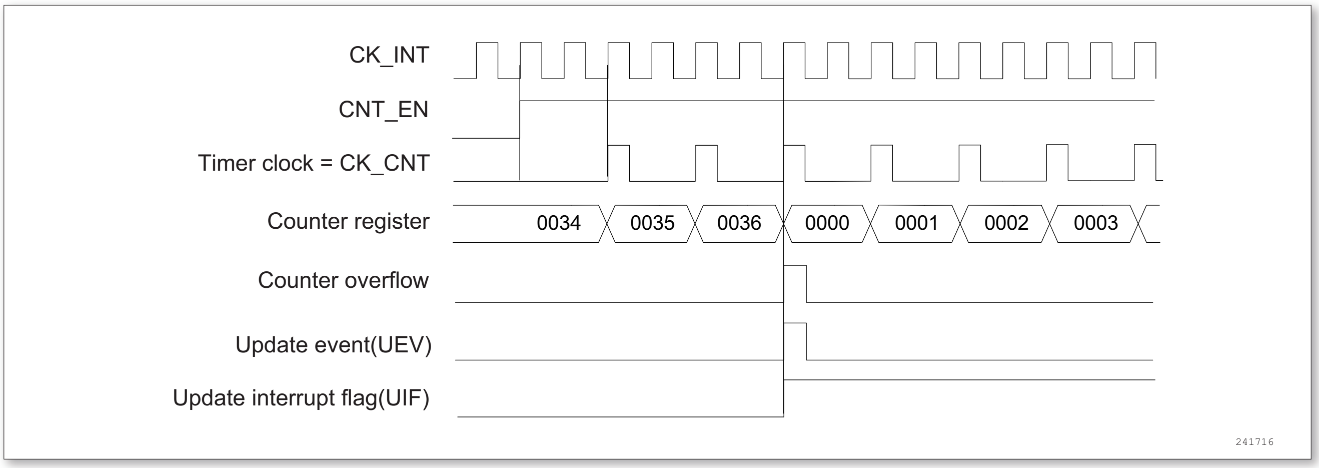


Figure 192. Counter Timing Diagram, Internal Clock Divided by 2

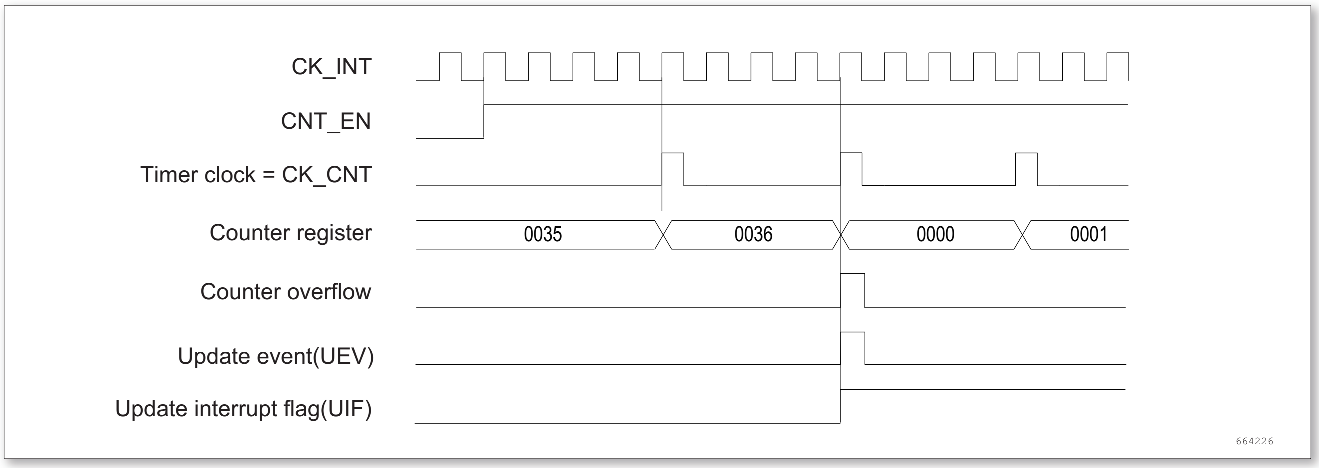


Figure 193. Counter Timing Diagram, Internal Clock Divided by 4

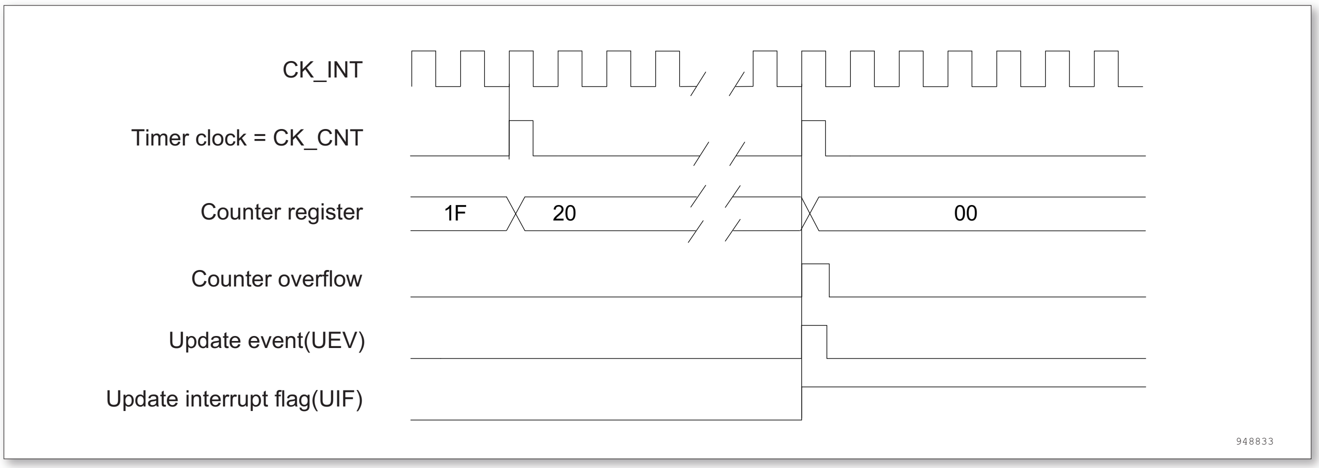


Figure 194. Counter Timing Diagram, Internal Clock Divided by N

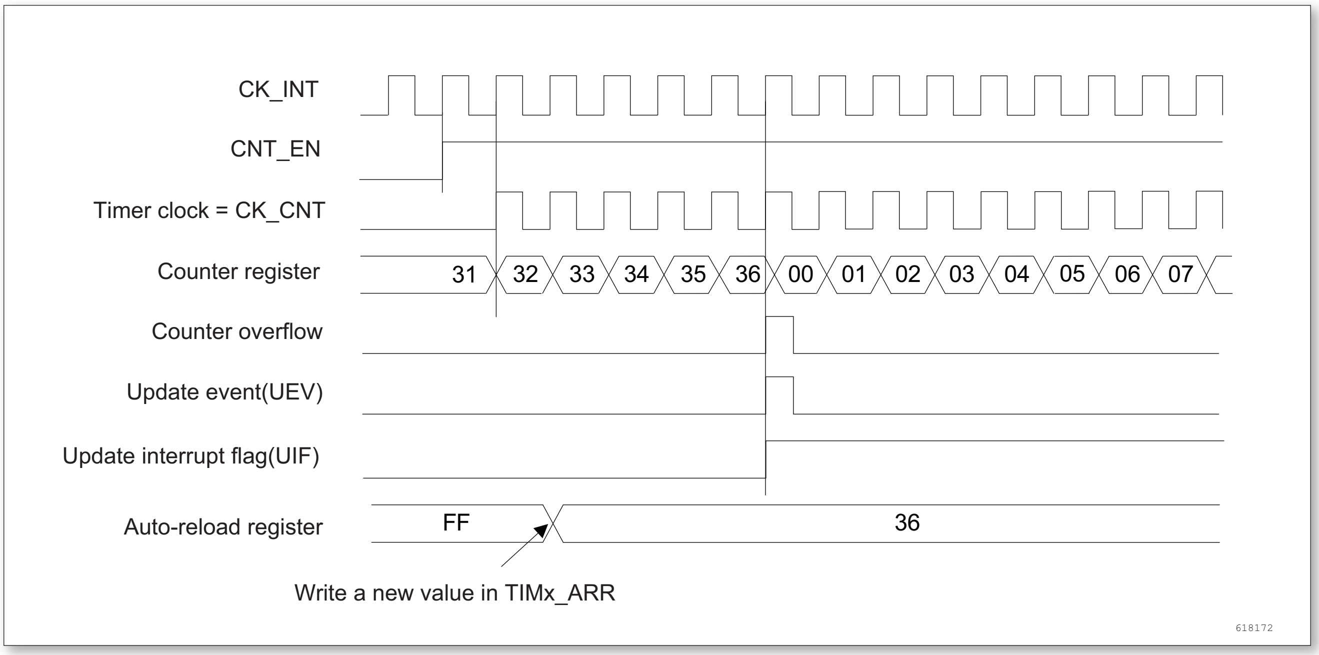


Figure 195. Counter Timing Diagram, Update Event When APRE=0 (TIMx_ARR Not Preloaded)

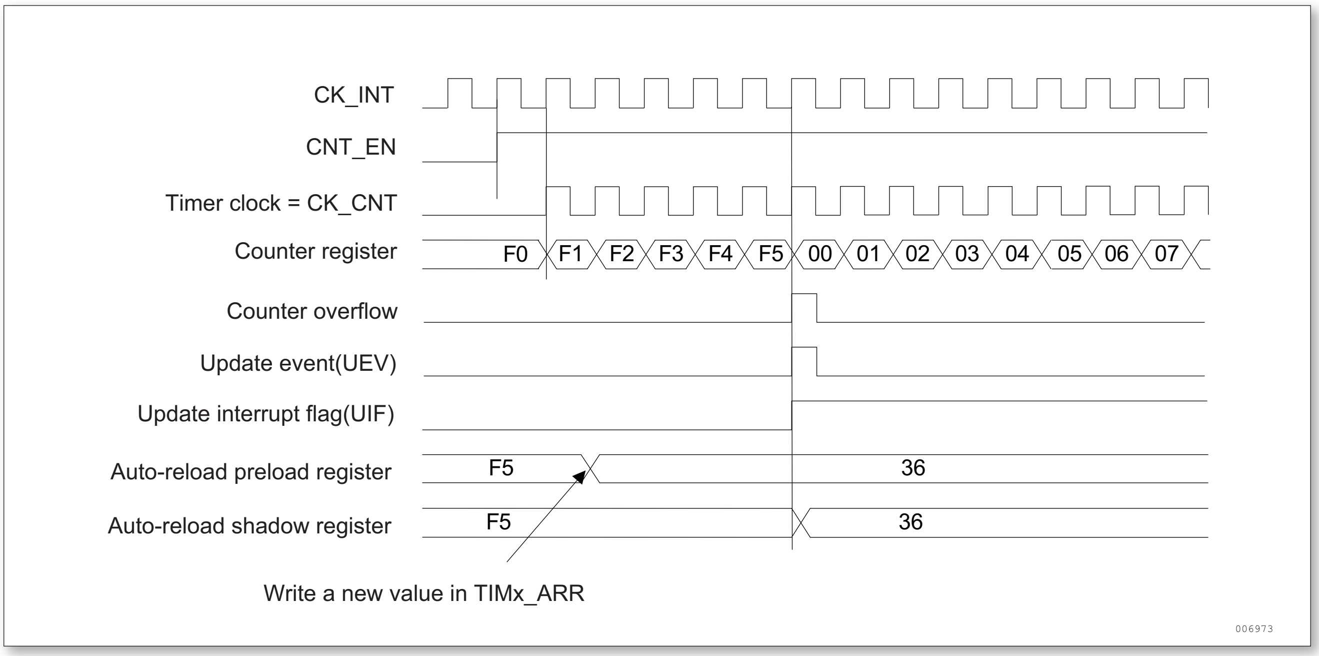


Figure 196. Counter Timing Diagram, Update Event When APRE=1 (TIMx_ARR Preloaded)

15.3.3 Repetition counter

Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows. It is actually generated only when the repetition counter has reached zero, which is very useful to generate PWM signal.

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

The repetition counter is decremented:

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

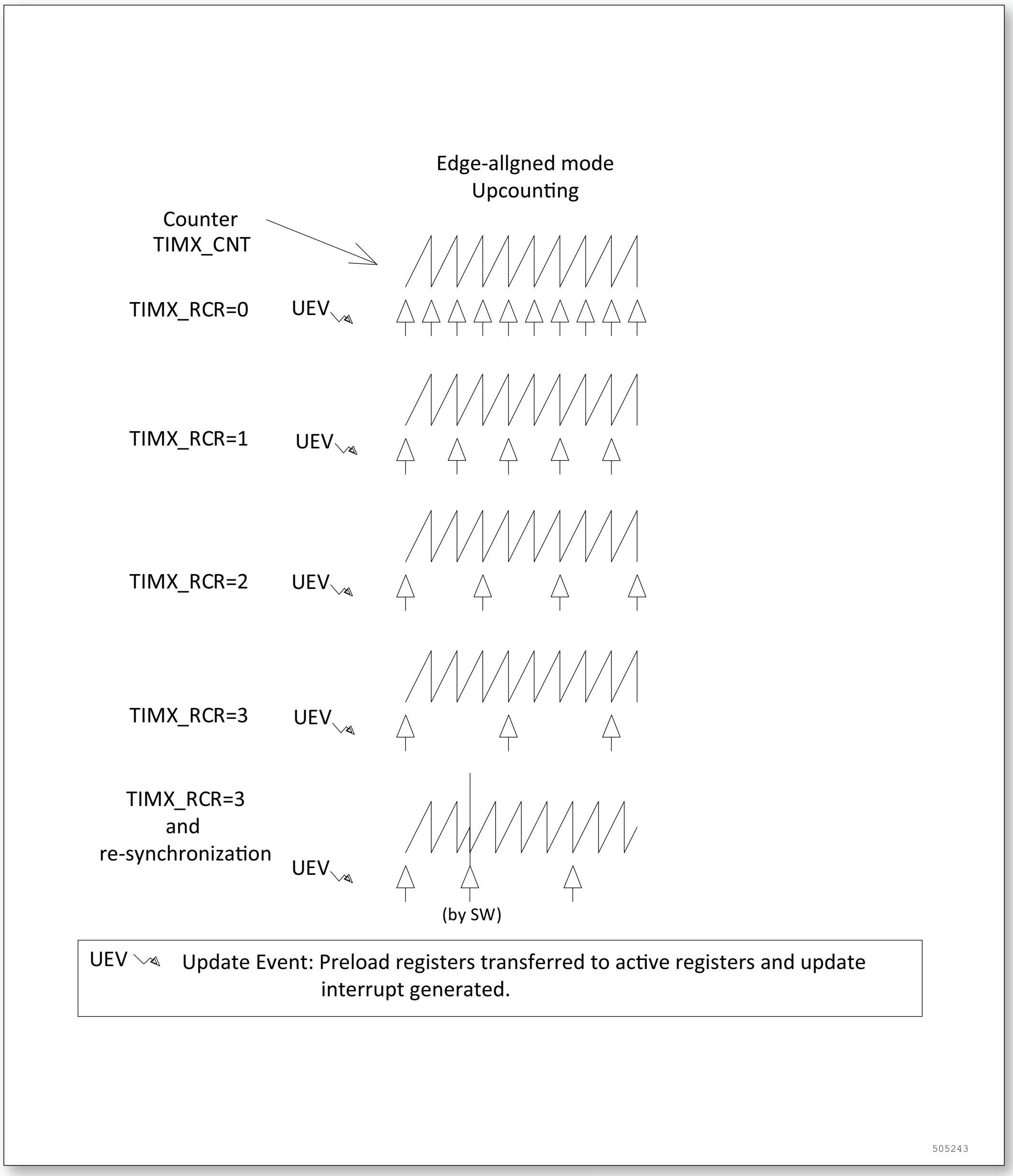


Figure 197. Example of Update Rates in Different Modes and Different TIMx_PCR Register Settings

15.3.4 Clock source

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

- Internal clock(CK_INT).

Internal clock source(CK_INT)

If the slave mode controller is disabled (SMS=000 in the TIMx_SMCR register), 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 following figure shows the behavior of the control circuit and the upcounter in normal mode, without prescaler.

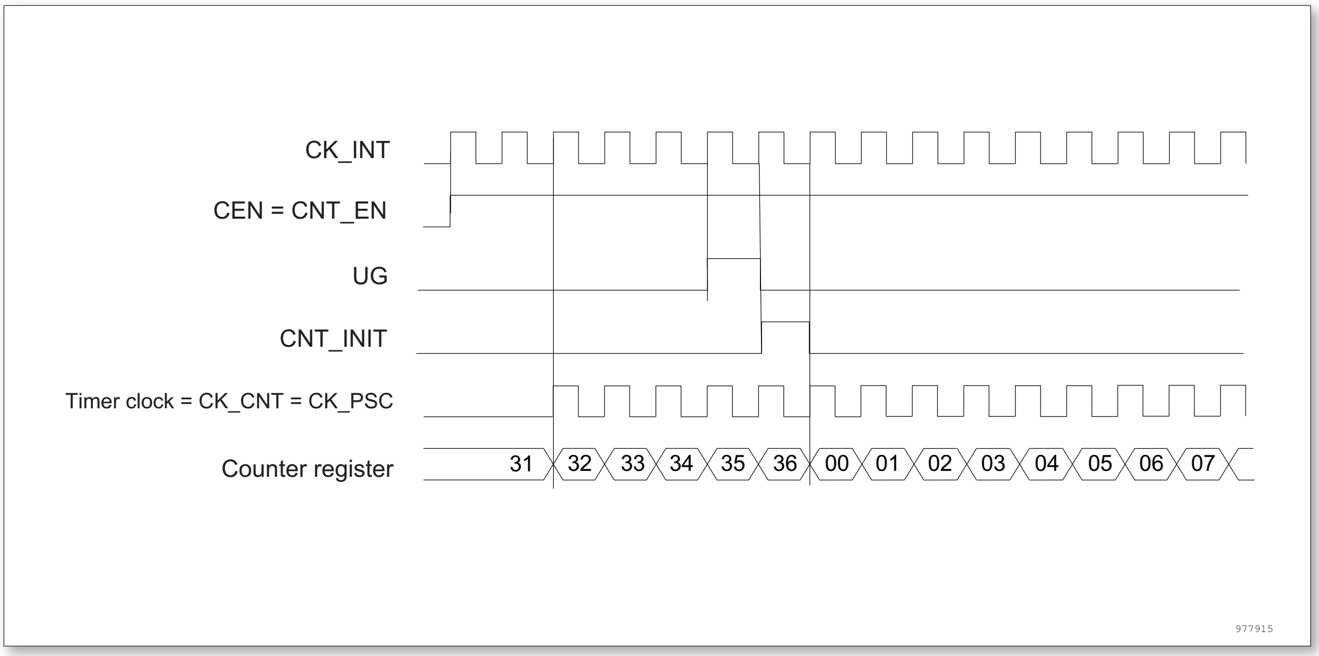


Figure 198. Control Circuit in Normal Mode, Internal Clock Divided By 1

15.3.5 Capture/compare channels

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

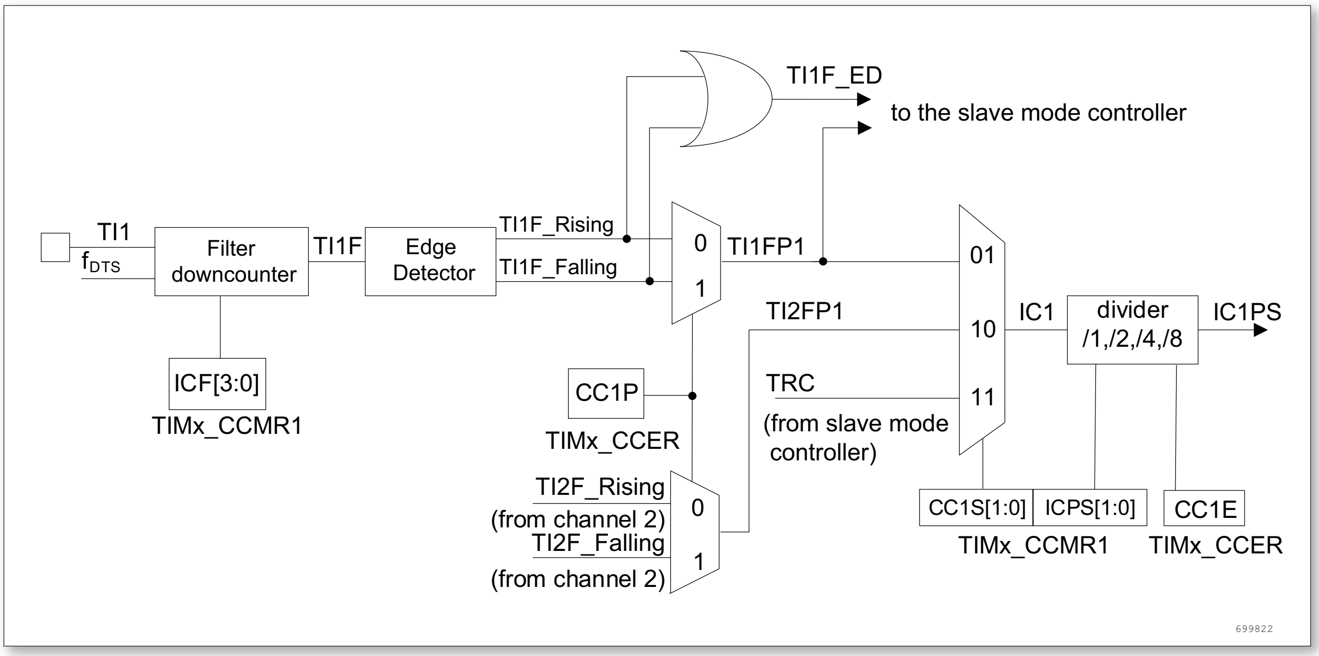


Figure 199. Capture/Compare Channel (Example: Channel 1 Input Stage)

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.

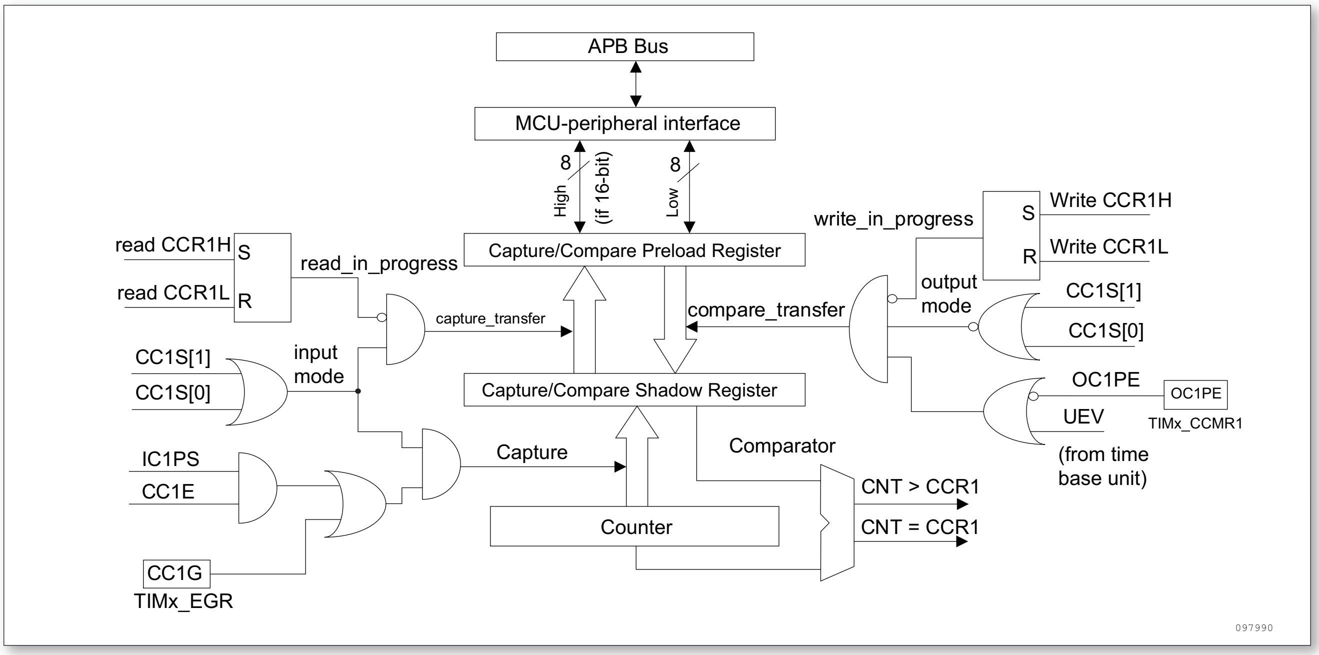


Figure 200. Capture/Compare Channel 1 Main Circuit

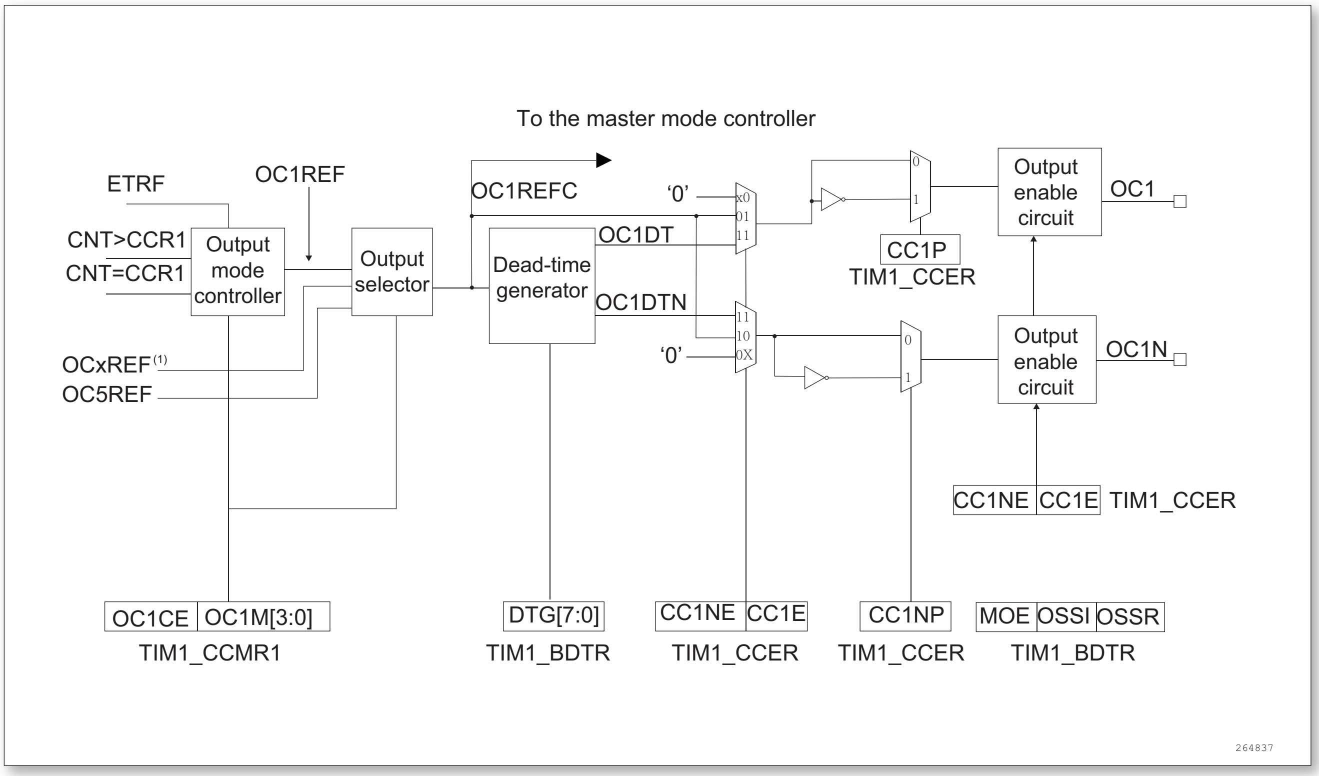


Figure 201. Output Stage of Capture/Compare Channel (Channel 1)

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.

15.3.6 Input capture mode

In Input capture mode, the Capture/Compare Registers (TIMx_CCRx) are used to latch the value of the counter after a transition detected by the corresponding ICx signal. When a capture occurs, the corresponding CCxIF flag (TIMx_SR register) is set and an interrupt or a DMA request can be sent if they are enabled. If a capture occurs while the CCxIF flag was already high, then the over-capture flag CCxOF (TIMx_SR register) is set, CCxIF can be cleared by software by writing it to 0 or by reading the captured data stored in the TIMx_CCRx register. CCxOF is cleared when written to 0.

The following example shows how to capture the counter value in TIMx_CCR1 when TI1 input rises.

Procedures:

- Select the active input: TIMx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TM1_CCR1 register becomes read-only.
- Program the needed input filter duration with respect to the signal connected to the timer (by programming ICxF bits in the TIMx_CCMRx register if the input is a TIx input). Let’s imagine that, when toggling, the input signal is not stable during at must five

internal clock cycles. We must program a filter bandwidth longer than these five clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been detected (sampled at fDTS frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.

- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMx_CCER register (rising edge in this case).
- Configure the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to '00' in the TIMx_CCMR1 register).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register to '1'.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

- The TIMx_CCR1 register gets the value of the counter on the active transition.
- C1IF flag is set (interrupt flag). CC1OF is also set if at least two consecutive captures occurred whereas the CC1IF flag was not cleared.
- An interrupt is generated depending on the CC1IE bit.
- A DMA request is generated depending on the CC1DE bit.

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

Note: IC interrupt and/or DMA requests can be generated by software by setting the corresponding CCxG bit in the TIMx_EGR register.

15.3.7 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 101 in the OCxM bits in the corresponding TIMx_CCMRx register. Thus OCxREF is forced high (OCxREF is always active high) and OCx get opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIMx_CCMRx register.

Anyway, in this mode, the comparison between the TIMx_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt and DMA requests can be sent accordingly. This is described in the output compare mode section below.

15.3.8 Output compare mode

This function is used to control an output waveform or indicating when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to a programmable value defined by the output compare mode (OCxM bits in the TIMx_CCMRx register) and the output polarity (CCxP bit in the TIMx_CCER register). The output pin can keep its level (OCxM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIMx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIMx_DIER register).
- Sends a DMA request if the corresponding enable bit is set (CCxDE bit in the TIMx_DIER register, CCDS bit in the TIMx_CR2 register for the DMA request selection).

The TIMx_CCRx registers can be programmed with or without preload registers using the OCxPE bit in the TIMx_CCMRx register. In output compare mode, the update event UEV has no effect on OCxREF and OCx output. The timing precision is one count of the counter. Output compare mode can also be used to output a single pulse (in One-pulse mode).

Procedure:

- Select the counter clock (internal, external, 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:
  - 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 OCx
- Enable the counter by setting the CEN bit in the TIMx_CR1 register.

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

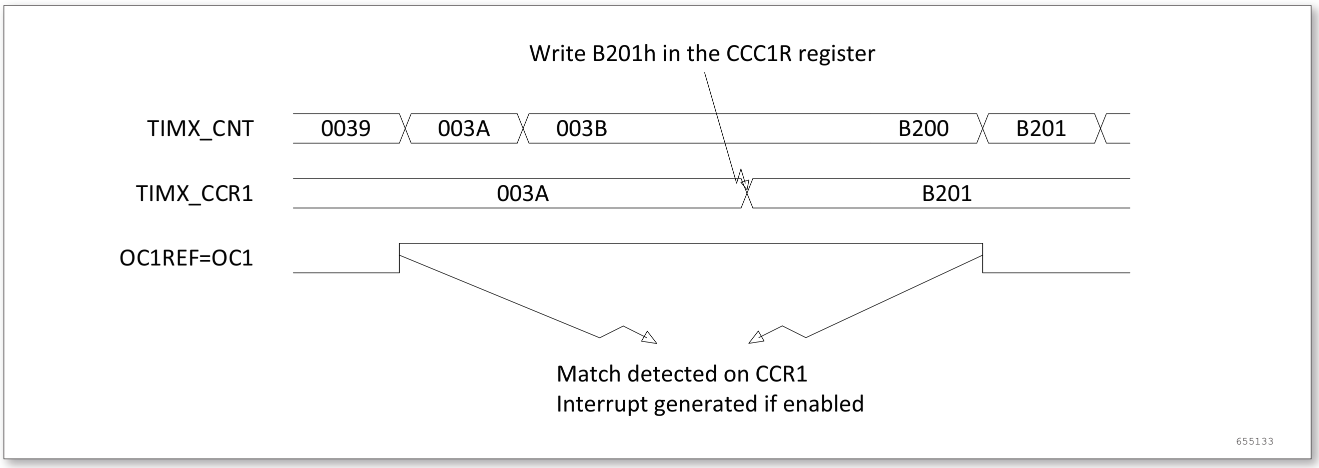


Figure 202. Output Compare Mode, Toggle on OC1

15.3.9 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 bits in the TIMx_CCMRx register. The corresponding preload register must be enabled by setting the OCxPE bit in the TIMx_CCMRx register, and eventually the auto-reload preload register (in upcounting mode) by setting the ARPE bit in the TIMx_CR1 register. As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, the user must initialize all the registers by setting the UG bit in the TIMx_EGR register.

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

In PWM mode (1 or 2), TIMx_CNT and TIMx_CCRx are always compared to determine whether TIMx_CCRx ≤TIMx_CNT or TIMx_CNT ≤TIMx_CCRx (depending on the direction of the counter). The timer is able to generate PWM in edge-aligned mode or center-aligned mode, depending on the CMS bits in the TIMx_CR1 register.

PWM edge-aligned mode

Upcounting configuration

Upcounting is active when the DIR bit in the TIMx_CR1 register is low. Refer to Section: Upcounting Mode. In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMx_CNT <TIMx_CCRx else it becomes low. If the compare value in TIMx_CCRx is greater than the auto-reload value (in TIMx_ARR) then OCxREF is held at ’1’. If the compare value is 0 then OCxREF is held at ’0’. Figure 203 shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

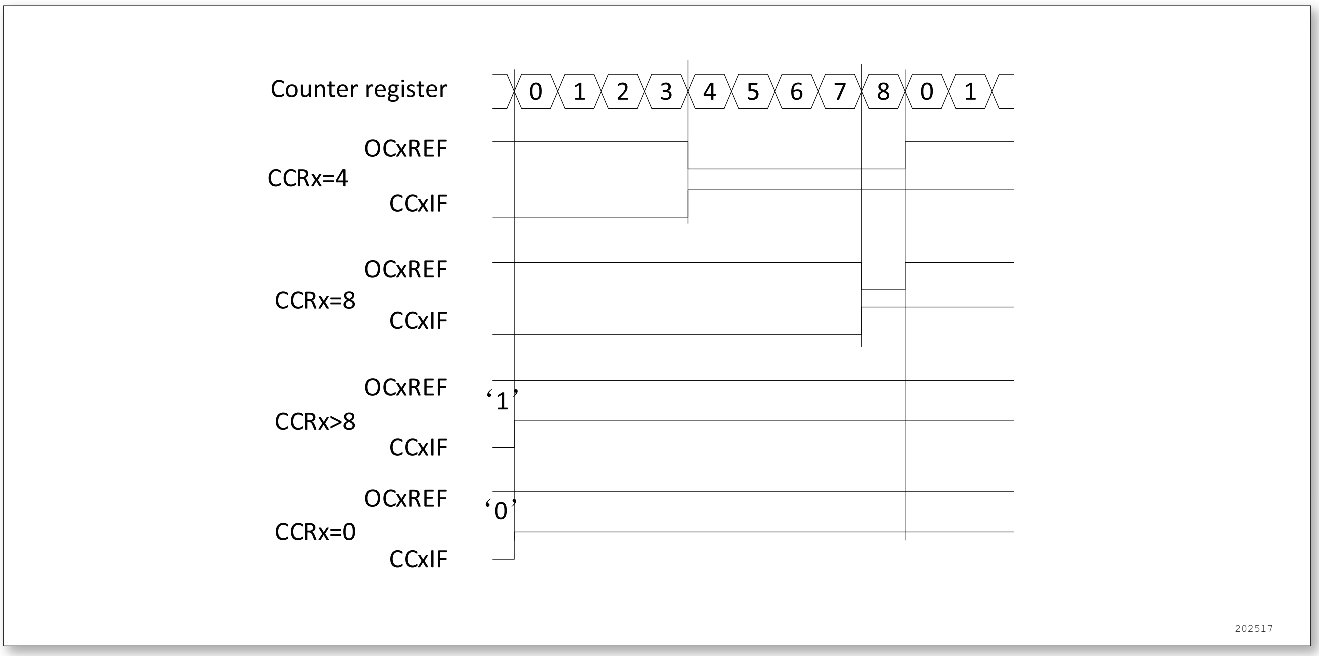


Figure 203. Edge-aligned PWM Waveforms (ARR=8)

15.3.10 Complementary outputs and dead-time insertion

The basic timers (TIM16/17) 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 outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches).

User can select the polarity of the outputs (main output OCx or complementary OCxN) independently for each output. This is done by writing to the CCxP and CCxNP bits in the TIMx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMx_BDTR and TIMx_CR2 registers. Refer to Table 52 Output Control Bits for Complementary OCx and OCxN Channels with Break Feature for more details. In particular, the dead-time is activated when switching to the IDLE state (MOE falling down to 0).

Dead-time insertion is enabled by setting both CCxE and CCxNE bits, and the MOE bit if the break circuit is present. Each channel is provided with a 10-bit dead-time generator. 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 opposite with 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).

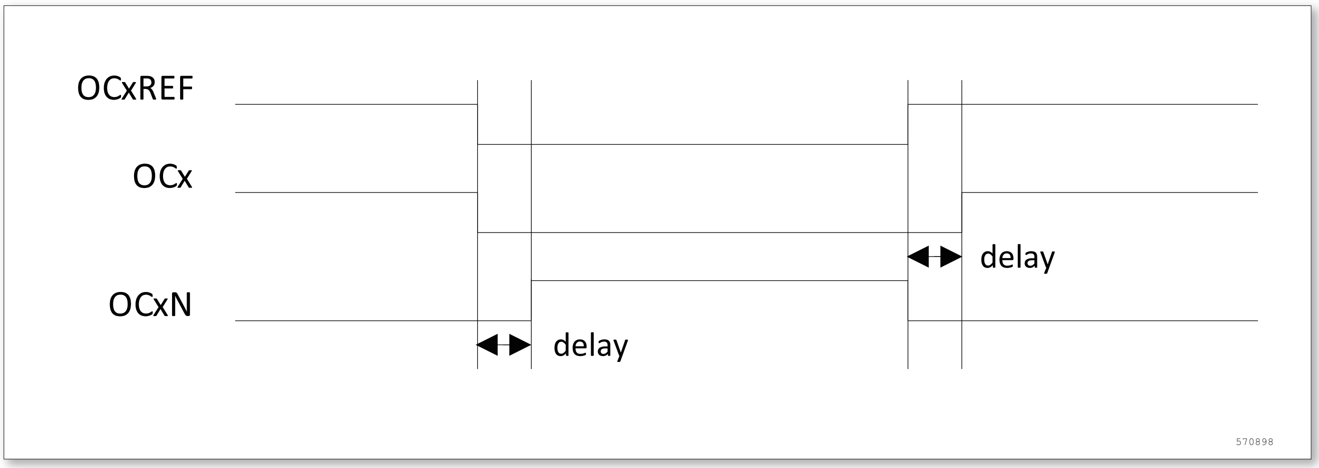


Figure 204. Complementary Output with Dead-time Insertion

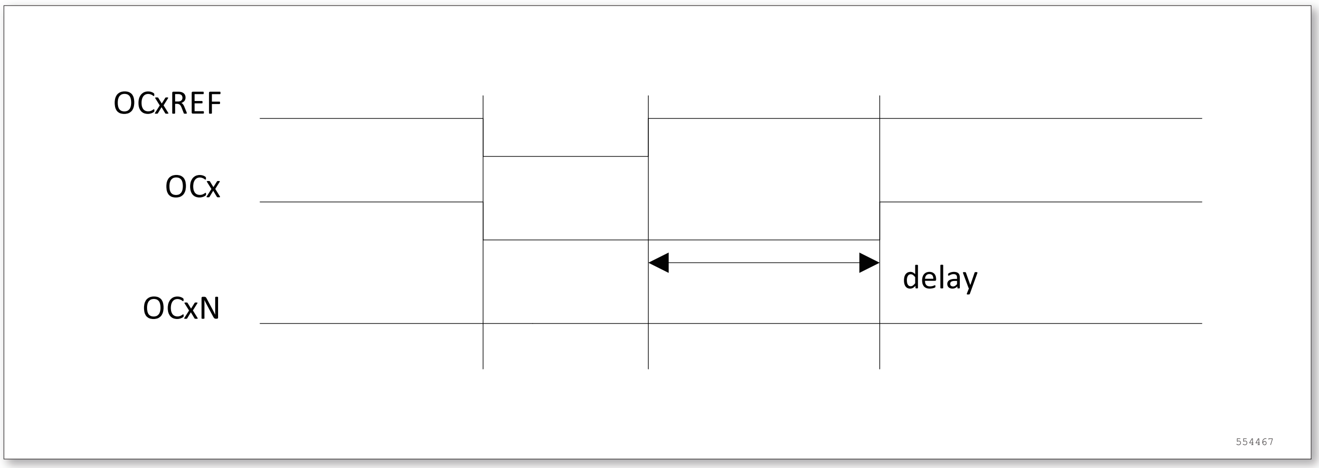


Figure 205. Dead-time Waveforms with Delay Greater Than the Negative Pulse

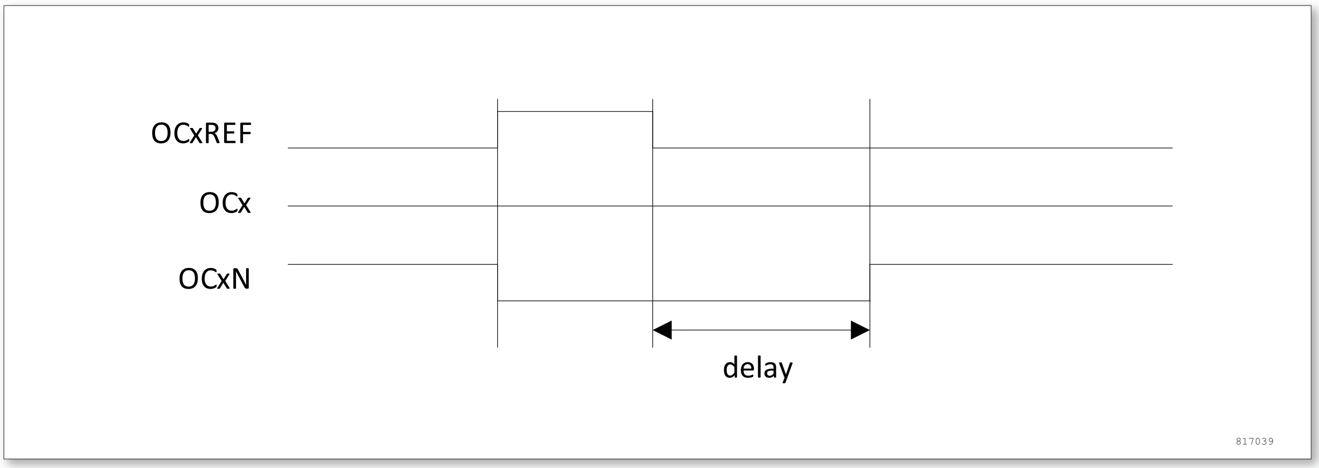


Figure 206. Dead-time Waveforms with Delay Greater Than the Positive Pulse

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 section 15.4.13: TIM16/17 Break and Dead-time Register (TIMx_BDTR) for delay calculation.

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.

15.3.11 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMx_BDTR register, OISx and OISxN bits in the TIMx_CR2 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 52 Output Control Bits for Complementary OCx and OCxN Channels with Break Feature for more details. 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. When exiting from 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. The delay of 1 APB clock period occurs before writing BKE and BKP bits. Therefore, the written bits can be read after one APB clock period.

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 programmed in the OISx bit in the TIMx_CR2 register as soon as MOE=0. If OSSI=0 then the timer releases the enable output else the enable output remains high.
- In case of complementary output:
  - The outputs are first put in reset state inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still present, then the dead-time generator is reactivated in

order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).

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

- The break status flag (BIF bit in the TIMx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMx_DIER register is set. A DMA request can be sent if the BDE bit in the TIMx_DIER register is set.
- If the AOE bit in the TIMx_BDTR register is set, the MOE bit is automatically set again at the next update event UEV. This can be used to perform a regulation, for instance. Else, MOE remains low until it is written 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 section 15.4.13: Break and Dead-time Register (TIM16/17_BDTR). The LOCK bits can be written only once after an MCU reset.

The following figure shows an example of behavior of the outputs in response to a break:

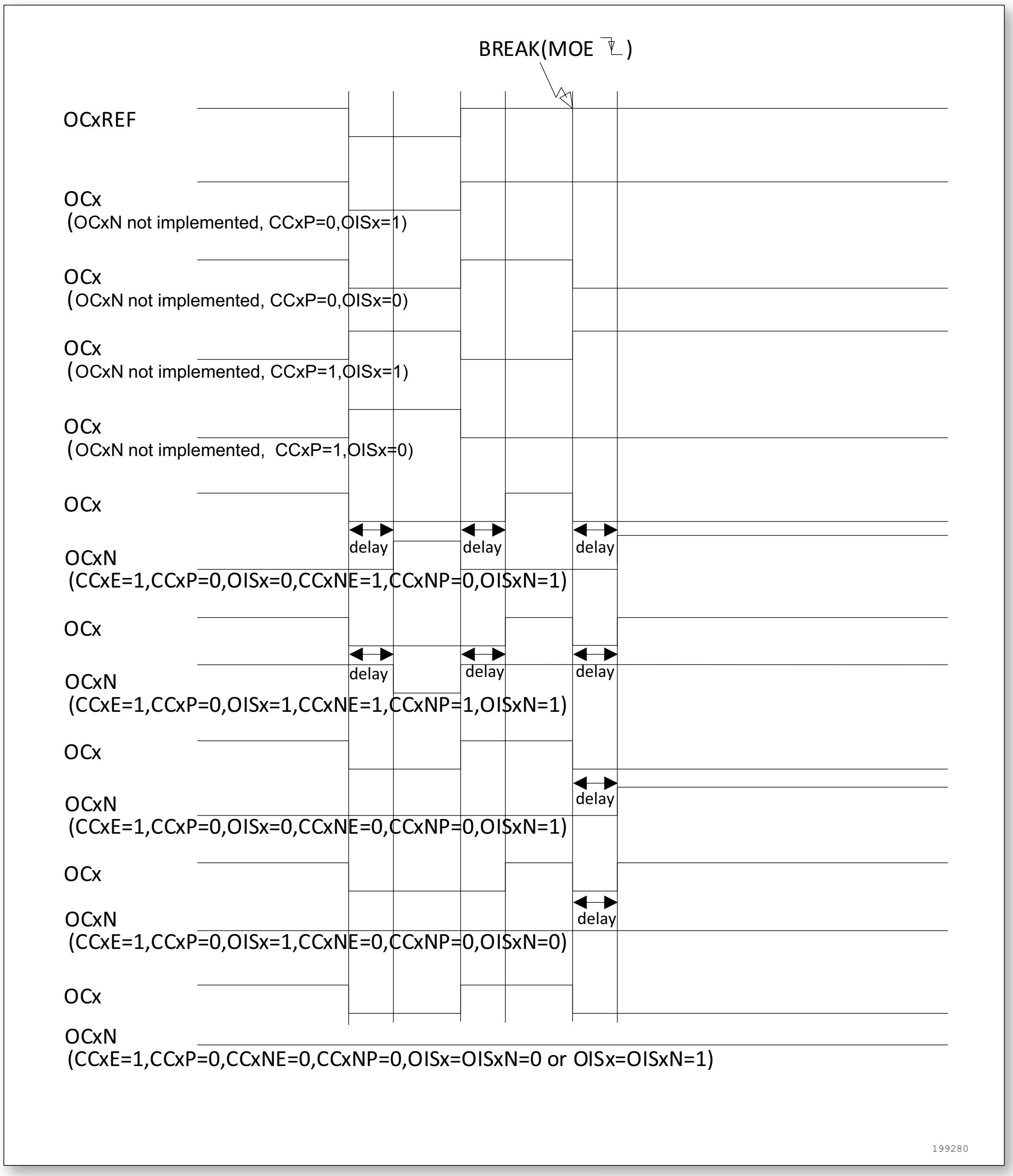

BREAK(MOE ⌐)
OCxREF
OCx
(OCxN not implemented, CCxP=0,OISx=1)
OCx
(OCxN not implemented, CCxP=0,OISx=0)
OCx
(OCxN not implemented, CCxP=1,OISx=1)
OCx
(OCxN not implemented, CCxP=1,OISx=0)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=0,CCxNE=1,CCxNP=0,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=1,CCxNE=1,CCxNP=1,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=0,CCxNE=0,CCxNP=0,OISxN=1)
OCx
OCxN
(CCxE=1,CCxP=0,OISx=1,CCxNE=0,CCxNP=0,OISxN=0)
OCx
OCxN
(CCxE=1,CCxP=0,CCxNE=0,CCxNP=0,OISx=OISxN=0 or OISx=OISxN=1)
delay
199280

Figure 207. Output Behavior in Response to A Break

15.3.12 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay. Starting the counter can be controlled through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. 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)

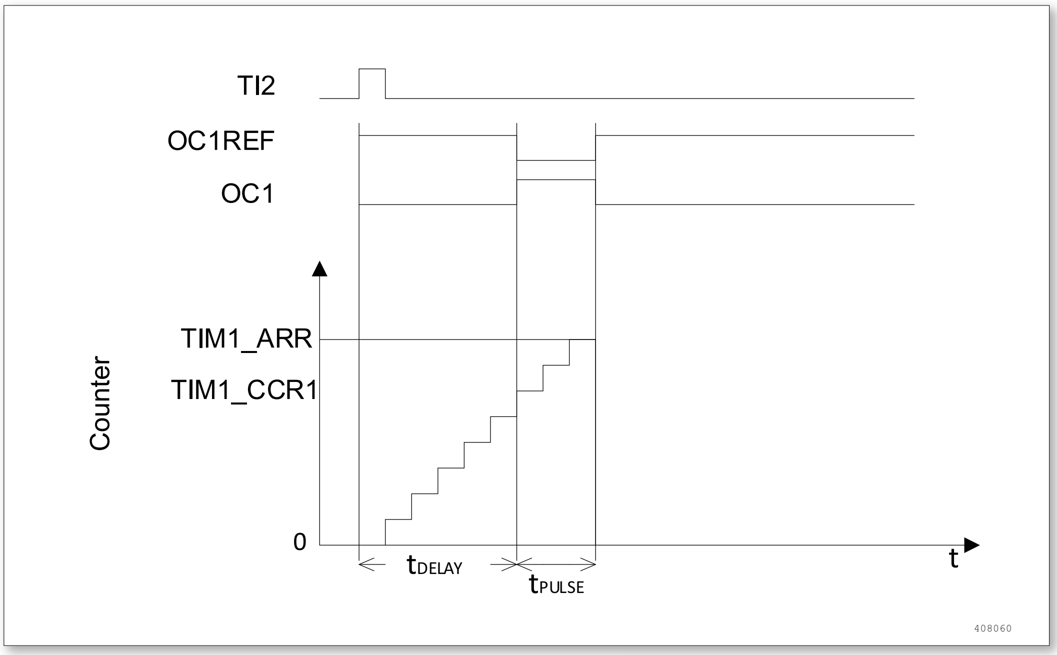


Figure 208. Example of One Pulse Mode

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

Let’s use TI2FP2 as trigger 1:

- Map TI2FP2 to TI2 by writing CC2S=’01’ in the TIMx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P=’0’ in the TIMx_CCER register.
- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=’110’ in the TIMx_SMCR register.
- TI2FP2 is used to start the counter by writing SMS to ‘110’ in the TIMx_SMCR register (trigger mode).
- The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler)
- The tDELAY is defined by the value written in the TIMx_CCR1 register.
- tPULSE is defined by the differrence between the auto-load value and the comparison 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 modifying the UG bit and wait for external trigger event on TI2. CC1P is written to ‘0’ in this example.

In our example, the DIR and CMS bits in the TIMx_CR1 register should be low.The user only wants one pulse (Single mode), so ’1’ must be written in the OPM bit in the TIMx_CR1 register to stop the counter at the next update event (when the counter rolls over from the auto-reload value back to 0). The repetition mode can be selected by setting OPM = ‘0’ in TIMx_CR1 register.

15.3.13 Debug mode

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

15.4 Register description

Table 51. TIM16/17 Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	TIM16/17_CR1	TIM16/17 control register 1	0x00000000	section 15.4.1
0x04	TIM16/17_CR2	TIM16/17 control register 2	0x00000000	section 15.4.2
0x0C	TIM16/17_DIER	TIM16/17 interrupt enable register	0x00000000	section 15.4.3
0x10	TIM16/17_SR	TIM16/17 status register	0x00000000	section 15.4.4
0x14	TIM16/17_EGR	TIM16/17 event generation register 1	0x00000000	section 15.4.5
0x18	TIM16/17_CCMR1	TIM16/17 capture/compare mode register 1	0x00000000	section 15.4.6
0x20	TIM16/17_CCER	TIM16/17 capture/compare enable register	0x00000000	section 15.4.7
0x24	TIM16/17_CNT	TIM16/17 counter	0x00000000	section 15.4.8
0x28	TIM16/17_PSC	TIM16/17 prescaler register	0x00000000	section 15.4.9
0x2C	TIM16/17_ARR	TIM16/17 auto-reload register	0x00000000	section 15.4.10
0x30	TIM16/17_RCR	TIM16/17 repetition counter register	0x00000000	section 15.4.11
0x34	TIM16/17_CCR1	TIM16/17 capture/compare register 1	0x00000000	section 15.4.12
0x44	TIM16/17_BDTR	TIM16/17 break and dead-time register	0x00000000	section 15.4.13
0x48	TIM16/17_DCR	TIM16/17 DMA control register	0x00000000	section 15.4.14
0x4C	TIM16/17_DMAR	TIM16/17 full transfer address register	0x00000000	section 15.4.15

15.4.1 TIM16/17 control register 1(TIM16/17_CR1)

Offset address: 0x00

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9: 8	CKD	rw	0x00	Clock division The 2 bits indicates the division ratio between the timer clock (CK_INT) frequency and the sampling frequency used by the digital filters (ETR, TIx). 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: Reserved
7	ARPE	rw	0x00	Auto-reload preload enable 0: TIMx_ARR register is not buffered 1: TIMx_ARR register is buffered
6: 4	Reserved			Reserved, always read as 0.
3	OPM	rw	0x00	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)
2	URS	rw	0x00	Update request source This bit is set and cleared by software to select the UEV. 0: Any of the following events generate an update interrupt or DMA request if enabled.These events can be: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
1	UDIS	rw	0x00	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller Buffered registers are then loaded with their preload values 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However, the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.

Bit	Field	Type	Reset	Description
0	CEN	rw	0x00	Counter enable 0: Counter disabled 1: Counter enabled. In the one pulse mode, CEN is automatically cleared when an update event occurs. Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However, trigger mode can set the CEN bit automatically by hardware.

15.4.2 TIM16/17 control register 2(TIM16/17_CR2)

Offset address: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						OIS1N	OIS1	Reserved				CCDS	CCUS	Res.	CCPC
						rw	rw					rw	rw		rw

Bit	Field	Type	Reset	Description
15:10	Reserved			Reserved, always read as 0.
9	OIS1N	rw	0x00	Output Idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=1 Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
8	OIS1	rw	0x00	Output Idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0. 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=1. Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).
7:4	Reserved			Reserved, always read as 0.
3	CCDS	rw	0x00	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs

Bit	Field	Type	Reset	Description
2	CCUS	rw	0x00	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only. 1: Capture/compare control bits are preloaded (CCPC = 1), they are updated only when COMG bit is set or rising edge is generated in TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved			Reserved, always read as 0.
0	CCPC	rw	0x00	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when COM is set Note: This bit acts only on channels that have a complementary output.

15.4.3 TIM16/17 interrupt enable register (TIM16/17_DIER)

Offset address: 0x0C

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CC1 DE	UDE	BIE	Res.	COM IE	Reserved			CC1 IE	UIE
						rw	rw	rw		rw				rw	rw

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9	CC1DE	rw	0x00	CC1 DMA request enabled 0: CC1 DMA request disabled 1: CC1 DMA request enabled
8	UDE	rw	0x00	Update DMA request enable 0: Update DMA request disabled 1: Update DMA request enabled
7	BIE	rw	0x00	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
6	Reserved			Reserved, always read as 0.
5	COMIE	rw	0x00	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4: 2	Reserved			Reserved, always read as 0.
1	CC1IE	rw	0x00	Capture/compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled

Bit	Field	Type	Reset	Description
0	UIE	rw	0x00	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

15.4.4 TIM16/17 interrupt enable register(TIM16/17_SR)

Offset address: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CC1 OF	Res.	BIF	TIF	COM IF	Reserved			CC1 IF	UIF
						rc_w0		rc_w0	rc_w0	rc_w0				rc_w0	rc_w0

Bit	Field	Type	Reset	Description
15: 10	Reserved			Reserved, always read as 0.
9	CC1OF	rc_w0	0x00	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode 1. It is cleared by software by writing it to ‘0’. 0: No overcapture has been detected. 1: The counter value has been captured in TIMx_CCR1 register while CC1IF flag was already set.
8	Reserved			Reserved, always read as 0.
7	BIF	rc_w0	0x00	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input
6	TIF	rc_w0	0x00	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode, both edges in case gated mode is selected). It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.

Bit	Field	Type	Reset	Description
5	COMIF	rc_w0	0x00	COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 0: No COM event occurred. 1: COM interrupt pending.
4: 2	Reserved			Reserved, always read as 0.
1	CC1IF	rc_w0	0x00	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match 1: The content of the counter TIMx_CNT matches the content of the TIMx_CCR1 register. If the content of TIMx_CCR1 is greater than that of TIMx_ARR, CC1IF flag becomes high in case of counter overflow. If channel CC1 is configured as input: This bit is set by hardware on an update event. It is cleared by software or by reading TIMx_CCR1. 0: No input capture occurred 1: The counter value has been captured (copied) in TIMx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	rc_w0	0x00	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - The update event is generated at overflow regarding the repetition counter value (update if repetition counter= 0) and if the UDIS=0 in the TIMx_CR1 register. - The update event is generated if URS=0 and UDIS=0 in the TIMx_CR1 register and CNT is reinitialized by setting UG bit in TIMx_EGR register through software. - The update event is generated when CNT is reinitialized by a trigger event, and if URS=0 and UDIS=0 in the TIMx_CR1 register.

15.4.5 TIM16/17 event generation register 1(TIM16/17_EGR)

Offset address: 0x14

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BG	TG	COMG	Reserved			CC1G	UG
								w	w	w				w	w

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7	BG	w	0x00	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOET bit is cleared and TIF in TIMx_SR register is set. Related interrupt or DMA transfer can occur if enabled.
6	TG	w	0x00	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIMx_SR register. Related interrupt or DMA transfer can occur if enabled.
5	COMG	w	0x00	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits Note: This bit acts only on channels that have a complementary output.
4:2	Reserved			Reserved, always read as 0.
1	CC1G	w	0x00	Capture/compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1 If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIMx_CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.

Bit	Field	Type	Reset	Description
0	UG	w	0x00	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Reinitialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected).

15.4.6 TIM16/17 capture/compare mode register 1(TIM16/17_CCMR1)

Offset address: 0x18

Reset value: 0x0000

The channels can be used in input (capture mode) or in output (compare mode). The direction of a channel is defined by configuring the corresponding CCxS bits. All the other bits of this register have a different function in input and in output mode. For a given bit, OCxx describes its function when the channel is configured in output, ICxx describes its function when the channel is configured in input. So the user must take care that the same bit can have a different meaning for the input stage and for the output stage.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								Res.	OC1M			OC1PE	OC1FE	CC1S	
								IC1F				IC1PSC			
								rw	rw	rw	rw	rw	rw	rw	rw

Output compare mode:

Bit	Field	Type	Reset	Description
15: 7	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
6: 4	OC1M	rw	0x00	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 are derived. OC1REF is active high whereas OC1 and OC1N active levels depend on CC1P and CC1PN bits respectively. 000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on OC1REF (only applicable to generating time base). 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMx_CNT matches the capture/compare register 1 (TIMx_CCR1). 011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIMx_CNT ＜ TIMx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF= ‘0’) as long as TIMx_CNT>TIMx_CCR1 else active (OC1REF=’ 1’). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMx_CNT<TIMx_CCR1 else active. In downcounting, channel 1 is active as long as TIMx_CNT>TIMx_CCR1 else inactive. Note 1: This bit is not writable as soon as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S= ’00’ (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OCREF level changes only when the result of the comparison changes or when the output compare mode switches from “frozen” mode to “PWM” mode.

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x00	Output compare 1 preload enable 0: Preload register on TIMx_CCR1 disabled. TIMx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMx_CCR1 enabled. Read/Write operations access the preload register. TIMx_CCR1 preload value is transferred to the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMx_BDTR register) and CC1S=00 (the channel is configured in output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIMx_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	rw	0x00	Output compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1: 0	CC1S	rw	0x00	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIMx_CCER).

Input capture mode:

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7: 4	IC1F	rw	0x00	Input capture 1 ?lter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at f_{DTS} 0001: Sampling frequency $f_{SAMPLING} = f_{CK\_INT}$, N = 2 0010: Sampling frequency $f_{SAMPLING} = f_{CK\_INT}$, N = 4 0011: Sampling frequency $f_{SAMPLING} = f_{CK\_INT}$, N = 8 0100: Sampling frequency $f_{SAMPLING} = f_{DTS}$, N = 6 0101: Sampling frequency $f_{SAMPLING} = f_{DTS}$, N = 8 0110: Sampling frequency $f_{SAMPLING} = f_{DTS}$, N = 6 0111: Sampling frequency $f_{SAMPLING} = {}_{DTS}/4$, 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 $f_{SAMPLING} = ff_{DTS}/16$, 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	rw	0x00	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=’0’(TIMx_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input. 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events

Bit	Field	Type	Reset	Description
1: 0	CC1S	rw	0x00	Capture/compare 1 This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIMx_SMCR register) Note: CC1S bits are writable only when the channel is OFF (CC1E = ’0’ in TIMx_CCER).

15.4.7 TIM16/17 Capture/compare enable register(TIM16/17_CCER)

Offset address: 0x20

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CC1 NP	CC1 NE	CC1P	CC1E
												rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 4	Reserved			Reserved, always read as 0.
3	CC1NP	rw	0x00	Capture/Compare 1 complementary output Polarity 0: OC1N active high 1: OC1N active low
2	CC1NE	rw	0x00	Capture/Compare 1 complementary output enable 0: Off - OC1N inactive 1: On - Signal of OC1N output to relevant pin depends on MOE, OSSI, OSSR, OIS1, OIS1 and CC1E bits.

Bit	Field	Type	Reset	Description
1	CC1P	rw	0x00	Capture/compare 1 output polarity If channel CC1 is configured as output: 0: OC1 active high 1: OC1 active low CC1 channel is configured as input: CC1P/CC1NP bit (IC1 or inverted IC1) is used to select the polarity of TI1FP1 and TI2FP1 as trigger or capture. 00: non-inverted/rising edge: capture is done on a rising edge of TIxFP1 (capture mode), and TIxFP1 is non-inverted; 01: inverted/falling edge: capture is done on a falling edge of TIxFP1 (capture mode), and TIxFP1 is inverted; 10: Reserved, this configuration is not used 11: non-inverted/rising edge: capture is done on the rising and falling edges of non-inverted TIxFP1 (capture mode).
0	CC1E	rw	0x00	Capture/Compare 1 output enable CC1 channel is configured as output: 0: Off - OC1 is not active 1: On - OC1N signal output to relevant pin depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. CC1 channel is configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not. 0: Capture disabled. 1: Capture enabled.

Table 52. Output Control Bits for Complementary OCx and OCxN Channels with Break Feature

Control bits					Output states[1]	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx OCx output state	OCxN output state
		0	0	0	Output Disabled (not driven by the timer) OCx = 0, OCx_EN = 0	Output Disabled (not driven by the timer) OCxN = 0, OCxN_EN = 0
		0	0	1	Output Disabled (not driven by the timer) OCx = 0, OCx_EN = 0	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		0	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Output Disabled (not driven by the timer) OCxN = 0, OCxN_EN = 0
1	X					

Control bits					Output states[1]	
MOE bit	**OSSI bit**	**OSSR bit**	**CCxE bit**	**CCxNE bit**	**OCx OCx output state**	**OCxN output state**
		0	1	1	OCxREF + Polarity + dead-time, OCx_EN=1	OCxREF (inverted) + Polarity + dead-time, OCxN_EN = 1
		1	0	0	Output Disabled (not driven by the timer) OCx = CCxP, OCx_EN = 0	Output Disabled (not driven by the timer), OCxN = CCxNP, OCxN_EN = 0
		1	0	1	Off-State (output enabled with inactive state) OCx = CCxP, OCx_EN = 1	OCxREF + Polarity, OCxN = OCxREF xor CCxNP, OCxN_EN = 1
		1	1	0	OCxREF + Polarity, OCx = OCxREF xor CCxP, OCx_EN = 1	Off-State (output enabled with in-active state) OCxN = CCxNP, OCxN_EN = 1
		1	1	1	OCREF + Polarity + dead-time, OCx_EN = 1	OCxREF (inverted) + Polarity + dead-time, OCxN_EN = 1
0	0	X	0	0	Output Disabled (not driven by the timer) Asynchronously: OCx = CCxP , OCx_EN = 0 , OCxN = CCxNP, OCxN_EN = 0； Then if the clock is present: after a dead-time OCx = OISx , OCxN = OISxN, Assuming that OISx and OISxN do not correspond to OCx and OCxN both in activ	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	Off-State (output enabled with inactive state) Asynchronously: OCx = CCxP, OCx_EN = 1, OCxN = CCxNP, OCxN_EN = 1； Then 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 activ	
	1		0	1		
	1		1	0		
	1		1	1		

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

Note: The state of the external IO pins connected to the omplementary OCx and OCxN channels depends on the OCx and OCxN channel states and the GPIO and AFIO registers.

15.4.8 TIM16/17 counter(TIM16/17_CNT)

Offset address: 0x24

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CNT	rw	0x0000	Counter value

15.4.9 TIM16/17 prescaler register(TIM16/17_PSC)

Offset address: 0x28

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	PSC	rw	0x0000	Prescaler value The counter clock frequency (CK_CNT) is equal to f_{CK_PSC} /(PSC + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in ”reset mode”)

15.4.10 TIM16/17 auto-reload register(TIM16/17_ARR)

Offset address: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	ARR	rw	0x0000	Prescaler value ARR is the value to be loaded in the actual auto-reload register. Refer to section 15.3.1 for more details about ARR update and behavior. The counter is blocked while the auto-reload value is null.

15.4.11 TIM16/17 repetition counter register(TIM16/17_RCR)

Offset address: 0x30

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 8	Reserved			Reserved, always read as 0.
7: 0	REP	rw	0x00	Repetition counter value These bits allow the user to set up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related upcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event U_RC, any write to the TIMx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP + 1) corresponds to the number of PWM periods.

15.4.12 TIM16/17 Capture/compare register 1(TIM16/17_CCR1)

Offset address: 0x34

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	CCR1	rw	0x0000	Capture/Compare 1 value If CC1 channel is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

15.4.13 TIM16/17 break and dead-time register(TIM16/17_BDTR)

Offset address: 0x44

Reset value: 0x0000

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	rw	rw	rw	rw	rw	rw	rw	rw

Note: As the bits AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] can be write-locked depending on the LOCK configuration, it can be necessary to configure all of them during the first write access to the TIMx_BDTR register.

Bit	Field	Type	Reset	Description
15	MOE	rw	0x00	Main output enable This bit is cleared asynchronously by hardware as soon as the break input is active. It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in TIMx_CCER register). See section 15.4.7: TIM16/17 Capture/compare Enable Register (TIM16/17_CCER) for more details.
14	AOE	rw	0x00	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if the break input is not be active) Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register).
13	BKP	rw	0x00	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high Note 1: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register). Note 2: This bit can only be written after the delay of one APB clock.
12	BKE	rw	0x00	Break enable 0: Break inputs (BRK and CSS clock failure event) disabled 1: Break inputs (BRK and CSS clock failure event) enabled Note 1: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMx_BDTR register). Note 2: This bit can only be written after the delay of one APB clock.

Bit	Field	Type	Reset	Description
11	OSSR	rw	0x00	Off-state selection for Run mode This bit is used when MOE=1 on channels configured as complementary outputs. OSSR is not implemented if no complementary output is implemented in the timer. See OC/OCN enable description for more details (section 15.4.7: capture/compare enable register (TIMx_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are enabled and inactive level is output as soon as CCxE=1 or CCxNE=1, and then set OC/OCN enable output signal to 1. Note: This bit can not be modified as long as LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).
10	OSSI	rw	0x00	Off-state selection fo Idle mode This bit is used when MOE=0 on channels configured as outputs. See OC/OCN enable description for more details (section 15.4.7: capture/compare enable register (TIMx_CCER)). 0: When inactive, OC/OCN outputs are disabled (OC/OCN enable output signal=0). 1: When inactive, OC/OCN outputs are enabled and inactive level is output as soon as CCxE=1 or CCxNE=1, and then set OC/OCN enable output signal to 1 Note: This bit can not be modified as long as LOCK level 2 has been programmed (LOCK bits in TIMx_BDTR register).

Bit	Field	Type	Reset	Description
9: 8	LOCK	rw	0x00	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write-protected. 01: LOCK Level 1 = DTG, BKE, BKP, AOE bits in TIMx_BDTR register, and OISx/OISxN bits in TIMx_CR2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMx_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in TIMx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. Note: The LOCK bits can be written only once after the reset. Once the TIMx_BDTR register has been written, their content is frozen until the next reset.

7: 0	DTG	rw	0x00	Dead-time generator setup
				This bit-field defines the duration of the dead-time inserted between the complementary outputs. It is assumed that DT corresponds to this duration.
				DTG[7: 5] = 0xx:
				DT = (DTG[7: 0] + 1) × t_{dtg}, t_{dtg} = t_{DTS};
				DTG[7: 5] = 10x:
				DT = (DTG[5: 0] + 1 + 64) × t_{dtg}, t_{dtg} = 2 × t_{DTS};
				DTG[7: 5] = 110:
				DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 8 × t_{DTS};
				DTG[7: 5] = 111:
				DT = (DTG[4: 0] + 1 + 32) × t_{dtg}, t_{dtg} = 16 × t_{DTS};
				Example if t_{DTS} = 125ns(8MHz), dead-time possible values are:
				125ns to 15875ns by 125 nS steps;
				16µs to 31750ns by 250 nS steps;
				32µs to 63µs by 1 µs steps;
				64µs to 126µs by 2 µs steps;
				Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMx_BDTR register).

15.4.14 TIM16/17 DMA DMA control register(TIM16/17_DCR)

Offset address: 0x48

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 13	Reserved			Reserved, always read as 0.
12: 8	DBL	w	0x00	DMA burst length This bit field defines the burst transfer in the continuous mode (the timer detects a burst transfer when a read or a write access to the TIMx_DMAR register address is performed), namely, the number of transfers: 00000: 1 byte 00001: 2 bytes 00010: 3 bytes .... 10001: 18 bytes
7: 5	Reserved			Reserved, always read as 0.
4: 0	DBA	w	0x00	DMA base address These bits define the base-address for DMA transfers (when read/write access are done through the TIMx_DMAR address). DBA is defined as an offset starting from the address. Example: 00000: TIMx_CR1 00001: TIMx_CR2 00010: TIMx_SMCR ...... Example: To complete the following transfer: DBL = 7 , DBA = TIMx_CR1, at this time, the transfer starts from the address of TIMx_CR1 to/from 7 consecutive registers.

15.4.15 TIM16/17 address for full transfer(TIM16/17_DMAR)

Offset address: 0x4C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15: 0	DMAB	w	0x0000	DMA register for burst accesses A read or write operation to the TIMx_DMAR register accesses the register located at the following address: (Address of TIMx_CR1) + (DBA + DMA index) x 4, where, TIMx_CR1 address is the address of the control register 1 (TIMx_CR1); DBA is the DMA base address configured in TIMx_DCR register; DMA index is the offset automatically controlled by the DMA transfer, depending on DBL configured in TIMx_DCR.

Example of how to use DMA concurrent operation:

In this example, the concurrency function of the timer DMA is used, to transfer the contents of the CCRx register to the CCRx register in half words. The procedures are as follows:

1. Configure the relevant DMA channels:
   (a) The device address of DMA channel is DMAR register address
   (b) The memory address of DMA channel covers the RAM buffer address of data transferred to CCRx register by DMA.
   (c) Data transferred = 3 (see the following Note)
   (d) Notification mode disabled
2. Configure the DBA and DBL bits of the DCR register: DBL = 3 transfers, DBA = 0xE.
3. Enable TIMx Update DMA Request (set UDE bit in DIER register)
4. Enable TIMx
5. Enable DMA channel

Note: In this example, all CCRx registers are updated all at one time. If the CCRx register needs to be updated twice, the data transferred shall be 6, and the RAM buffer zone shall contain data1, data2, data3, data4, data5 and data6. The data is transferred to the CCRx register as follows: in case of the first DMA update request, data1 is transferred to CCR2, data2 transferred to CCR3, and data3 transferred to CCR4; data4 is transferred to CCR2 in case of the second DMA update interrupt request, data5 transferred to CCR3 and data6 transferred to CCR4.

16 Independent watchdog(IWDG)

Independent watchdog(IWDG)

16.1 (IWDG introduction)

The devices have two embedded watchdog peripherals which offer a combination of high safety level, timing accuracy and flexibility of use. Both watchdog peripherals (Independent and Window) serve to detect and resolve malfunctions due to software failure, and to trigger system reset or an interrupt (window watchdog only) when the counter reaches a given timeout value.

The independent watchdog (IWDG) is clocked by its own dedicated low-speed clock (LSI) and thus stays active even if the main clock fails. The window watchdog (WWDG) clock is prescaled from the APB1 clock and has a configurable time-window that can be programmed to detect abnormally late or early application behavior.

The IWDG 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 WWDG is best suited to applications which require the watchdog to react within an accurate timing window.

16.2 IWDG main features

- Free-running downcounter
- Clocked from an independent RC oscillator (can operate in Standby and Stop modes)
- Reset (if watchdog activated) when the downcounter value of 0x000 is reached

16.3 Functional description

The following figure shows the functional blocks of the independent watchdog module.

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

Whenever the key value 0xAAAA is written in the IWDG_KR register, the IWDG_RLR value is reloaded in the counter and the watchdog reset is prevented.

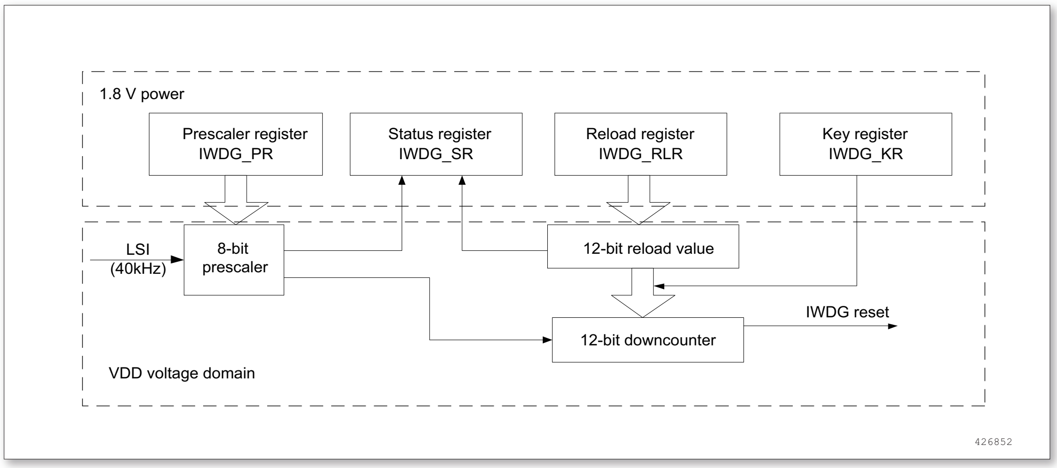


Figure 209. Independent Watchdog Block Diagram

Note: The watchdog function is implemented in the V_{DD} voltage domain, still functional in Stop and Standby modes.

Table 53. Min/max IWDG Timeout Period (in ms) at 40 kHz (LSI)

Prescaler divider	PR [2:0] bits	Min timeout RL[11:0] = 0x000	Max timeout
/4	0	0.1	409.6
/8	1	0.2	819.2
/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 timings are given for a 40 kHz clock but the microcontroller internal RC frequency can vary between 30 kHz -60 kHz.
In addition, even if the frequency of the oscillator is accurate, the exact timing still depends on the phase difference between the APB interface clock and the oscillator clock.
As a result, there will always be a complete oscillator period that is uncertain.

16.3.1 Hardware watchdog

If the user activates the ’Hardware Watchdog’ function in the select bit (refer to the section ”Embedded Flash”), the watchdog will automatically run after the system power-on reset; if the software does not write the corresponding value to the key register before the counter ends counting, the system reset will occur.

16.3.2 Register access protection

The IWDG_PR and IWDG_RLR registers are write-protected. To modify the values of these two registers, you must first write 0x5555 to the IWDG_KR register. Writing to this register with a different value will disrupt the sequence and the registers will be reprotected. The reload operation (i.e. writing 0xAAAA) will also activate the write protection.

The status register indicates whether the prescaler value and the downcounter are being updated.

16.3.3 Debug mode

When the microcontroller enters debug mode (CPU core halted), the IWDG counter either continues to work normally or stops, depending on DBG_IWDG_STOP configuration bit in DBG module. For more details, refer to sections of debug modules.

16.4 IWDG register description

Table 54. Overview of IWDG Registers

Offset	Acronym	Register Name	Reset	Section
0x00	IWDG_KR	Key register	0x00000000	section 16.4.1
0x04	IWDG_PR	Prescaler register	0x00000000	section 16.4.2
0x08	IWDG_RLR	Reload register	0x00000FFF	section 16.4.3
0x0C	IWDG_SR	Status register	0x00000000	section 16.4.4
0x10	IWDG_CR	Control register	0x00000000	section 16.4.5

16.4.1 Key register(IWDG_KR)

Offset address: 0x00

Reset value: 0x0000 0000(reset by Standby mode)

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

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

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.
15 : 0	KEY	w	0x0000	Key value (write only, read 0x0000) These bits must be written by software at regular intervals with the key value 0xAAAA, otherwise the watchdog generates a reset when the counter reaches 0. Writing the key value 0x5555 to enable access to the IWDG_PR and IWDG_RLR registers. Writing the key value 0xCCCC starts the watchdog.

16.4.2 Prescaler register(IWDG_PR)

Offset address: 0x04

Reset value: 0x0000 0000(reset by Standby mode)

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													PR		
													rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 3	Reserved			Always read as 0.
2 : 0	PR	rw	0x00	Prescaler divider These bits are write access protected. They are written by software to select the prescaler divider feeding the counter clock. PVU bit of IWDG_SR must be reset in order to be able to change the prescaler divider. 000: divider / 4 100: divider / 64 001: divider / 8 101: divider / 128 010: divider / 16 110: divider / 256 011: divider / 32 111: divider / 256 Note: Reading this register returns the prescaler value from the V_{DD} voltage domain. This value may not be up to date/valid if a write operation to this register is ongoing. For this reason, the value read from this register is valid only when the PVU bit in the IWDG_SR register is reset.

16.4.3 Reload register(IWDG_RLR)

Offset address: 0x08

Reset value: 0x0000 0FFF(reset by Standby mode)

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

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

Bit	Field	Type	Reset	Description
31 : 12	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
11 : 0	RL	rw	0xFFF	Watchdog counter reload value These bits are write access protected. They are written by software to define the value to be loaded in the watchdog counter each time the value 0xAAAA is written in the IWDG_KR register. The watchdog counter counts down from this value. The timeout period is a function of this value and the clock prescaler. Note: Reading this register returns the reload value from the VDD voltage domain. This value may not be up to date/valid if a write operation to this register is ongoing on this register. For this reason, the value read from this register is valid only when the RVU bit in the IWDG_SR register is reset.

16.4.4 Status register(IWDG_SR)

Offset address: 0x0C

Reset value: 0x0000 0000 (not reset by Standby mode)

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														RVU	PVU
														r	r

Bit	Field	Type	Reset	Description
31 : 2	Reserved			Always read as 0.
1	RVU	r	0x00	Watchdog counter reload value update This bit is set by hardware to indicate that an update of the reload value is ongoing. It is reset by hardware when the reload value update operation is completed in the V_{DD} voltage domain (takes up to 5 RC 40 kHz cycles). Reload value can be updated only when RVU bit is reset.
0	PVU	r	0x00	Watchdog prescaler value update This bit is set by hardware to indicate that an update of the prescaler value is ongoing. It is reset by hardware when the prescaler update operation is completed in the V_{DD} voltage domain (takes up to 5 RC 40 kHz cycles). Prescaler value can be updated only when PVU bit is reset.

Note: If several reload values or prescaler values are used by application, it is mandatory to wait until RVU bit is reset before changing the reload value and to wait until PVU bit is reset before changing the prescaler value. However, after updating the prescaler and/or the reload value it is not necessary to wait until RVU or PVU is reset before continuing code execution (even in case of low-power mode entry, the write operation is taken into account and will complete).

16.4.5 IWDG Control register(IWDG_CR)

Offset address: 0x10

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31:2	Reserved			Always read as 0.
1	IRQ_CLR	rw	0x00	IWDG interrupt clear 1: Write 1 clear interrupt 0: No operation
0	IRQ_SEL	rw	0x00	IWDG overflow operation selection 1: Interrupt after overflow 0: Reset after overflow

17 Window watchdog(WWDG)

Window watchdog(WWDG)

17.1 WWDG introduction

The window watchdog is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared. An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value. This implies that the counter must be refreshed in a limited window.

17.2 WWDG main features

- Programmable free-running downcounter
- Conditional reset:
  - Reset (if watchdog activated) when the downcounter value becomes less than 0x40
  - Reset (if watchdog activated) if the downcounter is reloaded outside the window
- Early wakeup interrupt (EWI): triggered (if enabled and the watchdog activated) when the downcounter is equal to 0x40. It is used to reload the counter, thus preventing WWDG reset.

17.3 Functional description

If the watchdog is activated (the WDGA bit is set in the WWDG_CR register) and when the 7-bit downcounter (T[6:0] bits) rolls over from 0x40 to 0x3F (T6 becomes cleared), it initiates a reset. If the software reloads the counter while the counter is greater than the value stored in the window register, then a reset is generated.

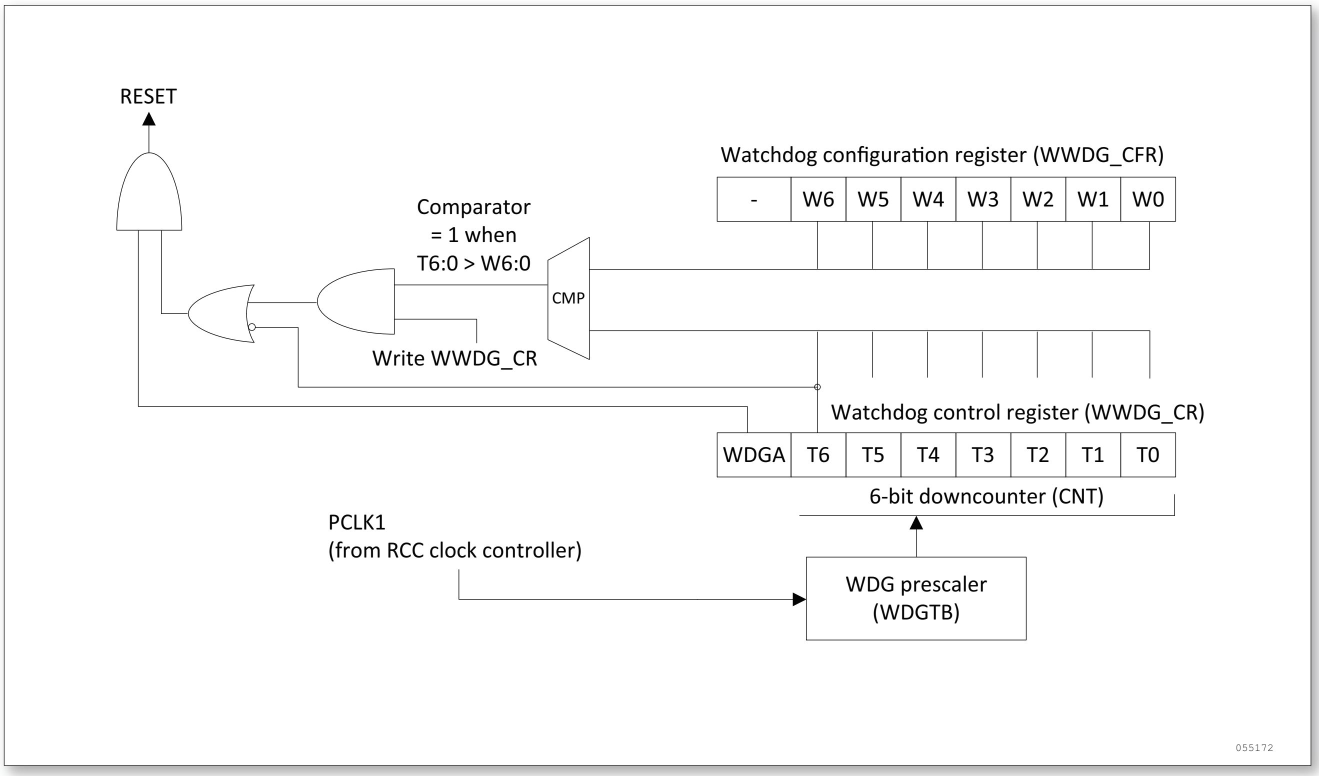


Figure 210. Watchdog Block Diagram

The application program must write in the WWDG_CR register at regular intervals during normal operation to prevent an MCU reset. This operation must occur only when the counter value is lower than the window register value. The value to be stored in the WWDG_CR register must be between 0xFF and 0xC0.

- Enabling the watchdog

The watchdog is always disabled after a reset. It is enabled by setting the WDGA bit in the WWDG_CR register, then it cannot be disabled again except by a reset.

- Controlling the downcounter

This downcounter is free-running, counting down even if the watchdog is disabled. When the watchdog is enabled, the T6 bit must be set to prevent generating an immediate reset.

The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset. The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WWDG_CR register.

The Configuration register (WWDG_CFR) contains the high limit of the window: To prevent a reset, the downcounter must be reloaded when its value is lower than the window register value and greater than 0x3F. The above figure describes the window watchdog process.

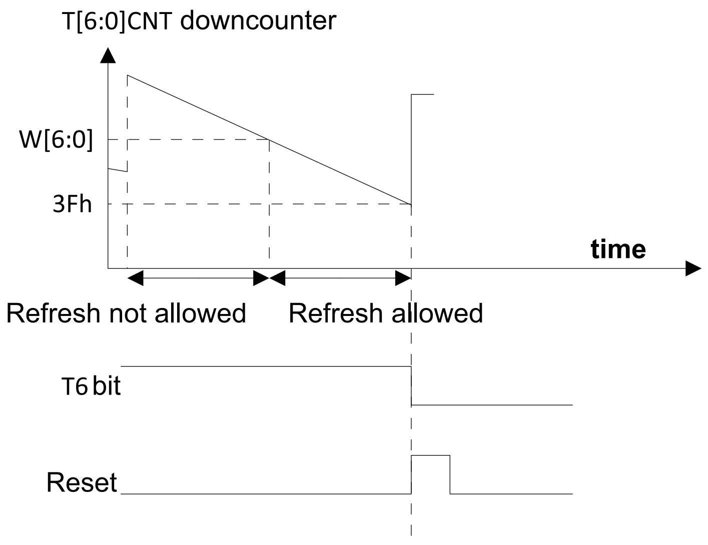
Another way to reload the counter is to use the early wakeup interrupt (EWI). This interrupt is enabled by setting the WEI bit in the WWDG_CFR register. It is generated when the downcounter reaches 0x40, and the corresponding interrupt service routine (ISR) can be used to load counters, so as to prevent WWDG reset. The interrupt can be cleared by writing a ’0’ to the WWDG_SR register.

Note: The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

17.4 How to program the watchdog timeout

The figure below shows the linear relationship (in mS) between the 6-bit count value loaded into the Watchdog Counter (CNT) and the watchdog delay time. This figure can be used as a reference for rapid calculation without taking into account time deviations. If higher precision is required, you can use the calculation formula provided in the figure below.

Warning: When writing to the WWDG_CR register, always write 1 in the T6 bit to avoid generating an immediate reset.



The formula to calculate the WWDG timeout value is given by:

$$T_{WWDG} = T_{PCLK1} \times 4096 \times 2^{WDGTB} \times (T[5:0]+1) \qquad (mS)$$

Where:

T_{WWDG}: WWDG timeout

T_{PCLK1}: APB1 clock period measured in ms

Minimum and maximum timeout values at PCLK1=36 MHz

WDGTB	Min. timeout value	Max. timeout value
0	**113µS**	**7.28mS**
1	**227µS**	**14.56mS**
2	**455µS**	**29.12mS**
3	**910µS**	**58.25mS**

399420

Figure 211. Window Watchdog Timing Diagram

17.5 Debug mode

When the microcontroller enters debug mode (CPU core halted), the WWDG counter either continues to work normally or stops, depending on DBG_WWDG_STOP configuration bit in DBG module. For more details, refer to sections of debug modules.

17.6 WWDG register description

Table 55. Overview of WWDG Registers

Offset	Acronym	Register Name	Reset	Section
0x00	WWDG_CR	Control register	0x0000007F	section 17.6.1
0x04	WWDG_CFGR	Configuration register	0x0000007F	section 17.6.2
0x08	WWDG_SR	Status register	0x00000000	section 17.6.3

17.6.1 Control register(WWDG_CR)

Offset address: 0x00

Reset value: 0x0000 007F

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								WDGA	T						
								rw	rw						

Bit	Field	Type	Reset	Description
31:8	Reserved			Reserved, always read as 0.
7	WDGA	rw	0x00	Activation bit This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset. 0: Watchdog disabled 1: Watchdog enabled
6:0	T	rw	0x7F	7 - bit counter These bits contain the value of the watchdog counter. It is decremented every(4096x2^{WDGTB})PCLK1 cycles.A reset is produced when it rolls over from 0x40 to 0x3F (T6 becomes cleared).

17.6.2 Configuration register(WWDG_CFGR)

Offset address: 0x04

Reset value: 0x0000 007F

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						EWI	WDGTB		W						
						rw	rw		rw						

Bit	Field	Type	Reset	Description
31:10	Reserved			Reserved, always read as 0.
9	EWI	rw	0x00	Early wakeup interrupt When set, an interrupt occurs whenever the counter reaches the value 0x40. This interrupt is only cleared by hardware after a reset.
8:7	WDGTB	rw	0x00	Timer base The time base of the prescaler can be modified as follows: 00: CK Counter Clock (PCLK1 div 4096) div1 01: CK Counter Clock (PCLK1 div 4096) div2 10: CK Counter Clock (PCLK1 div 4096) div4 11: CK Counter Clock (PCLK1 div 4096) div8
6:0	WINDOW	rw	0x7F	7-bit window value These bits contain the window value to be compared to the downcounter.

17.6.3 Status register(WWDG_SR)

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															EWIF
															rc_w0

Bit	Field	Type	Reset	Description
31:1	Reserved			Reserved, always read as 0.
0	EWIF	rc_w0	0x00	Early wakeup interrupt flag This bit is set by hardware when the counter has reached the value 0x40. It must be cleared by software by writing ‘0’ . A write of ‘1’ has no effect. This bit is also set if the interrupt is not enabled.

18 Serial peripheral interface(SPI)

Serial peripheral interface(SPI)

18.1 SPI description

The SPI interface is widely used for board-level communication between different devices, such as the extended serial Flash, ADC, etc. Devices from many IC manufacturers support the SPI interface.

The serial peripheral interface (SPI) allows full-duplex, synchronous, serial communication with external devices. Applications can communicate by querying status or SPI interrupts.

18.2 Main features

- Fully compatible with Motorola SPI specification
- Support DMA requests
- Full-duplex synchronous transfers on three lines
- 16-bit programmable baud rate generator
- Master or slave operation
- 8-byte FIFO receiving/transmitting
- In master mode, SPI clock reaches pclk/2 (pclk: APB clock); in slave mode, SPI clock is up to pclk/4
- Programmable clock polarity and phase
- Programmable data sequence, MSB first or LSB first
- Support one-host and multiple-slave operations
- Support simultaneous transmission and reception of 1 ~ 32-bit data
- In addition to 8-bit data transmission and reception, the remaining 1 ~ 32-bit data transmission and reception only supports LSB mode, not supporting MSB mode.
- Support 8 transmit buffers and receive buffers for corresponding configuration data bits (Data size)
- Interrupt drive operation
  - The transmitting end is null and the end overflows
  - Received data is valid, and data overflows at the receiving end
  - Complete reception in SPI master mode, and the transmitting end is null

18.3 Functional description

18.3.1 General

The block diagram of SPI is shown below

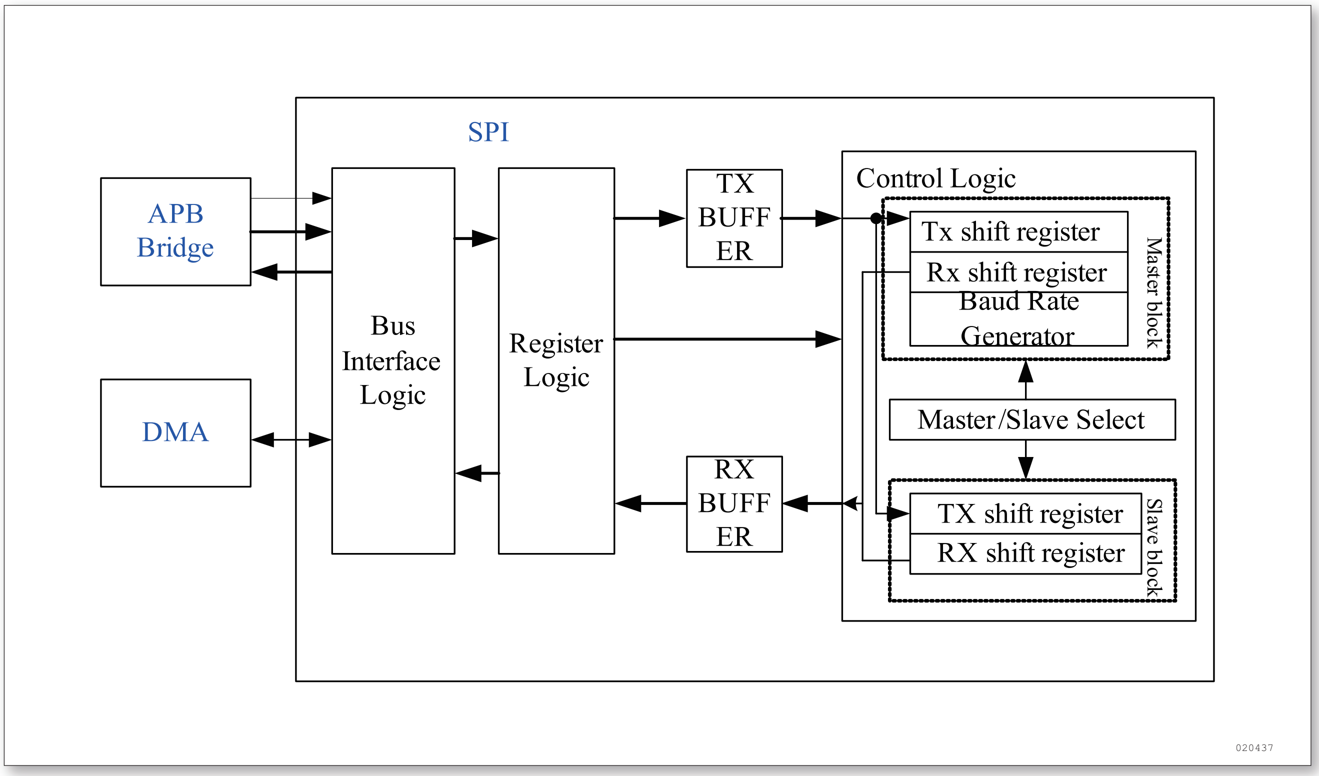


Figure 212. SPI Block Diagram

The SPI enables receiving and transmitting 1 ~ 32-bit data simultaneously. The SPI can be configured as the slave mode or the master mode in a host environment. Four possible timing relationships can be selected by configuring the clock polarity CPOL and phase CPHA. It supports programmable data sequence, i.e. MSB first or LSB first.

The same clock is used for the transmission and reception. Data is clocked on the rising or falling edge of the clock, latching the data on the opposite valid edge of SCLK. Since the SPI is used to exchange data, the data must be read after transferring even if the data is invalid. In SPI mode, the clock and polarity of the master shall be the same as that of the slave to be communicated.

Usually, the SPI is connected to external devices through four pins:

MOSI: Master Out / Slave In data. This pin can be used to transmit data in master mode and receive data in slave mode.

SCK: Serial Clock output for SPI masters and input for SPI slaves.

NSS: Slave select. This is an optional pin to select a master/slave device. This pin acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave NSS inputs can be driven by standard IO ports on the master device. The NSS pin may also be used as an output if enabled and driven low if the SPI is in master mode. At this time, all NSS pins from devices connected to the Master NSS pin see a low level.

- MISO: Master In / Slave Out data. This pin can be used to transmit data in slave mode and receive data in master mode.
- MOSI: Master Out / Slave In data. This pin can be used to transmit data in master mode

and receive data in slave mode.

- SCK: Serial Clock output for SPI masters and input for SPI slaves.
- NSS: Slave select. This is an optional pin to select a master/slave device. This pin acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave NSS inputs can be driven by standard IO ports on the master device. The NSS pin may also be used as an output if enabled and driven low if the SPI is in master mode. At this time, all NSS pins from devices connected to the Master NSS pin see a low level.

An example of interconnections between a single master and a single slave is illustrated below.

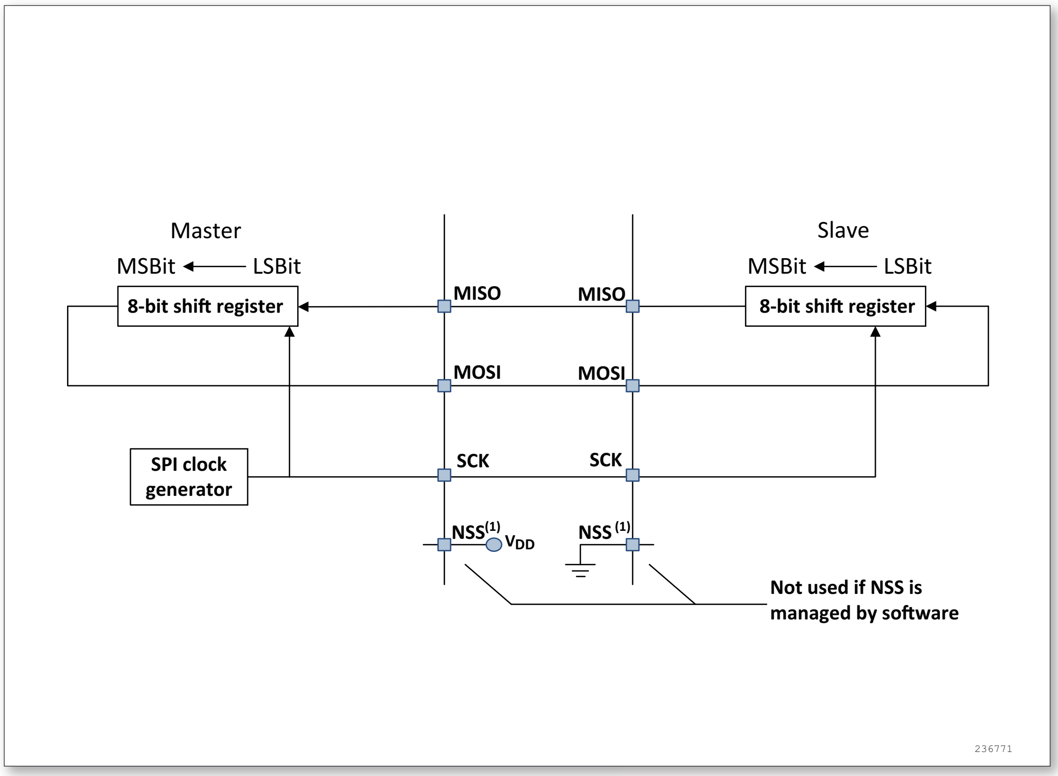


Figure 213. Single Master/Single Slave Application

The MOSI pins are connected together and the MISO pins are connected together. In this way, data is transferred serially between master and slave (most significant bit (MSB) first).

The communication is always initiated by the master. When the master device transmits data to a slave device via the MOSI pin, the slave device responds via the MISO pin. This implies full-duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

Clock phase and clock polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits in the SPI_CCTL register. The CPOL (clock polarity) bit controls the steady state value of the clock when no data is being transferred. This bit affects both master and slave modes. If CPOL is reset, the SCK pin has a low-level idle state, namely, the low level between two transfers. If CPOL is set, the SCK pin has a high-level idle state, namely, the high level between two transfers.

If the CPHA (clock phase) bit is set, the first data bit is latched on the second edge on the SCK (falling edge if the CPOL bit is set; rising edge if the CPOL bit is reset), and the data bit received is sampled. The SPI changes the serial data during the first SCK clock transition (when the clock changes in the opposite direction of the idle state) and captures the data on the next edge.

If the CPHA (clock phase) bit is cleared, the first data bit is latched on the first edge on the SCK (falling edge if the CPOL bit is reset; rising edge if the CPOL bit is set), and the data bit received is sampled. The SPI captures the the serial data during the first SCK clock transition (the clock changes in the opposite direction of the idle state) and the data is changed on the next edge.

The combination of the CPOL (clock polarity) and CPHA (clock phase) bits selects the data capture clock edge. Figure 214 shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin,the MISO pin, the MOSI pin are directly connected between the master and the slave device.

High-speed transmission

Considering the sensitivity to board-level delay in high-speed transmission mode, adjust the time for adjust the time of the transmitting phase and the received samples by setting TXEDGE and RXEDGE control bits in the SPI_CCTL register.

- In the slave mode, if TXEDGE is 1, the data is immediately sent to the data bus for high-speed mode (SPBRG = 4); when the bit is 0, the data is sent to the data bus after a valid clock edge for low-speed mode (SPBRG > 4).
- In master mode, if RXEDGE is 1, the data is sampled in the middle of the transmitted data bit; when the bit is 0, the data is sampled on the tail clock edge of the transmitted data bit (for high-speed mode)

1. The SPIEN bit must be cleared to disable the SPI before changing the CPOL/CPHA bit.
2. The master and slave must be configured as the same timing mode.
3. The idle state of SCK must be the same as the polarity specified by the SPI_CCTL register. When CPOL is 1, the SCK shall be set high in the idle state. When CPOL is 0, SCK shall be set low in the idle state.

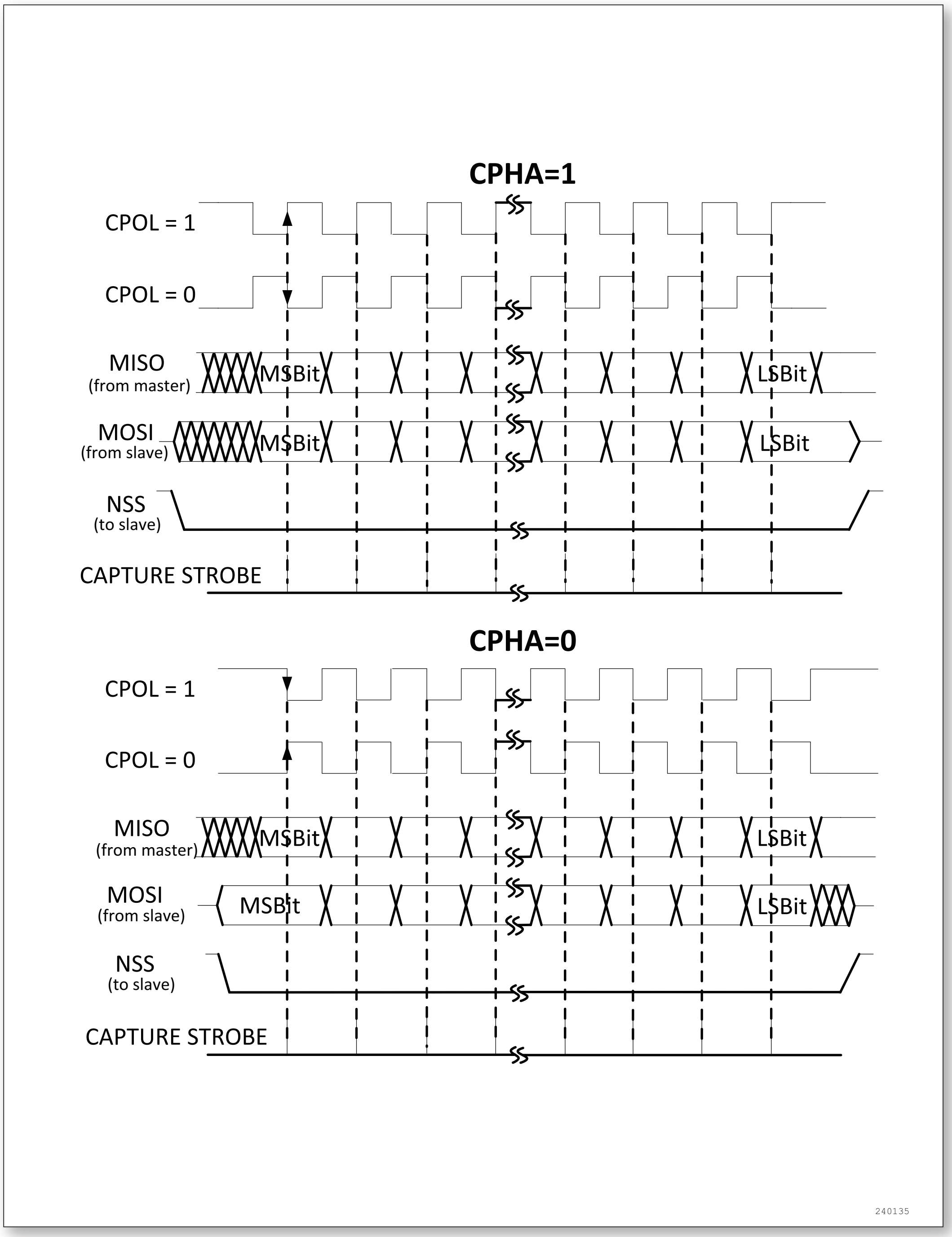


Figure 214. Data Clock Timing Diagram

Data frame format

Data can be shifted out either MSB-first or LSB-first depending on the value of the LSBFE bit in the SPI_CCTL Register. Each data frame is 7 or 8 bits long depending on the SPILEN bit in the SPI_CCTL register. The selected data frame format is applicable to transmission and/or reception.

In addition, the register SPI_EXTCTL can be configured, with the data frame length of 1 ~ 32 bits. Configuration is required to during the application: the DW8_32 bit of the SPI_CCTL register is set to '0', and the LSBFE bit of the SPI_CCTL register is set to '1' and the SPILEN bit is set to '1'. In conjunction with DMA data transmission, the data length of the DMA needs to be configured as 8 bits.

18.3.2 SPI slave mode

In the slave configuration, the serial clock is received on the SCK pin from the master device. The setting in the SPI_SPBRG register does not affect the data transfer rate.

Procedure

1. Set the SPILEN bit to define 7- or 8-bit data frame format.
2. Select the CPOL and CPHA bits to define one of the four relationships between the data transfer and the serial clock. For correct data transfer, the CPOL and CPHA bits must be configured in the same way in the slave device and the master device.
3. The frame format (MSB-first or LSB-first depending on the value of the LSBFE bit in the SPI_CCTL register) must be the same as the master device.
4. Clear the MDOE bit and set the SPIEN bit, making corresponding pins work in SPI mode. In this configuration, the MOSI pin is used as the data input, and MISO as data output.

Transmit sequence

The data byte is parallel-loaded into the transmitting buffer during the write operation.

The transmit sequence begins when the slave device receives the clock signal and the first data bit appears on its MOSI pin, then the first bit is sent. The remaining bits are loaded into the shift-register. The TX_INTF flag in the SPI_INTSTAT register is set on the transfer of data from the transmitting buffer to the shift register, and an interrupt is generated if the TXIEN bit in the SPI_INTEN register is set.

Receive sequence

For the receiver, when data transfer is complete:

- The Data in shift register is transferred to receiving buffer and the RX_INTF flag (SPI_INTSTAT register) is set.
- An Interrupt is generated if the RXIEN bit is set in the SPI_INTEN register.

After the last sampling clock edge, the RXNE bit is set, a copy of the data byte received in the shift register is moved to the receiving buffer. When the SPI_RXREG register is read, the SPI peripheral returns this buffered value.

18.3.3 SPI master mode

In the master configuration, the serial clock is generated on the SCK pin.

Procedure

1. Define the serial clock baud rate through SPI_SPBRG register.
2. Select the CPOL and CPHA bits to define the phase relation between the data transfer and the serial clock.
3. Set the SPILEN bit to define 8- or 7-bit data frame format.
4. Configure the LSBFE bit in the SPI_CCTL register to define the frame format.

5. If data is only received and not sent, the SPI_RNDNR register shall be set, and the bytes to be received shall be defined.
6. The MDOE and SPIEN bits must be set.

In this configuration, the MOSI pin is a data output, the MISO pin is a data input, and NSS is the select signal output of slave device.

Transmit sequence

The transmit sequence begins when a byte is written in the transmitting buffer. The data byte is parallel-loaded into the shift register (from the internal bus) during the first bit transmission, and then shifted out serially to the MOSI pin; MSB first or LSB first depends on the LSBFE bit in the SPI_CCTL register. The TX_INTF flag is set on the transfer of data from the transmitting buffer to the shift register, and an interrupt is generated if the TXIEN bit in the SPI_INTEN register is set.

Receive sequence

For the receiver, when data transfer is complete:

- The Data in shift register is transferred to receiving buffer and the RX_INTF flag (SPI_INTSTAT register) is set.
- An Interrupt is generated if the RXIEN bit is set in the SPI_INTEN register.

After the last sampling clock edge, the RXNE bit is set, a copy of the data byte received in the shift register is moved to the receiving buffer. When the SPI_RXREG register is read, the SPI peripheral returns this buffered value.

If only data is received and not transmitted, the RXMATCH_INTF bit is set to ’1’ after receiving the bytes defined by RXDNR, indicating that all data has been received, and the clock signal is no longer transmitted in Master mode.

18.3.4 Status flags

For convenient software operation, the application can monitor the state of the SPI bus with four current status flags and seven interrupt status flags. The current status flag is read-only and is automatically set and cleared by hardware; the interrupt status flag is set when an event occurs and generates a CPU interrupt when the interrupt is enabled, and it can be cleared by software.

The SPI is configured with 8-byte transmit buffer and receive buffer. The CPU enables reading and writing 1 or 4 bytes at a time according to the DW8_32-bit setting of SPI_GCTL. Based on the configuration of DW8_32, the transmit and receive buffers respectively have one-byte flag or a status flag of valid data.

Table 56. SPI Status

Classification	Status flag	Buffer and signal status
Interrupt status	TX_INTF	According to the DW8_32 setting, there is at least one space for valid data, enabling writing the transmitting data register for one time.

Interrupt status	RX_INTF	According to the DW8_32 setting, there is at least one space for valid data, enabling reading the transmitting data register for one time.
Interrupt status	UNDERRUN_INTF	Transmitting buffer is null and repeated transmission occurs
Interrupt status	RXOERR_INTF	Receiving buffer is non-null and overwritten
Interrupt status	RXMATCH_INTF	Non-null, the last data is transferred to the receiving buffer
Interrupt status	RXFULL_INTF	Receiving buffer is full and unable to receive new data
Interrupt status	TXEPT_INTF	Transmitting buffer is null and unable to send data
Current status	RXAVL_4BYTE	Receiving buffer is loaded with valid data more than 4 bytes
Current status	TXFULL	Transmitting buffer is full
Current status	TXEPT	Transmitting buffer is null
Current status	RXAVL	The receiving buffer is non-null and enables receiving at least one byte

When the TXTLF of the SPI_GCTL register is 00, TX_INTF is set when the transmitting buffer is configured with one or more free data spaces. When TXTLF is 01, TX_INTF is set when the transmitting buffer is configured with more than half of the free space.

When the RXTLF of the SPI_GCTL register is 00, RX_INTF is set when the receiving buffer is loaded with one or more valid data. When RXTLF is 01, RX_INTF is set when the receiving buffer is loaded with more than half of the valid data.

18.3.5 Baud rate setting

The baud rate is the frequency of the generated SCLK, which is typically the division of PCLK. The BRG is a 16-bit baud rate generator, and the SPBREG register is used to control the count period of the 16-bit counter.

The desired baud rate and f_{pclk} (frequency of the APB module) are given, and the value calculated from the formula shown in the table below is set to the SPBRG register. The X in the table below is equal to the value of the SPBRG register (2 ~ 65535).

Table 57. Baud Rate Formula

Mode	**Formula**
SPI mode	Baud rate = f_{pclk}/X

18.3.6 SPI communication using DMA

To operate at its maximum speed, the SPI needs to be fed with the data for transmission and the data received on the receive buffer should be read to avoid overrun. To facilitate the transfers, the SPI features a DMA capability implementing a simple request/acknowledge protocol.

A DMA access is requested when DMAEN bit in the SPI_GCTL register is enabled. DMA requests of transmitting buffer and receive buffer are enabled by DMAEN.

- In transmission, a DMA request is issued if TXTLF in SPI_GCTL register is set to 00 and the transmitting buffer is configured with one or more free data spaces; when TXTLF is set to 01 and the transmitting buffer is configured with more than half free space, a DMA request is generated, enabling only one DMA transfer. In the process, the data size and that of transmitting buffer depend on DW8_32.
- In reception, a DMA transfer request is issued if RXTLF in SPI_GCTL register is set to 00 and the receive buffer is loaded with one or more valid data; when RXTLF is set to 01 and the receive buffer is loaded with more than half valid data, a DMA request is generated, enabling only one DMA transfer. In the process, the data size and that of receive buffer depend on DW8_32.

18.4 Register file and memory mapping description

Table 58. Overview of SPI Register

Offset	Acronym	Register Name	Reset	Section
0x00	SPI_TXREG	Transmit data register	0x00000000	section 18.4.1
0x04	SPI_RXREG	Receive data register	0x00000000	section 18.4.2
0x08	SPI_CSTAT	Current status register	0x00000001	section 18.4.3
0x0C	SPI_INTSTAT	Interrupt status register	0x00000000	section 18.4.4
0x10	SPI_INTEN	Interrupt enable register	0x00000000	section 18.4.5
0x14	SPI_INTCLR	Interrupt clear register	0x00000000	section 18.4.6
0x18	SPI_GCTL	Global control register	0x00000004	section 18.4.7
0x1C	SPI_CCTL	General-purpose control register	0x00000008	section 18.4.8
0x20	SPI_SPBRG	Baud rate generator register	0x00000002	section 18.4.9
0x24	SPI_RXDNR	Receive data count register	0x00000001	section 18.4.10
0x28	SPI_NSSR	Slave chip select register	0x000000FF	section 18.4.11
0x2C	SPI_EXTCTL	Data control register	0x00000008	section 18.4.12

18.4.1 Transmit data register(SPI_TXREG)

Offset address: 0x00

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	TXREG	rw	0x0000 0000	Transmit data register Valid data bits are controlled by DW8_32. 0: Lower 8 bits active 1:TXREG 31:0 active

18.4.2 Receive data register(SPI_RXREG)

Offset address: 0x04

Reset value: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	RXREG	r	0x0000 0000	Receive data register Valid data bits are controlled by DW8_32. 0: Lower 8 bits active 1: RXREG 31:0 active This register is readable and non-writable.

18.4.3 Current status register(SPI_CSTAT)

Offset address: 0x08

Reset value: 0x0000 0001

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				RXFADDR				TXFADDR				RXAVL_4BYTE	TXFULL	RXAVL	TXEPT
				r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:12	Reserved			Always read as 0.
11:8	RXFADDR	r	0x00	Receive FIFO address
7:4	TXFADDR	r	0x00	Transmit FIFO address
3	RXAVL_4BYTE	r	0x00	Receive available 4 byte data message 1: Receive buffer is loaded with data more than 4 bytes 0: Receive buffer is loaded with data less than 4 bytes
2	TXFULL	r	0x00	Transmitter FIFO full status bit 1: Transmitting buffer full 0: Transmitting buffer not full
1	RXAVL	r	0x00	Receive available byte data message This bit is set when the receiver buffer receives data of one full byte. 1: Receiver buffer has received a valid byte of data 0: Receiver buffer is null This bit is read-only and is automatically set and cleared by hardware.
0	TXEPT	r	0x01	Transmitter empty bit The transmitter buffer and the transmit shift register are null. 0: The transmitter buffer is non-null This bit is read-only and is automatically set and cleared by hardware.

18.4.4 Interrupt status register(SPI_INTSTAT)

Offset address: 0x0C

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									TXEPT_INTF	RXFULL_INTF	RX MATCH_INFT	RXO ERR_INFT	UNDE RRUN_INTF	RX_INTF	TX_INTF
									r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:7	Reserved			Always read as 0.
6	TXEPT_INTF	r	0x00	Transmitter empty interrupt flag bit This bit is automatically reset by hardware, and can be cleared by writing TXEPT_ICLR bit in INTCLR register. 1: The transmitter buffer and TX shift register are null 0: The transmitter buffer is non-null Note: This bit is the interrupt status signal and TXEPT is the status signal.
5	RXFULL_INTF	r	0x00	RX FIFO full interrupt flag bit This bit is automatically reset by hardware, and can be cleared by writing RXFULL_ICLR bit in INTCLR register 1: RX buffer full 0: RX buffer not full
4	RXMATCH_INTF	r	0x00	Receive specified bytes interrupt flag bit(Receive data match the RXDNR number, the receive process will be completed and generate the interrupt) This bit is automatically reset by hardware, and can be cleared by writing RXMATCH_ICLR bit in INTCLR register. 1: Bytes specified by the RXDNR register are received 0: Bytes specified by the RXDNR register are not received
3	RXOERR_INTF	r	0x00	Receive overrun error interrupt flag bit This bit is automatically reset by hardware, and can be cleared by writing RXOERR_ICLR bit in INTCLR register. 1: Overrun error 0: No overrun error
2	UNDERRUN_INTF	r	0x00	SPI underrun interrupt flag bit This bit is automatically reset by hardware, and can be cleared by writing UNDERRUN_ICLR bit in INTCLR register. 1: Underrun error 0: No underrun error

Bit	Field	Type	Reset	Description
1	RX_INTF	r	0x00	Receive data available interrupt flag bit This bit is automatically reset by hardware, and can be cleared by writing RX_ICLR bit in INTCLR register. This bit is set when the receiver buffer receives data of one full byte. 1: Receiver buffer has received valid data 0: Receiver buffer is null
0	TX_INTF	r	0x00	Transmit FIFO available interrupt flag bit (sending data of one byte) This bit is automatically reset by hardware, and can be cleared by writing TX_ICLR bit in INTCLR register. 1: Transmitter buffer enabled 0: Transmitter buffer disabled

18.4.5 Interrupt enable register(SPI_INTEN)

Offset address: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TXEPT_ IEN	RXFULL _IEN	RX MATCH _IEN	RXO ERR_ IEN	UNDE RRUN _IEN	RX_ IEN	TX_ IEN
									rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:7	Reserved			Always read as 0.
6	TXEPT_IEN	rw	0x00	Transmit empty interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled
5	RXFULL_IEN	rw	0x00	Receive FIFO full interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled
4	RXMATCH_ IEN	rw	0x00	Receive data complete interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled
3	RXOERR_ IEN	rw	0x00	Overrun error interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled

Bit	Field	Type	Reset	Description
2	UNDERRUN_IEN	rw	0x00	Transmitter underrun interrupt enable bit(SPI slave mode only) 1: Interrupt enabled 0: Interrupt disabled
1	RX_IEN	rw	0x00	Receive FIFO interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled
0	TX_IEN	rw	0x00	Transmit FIFO empty interrupt enable bit 1: Interrupt enabled 0: Interrupt disabled

18.4.6 Interrupt clear register

Offset address: 0x14

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TXEPT_ICLR	RXFULL_ICLR	RXMATCH_ICLR	RXOERR_ICLR	UNDERRUN_ICLR	RX_ICLR	TX_ICLR
									w	w	w	w	w	w	r

Bit	Field	Type	Reset	Description
31:7	Reserved			Always read as 0.
6	TXEPT_ICLR	w	0x00	Transmitter empty interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared
5	RXFULL_ICLR	w	0x00	Receiver buffer full interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared
4	RXMATCH_ICLR	w	0x00	Receive completed interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared
3	RXOERR_ICLR	w	0x00	Overrun error interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared
2	UNDERRUN_ICLR	w	0x00	Transmitter underrun interrupt clear bit (SPI slave mode only) 1: Interrupt cleared 0: Interrupt not cleared

Bit	Field	Type	Reset	Description
1	RX_ICLR	w	0x00	Receive interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared
0	TX_ICLR	r	0x00	Transmitter FIFO empty interrupt clear bit 1: Interrupt cleared 0: Interrupt not cleared

18.4.7 Global control register(SPI_GCTL)

Offset address: 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			NSS TOG	DW8_ 32	NSS	DMAEN	TXTLF		RXTLF		RXEN	TXEN	MODE	INTEN	SPIEN
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:13	Reserved			Always read as 0.
12	NSSTOG	rw	0x00	Slave select toggle 1：NSS toggle after transmit 0：NSS not toggle after transmit note: only work in master mode
11	DW8_32	rw	0x00	Valid byte or double-word data select signal 0: Lower 8 bits active 1: 32-bit data active Note: When using CPU or DMA, access data in the specified format.
10	NSS	rw	0x00	NSS select signal that from software or hardware 0: Controlled by the NSSR register value 1: Automatically controlled by hardware during data transmission
9	DMAEN	rw	0x00	DMA access mode enable 0: DMA mode disabled 1: DMA mode enabled

Bit	Field	Type	Reset	Description
8:7	TXTLF	rw	0x00	TX FIFO trigger level bit 00: The DMA request or transmit interrupt request is generated if the transmitting buffer is configured with 1 or more free data spaces. 01: The DMA request or transmit interrupt request is generated if the transmitting buffer is configured with more than half free spaces. 1x: Reserved Note: If DW8_32 is 0, one data space represents 1 byte; when it is 1, a data space represents 4 bytes.
6:5	RXTLF	rw	0x00	RX FIFO trigger level bit 00: The DMA request or receive interrupt request is generated if the receive buffer is loaded with 1 or more valid data. 01: The DMA request or receive interrupt request is generated if the receive buffer is loaded with more than half valid data. 1x: Reserved Note: If DW8_32 is 0, one valid data represents 1 byte; when it is 1, one valid data represents 4 bytes.
4	RXEN	rw	0x00	Receive enable bit 1: Reception enabled 0: Reception disabled and RX buffer cleared Note: txen must be set to 0 when the SPI only runs in master receive mode.
3	TXEN	rw	0x00	Transmit enable bit 1: Transmission enabled 0: Transmission disabled and TX buffer cleared Note: Transmission and receiving are completed simultaneously in master mode.
2	MODE	rw	0x01	Master mode bit 1: Master mode (serial clock generated by internal BRG) 0: Slave mode (serial clock from external master)
1	INTEN	rw	0x00	SPI interrupt enable bit 1: SPI interrupt enabled 0: SPI interrupt disabled
0	SPIEN	rw	0x00	SPI select bit 0: SPI disabled (reset state) 1: SPI enabled

18.4.8 General-purpose control register(SPI_CCTL)

Offset address: 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
Reserved								HISPD	CPH ASEL	TX EDGE	RX EDGE	SPI LEN	LSB FE	CPOL	CPHA
								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			Always read as 0.
7	HISPD	rw	0x00	High speed slave mode 1：SPI is working in high speed 0：SPI is working in high speed only working in slave mode
6	CPHASEL	rw	0x00	CPHA polarity select 1：CPHA is 0, The first data bit is sampled from the second clock edge 0：CPHA is 0, The first data bit is sampled from the first clock edge
5	TXEDGE	rw	0x00	Transmit data edge select (slave mode) 1: Data is immediately sent to the data bus when the high-speed mode is available (SPBRG = 4). 0: The data is sent to the data bus after a valid clock edge when the low-speed mode is available (SPBRG > 4).
4	RXEDGE	rw	0x00	Receive data edge select (master mode) 1: Sample data at the tail clock edge of the transmit data bit (for high speed mode) 0: Intermediate sample data in the transmit data bit
3	SPILEN	rw	0x01	SPI character length bit This bit is active after DW8_32 is reset (DW8_32=0). 1: 8-bit data (default) 0: 7-bit data
2	LSBFE	rw	0x00	LSBFE: LSI first enable bit 1: Data transmission or reception lowest bit first 0: Data transmission or reception highest bit first
1	CPOL	rw	0x00	Clock polarity select bit 1: The clock is high in the idle state (between two transmissions) 0: The clock is low in the idle state (between two transmissions)

Bit	Field	Type	Reset	Description
0	CPHA	rw	0x00	Clock phase select bit 1: The first data bit is sampled from the first clock edge 0: The first data bit is sampled from the second clock edge

18.4.9 Baud rate generator(SPI_SPBRG)

Offset address: 0x20

Reset value: 0x0000 0002

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
SPBRG															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			Always read as 0.
15:0	SPBRG	rw	0x0002	SPI baud rate control register for baud rate Baud rate formula: Baud rate = fpclk/SPBRG ($f_{pclk}$is the APB clock frequency) Note: Do not write 0 and 1 to this register.

18.4.10 Receive data count register (SPI_RXDNR)

Offset address: 0x24

Reset value: 0x0000 0001

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
RXDNR															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
15:0	RXDNR	rw	0x0001	The register is used to hold a count of to be received bytes in next receive process The value (1 by default) of this register is valid when the SPI is in master receive mode. This register value is modified by writing MCU. Note: Do not write 0 to this register.

18.4.11 Slave chip select register(SPI_NSSR)

Offset address: 0x28

Reset value: 0x0000 00FF

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

Bit	Field	Type	Reset	Description
15:1	Reserved			Always read as 0.
0	NSS	rw	0xFF	Chip select output signal in Master mode This bit is active-low, and is inactive in the slave mode. 0: Slave device selected 1: Slave device not selected

18.4.12 Data control register(SPI_EXTCTL)

Offset address: 0x2C

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
Reserved											EXTLEN				
											rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:5	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
4：0	EXTLEN	rw	0x08	This bit is used to 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 is valid only when the DW8_32 bit of the SPI_GCTL register is set to ’0’, the LSBFE bit of the SPI_CCTL register is set to ’1’, and SPILEN is also set to ’1’.

19 I2C interface (I2C)

I2C interface (I2C)

19.1 I2C introduction

I2C (inter-integrated circuit) bus Interface serves as an interface between the microcontroller and the serial I2C bus. It provides multimaster capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing.

The I2C bus is a two-wire serial interface, in which two-wire bit serial data (SDA) and serial clock (SCL) lines are used to transmit information between the devices connected to bus. Each device is configured with a unique address identification and can be used as a transmitter or receiver. Moreover, the device can also be considered as a master or slave during data transmission. The master is the device that initializes the data transfer of the bus and generates a clock signal enabling the transmission. At that time, any addressed device is considered as a slave.

I2C can operate in standard mode (data transmission rate: 0 ～100 Kbps) and fast mode (maximum data transmission rate: 400 Kbps).

19.2 I2C main features

- Parallel-bus2C protocol converter
- Half-duplex synchronous operation
- Act as Master or Slave
- Support 7-bit and 10-bit addresses
- Support standard mode (100 Kbps) and fast mode (400 Kbps)
- Generate Start, Stop, Resend Start, Acknowledge Signals for detection
- Only support one master device in master mode
- 2 bytes of transmitting and receive buffers respectively
- Provide burr-free circuit to SCLI and SDAI
- Support DMA operation
- Support interrupt and query operations

19.3 I2C protocol

19.3.1 Start and Stop conditions

When the bus is in the idle state, SCL and SDA are simultaneously tied high with the external pull-up resistor. Before the master enables the data transfer, a Start condition must be generated. When the SCL line is high, the SDA line switches from high to low, to indicate the Start condition. A Stop condition is generated when the master disables the transfer. The SCL line is high and the SDA line switches from low to high, indicating a Stop condition.

The figure below shows the timing diagram for the Start and Stop conditions. During data transfer, SDA must remain stable when SCL is 1.

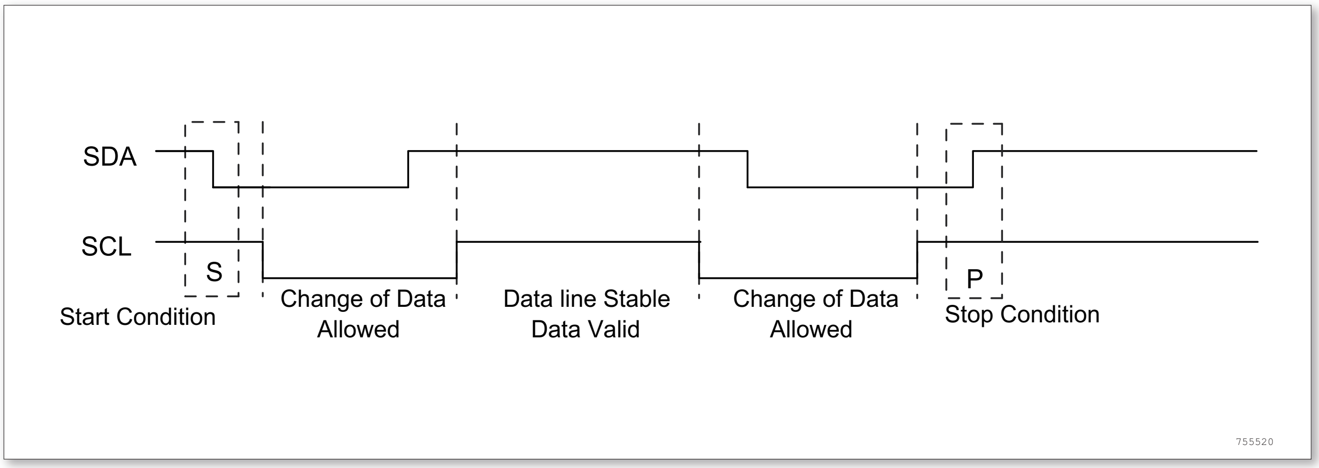


Figure 215. Start and Stop Conditions

19.3.2 Slave address protocol

I2C has two address formats: 7-bit address format and 10-bit address format.

7-bit address format

The following figure shows the first 7 bits (bit 7:1; slave address) of a byte transmitted after the Start condition (S), and the lowest bit (bit 0) is the data direction bit. When bit 0 is 0, it indicates that the master writes data to the slave; 1 represents that the master reads data from the slave.

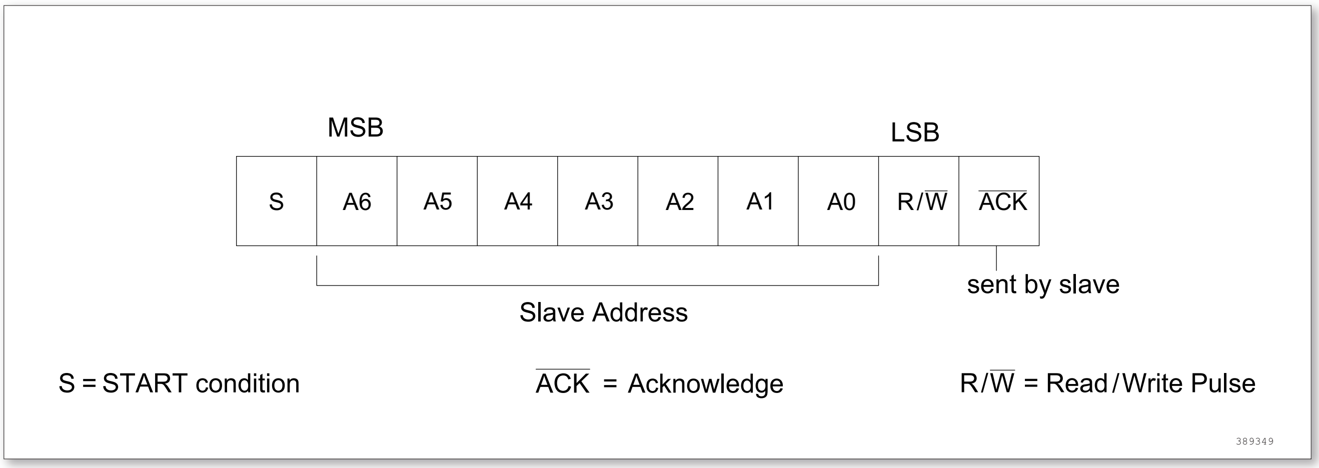


Figure 216. 7-bit Address Format

10-bit address format

In the 10-bit address format, 2 bytes are transmitted to transfer the 10-bit address. The description of the first byte of the transmission bit is as follows: The first 5 bits (bit 7:3) are used to signal the next 10-bit transmission of the slave. The last two bits (bit 2:1) of the first byte are bit 9:8 of the slave address, and the lowest bit (bit 0) is the data direction bit

(R/W). The second byte of the data transferred is the lower eight bits of the 10-bit address.

The details are as follows:

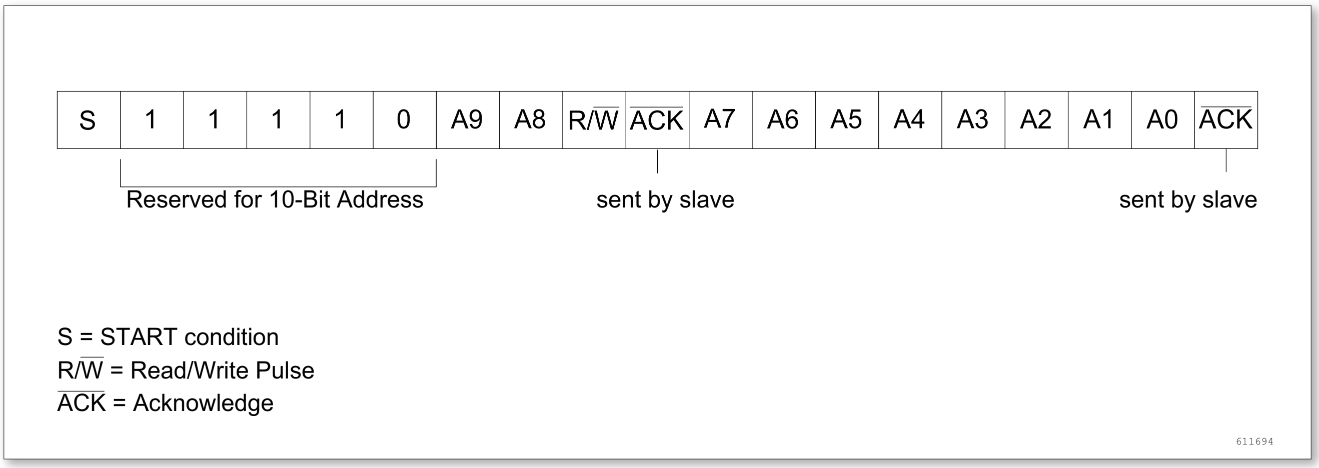


Figure 217. 10-bit Address Format

The following table defines the special purpose and reserved addresses of the first byte of I2C:

Table 59. First Byte of I2C

Slave address	**R/W bit**	**Description**
0000 000	0	General call address: the data is sent to the receive buffer via I2C, so as to generate a general call interrupt
0000 000	1	Start byte
0000 001	X	CBUS address: I2C interface ignores this access
0000 010	X	Reserved
0000 011	X	Reserved
0000 1xx	X	Reserved
1111 1xx	X	Reserved
1111 0xx	X	10-bit slave addressing

19.3.3 Transmission and reception protocols

The master, master transmitter or receiver, initializes data transfer and sends or receives data from the bus. The slave, as a slave transmitter or slave receiver, responds to the master's request, sending or receiving data.

Master transmitter and slave receiver

All data is transmitted in byte format, with the unlimited bytes per transfer. When the master sends the address and R/W bit or the master sends a byte of data to the slave, the slave shall generate a response signal (ACK). When the slave receiver fails to generate an ACK response, the master will generate a Stop condition, to abort the transmission. When the slave fails to respond, SDA shall be set high, making the master generate a Stop condition.

When the master transmitter transmits the data, the slave receiver responds to the master transmitter by generating an ACK after each byte received, as shown in the following figure.

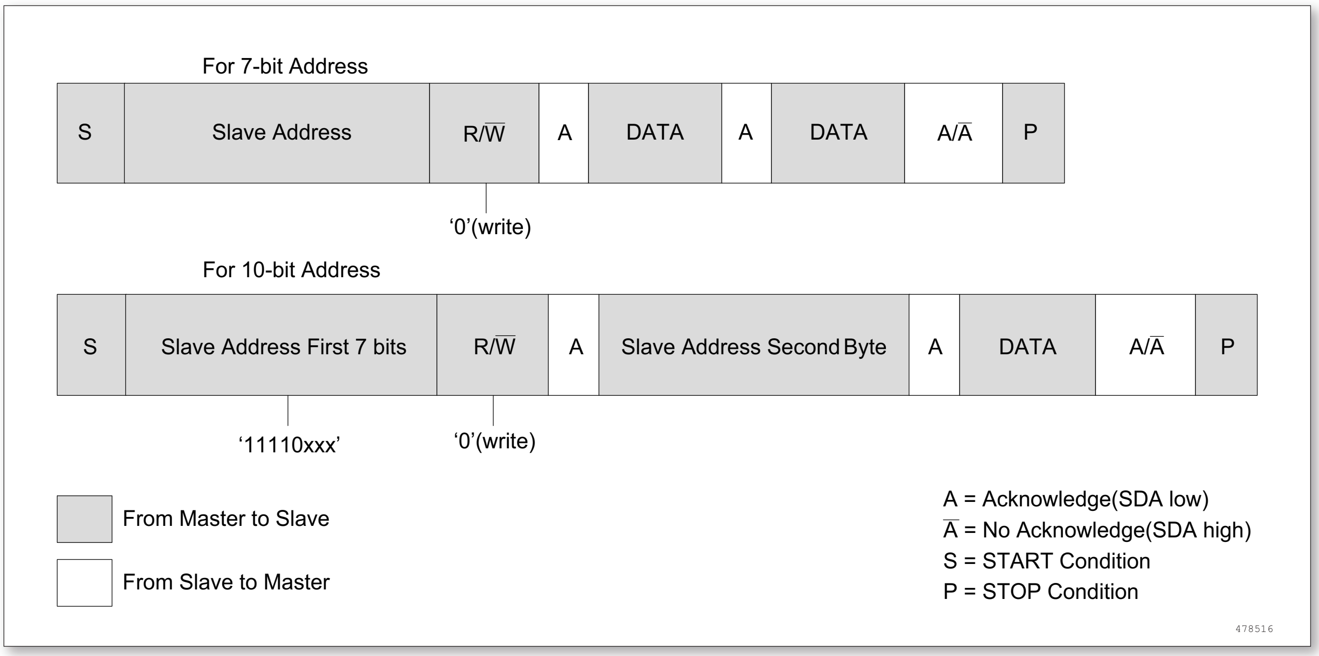


Figure 218. Master Transmitting Protocol

Master receiver and slave transmitter

When the master receives the data, as shown in the figure below, the master shall respond to the slave transmitter after receiving data of one byte, except for the last byte. In this way, the master receiver sends signal indicating whether it is the last byte to the slave transmitter. The slave transmitter shall release the SDA when detecting a NACK so that the master generates a Stop condition.

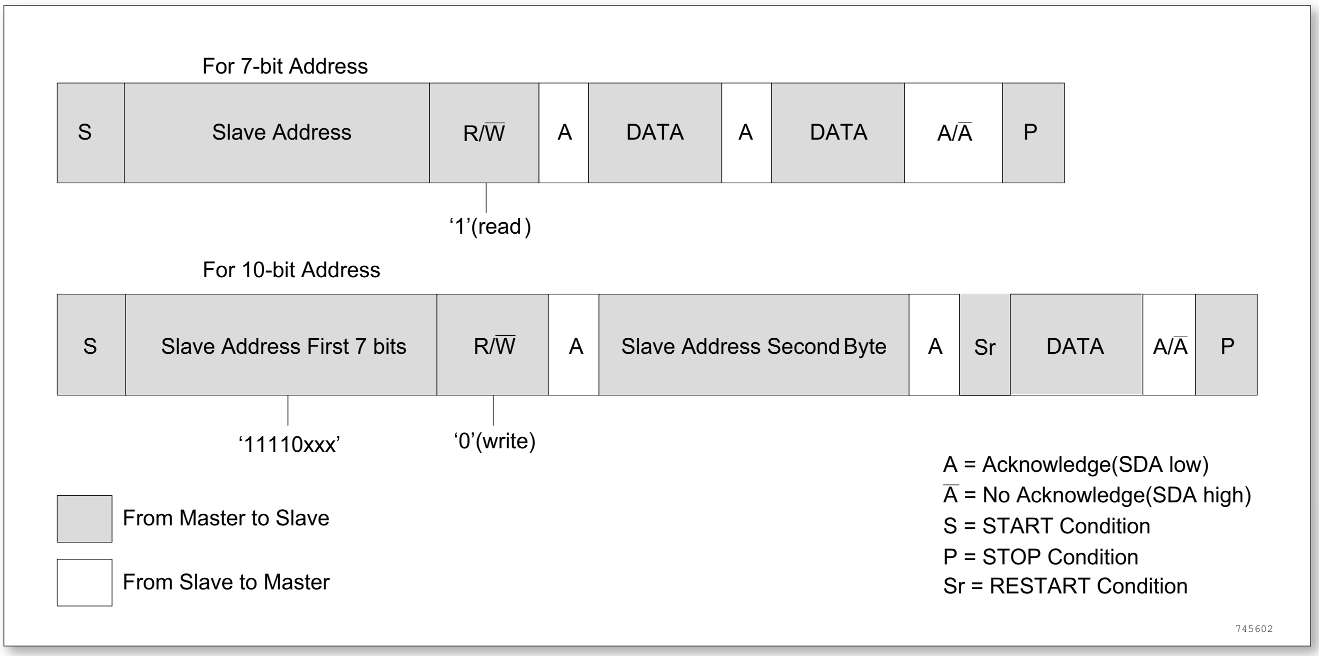


Figure 219. Master Receiving Protocol

When the master needs not to generate Stop conditions, to release the bus, a repeated Start condition can be generated, which is the same as the Start condition except that it is generated after the ACK. In the master mode, the I2C interface communicates with the same slave using different directions of transmission.

Start byte transmission protocol

The start byte transmission protocol is used by systems without dedicated I2C hardware module. When the I2C module is used as the master, the start byte output can be generated for the required slave at the beginning of each transfer.

The protocol consists of seven 0s and one 1, as shown in the following figure. The processor enables inquiring the bus with a low speed sampling flag 0 during the address phase. Once 0 is detected, the processor switches from the low-speed sampling mode to the normal-speed mode of the master.

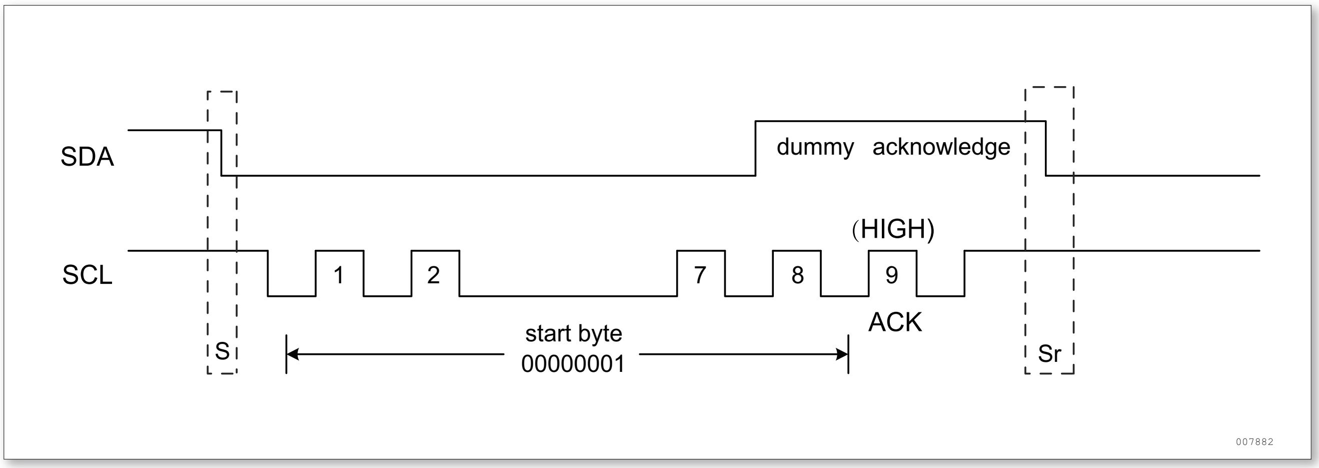


Figure 220. Start Byte Transmission

The start byte procedure is as follows:

1. The master generates a starting condition
2. The master sends the start byte (0000 0001)
3. The master sends an ACK clock pulse (ACK)
4. No slave responds to the ACK signal
5. The master generates a repeated Start condition (RESTART)

The hardware I2C receiver needs not to respond to the start byte since it is a reserved address and the address is reset after RESTART.

19.3.4 Transmitting buffer management and generation of Start, Stop, and repeated Start conditions

In the master mode, the I2C module generates a Stop condition on the bus whenever the transmitter is null. If the repeated Start condition generation is enabled (RESTART = 1), a repeated Start condition is generated when the transfer direction changes from read to write or from write to read. If the repeated Start condition is disabled, a Start condition will be generated after the Stop condition.

The figure below shows the bits of the DR register.

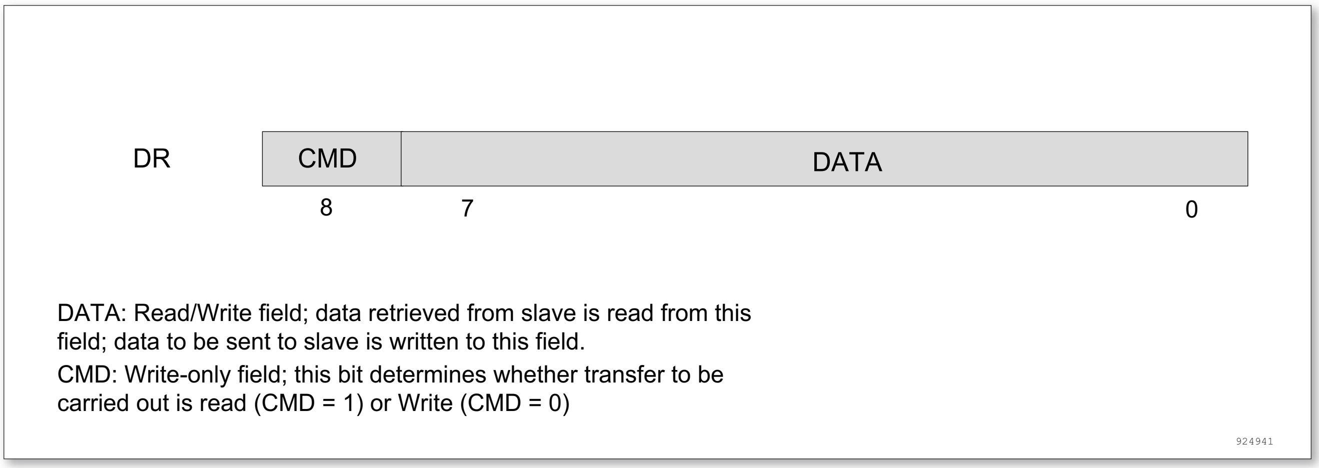

DR
CMD
DATA
8
7
0
DATA: Read/Write field; data retrieved from slave is read from this field; data to be sent to slave is written to this field.
CMD: Write-only field; this bit determines whether transfer to be carried out is read (CMD = 1) or Write (CMD = 0)
924941

Figure 221. DR Register

The timing diagram below describes the behavior of the I2C module in the master transmitting mode when the Tx FIFO becomes null.

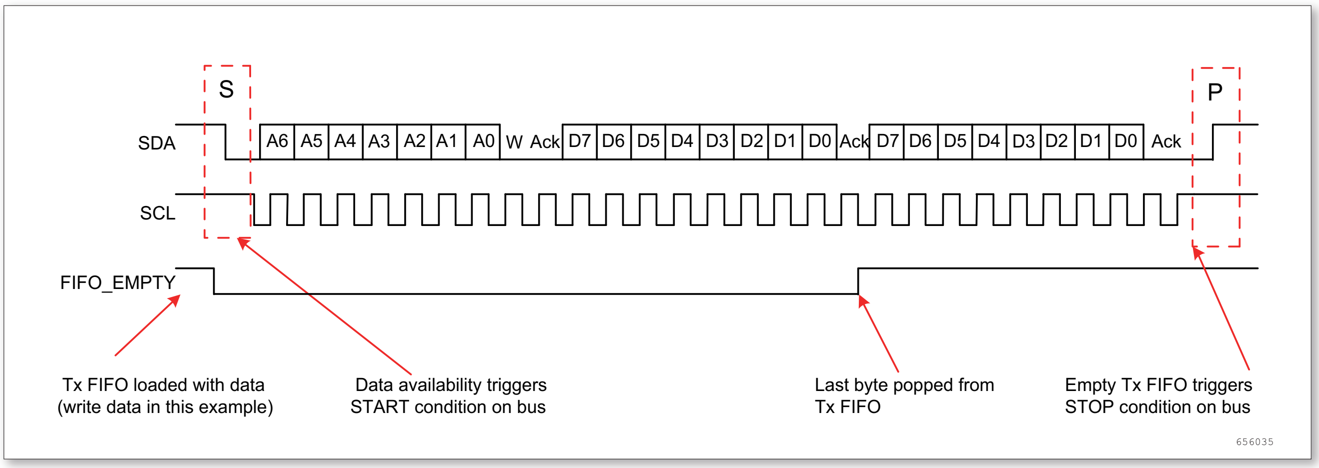

S
P
SDA
A6 A5 A4 A3 A2 A1 A0 W Ack D7 D6 D5 D4 D3 D2 D1 D0 Ack D7 D6 D5 D4 D3 D2 D1 D0 Ack
SCL
FIFO_EMPTY
Tx FIFO loaded with data (write data in this example)
Data availability triggers START condition on bus
Last byte popped from Tx FIFO
Empty Tx FIFO triggers STOP condition on bus
656035

Figure 222. Master Transmitting - Null Tx FIFO

The timing diagram below describes the behavior of the I2C module in the master receiving mode when the Tx FIFO becomes null.

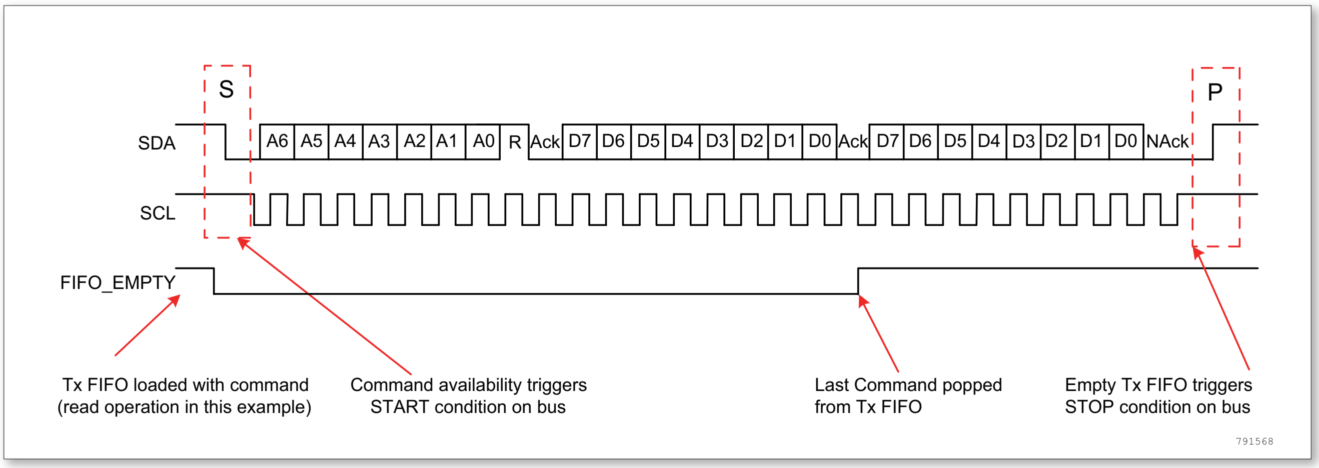

S
P
SDA
A6 A5 A4 A3 A2 A1 A0 R Ack D7 D6 D5 D4 D3 D2 D1 D0 Ack D7 D6 D5 D4 D3 D2 D1 D0 NAck
SCL
FIFO_EMPTY
Tx FIFO loaded with command (read operation in this example)
Command availability triggers START condition on bus
Last Command popped from Tx FIFO
Empty Tx FIFO triggers STOP condition on bus
791568

Figure 223. Master Receiving - Null Tx FIFO

19.3.5 Multiple-master arbitration

The I2C bus is a multi-master bus. Arbitration is a process that multiple masters simultaneously attempt to control the bus, but only one of them is allowed to control the bus and the message is not damaged. Once one of the masters has controlled the bus, the other master can not control the bus until the master sends a Stop condition and sets the bus to the idle state.

Arbitration occurs on the SDA when the SCL line is high. If two or more masters attempt to send information to the bus, and if the other master generates a ”0”, the master that first generated a ”1” will lose the arbitration. Those that lost the arbitration continue to generate clock pulses until the end of the byte transfer. If each master attempts to address the same device, arbitration will continue during the data phase.

After detecting the arbitration lost, the I2C interface stops generating the SCL signal.

The following figure shows the bus timing for arbitration of two masters

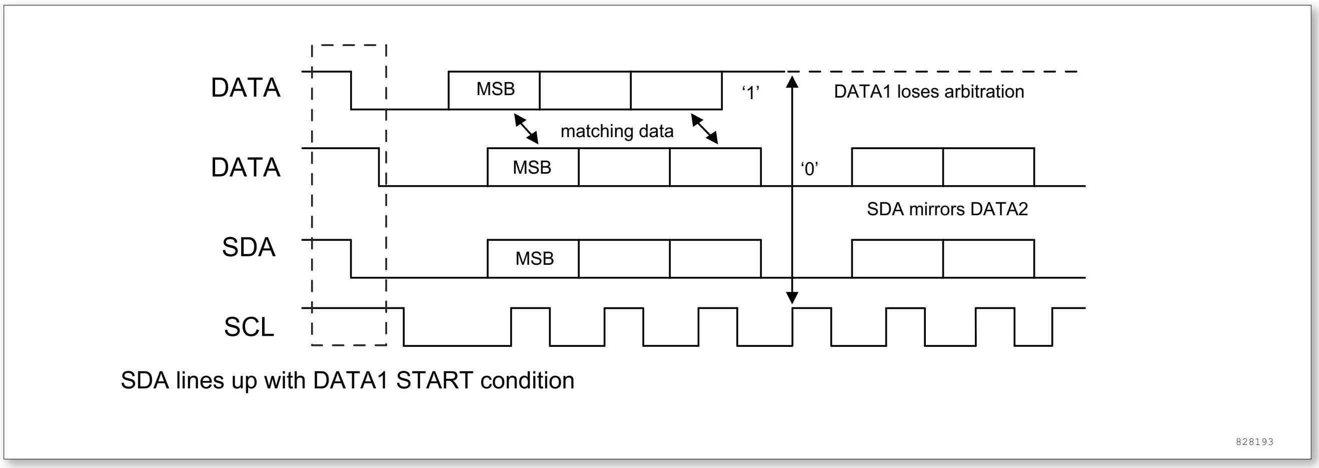


Figure 224. Multiple-master Arbitration

19.3.6 Clock synchronization

When two or more masters attempt to transmit information on the bus simultaneously, they must arbitrate and synchronize the SCL clock. All masters generate their own clocks to transmit messages. The data is only active when the clock is high. Clock synchronization is achieved with ’AND’ connection of the SCL signal. When the master turns the SCL clock to 0, the master will calculate the SCL low time and set the SCL clock to 1 at the beginning of next clock cycle. However, the master will enter a wait state until the SCL clock changes to 1 if another master keeps SCL at 0.

All masters will calculate their high time, and those with the shortest high time will change SCL to 0. Then, the master will calculate the low time, and those with the longest low time will force other masters to enter the wait state, generating a synchronized SCL clock, as shown in the following figure.

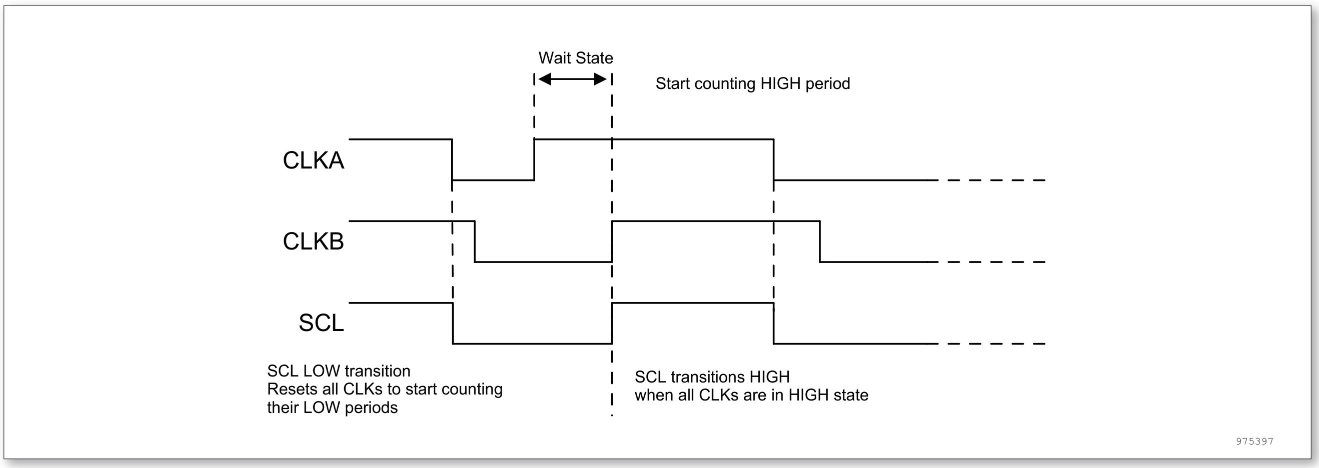

Wait State
Start counting HIGH period
CLKA
CLKB
SCL
SCL LOW transition
Resets all CLKs to start counting
their LOW periods
SCL transitions HIGH
when all CLKs are in HIGH state
975397

Figure 225. Clock Synchronization of Multiple Masters

19.4 I2C operating mode

The I2C interface operates in one of four modes:

- Slave transmitter mode
- Slave receiver mode
- Master transmitter mode
- Master receiver mode

Note: The I2C interface module can only operates in either master mode or slave mode, and not in both modes at the same time. Therefore, Bit 6 (DISSLAVE) and Bit 0 (MASTER) in register CR shall not be set to 0 and 1 respectively (or 1 and 0 respectively).

The functional block diagram of I2C is as follows:

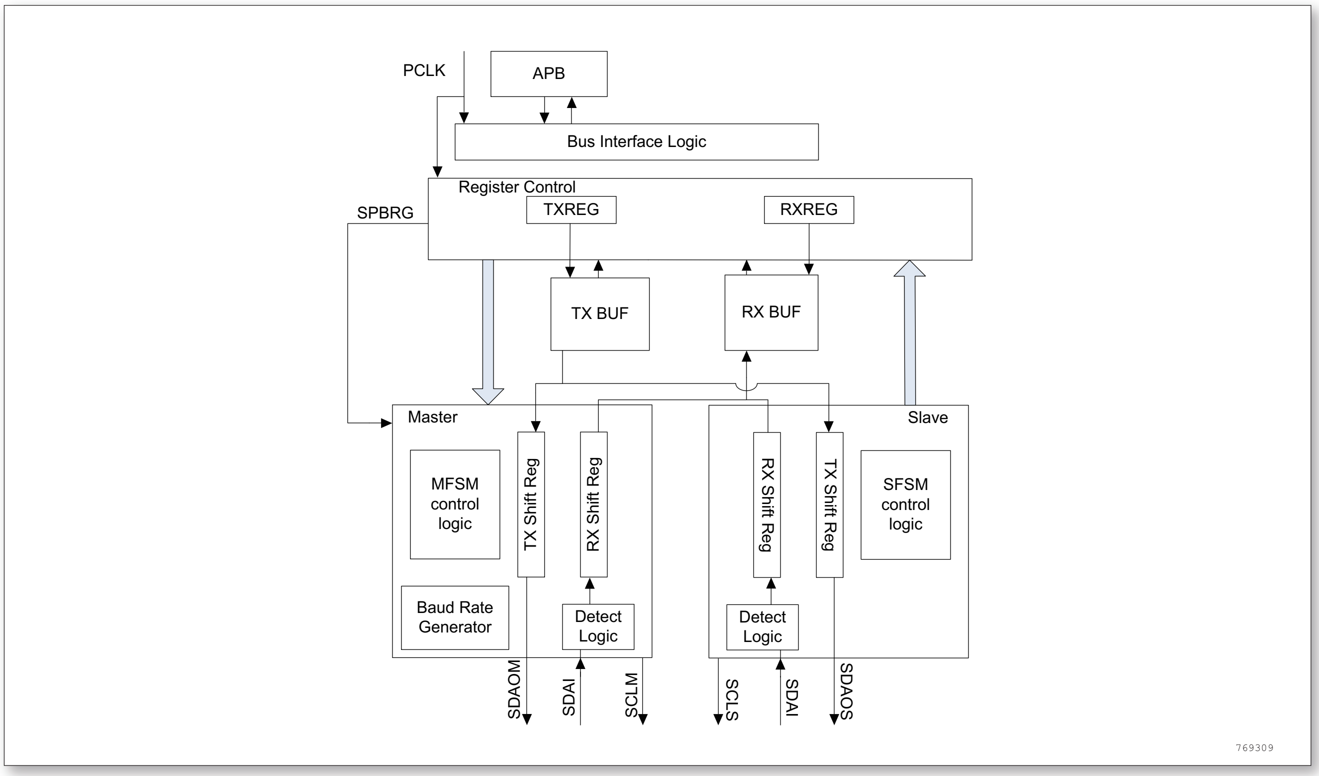


Figure 226. I2C Functional Block Diagram

19.4.1 Slave mode

The following describes the procedure flow chart of the slave mode

Initial configuration

1. Write 0 to Bit 0 of ENR register, to disable I2C.
2. Configure the slave address by initializing the SAR register. The address is that the I2C interface responds to.
3. Configure the specified address format of CR register (set bit 3 to select the 7-bit or 10-bit address format). Write 0 to Bit 6 of the CR register (DISSLAVE) and write 0 to bit 0 (MASTER).
4. Write 1 to bit 0 in the ENR register, enabling the I2C interface module.

Single byte operation of slave transmitter

When the I2C interface is addressed by another I2C master and requests data, the I2C interface operates in the slave transmitter mode as follows:

1. Other I2C master devices initialize the I2C transfer, with the transmit address matching the slave address in the SAR register.
2. The I2C interface responds to the address sent, identifying the transmission direction is in the slave transmitter mode.
3. The I2C interface generates the RD_REQ interrupt (Bit 5 of Register RAWISR) and sets the SCL line low. The bus is always in waiting state until the software responds.

If the RD_REQ interrupt is masked (register IMR5 = 0), it is recommended that the CPU periodically inquires the RAWISR register.

1. Setting bit 5 of RAWISR is equivalent to generating an RD_REQ interrupt.
2. The software shall meet the requirements for I2C transmission.
3. The time interval is usually around 10 SCL clock cycles. For example, at 400 kbps, the time interval is 25 us.
4. If the Tx FIFO is still loaded with data before receiving a read request, the I2C interface will generate a TX_ABRT interrupt (RAWISR6) and clear the data in the Tx FIFO.
5. The software writes the data to the DR register (its bit 8 is set to 0).
6. Software shall first clear RD_REQ and TX_ABRT interrupts in the RAWISR registers (bits 5 and 6 respectively)
7. The I2C interface releases SCL and sends a data byte.
8. The master device sends a repeated Start condition, to control the bus, or sends a Stop condition to release the bus.

Single byte operation of slave receiver

When the I2C interface is addressed by other master devices and data is sent, the I2C interface operates in the slave receiver mode, as follows:

1. Other I2C master devices initialize the I2C transfer, with the transmit address matching the slave address in the SAR register.
2. The I2C interface responds to the address sent, identifying the transmission direction is in the slave receiver mode.
3. The I2C interface receives the data sent by the master and stores it in the receive buffer.
4. The I2C interface generates an RX_FULL interrupt (RAWISR2). If the RX_FULL interrupt is masked (IMR2 = 0), it is recommended that the software periodically inquire the SR register. When bit 3 of SR register (RFNE) is 1, it is equivalent to the RX_FULL interrupt generated.
5. The software obtains the received data by reading bit 7:0 in the DR register.
6. The master device sends a repeated Start condition, to control the bus, or sends a Stop condition to release the bus.

Block transmission of slave

In the standard I2C protocol, all data is processed in single byte, and the program responds to the master's read request by writing a byte to the slave's Tx FIFO. When a slave (salve transmitter) receives a read request (RD_REQ) from the master (master receiver), at least one data is sent to the Tx FIFO of slave transmitter. This I2C interface module enables processing multiple data in the x FIFO, therefore, for the next read request, an interrupt is not needed to fetch data, thus greatly reducing the waiting time caused by each data interruption.

This mode only acts when the I2C interface is in slave transmitter mode. If the master transmitter responds to the data transferred by the slave transmitter, there is no data in the slave's TX FIFO; the I2C interface will set the SCL line of the I2C bus low until the read request interrupt (RD_REQ) is generated and the TX FIFO data is ready before releasing the SCL line.

If the RX_REQ interrupt is masked (ISR5 = 0), the software can periodically inquire and read RAWISR register. When reading RAWISR5, returning to 1 is equivalent to generating the RX_REQ interrupt.

The RD_REQ interrupt is generated due to the read request, like an interrupt, it must be cleared when exiting the Interrupt Service Routine (ISR). One or more bytes of data can be written to the TX FIFO in an Interrupt Service Routine (ISR). In the process of transferring these bytes to the master, if the master responds to the last byte, the slave must generate the RD_REQ interrupt request again. This is because that the master requires more data.

If the master receives n bytes from the I2C interface, but the number of data written to the Tx FIFO by the program is greater than n, the slave will clear the Tx FIFO and ignore the extra words after transmitting data of required n bytes.

19.4.2 Main mode

Initial configuration

1. Disable the I2C interface by setting ENR0 = 0
2. Configure bit 2:1 of the CR register, to set the rate mode (standard mode and fast mode) for I2C operation. In addition, ensure bit 6 (DISSLAVE) is 1, and bit 0 (MASTER) is 1.
3. Write the I2C device address to the TAR register. Set this register, which can be configured as a broadcast address or start byte command.
4. Set ENR0, to enable the I2C interface.
5. Write the transferred data and the transfer direction to the DR register. If the DR register is configured before the I2C interface is enabled, data and commands will be lost since the buffer is cleared when the I2C interface is disabled.

By following the above steps, I2C interface will generate a Start condition and send the address byte data to the I2C bus.

Master transmitter and master receiver

The I2C interface supports dynamic switching of reads and writes. When transmitting data, write data to the lower byte (DR) of the I2C RX/TX data buffer and command register, and set the CMD bit to 0, to allow a write operation. For the next read command, it is unnecessary to set the low byte of the DR register, and the CMD bit shall be 1. If the transmitter's FIFO is null, the I2C module sets SCL low until the next command is written to the transmitter's FIFO.

Program flow chart

The following flow chart is a program example of the I2C interface used as a master:

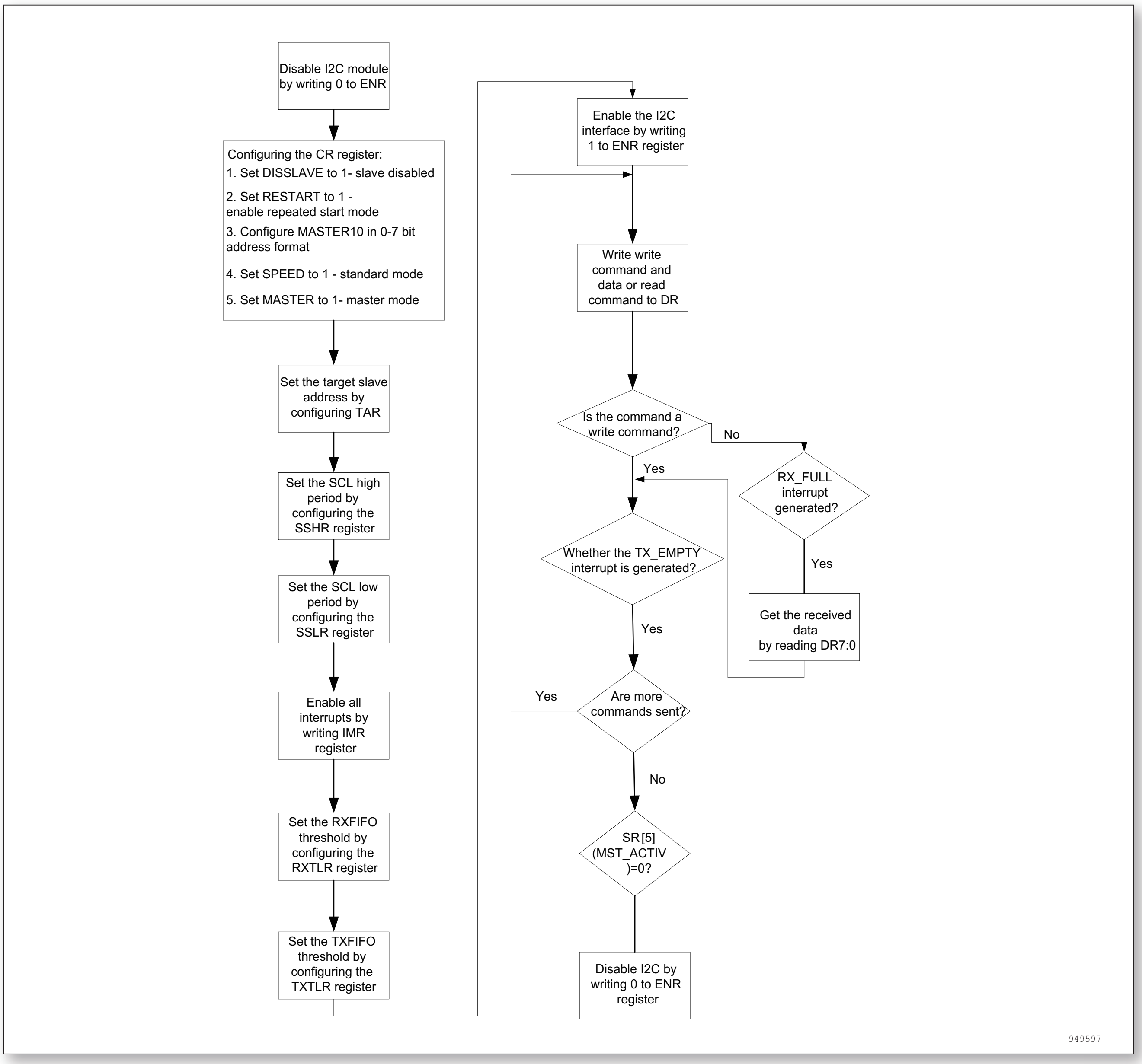

Disable I2C module by writing 0 to ENR
Configuring the CR register:
1. Set DISSLAVE to 1- slave disabled
2. Set RESTART to 1 - enable repeated start mode
3. Configure MASTER10 in 0-7 bit address format
4. Set SPEED to 1 - standard mode
5. Set MASTER to 1- master mode
Set the target slave address by configuring TAR
Set the SCL high period by configuring the SSHR register
Set the SCL low period by configuring the SSLR register
Enable all interrupts by writing IMR register
Set the RXFIFO threshold by configuring the RXTLR register
Set the TXFIFO threshold by configuring the TXTLR register
Enable the I2C interface by writing 1 to ENR register
Write write command and data or read command to DR
Is the command a write command?
No
Yes
RX_FULL interrupt generated?
Yes
Get the received data by reading DR7:0
Whether the TX_EMPTY interrupt is generated?
Yes
Are more commands sent?
Yes
No
SR[5] (MST_ACTIV)=0?
Disable I2C by writing 0 to ENR register
949597

Figure 227. Flow Chart of I2C Interface Master

19.4.3 I2C abort transmission

The ABRT control bit in the ENR register allows the software to disable the I2C bus before transmitting the command in the TX FIFO. In response to the ABORT request, the I2C module issues a Stop condition to the I2C bus while clearing the TX FIFO. The transfer of operation value can be aborted in the master mode.

Procedure

1. Stop writing new commands to the Tx FIFO (DR)
2. Disable the transmission of DMA by setting TDMAE = 0 when operating in DMA mode
3. Set the ABRT bit of ENR register to 1
4. Wait for TX_ABRT interrupt

19.5 Communication using DMA

The I2C interface supports data transmission and reception through DMA, namely, DMA transmission or DMA reception can be enabled separately by setting the corresponding bit in the DMA register. A DMA request will be generated when the data register becomes null during the transmission or becomes full upon reception. The DMA request must be acknowledged before the end of the current byte transfer.

Transmission using DMA

DMA mode can be enabled for transmission by setting the TXEN bit in the DMA register. Data will be loaded from a Memory area predefined to the DR register during data transfer.

Reception using DMA

DMA mode can be enabled for reception by setting the RDMAE bit in the DMA register. Data will be loaded from the DR register to a Memory area predefined during each data reception after DMA channel is configured by I2C.

19.6 I2C interrupt

The following table lists the I2C interrupt bits and how they are set and cleared. Some bits are set by hardware and cleared by software; the other bits are set and cleared by hardware.

Table 60. Set and Clear 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 following figure depicts that interrupt bits in the interrupt register are set by hardware and cleared by software.

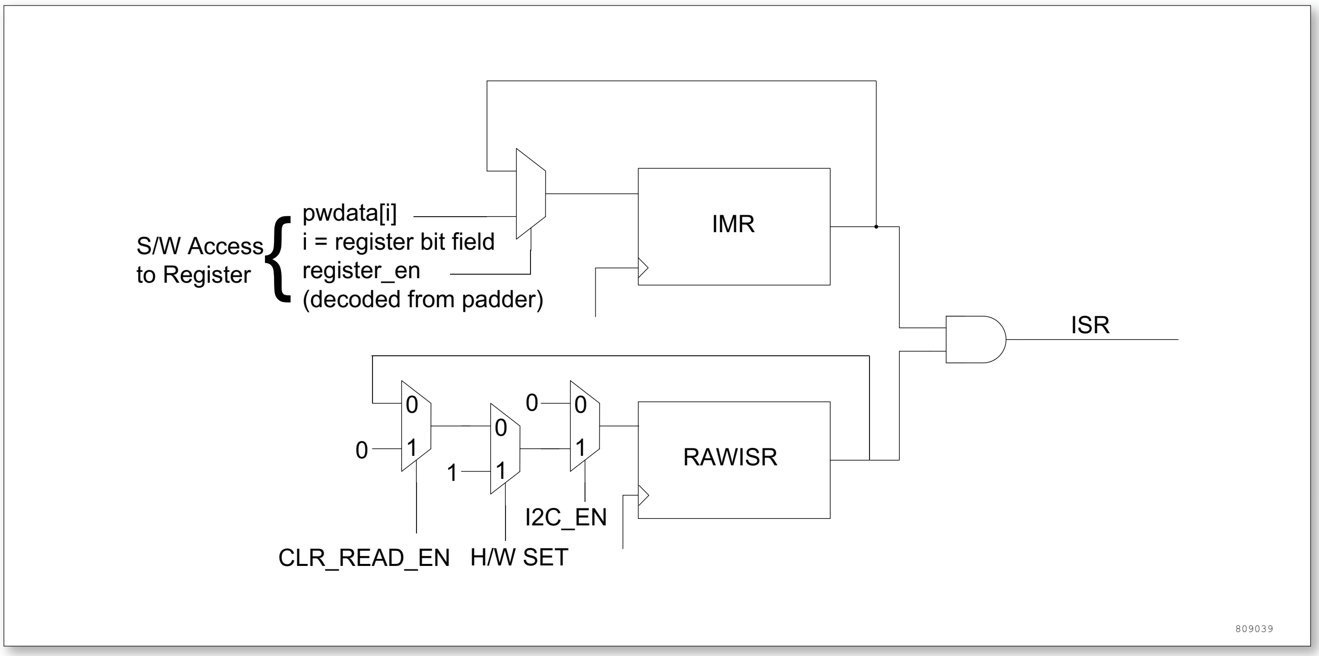


Figure 228. Interrupt Mechanism

19.7 I2C register description

Table 61. I2C Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	I2C_CR	I2C control register	0x007F	section 19.7.1
0x04	I2C_TAR	I2C destination address register	0x0055	section 19.7.2
0x08	I2C_SAR	I2C slave address register	0x0055	section 19.7.3
0x10	I2C_DR	I2C data command register	0x0001	section 19.7.4
0x14	I2C_SSHR	Standard mode I2C clock high counter register	0x0190	section 19.7.5
0x18	I2C_SSLR	Standard mode I2C clock low counter register	0x01D6	section 19.7.6
0x1C	I2C_FSHR	Fast mode I2C clock high counter register	0x0036	section 19.7.7
0x20	I2C_FSLR	Fast mode I2C clock low counter register	0x0082	section 19.7.8
0x2C	I2C_ISR	I2C interrupt status register	0x0000	section 19.7.9
0x30	I2C_IMR	I2C interrupt mask register	0x08FF	section 19.7.10
0x34	I2C_RAWISR	I2C RAW interrupt register	0x0000	section 19.7.11
0x38	I2C_RXTLR	I2C reception threshold	0x0000	section 19.7.12
0x3C	I2C_TXTLR	I2C transmission threshold	0x0000	section 19.7.13
0x40	I2C_ICR	I2C combined and independent interrupt clear register	0x0000	section 19.7.14
0x44	I2C_RX_UNDER	I2C clear RX_UNDER interrupt register	0x0000	section 19.7.15
0x48	I2C_RX_OVER	I2C clears RX_OVER interrupt register	0x0000	section 19.7.16

Offset	Acronym	Register Name	Reset	Section
0x4C	I2C_TX_OVER	I2C clear TX_OVER interrupt register	0x0000	section 19.7.17
0x50	I2C_RD_REQ	I2C clear RD_REQ interrupt register	0x0000	section 19.7.18
0x54	I2C_TX_ABRT	I2C clear TX_ABRT interrupt register	0x0000	section 19.7.19
0x58	I2C_RX_DONE	I2C clear RX_DONE interrupt register	0x0000	section 19.7.20
0x5C	I2C_ACTIV	I2C clear ACTIVITY interrupt register	0x0000	section 19.7.21
0x60	I2C_STOP	I2C clear STOP_DET interrupt register	0x0000	section 19.7.22
0x64	I2C_START	I2C clear START_DET interrupt register	0x0000	section 19.7.23
0x68	I2C_GC	I2C clear GEN_CALL interrupt register	0x0000	section 19.7.24
0x6C	I2C_ENR	I2C enable register	0x0000	section 19.7.25
0x70	I2C_SR	I2C status register	0x0006	section 19.7.26
0x74	I2C_TXFLR	I2C transmitter buffer level register	0x0000	section 19.7.27
0x78	I2C_RXFLR	I2C receiver buffer level register	0x0000	section 19.7.28
0x7C	I2C_HOLD	I2C SDA hold time register	0x0001	section 19.7.29
0x88	I2C_DMA	I2C DMA control register	0x0000	section 19.7.30
0x94	I2C_SETUP	I2C SDA setup time register	0x0064	section 19.7.31
0x98	I2C_GCR	I2C general call ACK register	0x0001	section 19.7.32

19.7.1 I2C control register(I2C_CR)

Offset address: 0x00

Reset value: 0x007F

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					RESTART	STOP	EMPINT	STOPINT	DISSLAVE	REPEN	MASTER10	SLAVE10	SPEED		MASTER
					rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15 : 11	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
10	RESTART	rw	0x00	Whether to generate a RESTART signal before transmission or reception This bit is only valid when IC_EMPTYFIFO_HOLD_MASTER_EN is set to ’1’ 1: If the RESTART signal is ”1”, the data is received or sent (according to a RESTART signal will be generated before data reception or transmission (depending on CMD), regardless of the previous command changes the direction of data transmission. If IC_RESTART_EN signal is set to ’0’, the STOP signal will follow the START signal 0: If the RESTART signal is set to ’1’, the RESTART signal is generated only when the previous command changes the direction of data transmission. If the RESTART signal is set to ’0’, the STOP signal will follow the START signal
9	STOP	rw	0x00	STOP: Whether to generate a STOP signal after transmission or reception This bit is only valid when IC_EMPTYFIFO_HOLD_MASTER_EN is set to ’1’ 1: A STOP signal is generated after the current byte, regardless of the Tx FIFO is null. If the Tx FIFO is not null, the master immediately issues a new transmission and bus arbitration signal. 0: A STOP signal is not generated after the current byte, regardless of the Tx FIFO is null. The master continues the current transmission (data transmission or reception depending on CMD). If the Tx FIFO is null, the master will set SCL low, to pend the bus until Tx FIFO receives new data
8	EMPINT	rw	0x00	This bit controls the generation of TX_EMPTY interrupt. Refer to the IC_RAW_INTR_STAT register for details.
7	STOPINT	rw	0x00	In the slave mode, whether a STOP_DET interrupt is generated. 1:STOP_DET interrupt is generated when the address matches 0:STOP_DET interrupt is generated regardless of the address match. This bit is only applicable to slave mode. Note: During the addressing of broadcasting address, if this bit is set, the slave will not generate a STOP_DET interrupt. The STOP_DET interrupt is generated only when the transmitted address matches the slave address.

Bit	Field	Type	Reset	Description
6	DISSLAVE	rw	0x01	This bit controls whether I2C has its slave disabled 0: Slave enabled 1: Slave disabled
5	REPEN	rw	0x01	Determines whether RESTART conditions may be sent when acting as a master 0: Disabled 1: Enabled When RESTART is disabled, the following functions cannot be performed when the I2C interface acts as a master: Send start byte Change the transmission direction in combined format mode Read data 10-bit address format The RESTART condition is replaced by sending a Stop condition first and then transmitting a Start condition. If the above operation is executed, bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register is set.
4	MASTER10	r	0x01	Address mode when acting as a master 0: 7-bit address format 1: 10-bit address format
3	SLAVE10	rw	0x01	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses 0: 7-bit addressing address. The I2C interface ignores 10-bit addressing. For 7-bit addressing, only the lower 7 bits of IC_SAR register is compared. 1: 10-bit addressing address. I2C only responds to 10-bit addressing; the receiving address is compared with 10 bits of IC_SAR.
2:1	SPEED	rw	0x03	These bits control at which speed the I2C operates This configuration is only valid when the I2C interface operates in the master mode. 1: Standard mode (0 ~ 100 Kbps) 2: Fast mode (400 Kbps)
0	MASTER	rw	0x01	This bit controls whether the I2C master is enabled 0: Master disabled 1: Master enabled

The DISSLAVE (bit 6) and MASTER (bit 0) configurations are listed in the following table:

Table 62. DISSLAVE (Bit 6) and MASTER (Bit 0) Configurations

DISSLAVE CR[6]	MASTER CR[0]	Status
0	0	Slave device

DISSLAVE CR[6]	MASTER CR[0]	Status
0	1	Configuration error
1	0	Configuration error
1	1	Master device

19.7.2 I2C destination address register(I2C_TAR)

Offset address: 0x04

Reset value: 0x0055

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

Bit	Field	Type	Reset	Description
15 : 12	Reserved			Always read as 0.
11	SPECIAL	rw	0x00	This bit indicates whether the software executes a special command (general call or start byte command) 0: Ignore the 10-bit GC, and use the ADDR bit normally 1: Execute special I2C commands such as GC bit
10	GC	rw	0x00	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 address. Only perform write operation when sending a general call address. The I2C interface always operates in broadcast address mode until SPECIAL (bit 11) is cleared. 1: Start byte command
9 : 0	ADDR	rw	0x55	This is the target address for any master transaction When a broadcast address is sent, these bits can be ignored. To generate the start byte command, the CPU only needs to write these bits once.

19.7.3 I2C slave address register(I2C_SAR)

Offset address: 0x08

Reset value: 0x0055

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

Bit	Field	Type	Reset	Description
15 : 10	Reserved			Always read as 0.
9 : 0	ADDR	rw	0x55	When the I2C interface operates in the slave mode, for slave memory address in 7-bit address format, ADDR [6:0] is only valid.

19.7.4 I2C data command register(I2C_DR)

Offset address: 0x10

Reset value: 0x0001

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

Bit	Field	Type	Reset	Description
15 : 9	Reserved			Always read as 0.
8	CMD	w	0x00	Control read or write operations in master mode 1: Read 0: Write When a command is sent to TX FIFO, this bit is used to distinguish between read and write commands. In the slave receiver mode, the write operation of this bit is ignored. In the slave transmitter mode, writing 0 indicates that the data of the DR register is ready to be sent.
7 : 0	DAT	rw	0x01	I2C bus data to be sent or received

19.7.5 Standard mode I2C clock high counter register(I2C_SSHR)

Offset address: 0x14

Reset value: 0x0190

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

Bit	Field	Type	Reset	Description
15 : 0	CNT	rw	0x0190	SCL clock high period in standard mode of I2C interface Note: This register has a configurable value between 6 and 65525. This is because the I2C interface uses a 16-bit counter; the I2C bus is in the idle state when the counter value is SSHR + 10.

19.7.6 Standard mode I2C clock low counter register(I2C_SSLR)

Offset address: 0x18

Reset value: 0x01D6

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

Bit	Field	Type	Reset	Description
15 : 0	CNT	rw	0x01D6	The minimum SCL clock low period is 8 in the I2C interface standard mode.

19.7.7 Fast mode I2C clock high counter register(I2C_FSHR)

Offset address: 0x1C

Reset value: 0x0036

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

Bit	Field	Type	Reset	Description
15 : 0	CNT	rw	0x0036	SCL clock high period in fast mode of I2C interface This register is read-only and returns 0 when I2C operates in standard mode, with the minimum value of 6.

19.7.8 Fast mode I2C clock low counter register(I2C_FSLR)

Offset address: 0x20

Reset value: 0x0082

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

Bit	Field	Type	Reset	Description
15 : 0	CNT	rw	0x0082	SCL clock low period in fast mode of I2C interface This register is read-only and returns 0 when I2C operates in standard mode, with the minimum value of 8.

19.7.9 I2C interrupt status register(I2C_ISR)

Offset address: 0x2C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 14	Reserved			Always read as 0.
13 : 0	ISR	r	0x0000	Refer to the RAWISR register for specific description of each bit

19.7.10 I2C interrupt mask register(I2C_IMR)

Offset address: 0x30

Reset value: 0x08FF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				GC	START	STOP	ACTIV	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15 : 12	Reserved			Always read as 0.
11 : 0	IMR	rw	0x08FF	Each bit is masked and mapped to the ISR.

19.7.11 I2C RAW interrupt register(I2C_RAWISR)

Offset address: 0x34

Reset value: 0x0000

The difference between RAWISR and ISR registers is that the former is not masked.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				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	Type	Reset	Description
15 : 12	Reserved			Always read as 0.
11	GC	r	0x00	General call This bit is set when a general call address is received. The I2C interface is disabled or cleared when the CPU reads the GC register. I2C stores the received data in the receiver buffer.

Bit	Field	Type	Reset	Description
10	START	r	0x00	Start condition detection Regardless of the I2C interface operates in the master or slave mode, this bit is set once the Start or repeated Start condition is detected on the I2C interface.
9	STOP	r	0x00	Stop condition detection The status of this bit is based on the status of the STOPINT in the CR register when STOPINT = 0 Regardless of the I2C interface operates in the master or slave mode, this bit is set once the Stop condition is detected on the I2C interface.In slave mode, a STOP interrupt is generated regardless of whether the addressing is matched or not. When STOPINT = 1 In master mode (MASTER = 1), this bit shows if a Stop condition occurs on the I2C interface. In slave mode (MASTER = 0), a STOP interrupt is generated only when the slave address is matched successfully.
8	ACTIV	r	0x00	I2C interface is enabled, this bit is used to capture the active state of the I2C module After being set, this bit can only be cleared by the following four methods: Disable I2C interface Read ACTIV register Rread ICR register System reset Once set, this bit can only be cleared by the above method. Even if I2C is idle, this bit also remains high until it is cleared.
7	RX_DONE	r	0x00	Transmit done When I2C is used as a slave transmitter, this bit will be set if the master fails to respond after sending one byte of data. This case happens at the last byte transferred, indicating the end of the transfer.
6	TX_ABRT	r	0x00	Transmit abort When the I2C interface acts as a transmitter, this bit is set if the data in the buffer is failed to be fully sent. Note: The transmit abort bit will clear the receiver and transmitter buffers in the I2C interface. The transmitter buffer is in a refresh state until the TX_ABRT register is read. Once the read operation is performed, the transmitter will receive new data from the APB bus.

Bit	Field	Type	Reset	Description
5	RD_REQ	r	0x00	Read request When I2C acts as a slave, other masters are set when attemptting to read data from the I2C interface. The I2C interface keeps the bus in a wait state (SCL = 0) until the interrupt is processed, which means that the I2C interface is successfully addressed by other masters as a slave and requires data transmission. The processor must respond to the interrupt and then write data to the DR register. This bit is cleared when the processor reads the RD_REQ register.
4	TX_EMPTY	r	0x00	Transmit buffer empty The status of this bit depends on the EMPINT state in the CR register: when EMPINT is 0 and the transmitter buffer is null, this bit is set; when EMPINT is 1, the transmitter buffer is null and the operation of internal shift register is finished, this bit is set This bit is automatically cleared by hardware when the transmitter buffer is not null.
3	TX_OVER	r	0x00	Transmit buffer over If the transmit buffer is full, and the processor writes new data, causing overflow, this bit will be set.
2	RX_FULL	r	0x00	Receive buffer not empty This bit is set when the receive buffer is not null. This bit will be cleared by hardware when the receive buffer is not null.
1	RX_OVER	r	0x00	Receive buffer over This bit will be set when the receive buffer is full and new data is received. The I2C interface will respond at this point, but new data will be lost.
0	RX_UNDER	r	0x00	Receive buffer under This bit will be set when the processor reads the DR register and the RX FIFO is null.

19.7.12 I2C reception threshold(I2C_RXTLR)

Offset address: 0x38

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 8	Reserved			Always read as 0.
7 : 0	TL	r	0x00	Receive FIFO threshold level This bit enables controlling the RX_FULL interrupt trigger.

19.7.13 I2C transmission threshold(I2C_TXTLR)

Offset address: 0x3C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 8	Reserved			Always read as 0.
7 : 0	TL	r	0x00	Receive FIFO threshold level This bit enables controlling the TX_EMPTY interrupt trigger.

19.7.14 I2C combined and independent interrupt clear register(I2C_ICR)

Offset address: 0x40

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	ICR	r	0x00	When this register is read, all combined interrupts and independent interrupts. Those that can be automatically cleared by hardware will not be cleared b this bit, and only those that can be cleared by software will be cleared.

19.7.15 I2C clear RX_UNDER interrupt register(I2C_RX_UNDER)

Offset address: 0x44

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	RX_UNDER	r	0x00	Clear RX_UNDER interrupt (RAWISR0) by reading this register.

19.7.16 I2C clears RX_OVER interrupt register(I2C_RX_OVER)

Offset address: 0x48

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	RX_OVER	r	0x00	Clear RX_OVER interrupt (RAWISR1) by reading this register.

19.7.17 I2C clear TX_OVER interrupt register(I2C_TX_OVER)

Offset address: 0x4C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	TX_OVER	r	0x00	Clear TX_OVER interrupt (RAW_ISR3) by reading this register.

19.7.18 I2C clear RD_REQ interrupt register(I2C_RD_REQ)

Offset address: 0x50

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	RD_REQ	r	0x00	Clear RD_REQ interrupt (RAW_ISR5) by reading this register.

19.7.19 I2C clear TX_ABRT interrupt register(I2C_TX_ABRT)

Offset address: 0x54

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	TX_ABRT	r	0x00	Clear the TX_ABRT interrupt (RAWISR6) by reading this register Release the TX FIFO from the refresh/reset state, to receive the written data.

19.7.20 I2C clear RX_DONE interrupt register(I2C_RX_DONE)

Offset address: 0x58

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	RX_DONE	r	0x00	Clear RX_DONE interrupt (RAWISR7) by reading this register.

19.7.21 I2C clear ACTIVITY interrupt register (I2C_ACTIV)

Offset address: 0x5C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	ACTIV	r	0x00	Read this register to clear ACTIV interrupt (RAWISR8) if the I2C bus is inactive If I2C is still active, the ACTIV interrupt will continue to be set. This bit is cleared by hardware when the I2C module is disabled or when the I2C bus is no longer active. The status of ACTIV (bit 8) in the RAWISR can be obtained by reading this register.

19.7.22 I2C clear STOP_DET interrupt register(I2C_STOP)

Offset address: 0x60

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	STOP	r	0x00	Clear STOP interrupt (RAWISR9) by reading this register.

19.7.23 I2C clear START_DET interrupt register(I2C_START)

Offset address: 0x64

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	START	r	0x00	Clear START interrupt (RAWISR10) by reading this register

19.7.24 I2C clear GEN_CALL interrupt register(I2C_GC)

Offset address: 0x68

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	GC	r	0x00	Clear GC interrupt (RAWISR11) by reading this register

19.7.25 I2C enable register(I2C_ENR)

Offset address: 0x6C

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 2	Reserved			Always read as 0.
1	ABORT	rw	0x00	I2C transfer abort 0: The abort did not occur or has ended 1: Abort is in progress The I2C transfer can be aborted by software when the I2C module is set as a master. Once being set, this bit cannot be cleared immediately, the I2C module control logic will generate a STOP condition and clear the transmit buffer after completing the current transfer, and generates a TX_ABRT interrupt after the abort. This ABORT bit is automatically cleared after the abort operation.

Bit	Field	Type	Reset	Description
0	ENABLE	rw	0x00	I2C mode enable 0: Disable the I2C module (transmit and receive buffers remain erased) 1: Enable the I2C module

19.7.26 I2C status register(I2C_SR)

Offset address: 0x70

Reset value: 0x0006

This register is read-only and indicates the current transfer and buffer status. The status bits do not generate an interrupt.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									SLV_ACTIV	MST_ACTIV	RFF	RFNE	TFE	TFNE	ACTIV
									r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15 : 7	Reserved			Always read as 0.
6	SLV _ACTIV	r	0x00	Slave FSM activity status 0: The slave state machine is in the IDLE state, so the I2C slave part is inactive. 1: The slave state machine is not in the IDLE state, so the I2C slave part is active.
5	MST _ACTIV	r	0x00	Master FSM activity status 0: The master state machine is in the IDLE state, so the I2C master part is inactive. 1: The master state machine is not in the IDLE state, so the I2C master part is active.
4	RFF	r	0x00	Receive FIFO completely full 0: receive buffer not full 1: Receive buffer full
3	RFNE	r	0x00	Receive FIFO not empty 0: Receive buffer empty 1: Receive buffer not empty
2	TFE	r	0x01	Transmit FIFO completely empty 0: Transmit buffer not empty 1: Transmit buffer empty
1	TFNF	r	0x01	Transmit FIFO not full 0: Transmit buffer full 1: Transmit buffer not full
0	ACTIV	r	0x00	I2C activity status The MST_ACTIV bit has the OR relationship with SLV_ACTIV bit.

19.7.27 I2C transmitter buffer level register(I2C_TXFLR)

Offset address: 0x74

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 2	Reserved			Always read as 0.
1 : 0	CNT	r	0x00	The number of valid data in the transmit buffer (0 ～2)

19.7.28 I2C receiver buffer level register(I2C_RXFLR)

Offset address: 0x78

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 2	Reserved			Always read as 0.
1 : 0	CNT	r	0x00	The number of valid data in the receive buffer (0 ～2)

19.7.29 I2C SDA hold time register(I2C_HOLD)

Offset address: 0x7C

Reset value: 0x0001

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

Bit	Field	Type	Reset	Description
31 : 24	Reserved			Always read as 0.
23 : 16	RX_HOLD	r	0x00	SDA hold time when the I2C device acts as receiver, with the unit of APB1 system clock cycle

Bit	Field	Type	Reset	Description
15 : 0	TX_HOLD	r	0x01	SDA hold time when the I2C device acts as transmitter, with the unit of APB1 system clock cycle

19.7.30 I2C DMA control register(I2C_DMA)

Offset address: 0x88

Reset value: 0x0000

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

Bit	Field	Type	Reset	Description
15 : 2	Reserved			Always read as 0.
1	TXEN	rw	0x00	Transmit DMA enable 0: transmit DMA disabled 1: Receive DMA enabled
0	RXEN	rw	0x00	Receive DMA enable 0: Receive DMA disabled 1: Receive DMA enabled

19.7.31 I2C SDA setup time register(I2C_SETUP)

Offset address: 0x94

Reset value: 0x0064

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

Bit	Field	Type	Reset	Description
15 : 8	Reserved			Always read as 0.
7 : 0	CNT	rw	0x64	SDA setup If the recommended delay time is 1000 nS, and the APB1 clock frequency is 10 MHz, the register is set to 11, with the minimum of 2.

19.7.32 I2C general call ACK register(I2C_GCR)

Offset address: 0x98

Reset value: 0x0001

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

Bit	Field	Type	Reset	Description
15 : 1	Reserved			Always read as 0.
0	GC	rw	0x01	ACK general call 1: Response after receiving a general call 0: No response and no interruption after receiving a general call

20 Universal asynchronous receiver transmitter(UART)

Universal asynchronous receiver transmitter(UART)

20.1 UART introduction

The universal asynchronous receiver transmitter (UART) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The UART offers a very wide range of baud rates using a fractional baud rate generator. It supports synchronous one-way communication and half-duplex single wire communication as well as modem operations (CTS/RTS).

High speed data communication is possible by using the DMA for multibuffer configuration.

20.2 UART main features

- Support RS-232S protocol in asynchronous mode, and meet Industry Standard 16550
- Support DMA requests
- Full-duplex synchronous operation
- Fractional baud rate generator system
- Programmable baud rate shared by transmitter and receiver
- Separate transmit and receive buffer registers
- Embedded 1 byte transmit and 32 byte receive buffer
- low level first for data transmission and reception
- Starting from start bit, followed by a data bit (the output data length includes 5 bits, 6 bits, 7 bits, 8 bits), and ending with the stop bit. Alternatively, parity check bit is optional, which is set before the stop bit and after the data bit.
- The 9th bit can be used for synchronous frame configuration.
- Support hardware odd or even check, generation and detection
- Line break generation and detection
- Support hardware auto flow control
- Support the following interrupt sources:
  - Transmitter BUFFER empty
  - Receiver data valid
  - Receive buffer overflow
  - Frame error
  - Parity error
  - Receive break frame
  - Transmit shift register completed

- Send disconnected frame complete
- Receiving sync frame

20.3 UART functional description

At least two pins are required for any UART bidirectional communication: receive data input (RX) and transmit data output (TX).

RX: Receive data serial input. Data is recovered by oversampling techniques, to distinguish between data and noise.

TX: Transmit data output. When the transmitter is disabled, the output pin is returned to its I/O port configuration. The TX pin is high when the transmitter is activated and no data is transmitted.

- The bus shall be idle before transmission or reception
- One start bit
- One data word (5, 6, 7 or 8 bits) with the least significant bit first
- 0.5, 1, 1.5, 2 stop bits, thus indicating the end of the data frame
- Use the fractional baud rate generator –a representation of 16-bit integers and 4bit decimals

The following pin is required in hardware flow mode:

- nCTS: Clear transmission. If it is high, it blocks the next data transmission at the end of the current data transmission.
- nRTS: Transmit request; the low level indicates that the UART is ready to receive data.

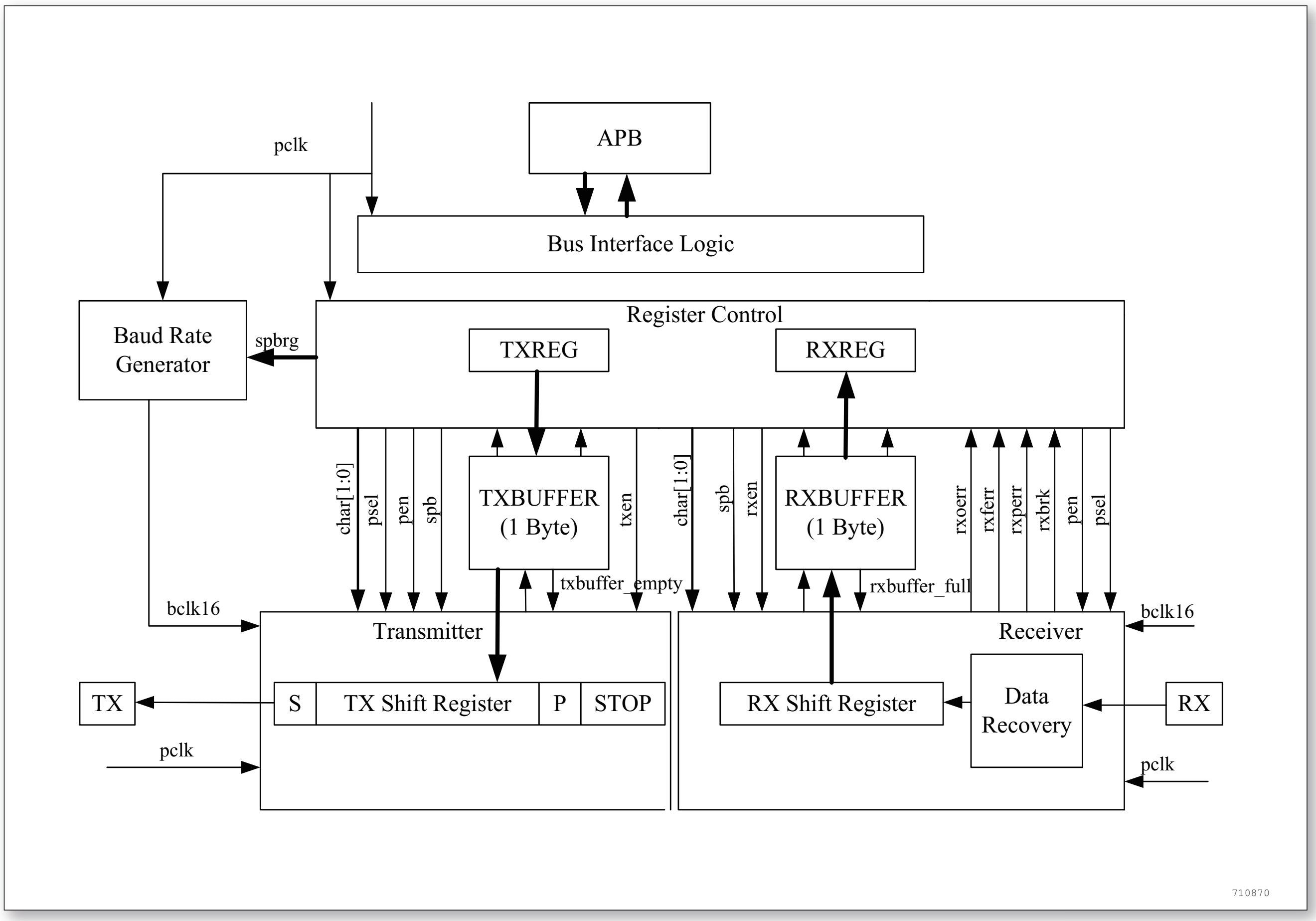


Figure 229. UART Block Diagram

20.3.1 UART character description

Word length may be selected as 5～8 bits by programming the CHAR bit in the UART_CCR register. The TX pin is in low state during the start bit, and is in high state during the stop bit.

An Idle character is interpreted as an entire frame of “1” s followed by the start bit of the next frame which contains data (the number of ‘1’ s will include the number of stop bits).

A Break character is interpreted on receiving ‘0’ s for a frame period. At the end of the break frame the transmitter inserts either 1 or 2 stop bits (logic “1” bit) to acknowledge the start bit.

Transmission and reception are driven by a common baud rate generator; the clock for each is generated when the enable bit is set respectively for the transmitter and receiver.

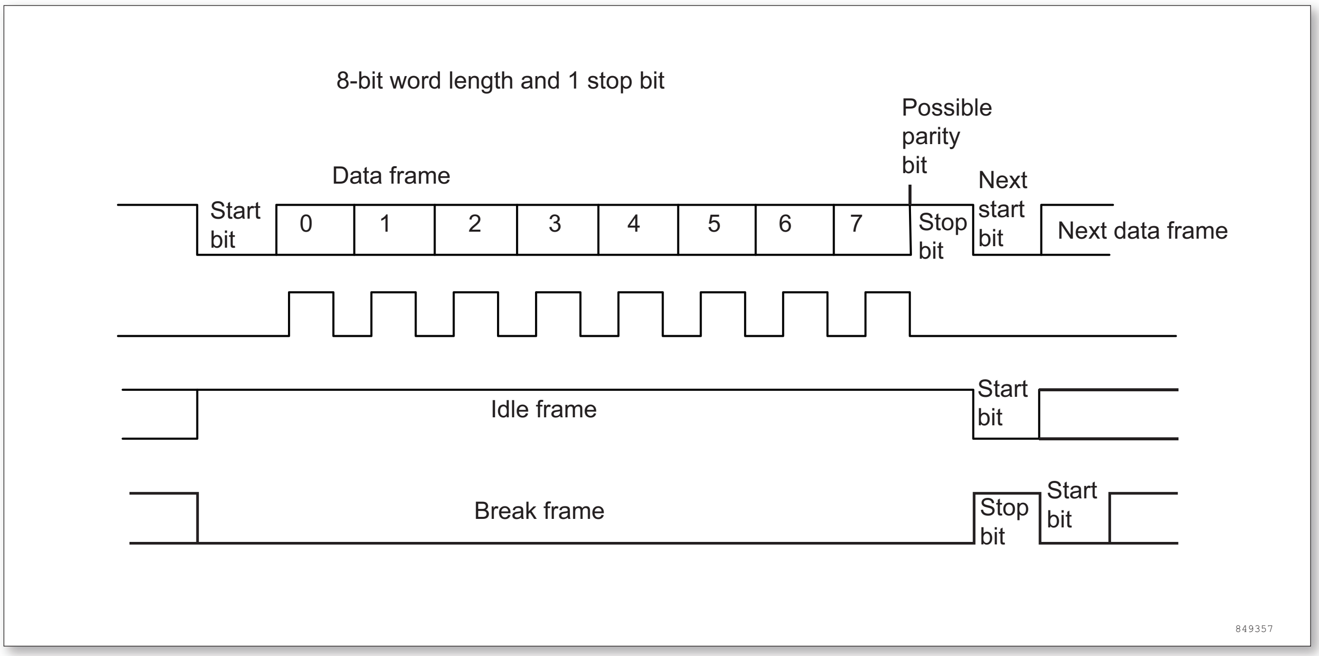


Figure 230. UART Timing

20.3.2 Transmitter

The transmitter can send data words of 5 ~ 8 bits depending on the CHAR bit status. When the transmit enable bit (TXEN) is set, the data in the transmit shift register is output on the TX pin.

Character transmission

During a UART transmission, data shifts out least significant bit first on the TX pin. In this mode, the UART_TDR register consists of a buffer between the internal bus and the transmit shift register.

Every character is preceded by a start bit. The character is terminated by a configurable number of stop bits.

The TXEN bit shall not be reset during transmission of data. Otherwise, the data on the TX pin will be corrupted as the baud rate counters will get frozen, and the current data being transmitted will be lost.

Configurable stop bits

The number of stop bits to be transmitted with every character can be programmed by setting SPB bits.

A break frame will be 10 low bits followed by the stop bits, or 11 low bits followed by the stop bits.The RXBRK_INTF bit in the interrupt status register will be set when the break frame is received.

Procedure

1. Enable the UART by writing the UARTEN bit in UART_GCR register to 1.
2. Program the CHAR bit in UART_CCR to define the word length.
3. Program the number of stop bits (SPB) in UART_CCR.

4. Set the TXEN bit in UART_GCR.
5. Select the desired baud rate using the UART_BRR register.
6. Write the data to be sent in the UART_TDR register (this clears the TX_INTF bit). Repeat Step 6 for each data to be transmitted in case of single buffer.

Single byte communication

The TX_INTF bit is always cleared by a write to the data register. The TX_INTF bit is set by hardware and it indicates:

- The data has been moved from TDR to the shift register and the data transmission has started.
- The TDR register is cleared.
- The next data can be written in the UART_TDR register without overwriting the previous data.

This flag generates an interrupt if the TXIEN bit is set. When a transmission is taking place in UART, a write instruction to the UART_TDR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place in idle UART, a write instruction to the UART_TDR register places the data directly in the shift register, the data transmission starts, and the TX_INTF bit is immediately set. Meanwhile, TXBUF_EMPTY of UART_CSR is set. If a frame is transmitted (after the stop bit) and no new data is written to UART_TDR (TDR register null), the TXC bit will be set, indicating all transmission has been completed.

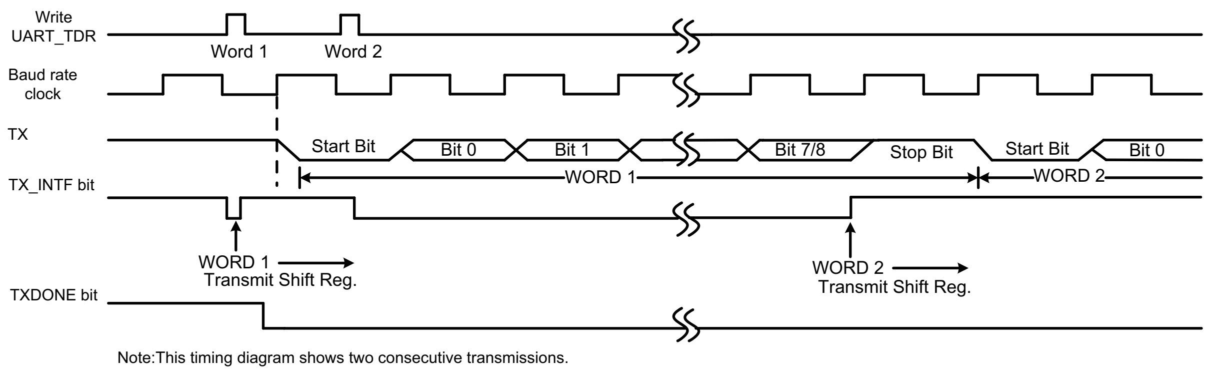


Figure 231. Status Bit Change During Transmission

Break character

Setting the BRK bit transmits a break character. If the BRK bit is set to '1', a break character is sent on the TX line after completing the current character transmission. This bit is reset by software when the break character is completed (during the stop bit of the break character). The UART inserts a logic 1 bit at the end of the last break frame, to guarantee the recognition of the start bit of the next frame.

20.3.3 Receiver

Character reception

During a UART reception, data shifts in least significant bit first through the RX pin. In this mode, the UART_RDR register consists of a buffer between the internal bus and the received shift register.

Procedure:

1. Enable the UART by writing the UARTEN bit in UART_GCR register to 1.
2. Program the CHAR bit in UART_CCR to define the word length.
3. Program the number of stop bits (SPB) in UART_CCR.
4. Select the desired baud rate using the UART_BRR register.
5. Set the RXEN bit UART_GCR. This enables the receiver which begins searching for a start bit.

When a character is received:

- The RX_INTF bit is set. It indicates that the content of the shift register is transferred to the RDR. In other words, data has been received and can be read (as well as its associated error flags).
- An interrupt is generated if the RXIEN bit is set.
- The error flags can be set if a frame error or an overrun error has been detected during reception.
- UART_RDR register is read by software, RX_INTF bit shall be cleared before the next character is received.

The RXEN bit shall not be reset while receiving data. If the RXEN bit is cleared during reception, the current byte to be received will be lost.

Break character

When a break frame is received, the UART is set and RXBRK_INTF interrupt is generated.

Overrun error

An overrun error occurs when a character is received before being read by UART_RDR.

When an overrun error occurs:

- The RXOERR_INTF bit is set.
- The RDR content will not be lost. The previous data is available when a read to UART_RDR is performed.
- The shift register will be overwritten. After that point, any data received will be lost.
- An interrupt is generated if the RXOERREN bit is set.

Frame error

The frame error is detected when the Stop bit is failed to be received and identified, then:

- RXFERR_INTF bit is set by hardware.

- Invalid data will not be transmitted to UART_RDR register from the shift register.
- An interrupt will be generated if RXFERREN bit is set.

20.3.4 9-bit data communication

If the B8EN control bit of the UART_CCR register is enabled, the UART enables the transmission and reception of 9-bit data, and can transmit and receive 9-bit data. Note: The parity enable bit PEN has no effect after B8EN is enabled.

When data is being transmitted, B8TXD needs to be set before writing data to the transmit register UART_TDR. The B8TXD is transmitted as the MSB of the transmitted data and the value of the UART_TDR. If B8TOG is set, if B8TXD is the same as B8POL, it means that the data is used as an address frame or a synchronization frame. After the transmission is finished, B8TXD will automatically flip. In the next data transmission process, it is not necessary to set B8TXD to the inactive level.

When data is received, the most significant bit of the received data can be read from register bit B8RXD. If the received B8RXD is the same as B8POL, the RXB8_INTF bit in the Interrupt Status Register UART_ISR will be set.

20.3.5 Multiprocessor communication

Multiprocessor communication is possible through the UART (connecting several UARTs to a single network). For example, a UART device can be the master, its TX output is connected to the RX input of the other UART slave devices; the UART slaves are logically coupled together with the respective TX outputs and connected to the RX input of the master device.

In a multi-processor configuration, we usually want only the addressed receiver to be activated to receive subsequent data, thus reducing the extra UART service overhead caused by the participation of unaddressed receivers.

Devices that are not addressed can have their quiescing enabled in silent mode. In silent mode:

- Any receive status bit will not be set.
- All receive interrupts are disabled.
- The RWU bit in the UART_CCR register is set. The RWU can be automatically controlled by hardware or written by software under certain conditions.

Depending on the WAKE bit status in the UART_CCR register, the UART can enter or exit the Silent mode in two ways.

- If the WAKE bit is reset: an idle bus detect is made.
- If the WAKE bit is set: Address mark detection is performed.

Idle bus detection (WAKE=0)

When the RWU bit is written 1, the UART enters silent mode. When an idle frame is detected, it is woken up. The RWU is then cleared by hardware and the interrupt status flag RX_INTF is not set. RWU can also be written to 0 by software.

Address mark detection (WAKE=1)

In this mode, if the MSB is B8POL, the byte is considered an address, otherwise it is considered data. In an address byte, the address of the target receiver is compared to its own address, and the address and mask bits of the receiver are programmed in the UART_RXADDR and UART_RXMASK registers.

If the received byte does not match its programmed address, the UART enters silent mode. At this point, the hardware sets the RWU bit. Receiving this byte will neither set the interrupt status flag RX_INTF nor generate an interrupt or issue a DMA request because the UART is already in silent mode.

When the received byte matches the in-receiver programming address, the UART exits the silent mode. The RWU bit is then cleared and the subsequent bytes are normally received. The interrupt status flag RX_INTF is set when this matched address byte is received because the RWU bit has been cleared.

20.3.6 Single-line half-duplex communication

Single-line half-sided mode is selected by setting the HDSEL bit in the UART_SCR register. In this mode, the SCEN bit of the UART_SCR register must be kept clear.

The UART can be configured to follow a single-wire half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are interconnected inside the chip. Half-duplex and full-duplex communication is selected using the control bit ”HALF DUPLEX SEL” (HDSEL bit in UART_SCR).

When HDSEL is 1

- RX is no longer used
- When there is no data transmission, TX is always released. Therefore, it appears as a standard I/O port in the idle state or the receiving state. This means that the I/O must be configured as a floating input (or an open drain output high) when not being driven by the UART.

In addition, communication is similar to the normal UART mode. Software conflicts are managed online (for example by using a central arbiter). In particular, the transmission is never blocked by hardware. When the TXEN bit is set, the transmission continues as soon as the data is written to the data register.

20.3.7 smart card

Set the SCEN bit in the UART_SCR register to select the smart card mode.

The interface complies with the ISO7816-3 standard and supports the smart card asynchronous protocol. The UART should be set to:

- 8-bit data bit plus parity bit: CHAR=11, PEN=1 in the UART_CCR register at this time
- 1.5 stop bits for transmission and reception: SPB1=1, SPB0=1 of the UART_CCR register

The example given below illustrates the signal on the data line in both the verify error and the no parity error.

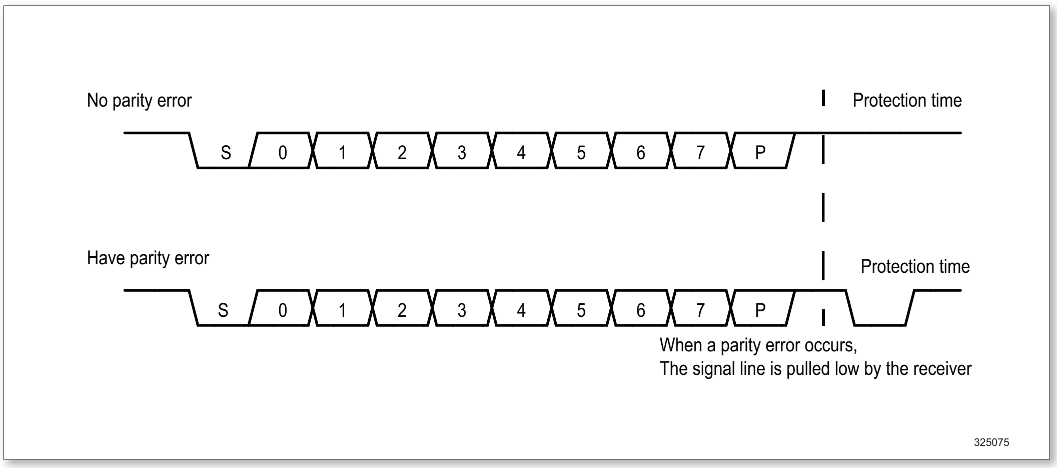


Figure 232. UART block diagram

When connected to a smart card, the TX of the UART drives a bidirectional line of a smart card. In order to do this, the RX must be connected to the same I/O port as the TX. During the transmit start bit and data byte, the transmitter's output enable bit, TXEN, is set and released during the transmit stop bit (weak pull-up), so the receiver can assert the data line if a verify error is found Pull down. If TXEN is not used, TX is pulled high during the stop bit: in this case, the receiver can drive this line as long as the TX is configured to open drain.

Smart card is a single-line half-duplex communication protocol

- The data is sent out from the transmit shift register and is delayed by a minimum of 1/2 baud clock. In normal operation, a full transmit shift register will shift data out at the next baud clock edge. In smart card mode, this transmission is delayed by 1/2 baud clock.
- If a parity error is detected during reception of a data frame set to 0.5 or 1.5 stop bits, the transmission line is pulled low for one baud clock cycle after the completion of reception of the frame (ie, at the end of the stop bit). This is to tell the smart card that the data sent to the UART has not been received correctly. This NACK signal (lower the transmission line for one baud clock period) will generate a framing error at the transmitting end (the transmitting end is configured as 1.5 stop bits). The application can resend the data according to the protocol. If the NACK control bit is set, the receiver will give a NACK signal when a check error occurs; otherwise, no NACK will be sent.
- The setting of the TXC flag can be delayed by programming the protection time register. In normal operation, TXC is asserted when the transmit shift register becomes empty and no new transmit request occurs. In smart card mode, an empty transmit shift register will trigger the guard time counter to start counting up until the value in the guard time register. The TXC was forced to pull low during this time. When the guard time counter reaches the value in the guard time register, TXC is set high.
- The revocation of the flag is not affected by the smart card mode.
- If the transmitter detects a framing error (receives the receiver's NACK signal), the transmitter's receive function module does not detect the NACK as a start bit. According to the ISO protocol, the duration of the received NACK can be 1 or 2 baud clock cycles.
- On the receiver side, if a check error is detected and a NACK is sent, the receiver will

not detect the NACK as the start bit.

Note: 1. The disconnect symbol has no meaning in smart card mode. A 00h data with a frame error will be treated as data instead of a broken symbol.

2. When the TXEN bit is toggled back and forth, no IDLE frame is sent. The ISO protocol does not define an IDLE frame.

The figure below details how the UART samples the NACK signal. In this example, the UART is transmitting data and is configured with 1.5 stop bits. In order to check the integrity of the data and the NACK signal, the receive function block of the UART is activated.

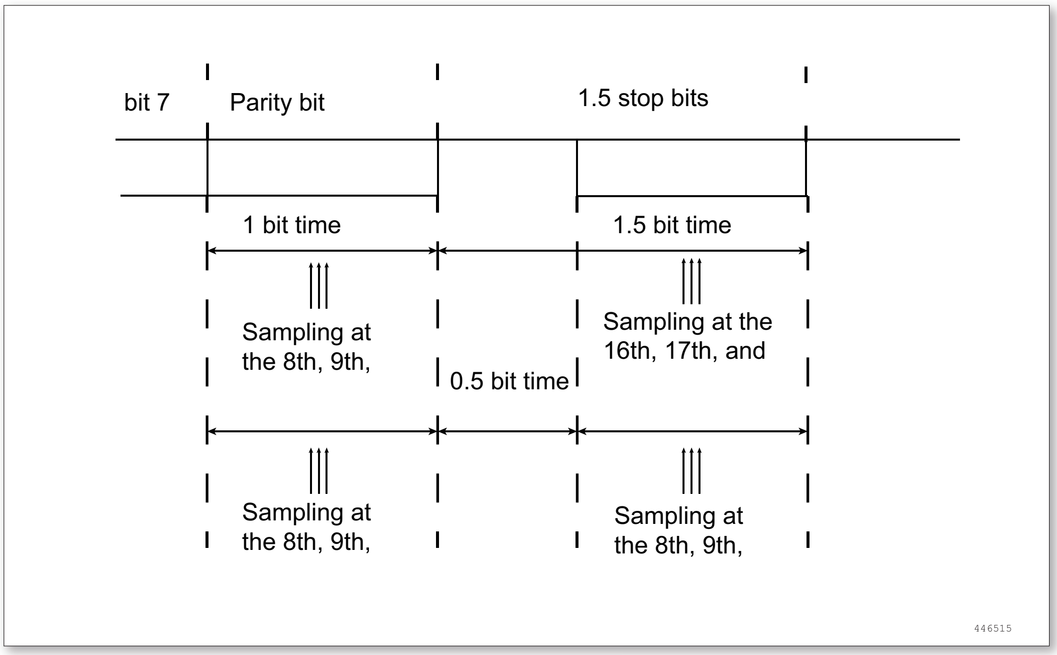

bit 7
Parity bit
1.5 stop bits
1 bit time
1.5 bit time
Sampling at the 8th, 9th,
0.5 bit time
Sampling at the 16th, 17th, and
Sampling at the 8th, 9th,
Sampling at the 8th, 9th,
446515

Figure 233. UART block diagram

20.3.8 Fractional baud rate generator

Set the BRR and FRA registers to set the corresponding baud rate. Refer to the following formula:

$$f_{baudrate} = \frac{f_{PCLK}}{16 \times UARTDIV}$$

$$UARTDIV = BRR + \frac{FRA}{16}$$

Get:

$$f_{baudrate} = \frac{f_{PCLK}}{16 \times BRR + FRA}$$

The BRR register has a minimum value of 4.

20.3.9 Sampling

Since no separate clock is provided for asynchronous operation, the receiver requires synchronization to the receiver. In order to obtain the correct character data on the receive pin 'RX', the UART is configured with a detection circuit. The UART samples the RX pin through a 'bclk16' clock with a 16-times data baud rate. Each data has 16 clock samples, and the sampled values of the 7th, 8th, and 9th falling edges are taken.

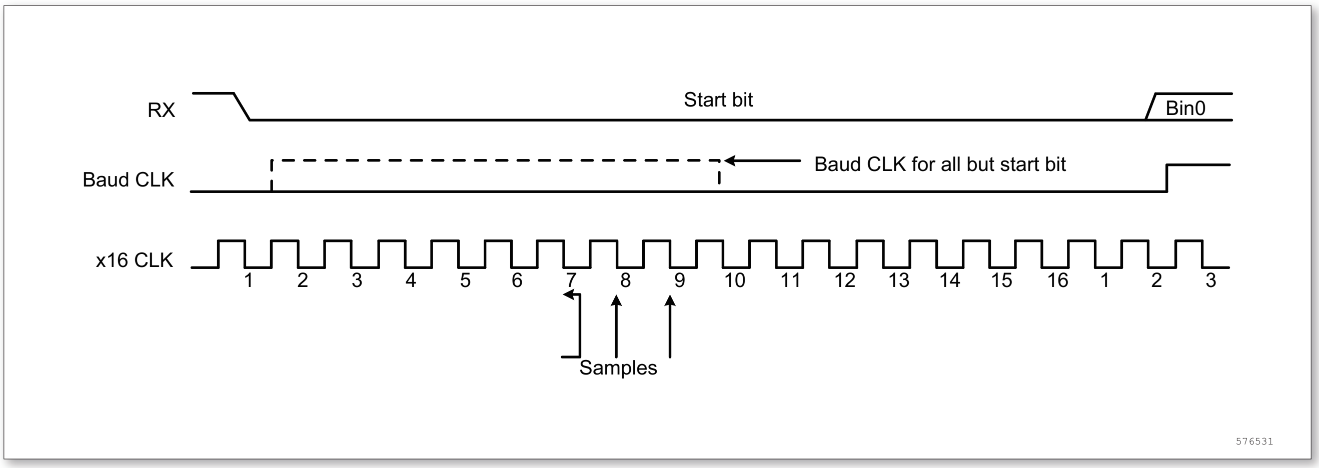


Figure 234. RX Pin Sampling Plan

20.3.10 Parity control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PEN bit in the UART_CCR register. The invalid data will not be transferred to UART_RDR register from the shift register in case of parity error.

Even parity: the parity bit is calculated to obtain an even number of “1s” inside the frame and the parity bit.

Example: data=00110101; 4 bits set => parity bit will be 0 if even parity is selected (PSEL bit in UART_CCR = 1).

Odd parity: the parity bit is calculated to obtain an odd number of “1s” inside the frame and the parity bit.

Example: data=00110101; 4 bits set => parity bit will be 1 if odd parity is selected (PSEL bit in UART_CCR = 0).

Transmission mode: If the PEN bit is set in UART_CCR, then the MSB bit of the data written in the data register is transmitted but is changed by the parity bit (even number of '1s' if even parity is selected or an odd number of '1s' if odd parity is selected). If the parity check fails, the RXPERR_INTF flag is set in the UART_ISR register and an interrupt is generated if RXPERREN is preset.

20.3.11 Hardware flow control

It is possible to control the serial data flow between two devices by using the nCTS input and the nRTS output. The following figure shows how to connect two devices in this mode.

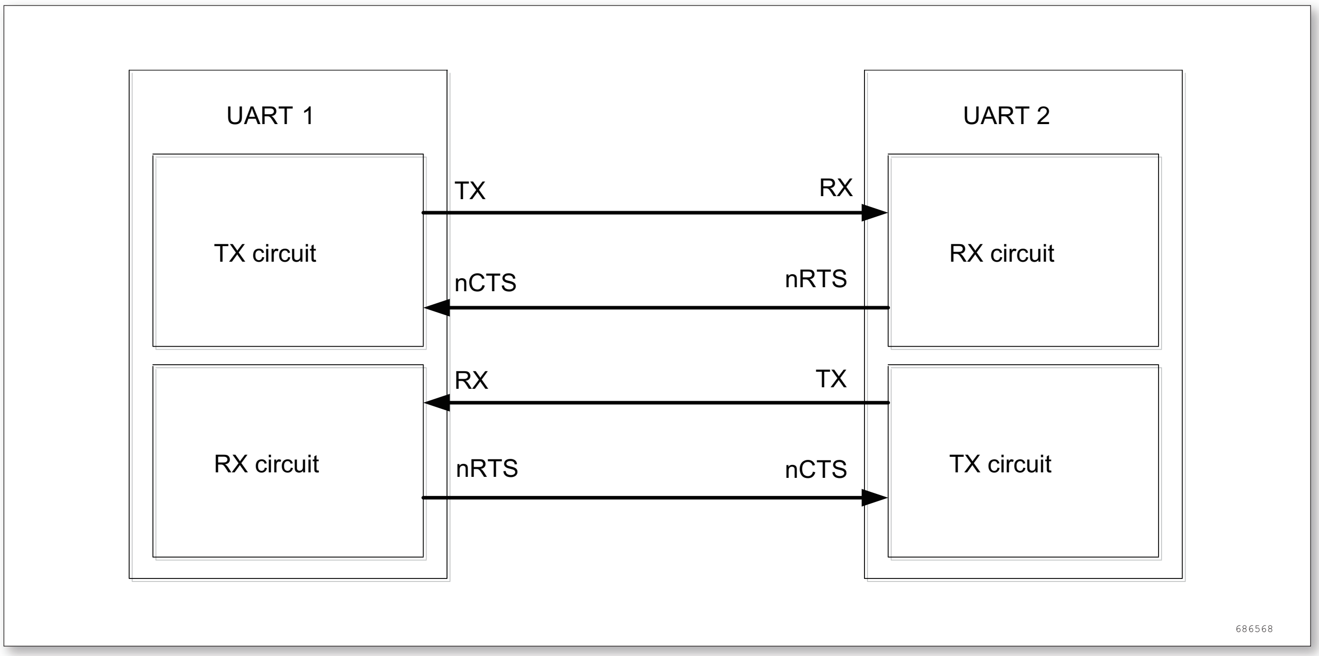


Figure 235. Hardware Flow Control Between Two UARTs

RTS and CTS flow control can be enabled by setting the AUTOFLOWEN bit in the UART_GCR.

RTS flow control

If the RTS flow control is enabled, then nRTS is active (tied low) as long as the UART receiver is ready to receive new data. When the receive register receives data, nRTS is released, indicating that the transmission is expected to stop at the end of the current frame. The following figure shows an example of communication with RTS flow control enabled.

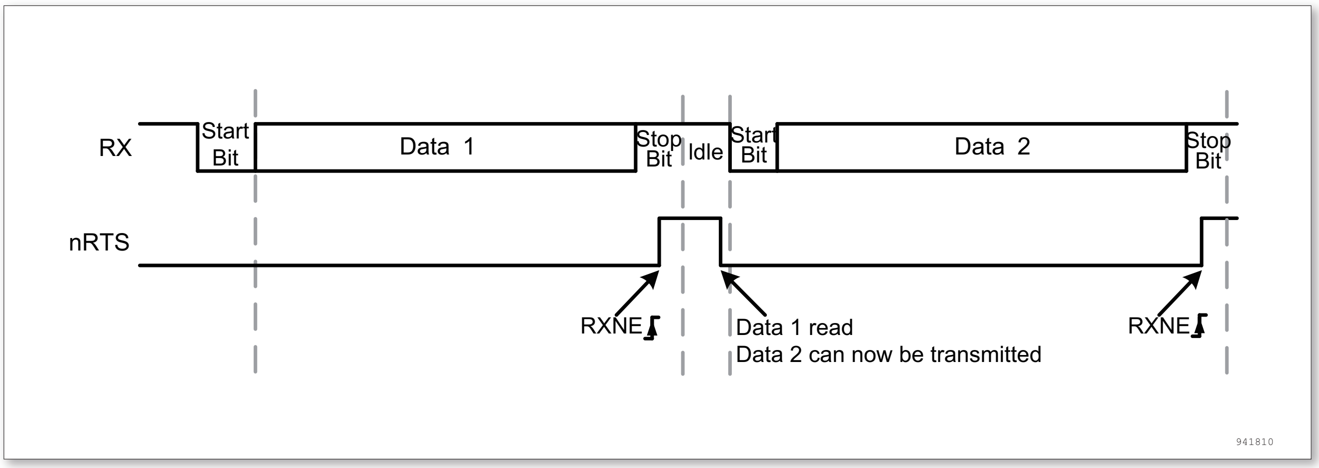


Figure 236. RTS Flow Control

CTS flow control

If the CTS flow control is enabled, then the transmitter checks the nCTS input before transmitting the next frame. If nCTS is active (tied low), then the next data is transmitted (assuming that a data is to be transmitted, in other words), else the transmission does not occur. When nCTS is inactive during a transmission, the current transmission is completed before the transmitter stops. The figure below shows an example of communication with

CTS flow control enabled.

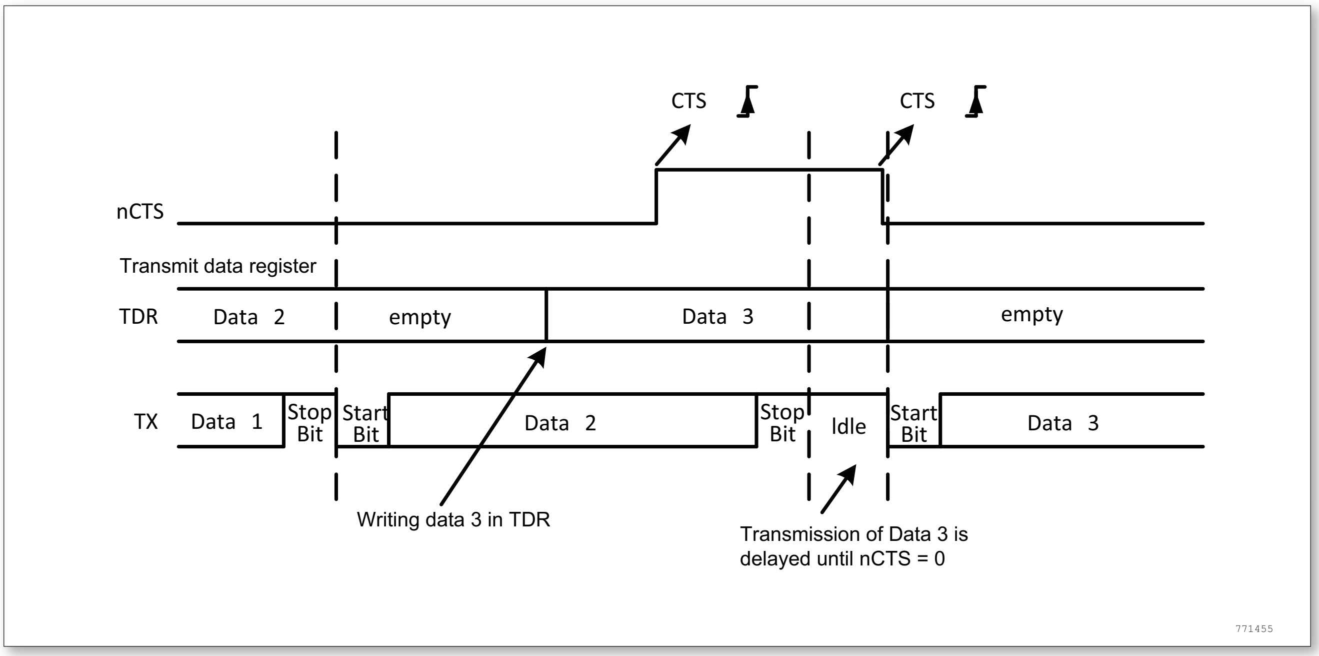


Figure 237. CTS Flow Control

20.3.12 Communication using DMA

The UART is capable of communicating using the DMA.

Transmission using DMA

During transmission using DMA, first configure the address of the UART_TDR register as the destination address of the DMA transfer in the DMA control register, configure the memory address as the source address of the DMA transfer, and configure the data volume. Then, enable DMA mode by setting the DMAMODE bit in the UART_GCR register. When the TXEN bit is set to ’1’, the DMA transfers data from the specified SRAM zone to the UART_TDR register.

Reception using DMA

During reception using DMA, first configure the address of the UART_RDR register as the source address of the DMA transfer in the DMA control register, configure the memory address as the destination address of the DMA transfer, and configure the data volume. Then, enable DMA mode by setting the DMAMODE bit in the UART_GCR register. When the RXEN bit is enabled, the DMA transfers data from the specified SRAM zone to the UART_RDR register.

20.4 UART interrupt requests

Table 64. UART interrupt requests

Interrupt event	Interrupt status	Enable bit
Transmit buffer null	TX_INTF	TXIEN
Valid data received	RX_INTF	RXIEN

Interrupt event	Interrupt status	Enable bit
Transmit shift register completed	TXC_INTF	TXC_EN
Receive overrun error	RXOERR_INTF	RXOERREN
Parity error	RXPERR_INTF	RXPERREN
Frame error	RXFERR_INTF	RXFERREN
Break frame	RXBRK_INTF	RXBRKEN
Send disconnect frame	TXBRK_INTF	TXBRK_EN
Receiving sync frame	RXB8_INTF	RXB8_EN

These events generate an interrupt if the corresponding Enable Control Bit is set.

20.5 UART registers

Table 65. UART Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	UART_TDR	UART transmit data register	0x00000000	section 20.5.1
0x04	UART_RDR	UART receive data register	0x00000000	section 20.5.2
0x08	UART_CSR	UART current status register	0x00000009	section 20.5.3
0x0C	UART_ISR	UART interrupt status register	0x00000000	section 20.5.4
0x10	UART_IER	UART interrupt enable register	0x00000000	section 20.5.5
0x14	UART_ICR	UART interrupt clear register	0x00000000	section 20.5.6
0x18	UART_GCR	UART global control register	0x00000000	section 20.5.7
0x1C	UART_CCR	UART general control register	0x00000030	section 20.5.8
0x20	UART_BRR	UART baud rate register	0x00000001	section 20.5.9
0x24	UART_FRA	UART fractional baud rate register	0x00000000	section 20.5.10
0x28	UART_RXADDR	UART receive address register	0x00000000	section 20.5.11
0x2C	UART_RXMASK	UART receive mask register	0x000000FF	section 20.5.12
0x30	UART_SCR	UART SCR register	0x00000000	section 20.5.13

20.5.1 UART transmit data register(UART_TDR)

Offset address: 0x00

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								TXREG							
								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Always read as 0.
7 : 0	TXREG	rw	0x00	Transmit data register

20.5.2 UART receive data register(UART_RDR)

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								RXREG							
								r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Always read as 0.
7 : 0	RXREG	r	0x00	Receive data register This register is read-only.

20.5.3 UART current status register(UART_CSR)

Offset address: 0x08

Reset value: 0x0000 0009

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												TXBUF_ EMPTY	TXFULL	RXAVL	TXC
												r	r	r	r

Bit	Field	Type	Reset	Description
31 : 4	Reserved			Always read as 0.
3	TXBUF_ EMPTY	r	0x01	Transmit buffer empty flag bit 1 : Transmit buffer null 0 : Transmit buffer non-null
2	TXFULL	r	0x00	Transmit buffer full flag bit 1 : Transmit buffer full 0 : Transmit buffer non-null

Bit	Field	Type	Reset	Description
1	RXAVL	r	0x00	Receive valid data flag bit This bit is set when the receive buffer receives data of one full byte. 1 : Receive buffer has received a complete and valid byte of data 0 : Receive buffer null
0	TXC	r	0x01	Transmit complete flag bit 1 : Both the transmit buffer and the transmit shift register are null 0 : The transmit register is non-null

20.5.4 UART interrupt 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
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RXB8 _INTF	TX BRK_ INTF	RX BRK_ INTF	Res.	RXP ERR_ INTF	RXO ERR_ INTF	TXC_ INTF	RX_ INTF	TX_ INTF
							r	r	r		r	r	r	r	r

Bit	Field	Type	Reset	Description
31 : 9	Reserved			Always read as 0.
8	RXB8_ INTF	r	0x00	UART sync frame interrupt flag bit. In the 9-bit communication mode, when the ninth bit of the received data is the same as the register CCR.B8POL, the RXB8_INT position. This bit can be used as an interrupt request signal 1 : Received sync frame 0 : No sync frames received
7	TXBRK_ INTF	r	0x00	The UART disconnect frame transmission completion interrupt flag bit. 1 : Shift register disconnect frame data transmission completed 0 : Shift register is empty or is being shifted Note: Disconnected frames cannot be sent continuously.
6	RXBRK_ INTF	r	0x00	UART receive frame break interrupt flag bit After the abnormal stop bit, the RX pin receives 10 or more low levels for a period of time. 1 : Break frame detected 0 : No break frame detected

Bit	Field	Type	Reset	Description
5	Reserved			Always read as 0.
4	RXPERR_ INTF	r	0x00	Parity error interrupt flag bit 1 : Parity error detected 0 : No parity error detected
3	RXOERR_ INTF	r	0x00	Receive overflow error interrupt flag bit This bit is set only when autoflowen=0. 1 : Receive overrun error 0 : No overrun error
2	TXC_INTF	r	0x00	The UART Transmit Shift Register completes the interrupt flag bit. 1 : Shift register data transmission completed 0 : Shift register is empty or is being shifted Note: This flag is related to the protection time.
1	RX_INTF	r	0x00	Receive valid data interrupt flag bit This bit is set when the receive buffer receives data of one full byte. 1 : Receive buffer receives valid byte data 0 : Receive buffer null
0	TX_INTF	r	0x00	Transmit buffer empty interrupt flag bit 1 : Transmit buffer null 0 : Transmit buffer non-null

20.5.5 UART interrupt enable register(UART_IER)

Offset address: 0x10

Reset value: 0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RXB8 _IEN	TXBRK _IEN	RX BRK EN	Res.	RXP ERR EN	RXO ERR EN	TXC_ IEN	RXIEN	TXIEN
							rw	rw	rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 9	Reserved			Always read as 0.
8	RXB8_IEN	rw	0x00	The UART sync frame interrupt enable control bit. 1 : Enable receive sync frame interrupt 0 : Do not receive sync frame interrupts

Bit	Field	Type	Reset	Description
7	TXBRK_IEN	rw	0x00	The UART break frame transmission complete interrupt enable control bit. 1：Enable send break frame completion interrupt 0：Do not send disconnected frame completion interrupt
6	RXBRKEN	rw	0x00	Receive frame break interrupt enable bit 1：Interrupt enabled 0：Interrupt disabled
4	RXPERREN	rw	0x00	Parity error interrupt enable bit 1：Interrupt enabled 0：Interrupt disabled
3	RXOERREN	rw	0x00	Receive overflow error interrupt enable bit 1：Interrupt enabled 0：Interrupt disabled
2	TXC_IEN	rw	0x00	The UART Transmit Shift Register completes the Interrupt Enable Control bit. 1：Shift register data transmission completed 0：Shift register is empty or is being shifted
1	RXIEN	rw	0x00	Receive buffer interrupt enable bit 1：Interrupt enabled 0：Interrupt disabled
0	TXIEN	rw	0x00	Transmit buffer empty interrupt enable bit 1：Interrupt enabled 0：Interrupt disabled

20.5.6 UART interrupt clear register(UART_ICR)

Offset address：0x14

Reset value：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RXB8_CLR	TXBRK_CLR	RX BRK CLR	Res.	RXP ERR CLR	RXO ERR CLR	TXC_CLR	RXICLR	TXICLR
							w	w	w		w	w	w	w	w

Bit	Field	Type	Reset	Description
31：9	Reserved			Always read as 0.
8	RXB8_CLR	w	0x00	The UART sync frame interrupt flag clear control bit. 1：Clear receive sync frame interrupt flag 0：No action

Bit	Field	Type	Reset	Description
7	TXBRK_CLR	w	0x00	The UART disconnect frame transmission completion interrupt flag clear control bit. 1 : Clear disconnect frame send completion interrupt flag 0 : No action
6	RXBRKCLR	w	0x00	Receive frame break interrupt clear bit 1 : Interrupt cleared 0 : Interrupt not cleared
4	RXPERRCLR	w	0x00	Parity error interrupt clear bit 1 : Interrupt cleared 0 : Interrupt not cleared
3	RXOERRCLR	w	0x00	Receive overflow error interrupt clear bit 1 : Interrupt cleared 0 : Interrupt not cleared
2	TXC_CLR	w	0x00	The UART Transmit Shift Register completes the Interrupt Enable Control bit. 1 : Clear shift register data transmission completion interrupt flag 0 : No action
1	RXICLR	w	0x00	Receive interrupt clear bit 1 : Interrupt cleared 0 : Interrupt not cleared
0	TXICLR	w	0x00	Transmit buffer empty interrupt clear bit 1 : Interrupt cleared 0 : Interrupt not cleared

20.5.7 UART global control register(UART_GCR)

Offset address: 0x18

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											TXEN	RXEN	AUTO FLOW EN	DMA MODE	UARTEN
											rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 5	Reserved			Always read as 0.
4	TXEN	rw	0x00	Enable transmit 1 : Transmission enabled 0 : Transmission disabled and TX buffer cleared

Bit	Field	Type	Reset	Description
3	RXEN	rw	0x00	Enable receive 1 : Reception enabled 0 : Reception disabled and RX buffer cleared
2	AUTO FLOWEN	rw	0x00	Automatic flow control enable bit 1 : Automatic flow control enabled 0 : Automatic flow control disabled
1	DMAMODE	rw	0x00	DMA mode selection bit 1 : Select DMA mode 0 : Select the normal mode
0	UARTEN	rw	0x00	UART mode selection bit 1 : UART module enabled 0 : UART mode disabled

20.5.8 UART general control register(UART_CCR)

Offset address: 0x1C

Reset value: 0x0000 0030

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		WAKE	RWU	B8EN	B8TOG	B8POL	B8TXD	B8RXD	SPB1	CHAR		BRK	SPB0	PSEL	PEN
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 14	Reserved			Always read as 0.
13	WAKE	rw	0x00	Wake up method. This bit determines the method of waking up the UART. 1 : Address mark wake up 0 : Idle bus wake up Note: The UART has already received a data byte before placing the UART in silent mode. Otherwise, in silent mode, it cannot be woken up by idle bus detection.
12	RWU	rw	0x00	Receive wake up. This bit is used to determine if the UART is placed in silent mode. This bit can be set or cleared by software. When the wake-up sequence arrives, the hardware also automatically clears it. 1 : Receiver is in silent mode 0 : The receiver is in normal working mode When the address mark wakes up, if the receive buffer is not empty, it cannot be modified by software.

Bit	Field	Type	Reset	Description
11	B8EN	rw	0x00	The ninth bit enable control bit of the UART sync frame. After this bit is enabled, the verify enable bit PEN has no effect. 1 : Enable ninth bit transmission of sync frame 0 : Disable synchronization frame ninth transmission
10	B8TOG	rw	0x00	The UART sync frame sends the ninth auto flip control bit. 1 : Enable ninth automatic flip 0 : Prohibit the ninth automatic flip Note: When the values of B8TXD and B8POL are the same, the second data transmitted after the register is configured starts to flip. The first data defaults to the address bit.
9	B8POL	rw	0x00	The ninth bit polarity control bit of the UART sync frame. 1 : The ninth bit of the sync frame is active high. 0 : The ninth bit of the sync frame is active low.
8	B8TXD	rw	0x00	The UART sync frame sends the ninth bit of data. 1 : The ninth bit of the transmission sync frame is high 0 : The ninth bit of the transmission sync frame is low
7	B8RXD	rw	0x00	The UART sync frame receives the ninth bit of data. Read only. 1 : The ninth bit of the receive sync frame is high 0 : The ninth bit of the receive sync frame is low
6	SPB1	rw	0x00	The stop bit selection bit is combined with SPB0 to set the stop bit number.
5 : 4	CHAR	rw	0x03	UART width bit 00: 5bits 01: 6bits 10: 7bits 11: 8bits
3	BRK	rw	0x00	UART transmit frame break 1 : Serial forced output logic '0' (break frame) 0 : Break disabled
2	SPB0	rw	0x00	Stop bit selection Set the transmit stop bits. The receiver usually detects a stop bit. SPB1, SPB0 : 00, 1 stop position SPB1, SPB0 : 01, 2 stop bits (5 bit data bit, SPB setting is not used, stop bit is forced to 1 bit) SPB1, SPB0 : 10, 0.5 stop bits SPB1, SPB0 : 11, 1.5 stop bits

Bit	Field	Type	Reset	Description
1	PSEL	rw	0x00	Parity selection bit When the check is enabled, this bit is used to select to use either even or odd parity. 1 : Even parity 0 : Odd parity
0	PEN	rw	0x00	Parity enable bit 1 : Transmit and receive check enabled 0 : Check disabled

20.5.9 UART baud rate register(UART_BRR)

Offset address: 0x20

Reset value: 0x0000 0001

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
DIV_Mantissa															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.
15 : 0	DIV_Mantissa	rw	0x0001	The integer part of UARTDIV These 16 bits define the integer part of the UART divider division factor (UARTDIV). DIV_Mantissa Minimum value is 4

20.5.10 UART fractional baud rate register(UART_FRA)

Offset address: 0x24

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 4	Reserved			Always read as 0.
3 : 0	DIV_Fraction	rw	0x00	The decimal part of UARTDIV These 4 bits define the decimal part of the UART divider division factor (UARTDIV).

20.5.11 UART receive address register(UART_RXADDR)

Offset address: 0x28

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								RXADDR							
								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Always read as 0.
7 : 0	RXADDR	rw	0x00	UART sync frame data native match address. If RXMASK = 0xFF, RXB8_INTF is generated when the received sync frame data is the same as the local match address. Address 0 is the broadcast address and will respond when received.

20.5.12 UART receive mask register(UART_RXMASK)

Offset address: 0x2C

Reset value: 0x0000 00FF

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								RXMASK							
								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 8	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
7 : 0	RXMASK	rw	0xFF	When the data bits are all ”0”, a sync frame interrupt request is generated when any data is received. If the data bit is ”1” and the corresponding bits of RDR and RXADDR match, a synchronous frame interrupt request is generated.

20.5.13 UART SCR register(UART_SCR)

Offset address: 0x30

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			HDSEL	SCFCNT								Res.	NACK	SCAEN	SCEN
			rw	rw	rw	rw	rw	rw	rw	rw	rw		r	rw	rw

Bit	Field	Type	Reset	Description
31 : 13	Reserved			Always read as 0.
12	HDSEL	rw	0x00	Single-line half-duplex mode selection. 1 : Enable half-duplex mode 0 : Half-duplex mode
11 : 4	SCFCNT	rw	0x00	ISO7816 protection counter. When the transmit data is low during the protection counter period, the start bit of the next data is disabled. 0 is 16 baud rate counting time, 15..1 is 15..1 time
3	Reserved			Always read as 0.
2	NACK	r	0x00	Master receive frame acknowledge bit
1	SCAEN	rw	0x00	ISO7816 verifies the auto answer bit. 1 : Enable auto answer 0 : Disable automatic answer
0	SCEN	rw	0x00	ISO7816 enables control bits. 1 : Enable ISO7816 function 0 : Prohibit ISO7816 function

21 Hardware division(HWDIV)

Hardware division(HWDIV)

21.1 Hardware Division Introduction

Hardware division is useful in some high-performance applications that automatically perform signed or unsigned 32-bit integer division operations.

21.2 Main features of hardware division

- signed or unsigned integer division
- 32-bit divisor and dividend, output 32-bit quotient and remainder
- 8 HCLK cycles completed
- If the divisor is zero, an overflow interrupt flag will be generated.
- write divisor automatically performs division operation
- automatically waits for the end of the operation when reading the quotient and remainder registers, no need to check the status bits

21.3 Hardware division function introduction

The hardware division unit consists of four 32-bit data registers, which are dividend, divisor, quotient and remainder, and can be done with signed or unsigned 32-bit division. The hardware division control register USIGN bit can be selected to be signed division or unsigned division.

Each time the divisor register is written, the divide operation is automatically triggered. After the end of the operation, the result is written to the quotient and remainder registers. If the quotient register, remainder register, or status register is read before the end, the read operation is suspended until the end of the operation.

If the divisor is zero, an overflow interrupt flag will be generated.

21.4 Hardware Division Register description

Table 66. Hardware Division Register Overview

Offset	Acronym	Register Name	Reset	Section
0x00	HWDIV_DVDR	Dividend register	0x00000000	section 21.4.1
0x04	HWDIV_DVSR	Divisor register	0x00000001	section 21.4.2
0x08	HWDIV_QUOTR	Quotient register	0x00000000	section 21.4.3
0x0C	HWDIV_RMDR	Remainder register	0x00000000	section 21.4.4

Offset	Acronym	Register Name	Reset	Section
0x10	HWDIV_SR	HWDIV status register	0x00000000	section 21.4.5
0x14	HWDIV_CR	HWDIV control register	0x00000001	section 21.4.6

21.4.1 Dividend register(HWDIV_DVDR)

Offset address: 0x00

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 0	DIVIDEND	rw	0x0000 0000	Dividend data

21.4.2 Divisor register(HWDIV_DVSR)

Offset address: 0x04

Reset value: 0x0000 0001

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

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

Bit	Field	Type	Reset	Description
31 : 0	DIVISOR	rw	0x0000 0001	Divisor data After the register is written, the division operation is automatically triggered.

21.4.3 Quotient register(HWDIV_QUOTR)

Offset address: 0x08

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 0	QUOTIENT	r	0x0000 0000	Quotient data

21.4.4 Remainder register(HWDIV_RMDR)

Offset address: 0x0C

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 0	REMAINDER	r	0x0000 0000	Remainder data

21.4.5 HWDIV status register(HWDIV_SR)

Offset address: 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.															OVF
															r

Bit	Field	Type	Reset	Description
31: 1	Reserved			Reserved, always read as 0.

Bit	Field	Type	Reset	Description
0	OVF	r	0x00	overflow status flag Automatically clear before the next division operation 1: The current operation divisor is zero. 0: The current operation divisor is not zero.

21.4.6 HWDIV control register(HWDIV_CR)

Offset address: 0x14

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.														OVFE	USIGN
														rw	rw

Bit	Field	Type	Reset	Description
31: 2	Reserved			Reserved, always read as 0.
1	OVFE	rw	0x00	Overflow interrupt enable 1: divide by zero overflow interrupt enable 0: divide by zero overflow interrupt is not enabled
0	USIGN	rw	0x01	Unsigned division enable 1: unsigned division 0: signed division

22 System configuration controller (SYSCFG)

System configuration controller (SYSCFG)

The device is provided with a set of system configuration registers, with the main functions as follows:

- The remapping section covers the TIM16 and TIM17, and UART1 and DMA trigger sources of ADC to other different DMA channels.
- Management device connected to the external interrupts of GPIO port.
- Remapping the memory to the code starting area.
- Pin configuration for external interrupt.

22.1 SYSCFG register description

Table 67. Summary of SYSCFG Register

Offset	Acronym	Register Name	Reset	Section
0x00	SYSCFG_CFGR	SYSCFG configuration register	0x0000000X	section 22.1.1
0x08	SYSCFG_EXTICR1	External interrupt configuration register 1	0x00000000	section 22.1.2
0x0C	SYSCFG_EXTICR2	External interrupt configuration register 2	0x00000000	section 22.1.3
0x10	SYSCFG_EXTICR3	External interrupt configuration register 3	0x00000000	section 22.1.4
0x14	SYSCFG_EXTICR4	External interrupt configuration register 4	0x00000000	section 22.1.5

22.1.1 SYSCFG configuration register (SYSCFG_CFGR)

This register is dedicated to configuring memory start area mapping and DMA request remapping, with two control bits of configurable memory start address (0x0000 0000) storage area type, these two control bits can be configured by software, to mask BOOT selection. After reset, these two control bits represent the actual BOOT mode configuration.

Offset address: 0x00

Reset value: 0x0000 000X(X: the selection control bit of the actual BOOT mode)

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			TIM17 _DMA _RMP	TIM16 _DMA _RMP	UART1 _RX _DMA _RMP	UART1 _TX _DMA _RMP	ADC _DMA _RMP	Reserved						MEM_ MODE	
			rw	rw	rw	rw	rw							rw	rw

Bit	Field	Type	Reset	Description
31 : 13	Reserved			Always read as 0.
12	TIM17_DMA _RMP	rw	0x00	TIM17 DMA request remapping bit This bit is set and cleared by the software, controlling the remapping of TIM17 DMA channel requests. 0: No remapping(TIM17_CH1 and TIM17_UP DMA request mapped on DMA Channel 1) 1: Remapping (TIM17_CH1 and TIM17_UP DMA request mapped on DMA Channel 2)
11	TIM16_DMA _RMP	rw	0x00	TIM16 DMA request remapping bit This bit is set and cleared by the software.controlling the remapping of TIM16 DMA channel requests. 0: No remapping(TIM16_CH1 and TIM16_UP DMA request mapped on DMA Channel 3) 1: Remapping (TIM16_CH1 and TIM16_UP DMA request mapped on DMA Channel 4)
10	UART1_RX _DMA_RMP	rw	0x00	UART1_RX DMA request remapping bit This bit is set and cleared by the software, controlling the remapping of UART1_RX DMA channel requests. . 0: No remapping(UART1_ RX DMA request mapped on DMA Channel 3) 1: Remapping (UART1_ RX DMA request mapped on DMA Channel 5)
9	UART1_TX _DMA_RMP	rw	0x00	UART1_TX DMA request remapping bit This bit is set and cleared by the software, controlling the remapping of UART1_TX DMA channel requests. 0: No remapping(UART1_TX DMA request mapped on DMA Channel 2) 1: Remapping (UART1_TX DMA request mapped on DMA Channel 4)

Bit	Field	Type	Reset	Description
8	ADC_DMA _RMP	rw	0x00	ADC DMA request remapping bit This bit is set and cleared by the software, controlling the remapping of ADC DMA channel requests. 0: No remapping(ADC DMA request mapped on DMA Channel 1) 1: Remapping (ADC DMA request mapped on DMA Channel 2)
7 : 2	Reserved			Always read as 0.
1 : 0	MEM_MODE	rw	0x00	Memory selection bit These bits are set and cleared by the software, controlling the internal mapping of the memory to address 0x0000 0000. When being reset, these bit values are determined by the BOOT0 pin configuration value and the nBOOT1 bit value. x0: main flash memory mapped to 0x0000 0000 01: System flash mapped to 0x0000 0000 11: Embedded RAM mapped to 0x0000 0000

22.1.2 External interrupt configuration register 1 (SYSCFG_EXTICR1)

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
EXTI3				EXTI2				EXTI1				EXTI0			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.
15 : 0	EXTIx	rw	0x00	EXTI x configuration(x = 0···3) These bits are available for software to read and write, and used for selecting the input sources of EXTIx external interrupts. 0000: PA[x] pin 0001: PB[x] pin 0010: PC[x] pin 0011: PD[x] pin

22.1.3 External interrupt configuration register 2 (SYSCFG_EXTICR2)

Offset address: 0x0C

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
EXTI7				EXTI6				EXTI5				EXTI4			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.
15 : 0	EXTIx	rw	0x00	EXTI x configuration(x = 4···7) These bits are available for software to read and write, and used for selecting the input sources of EXTIx external interrupts. 0000: PA[x] pin 0001: PB[x] pin 0010: PC[x] pin 0011: PD[x] pin

22.1.4 External interrupt configuration register 3 (SYSCFG_EXTICR3)

Offset address: 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
EXTI11				EXTI10				EXTI9				EXTI8			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.

Bit	Field	Type	Reset	Description
15 : 0	EXTIx	rw	0x00	EXTI x configuration(x = 8…11) These bits are available for software to read and write, and used for selecting the input sources of EXTIx external interrupts. 0000: PA[x] pin 0001: PB[x] pin 0010: PC[x] pin 0011: PD[x] pin

22.1.5 External interrupt configuration register 4 (SYSCFG_EXTICR4)

Offset address: 0x14

Reset value: 0x0000 0000

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

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

Bit	Field	Type	Reset	Description
31 : 16	Reserved			Always read as 0.
15 : 0	EXTIx	rw	0x00	EXTI x configuration(x = 12…15) These bits are available for software to read and write, and used for selecting the input sources of EXTIx external interrupts. 0000: PA[x] pin 0001: PB[x] pin 0010: PC[x] pin 0011: PD[x] pin

23 Device electronic signature (Device)

Device electronic signature (Device)

The electronic signature is stored in the System memory area in the Flash memory module, and can be read using the JTAG/SWD or the CPU. It contains factory-programmed identification data that allow the user firmware or other external devices to automatically match microcontrollers with different configurations.

23.1 Memory size registers

23.1.1 Unique device ID register (96 bits)

The unique device identifier is ideally suited:

- for use as serial numbers (for example USB string serial numbers or other end applications).
- 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 cryptographic primitives and protocols before programming the internal Flash memory.
- to activate secure boot processes, etc.

The 96-bit unique device identifier provides a reference number which is unique for any device 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) /full words (32 bits) in different ways and then be concatenated using a custom algorithm.

23.2 UID register description

Table 68. Memory Capacity Register Description Overview

Offset	Acronym	Register Name	Reset	Section
0x00	UID1	Unique identification code	0xXXXXXXXX	section 23.2.1
0x02	UID2	Unique identification code	0xXXXXXXXX	section 23.2.2
0x04	UID3	Unique identification code	0xXXXXXXXX	section 23.2.3
0x08	UID4	Unique identification code	0xXXXXXXXX	section 23.2.4

23.2.1 UID1

Base address: 0x1FFF F7E8

Address offset: 0x00

Read-only, the value is factory-programmed

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

Bit	Field	Type	Reset	Description
15 :0	U_ID	r		U_ID: 15: 0 unique ID bits This field value is also reserved for a future feature.

23.2.2 UID2

Address offset: 0x02

Read-only, the value is factory-programmed

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

Bit	Field	Type	Reset	Description
15 :0	U_ID	r		U_ID: 31: 16 unique ID bits This field value is also reserved for a future feature.

23.2.3 UID3

Address offset: 0x04

Read-only, the value is factory-programmed

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

Bit	Field	Type	Reset	Description
31 :0	U_ID	r		U_ID: 63: 32 unique ID bits This field value is also reserved for a future feature.

23.2.4 UID4

Address offset: 0x08

Read-only, the value is factory-programmed

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

Bit	Field	Type	Reset	Description
31：0	U_ID	r		U_ID: 95: 64 unique ID bits This field value is also reserved for a future feature.

24 Debug support(DBG)

Debug support(DBG)

24.1 Overview

The core of the series contains hardware debugging modules for complex debugging operations. The hardware debugging modules allow the core to be stopped either on a given instruction fetch (breakpoint) or data access (watchpoint).

When stopped, the core's internal state and the system's external state may be examined. Once examination is completed, the core and the system may be restored and program execution resumed.

The hardware debugging module is used by the debugger for relevant operations when it is connected to and used for debugging the microcontroller of the series.

Support:

- Serial debug interface

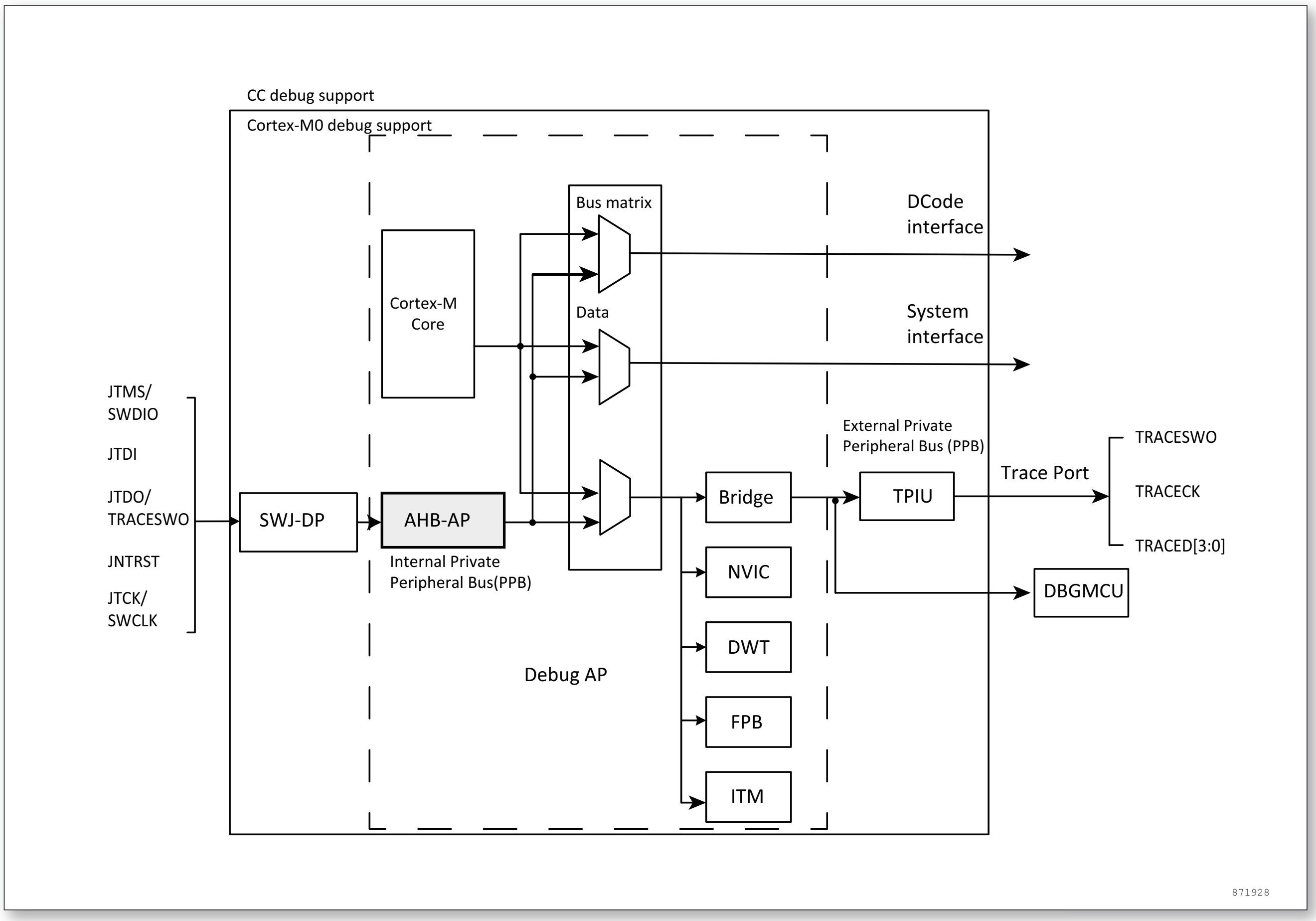


Figure 238. Block Diagram of MM32 Series Level and CPU Level Debug Support

The core provides integrated on-device debug support. It is comprised of:

- SW-DP: : serial debug port
- AHP-AP: AHB access port
- ITM: Instrumentation trace macrocell
- FPB: Flash patch breakpoint
- DWT: Data watchpoint trigger
- TPUI: Trace port unit interface

24.2 Pinout and debug port pins

The device microcontroller is available in various packages with different numbers of available pins. As a result, some functionality related to pin availability may differ between packages.

24.2.1 SWD debug port pins

2 ordinary I/O ports of the device are used as the SW-DP interface pins. These pins are available on all packages.

Table 69. SWJ Debug Port Pins

SWJ-DP pin name	SW debug interface		Pin assignment
	Type	Debugging function	
SW SW debug interface	Input/output	Serial data input/output	PA13
SWCLK	Input	Serial clock	PA14

24.2.2 Internal pull-up and pull-down on SWD pins

It is necessary to ensure that the SWD input pins are not floating since they are directly connected to D flip-flops to control the debug mode features. Special care must be taken with the SWCLK pin which is directly connected to the clock of some of these flip-flops.

To avoid any uncontrolled I/O levels, the device embeds internal pull-ups and pull-downs on the SWD input pins:

- SWDIO: Internal pull-up
- SWCLK: Input with pull-down

The software can use these I/Os as ordinary I/Os.

24.3 ID codes and locking mechanism

There are several ID codes inside the device.

24.3.1 MCU device ID code

The MCU integrates an MCU ID code. This ID identifies the MCU part number and the die revision. It is part of the DBG_MCU component and is mapped on the external APB bus. This code is accessible using the SW debug port (2 pins) or by the user code.

DBGMCU_IDCODE

Address: 0x40013400 Only 32-bits access supported.

Read-only =0xXXXXXXXX, where X is a bit with undefined content.

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

31: 0	DEV_ID: Device identifier

24.3.2 Cortex JEDEC-106 ID code

The CPU has a JEDEC-106 ID code. It is located in a 4KB ROM table mapped to the internal APB bus address 0xE00FF000_ 0xE00FFFFF.

The following table shows the individual ID codes for the series:

Table 71. ID code

ID name	chip
DEV_ID	0xCC4460B1
CPU TAP SW ID	0x0BB11477

24.4 SW debug port

24.4.1 SW protocol introduction

This synchronous serial protocol uses two pins:

- SWCLK: clock from host to target
- SWDIO: bidirectional

The protocol allows two banks of registers (DPACC registers and APACC registers) to be read and written to. Bits are transferred LSB-first on the wire. For SWDIO bidirectional management, the pin must be pulled-up on the board (100 KΩ recommended). Each time the direction of SWDIO changes in the protocol, a turnaround time is inserted where the line is not driven by the host nor the target. By default, this turnaround time is one bit time, however, this can be adjusted by configuring the SWCLK frequency.

24.4.2 SW protocol sequence

Each sequence consist of three phases:

- Packet request (8 bits) transmitted by the host
- Acknowledge response (3 bits) transmitted by the target

- Data transfer phase (33 bits) transmitted by the host or the target

Table 72. Packet Request (8-bit)

Bit	Name	Description
0	Start	Must be ’1’
1	APnDP	0: DP Access 1: AP Access
2	RnW	0: Write Request 1: Read Request
4:3	A(3: 2)	Address field of the DP or AP registers
5	Parity	Single bit parity of preceding bits
6	Stop	0
7	Park	Not driven by the host. Must be read as ’1’ by the target because of the pull-up

Refer to the CPU TRM (Technical Reference Manual) for a detailed description of DPACC and APACC registers.

The packet request is always followed by the turnaround time (default 1 bit) where neither the host nor target drives the line.

Table 73. Packet Request (3-bit)

Bit	Name	Description
0..2	ACK	001: Fault 010: Wait 100: OK

The ACK must be followed by a turnaround time only if it is a READ transaction or if a WAIT or FAULT acknowledge has been received.

Table 74. Packet Request (33-bit)

Bit	Name	Description
0..31	WDATA/RDATA	Write or read data
32	Parity	Single parity of the 32 data bits

The DATA transfer must be followed by a turnaround time only if it is a READ transaction.

24.4.3 SW-DP state machine (Reset, idle states, ID code)

The state machine of the SW-DP has an internal ID code which identifies the SW-DP. It follows the JEP-106 standard.

The SW-DP state machine is inactive until the debugger reads this ID code.

- The SW-DP state machine is in RESET STATE either after power-on reset, or after the DP has switched from JTAG to SWD or after the line is high for more than 50 cycles.
- The SW-DP state machine is in IDLE STATE if the line is low for at least two cycles after

RESET state.

- After RESET state, it is mandatory to first enter into an IDLE state AND to perform a READ access of the DP-SW ID CODE register. Otherwise, the debugger will issue a FAULT acknowledge response on another transactions.

24.4.4 DP and AP read/write accesses

- Read accesses to the DP are not posted: the target response can be immediate (if ACK=OK) or can be delayed (if ACK=WAIT).
- Read accesses to the AP are posted. This means that the result of the access is returned on the next transfer. If the next access to be done is NOT an AP access, then the DP-RDBUFF register must be read to obtain the result.
- The READOK flag of the DP-CTRL/STAT register is updated on every AP read access or RDBUFF read request to know if the AP read access was successful.
- The SW-DP implements a write buffer (for both DP and AP writes), that enables it to accept a write operation even when other transactions are still outstanding. If the write buffer is full, the debugger acknowledge response is ”WAIT”. With the exception of IDCODE read or CTRL/STAT read or ABORT write which is accepted even if the write buffer is full.
- Because of the asynchronous clock domains SWCLK and HCLK, two extra SWCLK cycles are needed after a write transaction (after the parity bit) to make the write effective internally. These cycles should be applied while driving the line low (IDLE state). This is particularly important when writing the CTRL/STAT for a power-up request. If the next transaction (requiring a power-up) occurs immediately, it will fail.

24.4.5 SW-DP register

Access to these registers are initiated when APnDP=0.

A(3: 2)	**Read/write**	**CTRLSEL bit** of SELECT register	**Register**	**Description**
00	Read		IDCODE	It is set to 0x1BA01477 (identifies the SW-DP)
00	Write		ABORT	
01	Read/Write	0	DP-CTRL/STAT	Request a system or debug power-up; configure the transfer operation for AP accesses; control the compare and verify operations; read some status flags (overrun, power-up acknowledges)
01	Read/Write	1	WIRE CONTROL	Configure the physical serial port protocol (like the duration of the turnaround time).
10	Read		READ RESEND	Enables recovery of the read data from a corrupted debugger transfer, without repeating the original AP transfer.
10	Write		SELECT	Select the current access port and the active 4-word register window.

A(3: 2)	Read/write	CTRLSEL bit of SELECT register	Register	Description
11	Read/Write		READ BUFFER	This read buffer is useful because AP accesses are posted (the result of a read AP request is available on the next AP transaction). This read buffer captures data from the AP, presented as the result of a previous read, without initiating a new transaction

24.4.6 SW-AP register

Access to these registers are initiated when APnDP=1.

There are many AP Registers addressed as the combination of:

- A[3:2]
- The current value of the DP SELECT register

24.5 MCU debug module (MCUDBG)

The MCU debug module assists the debugger with the following features:

- Low power mode
- Provide timer at breakpoint, and enable the clock control of watchdog
- Control the assignment of trace pin

24.5.1 Debugg support in low power mode

To enter low-power mode, the instruction WFI or WFE must be executed. The MCU implements several low-power modes which can either deactivate the CPU clock or reduce the power of the CPU. The core does not allow FCLK or HCLK to be turned off during a debug session. As these are required for the debugger connection, during a debug, they must remain active. The MCU integrates special means to allow the user to debug code in low-power modes.

For this, the debugger host must first set some debug configuration registers to change the low-power mode behavior:

- In Sleep mode, DBG_SLEEP bit of DBGMCU_CR register must be previously set by the debugger. This will feed HCLK with the same clock that is provided to FCLK (system clock previously configured by the software).
- In Stop mode, the bit DBG_STOP must be previously set by the debugger. This will enable the internal RC oscillator clock to feed FCLK and HCLK in STOP mode.

24.5.2 Support timer, watchdog

When generating a breakpoint, it is necessary to select the operating mode of the counter based on the different uses of the timer and the watchdog:

- The counter continues to count when a breakpoint is generated. This is often used when outputting a PWM controlled motor.

- When a breakpoint is generated, the counter stops counting. This is required for the watchdog counter.

24.5.3 Debug MCU configuration register

This register allows the configuration of the MCU under DEBUG. This concerns:

- Low-power mode support
- Timer and watchdog counter support
- Trace pin assignment

This DBGMCU_CR is mapped on the External APB bus at address 0x40013404. It is asynchronously reset by the PORESET (and not the system reset). It can be written by the debugger under system reset.

If the debugger host does not support these features, it is still possible for the user software to write to these registers.

24.6 Description of DBG Register

Table 76. Summary of DBG Register

Offset	Acronym	Register Name	Reset	Section
0x00	DBG	DBG Control Register	0x00000000	section 24.6.1

24.6.1 DBG Control Register(DBG_CR)

Address: 0x40013404 Only 32-bits access supported.

POR Reset: 0x0000 0000 (not reset by system reset)

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		DBG_TIMx_STOP				DBG_WWDG_STOP	DBG_IWDG_STOP	Reserved					DBG_STANDBY	DBG_STOP	DBG_SLEEP
		w	w	w	w	w	w						w	w	w

Bit	Field	Type	Reset	Description
31:14	Reserved			Always read as 0.
13:10	DBG_TIMx_STOP	w	0x00	TIMx stopped when core is halted (x=4.1) 0: The clock of the involved Timer Counter is fed even if the core is halted 1: The clock of the involved Timer counter is stopped when the core is halted

Bit	Field	Type	Reset	Description
9	DBG_WWDG_STOP	w	0x00	Debug window watchdog stopped when core is halted 0: The window watchdog counter clock continues even if the core is halted 1: The window watchdog counter clock is stopped when the core is halted
8	DBG_IWDG_STOP	w	0x00	Debug independent watchdog stopped when core is halted 0: The watchdog counter clock continues even if the core is halted 1: The watchdog counter clock is stopped when the core is halted
7:3	Reserved			Always read as 0.
2	DBG_STANDBY	w	0x00	Debug Standby mode 0: (FCLK=Off, HCLK=Off) The whole digital part is unpowered. From software point of view, exiting from Standby is identical than fetching reset vector (except a few status bit indicated that the MCU is resuming from Standby) 1: (FCLK=On, HCLK=On) In this case, the digital part is not unpowered and FCLK and HCLK are provided by the internal RC oscillator which remains active. In addition, the MCU generate a system reset during Standby mode so that exiting from Standby is identical than fetching from reset
1	DBG_STOP	w	0x00	Debug Stop mode
0	DBG_SLEEP	w	0x00	Debug Sleep mode 0: (FCLK=On, HCLK=Off) In Sleep mode, FCLK is clocked by the system clock as previously configured by the software while HCLK is disabled. In Sleep mode, the clock controller configuration is not reset and remains in the previously programmed state. Consequently, when exiting from Sleep mode, the software does not need to reconfigure the clock controller. 1: (FCLK=On, HCLK=On) In this case, when entering Sleep mode, HCLK is fed by the same clock that is provided to FCLK (system clock as previously configured by the software).

25 Revision history

Revision history

Table 77. Revision History

Date	Rversion	Changes
2019/08/06	Rev1.19	Modify the typo at the memory and bus architecture
2019/07/23	Rev1.18	SPI_EXTCTL is only valid when the DW8_32 bit is ’0’
2019/07/16	Rev1.17	The minimum UART BRR register is 4
2019/07/09	Rev1.16	Modify comparatorMode parameter description
2019/07/03	Rev1.15	Modify the capture/compare register in the advanced timer. 5 Describe the error and change CCMR5 to CCMR3.
2019/05/09	Rev1.14	Modify port mode configuration Brake and deadband registers in the advanced timer BDTR deadband generator setting bit DTG Correction
2019/04/16	Rev1.13	Modify the adc calculation formula
2019/03/11	Rev1.12	Modify the RCC_CIR register description
2019/01/17	Rev1.11	Add RCC register description and deleting the PWM module
2019/01/10	Rev1.10	Modify the PWR description
2019/01/09	Rev1.10	Modify comparator

Date	Rversion	Changes
2018/12/22	Rev1.09	TIM14 modifies picture 565511 to remove some of the extra changes. In the TIMX_16bit Function Description section, in the Input Capture section, change ”CC1S = 01 in the TIMx_CCR1 register” to ”CC1S = 01 in the TIMx_CCMR1 register”. TIM1/8 replaces ”capture” in the text with ”capture”. In the TIM1/8 Function Description section, the Input Capture section modifies ”CC1S = 01 in the TIMx_CCR1 register” to ”CC1S=01 in the TIMx_CCMR1 register”. The synchronization part of the TIM1/8 TIMx timer and external trigger changes the incorrect writing ”IMx_CR1” to ”TIMx_CR1”. DIV_Mantiss cannot be 0 in UART_BRR.
2018/12/20	Rev1.08	UART_CSR changed to UART current status register