

MICROS 32 BITS STM - OC

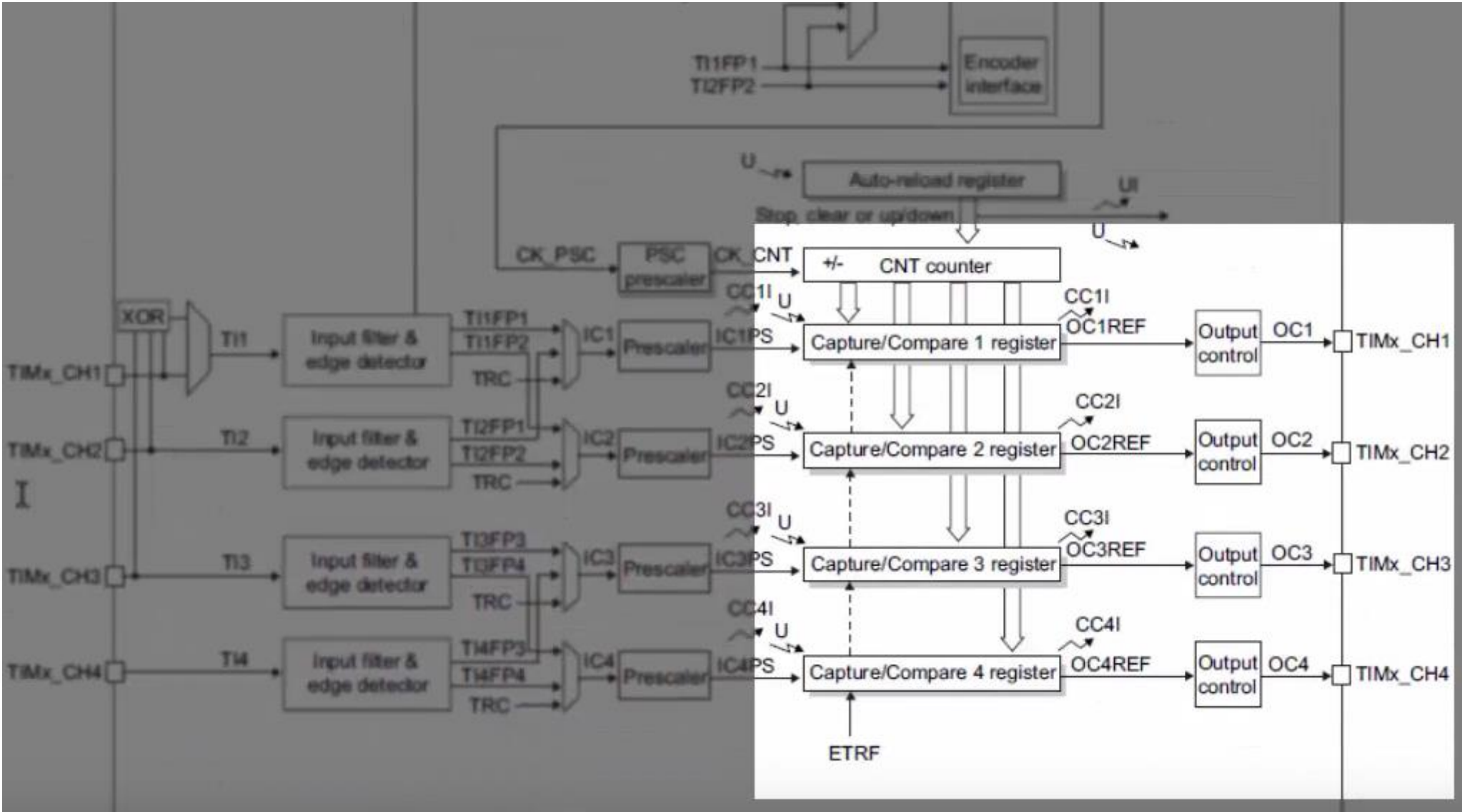
ROBINSON JIMENEZ MORENO



UNIVERSIDAD MILITAR
NUEVA GRANADA



TIMER - OUTPUT COMPARE



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 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, you 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

[Figure 236](#).

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.

23.4.7

TIMx capture/compare mode register 1 (TIMx_CCMR1)/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

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	1	1	0	0	0	0

BIN 0 0 1 1 0 0 0 0

Bits 6:4 **OC1M**: Output compare 1 mode

0X

3

0

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.

0000: Frozen - The comparison between the output compare register TIMx_CCR1 and the counter TIMx_CNT has no effect on the outputs.(this mode is used to generate a timing base).

0001: 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).

0010: 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).

0011: Toggle - OC1REF toggles when TIMx_CNT=TIMx_CCR1.

0100: Force inactive level - OC1REF is forced low.

0101: Force active level - OC1REF is forced high.

Bit 3 **OC1PE**: 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).

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.

Bit 2 **OC1FE**: 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.

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).

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 3 **CC1NP**: Capture/Compare 1 output Polarity.

CC1 channel configured as output: CC1NP must be kept cleared in this case.

CC1 channel configured as input: This bit is used in conjunction with CC1P to define TI1FP1/TI2FP1 polarity. refer to CC1P description.

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

CC1 channel configured as output:

0: OC1 active high

1: OC1 active low

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

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)

Address offset: 0x28

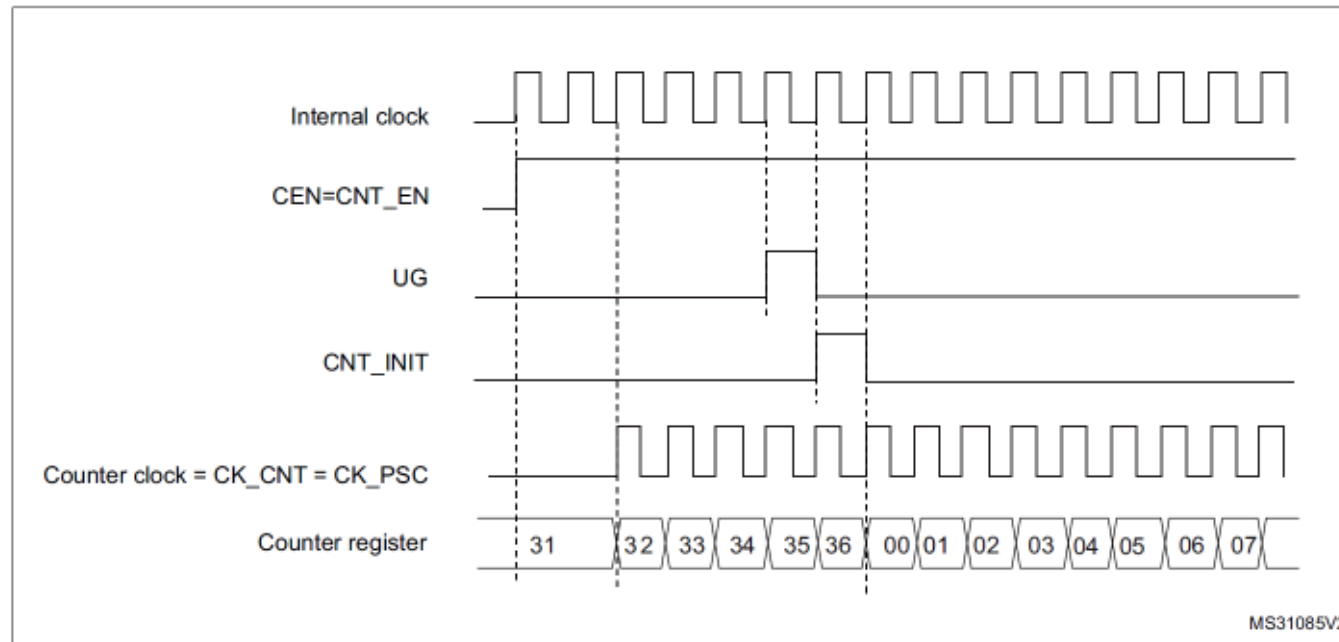
Reset value: 0x0000

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.

23.4.5 TIMx status register (TIMx_SR)

Address offset: 0x10

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	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 1 **CC1IF**: 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) and in retriggeable one pulse mode. It is cleared by software.

0: No match.

1: The content of the counter TIMx_CNT has matched 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).

Bit 4 **CC4IF**: Capture/Compare 4 interrupt flag

Refer to CC1IF description

Bit 3 **CC3IF**: Capture/Compare 3 interrupt flag

Refer to CC1IF description

Bit 2 **CC2IF**: Capture/Compare 2 interrupt flag

Refer to CC1IF description

Bit 0 **UIF**: 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:

23.4.12 TIMx auto-reload register (TIMx_ARR)

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

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.

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.

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

Table 12. STM32F745xx a

Port		AF0	AF1	AF2	AF3	AF4
		SYS	TIM1/2	TIM3/4/5	TIM8/9/10/ 11/LPTIM 1/CEC	I2C1/2/3/ 4/CEC
Port B	PB9	-	-	TIM4_C H4	TIM11_CH 1	I2C1_SD A
	PB10	-	TIM2_C H3	-	-	I2C2_SC L
	PB11	-	TIM2_C H4	-	-	I2C2_SD A
	PB12	-	TIM1_B KIN	-	-	I2C2_SM BA
	PB13	-	TIM1_C H1N	-	-	-

6.4.10 GPIO alternate function high register (GPIOx_AFRH) (x = A..J)

Address offset: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR15[3:0]				AFR14[3:0]				AFR13[3:0]				AFR12[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
AFR11[3:0]				AFR10[3:0]				AFR9[3:0]				AFR8[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 = 8..15)

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

AFSELy selection:

0000: AF0

0001: AF1

0010: AF2

0011: AF3

0100: AF4

0101: AF5

0110: AF6

1000: AF8

1001: AF9

1010: AF10

1011: AF11

1100: AF12

1101: AF13

1110: AF14

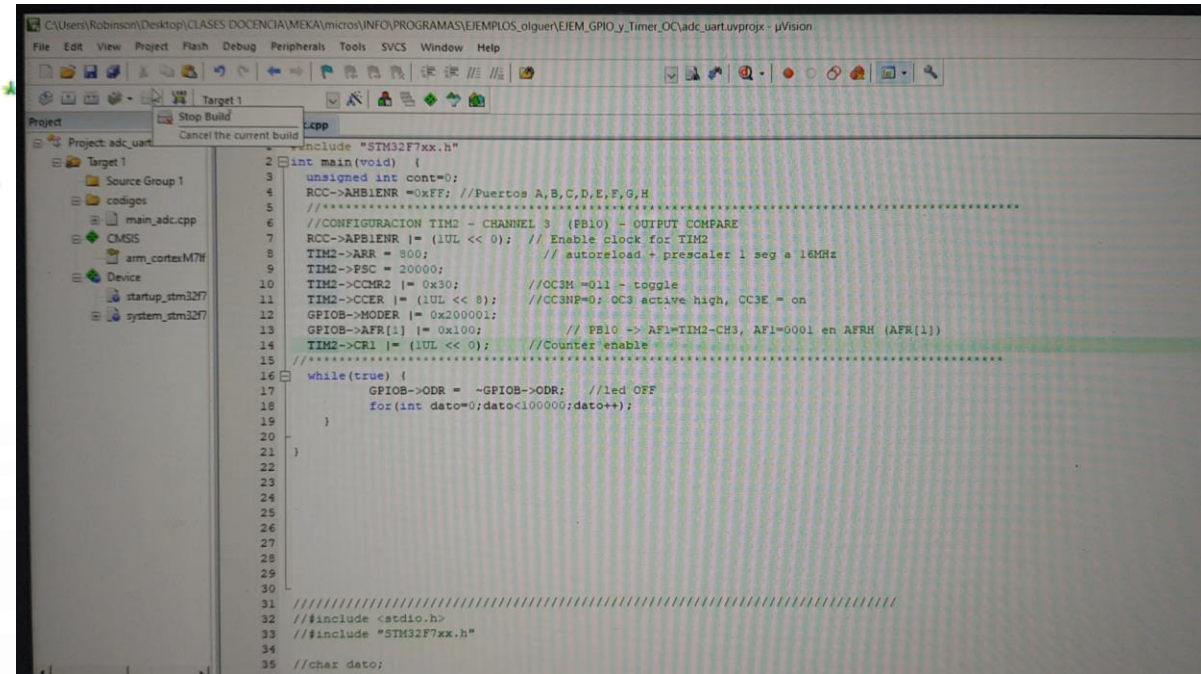
BIN 000100000000
OX 100


```
#include "STM32F7xx.h"

int main(void) {
    unsigned int cont=0;
    RCC->AHB1ENR =0xFF; //Puertos A,B,C,D,E,F,G,H
    //*****
    //CONFIGURACION TIM2 - CHANNEL 3 (PB10) - OUTPUT COMPARE
    RCC->APB1ENR |= (1UL << 0); // Enable clock for TIM2
    TIM2->ARR = 800; // autoreload + prescaler 1 seg a 16MHz
    TIM2->PSC = 20000;
    TIM2->CCMR2 |= 0x30; //OC3M =011 - toggle
    TIM2->CCER |= (1UL << 8); //CC3NP=0; OC3 active high, CC3E = on
    GPIOB->MODER |= 0x200001;
    GPIOB->AFR[1] |= 0x100; // PB10 -> AF1=TIM2-CH3, AF1=0001 en AFRH (AFR[1])
    TIM2->CR1 |= (1UL << 0); //Counter enable
    TIM2->CCR1 =1000;
    //*****

    while(true) {
        GPIOB->ODR = ~GPIOB->ODR; //led OFF
        for(int dato=0;dato<100000;dato++);
    }
}
```

QUE OCURRE SI EL REGISTRO CCR_x ES CERO ?

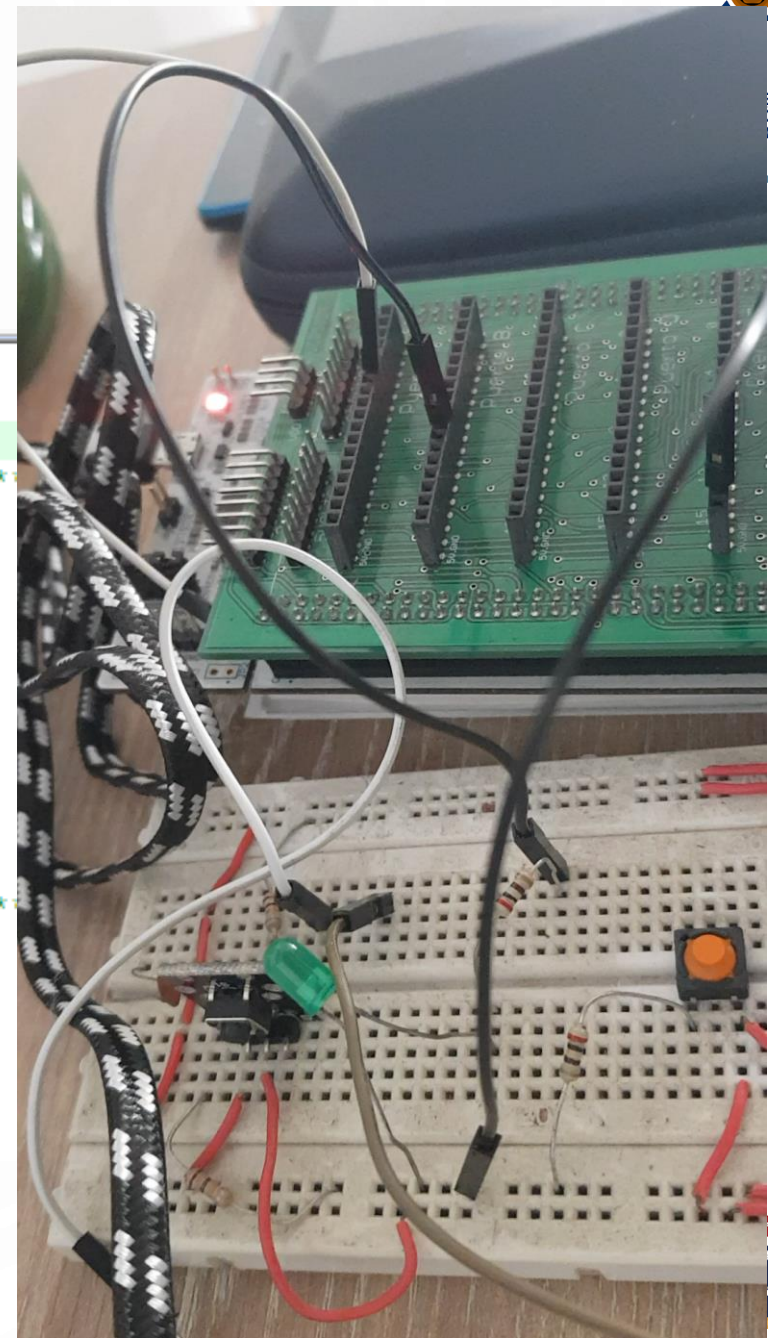


```
1  #include "STM32F7xx.h"
2
3  int main(void) {
4      unsigned int cont=0;
5      RCC->AHB1ENR =0xFF; //Puertos A,B,C,D,E,F,G,H
6      //*****
7      //CONFIGURACION TIM2 - CHANNEL 3 (PB10) - OUTPUT COMPARE
8      RCC->APB1ENR |= (1UL << 0); // Enable clock for TIM2
9      TIM2->ARR = 800; // autoreload + prescaler 1 seg a 16MHz
10     TIM2->PSC = 20000;
11     TIM2->CCMR2 |= 0x30; //OC3M =011 - toggle
12     TIM2->CCER |= (1UL << 8); //CC3NP=0; OC3 active high, CC3E = on
13     GPIOB->MODER |= 0x200001;
14     GPIOB->AFR[1] |= 0x100; // PB10 -> AF1=TIM2-CH3, AF1=0001 en AFRH (AFR[1])
15     TIM2->CR1 |= (1UL << 0); //Counter enable
16     //*****
17     while(true) {
18         GPIOB->ODR = ~GPIOB->ODR; //led OFF
19         for(int dato=0;dato<100000;dato++);
20     }
21 }
22
23
24
25
26
27
28
29
30
31 ///////////////////////////////////////////////////
32 //include <stdio.h>
33 //include "STM32F7xx.h"
34
35 //char dato;
```

```

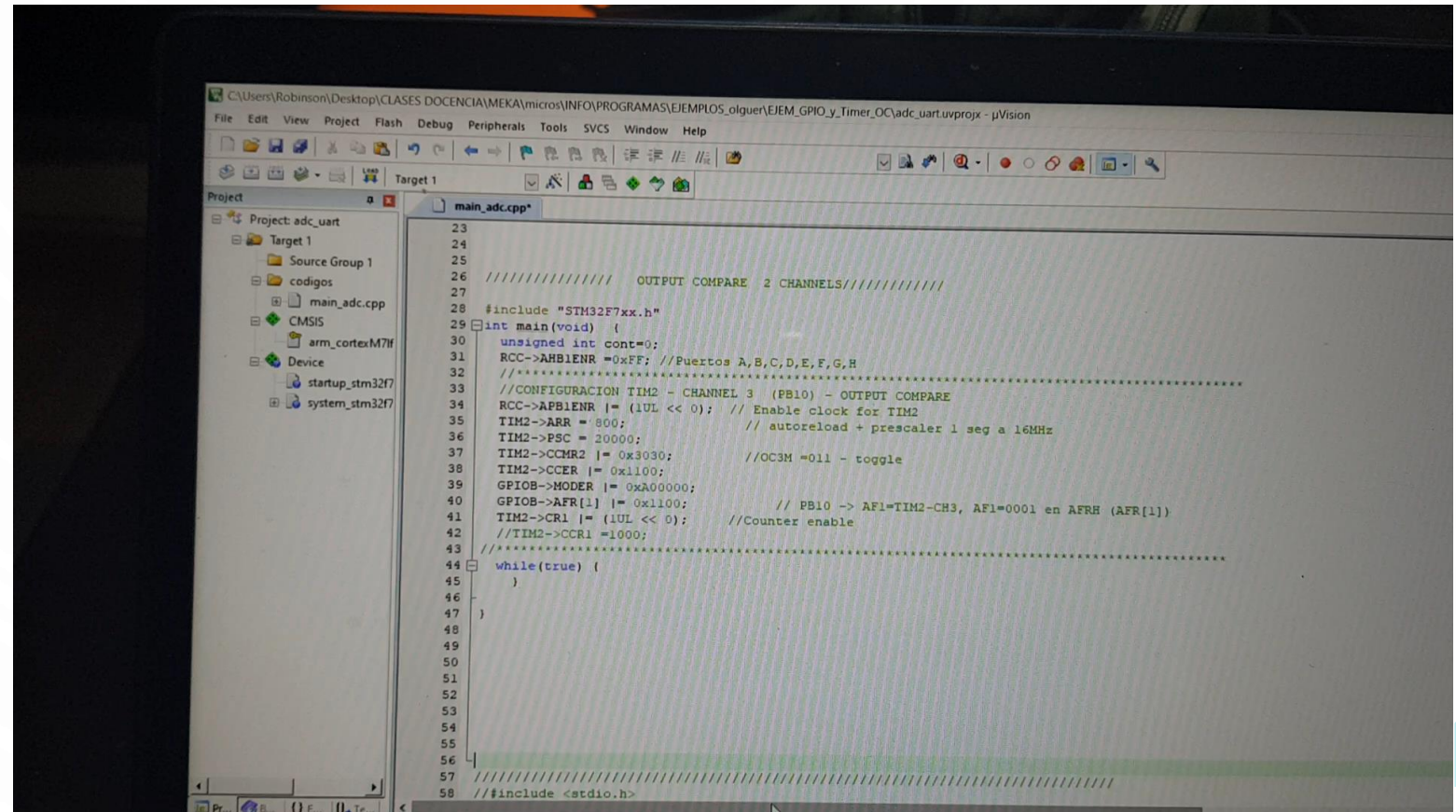
1  #include "STM32F7xx.h"
2  int main(void) {
3      RCC->AHB1ENR = 0x0F; //Puertos A,B,C,D,E,F,G,H
4      //*****
5      //CONFIGURACION TIM2 - CHANNEL 3 (PB10) - OUTPUT COMPARE
6      RCC->APB1ENR |= 1; // Enable clock for TIM2
7      TIM2->ARR = 800; // autoreload + prescaler 1 seg a 16MHz
8      TIM2->PSC = 20000;
9      TIM2->CCMR2 |= 0x30; //OC3M =011 - toggle
10     TIM2->CCER |= (1UL << 8); //CC3NP=0; OC3 active high, CC3E = on
11     GPIOB->MODER |= 0x200001;
12     GPIOB->AFR[1] |= 0x100; // PB10 -> AF1=TIM2-CH3, AF1=0001 en AFRH (AFR[1])
13     TIM2->CR1 |= (1UL << 0); //Counter enable
14     TIM2->CCR3 = 0;
15     //*****
16     while(true) {
17         //         GPIOB->ODR = ~GPIOB->ODR; //led OFF
18         //         for(int dato=0;dato<100000;dato++);
19     }
20
21 }

```



VIDEOS OC 1 CANAL Y 2 CANALES

Ejercicio: realizar el ajuste del código anterior para tener dos canales OC operando cada uno un led, conmute el segundo led al doble de frecuencia que el actual.



```
23
24
25
26 ////////////////////////////////////////////////// OUTPUT COMPARE 2 CHANNELS/////////////////////////////////
27
28 #include "STM32F7xx.h"
29 int main(void) {
30     unsigned int cont=0;
31     RCC->AHB1ENR = 0xFF; //Puertos A,B,C,D,E,F,G,H
32     //////////////////////////////////////////////////
33     //CONFIGURACION TIM2 - CHANNEL 3 (PB10) - OUTPUT COMPARE
34     RCC->APB1ENR |= (1UL << 0); // Enable clock for TIM2
35     TIM2->ARR = 800; // autoreload + prescaler 1 seg a 16MHz
36     TIM2->PSC = 20000;
37     TIM2->CCMR2 |= 0x3030; //OC3M =011 - toggle
38     TIM2->CCER |= 0x1100;
39     GPIOB->MODER |= 0xA000000;
40     GPIOB->AFR[1] |= 0x1100; // PB10 -> AF1=TIM2-CH3, AF1=0001 en AFRH (AFR[1])
41     TIM2->CR1 |= (1UL << 0); //Counter enable
42     //TIM2->CCR1 =1000;
43     //////////////////////////////////////////////////
44     while(true) {
45     }
46
47 }
48
49
50
51
52
53
54
55
56
57 //////////////////////////////////////////////////
58 //include <stdio.h>
```

Interrupción OC

```
#include "stm32f7xx.h"
#define WORK GPIOB -> ODR ^= 0x1;

int main(void) {

    RCC->AHB1ENR = 0x0F; //PORTS A,B,C,D,E,F,G,H
    GPIOB->MODER |= 0x201;
    GPIOB->AFR[0] |= 0x20000; // PB4 -> AF0=TIM3-CH1, AF1=0002 en AFRL (AFR[0])
    //CONFIGURACION TIM3 - CHANNEL 1 (PB4) - OUTPUT COMPARE
    RCC->APB1ENR |= 2; // Enable clock for TIM3
    TIM3->ARR = 3200; // autoreload + prescaler 1 seg a 16MHz
    TIM3->PSC = 20000;
    TIM3->CCMR1 |= 0x30; //OC1M =011 - toggle
    TIM3->CCER |= (1UL << 0); //CC1NP=0; OC1 active high, CC1E = on
    TIM3->CR1 |= (1UL << 0); //Counter enable
    TIM3->DIER = 1UL << 1;
    TIM3->CCR1 = 800;
    NVIC_EnableIRQ(TIM3_IRQn);
    //*****
    while(true) {
        }
    }

extern "C" {
    void TIM3_IRQHandler(void) {
        TIM3->SR = 0;
        WORK
        TIM3->EGR=1; //RESET TIM2
    }
}
```


EJERCICIO EN CLASE

- 1- Realizar un programa para STM que genere una señal de intermitencia de un LED a 1 seg, 2 seg y 3 segundos, consecutivamente mediante modulo OC, para 8 leds.
- 2-Generar una señal de salida OC que varié su frecuencia acorde a una entrada analógica por lectura de un potenciómetro, evidencie la frecuencia en una LCD alfanumérica.

TAREA

Implementar un Sistema tipo sonar basado en módulos IC, OC y PWM de la STM, en el cual mediante un servomotor que tenga acoplado un sensor de distancia tipo HCSR-04 visualice en una LCD grafica la distancia a la que se encuentran objetos en el rango de giro del servomotor, como se ilustra. El HCSR-04 debe comandarse por los módulos de IC y OC y el servomotor por PWM.

