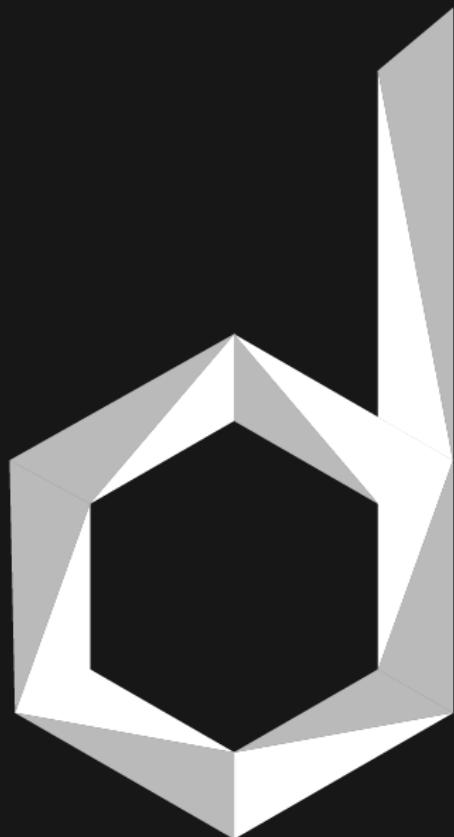


INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

LENGUAJE C, PROGRAMACIÓN MICROCONTROLADORES: ATMEGA328P (ARDUINO)

MICROCHIP STUDIO

Temporizadores y Señal PWM

Contenido

Temporizadores.....	2
Temporizador por Desbordamiento (Overflow).....	5
Temporizador por Comparación (Compare match).....	11
Temporizador por Comparación CTC (Clear to Compare)	16
PWM con Temporizadores.....	20
PWM Fase Correcta	27
PWM Rápido o Fast PWM.....	31
PWM Fase y Frecuencia Correcta	36
Puertos B, C y D Físicos en el Arduino	41
Programar Físicamente la Placa Arduino UNO con el Programa de Microchip Studio.....	43
Utilización de Librerías en Microchip Studio: Control de Pantalla LCD	45
Conexión Física del Circuito: Motor con control de señal PWM	46
Referencias:	47



Temporizadores

En el ATMEGA328P incluido en el Arduino existen 3 temporizadores, el T0, T1 y T2, la utilización de los temporizadores hará que la ejecución del código sea mucho más rápida y precisa, se pueden utilizar de 3 maneras distintas:

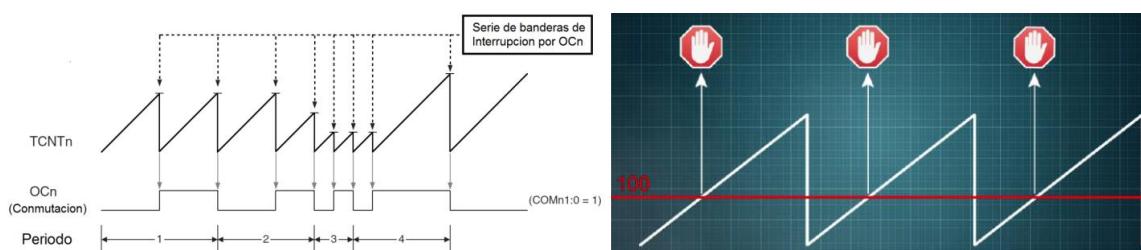
TIMER 0	8 BIT	255
TIMER 1	16 BIT	65536
TIMER 2	8 BIT	255

- **Timer por desbordamiento o normal (T0, T1 y T2):** Es el temporizador más sencillo y sirve para realizar una interrupción cuando se llega al número máximo que puede llegar el temporizador.



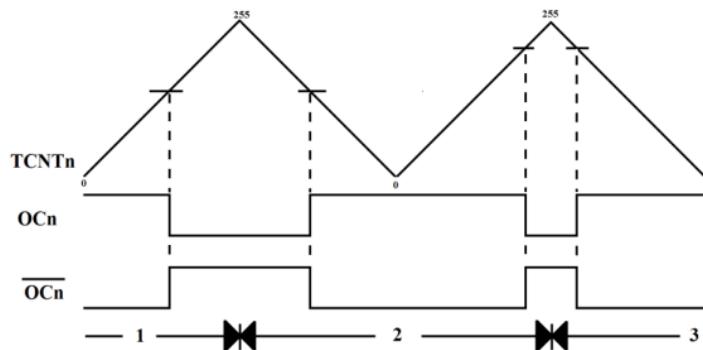
- **Timer por comparación (T0, T1 y T2):** Es un temporizador que realiza su interrupción cuando el conteo llega a un valor específico, comparando los registros **TCNT#** (que realiza el conteo) y **OCR#A** (que marca el límite del conteo) u **OCR#B**, dependiendo del tipo de comparador, ya que no todos permiten usar los dos canales A y B. Existen 3 modos de comparación descritos a continuación.

- **CTC (Clear to Compare):** Modo de comparación donde cuando el temporizador **TCNT#** alcanza el mismo nivel que el valor de comparación **OCR#A**, se reinicia a 0 el valor del temporizador para volver a realizar el conteo desde el inicio, generando una señal de diente de sierra.
 - Permite variar la frecuencia de la señal de salida, pero no el porcentaje de valor positivo y negativo en la señal (duty cycle).
 - Solamente se puede utilizar el canal A.



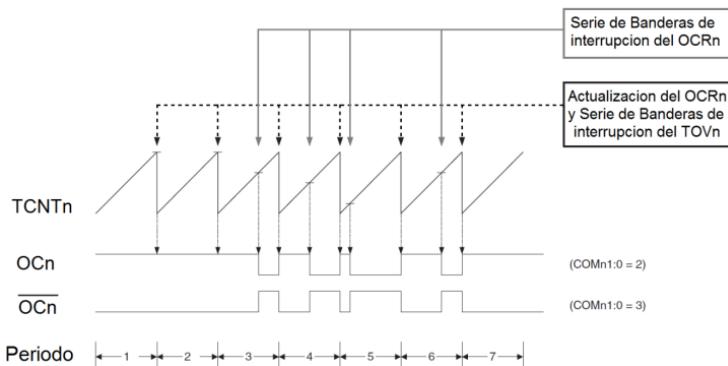
- **PWM fase correcta:** Se refiere a cuando el valor del registro contador llamado **TCNT#** se iguala al valor del registro comparador **OCR#A** u **OCR#B**, al ocurrir esto, en vez de que el valor del contador brinque a valer 0, solo irá decrementando su valor hasta llegar a cero poco a poco y cuando llegue a cero vuelva a subir, generando una señal triangular, con la amplitud igual al valor de **OCR#A** u **OCR#B**.

- No permite variar la frecuencia de la señal, ya que en las señales PWM nunca se puede cambiar la frecuencia, solo el duty cycle, que es un porcentaje de tiempo en el cual la señal va a tener un estado lógico positivo durante todo el periodo de la señal, controlado con el valor de **OCR#**, tomando en cuenta que el 100% es cuando vale el **valor máximo del comparador** y el 0% es cuando vale 0, dependiendo del número de bits del timer que se esté utilizando.
 - Es importante mencionar que el valor máximo del comparador puede ser modificado con registros **OCRA** o **ICR1**, además de poder tener una resolución de 8, 9 y 10 bits.
 - Con resolución nos referimos al conteo máximo que puede realizar:
 - **Resolución de 8 bits:** Valor máximo de conteo = 1111
1111 = 255.
 - **Resolución de 9 bits:** Valor máximo de conteo = 0001
1111 1111 = 511.
 - **Resolución de 10 bits:** Valor máximo de conteo = 0011
1111 1111 = 1023.
 - **Valor máximo personalizado:** Valor máximo del conteo redefinido por el registro **OCRA** o **ICR1**, esto reduce el valor de conteo de las demás variables que definen la frecuencia, periodo y tiempo de duty cycle de la señal PWM.
- La señal de PWM fase correcta **maneja bajas frecuencias**, este es más **recomendable para usarse en motores CD**, ya que no provoca una pérdida de amplitud en la señal de salida, porque los motores no pueden ser manejados por señales de altas frecuencias sin presentar pérdidas en la amplitud de su señal de salida.
- **Se puede utilizar el canal A y B**, a veces hasta se usa uno para fijar el valor máximo personalizado y el otro para obtener la señal PWM.



- **PWM rápido:** Modo de comparación donde al llegar a su valor máximo, cae a valer cero de la misma forma como lo hace el comparador CTC, creando una señal de diente de sierra. A comparación de la señal PWM de fase correcta, su periodo es menor, por lo que la señal se crea de forma más rápida, teniendo el **doble de frecuencia que la señal PWM de fase correcta**.

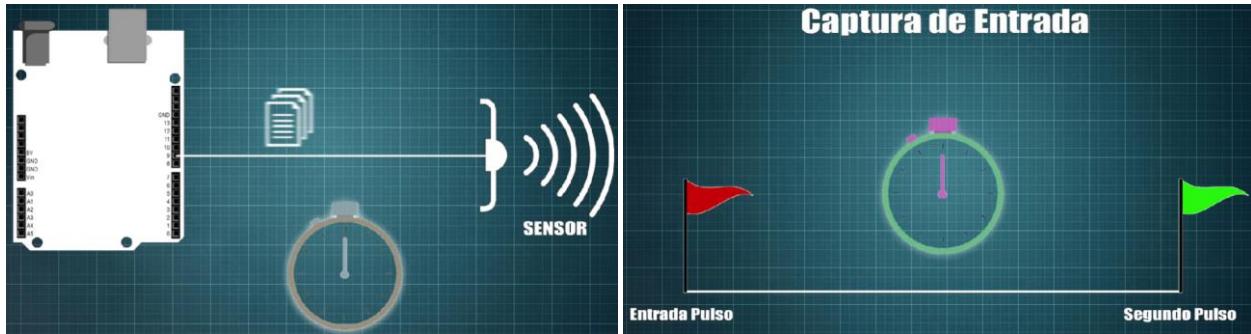
- No permite variar la frecuencia de la señal, ya que en las señales PWM nunca se puede cambiar la frecuencia, solo el duty cycle, que es un porcentaje de tiempo en el cual la señal va a tener un estado lógico positivo durante todo el periodo de la señal, controlado con el valor de **OCR#**, tomando en cuenta que el 100% es cuando vale el **valor máximo del comparador** y el 0% es cuando vale 0, dependiendo del número de bits del timer que se esté utilizando.
 - Es importante mencionar que el valor máximo del comparador puede ser modificado con registros **OCRA** o **ICR1**, además de poder tener una resolución de 8, 9 y 10 bits.
 - Con resolución nos referimos al conteo máximo que puede realizar:
 - **Resolución de 8 bits:** Valor máximo de conteo = 0000 1111 1111 = 255.
 - **Resolución de 9 bits:** Valor máximo de conteo = 0001 1111 1111 = 511.
 - **Resolución de 10 bits:** Valor máximo de conteo = 0011 1111 1111 = 1023.
 - **Valor máximo personalizado:** Valor máximo del conteo redefinido por el registro **OCRA** o **ICR1**, esto reduce el valor de conteo de las demás variables que definen la frecuencia, periodo y tiempo de duty cycle de la señal PWM.
- La señal de PWM rápido **maneja altas frecuencias**, pero esto puede causar que, si se utiliza en motores, la amplitud baje en su señal de salida, por lo cual **no es recomendable usarse en motores, excepto en servomotores**.
 - Después de 3dB de pérdida en la señal de salida, la señal necesita ser amplificada y se toma como una mala conversión de energía, se puede ver la pérdida en un diagrama de Bode que toma en cuenta la amplitud de la señal contra la frecuencia de la señal de entrada.
- **Se puede utilizar el canal A y B**, a veces hasta se usa uno para fijar el valor máximo personalizado y el otro para obtener la señal PWM.



- **Timer por captura (T1):** Es un tipo de temporizador que se activa solamente cuando hay un estímulo externo en uno de los pines del Arduino, se usa principalmente para medir el tiempo entre impulsos externos o medir la frecuencia cuando por ejemplo un sensor recaba datos

periódicos en un intervalo específico. En el ATMEGA328P del Arduino se puede utilizar solo con el temporizador T1.

- Este modo se puede habilitar a través del bit **ICIE1** del registro **TIMSK# (Timer Interrupt Mask Register T#)** ya que a través de este se elige el modo del comparador, donde si se pone como 1 lógico se activa el modo de interrupción por captura de entrada.
- El bit **ICF1** representa la **bandera de interrupción del temporizador por captura** en el registro **TIFR#**, poniéndose como 1 lógico cuando ocurre una interrupción de este tipo.



Temporizador por Desbordamiento (Overflow)

El temporizador depende directamente de la frecuencia del oscilador contenido en el microcontrolador, en el ATMEGA328P la frecuencia del oscilador es de 16 MHz, ya que el ciclo máquina, que es el paso en el que se basa el microcontrolador para realizar sus funciones, es igual al inverso de la frecuencia del reloj u oscilador.

$$CM = \frac{1}{f} = \frac{1}{16 \times 10^6} = 62.5 \text{ ns}$$

Esto significa que cada 62.5 nanosegundos el microcontrolador ejecuta una acción.

Los timers **T0 y T2 son registros de 8 bits**, osea que cuentan de 0 a 255, entonces la velocidad del conteo se obtiene al realizar la siguiente ecuación, que es el tiempo que tardará el contador en desbordarse, osea en llegar a su valor máximo que es 1111 1111:

$$Vel_{T0/T2} = \frac{1}{f} * 2^{\#bits} = \frac{1}{f} * 2^8 = 62.5 \times 10^{-9} * 255 = 16 \mu\text{s}$$

Esto se cumple para los temporizadores **T0 y T2 que son de 8 bits**, pero para el temporizador **T1 que es de 16 bits**, el tiempo de desbordamiento es el siguiente:

$$Vel_{T1} = \frac{1}{f} * 2^{\#bits} = \frac{1}{f} * 2^{16} = 62.5 \times 10^{-9} * 65,536 = 4.096 \text{ ms}$$

Cuando un registro contador **TCNT0**, **TCNT1** o **TCNT2** llega a su punto máximo, se desborda, osea que empieza a contar desde 0000 0000 de nuevo, levantando así el bit **TOV#** que representa la **bandera de interrupción del temporizador por desborde** en el registro **TIFR#**, la interrupción será atendida inmediatamente por el microcontrolador y el programa saltará a la sección del código que el usuario haya designado cada vez que exista un desbordamiento, que es una función llamada **ISR (Interrupt**

Service Routine), usada para manejar interrupciones en general, pero sabe que debe reaccionar ante un temporizador por desbordamiento u otro porque como parámetro recibe un vector llamado:

- **TIMER0_OVF_vect**: Para el temporizador **T0** cuando se está manejando una **interrupción por desbordamiento**.
- **TIMER1_OVF_vect**: Para el temporizador **T1** cuando se está manejando una **interrupción por desbordamiento**.
- **TIMER2_OVF_vect**: Para el temporizador **T2** cuando se está manejando una **interrupción por desbordamiento**.

Además, se tiene que tomar en cuenta la escala, que indica cuántos ciclos de máquina deben cumplirse para incrementar el conteo en uno, esto indica cuántas veces más lento es el conteo en comparación con la frecuencia del oscilador interno de 16 MHz, que dura 62.5 ns.

A continuación, se muestran las escalas disponibles para los contadores de 8 y 16 bits:

T0 y T2 = Contadores de 8 bits		
Pre-escala (Pscla)	Pscla*CM*Número de cuentas = TM	Tiempo máximo ™
1:1 = 2^0	$1*62.5X10^{-9}*2^8$	16 µs
1:8 = 2^3	$8*62.5X10^{-9}*2^8$	128 µs
1:64 = 2^6	$64*62.5X10^{-9}*2^8$	128 µs
1:256 = 2^8	$256*62.5X10^{-9}*2^8$	1.024 ms
1:1024 = 2^{10}	$1024*62.5X10^{-9}*2^8$	16.384 ms

T1 = Contador de 16 bits		
Pre-escala (Pscla)	Pscla*CM*Número de cuentas = TM	Tiempo máximo ™
1:1 = 2^0	$1*62.5X10^{-9}*2^{16}$	4.096 ms
1:8 = 2^3	$8*62.5X10^{-9}*2^{16}$	32.2768 ms
1:64 = 2^6	$64*62.5X10^{-9}*2^{16}$	0.2621 s
1:256 = 2^8	$256*62.5X10^{-9}*2^{16}$	1.0485 s
1:1024 = 2^{10}	$1024*62.5X10^{-9}*2^{16}$	4.1943 s

Lo anteriormente mencionado se maneja en el **registro de control del temporizador TCCR#B**, que indica la **escala de conteo** o si se va a utilizar un **oscilador externo o el interno** de 16 MHz, con el TM podremos después calcular el tiempo unitario de cada timer:

Registro e Control del Temporizador/Contador (TCCR0B)

BIT								
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00	

Los Bits CS02, CS01 y CS00 se emplean para seleccionar el reloj que emplea el temporizador para su cuenta, es decir es la pre escala que se le da al reloj en la tabla 5.2

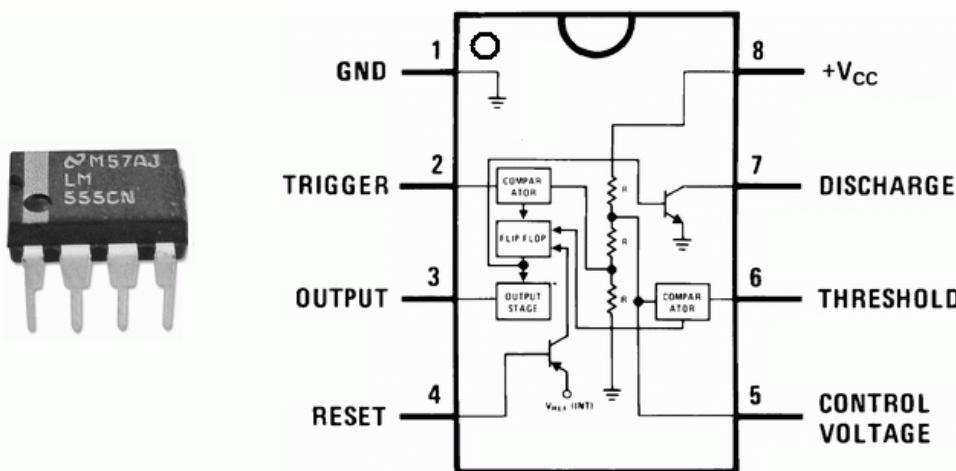
Los bits **CS## (Clock Select)** del registro **TCCR#B (Timer Counter Control Register T# B)** son usados para indicar cierta configuración, incluyendo si el reloj no está activo, la escala o si se está utilizando un oscilador externo. Puede apagarse el oscilador si es que se quiere ahorrar energía en el microcontrolador. Además, como T2 no tiene pin externo su tabla es distinta.

TABLA 5.2 TIEMPOS DE LA FUENTE DE RELOJ PARA EL TEMPORIZADOR

CS02	CS01	CS00	Descripción
0	0	0	Ningún fuente de reloj (Temporizador detenido)
0	0	1	Sin pre escalador 1:1
0	1	0	Pre escalador a 1:8
0	1	1	Pre escalador a 1:64
1	0	0	Pre escalador a 1:256
1	0	1	Pre escalador a 1:1024
1	1	0	Fuente externa de reloj en T0 - Reloj en Flanco de bajada
1	1	1	Fuente externa de reloj en T0 - Reloj en Flanco de subida

El oscilador externo se debe conectar al pin **T0** o **T1**, que pertenecen al puerto D, específicamente al **pin 4 (PD2)** y **5 (PD3)** del Arduino, **T2** no tiene pin para conectar un oscilador externo, por lo cual puede acceder a escalas adicionales. El reloj externo se puede hacer con un **circuito integrado LM555** o se pueden contar pulsos con un **sensor** o **botón**, pero en este último caso se debe inicializar el registro que hace el conteo en su valor máximo que es de **255 si se está utilizando un timer de 8 bits** o **65,536 si se está utilizando un timer de 16 bits**.

Oscilador LM555:



Existen registros **TCCR#B** para configurar la escala de los 3 temporizadores:

- **TCCR0B:** Sirve para indicar la escala del conteo del temporizador **T0**.
 - **CS02, CS01** y **CS00** son los bits menos significativos del registro **TCCR0B** que sirven para indicar la escala o si se utiliza un temporizador interno o externo para el timer **T0 de 8 bits**.
- **TCCR1B:** Sirve para indicar la escala del conteo del temporizador T1.
 - **CS12, CS11** y **CS10** son los bits menos significativos del registro **TCCR1B** que sirven para indicar la escala o si se utiliza un temporizador interno o externo para el timer **T1 de 16 bits**.
- **TCCR2B:** Sirve para indicar la escala del conteo del temporizador **T2**.
 - **CS22, CS21** y **CS20** son los bits menos significativos del registro **TCCR2B** que sirven para indicar la escala del timer **T2 de 8 bits** (**no existe pin para usar un temporizador externo para T2**).

Como el temporizador T2 no tiene un pin del Arduino por el cual se pueda conectar un oscilador o pulso externo, su tabla de escalas es distinta:

17.11.2 TCCR2B – Timer/Counter Control Register B								
Bit	7	6	5	4	3	2	1	0
(0xB1)	FOC2A	FOC2B	-	-	WGM22	CS22	CS21	CS20
Read/Write	W	W	R	R	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Table 17-9. Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S} (no prescaling)
0	1	0	$\text{clk}_{\text{T2S}}/8$ (from prescaler)
0	1	1	$\text{clk}_{\text{T2S}}/32$ (from prescaler)
1	0	0	$\text{clk}_{\text{T2S}}/64$ (from prescaler)
1	0	1	$\text{clk}_{\text{T2S}}/128$ (from prescaler)
1	1	0	$\text{clk}_{\text{T2S}}/256$ (from prescaler)
1	1	1	$\text{clk}_{\text{T2S}}/1024$ (from prescaler)

Pines del Microcontrolador ATMEGA328P:

(PCINT14/RESET) PC6	1	28	PC5 (ADC5/SCL/PCINT13)
(PCINT16/RXD) PD0	2	27	PC4 (ADC4/SDA/PCINT12)
(PCINT17/TXD) PD1	3	26	PC3 (ADC3/PCINT11)
(PCINT18/INT0) PD2	4	25	PC2 (ADC2/PCINT10)
(PCINT19/OC2B/INT1) PD3	5	24	PC1 (ADC1/PCINT9)
(PCINT20/XCK/T0) PD4	6	23	PC0 (ADC0/PCINT8)
VCC	7	22	GND
GND	8	21	AREF
(PCINT6/XTAL1/TOSC1) PB6	9	20	AVCC
(PCINT7/XTAL2/TOSC2) PB7	10	19	PB5 (SCK/PCINT5)
(PCINT21/OC0B/T1) PD5	11	18	PB4 (MISO/PCINT4)
(PCINT22/OC0A/AIN0) PD6	12	17	PB3 (MOSI/OC2A/PCINT3)
(PCINT23/AIN1) PD7	13	16	PB2 (SS/OC1B/PCINT2)
(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)

El **registro de máscara de interrupción del temporizador** o **TIMSK** sirve para indicar si el contador será de tipo desbordamiento o comparación.

Registro de la Máscara de Interrupción del Temporizador/Contador(TIMSK0)

BIT	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0

- **TIMSK0:** Registro de la máscara de interrupción para el temporizador **T0 de 8 bits**.

- **TOIE0 (Timer Overflow Interruption Enable – T0 de 8 bits):** Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T0** por desbordamiento.
 - **Tipo desbordamiento:** Si se le manda un 1 lógico al bit **TOIE0**.
 - **Tipo comparación:** Si se le deja como 0 lógico al bit **TOIE0**.
- **TIMSK1:** Registro de la máscara de interrupción para el temporizador **T1 de 16 bits**.
 - **TOIE1 (Timer Overflow Interruption Enable – T1 de 16 bits):** Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T1** por desbordamiento.
 - **Tipo desbordamiento:** Si se le manda un 1 lógico al bit **TOIE1**.
 - **Tipo comparación:** Si se le deja como 0 lógico al bit **TOIE1**.
- **TIMSK2:** Registro de la máscara de interrupción para el temporizador **T2 de 8 bits**.
 - **TOIE2 (Timer Overflow Interruption Enable – T2 de 8 bits):** Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T2** por desbordamiento.
 - **Tipo desbordamiento:** Si se le manda un 1 lógico al bit **TOIE2**.
 - **Tipo comparación:** Si se le deja como 0 lógico al bit **TOIE2**.

Los registros contadores **TCNT0**, **TCNT1** y **TCNT2** sirven para llevar la cuenta de los temporizadores **T0** y **T2** para contadores de 8 bits y **T1** para contadores de 16 bits correspondientemente.

Registro Temporizador/Contador (TCNT0)

BIT	7	6	5	4	3	2	1	0

Este registro sirve para llevar la cuenta del temporizado, al ser un registro de 8 bits cuenta desde 0 hasta 255 antes de desbordarse y llevar el registro de nueva cuenta a ceros, el cambio de un número a otro dentro del registro dependerá de el preescalador y el frecuencia del reloj.

Recordemos que las interrupciones pueden ser activadas por temporizadores, esto se hace a través de las banderas descritas a continuación, que se pueden meter en las funciones **ISR (Interrupt Service Routine)** que detonan las acciones a realizar cuando se detecte una interrupción. En los temporizadores por desbordamiento, el bit más significativo llamado **TOV#** del registro **TIFR#** se toma como bandera para detectar cuando se debe detonar una interrupción, estos bits para cada temporizador se llaman **TOV0**, **TOV1** y **TOV2** correspondientemente, cuando el bit se pone en 1 lógico es porque se ha llegado al desbordamiento, si se queda como 0 lógico es porque todavía no ocurre el desbordamiento, que lleva el valor del registro contador a cero.

Registro de Banderas de Interrupción del Temporizador/Contador (TIFR0)

BIT	7	6	5	4	3	2	1	0
						OCF0A	OCF0A	TOV0

Los bits **TOV2**, **TOV1** y **TOV0**, se ponen a uno cuando ocurre el desbordamiento del temporizador correspondiente, estas bandera se limpian por hardware cuando se ejecuta la interrupción correspondiente.

Ejemplo para obtener un tiempo específico con cierto contador utilizando una escala:

Cuando queremos obtener cierto tiempo en el contador, por ejemplo, de 5 segundos y en el desborde del conteo no llegamos a hacer el conteo exacto o es demasiado grande para guardarse en el registro contador, se pone cierta condición para que el desborde del timer no se realice al alcanzar el número 1111 1111, sino un número binario distinto o por medio del uso de ciclos con un condicional **if**.

Una manera de calcular el número de conteo que se le debe asignar al registro contador **TCNT** es la siguiente, en donde debemos considerar que, al obtener el resultado, si este es más grande que el alcance de los temporizadores, osea mayor a 255 para **T0** y **T2** o mayor a 65,535 para **T1**, se deberán crear condicionales **if**, que utilicen el número de ciclos para permitir que solamente tras haber pasado ese número de ciclos, se efectué la acción que queremos detonar con la interrupción:

$$\text{Conteo}_{\text{AUX}} = \text{Límite}_{\text{conteo}} - \frac{f_{\text{micro}} * t_{\text{deseado}}}{\text{escala}} = \frac{16 \times 10^6 * t_{\text{deseado}}}{\text{escala}}$$
$$\text{Conteo}_{\text{TCNT}} = \text{Conteo}_{\text{AUX}}$$

Si el límite del conteo para **T0** y **T2** que es 255 o para **T1** que es 65,535 es sobrepasado por el resultado de la ecuación anterior, se deben realizar pasos adicionales, donde se usará dentro de la función **ISR** un condicional que solo permita la ejecución cada que se cumpla el número de ciclos obtenidos con la siguiente ecuación y donde cada que se ejecute una interrupción, se aumente el número de ciclos hasta llegar al ciclo calculado:

$$\text{Conteo}_{\text{AUX}} > \text{Límite}_{\text{conteo}}$$
$$\text{ciclos} = \frac{\text{Conteo}_{\text{AUX}}}{\text{ConteoMáximo}_{\text{temporizador}}}$$
$$\text{Conteo}_{\text{TCNT}} = \text{ConteoMáximo}_{\text{temporizador}} - \frac{\text{Conteo}_{\text{AUX}}}{\text{ciclos}}$$

Con este resultado entonces se debe asignar el valor de **Conteo_{TCNT}** al registro **TCNT** y el valor de **ciclos** a una variable cualquiera que se encuentre dentro de un condicional **if** para que, al detonarse una interrupción, la variable que almacena el número de ciclos empiece a contar desde cero y cuando llegue al valor de ciclos calculado, ya le sea posible a la interrupción ejecutar su acción, de esta manera se puede crear un temporizador de cualquier valor de tiempo.

En conclusión, los registros de configuración para el uso de temporizadores por desbordamiento son:

- **TCCR#B**: Los bits **CS## (Clock Select)** del registro **Timer Counter Control Register T# B** indican la escala del conteo o si se usa un temporizador interno o externo.
- **TIMSK#**: El bit **TOIE# (Timer Overflow Interruption Enable)** indica si el contador será de tipo desbordamiento (1 lógico) o comparación (0 lógico).
- **TCNT0, TCNT1 y TCNT2**: Son los registros que realizan el conteo de 0 a 255 para T0 y T2 o de 0 a 65,535 para T1.
- **TIMER#_OVF_vect**: Vector de interrupción dado a la función **ISR (Interruption Service Routine)** para que la interrupción sea activada por un temporizador por desbordamiento.

- **TIFR#**: Registro **Timer Interruption Flag Register** que levanta una bandera por cada temporizador T0, T1 o T2 que indica cuando ya se haya hecho una interrupción por temporizador.

Temporizador por Comparación (Compare match)

Cuando un registro contador **TCNT0**, **TCNT1** o **TCNT2** llega al punto de comparación indicado por el registro **OCR#A** o **OCR#B** se activa la interrupción, levantando así los bits **OCF#A** y **OCF#B** que representan la **bandera de interrupción del temporizador por comparación** en el registro **TIFR#**, la interrupción será atendida inmediatamente por el microcontrolador y el programa saltará a la sección del código que el usuario haya designado cada vez que el temporizador llegue a un punto indicado de conteo, esa función es llamada **ISR (Interrupt Service Routine)** y es usada con el fin de manejar interrupciones en general, pero sabe que debe reaccionar ante un **temporizador de comparación** porque como parámetro recibe un vector llamado:

- Para el temporizador **T0 de 8 bits**:
 - **TIMER0_COMPA_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal A** y utilizando un número de comparación guardado en el registro **OCR0A (Output Compare Register A)**.
 - **TIMER0_COMPB_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal B** y utilizando un número de comparación guardado en el registro **OCR0B**.
- Para el temporizador **T1 de 16 bits**:
 - **TIMER1_COMPA_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal A** y utilizando un número de comparación guardado en el registro **OCR1A**.
 - **TIMER1_COMPB_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal B** y utilizando un número de comparación guardado en el registro **OCR1B**.
- Para el temporizador **T2 de 8 bits**:
 - **TIMER2_COMPA_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal A** y utilizando un número de comparación guardado en el registro **OCR2A**.
 - **TIMER2_COMPB_vect**: Parámetro usado cuando se está manejando una **interrupción por comparación del canal B** y utilizando un número de comparación guardado en el registro **OCR2B**.

Los registros **OCR2A** y **OCR2B** ambos sirven para detonar una interrupción, su única diferencia son los pines con los que están asociados y que no se pueden ejecutar los 2 en todos los tipos de comparadores.

Además, se tiene que tomar en cuenta la escala, que indica cuántos ciclos de máquina deben cumplirse para incrementar el conteo en uno, esto indica cuántas veces más lento es el conteo en comparación con la frecuencia del oscilador interno de 16 MHz, que dura 62.5 ns.

Así como se hacía con el temporizador por desbordamiento, se maneja el **registro de control del temporizador TCCR#B** para indicar la **escala de conteo** o si se va a utilizar un **oscilador externo o el**

interno de 16 MHz, en conjunto con sus bits **CS## (Clock Select)** del **registro TCCR#B** que son usados para indicar cierta configuración, incluyendo si el reloj no está activo, la escala o si se está utilizando un oscilador externo. Puede apagarse el oscilador si es que se quiere ahorrar energía en el microcontrolador.

TABLA 5.2 TIEMPOS DE LA FUENTE DE RELOJ PARA EL TEMPORIZADOR

CS02	CS01	CS00	Descripción
0	0	0	Ningún fuente de reloj (Temporizador detenido)
0	0	1	Sin pre escalador 1:1
0	1	0	Pre escalador a 1:8
0	1	1	Pre escalador a 1:64
1	0	0	Pre escalador a 1:256
1	0	1	Pre escalador a 1:1024
1	1	0	Fuente externa de reloj en T0 - Reloj en Flanco de bajada
1	1	1	Fuente externa de reloj en T0 - Reloj en Flanco de subida

Luego además en el registro **TCCR#B**, fuera de usarse los bits **CS##** para la configuración de la escala, se usa un bit o par de bits nuevos llamado **WMG#2 (Wave Generator Mode T# 2)**, que no solo se toman del registro **TCCR#B**, sino igual del registro **TCCR#A**.

Los Bits WGM00, WGM01 y WGM002 (*Waveform Generation Mode*) se emplean para definir la generación de la forma de onda que será usada, esta dependerá de los modos a usarse, los cuales son CTC y dos tipos de PWM (ver tabla 6.2). El modo normal se indica en la tabla para definir el uso normal del puerto.

Los Bits COM0A0 y COM0A1 (*Compare Match Output Mode*) control el comportamiento del bit de comparación de salida OC0A si uno ambos están en uno la salida OC0A se invalida la función normal del puerto, sin embargo el bit correspondiente al DDRD deberá de estar habilitado como salida.

Registro e Control del Temporizador/Contador (TCCR0B)

BIT								
7	6	5	4	3	2	1	0	
FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00	

Para indicar el tipo de comparador no solo se modifica el bit **WMG#2** del registro **TCCR#B**, sino que se debe de modificar también los bits **WGM#1** y **WGM#0** del registro **TCCR#A**, para el caso de **T0** y **T2**, ya que cuando se configura **T1** se utilizan los bits **WGM#3**, **WGM#2**, **WGM#1** y **WGM#0**. La combinación de bits indica el tipo de temporizador comparador.

Registro e Control del Temporizador/Contador (TCCR0A)

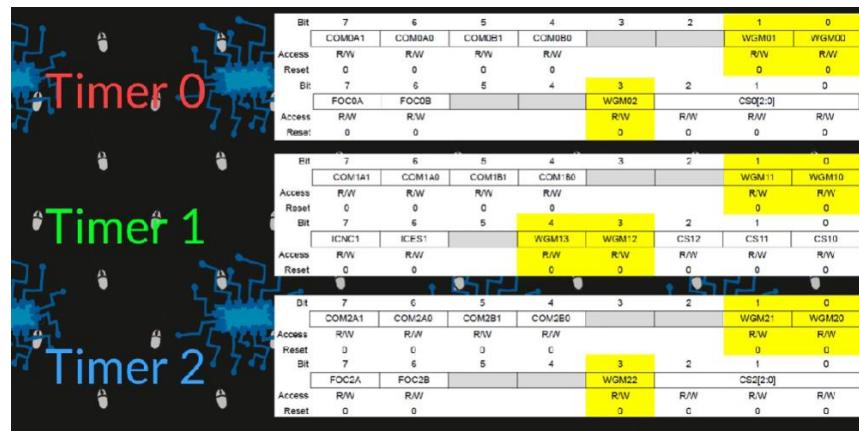
BIT								
7	6	5	4	3	2	1	0	
COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00	

La siguiente tabla funciona para configurar los temporizadores **T0** y **T2** se usan 3 bits:

TABLA 6.2 Modos de la generación de la forma de onda

WGM02	WGM01	WGM00	Descripción
0	0	0	Normal
0	0	1	PWM Fase Correcta
0	1	0	Limpia temporizador tras la comparación (CTC) OCRA
0	1	1	PWM Rápido
1	0	0	Reservado
1	0	1	PWM Fase Correcta OCRA
1	1	0	Reservado
1	1	1	PWM Rápido OCRA

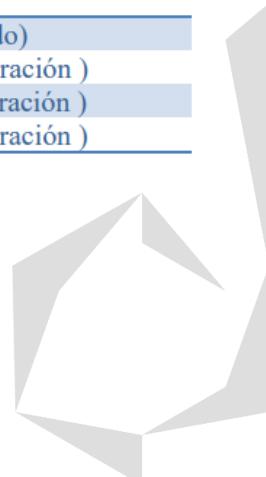
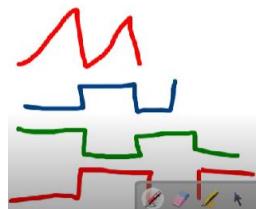
Para modificar la configuración del temporizador **T1** se usan 4 bits, en los otros se usan 3:



El registro de control de temporizador **TCCR#A** sirve para indicar con sus bits **WGM#1** y **WGM#0** en conjunto con el bit **WGM#2** del registro **TCCR#B** (si es que se están usando los temporizadores **T0** y **T2**) el tipo de comparador que se quiere elegir, pero además con los bits **COM#A1 (CompareMatch Output Mode T# A1)**, **COM#A0**, **COM#B1** y **COM#B0** del registro **TCCR#A** se indica el tipo de salida de los registros de comparación **OCR#A (Output Compare Register T# Canal A)** u **OCR#B (Output Compare Register T# Canal B)** que sirven para indicar el punto del conteo del temporizador por comparación donde se ejecutará la interrupción y se define el tipo de señal obtenida en la salida cuando se llegue a igualar el valor del registro **OCR#A** y el contador **TCNT#**.

TABLA 6.1 Modos de Comparación de la Salida

COM0A1	COM0A0	Descripción
0	0	Operación Normal del Puerto (OC0 desconectado)
0	1	Cambio Lógico de OC0 (al igualar en la comparación)
1	0	Pone en cero lógico OC0 (al igualar en la comparación)
1	1	Pone en uno lógico OC0 (al igualar en la comparación)



Cuando se está utilizando el temporizador **T1** se utilizan 4 bits de configuración, donde registro de control de temporizador **TCCR#A** sirve para indicar con sus bits **WGM#0** y **WGM#1** en conjunto con el bit **WGM#2** y **WGM#3** del registro **TCCR#B** (si es que se está usando el temporizador **T1**) el tipo de comparador que se quiere elegir, en donde además en la columna de TOP, se muestra el valor máximo al que puede llegar la señal para que active la interrupción.

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Aunado a esto, existe un registro de sincronización **GTCCR** (**General Timer Counter Control Register**) este se usa principalmente cuando se está usando más de 1 temporizador a la vez y lo que hace es reiniciar el conteo y escala y declarar el modo de sincronización de los temporizadores **T0**, **T1** y **T2**, esto se refiere a que podemos reiniciar el conteo de los timers y elegir si empezarán a funcionar y contar de forma sincronizada:

- **TSM**: Bit del registro **GTCCR** para la sincronización de timers:
 - **1 lógico**: Modo de sincronización activado para que todos los temporizadores corran al mismo tiempo y no manejen tiempos distintos.
 - **0 lógico**: Modo desincronizado, esto se refiere a que se puede manejar varios temporizadores, pero puede que empiecen a correr en tiempos distintos.
- **PSRASY**: Bit del registro **GTCCR** para la restauración del timer **T2** y fijar su escala en 0:
 - **1 lógico**: Reinicia el conteo de **T2** a cero y su escala a 0, poniéndolo en su estado inicial.
 - **0 lógico**: Deja activo y corriendo al temporizador **T2** con el último valor de conteo y escala con el que se quedó.
- **PSRSYNC**: Bit del registro **GTCCR** para la restauración del timer **T0** y **T1**, fijando su escala en 0:
 - **1 lógico**: Reinicia los conteos de **T0** y **T1** a cero y su escala a 0, poniéndolos en su estado inicial.
 - **0 lógico**: Deja activo y corriendo a los temporizadores **T0** y **T1** con el último valor de conteo y escala con la que se quedaron.

Una manera de calcular el número de conteo que se le debe asignar al registro comparador **OCR#** es la siguiente, en donde es muy importante considerar el límite de conteo de cada timer, ya que el resultado de la operación va muy de la mano de él, primero que nada, se debe calcular si el número de conteo es más grande que el que puede soportar cada timer, ya sea de 8 o 16 bits, para ello se tienen valores unitarios máximos que puede alcanzar cada temporizador:

- **T0** y **T0**: El valor máximo que se puede alcanzar con este temporizador de 16 bits **utilizando la escala más grande que es de 1024** es de **16 microsegundos**.
 - Conteo máximo de 0 a 255.
- **T1**: El valor máximo que se puede alcanzar con este temporizador de 16 bits **utilizando la escala más grande que es de 1024** es de **4.096 segundos**.
 - Conteo máximo de 0 a 65,536.

Ya sabiendo si el registro comparador va a aguantar el tiempo que se quiere alcanzar con el temporizador, se realizará la siguiente ecuación, donde al registro **OCR#** se asignará su valor más grande de conteo y se creará una variable que almacene el número de ciclos utilizados dentro de un condicional **if** que se encuentre dentro de la función **ISR** del temporizador y utilice el número de ciclos para permitir que solamente tras haber pasado ese número de ciclos, se efectúe la acción que queremos detonar con la interrupción:

$$Conteo_{UNITARIO} = \frac{1}{f_{micro}} * escala * ConteoMáximo_{temporizador}$$

Si el límite del conteo para **T0** y **T2** que es **255** o para **T1** que es **65,535** es sobrepasado por el resultado de la ecuación anterior, se deben realizar pasos adicionales, donde se usará dentro de la función **ISR** un condicional que solo permita la ejecución cada que se cumpla el número de ciclos obtenidos con la siguiente ecuación y donde cada que se ejecute una interrupción, se aumente el número de ciclos transcurridos hasta llegar al ciclo calculado:

$$ciclos = \frac{t_{deseado}}{Conteo_{UNITARIO}}$$

Con este resultado entonces se debe asignar el valor de **ConteoMáximo_{temporizador}** al registro comparador **OCR#** y el valor de **ciclos** a una variable cualquiera que se encuentre dentro de un condicional **if** para que, al detonarse una interrupción, la variable que almacena el número de ciclos empiece a contar desde cero y cuando llegue al valor de ciclos calculado, ya le sea posible a la interrupción ejecutar su acción, de esta manera se puede crear un temporizador de cualquier valor de tiempo.

$$Conteo_{OCR\#} = ConteoMáximo_{temporizador}$$

En conclusión, los registros de configuración para el uso de temporizadores por desbordamiento son:

- **TCCR#A**: Registro **Timer Counter Control Register T# A**:
 - Los bits **WGM#1** y **WGM#0 (Wave Generator Mode T#)** del registro **TCCR#A** indican en conjunto con el bit **WGM#2** del registro **TCCR#B** el tipo de temporizador comparador que se quiere usar, ya que existen 3: **CTC, PWM rápido y PWM fase correcta**.
 - Los bits **COM#A1 (CompareMatch Output Mode T# A1)** y **COM#A0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#A** del canal A.

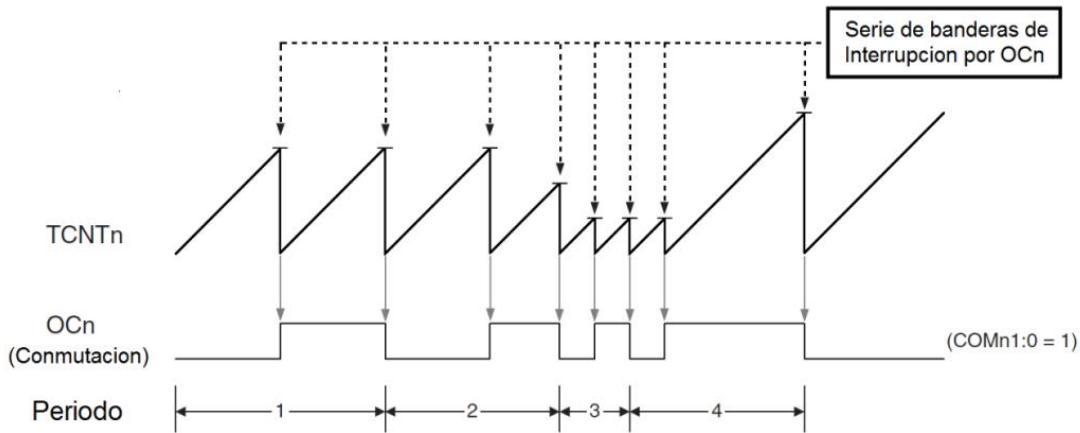
- Los bits **COM#B1** y **COM#B0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#B** del canal B.
- **TCCR#B**: Registro **Timer Counter Control Register T# B**:
 - El bit **WGM#2** (**Wave Generator Mode T# 2**) en conjunto con los bits **WGM#1** y **WGM#0** del registro **TCCR#A** indican el tipo de temporizador comparador que se quiere usar, ya que existen 3: **CTC**, **PWM rápido** y **PWM fase correcta**.
 - Los bits **CS##** (**Clock Select**) del registro **TCCR#B** indican la escala del conteo o si se usa un temporizador interno o externo.
- **TIMSK#**: Registro **Timer Interrupt Mask Register T#**:
 - El bit **TOIE#** (**Timer Overflow Interruption Enable**) indica si el contador será de tipo desbordamiento (1 lógico) o comparación (0 lógico).
 - El bit **OCIE#A** activa la interrupción por comparación del registro **OCR#A** (**Output Compare Register T# A**) perteneciente al canal A.
 - El bit **OCIE#B** activa la interrupción por comparación del registro **OCR#B** perteneciente al canal B.
- **TCNT0**, **TCNT1** y **TCNT2**: Son los registros que realizan el conteo de 0 a 255 para T0 y T2 o de 0 a 65,535 para T1.
- **TIMER#_COMP_A_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#A = TCNT#**.
- **TIMER#_COMP_B_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#B = TCNT#**.
- **TIFR#**: Registro **Timer Interruption Flag Register** que levanta una bandera por cada temporizador T0, T1 o T2 que indica cuando ya se haya hecho una interrupción por temporizador.
- **GTCCR (General Timer Counter Control Register)**: Registro usado principalmente cuando se está usando más de 1 temporizador a la vez, lo que hace es reiniciar sus conteos y escalas, logrando su sincronización.

Temporizador por Comparación CTC (Clear to Compare)

Los bits **WMG#2**, **WMG#1** y **WMG#0** sirven para definir el tipo de comparación que se va a realizar para **T0** y **T2**, mientras que Los bits **WMG#3**, **WMG#2**, **WMG#1** y **WMG#0** sirven para definir el tipo de comparación que se va a realizar para **T1**, ya que existen 3 tipos:

- **CTC (Clear to Compare)**: Modo de comparación donde cuando el temporizador alcanza el mismo nivel que el valor de comparación, se reinicia a 0 el valor del temporizador para volver a realizar el conteo desde el inicio. *Permite variar la frecuencia de la señal.*

Lo interesante con este tipo de comparación es que la señal de salida se ve afectada, ya que como el valor del contador **TCNT#** se reinicia cuando alcanza el valor de **OCR#A**, la amplitud y frecuencia de la señal se ve afectada, la amplitud es el mismo valor del registro **OCR#A**, pero la frecuencia se obtiene con la siguiente fórmula:



$$f_{OCn} = \frac{fmicro}{2 * escala * (1 + OCR\#A)} = \frac{16 \times 10^6}{2 * escala * (1 + OCR\#A)}$$

$$OCR\#A = \frac{fmicro}{f_{OCn} * 2 * escala} - 1 = \frac{fmicro}{f_{OCn} * 2 * escala} - 1$$

Después con el **registro de máscara de interrupción del temporizador o TIMSK#** se indica si el contador será de tipo desbordamiento o comparación.

Registro de la Máscara de Interrupción del Temporizador/Contador (TIMSK)

BIT	7	6	5	4	3	2	1	0
-	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0

Los bits 2 y 1 **OCIE0B** y **OCIE0A**, es el bit de habilitación de la interrupción del temporizador 0 por comparacion, cuando este bit esta en uno y junto con el bit de interrupción general en registro de estado también esta en uno la interrupción por comparación del Temporizador 0 es habilitada.

El bit 0 **TOIE0**, es el bit de habilitación de la interrupción del temporizador 0 por desbordamiento, cuando este bit esta en uno y junto con el bit de interrupción general en registro de estado también esta en uno la interrupción por desbordamiento del Temporizador 0 es habilitada. Es decir si ocurre un desbordamiento en el registró TCNT0 del temporizador 0 se ejecutara la interrupción.

- **TIMSK0:** Registro de la máscara de interrupción para el temporizador **T0 de 8 bits**.
 - **OCR0B:** Canal B de T0 que no puede ser utilizado en el modo CTC.
 - Conectado al **pin 5 (PD5)** perteneciente al **Puerto D** del Arduino.
 - **Por este pin se puede obtener una señal PWM, pero no una señal por comparador CTC.**
 - Activado por el registro **OCIE0B**.
 - **OCR0A (Output Compare Register T0 A):** Registro del canal A que indica el límite del conteo de 0 a 255 donde se va a detonar la interrupción, ya que se comparan los registros **TCNT0 (que realiza el conteo)** y **OCR0A (que marca el límite del conteo)**.
 - Conectado al **pin 6 (PD6)** perteneciente al **Puerto D** del Arduino.

- Por este pin se obtiene la señal requerida con la frecuencia calculada de la señal **OCR0A**.
 - Es mejor obtener la señal de OCR0A utilizando su interrupción y sacándola de otro puerto que no sea PD6.
 - Activado por el registro **OCIE0A**.
- **TOIE0 (Timer Overflow Interruption Enable – T0 de 8 bits)**: Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T0** por desbordamiento.
 - **Tipo desbordamiento**: Si se le manda un 1 lógico al bit **TOIE0**.
 - **Tipo comparación**: Si se le deja como 0 lógico al bit **TOIE0**.
- **TIMSK1**: Registro de la máscara de interrupción para el temporizador **T1 de 16 bits**.
 - **OCR1B**: Canal B de T1 que no puede ser utilizado en el modo CTC.
 - Conectado al **pin 10 (PB2)** perteneciente al **Puerto B** del Arduino.
 - Por este pin se puede obtener una señal PWM, pero no una señal por comparador CTC.
 - Activado por el registro **OCIE1B**.
 - **OCR1A (Output Compare Register T1 A)**: Registro del canal A que indica el límite del conteo de 0 a 65,535 donde se va a detonar la interrupción, ya que se comparan los registros **TCNT1 (que realiza el conteo)** y **OCR1A (que marca el límite del conteo)**.
 - Conectado al **pin 9 (PB1)** perteneciente al **Puerto B** del Arduino.
 - Por este pin se obtiene la señal requerida con la frecuencia calculada de la señal **OCR1A**.
 - Es mejor obtener la señal de OCR1A utilizando su interrupción y sacándola de otro puerto que no sea PB1.
 - Activado por el registro **OCIE1A**.
 - **TOIE1 (Timer Overflow Interruption Enable – T1 de 16 bits)**: Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T1** por desbordamiento.
 - **Tipo desbordamiento**: Si se le manda un 1 lógico al bit **TOIE1**.
 - **Tipo comparación**: Si se le deja como 0 lógico al bit **TOIE1**.
- **TIMSK2**: Registro de la máscara de interrupción para el temporizador **T2 de 8 bits**.
 - **OCR2B**: Canal B de T2 que no puede ser utilizado en el modo CTC.
 - Conectado al **pin 3 (PD3)** perteneciente al **Puerto D** del Arduino.
 - Por este pin se puede obtener una señal PWM, pero no una señal por comparador CTC.
 - Activado por el registro **OCIE2B**.
 - **OCR2A (Output Compare Register T2 A)**: Registro del canal A que indica el límite del conteo de 0 a 255 donde se va a detonar la interrupción, ya que se comparan los registros **TCNT2 (que realiza el conteo)** y **OCR2A (que marca el límite del conteo)**.
 - Conectado al **pin 11 (PB3)** perteneciente al **Puerto B** del Arduino.
 - Por este pin se obtiene la señal requerida con la frecuencia calculada de la señal **OCR2A**.
 - Es mejor obtener la señal de OCR2A utilizando su interrupción y sacándola de otro puerto que no sea PB3.
 - Activado por el registro **OCIE2A**.
 - **TOIE2 (Timer Overflow Interruption Enable – T2 de 8 bits)**: Bit del registro **TIMSK** que sirve para habilitar la interrupción del **timer T2** por desbordamiento.

- **Tipo desbordamiento:** Si se le manda un 1 lógico al bit **TOIE2**.
- **Tipo comparación:** Si se le deja como 0 lógico al bit **TOIE2**.

Los registros contadores **TCNT0**, **TCNT1** y **TCNT2** sirven para llevar la cuenta de los temporizadores **T0** y **T2** para contadores de 8 bits y **T1** para contadores de 16 bits correspondientemente.

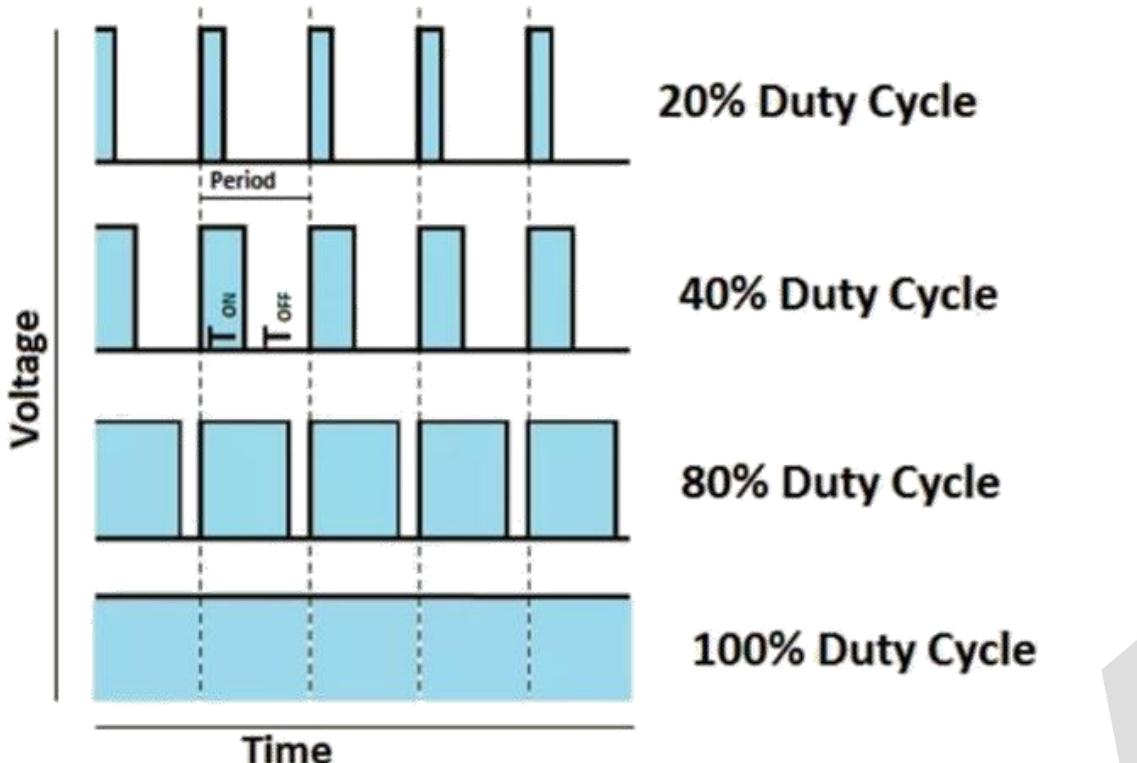
En conclusión, para poder activar el modo de timer por comparación, se deben modificar los siguientes registros para acceder a los distintos temporizadores comparadores de **T0**, **T1** y **T2**:

- **TCCR#A:** Registro **Timer Counter Control Register T# A**:
 - **T0 y T2:** 3 bits de configuración para elegir el tipo de temporizador comparador que se quiere usar.
 - Registro **TCCR#A**: Bits **WGM#1** y **WGM#0**.
 - Registro **TCCR#B**: Bit **WGM#2**.
 - **T1:** 4 bits de configuración para elegir el tipo de temporizador comparador que se quiere usar.
 - Registro **TCCR#A**: Bits **WGM#1** y **WGM#0**.
 - Registro **TCCR#B**: Bits **WGM#3** y **WGM#2**.
 - Los bits **COM#A1** y **COM#A0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#A**.
 - Los bits **COM#B1** y **COM#B0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#B**.
- **TCCR#B:** Registro **Timer Counter Control Register T# B**:
 - **T0 y T2:** 3 bits de configuración para elegir el tipo de temporizador comparador que se quiere usar.
 - Registro **TCCR#B**: Bit **WGM#2**.
 - Registro **TCCR#A**: Bits **WGM#1** y **WGM#0**.
 - **T1:** 4 bits de configuración para elegir el tipo de temporizador comparador que se quiere usar.
 - Registro **TCCR#B**: Bits **WGM#3** y **WGM#2**.
 - Registro **TCCR#A**: Bits **WGM#1** y **WGM#0**.
 - Los bits **CS## (Clock Select)** del registro **TCCR#B** indican la escala del conteo o si se usa un temporizador interno o externo.
- **TIMSK#:** Registro **Timer Interrupt Mask Register T#**:
 - El bit **TOIE# (Timer Overflow Interruption Enable)** indica si el contador será de tipo desbordamiento (**1 lógico**) o comparación (**0 lógico**).
 - El bit **OCIE#A** activa la interrupción por comparación del registro **OCR#A (Output Compare Register T# A)** que sirve para fijar el límite de conteo del temporizador por comparación.
 - El bit **OCIE#B** activa la interrupción por comparación del registro **OCR#B**.
- **TCNT0**, **TCNT1** y **TCNT2**: Son los registros que realizan el conteo de 0 a 255 para T0 y T2 o de 0 a 65,535 para T1.
- **TIMER#_COMP#_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#A = TCNT#**.

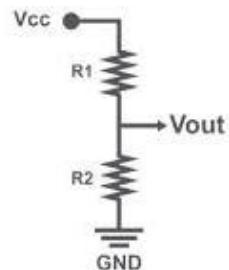
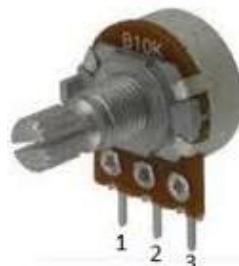
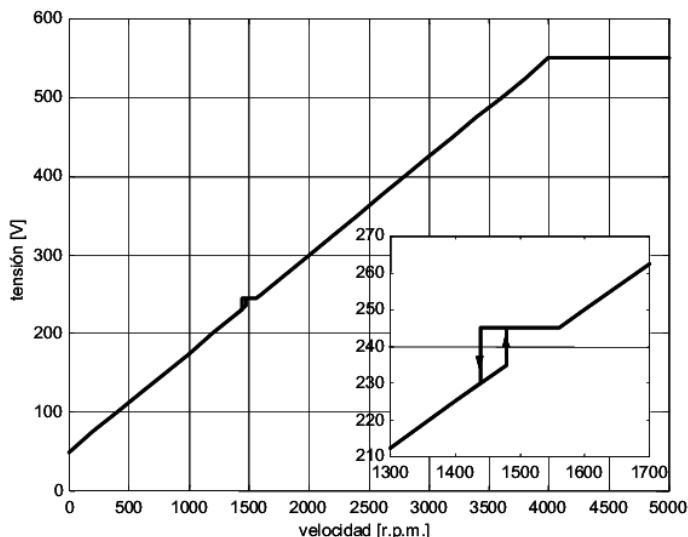
- **TIMER#_COMPB_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#B = TCNT#**.
- **TIFR#**: Registro **Timer Interruption Flag Register** que levanta una bandera por cada temporizador T0, T1 o T2 que indica cuando ya se haya hecho una interrupción por temporizador.
- **GTCCR (General Timer Counter Control Register)**: Registro usado principalmente cuando se está usando más de 1 temporizador a la vez, lo que hace es reiniciar sus conteos y escalas, logrando su sincronización.

PWM con Temporizadores

PWM (Pulse Width Modulation) se refiere a una señal cuadrada que **no varía su frecuencia**, pero sí su **porcentaje de nivel alto o bajo en un mismo periodo**, llamado duty cicle o ciclo de trabajo, esto sirve para variar la potencia entregada a un elemento electrónico, ya que de esta forma se puede controlar la luz vista en un foco, la velocidad de un motor, etc.



El gran aporte que tiene la señal PWM es el control de la velocidad de motores mediante un puente H sin perder su torque, ya que puede controlar la **velocidad del motor** variando la **tensión nominal entre sus conexiones**, pero el **torque del motor** depende de la **corriente**, entonces si se varía la velocidad del motor por ejemplo con un potenciómetro, se altera la corriente y por lo tanto el torque, por eso no es la mejor manera de variar la velocidad, ya que la potencia entregada al motor cae por usar una resistencia variable.

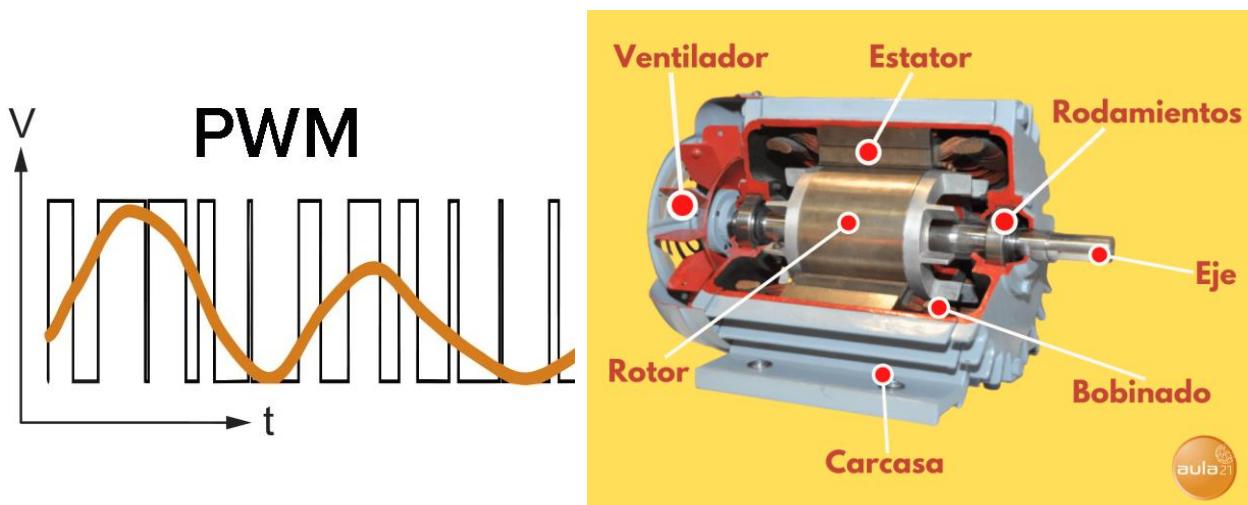


Con un potenciómetro se controla la velocidad a costa de la potencia.

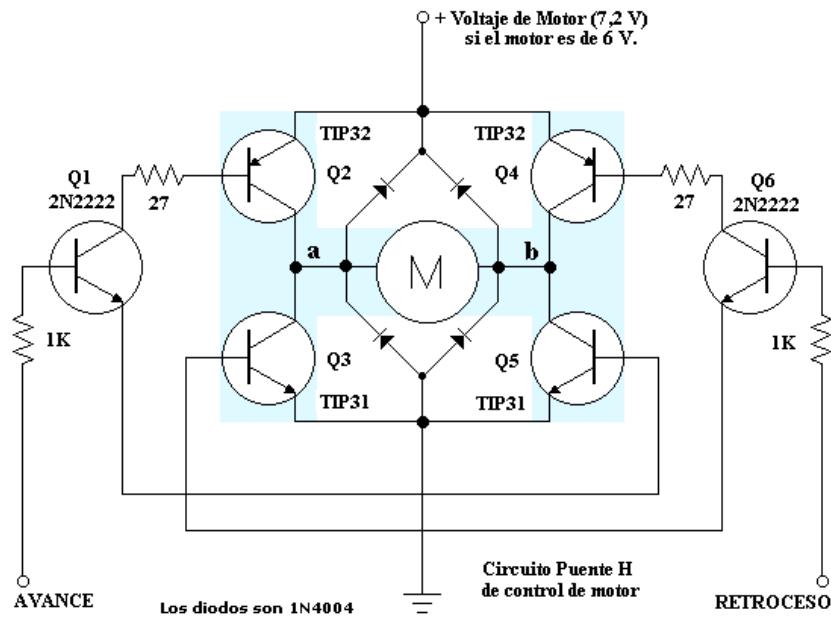
Cuando se usa una señal PWM para controlar la velocidad, la tensión nominal nunca varía, ya que la amplitud de la señal siempre es la misma, lo que ve el motor es como si se estuviera conmutando la tensión entregada, recibiendo por lo tanto un promedio de ella, siendo así capaz de variar la tensión sin dejar que caiga la potencia, por lo cual si se cambia el ciclo de trabajo puede verse como una reducción o aumento de la tensión entregada y además por lo mismo la corriente tampoco varía, entonces:

- **Tensión variable:** Velocidad del motor controlable.
- **Corriente no cambia:** No afecta al torque del motor.

Lo que el motor ve en su núcleo es una variación aproximada a la senoidal dependiendo del duty cycle de la señal cuadrada PWM.



La señal PWM que entrega el Arduino tiene una amplitud de 5V, por lo que se utiliza un Puente H, en donde se puede poner una alimentación externa que alimente al motor directamente, mientras que los pines de avance y retroceso son alimentados por la señal PWM, en donde cada controla la velocidad de giro del motor hacia alguno de sus lados.



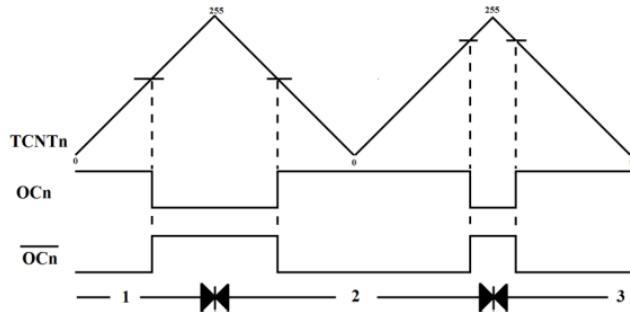
De esta forma es como se usa la señal PWM en motores CD, en motores CA se usa la señal PWM para controlar los variadores (inversores) de frecuencia, donde se convierte una señal de 1 fase (monofásica) a una de 3 fases (trifásica), no se ahondará mucho en este tema.

Se pueden activar 3 modos de PWM con el ATMEGA328P incluido en el Arduino y son los siguientes:

- **PWM fase correcta:** Se refiere a cuando el valor del registro contador llamado **TCNT#** se iguala al valor del registro comparador **OCR#A** u **OCR#B**, al ocurrir esto, en vez de que el valor del contador brinque a valer 0, solo irá decrementando su valor hasta llegar a cero poco a poco y cuando llegue a cero vuelva a subir, generando una señal triangular, con la amplitud igual al valor de **OCR#A** u **OCR#B**.
 - No permite variar la frecuencia de la señal, ya que en las señales PWM nunca se puede cambiar la frecuencia, solo el **duty cycle**, que es un porcentaje de tiempo en el cual la señal va a tener un estado lógico positivo durante todo el periodo de la señal, controlado con el valor de **OCR#**, tomando en cuenta que el 100% es cuando vale el **valor máximo del comparador** y el 0% es cuando vale 0, dependiendo del número de bits del timer que se esté utilizando.
 - Es importante mencionar que el valor máximo del comparador puede ser modificado con registros **OCRA** o **ICR1**, además de poder tener una resolución de 8, 9 y 10 bits.
 - Con resolución nos referimos al conteo máximo que puede realizar:
 - **Resolución de 8 bits:** Valor máximo de conteo = 1111 1111 = 255.
 - **Resolución de 9 bits:** Valor máximo de conteo = 0001 1111 1111 = 511.
 - **Resolución de 10 bits:** Valor máximo de conteo = 0011 1111 1111 = 1023.
 - **Valor máximo personalizado:** Valor máximo del conteo redefinido por el registro **OCRA** o **ICR1**, esto reduce el valor de

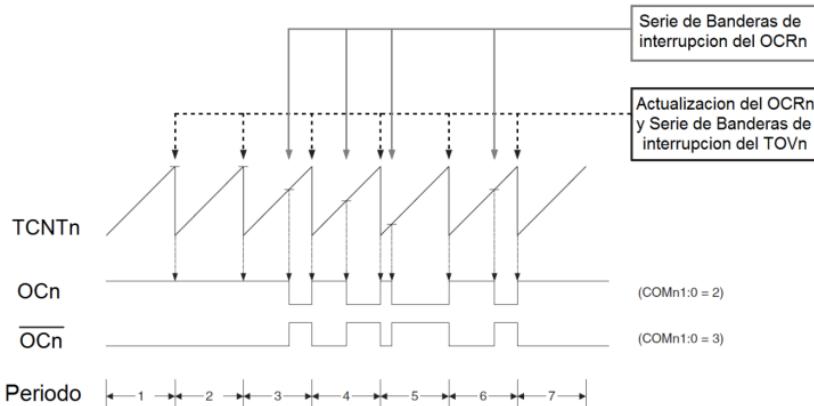
conteo de las demás variables que definen la frecuencia, periodo y tiempo de duty cycle de la señal PWM.

- La señal de PWM fase correcta **maneja bajas frecuencias**, este es más **recomendable para usarse en motores CD**, ya que no provoca una pérdida de amplitud en la señal de salida, porque los motores no pueden ser manejados por señales de altas frecuencias sin presentar pérdidas en la amplitud de su señal de salida.
- **Se puede utilizar el canal A y B**, a veces hasta se usa uno para fijar el valor máximo personalizado y el otro para obtener la señal PWM.

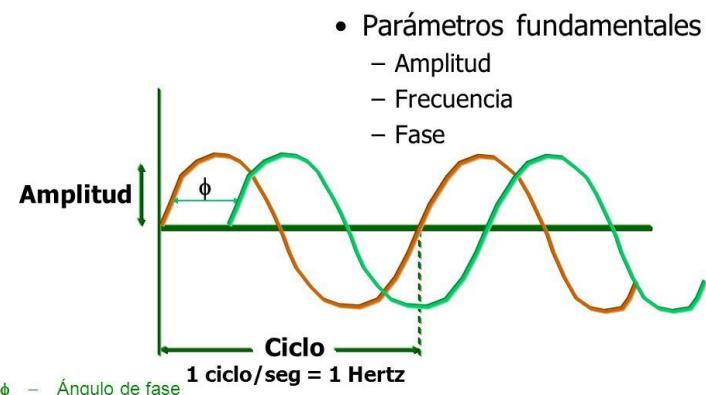


- **PWM rápido:** Modo de comparación donde al llegar a su valor máximo, cae a valer cero de la misma forma como lo hace el comparador CTC, creando una señal de diente de sierra. A comparación de la señal PWM de fase correcta, su periodo es menor, por lo que la señal se crea de forma más rápida, teniendo **el doble de frecuencia que la señal PWM de fase correcta**.
 - *No permite variar la frecuencia de la señal, ya que en las señales PWM nunca se puede cambiar la frecuencia, solo el duty cycle, que es un porcentaje de tiempo en el cual la señal va a tener un estado lógico positivo durante todo el periodo de la señal, controlado con el valor de OCR#, tomando en cuenta que el 100% es cuando vale el valor máximo del comparador y el 0% es cuando vale 0, dependiendo del número de bits del timer que se esté utilizando.*
 - Es importante mencionar que el valor máximo del comparador puede ser modificado con registros **OCRA** o **ICR1**, además de poder tener una resolución de 8, 9 y 10 bits.
 - Con resolución nos referimos al conteo máximo que puede realizar:
 - **Resolución de 8 bits:** Valor máximo de conteo = 1111 1111 = 255.
 - **Resolución de 9 bits:** Valor máximo de conteo = 0001 1111 1111 = 511.
 - **Resolución de 10 bits:** Valor máximo de conteo = 0011 1111 1111 = 1023.
 - **Valor máximo personalizado:** Valor máximo del conteo redefinido por el registro **OCRA** o **ICR1**, esto reduce el valor de conteo de las demás variables que definen la frecuencia, periodo y tiempo de duty cycle de la señal PWM.
 - La señal de PWM rápido **maneja altas frecuencias**, pero esto puede causar que, si se utiliza en motores, la amplitud baje en su señal de salida, por lo cual **no es recomendable usarse en motores, excepto en servomotores**.

- Despues de 3dB de perdida en la señal de salida, la señal necesita ser amplificada y se toma como una mala conversión de energía, se puede ver la perdida en un diagrama de Bode que toma en cuenta la amplitud de la señal contra la frecuencia de la señal de entrada.
- **Se puede utilizar el canal A y B**, a veces hasta se usa uno para fijar el valor máximo personalizado y el otro para obtener la señal PWM.



- **PWM fase y frecuencia correcta:** Se refiere a cuando se toma el registro contador del timer **TCNT#** y cuando su valor sea igual que el registro comparador **OCR#A** u **OCR#B**, en vez de que brinque a valer cero, solo vaya decrementando su valor hasta llegar a cero poco a poco y cuando llegue a cero vuelve a subir, generando una señal triangular, con la amplitud igual al valor de **OCR#A** u **OCR#B**.
 - A fuerza se debe definir el valor máximo del comparador por medio de los registros **OCRA** o **ICR1**.
 - **Se puede utilizar el canal A y B**, a veces hasta se usa uno para fijar el valor máximo personalizado y el otro para obtener la señal PWM.

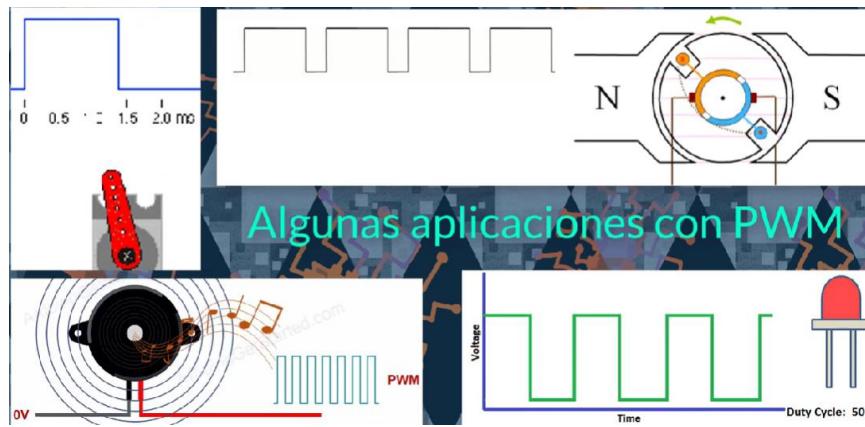


La fórmula para indicar el ciclo de trabajo de una señal PWM tomando en cuenta el tiempo perteneciente al periodo de la señal en el cual se busca tener un ciclo alto es:

$$Duty_{cycle} = \frac{t_{ON}}{T} * 100\% = t_{ON} * f * 100\%$$

De esta manera es posible tambien calcular una tensión promedio que se entregará por medio de la señal PWM, para que se sepa que tensión variable es la que está percibiendo el motor al que se está

alimentando o controlando, es importante mencionar que en este cálculo el ciclo de trabajo se manejará como decimal para obtener el voltaje promedio, para variar la velocidad de un motor, la intensidad luminosa de un led, la posición de un motor a pasos, un servomotor, el volumen de un Buzzer, etc. A continuación, serán descritas las fórmulas generales, pero estas varían dependiendo del modo de PWM que se elija:



$$V_{promedio} = Amplitud * Duty_{cycle} = V_{max} * Duty_{cycle}$$

Para poder activar el modo de PWM se utilizan los registros **TCCR#A (Timer Counter Control Register T# A)** y **TCCR#B (Timer Counter Control Register T# B)** de la misma forma con la que se elegía el comparador CTC, pero en este caso se elige simplemente otra de las opciones de la tabla, **T1** usa **4 bits de configuración**, mientras que **T0** y **T2** usa solo **3 bits de configuración**, con ellos se elige el tipo de temporizador comparador que se quiere implementar en el Arduino.

Table 15-5. Waveform Generation Mode Bit Description⁽¹⁾

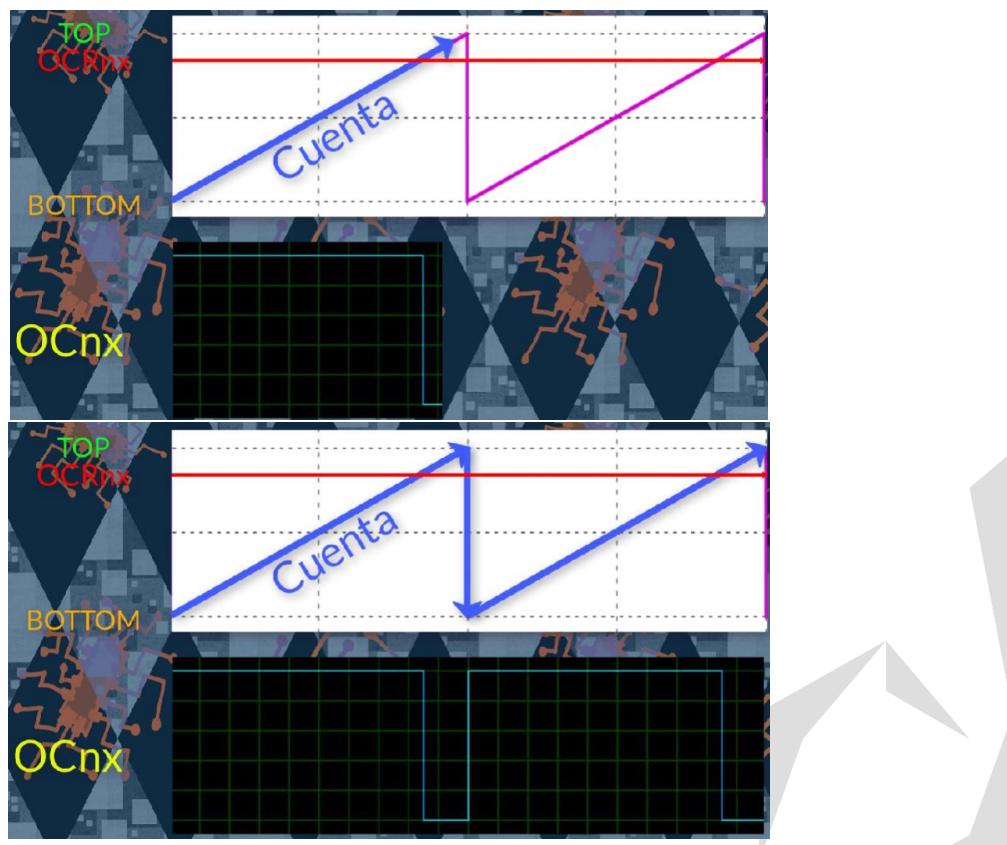
Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

TABLA 6.2 Modos de la generación de la forma de onda

WGM02	WGM01	WGM00	Descripción
0	0	0	Normal
0	0	1	PWM Fase Correcta
0	1	0	Limpia temporizador tras la comparación (CTC) OCRA
0	1	1	PWM Rápido
1	0	0	Reservado
1	0	1	PWM Fase Correcta OCRA
1	1	0	Reservado
1	1	1	PWM Rápido OCRA

En las señales PWM el registro comparador sirve para indicar el momento en el que se debe activar el modo alto en la señal de salida (o simplemente hacer un cambio de nivel lógico), la gran diferencia con el modo CTC es que cuando los registros **TCNT#** y **OCR#A** sean iguales, el valor de **TCNT#** no se irá a cero, sino que seguirá su conteo hasta que llegue a su **valor máximo** y cuando esto ocurra el contador se desborda y ahí si vuelve a valer 0, dictando así el porcentaje del ciclo de trabajo con el valor de **OCR#A**. De esta manera, el registro contador si tiene una forma de diente de sierra, pero cuando alcance el valor del registro comparador, cambiará de valor lógico, creando así el duty cycle. **El valor máximo del conteo puede ser redefinido por el registro OCRA o ICR1** o puede tener una resolución de 8, 9 o 10 bits, resolución se refiere al número máximo que puede alcanzar el conteo, ya que ni el temporizador T1 puede usar sus 16 bits en su conteo.



Una gran diferencia con las señales PWM en comparación con las CTC es que no necesitan de una función **ISR (Interruption Service Routine)** para crear la señal, simplemente se hace el cálculo para obtener cierto duty cycle y después se ve en el pin correspondiente la salida de la señal PWM, los pines de los canales A y B de cada temporizador son los siguientes:

- **T0:** Temporizador de 8 bits, siempre tiene resolución de 8 bits y su límite personalizado de conteo puede ser definido solo por el registro **OCRA**.
 - **OCR0B (Output Compare Register T0 B):** Registro del canal B conectado al **pin 5 (PD5)** perteneciente al **Puerto D** del Arduino por el cual se obtiene la **señal PWM**.
 - **OCR0A (Output Compare Register T0 A):** Registro del canal A conectado al **pin 6 (PD6)** perteneciente al **Puerto D** del Arduino por el cual se obtiene la **señal PWM**.
- **T1:** Temporizador de 16 bits, puede tener resolución de 8, 9 y 10 bits, su límite personalizado de conteo puede ser definido por el registro **OCRA** o **ICR1**.
 - **OCR1B (Output Compare Register T1 B):** Registro del canal B conectado al **pin 10 (PB2)** perteneciente al **Puerto B** del Arduino por el cual se obtiene la **señal PWM**.
 - **OCR1A (Output Compare Register T1 A):** Registro del canal A conectado al **pin 9 (PB1)** perteneciente al **Puerto B** del Arduino por el cual se obtiene la **señal PWM**.
- **T2:** Temporizador de 8 bits, siempre tiene resolución de 8 bits y su límite personalizado de conteo puede ser definido solo por el registro **OCRA**.
 - **OCR2B (Output Compare Register T2 B):** Registro del canal B conectado al **pin 3 (PD3)** perteneciente al **Puerto D** del Arduino por el cual se obtiene la **señal PWM**.
 - **OCR2A (Output Compare Register T2 A):** Registro del canal A conectado al **pin 11 (PB3)** perteneciente al **Puerto B** del Arduino por el cual se obtiene la **señal PWM**.

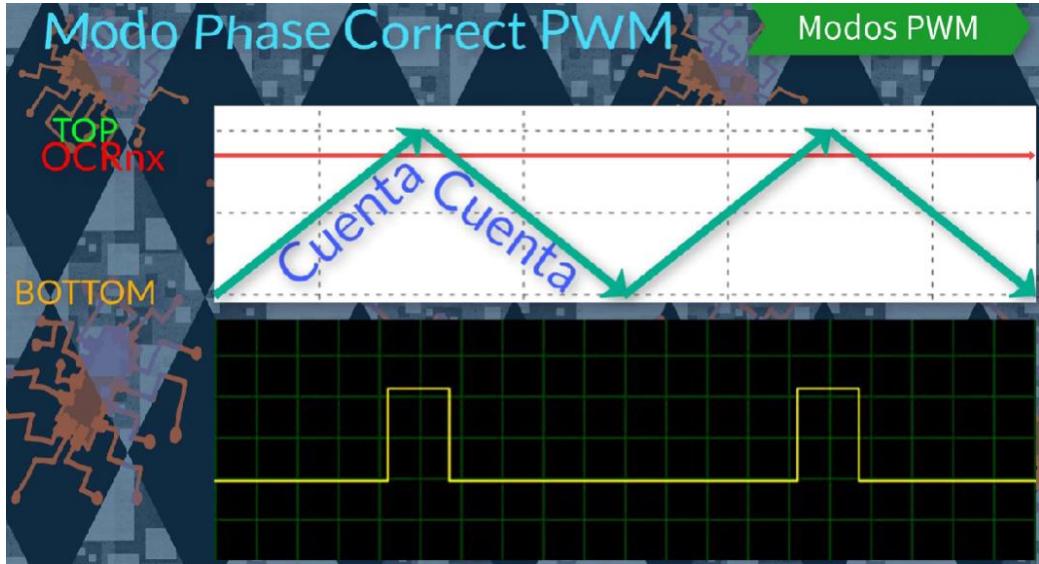
La forma en la que se crean los modos de PWM con el registro comparador es la siguiente:

PWM Fase Correcta

Realiza su cambio de valor lógico cuando **TCNT# = OCR#A**, pero no lleva a cabo un desborde cuando ocurre esta igualdad, sino que cuando llegue al valor máximo del contador, desciende su valor poco a poco hasta llegar nuevamente al cero. Crea una señal triangular de **baja frecuencia** y su conteo es ascendente y descendente contando durante todo el pico del triángulo de la señal, logrando así que el valor lógico del duty cycle se encuentre en la parte media del ciclo de la señal. NO se permite variar la frecuencia y se recomienda usarse en aplicaciones de: **Control de motores CD**.

- **T0 y T2:** Contadores de 8 bits.
 - En la tabla de **T0** y **T2** no se menciona nunca una resolución de un cierto número de bits, por lo que siempre el contador tendrá una resolución de 8 bits.
 - Se puede cambiar el valor máximo de conteo con el registro **OCRA**, a este nuevo valor máximo se le llama **TOP**.
- **T1:** Contador de 16 bits.
 - En la tabla de **T1** se menciona una resolución de un cierto número de bits, aunque siempre el contador es de 16 bits, el conteo para cambiar de estado lógico en la señal PWM se restringe a 8, 9 o 10 bits, por eso se pueden elegir estos modos.
 - Se puede cambiar el valor máximo de conteo con los registros **OCRA** o **ICR1**, a este nuevo valor máximo se le llama **TOP**.

- **OCRA** y **ICR1**: Utilizados mayormente para señales PWM con frecuencia constante.



Las fórmulas para obtener la frecuencia, el periodo y el ciclo de trabajo de la señal PWM son las siguientes, donde **TOP** es un valor máximo hipotético personalizado, si este no existe, se toma **OCR#**, que es el valor de conteo asignado al registro comparador **OCR#A** del canal A o **OCR#B** del canal B:

$$frecuencia = \frac{fmicro}{2 * escala * OCR\#} = \frac{fmicro}{2 * escala * TOP}$$

$$OCR\# = \frac{fmicro}{2 * escala * frecuencia}$$

$$Periodo = T = \frac{2 * escala * OCR\#}{fmicro} = \frac{2 * escala * TOP}{fmicro} = \frac{1}{frecuencia}$$

$$OCR\# = \frac{T * fmicro}{2 * escala}$$

$$t_{dutycycle} = T * Porcentaje_{deseado} = \frac{Porcentaje_{deseado}}{frecuencia}$$

$$OCR\#_{dutycycle} = \frac{t_{dutycycle} * fmicro}{2 * escala}$$

Se pueden utilizar los registros **OCRA** o **ICR1** para fijar un nivel máximo de conteo, esto se debe definir desde el modo de comparador que se elija, ya que en la tabla del contador **T1** los bits **WGM#3**, **WGM#2**, **WGM#1** y **WGM#0** definen un cierto tipo de comparador, podremos ver que el nombre de algunos comparadores PWM se repite, esto es porque el valor máximo del conteo es diferente en cada uno, ya sea porque tiene distinta resolución o porque se puede definir un valor máximo personalizado, específicamente en la columna de **TOP** de la tabla se verá cuál es el valor máximo al que puede llegar ese modo PWM en específico, que ya en la señal de salida ese valor máximo será la amplitud de la señal PWM y tendrá un nivel personalizado, para ello existen fórmulas igual, donde se reemplaza la variable **OCR#** de las ecuaciones anteriores por una nueva llamada **TOP** y esta a su vez debe ser reemplazada

en las ecuaciones originales para definir así nuevas variables de todo, en función del valor máximo definido.

La utilidad de esto también reside en que cuando el valor del conteo es demasiado grande para un contador en específico, se puede definir un valor máximo personalizado y de esta manera se podrá alcanzar la frecuencia o periodo deseado en la señal PWM, sin que no se pueda guardar su valor de conteo en el registro **OCRA** o **OCRB**, que almacena el valor de la variable ***OCR#dutycycle***.

$$TOP = \frac{f_{micro}}{2 * escala * frecuencia}$$

$$TOP = \frac{T * f_{micro}}{2 * escala}$$

La tabla para elegir este modo en el registro **TCCR#A (Timer Counter Control Register T# A)** y **TCCR#B (Timer Counter Control Register T# B)** es la siguiente para **T0** y **T2**:

WGM02	WGM01	WGM00	Descripción
0	0	0	Normal
0	0	1	PWM Fase Correcta
0	1	0	Limpia temporizador tras la comparación (CTC) OCRA
0	1	1	PWM Rápido
1	0	0	Reservado
1	0	1	PWM Fase Correcta OCRA
1	1	0	Reservado
1	1	1	PWM Rápido OCRA

TABLA 8.2 Modos de Comparación de la Salida, Modo PWM Fase Correcta

Recordemos que **T0** y **T2** tienen 3 bits de configuración, mientras **T1** tiene 4:

La tabla para elegir este modo en el registro **TCCR#A (Timer Counter Control Register T# A)** y **TCCR#B (Timer Counter Control Register T# B)** es la siguiente para T1:

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Los dos modos de configuración para las señales Fast PWM son los siguientes, donde se detalla la conmutación con la doble rampa de conteo, además esto afecta para saber el valor final de conteo que se debe asignar a los registros comparadores:

- **Modo no invertido (Cero lógico OC0):** La señal **inicia con estado alto** (1 lógico) y cuando el registro contador y comparador son iguales, $\text{TCNT\#} = \text{OCR\#A}$ cambia a de estado lógico, si estaba en 1 se hace 0 y viceversa. Con este modo el cálculo de OCR\#A para obtener un duty cycle específico se queda tal cual.
- **Modo invertido (Uno lógico OC0):** La señal **inicia con estado bajo** (0 lógico) y cuando el registro contador y comparador son iguales, $\text{TCNT\#} = \text{OCR\#A}$ cambia a de estado lógico, si estaba en 0 se hace 1 y viceversa. Con este modo el cálculo de OCR\#A para obtener un duty cycle específico se cambia, ya que se debe restar el máximo del contador menos el resultado del registro comparador obtenido.

COM1A1/ COM1B1	COM1A0/ COM1B0	Descripción
0	0	Aplica para los modos Phase Correct y Phase & Frequency Correct Operación de puerto normal. OC1A/OC1B desconectados.
0	1	Conmuta la señal PWM por OC1A en los modos 9 y 11. OC1B desconectado. Para cualquier otro modo el puerto funciona de forma normal y OC1A/OC1B están desconectados.
1	0	Nivel bajo por OC1A/OC1B cada coincidencia por comparación en la cuenta ascendente. Nivel alto por OC1A/OC1B cada coincidencia por comparación en la cuenta descendente. (Modo no invertido).
1	1	Nivel alto por OC1A/OC1B cada coincidencia por comparación en la cuenta ascendente. Nivel bajo por OC1A/OC1B cada coincidencia por comparación en la cuenta descendente. (Modo invertido).

COM0A1	COM0A0	Descripción
0	0	Operación Normal del Puerto (OC0 desconectado)
0	1	Cambio Lógico de OC0 (al igualar en la comparación)
1	0	Pone en cero lógico OC0 (al igualar en la comparación)
1	1	Pone en uno lógico OC0 (al igualar en la comparación)

Las fórmulas que describen el resultado dependiendo de si es modo invertido $\text{OCR\#I}_{dutycycle}$ o modo no invertido $\text{OCR\#NI}_{dutycycle}$ son las siguientes, donde el conteo máximo PWM depende de la resolución o de si se definió un valor máximo personalizado con los registros **OCRA** o **ICR1**:

$$\text{OCR\#I}_{dutycycle} = \text{ConteoMáximo}_{PWM} - \text{OCR\#}_{dutycycle}$$

$$\text{OCR\#NI}_{dutycycle} = \text{OCR\#}_{dutycycle}$$

TABLA 5.2 TIEMPOS DE LA FUENTE DE RELOJ PARA EL TEMPORIZADOR

CS02	CS01	CS00	Descripción
0	0	0	Ningún fuente de reloj (Temporizador detenido)
0	0	1	Sin pre escalador 1:1
0	1	0	Pre escalador a 1:8
0	1	1	Pre escalador a 1:64
1	0	0	Pre escalador a 1:256
1	0	1	Pre escalador a 1:1024
1	1	0	Fuente externa de reloj en T0 - Reloj en Flanco de bajada
1	1	1	Fuente externa de reloj en T0 - Reloj en Flanco de subida

Si se está utilizando el temporizador **T0** y **T2**, solo se podrá utilizar el registro **OCRA** para fijar el valor máximo de conteo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** y el resultado de la variable **OCR#dutycycle** en el registro **OCR#B**, pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar:

$$OCR\#A = TOP$$

$$OCR\#A = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

Si se está utilizando el temporizador **T1**, se podrá utilizar cualquiera de los dos registros **OCRA** o **ICR1** para fijar el nuevo valor del conteo máximo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** o **ICR1** y el resultado de la variable **OCR#dutycycle** en el registro **OCR#A** o **OCR#B** (**Si se eligió ICR1 para fijar TOP**) u solamente **OCR#B** (**Si se eligió OCR#A para fijar TOP**), pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar y será explicado más adelante:

$$ICR1 \text{ o } OCR\#A = TOP$$

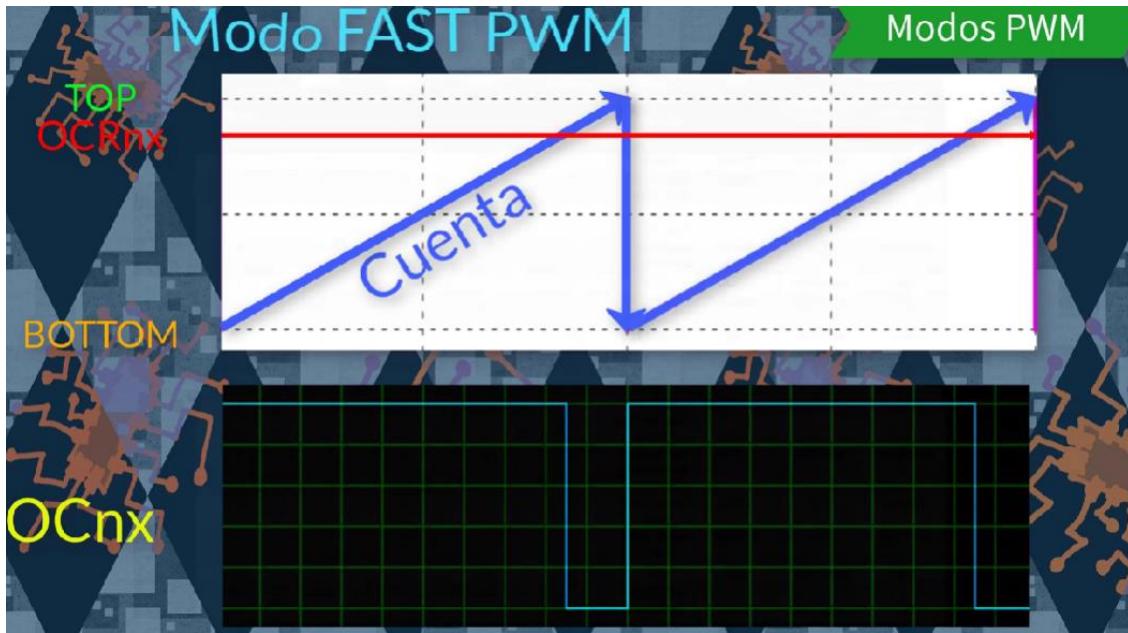
$$OCR\#A \text{ u } OCR\#B = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

PWM Rápido o Fast PWM

Realiza su cambio de valor lógico cuando **TCNT# = OCR#A**, pero lleva a cabo su desborde cuando se llega al valor máximo del contador, volviendo a contar desde cero. Crea una señal de diente de sierra de **alta frecuencia** muy similar a la del comparador CTC y su conteo es ascendente solamente. Si se permite variar la frecuencia. Se recomienda usarse en aplicaciones de: **Rectificación de señales, regulación de tensión, conversión de señal digital a analógica, convertidor CD-CD, etc.**

- **T0 y T2:** Contadores de 8 bits.
 - En la tabla de **T0** y **T2** se menciona una resolución de un cierto número de bits, aunque siempre el contador es de 8 bits, el conteo para PWM se restringe para 8, 9 o 10 bits, por eso se pueden elegir estos modos.
 - Se puede cambiar el valor máximo de conteo con el registro **OCRA**, a este nuevo valor máximo se le llama **TOP**.
- **T1:** Contador de 16 bits.

- En la tabla de **T1** se menciona una resolución de un cierto número de bits, aunque siempre el contador es de 16 bits, el conteo para cambiar de estado lógico en la señal PWM se restringe a 8, 9 o 10 bits, por eso se pueden elegir estos modos.
- Se puede cambiar el valor máximo de conteo con los registros **OCRA** o **ICR1**, a este nuevo valor máximo se le llama **TOP**.
 - **OCRA** y **ICR1**: Utilizados mayormente para señales PWM con frecuencia constante.



Las fórmulas para obtener la frecuencia, el periodo y el ciclo de trabajo de la señal PWM son las siguientes, donde **TOP** es un valor máximo hipotético personalizado, si este no existe, se toma **OCR#**, que es el valor de conteo asignado al registro comparador **OCR#A** del canal A o **OCR#B** del canal B:

$$\text{frecuencia} = \frac{f_{\text{micro}}}{\text{escala} * (1 + \text{OCR}\#)} = \frac{f_{\text{micro}}}{\text{escala} * (1 + \text{TOP})}$$

$$\text{OCR}\# = \frac{f_{\text{micro}}}{\text{frecuencia} * \text{escala}} - 1$$

$$\text{Periodo} = T = \frac{\text{escala} * (1 + \text{OCR}\#)}{f_{\text{micro}}} = \frac{\text{escala} * (1 + \text{TOP})}{f_{\text{micro}}} = \frac{1}{\text{frecuencia}}$$

$$\text{OCR}\# = \frac{T * f_{\text{micro}}}{\text{escala}} - 1$$

$$t_{\text{duty cycle}} = T * \text{Porcentaje}_{\text{deseado}}$$

$$\text{OCR}\#_{\text{duty cycle}} = \frac{t_{\text{duty cycle}} * f_{\text{micro}}}{\text{escala}} - 1$$

Se pueden utilizar los registros **OCR#A** o **ICR1** para fijar un nivel máximo de conteo, esto depende del modo que se elija en la tabla con los registros **TCCR#A** y **TCCR#B**, este nuevo valor máximo indicará la amplitud de la señal PWM y será un nivel personalizado, para ello existen fórmulas igual, donde se

reemplaza la variable **OCR#** de las ecuaciones anteriores por una nueva llamada **TOP** y esta se asigna a los registros **OCR#A** o **ICR1** al final.

Además, viendo las tablas nos podemos dar cuenta que para los temporizadores **T0** y **T2** solo se puede usar el registro **OCR#A** para fijar un valor máximo personalizado, mientras que para **T1** se pueden usar los dos registros **OCR#A** o **ICR1**:

$$TOP = \frac{f_{micro}}{frecuencia * escala} - 1$$

$$TOP = \frac{T * f_{micro}}{escala} - 1$$

La tabla para elegir este modo en el registro **TCCR#A** (**Timer Counter Control Register T# A**) es la siguiente para **T0** y **T2**:

WGM02	WGM01	WGM00	Descripción
0	0	0	Normal
0	0	1	PWM Fase Correcta
0	1	0	Limpia temporizador tras la comparación (CTC) OCRA
0	1	1	PWM Rápido
1	0	0	Reservado
1	0	1	PWM Fase Correcta OCRA
1	1	0	Reservado
1	1	1	PWM Rápido OCRA

La tabla para elegir este modo en el registro **TCCR#A** (**Timer Counter Control Register T# A**) es la siguiente para **T1**, tiene muchas más configuraciones que la tabla de **T0** y **T2** debido a que se consideran varias resoluciones del valor de conteo máximo llamado TOP y se puede definir de igual manera ese valor máximo por medio de dos registros en vez de uno, esto nos da la posibilidad de definir un valor máximo y aún ser capaces de usar los dos canales A y B para crear señales PWM.

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Los dos modos de configuración para las señales Fast PWM son los siguientes, donde se detalla la conmutación con la doble rampa de conteo:

Los modos se configuran en el registro **TCCR#A**:

TCCR1A								Registros PWM	
Bit	7	6	5	4	3	2	1	0	
Access	COM1A1	COM1A0	COM1B1	COM1B0			WGM11	WGM10	
Reset	R/W	R/W	R/W	R/W			R/W	R/W	0 0

Comutación:

La señal PWM alterna con el estado anterior cada coincidencia entre TCNT1 y OCR1A. Esto da un ciclo de trabajo del 50%.

Nivel alto: La señal PWM se pone en alto cada coincidencia entre TCNT1 y OCRnx. Éste sería el modo invertido.

Nivel bajo: La señal PWM se pone en bajo cada coincidencia entre TCNT1 y OCRnx. Éste sería el modo no invertido.

Registro e Control del Temporizador/Contador (TCCR0A)

BIT	7	6	5	4	3	2	1	0
COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00	

Registro e Control del Temporizador/Contador (TCCR0B)

BIT	7	6	5	4	3	2	1	0
FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00	

Los dos modos de configuración para las señales Fast PWM son los siguientes:

- **Modo no invertido:** La señal inicia con estado alto (1 lógico) y cuando el registro contador y comparador son iguales, **TCNT# = OCR#A** cambia a de estado lógico, si estaba en 1 se hace 0 y viceversa (cero lógico OC0).



- **Modo invertido:** La señal inicia con estado bajo (0 lógico) y cuando el registro contador y comparador son iguales, $TCNT\# = OCR\#A$ cambia a de estado lógico, si estaba en 0 se hace 1 y viceversa (uno lógico OC0).



Las fórmulas que describen el resultado dependiendo de si es modo invertido $OCR\#I_{dutycycle}$ o modo no invertido $OCR\#NI_{dutycycle}$ son las siguientes, donde el conteo máximo PWM depende de la resolución o de si se definió un valor máximo personalizado con los registros **OCRA** o **ICR1**:

$$OCR\#I_{dutycycle} = \text{ConteoMáximo}_{PWM} - OCR\#_{dutycycle}$$

$$OCR\#NI_{dutycycle} = OCR\#_{dutycycle}$$

Si se está utilizando el temporizador **T0** y **T2**, solo se podrá utilizar el registro **OCRA** para fijar el valor máximo de conteo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** y el resultado de la variable $OCR\#_{dutycycle}$ en el registro **OCR#B**, pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar:

$$OCR\#A = TOP$$

$$OCR\#A = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

Si se está utilizando el temporizador **T1**, se podrá utilizar cualquiera de los dos registros **OCRA** o **ICR1** para fijar el nuevo valor del conteo máximo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** o **ICR1** y el resultado de la variable $OCR\#_{dutycycle}$ en el registro **OCR#A** o **OCR#B** (**Si se eligió ICR1 para fijar TOP**) u solamente **OCR#B** (**Si se eligió OCRA para fijar TOP**), pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar y será explicado más adelante:

$$ICR1 \text{ o } OCR\#A = TOP$$

$$OCR\#A \text{ u } OCR\#B = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

COM1A1/ COM1B1	COM1A0/ COM1B0	Descripción Aplica para modos Fast PWM
0	0	Operación de puerto normal. OC1A/OC1B desconectados.
0	1	Conmuta la señal PWM por OC1A en los modos 14 y 15. OC1B desconectado. Para cualquier otro modo el puerto funciona de forma normal y OC1A/OC1B están desconectados.
1	0	Nivel bajo por OC1A/OC1B cada coincidencia por comparación. Nivel alto por OC1A/OC1B cada que se llega a BOTTOM. (Modo no invertido).
1	1	Nivel alto por OC1A/OC1B cada coincidencia por comparación. Nivel bajo por OC1A/OC1B cada que se llega a BOTTOM. (Modo invertido).

Las tablas son las mismas y son descripciones de las configuraciones alcanzadas con los bits **COM#A1**, **COM#A0**, **COM#B1** y, **COM#B0** del registro **TCCR#A**. Pero la primera es para las tablas del registro **T1** y la segunda para los registros **T0** y **T2**.

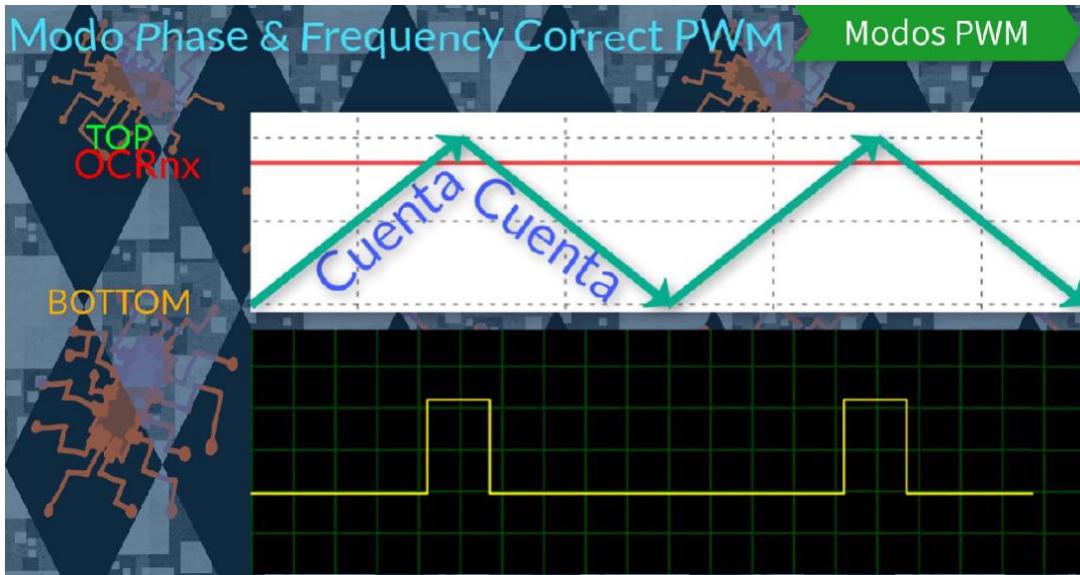
TABLA 6.1 Modos de Comparación de la Salida

COM0A1	COM0A0	Descripción
0	0	Operación Normal del Puerto (OC0 desconectado)
0	1	Cambio Lógico de OC0 (al igualar en la comparación)
1	0	Pone en cero lógico OC0 (al igualar en la comparación)
1	1	Pone en uno lógico OC0 (al igualar en la comparación)

PWM Fase y Frecuencia Correcta

Es muy parecido al modo **fase correcta**, ya que realiza su cambio de valor lógico cuando **TCNT# = OCR#A**, no lleva a cabo un desborde, sino que cuando llegue al valor máximo del contador, desciende de valor poco a poco hasta llegar nuevamente al cero. Crea una señal triangular de **baja frecuencia** y su conteo es ascendente y descendente contando durante todo el pico del triángulo de la señal. Su mayor diferencia es que si permite frecuencias variables y que a fuerza se debe declarar un valor máximo con los registros **OCRA** o **ICR1**, por lo que solo se puede usar con el temporizador **T1**.

- **T1:** Contador de 16 bits.
 - En la tabla de **T1** se menciona una resolución de un cierto número de bits, aunque siempre el contador es de 16 bits, el conteo para cambiar de estado lógico en la señal PWM se restringe a 8, 9 o 10 bits, por eso se pueden elegir estos modos.
 - Se puede cambiar el valor máximo de conteo con los registros **OCRA** o **ICR1**, a este nuevo valor máximo se le llama **TOP**.
 - **OCRA** y **ICR1**: Utilizados mayormente para señales PWM con frecuencia constante.



Antes que nada, se deben utilizar los registros **OCR#A** o **ICR1** para fijar un nivel máximo de conteo, este nuevo valor máximo afectará al valor del conteo para obtener una frecuencia y duty cycle específico:

$$TOP = \frac{fmicro}{2 * escala * frecuencia}$$

$$TOP = \frac{T * fmicro}{2 * escala}$$

Las fórmulas para obtener la frecuencia, el periodo y el ciclo de trabajo de la señal PWM son las siguientes, donde **TOP** es un valor máximo hipotético personalizado, si este no existe, se toma **OCR#**, que es el valor de conteo asignado al registro comparador **OCR#A** del canal A o **OCR#B** del canal B:

$$frecuencia = \frac{fmicro}{2 * escala * TOP}$$

$$Periodo = T = \frac{2 * escala * TOP}{fmicro} = \frac{1}{frecuencia}$$

$$t_{dutycycle} = T * Porcentaje_{deseado} = \frac{Porcentaje_{deseado}}{frecuencia}$$

$$OCR\#_{dutycycle} = \frac{t_{dutycycle} * fmicro}{2 * escala}$$

Las fórmulas que describen el resultado dependiendo de si es modo invertido **OCR#I_{dutycycle}** o modo no invertido **OCR#NI_{dutycycle}** son las siguientes, donde el conteo máximo PWM depende del valor máximo personalizado calculado con los registros **OCRA** o **ICR1**:

$$OCR\#I_{dutycycle} = ConteoMáximo_{PWM} - OCR\#_{dutycycle}$$

$$OCR\#NI_{dutycycle} = OCR\#_{dutycycle}$$

Si se está utilizando el temporizador **T0** y **T2**, solo se podrá utilizar el registro **OCRA** para fijar el valor máximo de conteo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** y el

resultado de la variable $OCR\#_{dutycycle}$ en el registro **OCR#B**, pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar:

$$OCR\#A = TOP$$

$$OCR\#A = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

Si se está utilizando el temporizador **T1**, se podrá utilizar cualquiera de los dos registros **OCRA** o **ICR1** para fijar el nuevo valor del conteo máximo, el resultado de lo obtenido en la variable **TOP** se almacenará en **OCR#A** o **ICR1** y el resultado de la variable $OCR\#_{dutycycle}$ en el registro **OCR#A** o **OCR#B**

(Si se eligió ICR1 para fijar TOP) u solamente **OCR#B** (**Si se eligió OCR#A para fijar TOP**), pero solo hasta después de haber indicado cuál es el tipo de salida PWM (Modo invertido o no invertido) que se quiere implementar y será explicado más adelante:

$$ICR1 \text{ o } OCR\#A = TOP$$

$$OCR\#A \text{ u } OCR\#B = OCR\#I_{dutycycle} \text{ o } OCR\#NI_{dutycycle}$$

En la tabla para temporizadores de 8 bits **T0** y **T2** no existe este modo, solamente para el registro el registro **TCCR#A** (**Timer Counter Control Register T# A**) para el temporizador **T1**:

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, phase correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, phase correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, phase correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, phase and frequency correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, phase and frequency correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, phase correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, phase correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	-	-	-
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP



COM1A1/ COM1B1	COM1A0/ COM1B0	Descripción
0	0	Aplica para los modos Phase Correct y Phase & Frequency Correct Operación de puerto normal. OC1A/OC1B desconectados.
0	1	Conmuta la señal PWM por OC1A en los modos 9 y 11. OC1B desconectado. Para cualquier otro modo el puerto funciona de forma normal y OC1A/OC1B están desconectados.
1	0	Nivel bajo por OC1A/OC1B cada coincidencia por comparación en la cuenta ascendente. Nivel alto por OC1A/OC1B cada coincidencia por comparación en la cuenta descendente. (Modo no invertido).
1	1	Nivel alto por OC1A/OC1B cada coincidencia por comparación en la cuenta ascendente. Nivel bajo por OC1A/OC1B cada coincidencia por comparación en la cuenta descendente. (Modo invertido).

COM0A1	COM0A0	Descripción
0	0	Operación Normal del Puerto (OC0 desconectado)
0	1	Cambio Lógico de OC0 (al igualar en la comparación)
1	0	Pone en cero lógico OC0 (al igualar en la comparación)
1	1	Pone en uno lógico OC0 (al igualar en la comparación)

En conclusión, para poder activar el modo de timer por comparación tipo PWM, se deben modificar los siguientes registros para acceder a los distintos temporizadores comparadores de T0, T1 y T2:

- **TCCR#A:** Registro **Timer Counter Control Register T# A:**
 - **T0 y T2:** 3 bits de configuración para elegir el **tipo de temporizador comparador** que se quiere usar.
 - Registro **TCCR#A:** Bits **WGM#1** y **WGM#0**.
 - Registro **TCCR#B:** Bit **WGM#2**.
 - **T1:** 4 bits de configuración para elegir el **tipo de temporizador comparador** que se quiere usar.
 - Registro **TCCR#A:** Bits **WGM#1** y **WGM#0**.
 - Registro **TCCR#B:** Bits **WGM#3** y **WGM#2**.
 - Los bits **COM#A1** y **COM#A0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#A**.
 - Los bits **COM#B1** y **COM#B0** del registro **TCCR#A** indican el tipo de señal obtenida en la salida **OCR#B**.
- **TCCR#B:** Registro **Timer Counter Control Register T# B:**
 - **T0 y T2:** 3 bits de configuración para elegir el **tipo de temporizador comparador** que se quiere usar.
 - Registro **TCCR#B:** Bit **WGM#2**.
 - Registro **TCCR#A:** Bits **WGM#1** y **WGM#0**.
 - **T1:** 4 bits de configuración para elegir el **tipo de temporizador comparador** que se quiere usar.
 - Registro **TCCR#B:** Bits **WGM#3** y **WGM#2**.
 - Registro **TCCR#A:** Bits **WGM#1** y **WGM#0**.

- Los bits **CS## (Clock Select)** del registro **TCCR#B** indican la escala del conteo o si se usa un temporizador interno o externo.
- **TIMSK#**: Registro **Timer Interrupt Mask Register T#**:
 - El bit **TOIE# (Timer Overflow Interruption Enable)** indica si el contador será de tipo desbordamiento (**1 lógico**) o comparación (**0 lógico**).
 - El bit **OCIE#A** activa la interrupción por comparación del registro **OCR#A (Output Compare Register T# A)** que sirve para fijar el límite de conteo del temporizador por comparación.
 - El bit **OCIE#B** activa la interrupción por comparación del registro **OCR#B**.
- **TCNT0, TCNT1 y TCNT2**: Son los registros que realizan el conteo de 0 a 255 para T0 y T2 o de 0 a 65,535 para T1.
- **OCRA** o **ICR1**: Con estos registros se puede declarar un nuevo valor máximo, dependiendo del modo comparador que se elija, específicamente en la columna TOP de la tabla viene si se puede declarar un valor máximo personalizado y con qué registro.
 - **T0 y T2**: Los contadores de 8 bits solo pueden declarar un valor máximo personalizado con el registro **OCRA**. En este caso la señal PWM saldría por el canal B en el puerto **OCRB**.
 - **T1**: El contador de 16 bits puede usar cualquiera de los dos registros **OCRA** o **ICR1** para declarar un valor máximo personalizado.
- **TIMER#_COMP_A_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#A = TCNT#**.
- **TIMER#_COMP_B_vect**: Vector de interrupción dado a la función **ISR (Interrupt Service Routine)** para que la interrupción sea activada por un temporizador por comparación del registro **OCR#B = TCNT#**.
- **TIFR#**: Registro **Timer Interruption Flag Register** que levanta una bandera por cada temporizador T0, T1 o T2 que indica cuando ya se haya hecho una interrupción por temporizador.
- **GTCCR (General Timer Counter Control Register)**: Registro usado principalmente cuando se está usando más de 1 temporizador a la vez, lo que hace es reiniciar sus conteos y escalas, logrando su sincronización.



Puertos B, C y D Físicos en el Arduino

PORTB*

11111111



PORTC*

11111111

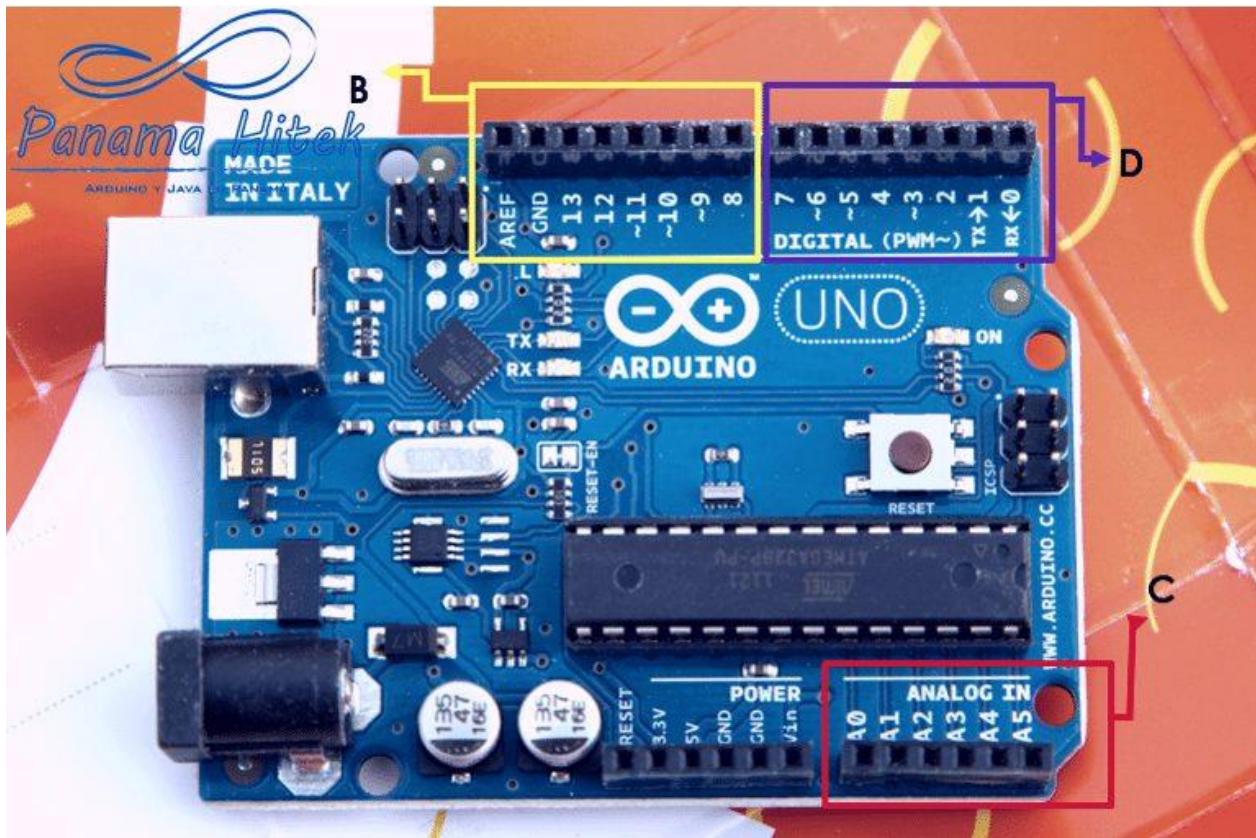


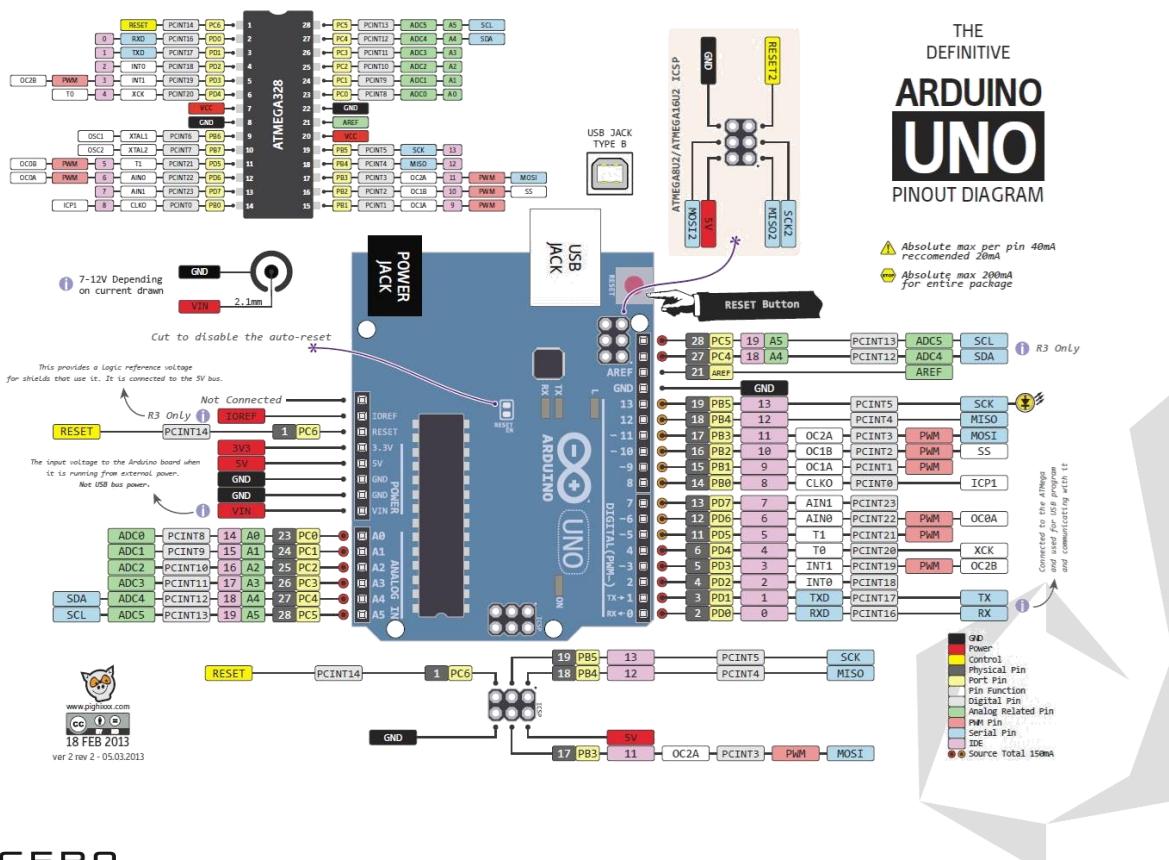
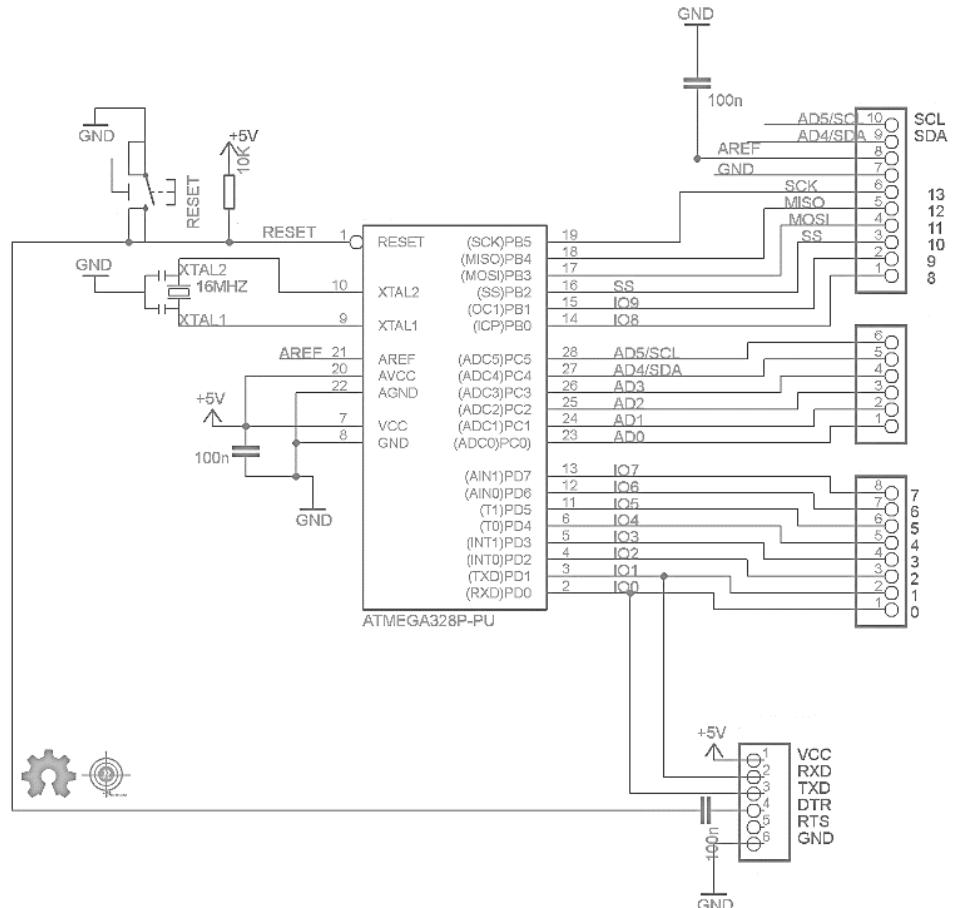
PORTD

11111111



Los pines de mayor peso de los puertos **B** y **C** no se usan, por lo cual en los puertos **B** y **C** existen **6 pines** y en el puerto **D** existen **8 pines**, los pines de cada una de las 3 familias de puertos de la placa Arduino UNO se muestran en la siguiente imagen.

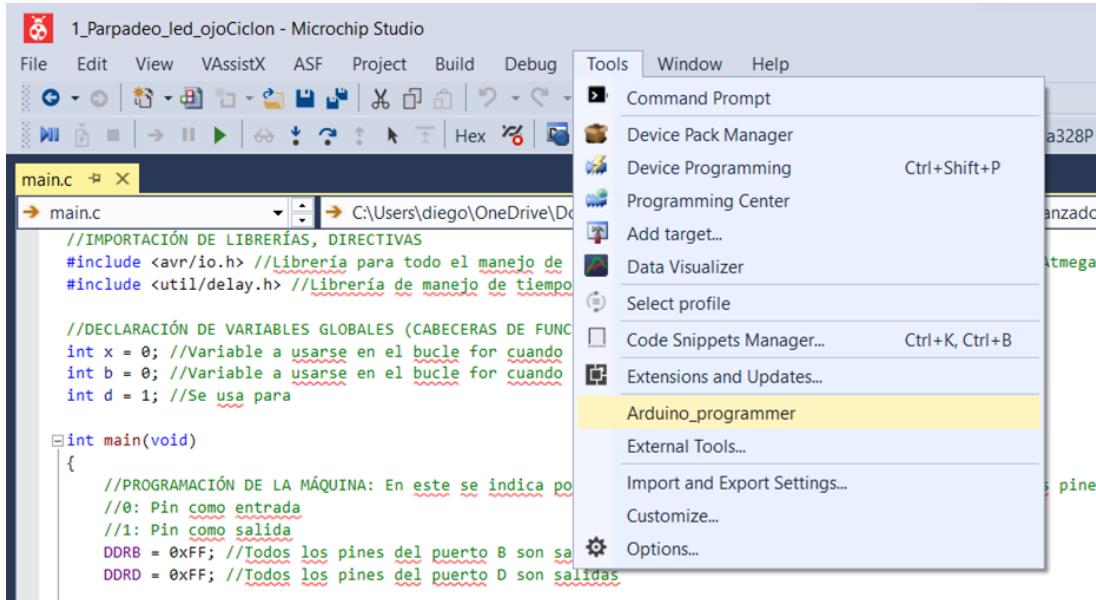




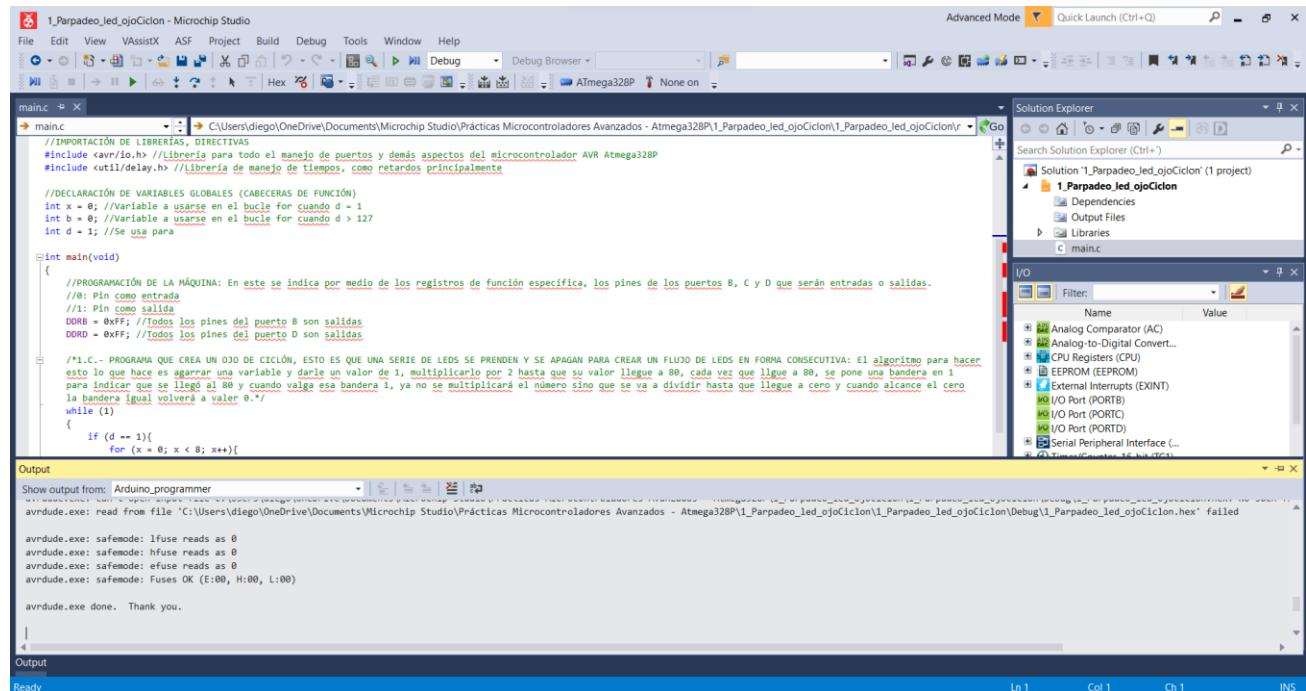
Programar Físicamente la Placa Arduino UNO con el Programa de Microchip Studio

Recordemos que la forma de programar la placa del Arduino UNO, ya habiendo realizado la configuración de su programador en el programa de Microchip Studio es seleccionando la opción de:

Tools → Arduino programmer (opción creada y nombrada cuando se configuró el Arduino).

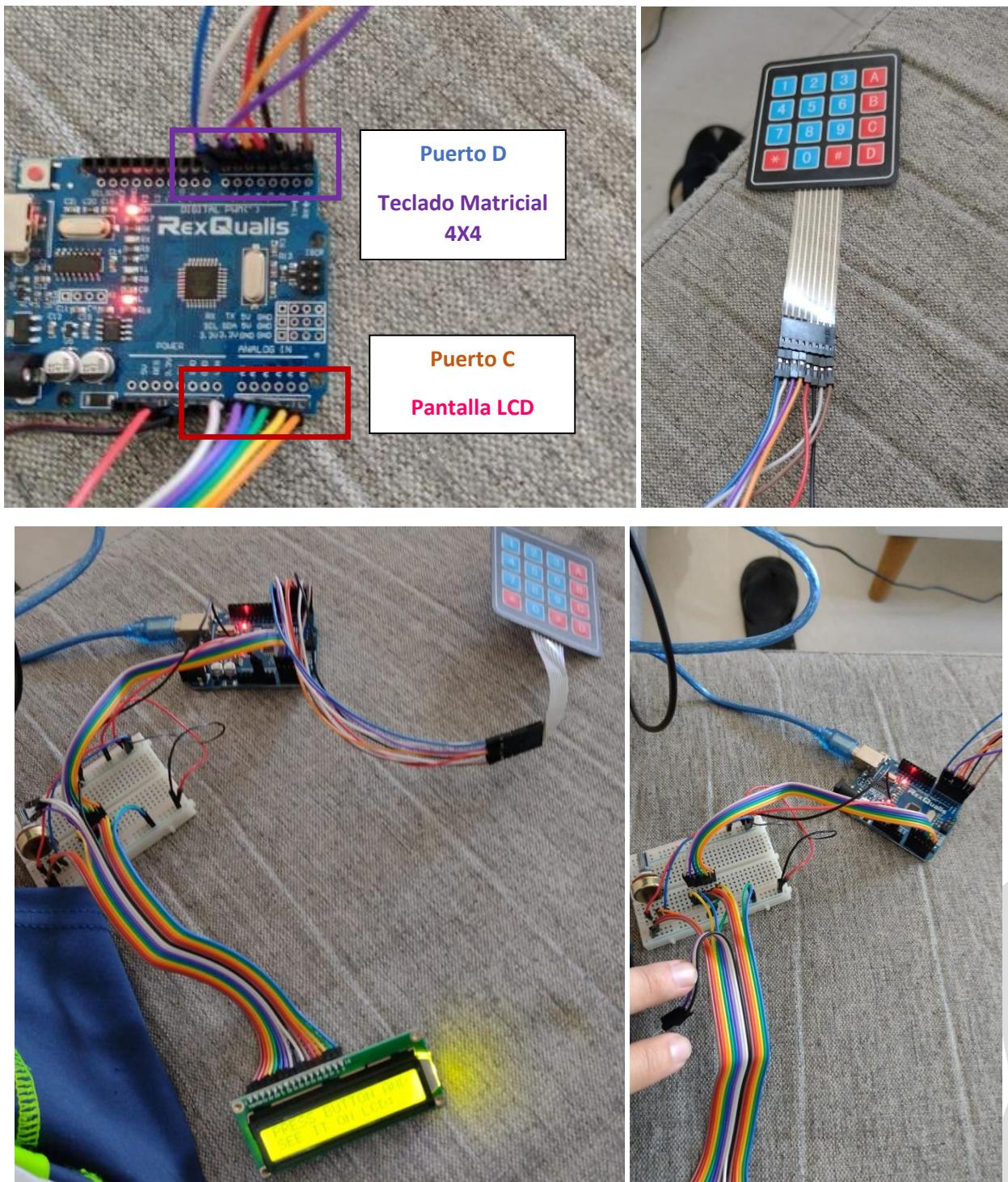


Es muy importante mencionar que cuando se carga un código a una placa Arduino, debemos tener cuidado que no haya nada conectado a los pines **TX** y **RX** del Puerto **D** porque si no Microchip Studio indicará un error cuando tratemos de subir el programa, ya habiéndolo subido podemos utilizar esos pines de nuevo y no habrá problema.



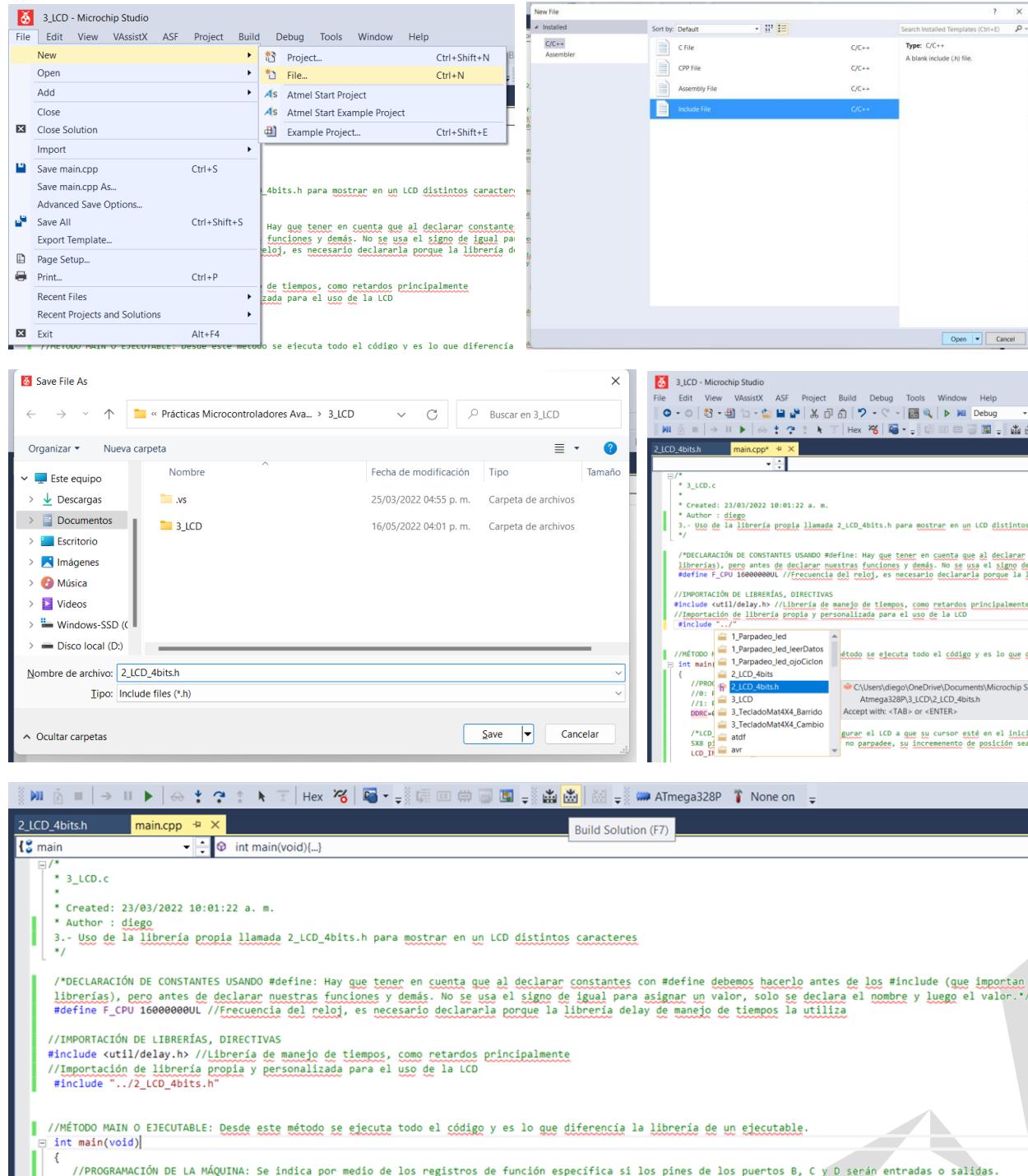
Para la creación de proyectos controlados con la placa Arduino UNO que además controlan más de un dispositivo periférico como lo es el teclado matricial 4X4, una pantalla LCD, etc. Normalmente se utilizarán los siguientes puertos para programar los dispositivos enlistados a continuación:

- **Puerto B**: Periféricos adicionales, como motores, sensores, etc.
- **Puerto C**: Pantalla LCD.
- **Puerto D**: Teclado matricial 4X4.



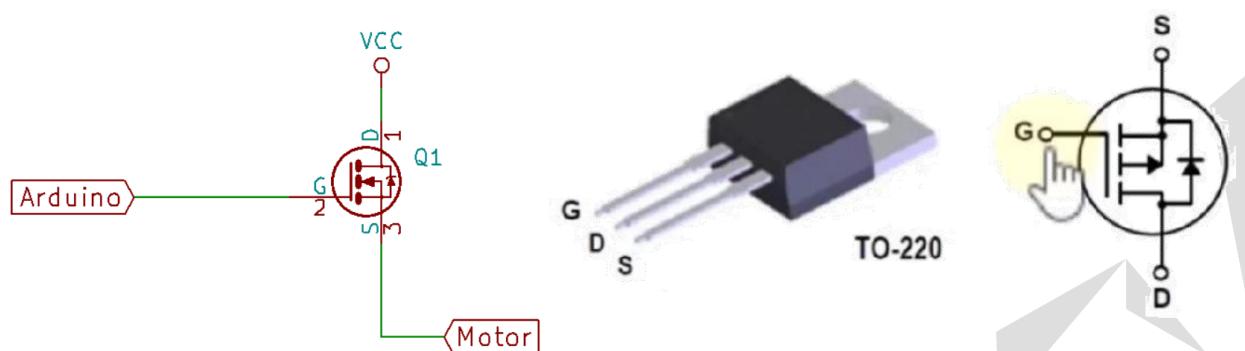
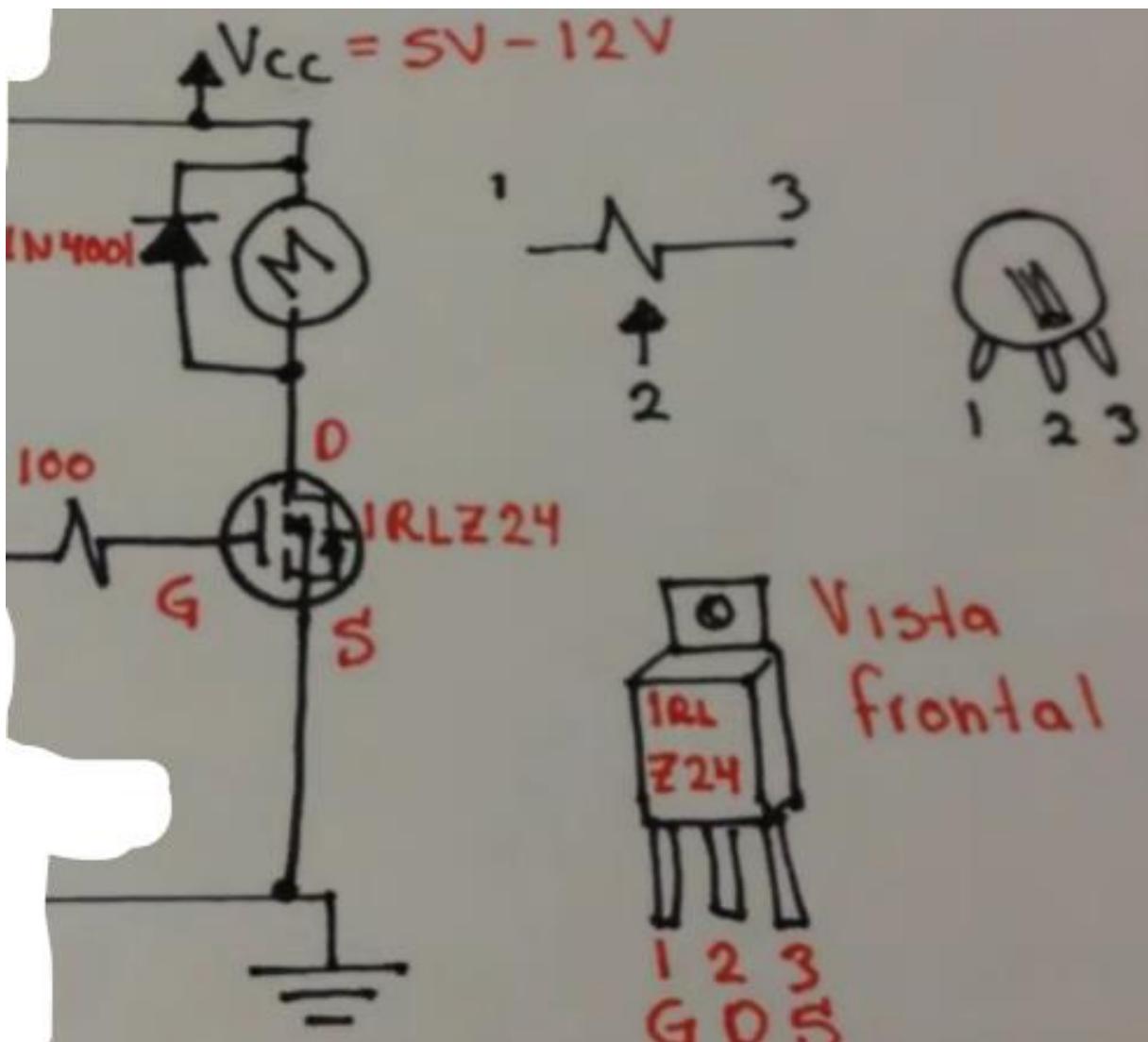
Utilización de Librerías en Microchip Studio: Control de Pantalla LCD

Cuando se quiera utilizar una librería propia para usarse al ejecutar el código de control de un LCD se deben seguir los siguientes pasos para crear un archivo `#include` con extensión .h y que pueda ser importado: `#include "../forma/de/llegar/al/archivo/nombre librería personalizada.h"`



Conexión Física del Circuito: Motor con control de señal PWM

Para controlar un motor se debe realizar la siguiente conexión, tomando en cuenta que se usa un MOSFET y un diodo de marcha libre solamente.



La resistencia R conectada entre las terminales S (Source) y D (Drain) del MOSFET puede ser de 100 o 220 Ω.

Referencias:

Electronoobs, “Interrupciones Por Cambio de Pin ISR | PCINT | Arduino101”, 2021 [Online], Available: <https://www.youtube.com/watch?v=Tbqdu2vrQ2E&t=42s>

Electronoobs, “Interrupciones por Temporizador ISR + Ejemplos | Arduino101 | Registros y Modos”, 2021 [Online], Available: <https://www.youtube.com/watch?v=Whmb9zAmpuk&t=8s>

