CIRCUITOS DIGITALES Y MICROCONTROLADORES 2022

Facultad de Ingeniería UNLP

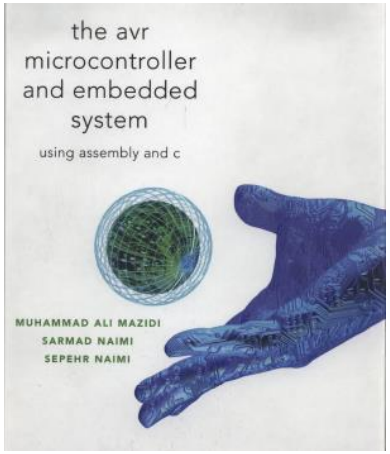
PERIFERICO TIMER (AVR)

Ing. José Juárez

Abril 2021

Bibliografía:

- *The AVR microcontroller & Embedded Systems*. Mazidi,Naimis, CH9



- Libro Digital de Felipe Espinosa, CH4

- Hojas de datos ATMEGA 328p, 2560
- tutorial: Timer arduino

Introducción

- Una de las características mas destacables de un MCU es la capacidad de realizar tareas temporizadas.

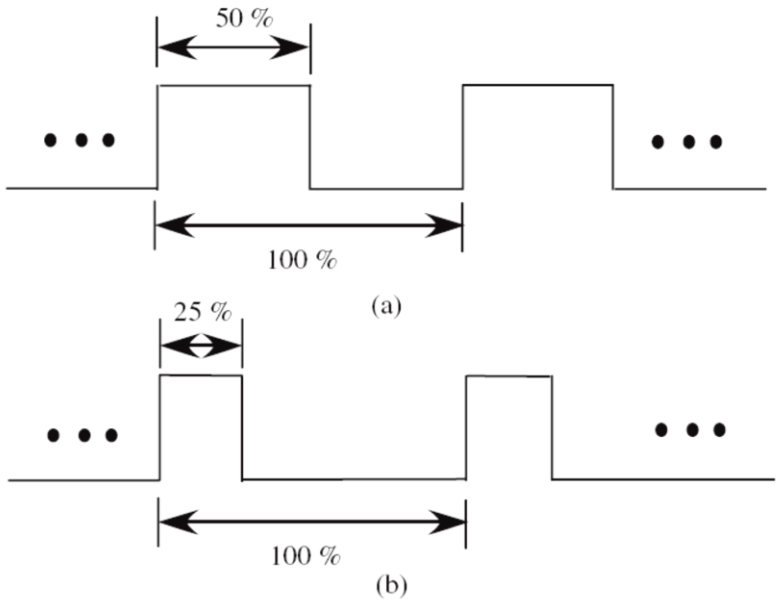
- Para esto cuentan con un periférico TIMER o TEMPORIZADOR

- Algunas de las aplicaciones pueden ser:
  - Generación de retardos
  - Interrupción periódica de tiempo-real (planificación de tareas)
  - Protección Watch-Dog

- Pero además un Timer se puede utilizar para:
  - Generación de señales digitales con frecuencia variables o ciclo de trabajo variable (PWM)
  - Medición de frecuencia y ancho de pulso
  - Registro y conteo de eventos (COUNTER)

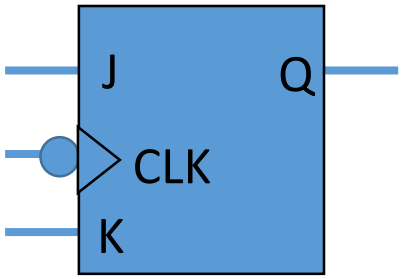
Periodo, Frecuencia y Ciclo de Trabajo

- El periodo T de una señal real x(t), es el menor numero entero que satisface: *x(t)=x(t+T)*
- La frecuencia se define como el numero de oscilaciones en el lapso de 1 seg. Es decir: *f=1/T*
- El ciclo de trabajo de una señal digital es el porcentaje de tiempo en que la misma está activa respecto del periodo total: *(ancho de pulso*100%)/T*



Sistema básico de temporización

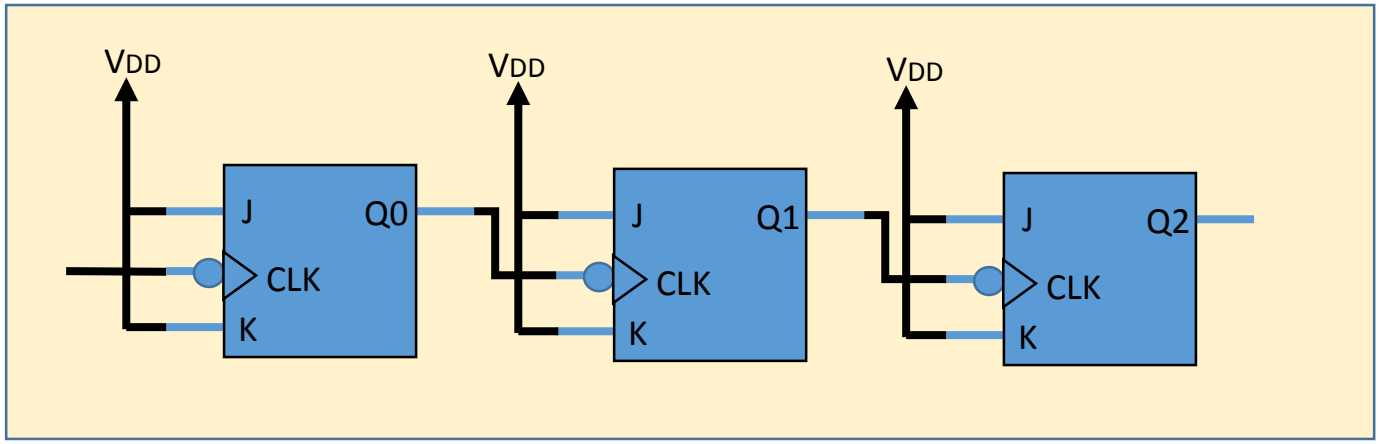
Flip flop J-K

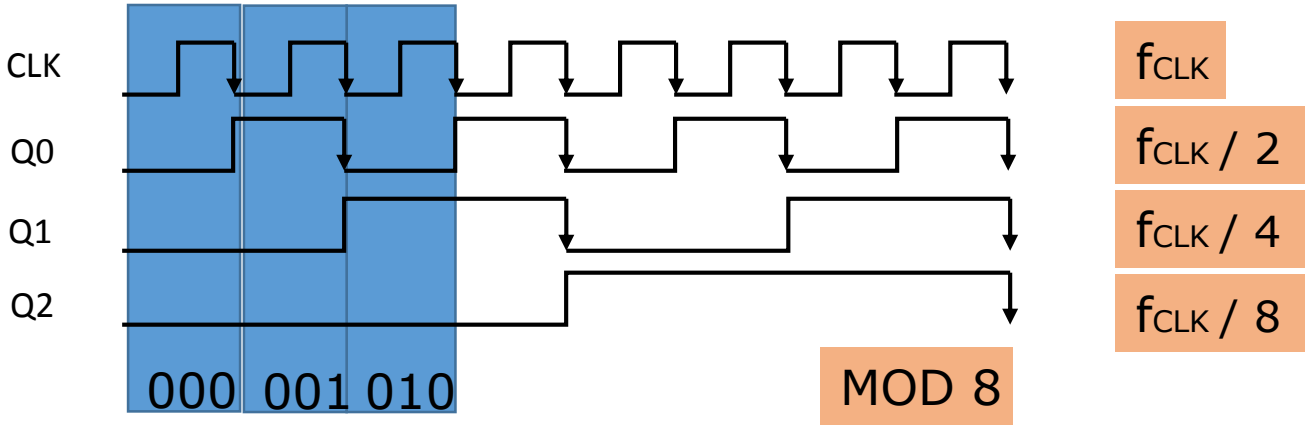


J	K	CLK	Q
0	0	↓	Sin cambio
0	1	↓	0
1	0	↓	1
1	1	↓	toggle

Sistema básico de temporización

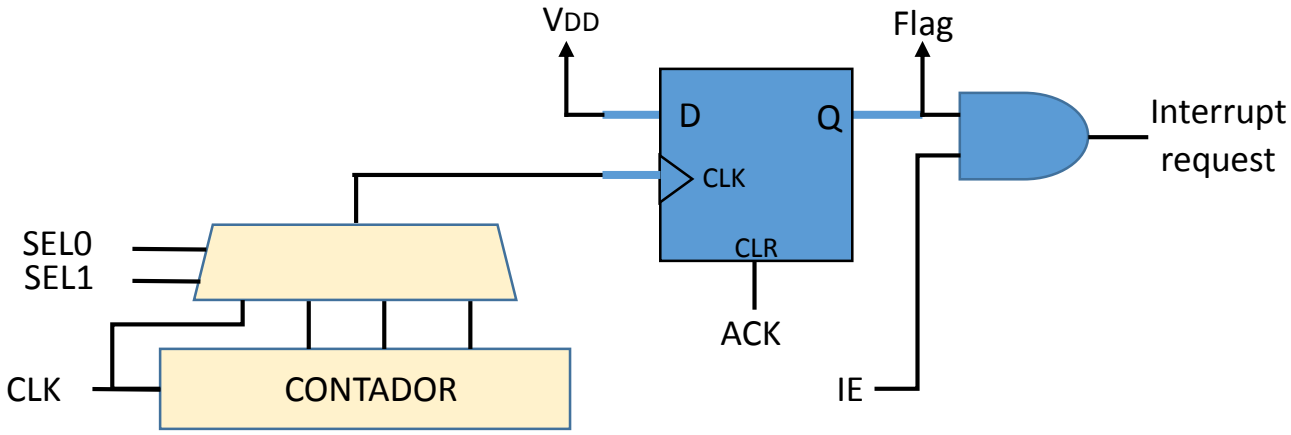
- Contador asincrónico de Rizo con 3 FF





Sistema básico de temporización

- Generador de Interrupciones periódicas



SEL1	SEL0	Frecuencia de interrupción
0	0	$f_{CLK}/1$
0	1	$f_{CLK}/2$
1	0	$f_{CLK}/4$
1	1	$f_{CLK}/8$

Sistema básico de temporización

Un TIMER/COUNTER cuenta pulsos de reloj de una señal digital de CLK por lo tanto el tiempo es discreto (solo se puede ejecutar una acción en un flanco de la señal de CLK).

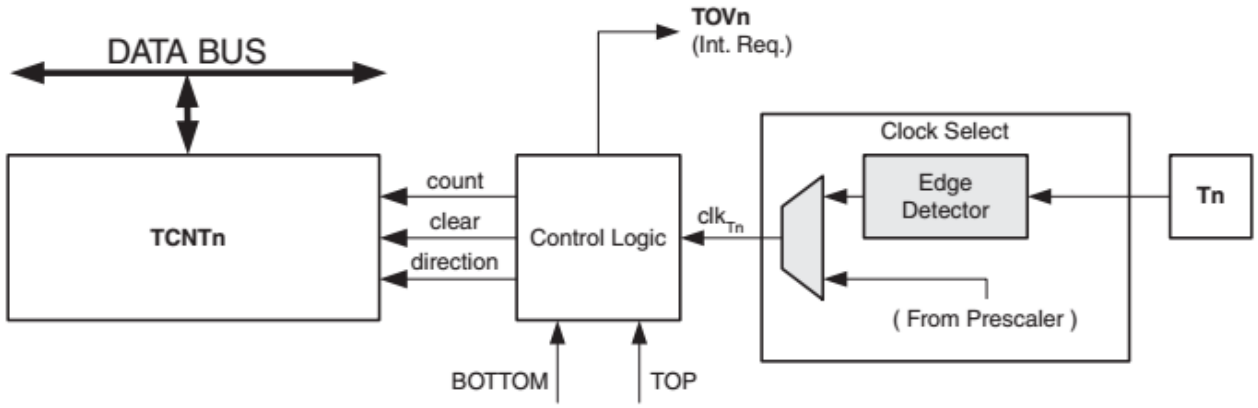
Vamos a definir algunos parámetros asociados a este problema y que vamos a utilizar después:

- **Resolución:** es el mínimo período de tiempo medible o contable y es 1 pulso de CLK o período de CLK (depende de la fuente de reloj y de la selección de los predivisores)
- **Rango:** es el rango de valores (Máx - Mín) que se utiliza para representar la información.
- **Precisión:** con cuantos bits puedo representar la información. Básicamente es el Número del bits del Timer.
- **Exactitud:** es cuanto difiere el “valor real” respecto al “valor medido”, depende de la exactitud del oscilador que genera la señal del reloj.
- **Estabilidad:** es un medida de cuan estable es la frecuencia del CLK frente a perturbaciones en la tensión de alimentación, en la temperatura y al envejecimiento de los componentes. Puede dividirse en estabilidad de corto término y estabilidad a largo plazo.

Temporizadores en AVR

Timer 0	Timer 1	Timer 2
- 8-bit timer/counter	- 16-bit timer/counter	- 8-bit timer/counter
- 10-bit clock prescaler	- 10-bit clock prescaler	- 10-bit clock prescaler
- Functions:	- Functions:	- Functions:
– Pulse width modulation	– Pulse width modulation	– Pulse width modulation
– Frequency generation	– Frequency generation	– Frequency generation
– Event counter	– Event counter	– Event counter
– Output compare	– Output compare – 2 ch	– Output compare
- Modes of operation:	– Input capture	- Modes of operation:
– Normal	- Modes of operation:	– Normal
– Clear timer on compare match (CTC)	– Normal	– Clear timer on compare match (CTC)
– Fast PWM	– Clear timer on compare match (CTC)	– Fast PWM
– Phase correct PWM	– Fast PWM	– Phase correct PWM
	– Phase correct PWM	

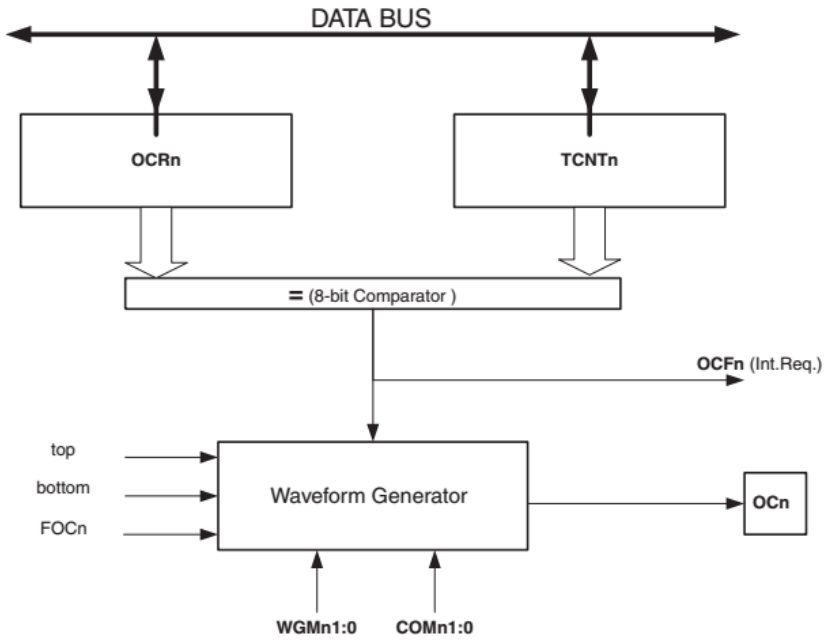
Sistema básico de temporización



- El corazón del sistema es un registro contador, llamado TCNT “Timer Counter”
- El contador suma 1 (o resta 1 según dirección de conteo) cada vez que en su entrada hay un flanco del reloj (count).
- La señal de reloj puede prevenir desde el Prescaler o por terminal externo (pin Tn)
  - Si contamos pulsos del reloj del MCU tenemos una base de tiempo con prestaciones conocidas.
  - Si contamos pulsos externos de una señal “desconocida”, estamos contando cuantas veces ocurrió un flanco en el terminal Tn.
- La capacidad de conteo depende del numero de bits del contador, por ejemplo con 16 bits puede contar hasta 65536 y cuando llega al máximo valor 0xFFFF se reinicia (clear) automáticamente generando un aviso de desborde u overflow (flag TOVn).
- Bottom es el valor incial (por defecto 0)
- TOP es el valor final por lo tanto el contador es MOD=TOP+1

Timer con Generación de señales

- El sistema se basa en comparar el valor actual del contador TCNT contra un registro que contiene un Valor almacenado (OCRn)

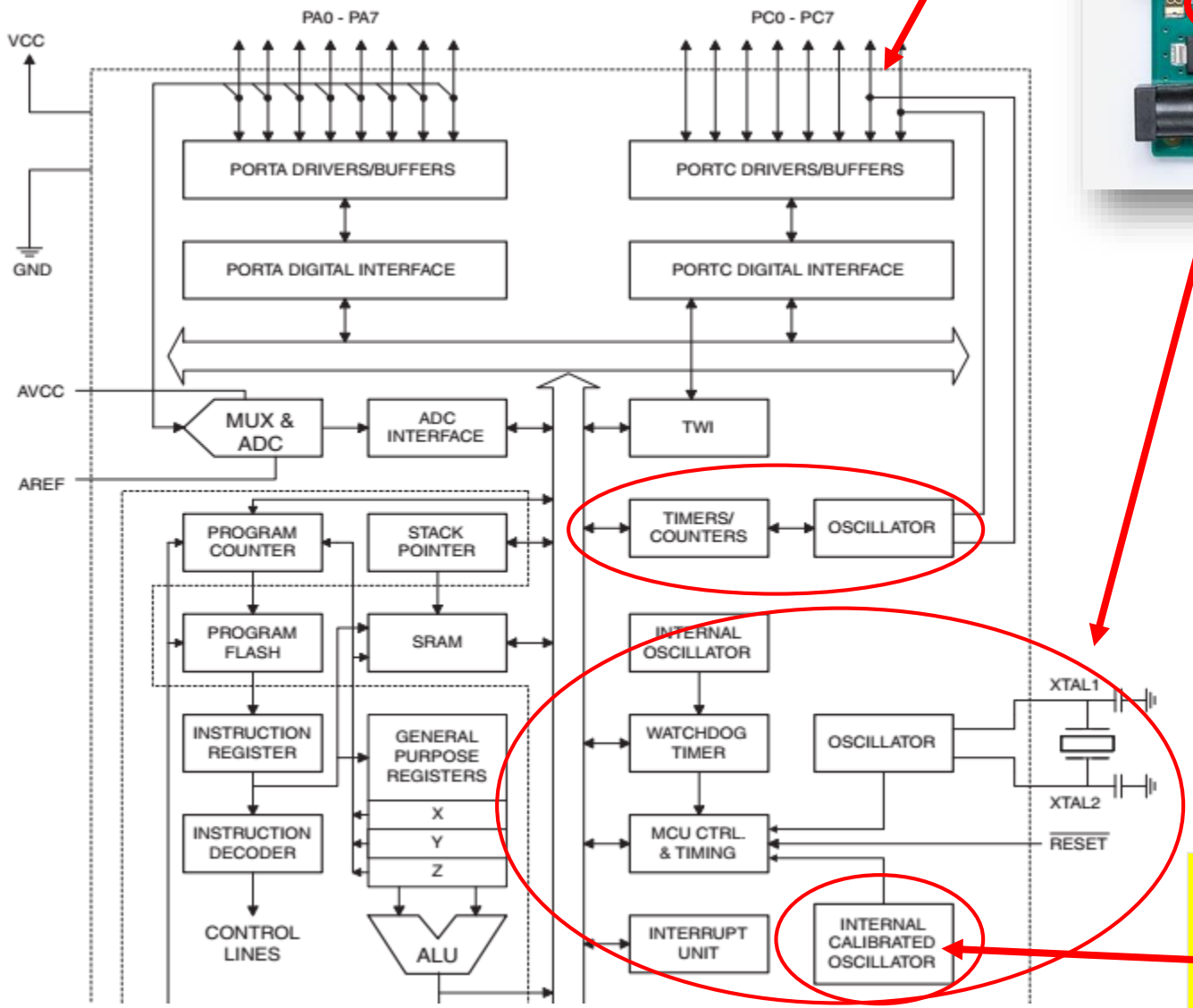


- Cuando el TCNT iguala al valor contenido en el registro de comparación, se genera un aviso interno (flag o interrupción) para que el programa tome alguna acción.
- Además, con un circuito especial (Waveform Generator) se puede modificar el estado del terminal Ocn cuando se active el flag.
  - Aplicaciones: generación precisa de retardos, generación de señales digitales de frecuencia variable o PWM.

Temporizadores en AVR

XTAL 32.768KHz
Timer2 (RTC)

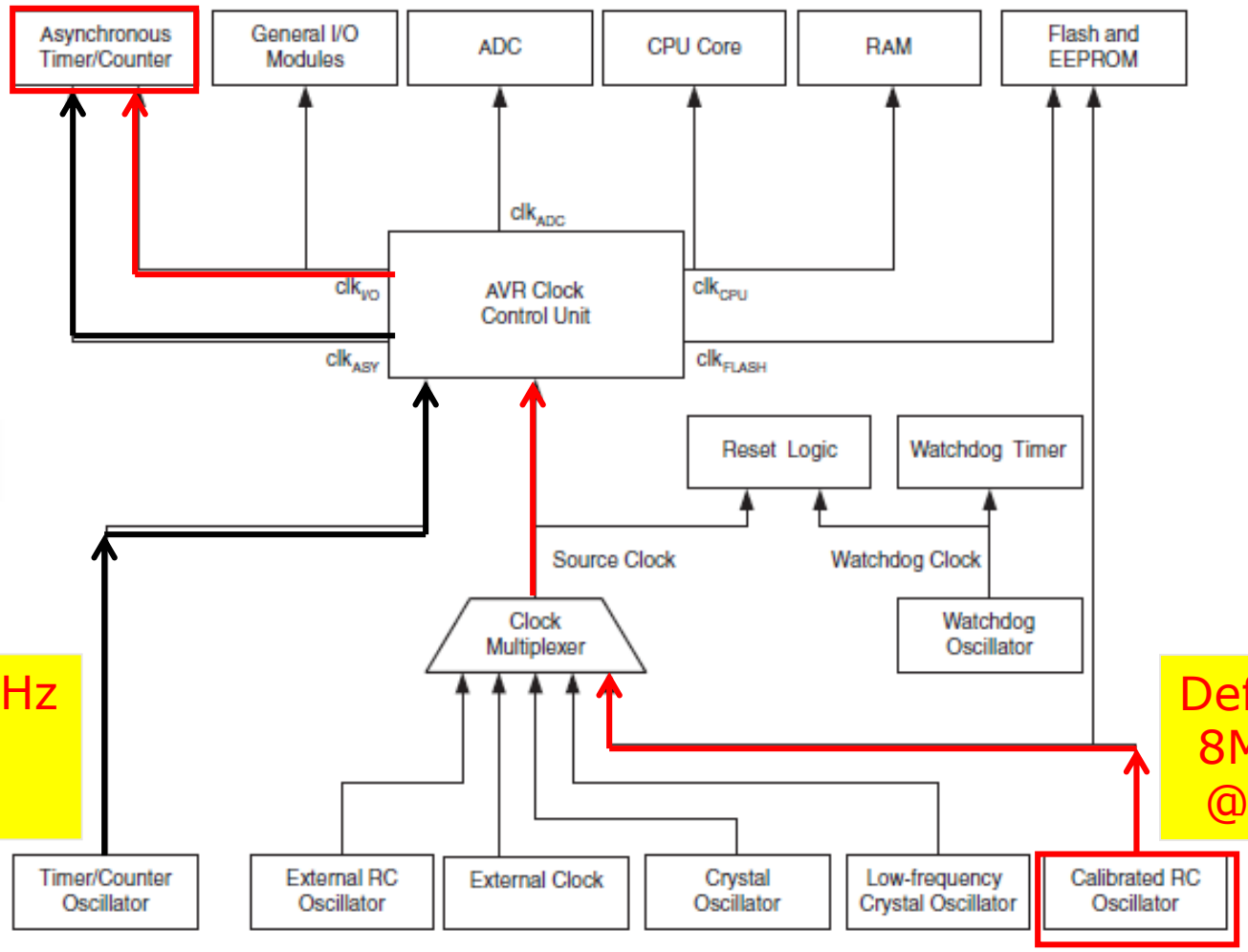
XTAL 16MHz
+-100ppm



Default interno:
8MHz ±1%
@5V, 25°C

Fuentes de reloj para los Temporizadores

XTAL 32.768KHz
Timer2 (RTC)
+-100ppm



Default:
8MHz ±1%
@5V, 25°c

Fuentes de reloj para los Temporizadores

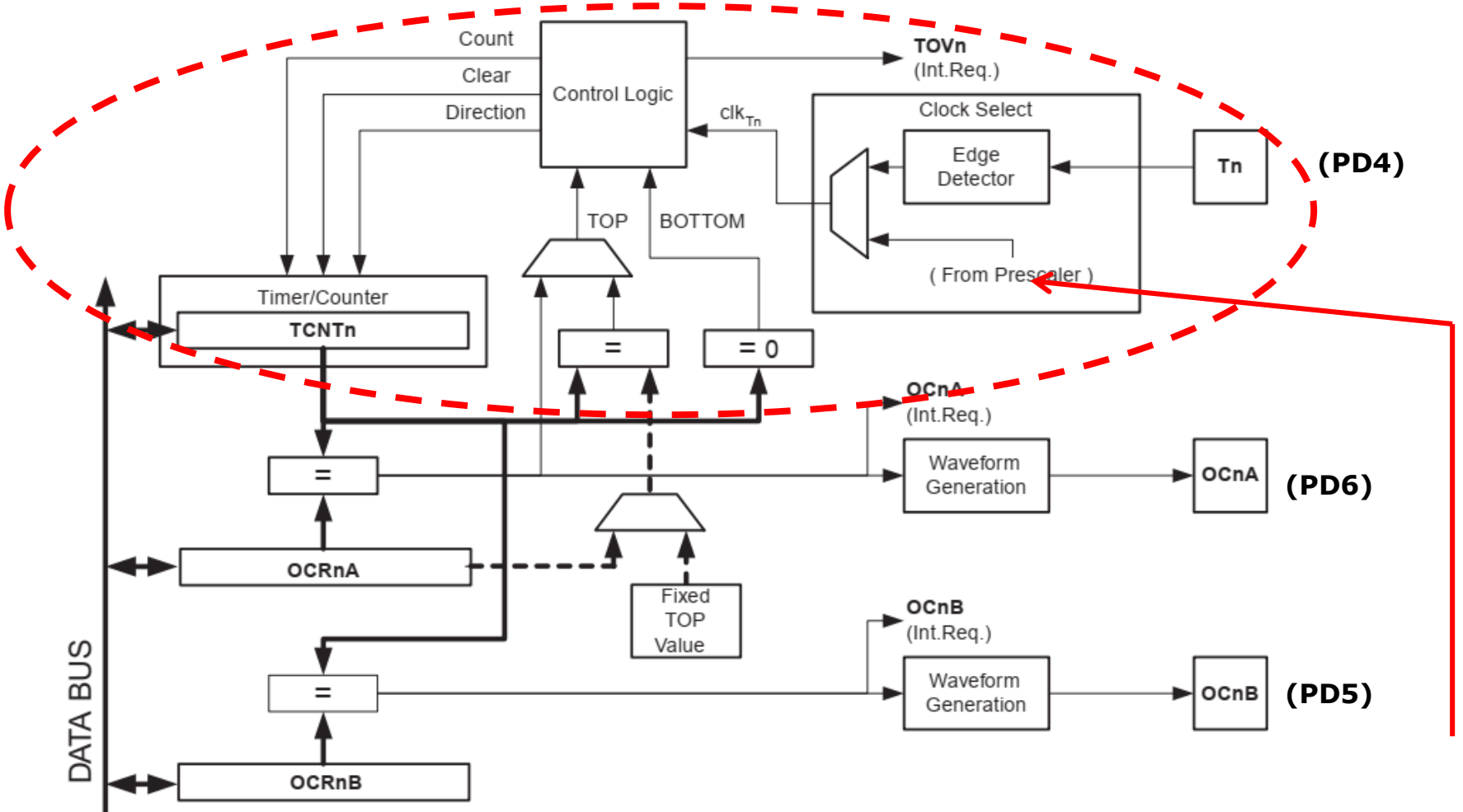
- Dependiendo de la aplicación y la exactitud de tiempo deseada optaremos por las diferentes fuentes de reloj para el módulo TIMER.

- Aplicaciones:
  - Generación de Retardos =>baja a mediana exactitud
  - Generación de Señales => de mediana a alta exactitud
  - Interrupción de tiempo real (sistemas de control) => alta exactitud
  - Reloj de tiempo real => alta exactitud
  - Medición de señales => alta exactitud

- La exactitud y estabilidad más alta se consigue utilizando Osciladores a Cristal de cuarzo (del orden de ±100ppm o mejor)

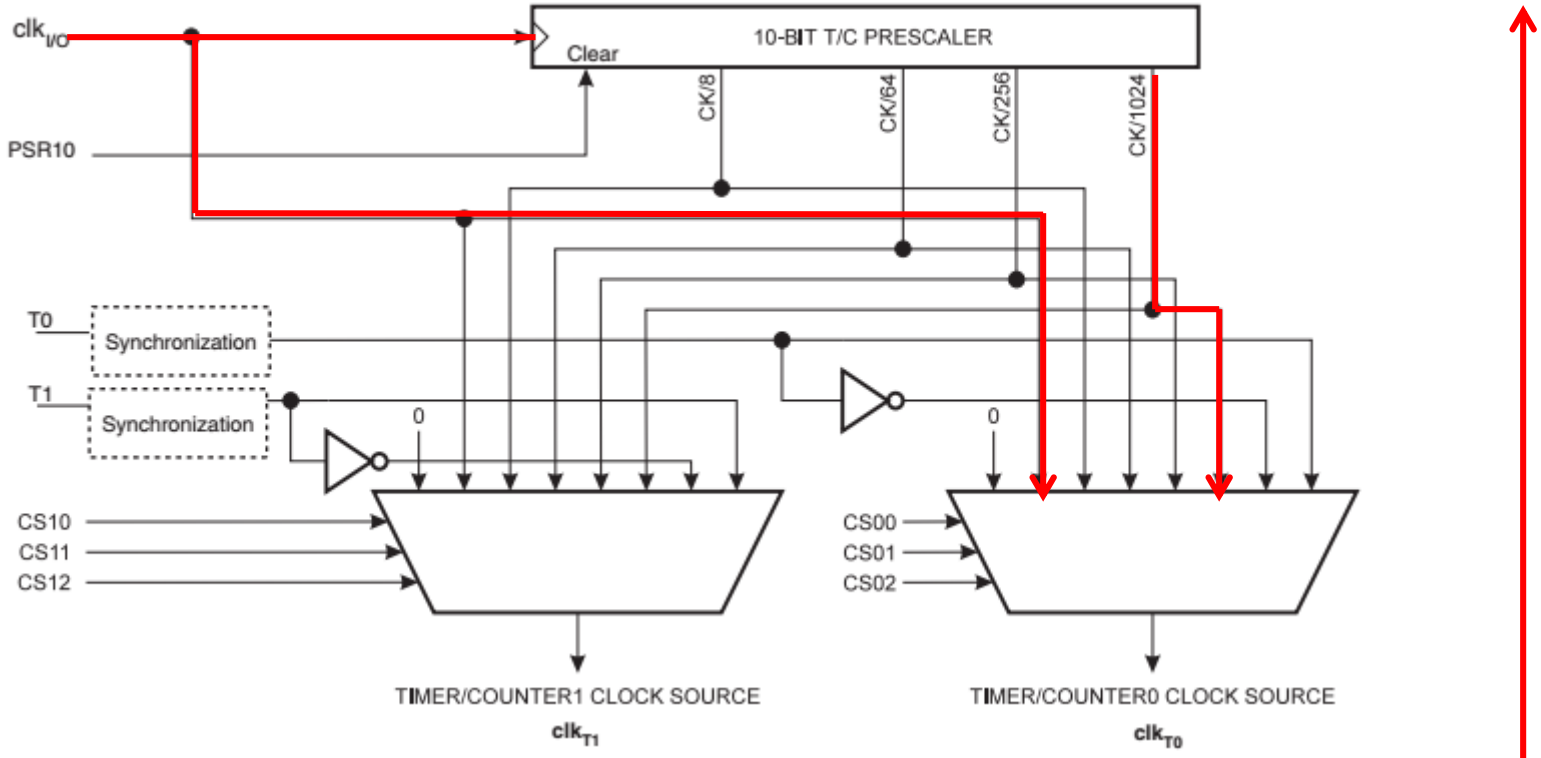
ATMEGA328 - TIMER 0

- Diagrama en bloques



TIMER 0

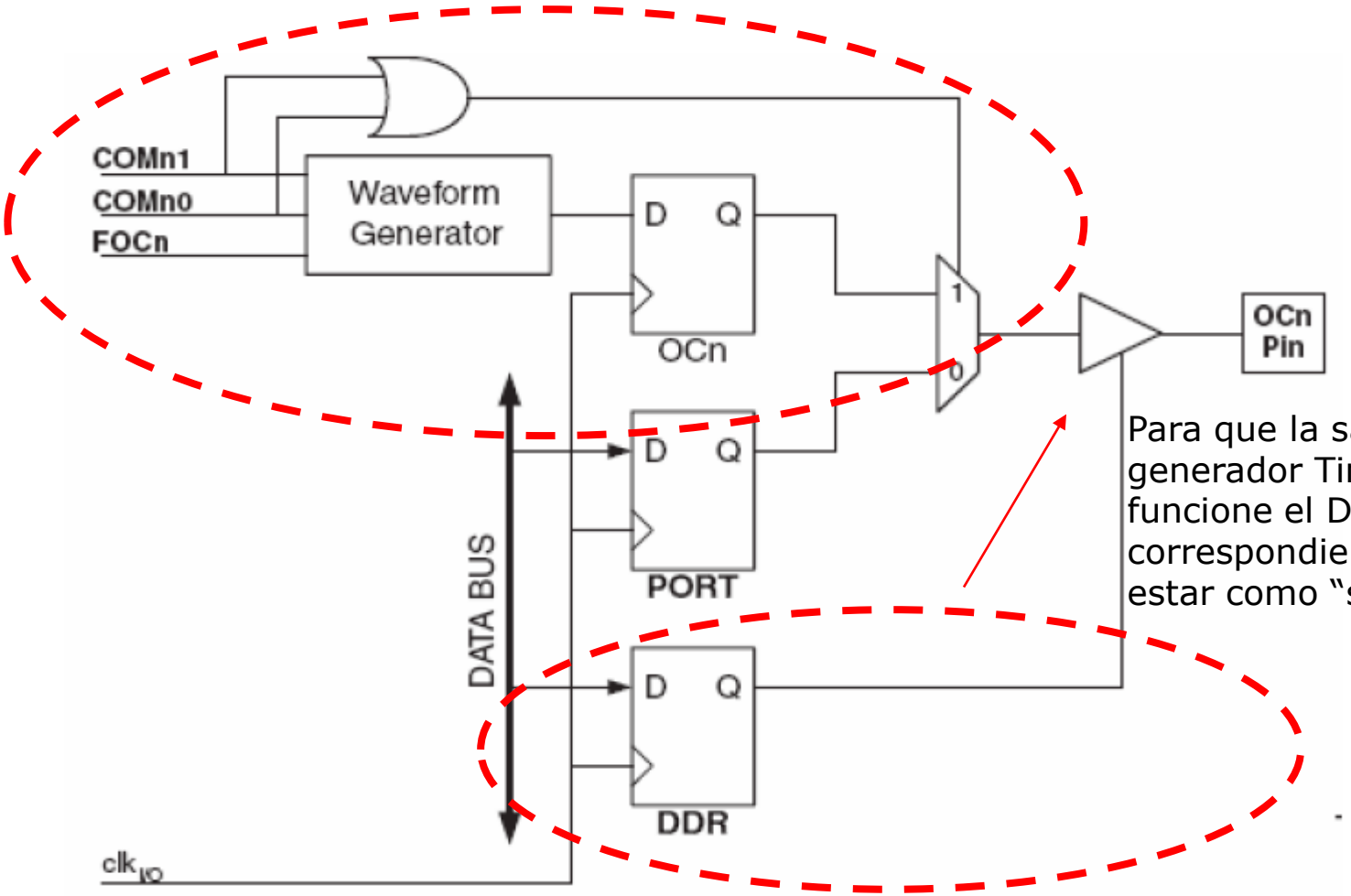
- TIMER 0 y 1: El PRESCALER permite configurar el reloj de conteo para adecuar la resolución. Además permite configurar las entradas T0,T1 para conteo de pulsos externos.



$f_{clkT0} = f_{clkIO} / PRESCALER$

TIMER 0

- Control de la salida I/O



TIMER 0

Registros para su programación

Registros de configuración

7	6	5	4	3	2	1	0	
COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
R/W	R/W	R/W	R/W	R	R	R/W	R/W	
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
W	W	R	R	R/W	R/W	R/W	R/W	

Contador

7	6	5	4	3	2	1	0	
TCNT0[7:0]								TCNT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Registros de comparación

7	6	5	4	3	2	1	0	
OCR0A[7:0]								OCR0A
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
7	6	5	4	3	2	1	0	
OCR0B[7:0]								OCR0B
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

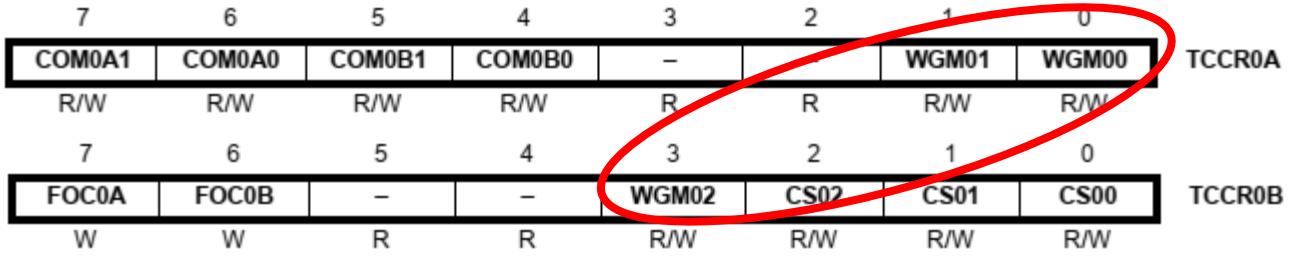
Conf. de interrupciones

7	6	5	4	3	2	1	0	
–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
R	R	R	R	R	R/W	R/W	R/W	

Banderas de notificación

7	6	5	4	3	2	1	0	
–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
R	R	R	R	R	R/W	R/W	R/W	

TIMER 0



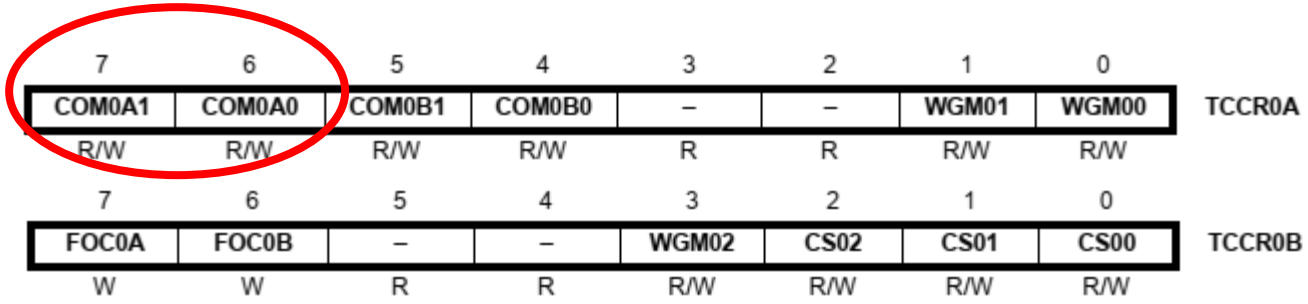
7	6	5	4	3	2	1	0	
COM0A1	COM0A0	COM0B1	COM0B0	–		WGM01	WGM00	TCCR0A
R/W	R/W	R/W	R/W	R	R	R/W	R/W	
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
W	W	R	R	R/W	R/W	R/W	R/W	

Table 14-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

TIMER 0



7	6	5	4	3	2	1	0	
COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
R/W	R/W	R/W	R/W	R	R	R/W	R/W	
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
W	W	R	R	R/W	R/W	R/W	R/W	

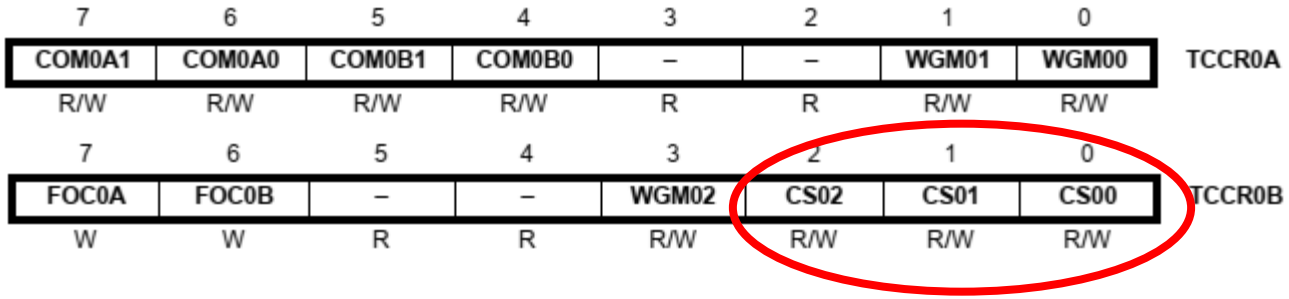
Table 14-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

- Idem para COM0Bx

No se actúa sobre los terminales
Los eventos se procesan x software
Ejemplo: retardos

TIMER 0



7	6	5	4	3	2	1	0	
COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
R/W	R/W	R/W	R/W	R	R	R/W	R/W	
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
W	W	R	R	R/W	R/W	R/W	R/W	

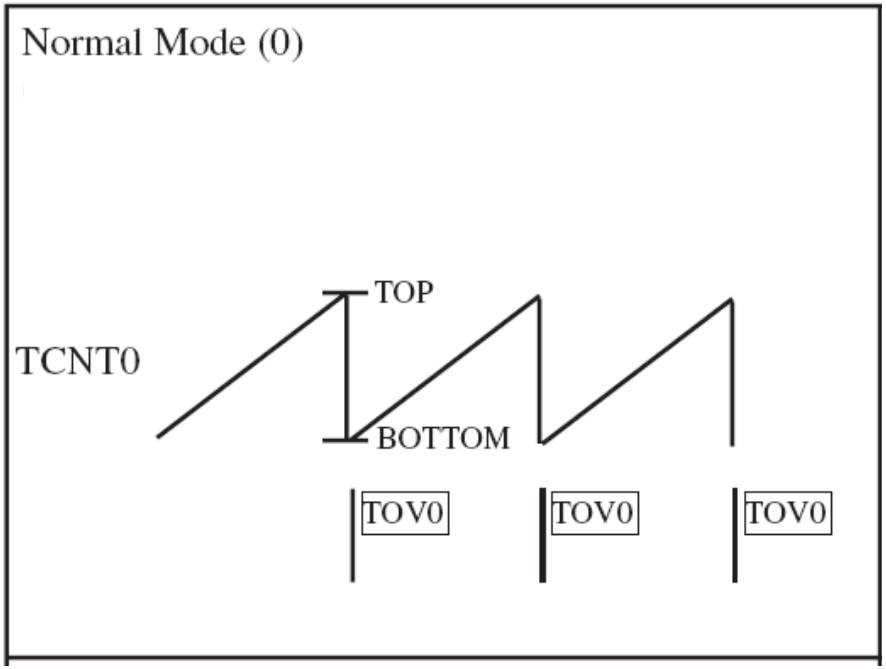
Timer detenido

Table 14-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

TIMER 0

- Modos de funcionamiento: normal



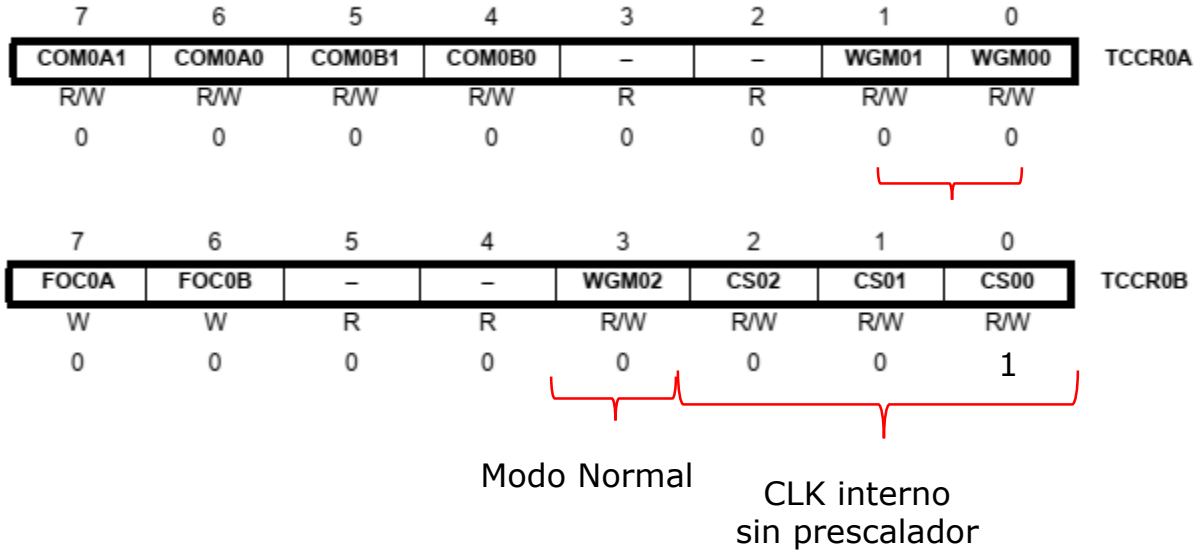
TCNT0: contador del Timer0

TOV0: flag de Overflow del Timer0

=> $f_{OVF} = f_{clkTn} / 2^8$

TIMER 0

Ejemplo 1: Configurar modo normal, reloj interno sin prescalador.



- Calcular la resolución de temporización para reloj interno de 1MHz y 8MHz:

  $T_{clkT0}=1 / f_{clkT0}$ => 1us y 0.125us respectivamente

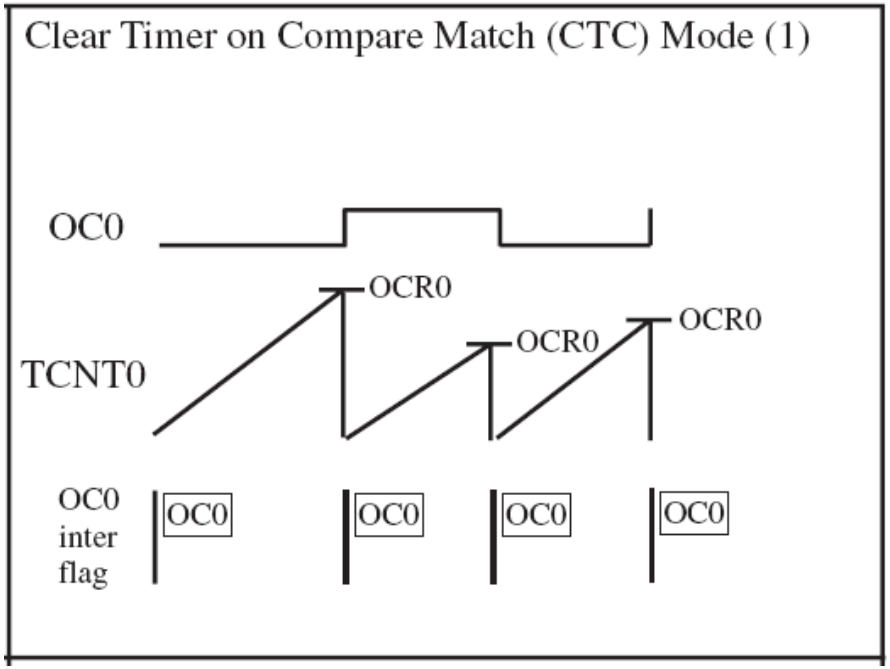
- Calcular el tiempo de overflow de dicho contador:

  $T_{OVF} = 2^8 . T_{clkT0}$ => 256us y 32us respectivamente

Mejor resolución => menor rango

TIMER 0

- Modos de funcionamiento: CTC



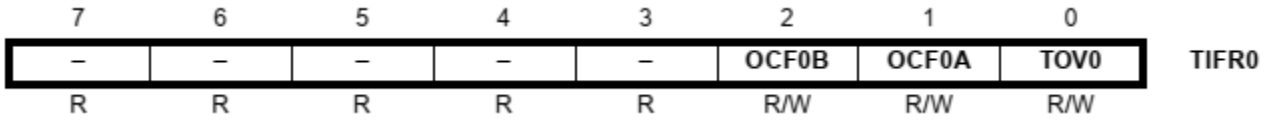
OC0: Pin salida del periférico

OCF0: flag de comparación

OCR0: Registro que contiene el valor a comparar con el TCNT0

TIMER 0

- Flags o banderas de condición del Timer 0 en el Registro **TIFR0**:



Activación de los flags:

TOV0 =1 cuando el contador pasa de FF a 00 (desborde u overflow)

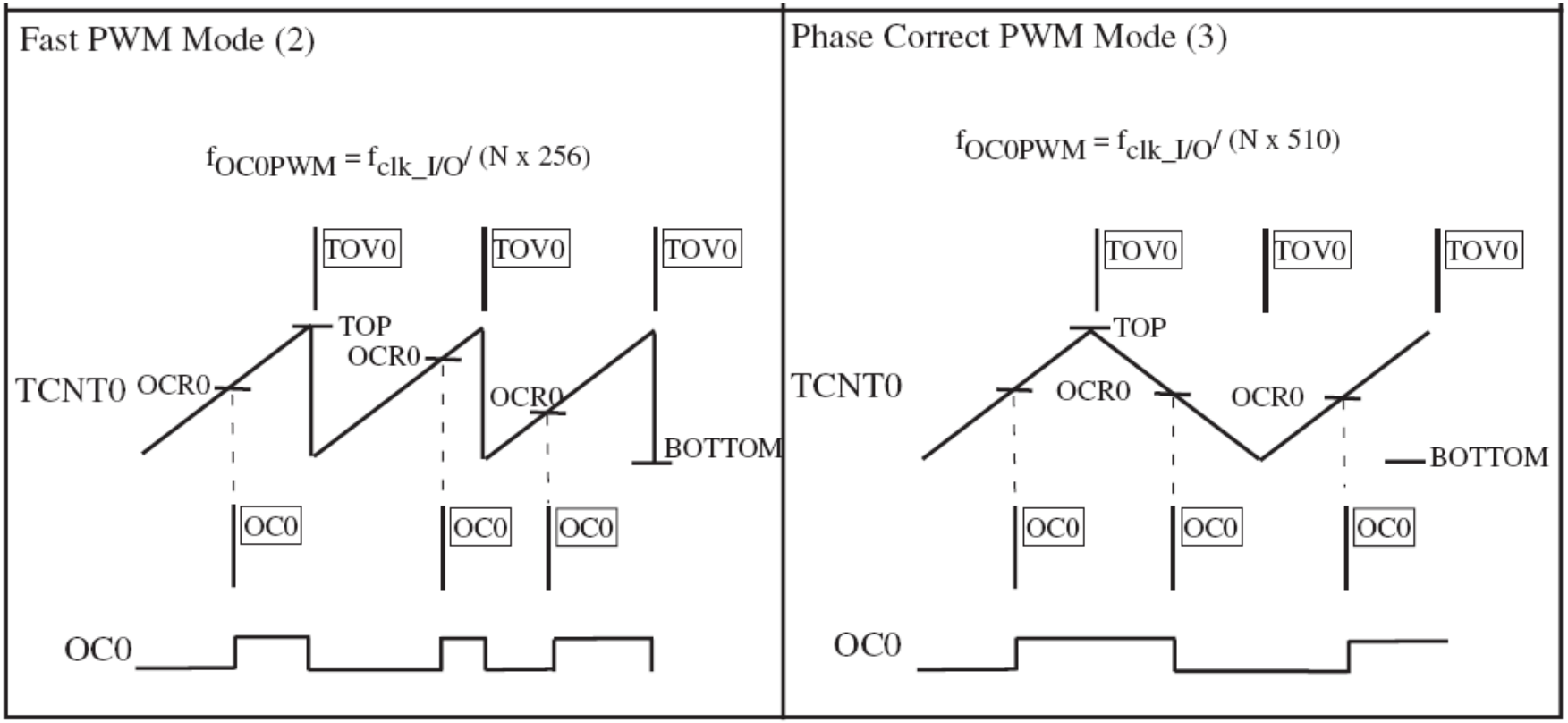
OCF0x =1 cuando el contador iguala al valor del registro OCR0x

- Borrado de los flags:
  - OCF0x se borra automáticamente cuando se ejecuta el vertor de interrupción correspondiente.
  - En modo polling el programador debe volver a 0 los flags escribiendo un “1” en dichos flags

```
TIFR0 |=(1<<TOV0);
TIFR0|=(1<<OCF0A);
```

TIMER 0

- **Modos de funcionamiento: PWM**



Lo veremos en el TP4

TIMER 0

- Ejemplo 2: invertir PB4 cada 70useg

```
#include "avr/io.h"

void T0Delay(void);

int main(void){

    DDRB|=(1<<PORTB4);
    while(1){
        T0Delay();
        PORTB^=(1<<PORTB4);

    }

}
```

TIMER 0

```
//70us=>70 ciclos de 1us
//Si fclk=8MHz => ftimer0=fclk/prescaler=8/8=1MHz
//Ttimer0=1us
//70 ciclos => 256-70=186
```

- Ejemplo 2: invertir PB4 cada 70useg

```
void T0Delay(void){
    TCNT0=186;
    TCCR0A=0x00;
    TCCR0B=0x02; //modo normal, prescaler 8
    while(TIFR0&(1<<TOV0)==0);
    TCCR0B=0x00; //apagar timer0
    TIFR0|=(1<<TOV0);
}
```

Algoritmo para Generar un retardo (delay)

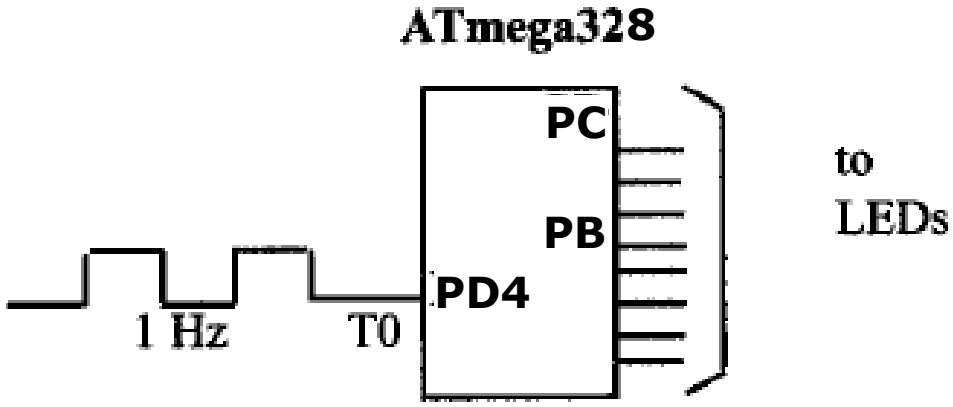
1) Cargar el TCNT0 con el valor inicial de conteo (el retardo será TOF-inicial)
2) Cargar en el TCCR0 la configuración deseada. Al elegir la fuente de reloj (el timer comienza el conteo)
3) Verificar mediante el flag de Tov0 si se alcanzo el valor FF en el contador
4) Detener el contador desconectando la fuente de reloj
5) borrar el flag Tov0

TIMER 0

- Ejemplo 3: Utilización como contador de eventos externos

- Supongamos que disponemos de un reloj externo de 1Hz conectado al pin T0

T0 (PD4) is connected to a 1-Hz external clock.



- Vamos a visualizar en los leds el conteo de pulsos (16bits) cada 1seg.

TIMER 0

- Ejemplo 3: Utilización como contador de eventos externos

```
#include "avr/io.h"

int main(void){

    DDRD&=~(1<<PORTD4);
    PORTD|=(1<<PORTD4);
    DDRB=DDRC=0xFF;
    TCCR0A=0x00;
    TCCR0B=0x06; //external clock T0 pin
    TCNT0=0;

    while(1){
        do{
            PORTB=TCNT0;
        }while(TIFR0&(1<<TOV0)==0);
        TIFR0|=(1<<TOV0);
        PORTC++;
    }
}
```

Notar que en cada overflow sumo 1 al PORTC, extendiendo la capacidad de conteo

TIMER 0

- Ejemplo 4: invertir PB5 utilizando interrupción de comparador A del TIMER 0

```
#include "avr/io.h"
#include "avr/interrupt.h"

int main(void){

    DDRB|=(1<<PORTB5);

    TCCR0A=0x02;     //modo CTC
    TCCR0B=0x01;     //clk prescaler=1
    OCR0A=40;        //módulo
    TIMSK0=(1<<OCIE0A);   //int comp A
    sei();

    while(1){
        //Tarea de Background
    }
}

ISR(TIMER0_COMPA_vect)
{
  PORTB^=(1<<PORTB5);
}
```

¿preguntas?

Tarea:

- Graficar la señal de salida en PB5 en función del tiempo
- Calcular la frecuencia de la señal de salida en PB5, asumiendo f_{clk}=8MHz
- Simular y verificar el comportamiento en Proteus con un Osciloscopio
- ¿afecta la latencia de la interrupción?