MICROS 32 BITS STM - IC

ROBINSON JIMENEZ MORENO

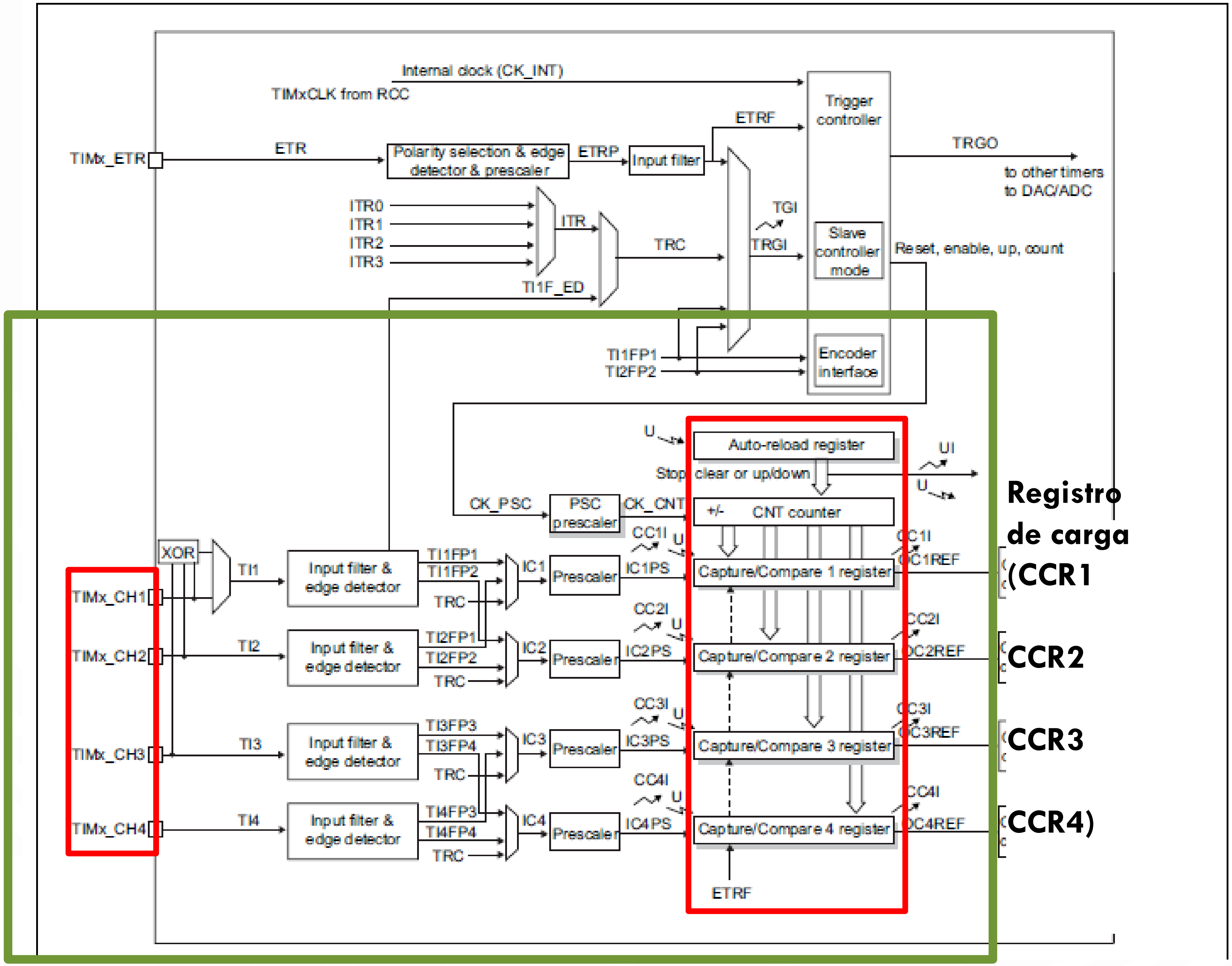
SCIENTIÆ · PATRIÆ · FAMILIÆ

UNIVERSIDAD MILITAR
NUEVA GRANADA

INGENIERÍA
MECATRÓNICA
UMNG

MODO IC

Figure 207. General-purpose timer block diagram



Canal de entrada (IC)

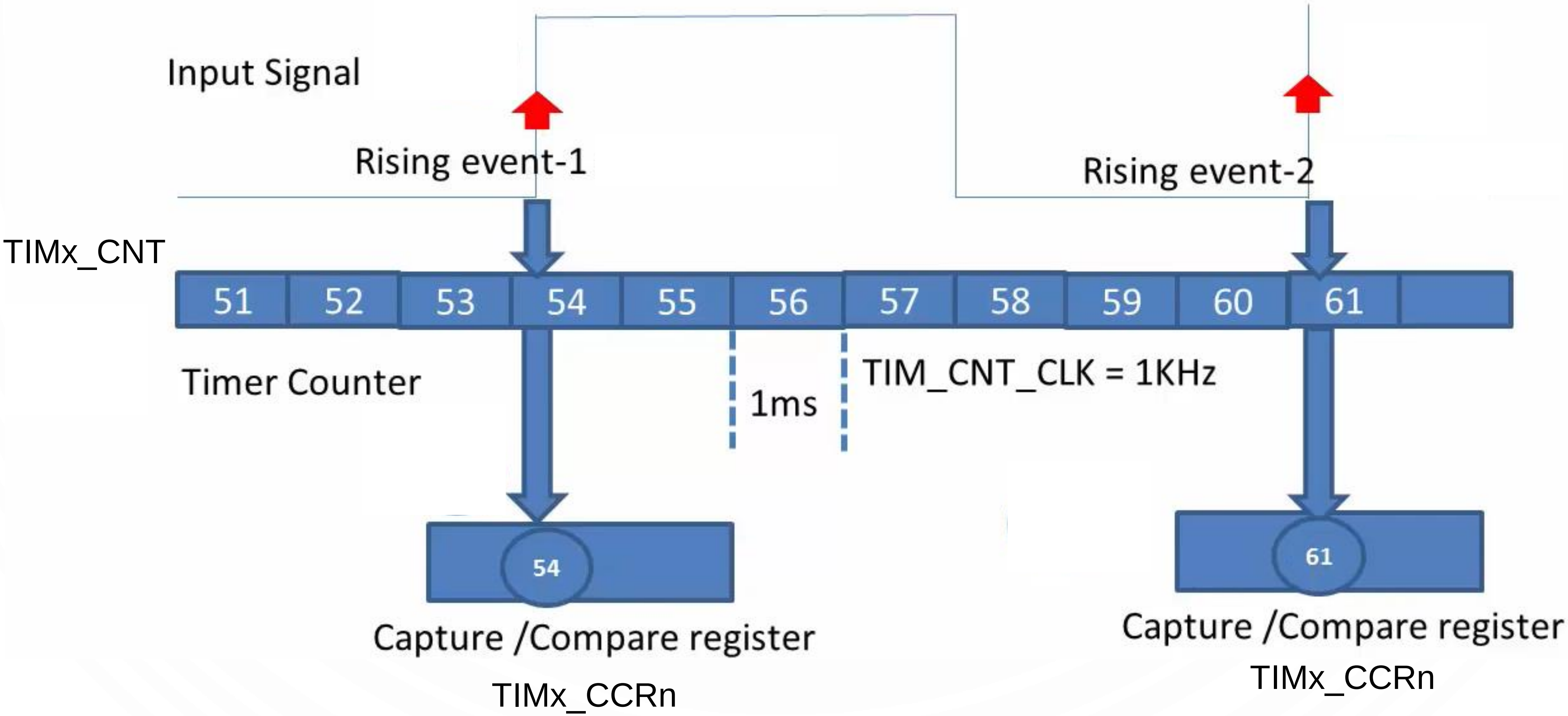
Registro de carga (CCR1

CCR2

CCR3

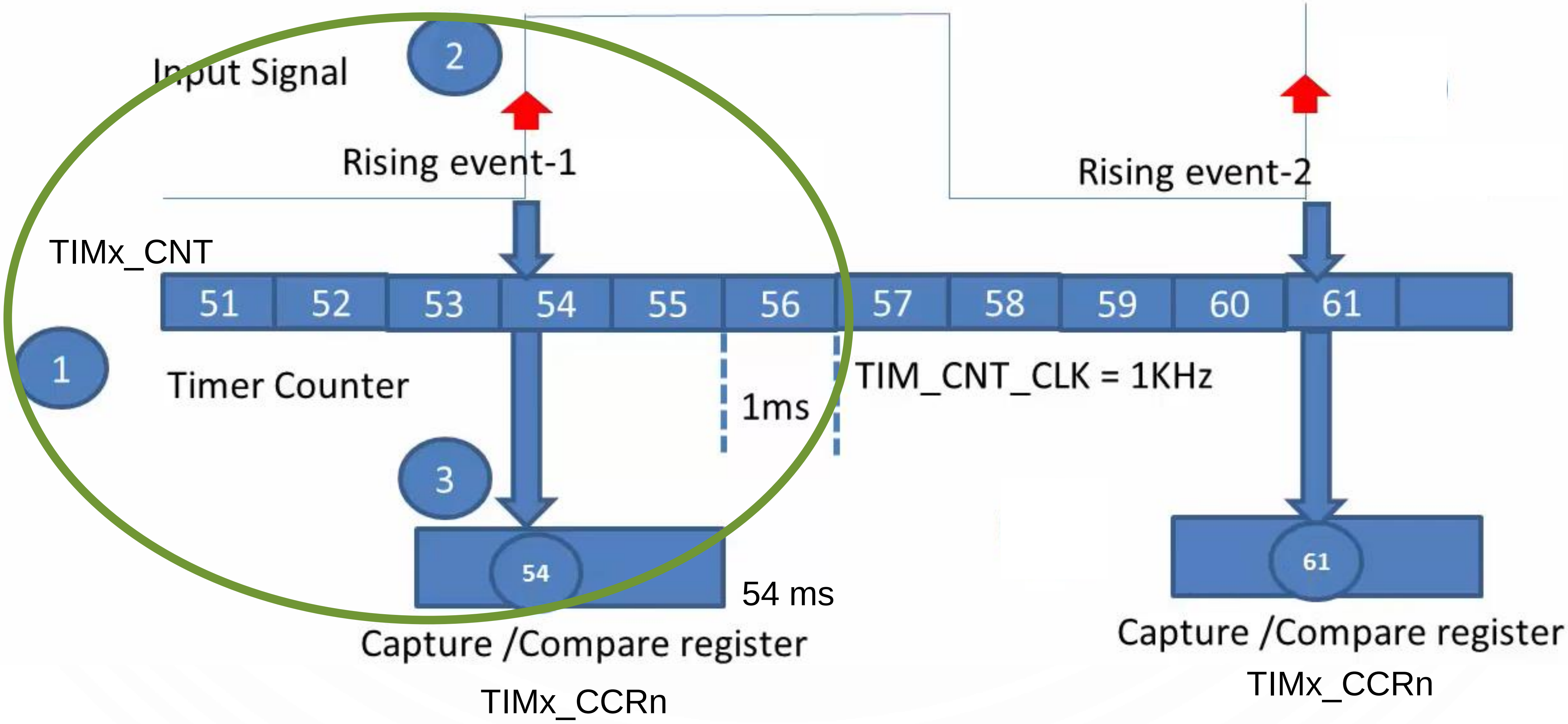
CCR4)

TIMERS - INPUT CAPTURE MODE



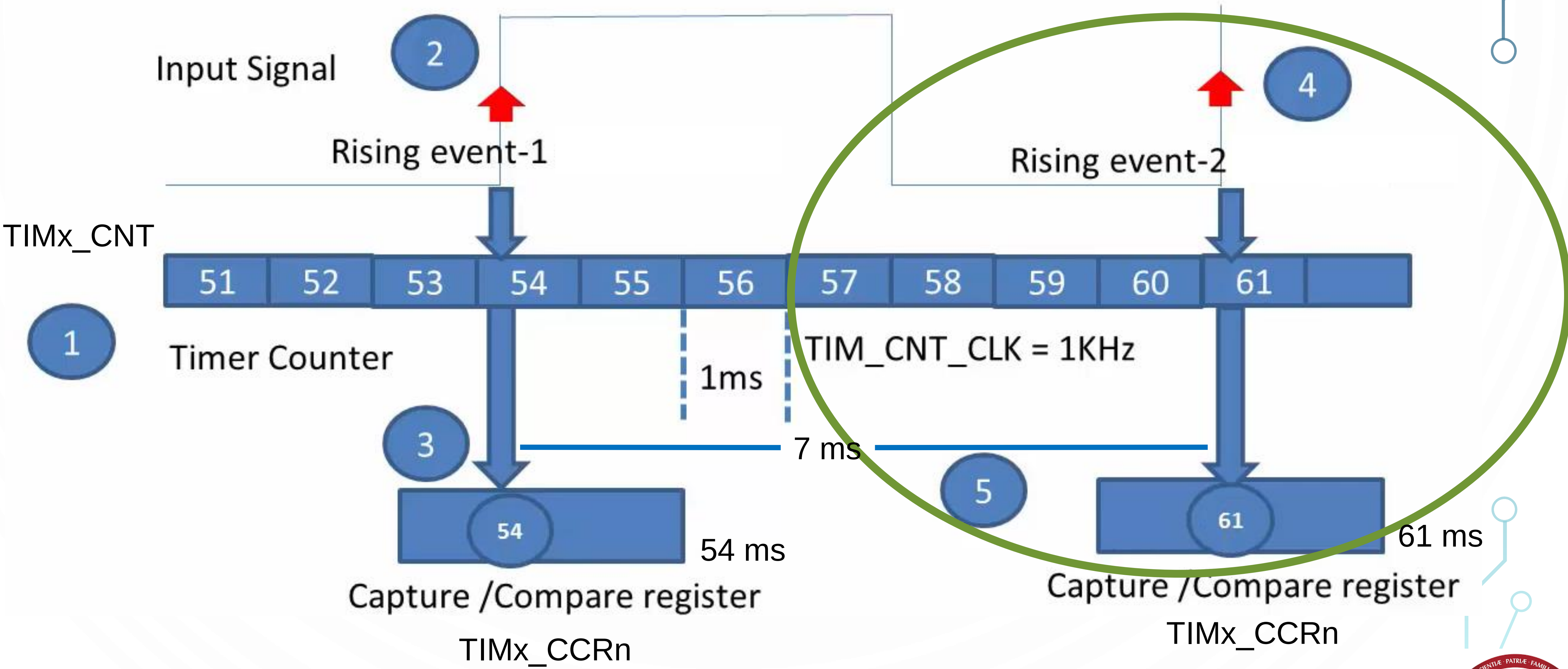
STM32F ⇨ x=1…..14 y n=1…4

TIMERS - INPUT CAPTURE MODE

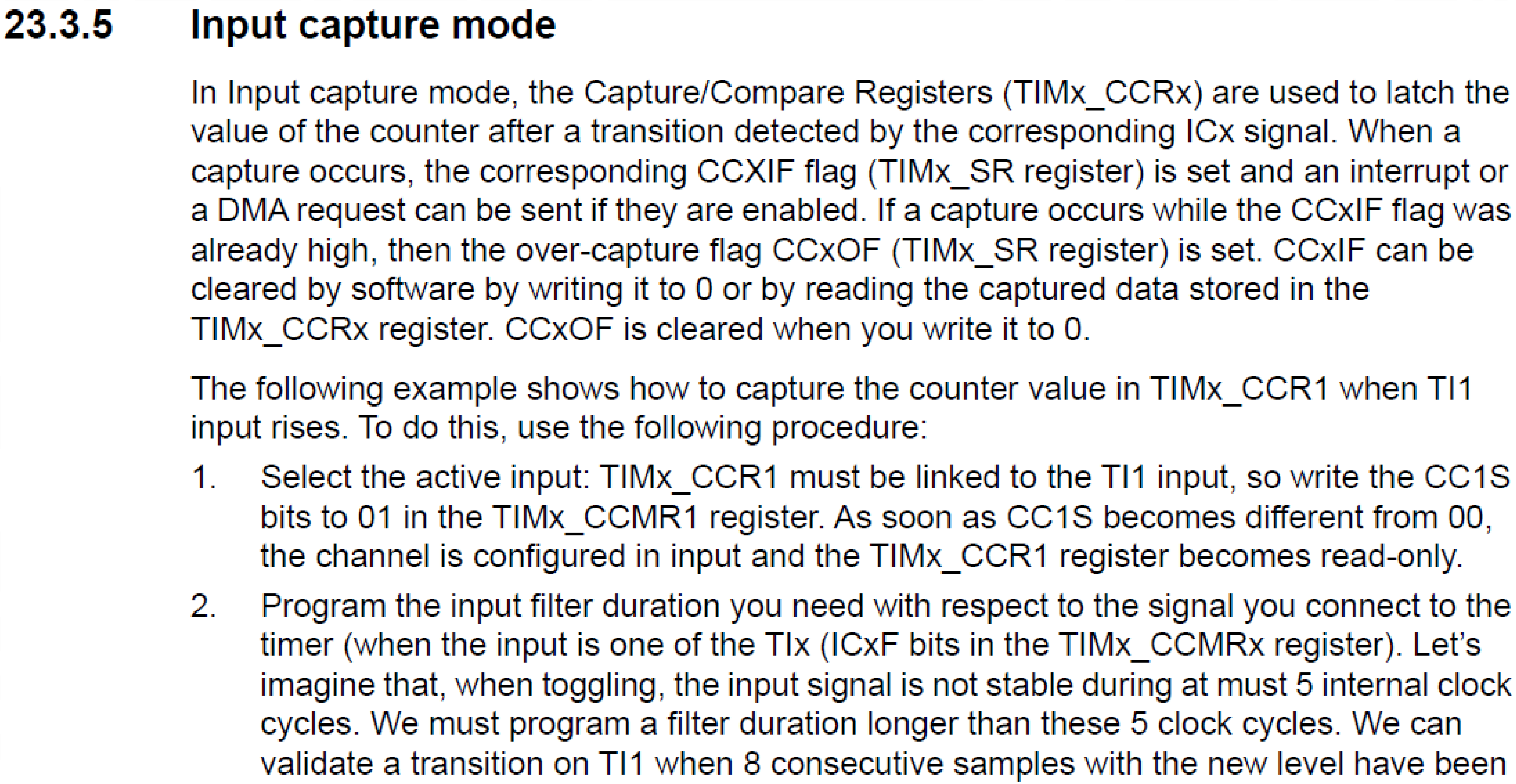


STM32F ⇨ x=1.....14 y n=1...4

TIMERS - INPUT CAPTURE MODE



STM32F ⇨ x=1…..14 y n=1…4

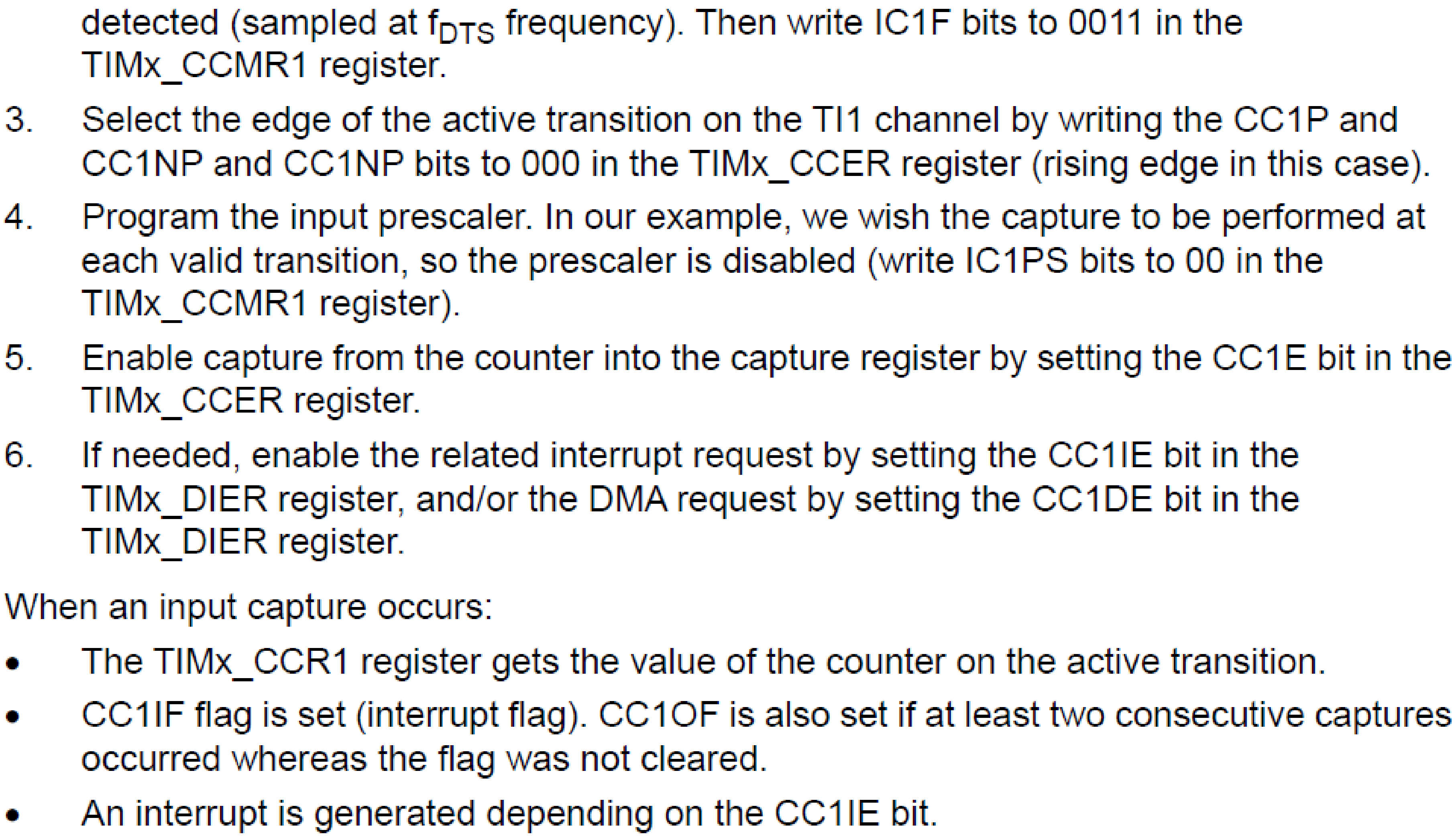


23.3.5 Input capture mode

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 you write it 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:

1. 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.
2. Program the input filter duration you need with respect to the signal you connect to the timer (when the input is one of the TIx (ICxF bits in the TIMx_CCMRx register). Let’s imagine that, when toggling, the input signal is not stable during at must 5 internal clock cycles. We must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples with the new level have been



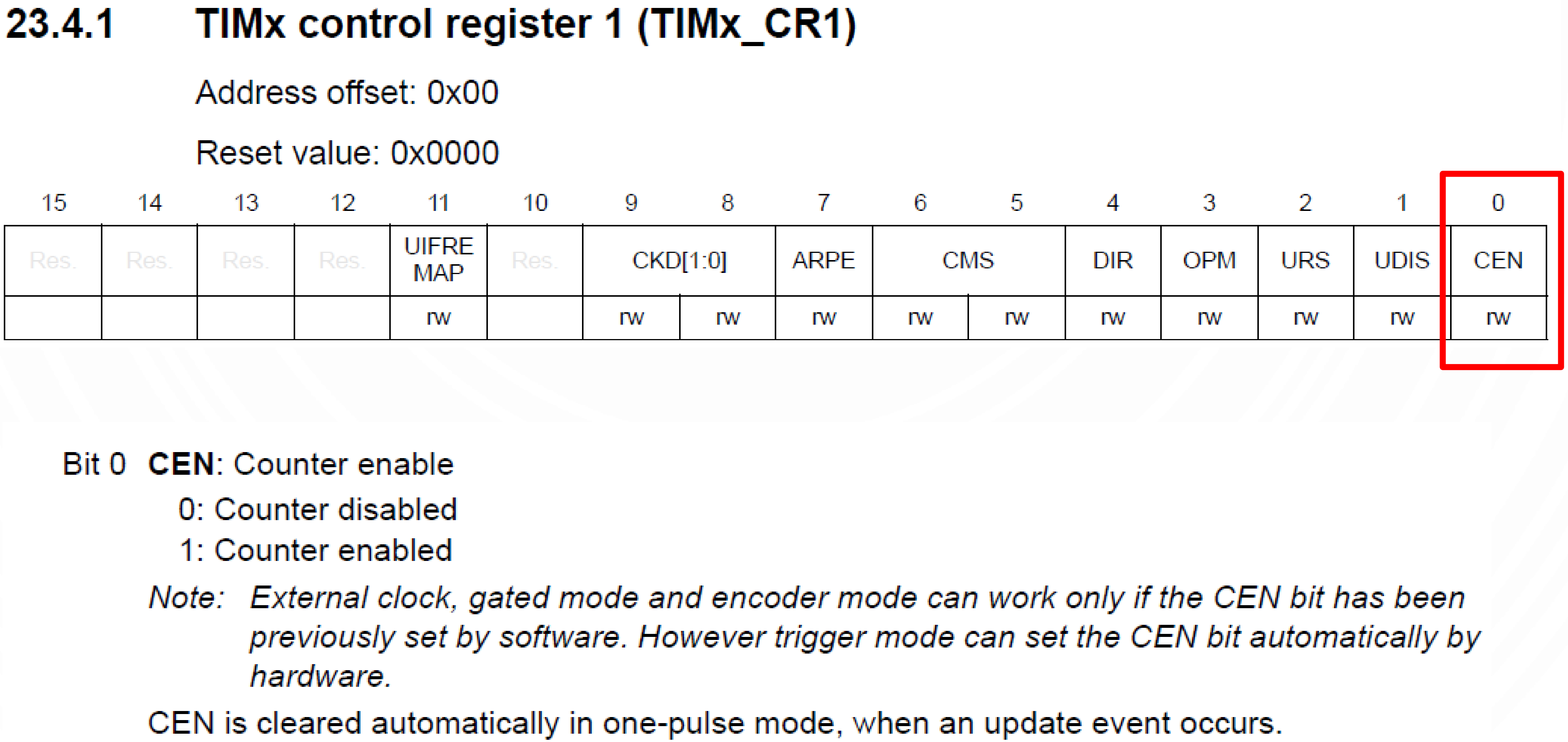
detected (sampled at f_{DTS} frequency). Then write IC1F bits to 0011 in the TIMx_CCMR1 register.

3. Select the edge of the active transition on the TI1 channel by writing the CC1P and CC1NP and CC1NP bits to 000 in the TIMx_CCER register (rising edge in this case).
4. Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS bits to 00 in the TIMx_CCMR1 register).
5. Enable capture from the counter into the capture register by setting the CC1E bit in the TIMx_CCER register.
6. If needed, enable the related interrupt request by setting the CC1IE bit in the TIMx_DIER register, and/or the DMA request by setting the CC1DE bit in the TIMx_DIER register.

When an input capture occurs:

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

```
TIM2->CR1=1;
```



23.4.1 TIMx control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	UIFRE MAP	Res.	CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
				rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

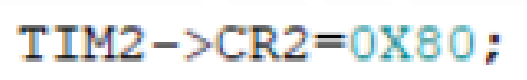
Bit 0 **CEN**: Counter enable

0: Counter disabled

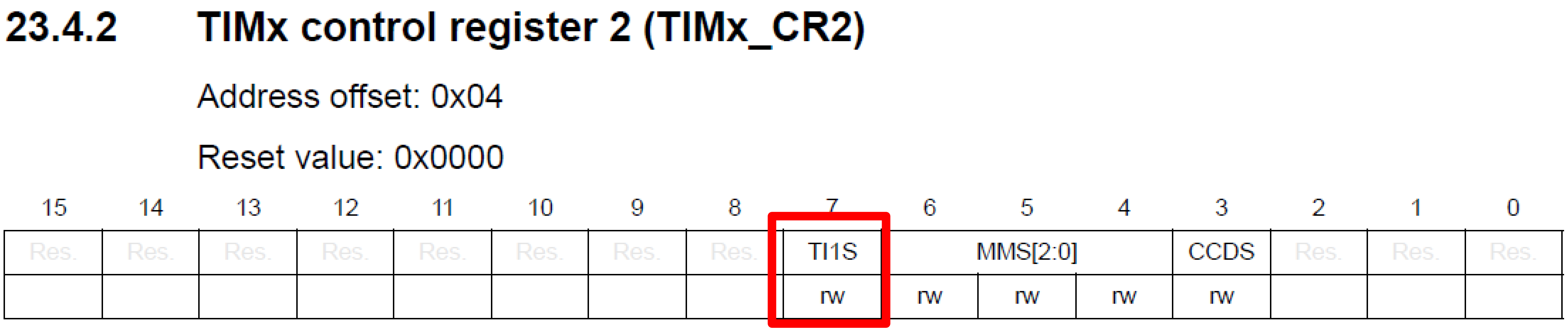
1: Counter enabled

Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.

CEN is cleared automatically in one-pulse mode, when an update event occurs.



```
TIM2->CR2=0X80;
```



23.4.2 TIMx control register 2 (TIMx_CR2)

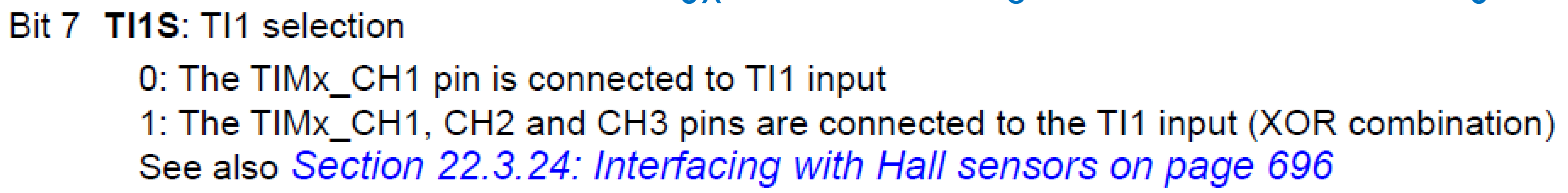
Address offset: 0x04

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
								rw	rw	rw	rw	rw			

BIN 1 0 0 0 0 0 0 0

0X 8 0

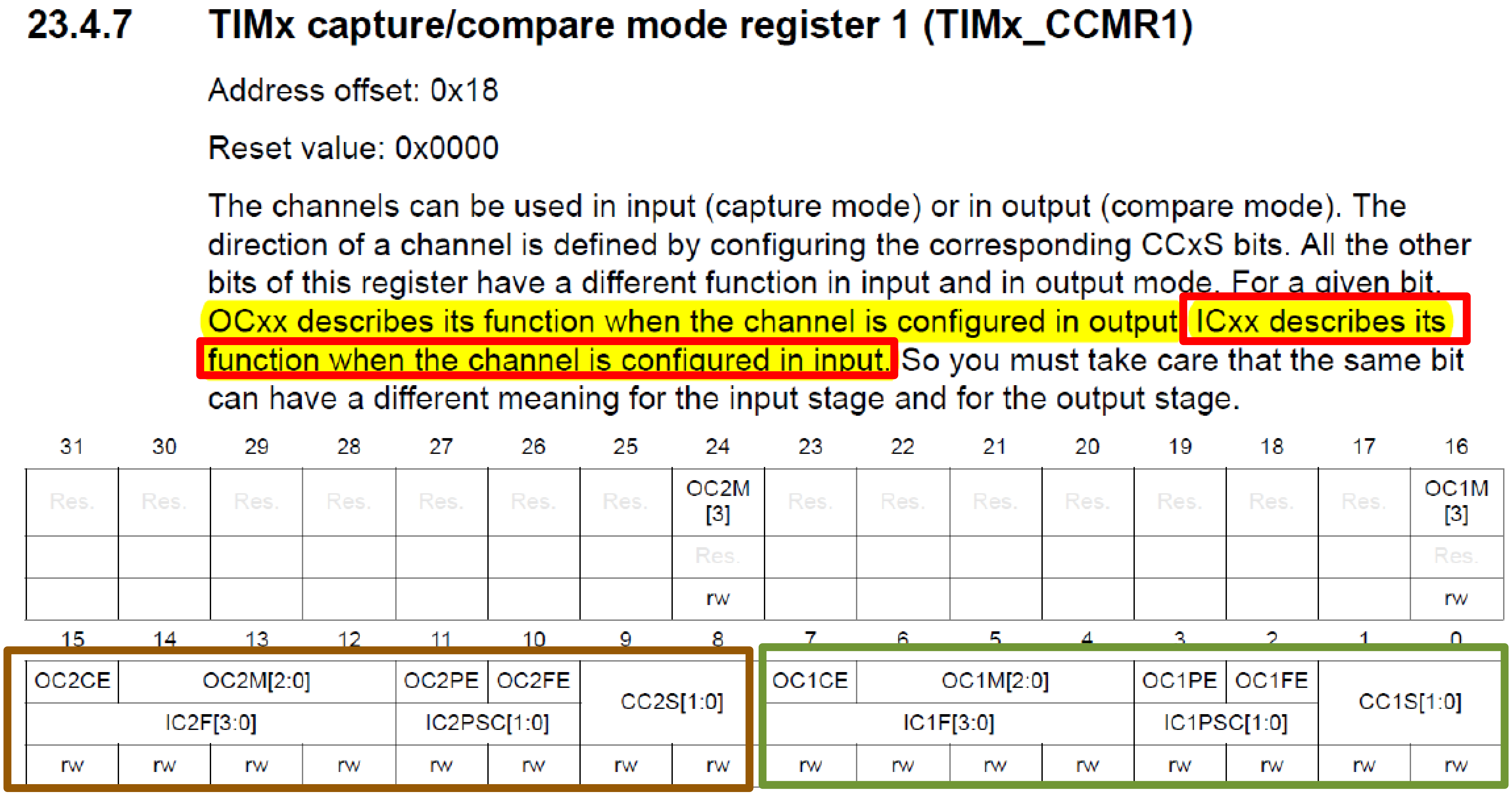


Bit 7 **TI1S**: TI1 selection

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)

See also *Section 22.3.24: Interfacing with Hall sensors on page 696*



23.4.7 TIMx capture/compare mode register 1 (TIMx_CCMR1)

Address offset: 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 you must take care that the same bit can have a different meaning for the input stage and for the output stage.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M [3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [3]
							Res.								Res.
							rw								rw

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

CANAL 2

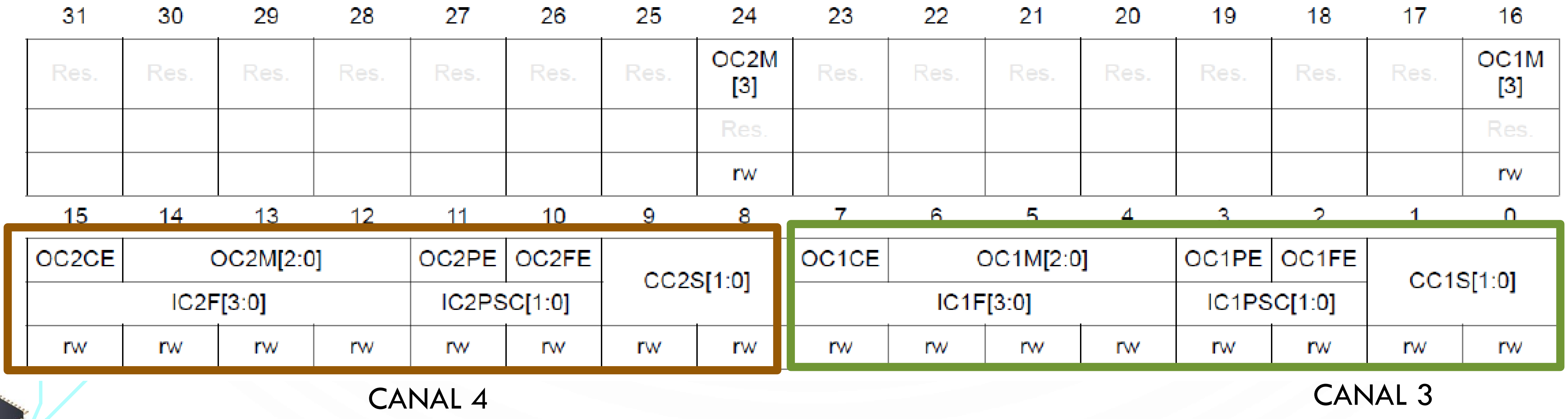
CANAL 1

23.4.7 TIMx capture/compare mode register 1 (TIMx_ CCMR2)

Address offset: 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 you must take care that the same bit can have a different meaning for the input stage and for the output stage.

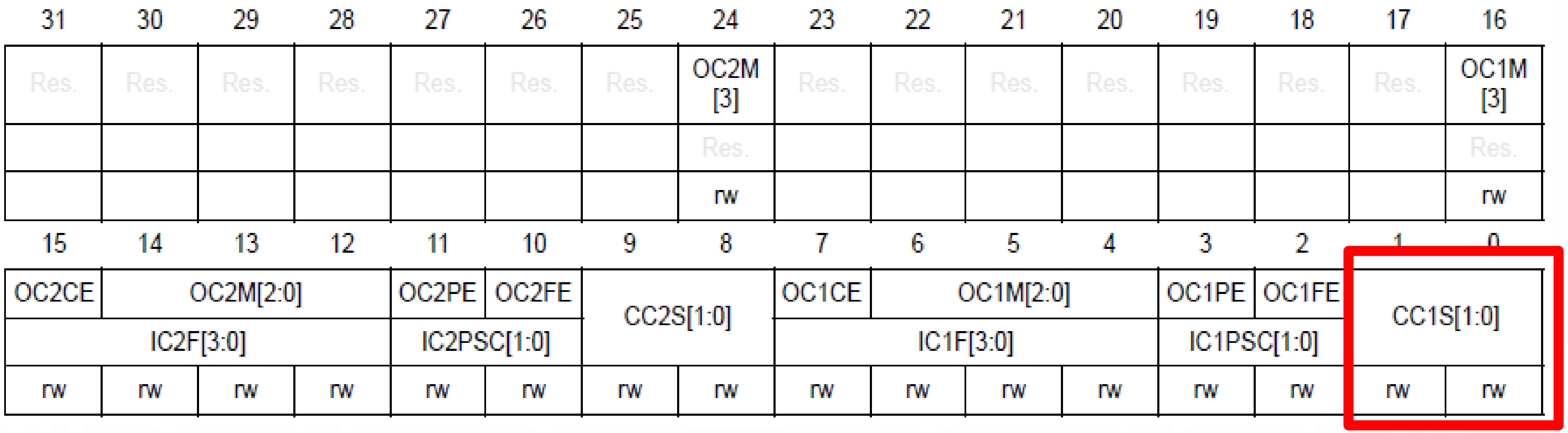


31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M [3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [3]
							Res.								Res.
							rw								rw

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

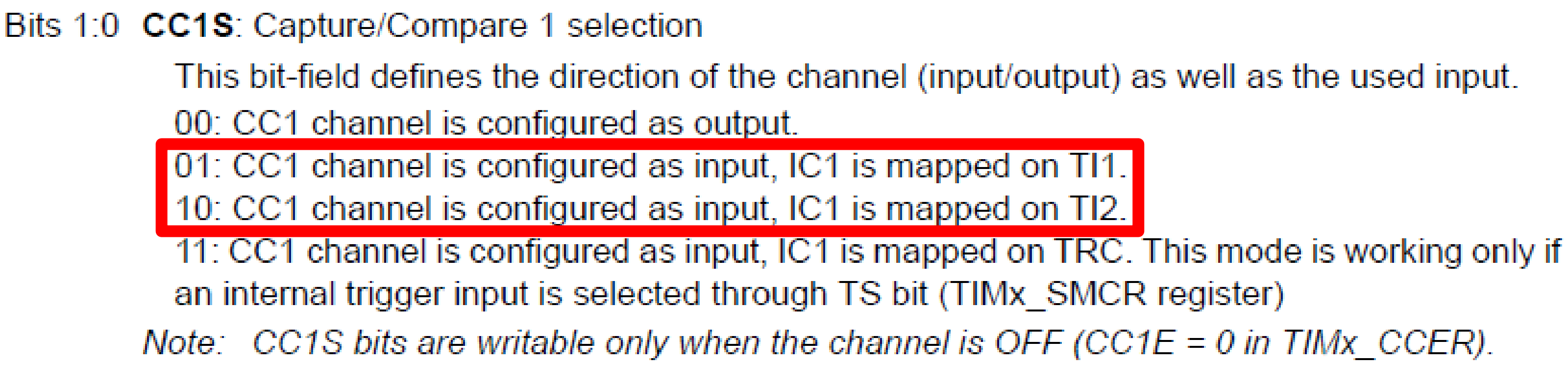
CANAL 4

CANAL 3



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M [3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [3]
							Res.								Res.
							rw								rw

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



Bits 1:0 **CC1S**: Capture/Compare 1 selection

This bit-field defines the direction of the channel (input/output) as well as the used input.

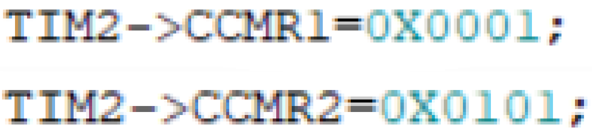
00: CC1 channel is configured as output.

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).

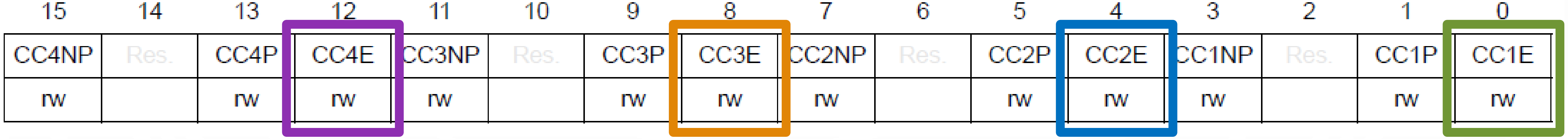


```
TIM2->CCMR1=0X0001;
TIM2->CCMR2=0X0101;
```

23.4.9 TIMx capture/compare enable register (TIMx_CCER)

Address offset: 0x20

Reset value: 0x0000



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

Bit 0 **CC1E**: *Capture/Compare 1 output enable.*

CC1 channel configured as output:

0: Off - OC1 is not active

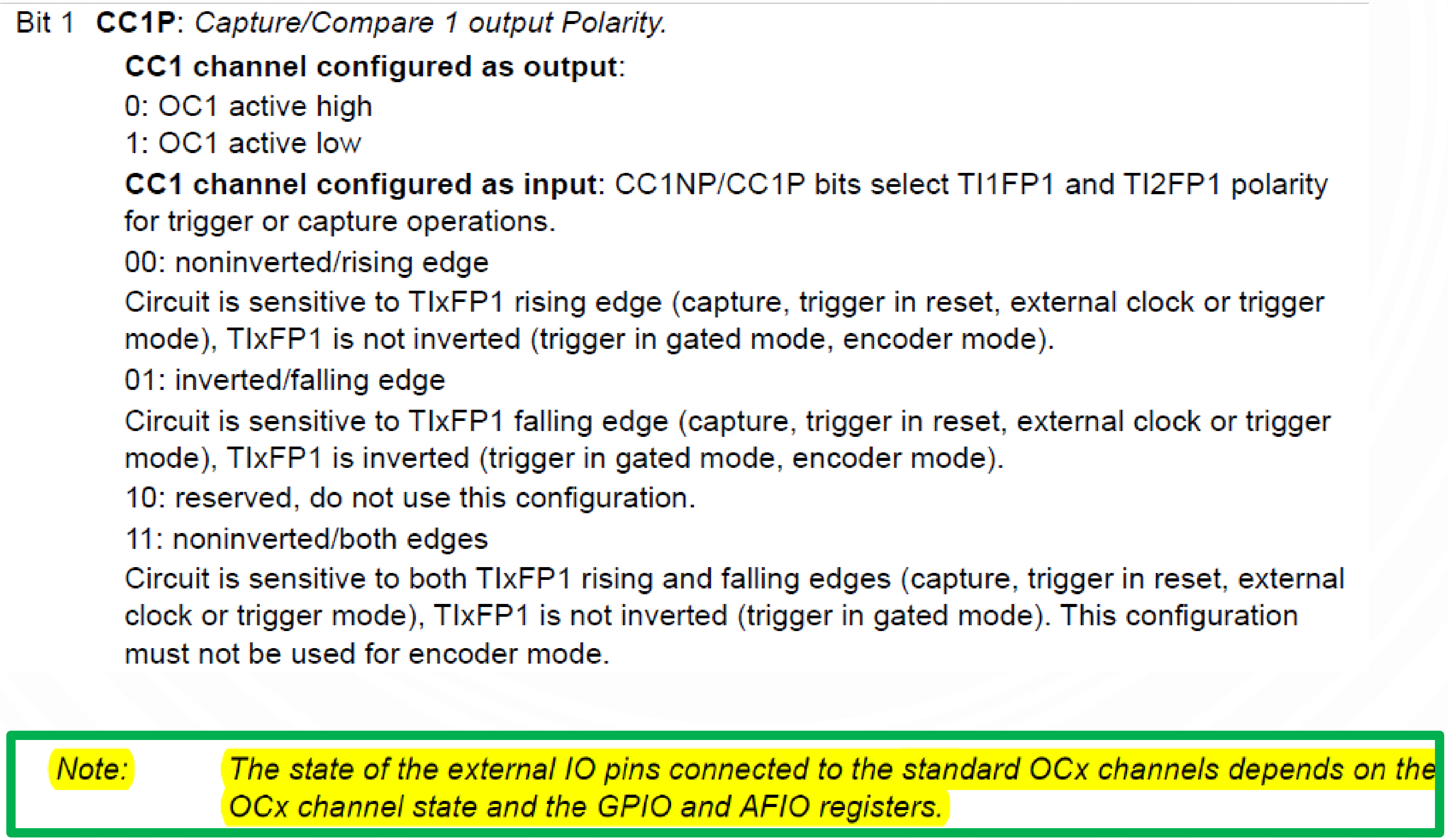
1: On - OC1 signal is output on the corresponding output pin

CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (TIMx_CCR1) or not.

0: Capture disabled

1: Capture enabled

```
TIM2->CCER=1;
```



Bit 1 **CC1P**: *Capture/Compare 1 output Polarity.*

CC1 channel configured as output:

0: OC1 active high

1: OC1 active low

CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations.

00: noninverted/rising edge

Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode).

01: inverted/falling edge

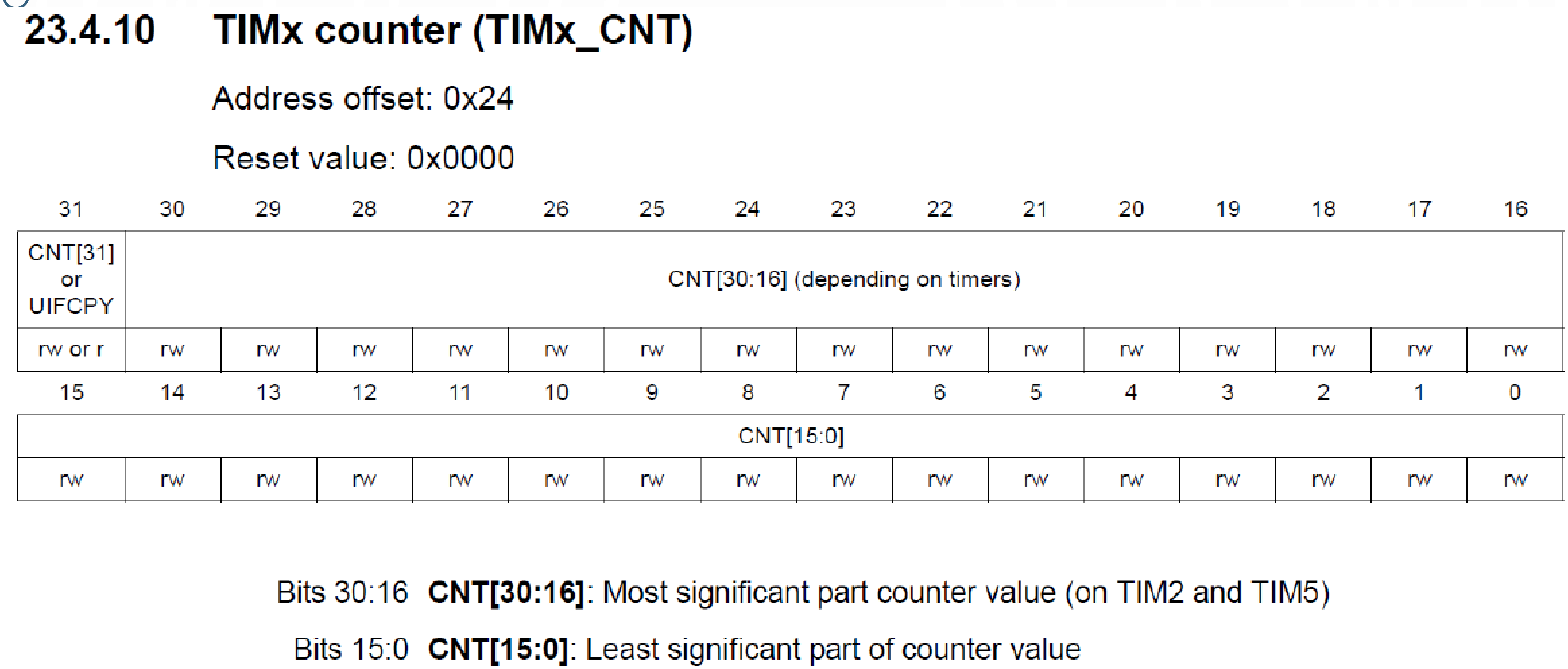
Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode).

10: reserved, do not use this configuration.

11: noninverted/both edges

Circuit is sensitive to both TIxFP1 rising and falling edges (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode). This configuration must not be used for encoder mode.

Note: *The state of the external IO pins connected to the standard OCx channels depends on the OCx channel state and the GPIO and AFIO registers.*



23.4.10 TIMx counter (TIMx_CNT)

Address offset: 0x24

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31] or UIFCPY	CNT[30:16] (depending on timers)														
rw or r	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[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 30:16 **CNT[30:16]**: Most significant part counter value (on TIM2 and TIM5)

Bits 15:0 **CNT[15:0]**: Least significant part of counter value

23.4.11 TIMx prescaler (TIMx_PSC)

```
TIM2->PSC=40000;
```

Address offset: 0x28

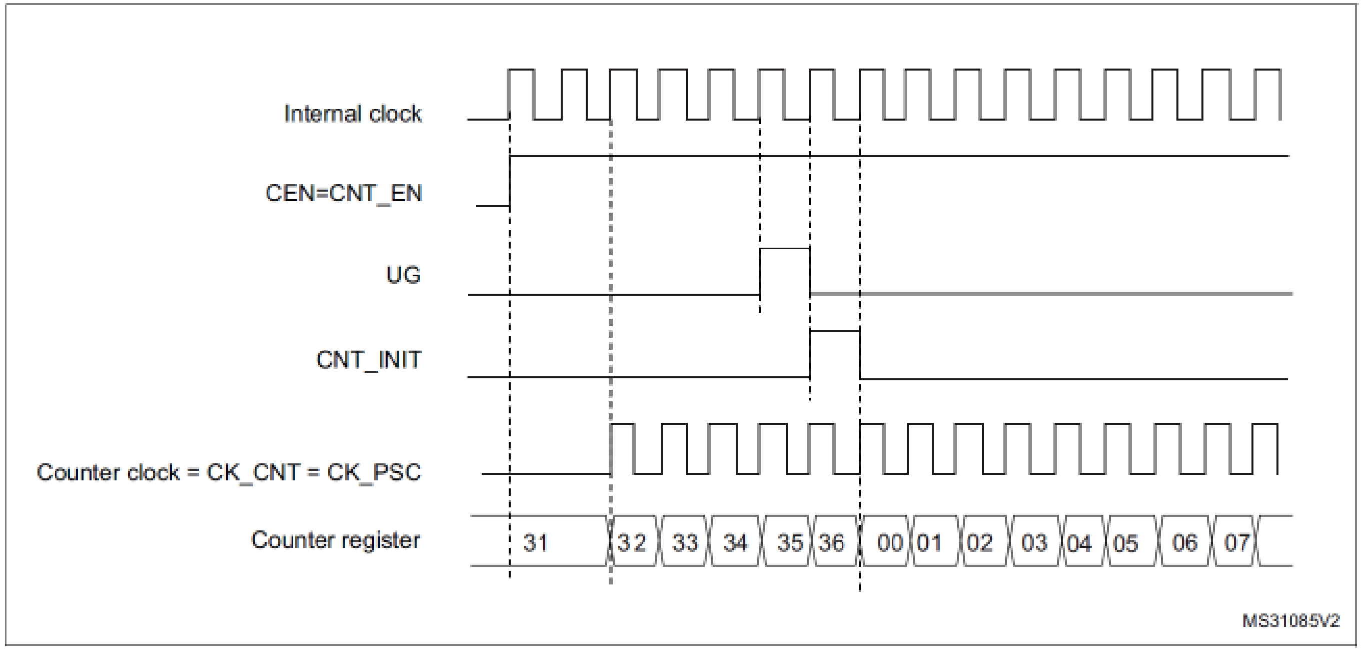
Reset value: 0x0000

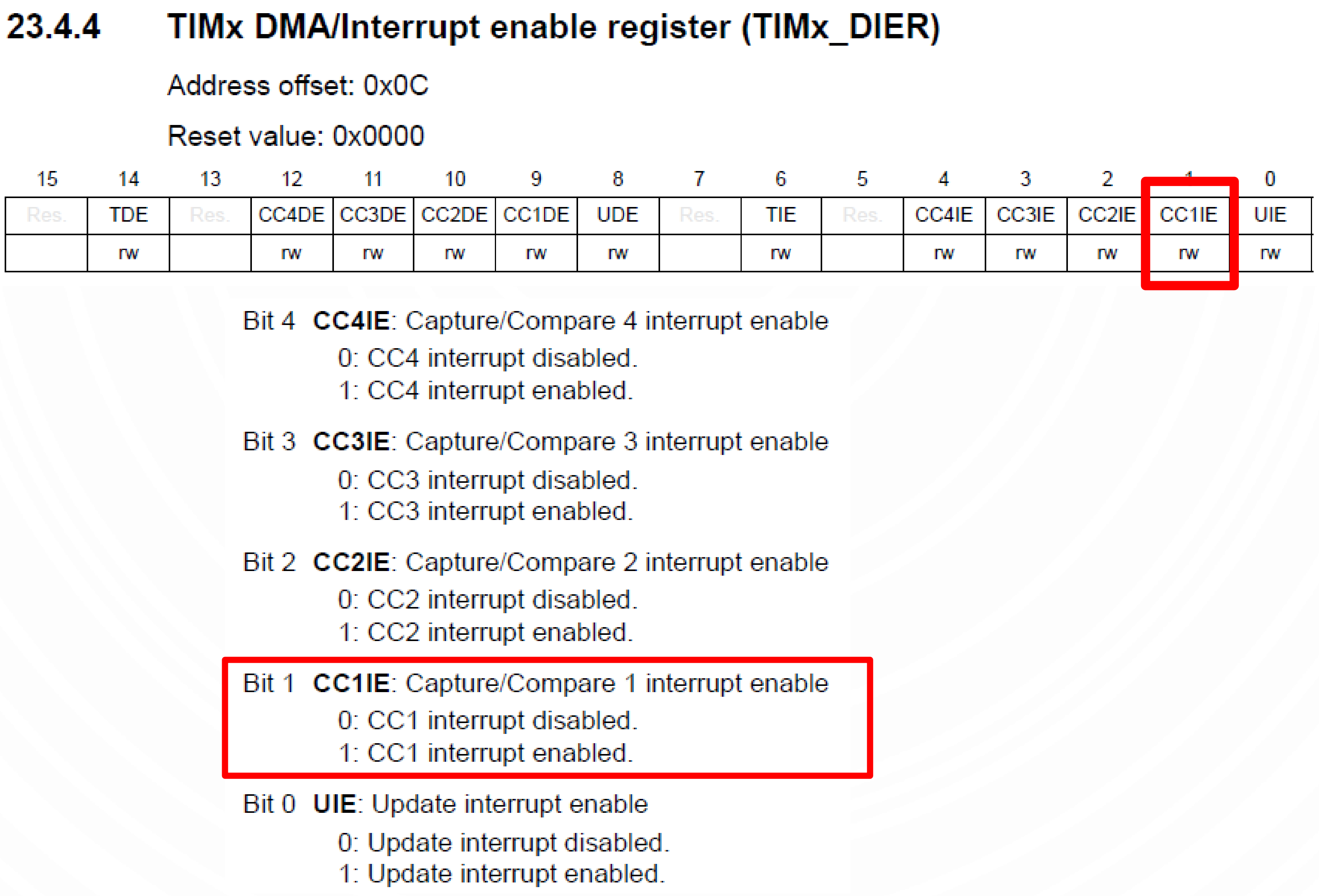
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 15:0 **PSC[15:0]**: Prescaler value

The counter clock frequency CK_CNT is equal to f_{CK_PSC} / (PSC[15:0] + 1).

PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIMx_EGR register or through trigger controller when configured in “reset mode”).





23.4.4 TIMx DMA/Interrupt enable register (TIMx_DIER)

Address offset: 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 4 **CC4IE**: Capture/Compare 4 interrupt enable

0: CC4 interrupt disabled.
1: CC4 interrupt enabled.

Bit 3 **CC3IE**: Capture/Compare 3 interrupt enable

0: CC3 interrupt disabled.
1: CC3 interrupt enabled.

Bit 2 **CC2IE**: Capture/Compare 2 interrupt enable

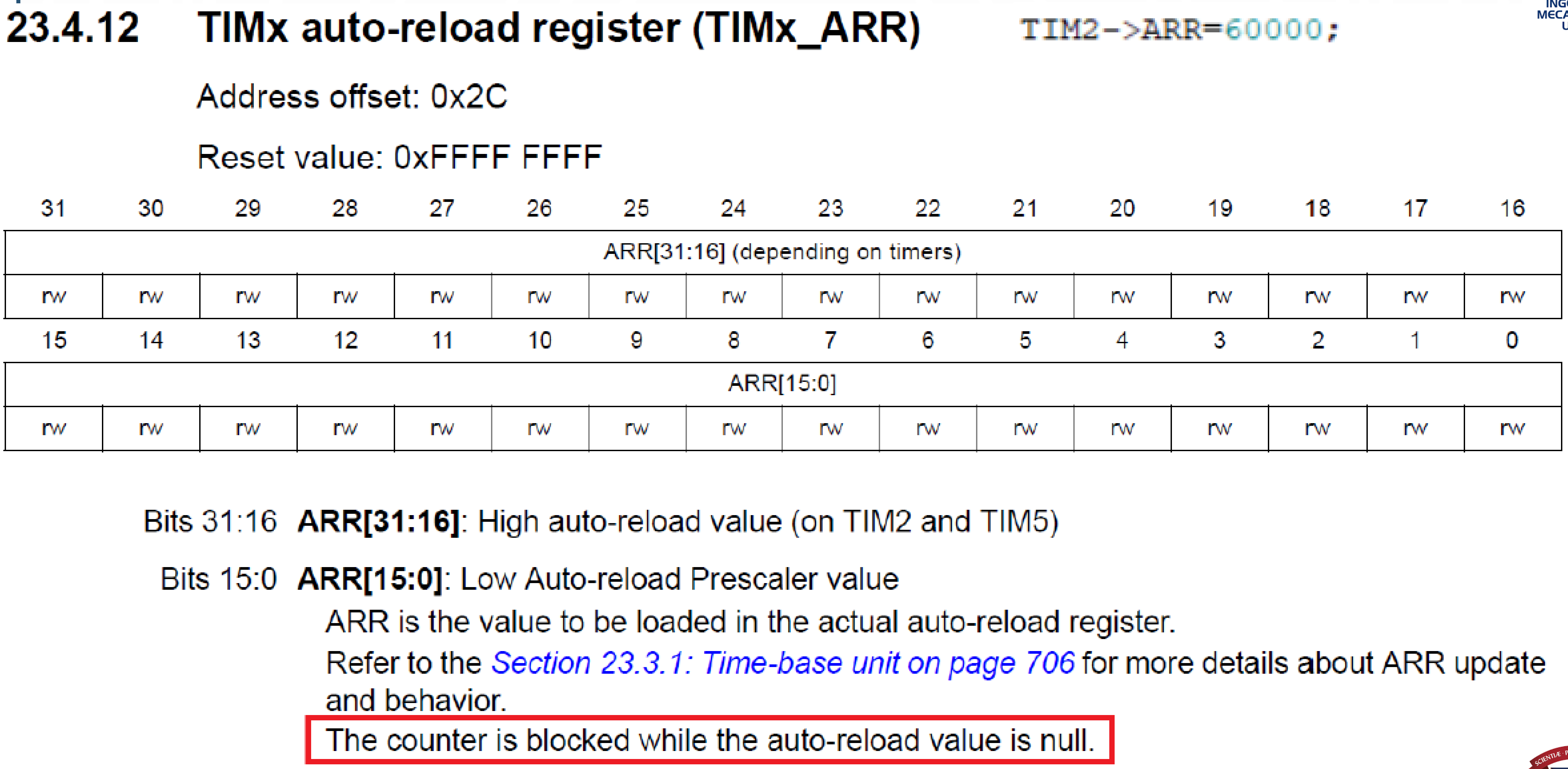
0: CC2 interrupt disabled.
1: CC2 interrupt enabled.

Bit 1 **CC1IE**: Capture/Compare 1 interrupt enable

0: CC1 interrupt disabled.
1: CC1 interrupt enabled.

Bit 0 **UIE**: Update interrupt enable

0: Update interrupt disabled.
1: Update interrupt enabled.



23.4.12 TIMx auto-reload register (TIMx_ARR)

```
TIM2->ARR=60000;
```

Address offset: 0x2C

Reset value: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16] (depending on timers)															
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[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

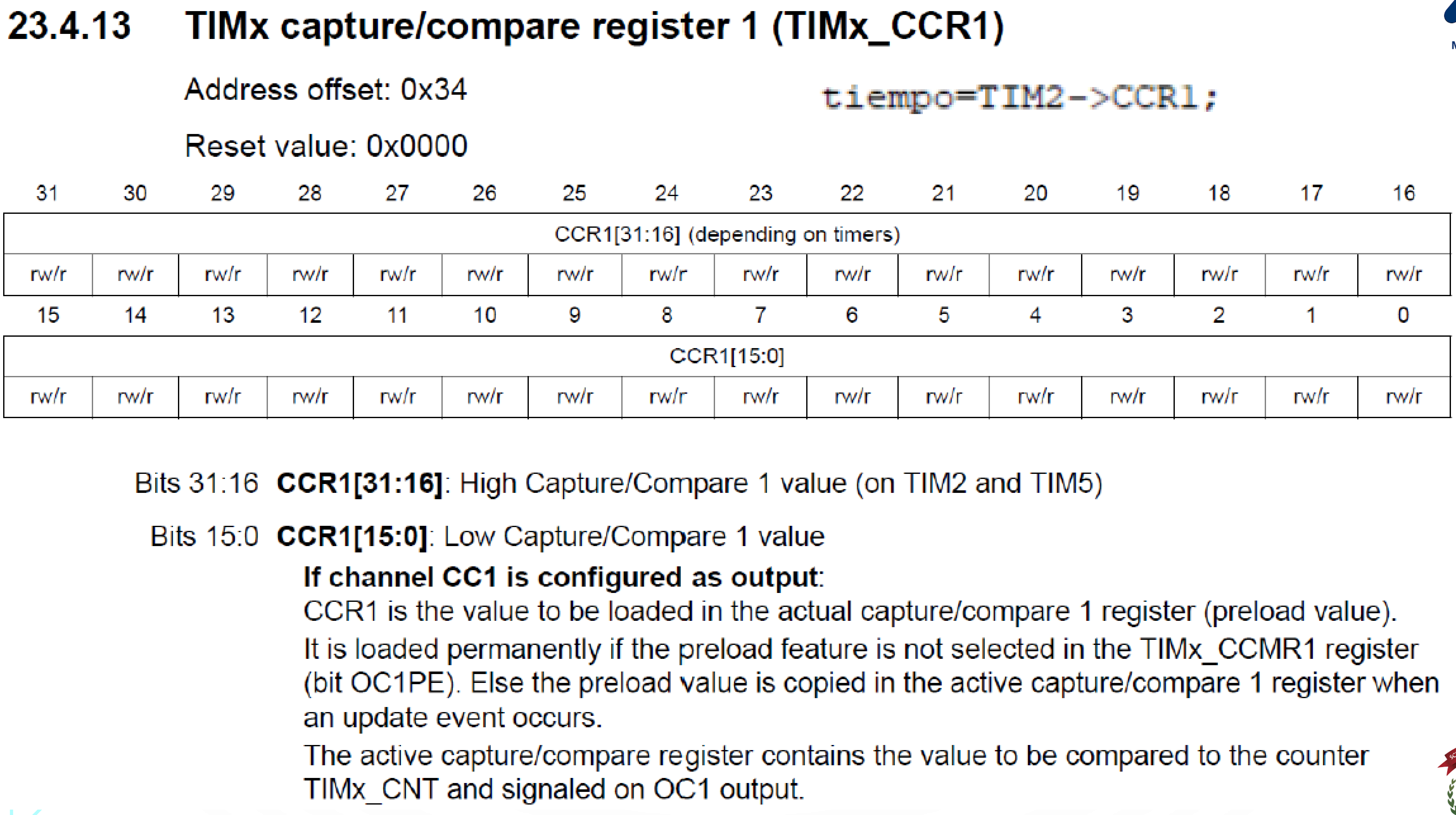
Bits 31:16 **ARR[31:16]**: High auto-reload value (on TIM2 and TIM5)

Bits 15:0 **ARR[15:0]**: Low Auto-reload Prescaler value

ARR is the value to be loaded in the actual auto-reload register.

Refer to the *Section 23.3.1: Time-base unit on page 706* for more details about ARR update and behavior.

The counter is blocked while the auto-reload value is null.



23.4.13 TIMx capture/compare register 1 (TIMx_CCR1)

Address offset: 0x34

Reset value: 0x0000

```
tiempo=TIM2->CCR1;
```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16] (depending on timers)															
rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r	rw/r

Bits 31:16 **CCR1[31:16]**: High Capture/Compare 1 value (on TIM2 and TIM5)

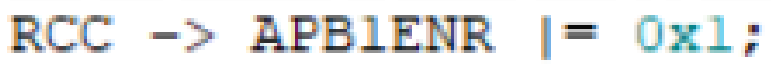
Bits 15:0 **CCR1[15:0]**: Low Capture/Compare 1 value

If channel CC1 is configured as output:

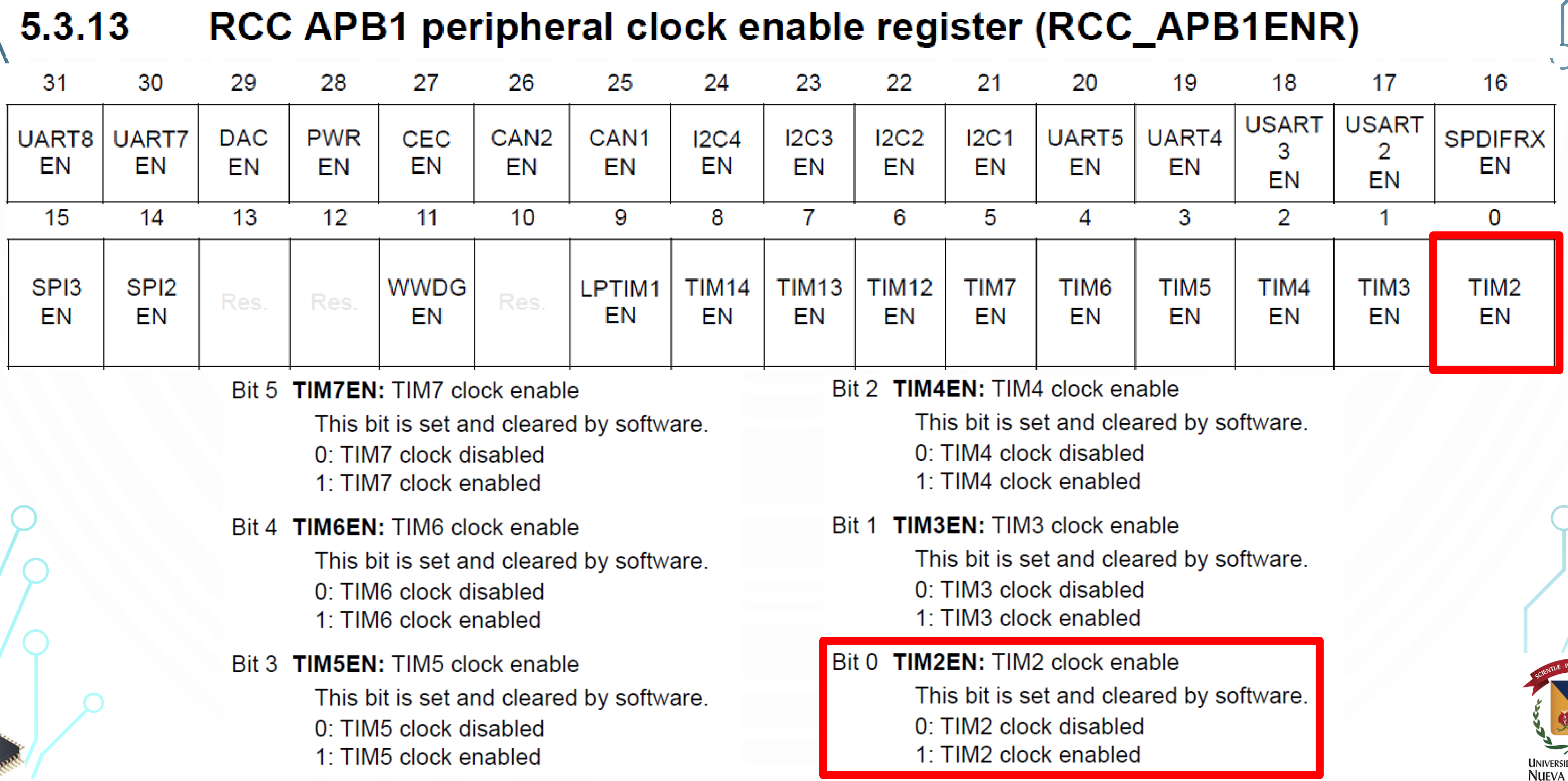
CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value).

It is loaded permanently if the preload feature is not selected in the TIMx_CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs.

The active capture/compare register contains the value to be compared to the counter TIMx_CNT and signaled on OC1 output.



```
RCC -> APB1ENR |= 0x1;
```



5.3.13 RCC APB1 peripheral clock enable register (RCC_APB1ENR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UART8 EN	UART7 EN	DAC EN	PWR EN	CEC EN	CAN2 EN	CAN1 EN	I2C4 EN	I2C3 EN	I2C2 EN	I2C1 EN	UART5 EN	UART4 EN	USART 3 EN	USART 2 EN	SPDIFRX EN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Res.	Res.	WWDG EN	Res.	LPTIM1 EN	TIM14 EN	TIM13 EN	TIM12 EN	TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN

Bit 5 **TIM7EN:** TIM7 clock enable

This bit is set and cleared by software.
0: TIM7 clock disabled
1: TIM7 clock enabled

Bit 4 **TIM6EN:** TIM6 clock enable

This bit is set and cleared by software.
0: TIM6 clock disabled
1: TIM6 clock enabled

Bit 3 **TIM5EN:** TIM5 clock enable

This bit is set and cleared by software.
0: TIM5 clock disabled
1: TIM5 clock enabled

Bit 2 **TIM4EN:** TIM4 clock enable

This bit is set and cleared by software.
0: TIM4 clock disabled
1: TIM4 clock enabled

Bit 1 **TIM3EN:** TIM3 clock enable

This bit is set and cleared by software.
0: TIM3 clock disabled
1: TIM3 clock enabled

Bit 0 **TIM2EN:** TIM2 clock enable

This bit is set and cleared by software.
0: TIM2 clock disabled
1: TIM2 clock enabled

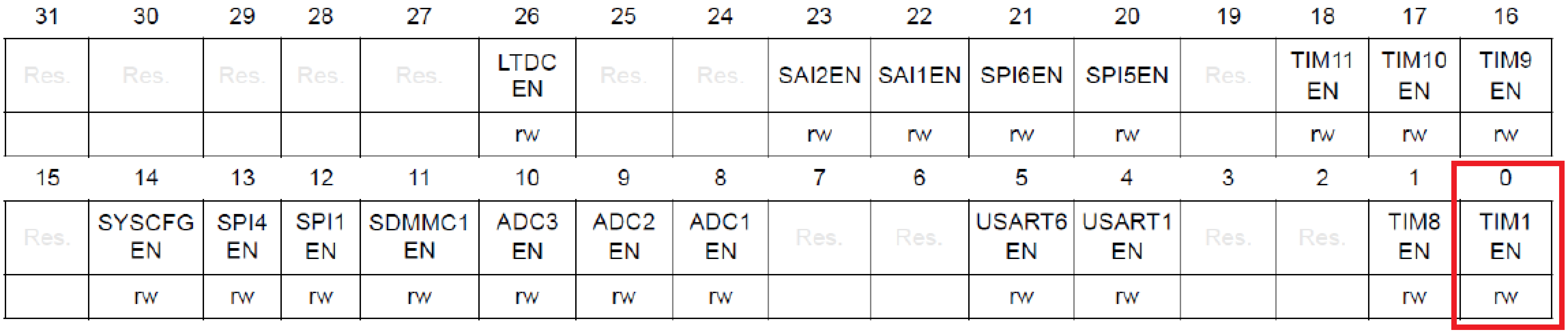
5.3.14 RCC APB2 peripheral clock enable register (RCC_APB2ENR)

Address offset: 0x44

Reset value: 0x0000 0000

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

```
RCC -> APB1ENR |= 0x1;
```



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	LTDC EN	Res.	Res.	SAI2EN	SAI1EN	SPI6EN	SPI5EN	Res.	TIM11 EN	TIM10 EN	TIM9 EN
					rw			rw	rw	rw	rw		rw	rw	rw

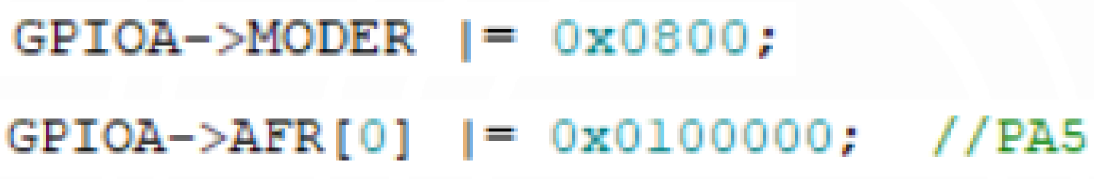
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SYSCFG EN	SPI4 EN	SPI1 EN	SDMMC1 EN	ADC3 EN	ADC2 EN	ADC1 EN	Res.	Res.	USART6 EN	USART1 EN	Res.	Res.	TIM8 EN	TIM1 EN
	rw	rw	rw	rw	rw	rw	rw			rw	rw			rw	rw

Bit 0 **TIM1EN:** TIM1 clock enable

This bit is set and cleared by software.

0: TIM1 clock disabled

1: TIM1 clock enabled



```
GPIOA->MODER |= 0x0800;

GPIOA->AFR[0] |= 0x0100000;  //PA5
```

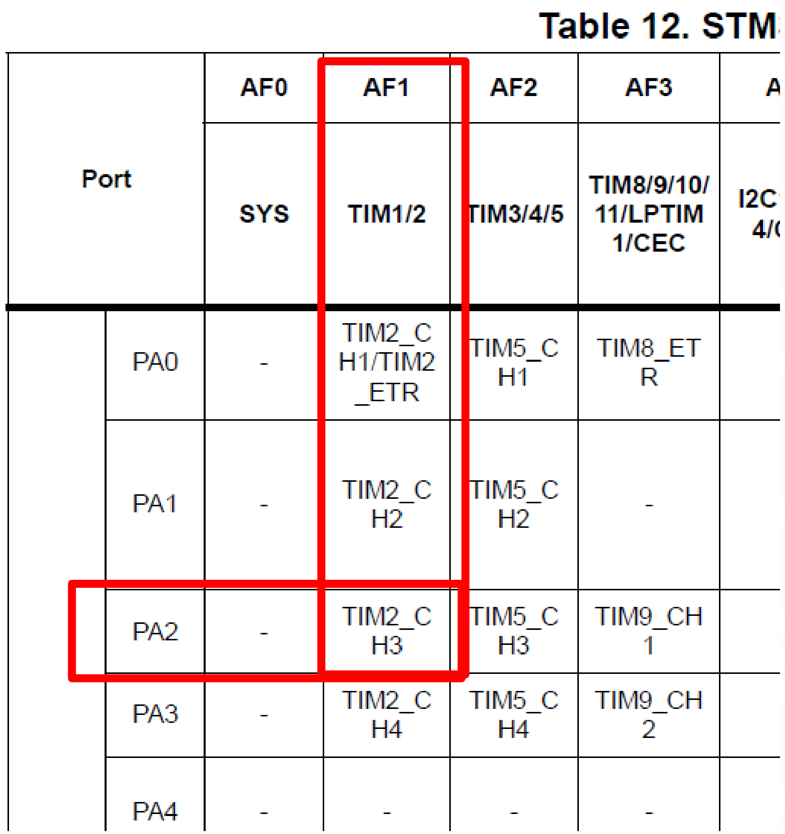
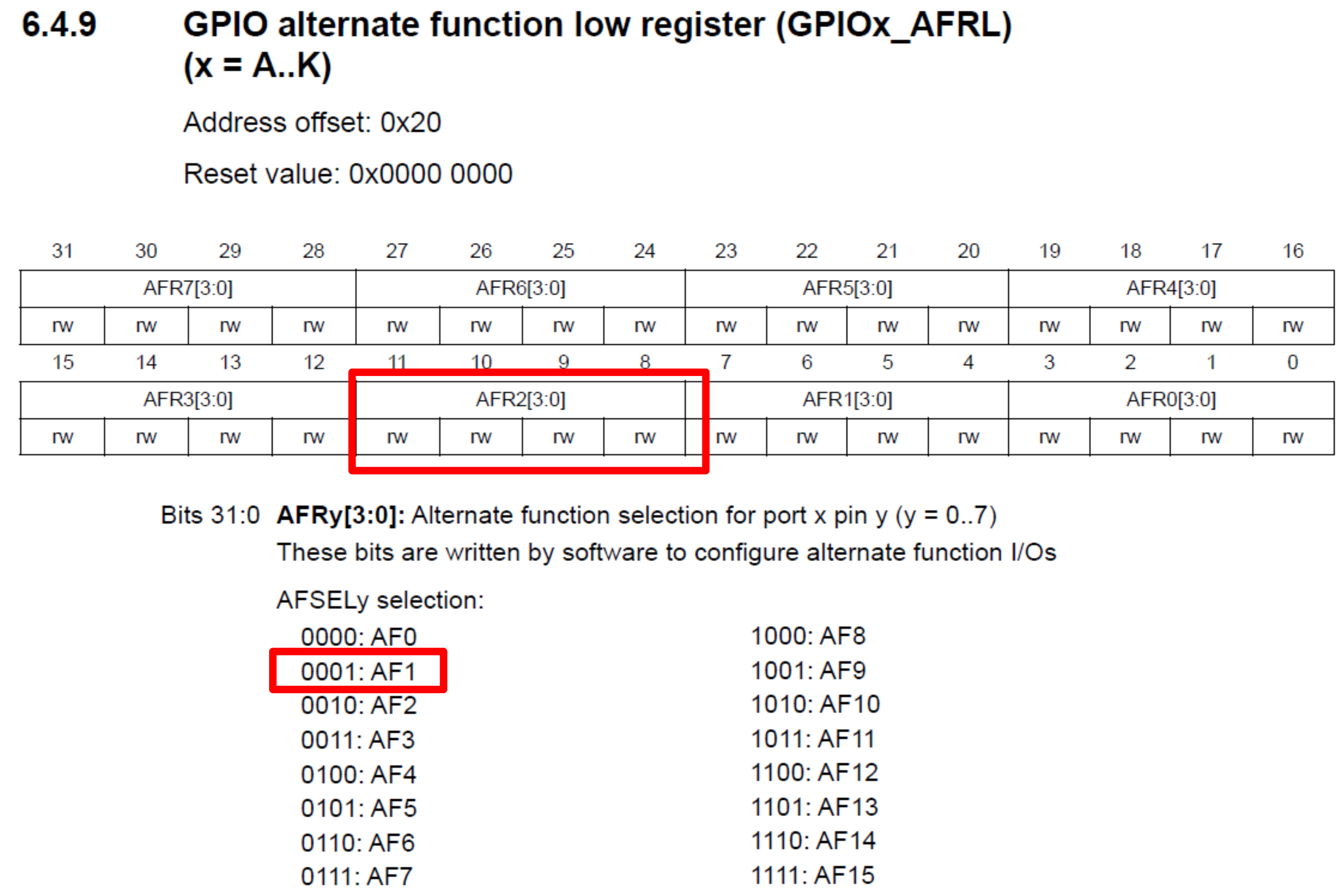


Table 12. STM

Port		AF0	AF1	AF2	AF3	A
		SYS	TIM1/2	TIM3/4/5	TIM8/9/10/ 11/LPTIM 1/CEC	I2C 4/(
	PA0	-	TIM2_C H1/TIM2 _ETR	TIM5_C H1	TIM8_ET R	
	PA1	-	TIM2_C H2	TIM5_C H2	-	
	PA2	-	TIM2_C H3	TIM5_C H3	TIM9_CH 1	
	PA3	-	TIM2_C H4	TIM5_C H4	TIM9_CH 2	
	PA4	-	-	-	-	



6.4.9 GPIO alternate function low register (GPIOx_AFRL) (x = A..K)

Address offset: 0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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

Bits 31:0 **AFRy[3:0]:** Alternate function selection for port x pin y (y = 0..7)

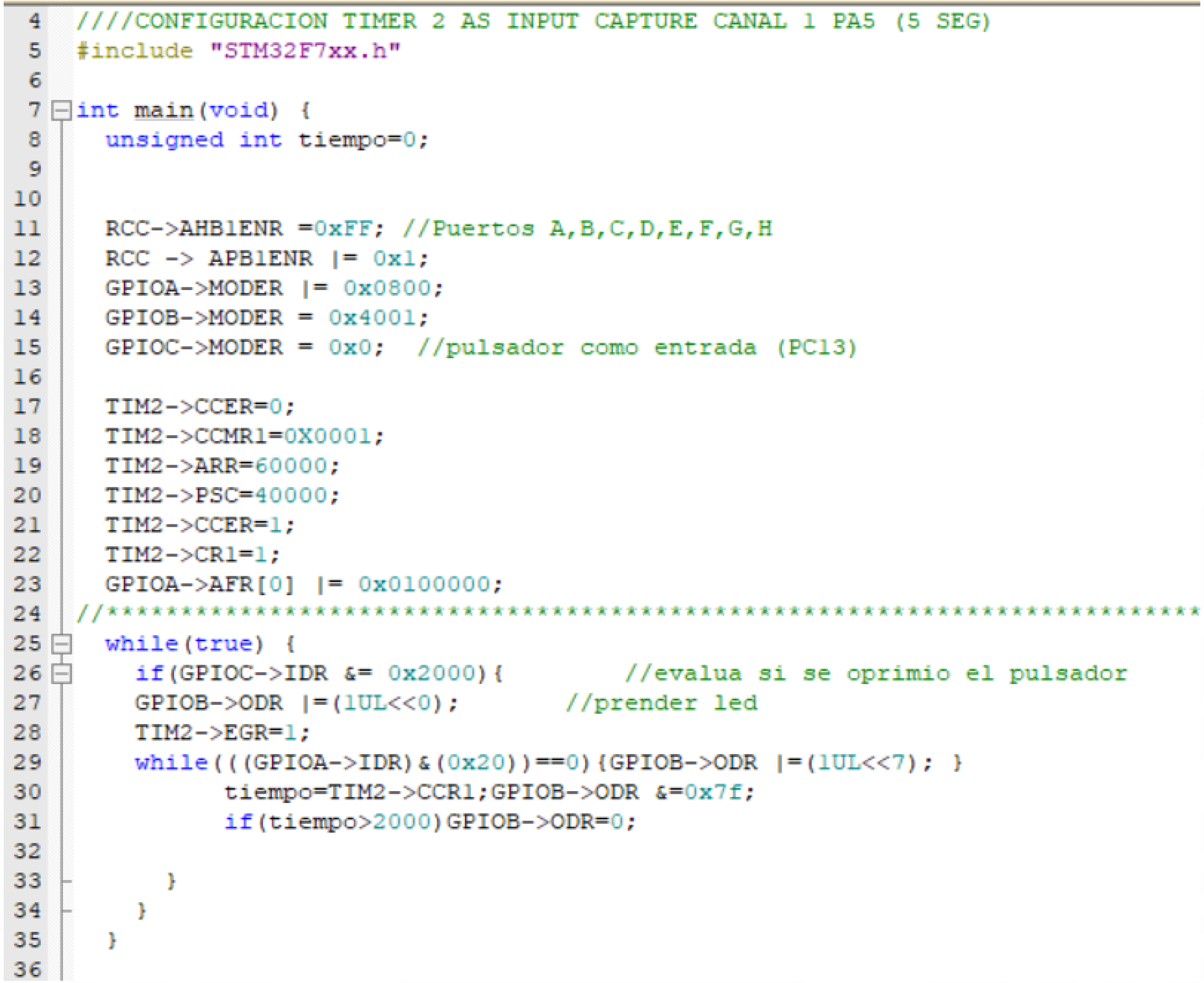
These bits are written by software to configure alternate function I/Os

AFSELy selection:

0000: AF0	1000: AF8
0001: AF1	1001: AF9
0010: AF2	1010: AF10
0011: AF3	1011: AF11
0100: AF4	1100: AF12
0101: AF5	1101: AF13
0110: AF6	1110: AF14
0111: AF7	1111: AF15

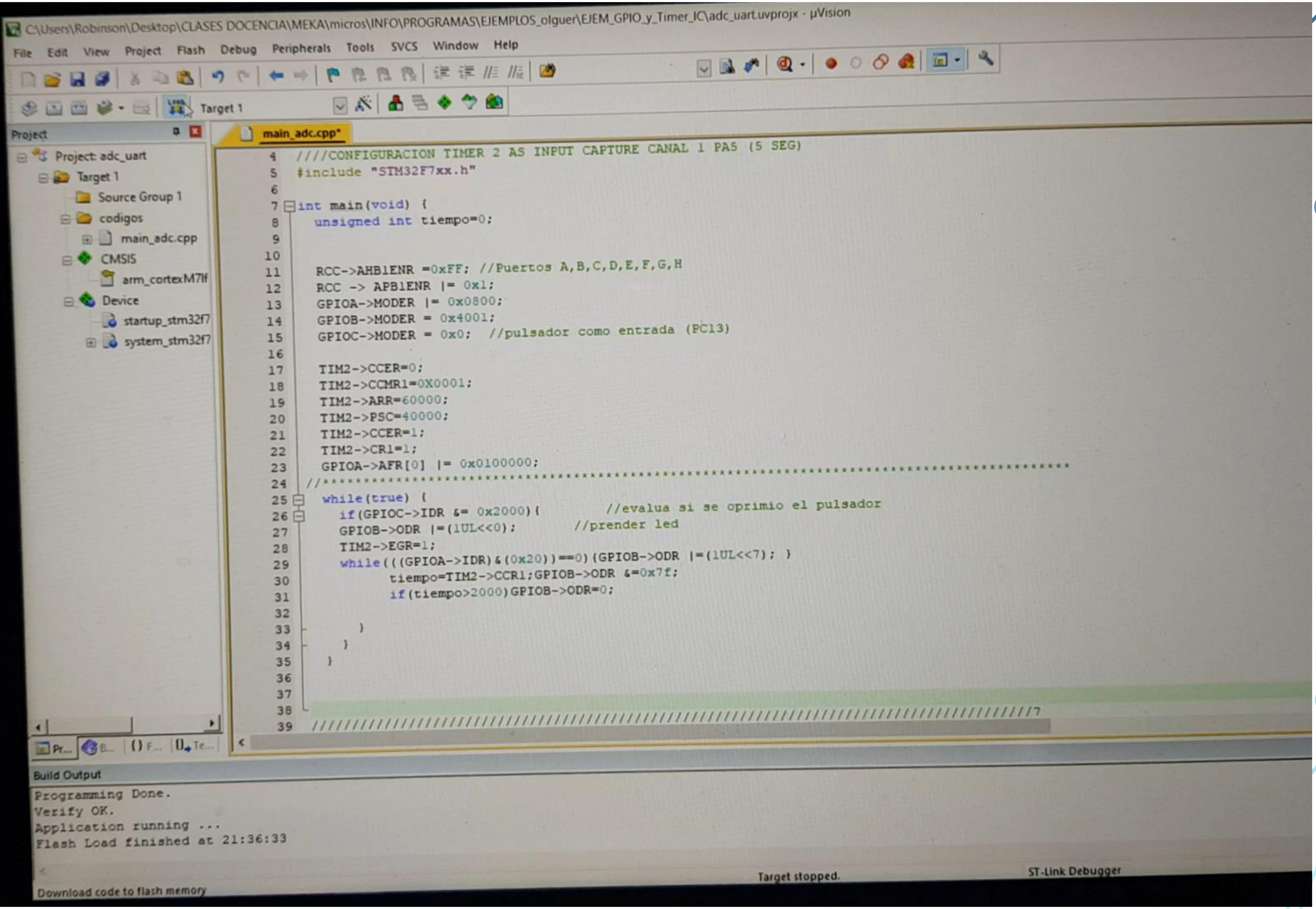
BIN 0 0 0 1 0 0 0 0 0 0 0 0

0X 1 0 0



```c
////CONFIGURACION TIMER 2 AS INPUT CAPTURE CANAL 1 PA5 (5 SEG)
#include "STM32F7xx.h"

int main(void) {
  unsigned int tiempo=0;


  RCC->AHB1ENR =0xFF; //Puertos A,B,C,D,E,F,G,H
  RCC -> APB1ENR |= 0x1;
  GPIOA->MODER |= 0x0800;
  GPIOB->MODER = 0x4001;
  GPIOC->MODER = 0x0;  //pulsador como entrada (PC13)

  TIM2->CCER=0;
  TIM2->CCMR1=0X0001;
  TIM2->ARR=60000;
  TIM2->PSC=40000;
  TIM2->CCER=1;
  TIM2->CR1=1;
  GPIOA->AFR[0] |= 0x0100000;
//***************************************************************************
  while(true) {
    if(GPIOC->IDR &= 0x2000){        //evalua si se oprimio el pulsador
    GPIOB->ODR |=(1UL<<0);       //prender led
    TIM2->EGR=1;
    while(((GPIOA->IDR)&(0x20))==0){GPIOB->ODR |=(1UL<<7); }
          tiempo=TIM2->CCR1;GPIOB->ODR &=0x7f;
          if(tiempo>2000)GPIOB->ODR=0;

      }
    }
  }

```

VIDEO


C:\Users\Robinson\Desktop\CLASES DOCENCIA\MEKA\micros\INFO\PROGRAMAS\EJEMPLOS_olguer\EJEM_GPIO_y_Timer_IC\adc_uart.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Project: adc_uart
Target 1
Source Group 1
codigos
main_adc.cpp
CMSIS
arm_cortexM7lf
Device
startup_stm32f7
system_stm32f7
Pr...
B...
{} F...
0+ Te...
main_adc.cpp*
Build Output
Download code to flash memory
Target stopped.
ST-Link Debugger

```
////CONFIGURACION TIMER 2 AS INPUT CAPTURE CANAL 1 PAS (5 SEG)
#include "STM32F7xx.h"

int main(void) {
  unsigned int tiempo=0;


  RCC->AHB1ENR =0xFF; //Puertos A,B,C,D,E,F,G,H
  RCC -> APB1ENR |= 0x1;
  GPIOA->MODER |= 0x0800;
  GPIOB->MODER = 0x4001;
  GPIOC->MODER = 0x0;  //pulsador como entrada (PC13)

  TIM2->CCER=0;
  TIM2->CCMR1=0X0001;
  TIM2->ARR=60000;
  TIM2->PSC=40000;
  TIM2->CCER=1;
  TIM2->CR1=1;
  GPIOA->AFR[0] |= 0x0100000;
//*************************************************************************************************
  while(true) {
    if(GPIOC->IDR &= 0x2000){         //evalua si se oprimio el pulsador
    GPIOB->ODR |=(1UL<<0);          //prender led
    TIM2->EGR=1;
    while(((GPIOA->IDR)&(0x20))==0){GPIOB->ODR |=(1UL<<7); }
          tiempo=TIM2->CCR1;GPIOB->ODR &=0x7f;
          if(tiempo>2000)GPIOB->ODR=0;


        }
      }
  }


///////////////////////////////////////////////////////////////////////////////////////////7
/////////////////////////////////////////////////////////////////////////////////////////////
```

```
Programming Done.
Verify OK.
Application running ...
Flash Load finished at 21:36:33
```

INGENIERÍA MECATRÓNICA UMNG

EJERCICIO EN CLASE:

1. Modificar el programa anterior para que opere por el canal 2 (5seg) y 4 (10 seg).

2. Medir la duración de un pulso de entrada mediante el modulo de IC y visualizar el tiempo en segundos por LCD alfanumérica.

3. Realizar un programa que basado en modo IC determine la frecuencia de una señal de entrada.

UNIVERSIDAD MILITAR NUEVA GRANADA