12-5-2017

Practica 10

Generador de frecuencia mediante los temporizadores del uC ATmega1280

CARLOS OMAR CALDERON MEZA
INGENIERO EN COMPUTACION

Objetivo:

Mediante esta práctica el alumno aprenderá la programación y uso avanzado del Temporizador 0 y 2 del microcontrolador ATmega1280.

Material:

- Computadora Personal (con AVR Studio)
- Tarjeta T-Juino.
- Componentes Electrónicos.

Equipo:

- Computadora Personal con USB, AVRStudio y WinAVR

Teoría:

- Teoría Básica de Música.
- Programación del Timer2 del microcontrolador como generador de frecuencia (Diagrama, Funcionamiento, Registros de configuración y operación)

Teoría básica de música

La teoría musical es un campo de estudio que tiene por objeto la investigación de los diversos elementos de la música, entre ellos el desarrollo y la metodología para analizar, escuchar, comprender y componer música.

Mientras que la musicología puede incluir cualquier declaración, creencia o concepción de lo que es la música, la teoría musical está limitada a las discusiones concernientes a los eventos sincrónicos (o diacrónicos) de una composición específica (o varias composiciones), y a los capítulos músico-teoréticos abstractos (por ejemplo teoría de conjuntos, teoría de grupos, teoría de tensión tonal, etc.).

Una persona especializada en teoría musical es un *teórico musical*.

La notación musical Abc es un lenguaje para escribir música que utiliza el conjunto de caracteres ASCII. Fue inicialmente creado por Chris Walshaw. Si bien es un lenguaje musical basado en ordenadores, uno de los principales objetivos ha sido que pueda ser leído con facilidad por los humanos. Inicialmente fue desarrollado para ser utilizado con composiciones folk y melodías tradicionales provenientes del oeste de Europa (inglesas, irlandesas, escocesas) que por lo general son melodías de una sola voz que pueden ser escritas en un solo pentagrama en notación estándar. La sintaxis también permite utilizar metadata para cada tono.[1]

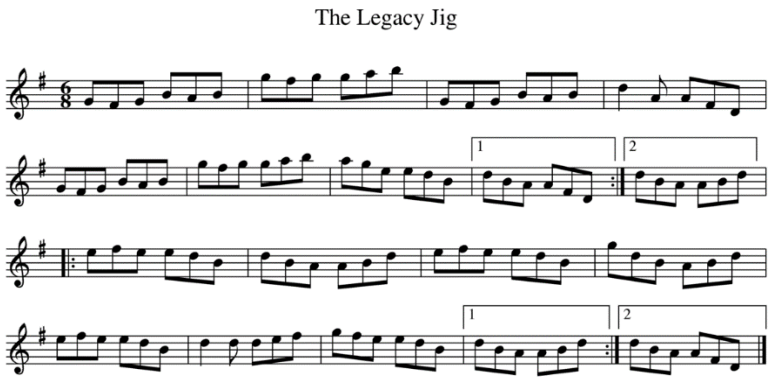
Dado que el sistema abc está basado en los caracteres ASCII, se puede utilizar cualquier editor de texto para editar música. Sin embargo, existen varios paquetes de software con diversas facilidades que permiten leer y procesar música escrita en sistema abc. La mayoría de este software es de distribución libre o shareware, y se encuentran disponibles para los diversos sistemas operativos existentes tales como Microsoft Windows, Unix/Linux, Macintosh, PalmOS, y aquellos que soportan la web.

Posteriormente otros paquetes de software han provisto salida directa (evitando la tipografía TeX), y han extendido la sintaxis para permitir la presentación de la letra de la canción alineada con las notas, múltiples voces y notación con múltiples pentagramas, tablatura, y MIDI.

El siguiente ejemplo ilustra el uso de la notación musical abc

```
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd |:
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
```

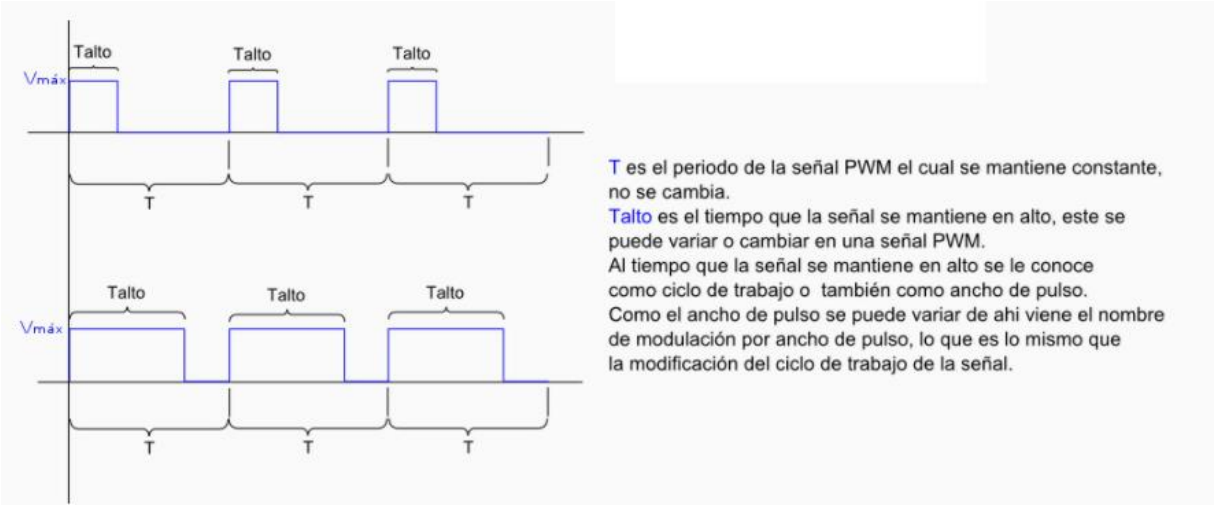
Las líneas en la primera parte de la notación de la melodia, que comienzan con una letra seguida por el signo de dos puntos, indican diversos aspectos de la melodia tales como el index, cuando hay más de una melodía en un archivo (X:), el título (T:), el tipo de melodía (R:), el compás (M:), la duración por omisión de las notas musicales (L:) y la clave (K:). Las líneas que siguen a la indicación de la clave con la melodía propiamente dicha. Este ejemplo puede ser traducido a notación musical tradicional utilizando una de las herramientas de conversión abc existentes. Por ejemplo, el software abcm2ps produce una salida que es similar a la que se muestra en la imagen a continuación:



Timer 2 como generador de frecuencia

PWM timer2 AVR, Con el timer2 del microcontrolador AVR se pueden generar 2 tipos de señales PWM, PWM en modo rápido y PWM en modo fase correcta, en este caso se comentará como obtener la señal PWM modo rápido timer2 AVR.

Una señal PWM es una onda rectangular de periodo fijo y como la frecuencia es la inversa del periodo entonces también será de frecuencia fija, lo que si se cambiará o modificará normalmente en una señal PWM es el tiempo en alto que estará su valor máximo Vmáx, a este tiempo en alto se le llama ancho de pulso, por lo que en una señal PWM se modificará su ancho pulso.



Si el ancho de pulso de la señal PWM se representa en forma de porcentaje se le suele llamar ciclo de trabajo, el ciclo de trabajo se obtienen mediante la siguiente relación:

Ciclo de trabajo=(Talto/T)*100%

El ciclo de trabajo puede ser desde un 0% cuando el Talto=0, hasta un 100% cuando el Talto es igual al periodo de la señal PWM Talto=T, si el tiempo en alto es igual a la mitad del periodo entonces el ciclo de trabajo será del 50%.

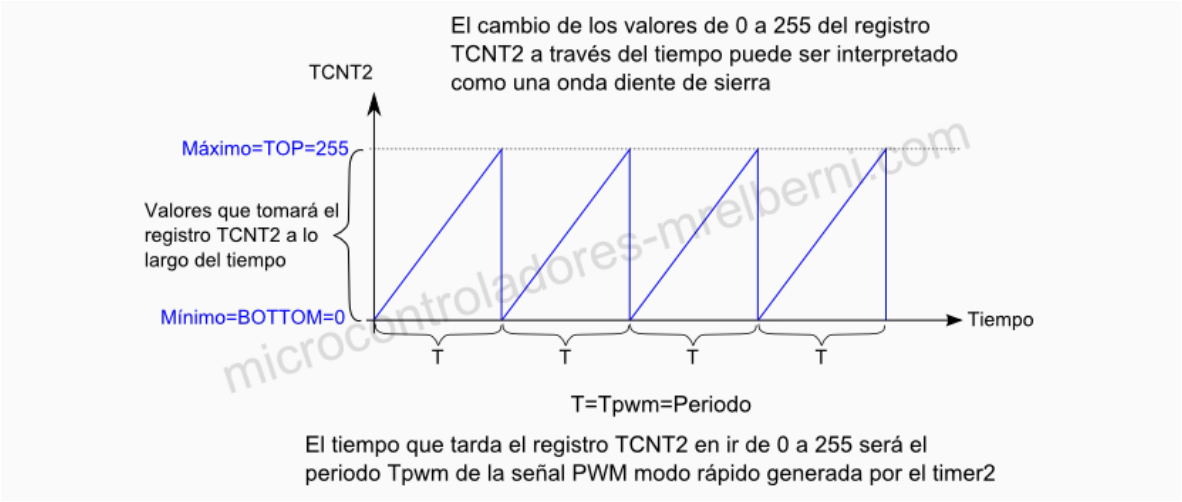
Al proceso de cambiar o modificar el tiempo en alto de la señal PWM es lo que se llama modificación por ancho de pulso, y por sus siglas del inglés se le llama PWM, ahora se verá cómo lograr señales PWM con el módulo PWM modo rápido timer2 AVR.

PWM timer2 AVR en modo rápido

El temporizador timer2 es de 8 bits, el registro **TCNT2** es el que en si es el temporizador, es el que irá aumentando sus valores de 0 a 255 en el proceso de temporización, el aumento de sus valores en una unidad puede ser con cada ciclo de trabajo del microcontrolador AVR, o si se utilizan los prescaler del timer2 el aumento en una unidad de sus valores tardará mas, dependiendo del prescaler utilizado.

En el PWM timer2 AVR modo rápido el registro TCNT2 irá de 0 a 255, luego se reiniciará a 0 para volver hasta 255 y luego otra vez de 0 a 255 y así continuará en el modo rápido; a este ir de 0 a 255 le tomará un tiempo que dependerá del prescaler utilizado para el timer2, y ese tiempo que le tome al registro TCNT2 para ir de 0 a 255 será el periodo de la señal PWM timer2 AVR modo rápido, al registro TCNT2 se lo puede imaginar como si estuviese generando una onda diente de sierra en el transcurso del tiempo.

Una cosa a tener en cuenta es que al pasar el registro TCNT2 de 255 a 0 se desborda, por lo cual se puede habilitar el uso de la interrupción por desborde del timer2.



Normalmente el registro **TCNT2** irá aumentando su conteo con cada ciclo de reloj del microcontrolador, si se usa por ejemplo una frecuencia de trabajo (lo que se conoce como **FCPU**) de 1Mhz entonces el registro TCNT2 aumentará en una unidad cada microsegundo, y como este registro es de 8 bits este aumentará desde un mínimo de 0 que es su valor BOTTOM, hasta un máximo de 255 que es su valor TOP, al ir desde 0 hasta 255 habrán transcurrido 255us luego volverá a 0, pero en esa vuelta a 0 transcurre 1us mas, por lo que en ir de 0 a 255 y volver a 0 han pasado 256us, que vendría a ser el periodo de la señal PWM timer2 AVR **Tpwm** cuando la FCPU es de 1Mhz; no siempre se utiliza una FCPU de 1Mhz esto puede variar, y como consecuencia variará el tiempo que pasa para que el registro TCNT2 aumente su valor desde 0 a 255.

El tiempo que trascurre para que el registro TCNT2 aumente su valor desde 0 a 255 se puede modificar también mediante el uso de los prescaler, la siguiente es la tabla para la elección de los prescaler para el timer2 mediante las combinaciones de los bits 2, 1, y 0 del registro TCCR2B.

CS22	CS21	CS20	Description
0	0	0	No clock source (timer/counter stopped)
0	0	1	clk_{T2S}/(no prescaling)
0	1	0	clk_{T2S}/8 (from prescaler)
0	1	1	clk_{T2S}/32 (from prescaler)
1	0	0	clk_{T2S}/64 (from prescaler)
1	0	1	clk_{T2S}/128 (from prescaler)
1	1	0	clk_{T2S}/256 (from prescaler)
1	1	1	clk_{T2S}/1024 (from prescaler)

Si por ejemplo se elige un prescaler de 8 y la FCPU=1Mhz, entonces el registro TCNT2 aumentará en una unidad cada 8/FCPU lo que es cada 8 microsegundos, y si el registro TCNT2 va de 0 a 255 y vuelve a 0 para completar un periodo de la señal PWM timer2 AVR **Tpwm**, habrá realizado el contero de 256 unidades, por lo que habrán transcurrido (256*8)/FCPU microsegundos.

Para cualquier otro prescaler que se utilice el periodo de la señal PWM timer2 AVR en modo rápido tendrá la siguiente forma:

Tpwm=(256*prescaler)/FCPU

Si en lugar del periodo se quiere utilizar la frecuencia de la señal PWM timer2 AVR en modo rápido, solo hay que invertir la ecuación anterior y se obtendrá:

Fpwm=FCPU/(prescaler*256)

PWM timer2 AVR modo rápido obtención de la señal PWM

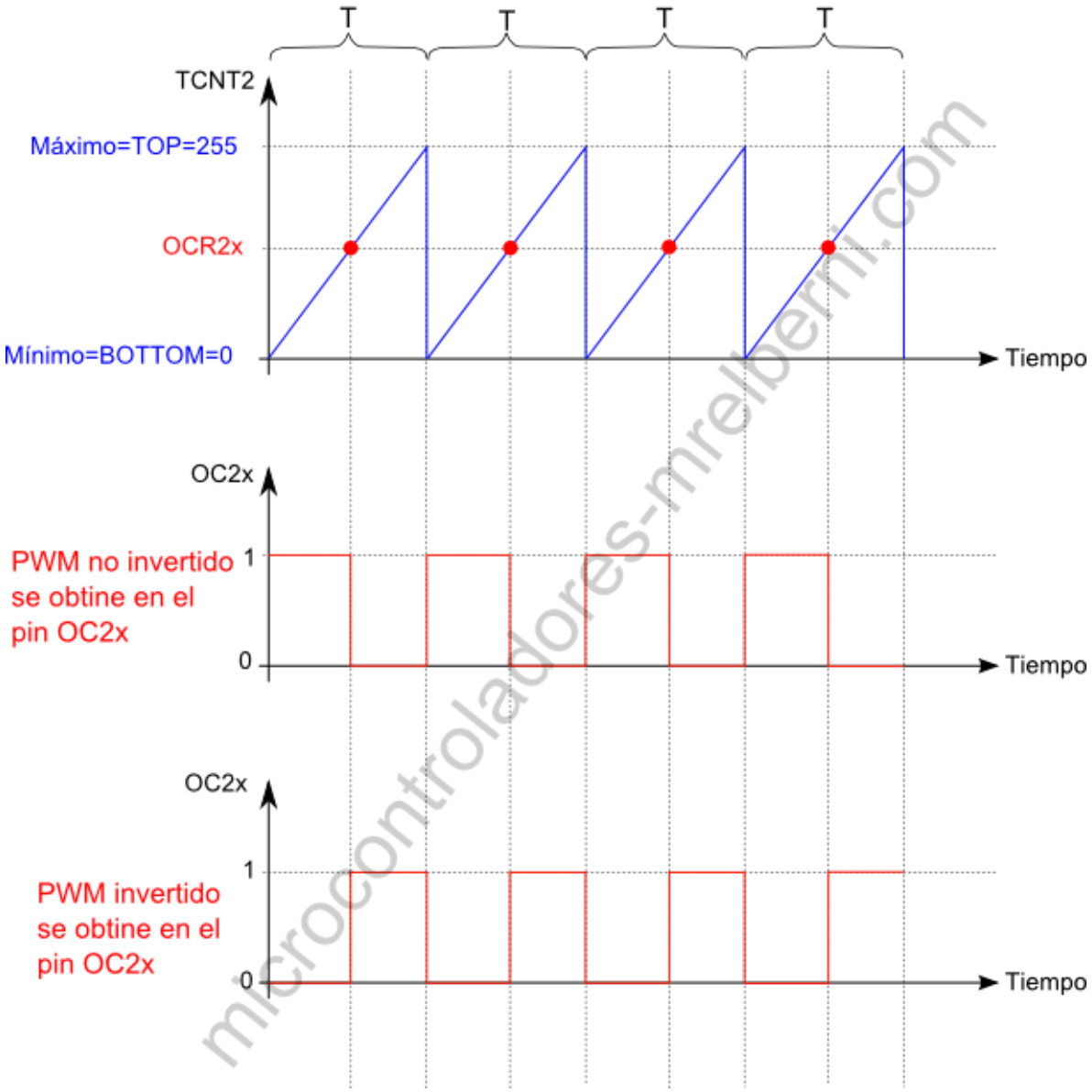
La señal PWM timer2 AVR se genera cuando el valor del registro TCNT2 el que se encuentra incrementándose de unidad en unidad se iguala al valor almacenado en el registro OCR2A o en el registro OCR2B, cuando se utilizó el timer2 en modo comparación al ocurrir esta igualdad el registro TCNT2 se reiniciaba a 0, en el modo PWM el registro TCNT2 no se reiniciará tras ocurrir la igualdad sino que seguirá incrementado sus valores hasta llegar a su máximo de 255, momento en el que recién se reiniciará, la señal PWM timer2 AVR generada se obtendrá en cualesquiera de los pines OC2A o OC2B del ATmega88, dependiendo si para la comparación se utiliza el registro OCR2A o el registro OCR2B, en adelante se utilizará OC0x para los pines y OCR0x para los registros de comparación donde x puede ser A o B.

Una cosa a tener en cuenta es que al ocurrir la igualdad entre el registro TCNT2 y el registro OCR2A, se puede habilitar el uso de la interrupción por comparación del timer2.

La señal PWM timer2 AVR obtenida en el pin OC2x puede ser en forma no invertida o en forma invertida, estos pines deben ser declarados como salidas digitales mediante sus registros DDRnx respectivos.

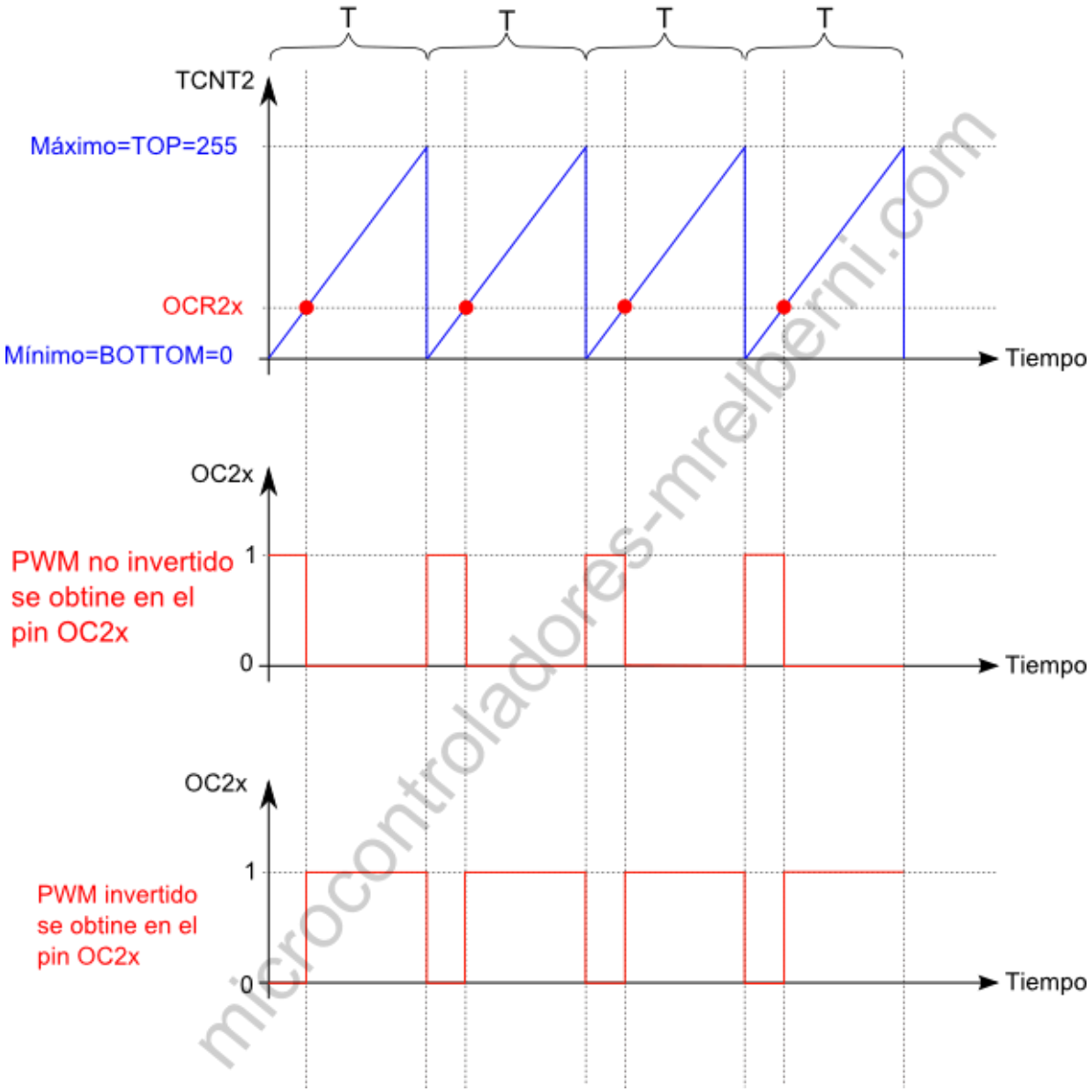
Para la forma no invertida por el pin OC2x se obtendrá un alto mientras el valor del registro TCNT2 sea menor al valor almacenado en el registro OCR2x, al ocurrir la igualdad entre los registros TCNT2 y OCR2x el estado del pin OC2x cambiará a un bajo y se mantendrá así hasta que el registro TCNT2 llegue a su máximo que es 255 y vuelva 0, momento en el cual el estado del pin OC2x cambiará nuevamente a un alto y el ciclo se repetirá.

Para la forma invertida por el pin OC2x se obtendrá un bajo mientras el valor del registro TCNT2 sea menor al valor almacenado en el registro OCR2x, al ocurrir la igualdad entre los registros TCNT2 y OCR2x el estado del pin OC2x cambiará a un alto y se mantendrá así hasta que el registro TCNT2 llegue a su máximo que es 255 y vuelva 0, momento en el cual el estado del pin OC2x cambiará nuevamente a un bajo y el ciclo se repetirá.



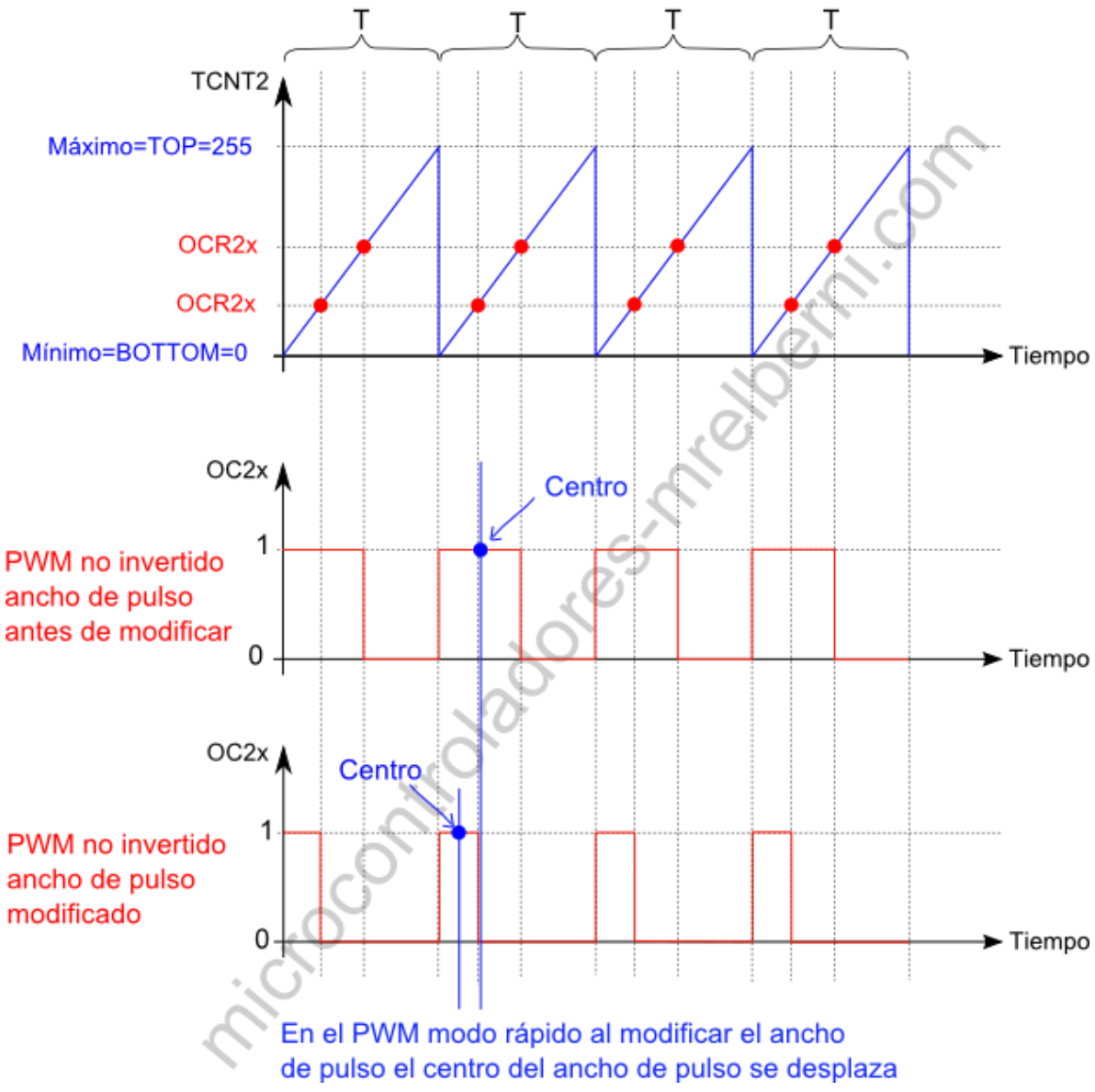
Para modificar el ancho de pulso de la señal PWM timer2 AVR modo rápido solo hay que modificar el valor almacenado en el registro OCR2x, con esto se logra que los cambios de estado del pin OC2x sean en diferentes tiempos, por ejemplo si se toma como referencia la imagen anterior para el nivel del registro OCR2x, cambiando su nivel se obtendrá la señal PWM timer2 AVR con un ancho de pulso modificado pero manteniendo constante su periodo, esto se puede ver en la siguiente imagen.

Es importante señalar que el cambio en el valor almacenado en el registro OCR2x, el microcontrolador AVR lo hará cuando el registro TCNT2 llegue a su máximo valor no antes, esto es si TCNT2 se ha estado incrementando y en ese momento se modifica el valor a cargar en OCR2x, este no se modificará inmediatamente, sino que la modificación se realizará cuando TCNT2 llegue a su máximo.



Una observación a tener muy en cuenta y es lo que diferencia al modo PWM rápido con el modo fase correcta, es que si se toma el centro de los anchos de pulso de la señal PWM timer2 AVR modo rápido y se hace un cambio en el ancho del pulso, el centro del ancho de pulso se moverá, cambiará de lugar, se dice que se modifica la fase del ancho de pulso, o que la fase no es correcta, esto se puede ver en la siguiente imagen.

Cuando el valor del registro TCNT2 se iguale al valor del registro OCR2x, el estado de pin OC2x cambiará.



PWM timer2 AVR modo rápido Programación

Para programar el módulo PWM timer2 AVR en el ATmega88, se utilizan 2 registros los que son el registro **TCCR2A** y el registro **TCCR2B.**

Bit	7	6	5	4	3	2	1	0	
(0xB0)	COM2A1	COM2A0	COM2B1	COM2B0	–	–	WGM21	WGM20	TCCR2A
Read/write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Los bits 7 y 6 son para elegir que la obtención de la señal PWM timer2 AVR será por el pin **OC2A** y si será en forma no invertida o en forma invertida, mediante las combinaciones de estos bits según se indica en la siguiente tabla.

Los bits 5 y 4 son para elegir que la obtención de la señal PWM timer2 AVR será por el pin **OC2B** y si será en forma no invertida o en forma invertida, mediante las combinaciones de estos bits según se indica en la siguiente tabla.

Table 18-6. Compare output mode, fast PWM mode(1).

COM2B1	COM2B0	Description
0	0	Normal port operation, OC2B disconnected
0	1	Reserved
1	0	Clear OC2B on compare match, set OC2B at BOTTOM, (non-inverting mode)
1	1	Set OC2B on compare match, clear OC2B at BOTTOM, (invertiing mode)

Si las combinaciones de estos bits son 10 la señal PWM obtenida por el pin OC2B será no invetida, si las combinaciones de estos bits son 11 la señal PWM obtenida por el pin OC2B será invetida

Los bits 3 y 2 no se utilizan por lo que se les pone a 0.

Los bits 1 y 0 junto con el bit3 del registro TCCR2B son con los cuales se elige el modo de obtener las señales PWM timer2 AVR, mediante las combinaciones de estos bits según se indica en la siguiente tabla.

Table 18-8. Waveform generation mode bit description.

Mode	WGM2	WGM1	WGM0	Timer/counter mode of operation	TOP	Update of OCRx at	TOV flag set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, phase correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, phase correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

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

La combinación de estos bits que se utilizara para la generación de la señal PWM timer2 AVR modo rápido será la tercera, la que está resaltada, esto es el bit 3 del registro TCCR2B se pondrá a 0 y los bits 1 y 0 del registro TCCR2A se pondrán a 1.

El registro TCCR2B

Bit	7	6	5	4	3	2	1	0	
(0xB1)	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	TCCR2B
Read/write	W	W	R	R	R	R	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Los bits 7, 6, 5, y 4 no se utilizarán en la obtención de la señal PWM timer2 AVR modo rápido por lo que se les pondrá a 0.

El bit3 trabaja junto con los bits 1 y 0 del registro TCCR2A tal como se comentó.

Los bits 2, 1 y 0 son para elegir el prescaler a utilizar para obtener la frecuencia de la señal PWM timer2 AVR, las combinaciones de estos bits para los diversos prescaler del timer2 son los que se indican en la siguiente tabla:

CS22	CS21	CS20	Description
0	0	0	No clock source (timer/counter stopped)
0	0	1	clk_{T2S}/(no prescaling)
0	1	0	clk_{T2S}/8 (from prescaler)
0	1	1	clk_{T2S}/32 (from prescaler)
1	0	0	clk_{T2S}/64 (from prescaler)
1	0	1	clk_{T2S}/128 (from prescaler)
1	1	0	clk_{T2S}/256 (from prescaler)
1	1	1	clk_{T2S}/1024 (from prescaler)

Los registros OCR2A y OCR2B

Se usará OCR2x para referirse a cualesquiera de ellos, x será A o B, en este registro se cargará el valor mediante el cual se controlará el ancho de pulso de la señal PWM que se obtendrá por el pin OC2x, este valor va de 0 a 255, cuando sea 0 el ancho de pulso será 0 o del 0%, cuando sea 255 el ancho de pulso será igual al periodo o del 100% de la señal PWM timer2 AVR, y entre ese intervalo se obtienen los diferentes anchos de pulso de la señal PWM.

Se puede utilizar la siguiente forma para el ciclo de trabajo de la señal PWM timer2 AVR modo rápido.

Ciclo de trabajo=((OCR2x)/255)*100%

Por ejemplo si se quiere un ciclo de trabajo del 75%, se puede proceder así

75%=((OCR2x)/255)x100%

De donde al despejar OCR2x se tendrá:

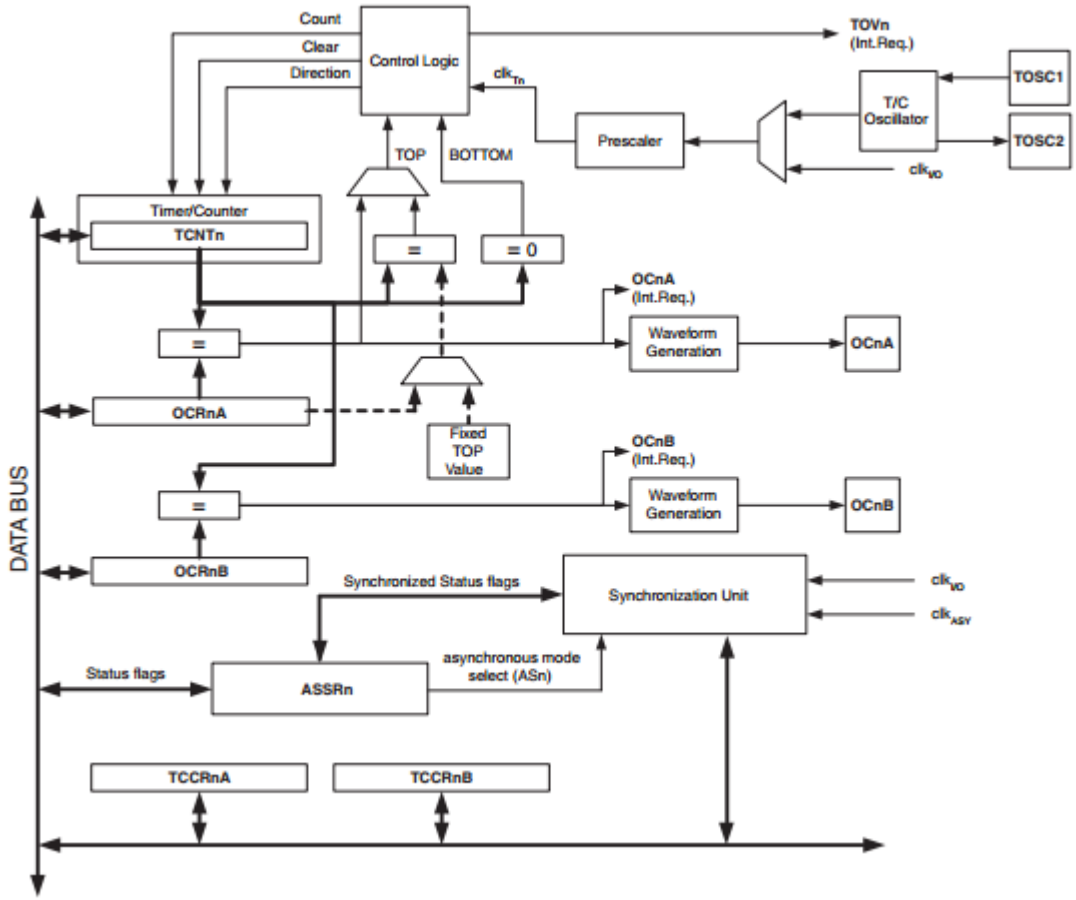
OCR2x=191

Esto indica que si se quiere un ciclo de trabajo del 75% de la señal PWM en el registro OCR2x se tendrá que cargar el valor de 191 y esto será lo que se haga en el primer ejemplo.

Normalmente lo que se quiere es controlar todo el ancho de pulso de la señal PWM, por lo que de alguna forma se hace variar el valor de OCR2x, se verán algunos ejemplo de como realizar esto.

Si se utilizan ambos pines OC2A y OC2B, la frecuencia de las señales PWM obtenidas en ambos pines serán las mismas, ya que ambas utilizan el mismo registro TCNT2, que es con el que decide la frecuencia o su inversa que es periodo de la señal PWM.

Figure 20-1. 8-bit Timer/Counter Block Diagram



Para el caso de la práctica se utilizara el pin OC2B para generar tonos musicales en base a una frecuencia dada.

Para generar las distintas frecuencias se utiliza el en el modo PWM con tope en OCR2A y se maneja el ciclo de trabajo con OCR2B.

Conclusiones y comentarios

Esta práctica me ayudo a comprender bien cómo funciona el timer/counter2 como temporizador para generar frecuencias, el tema se vio en clase pero al llevarlo a la práctica pude comprender bien cómo funciona este modo de operación, muchas aplicaciones se le pueden dar a un generador de ondas como controlar la energía motor o cualquier aparato controlado por un sistema embebido.

Bibliografía

[1] PWM timer2 AVR modo rápido. Mayo 12, 2017, de Programación de Microcontroladores PIC, AVR, ARDUINO Sitio web: http://microcontroladores-mrelberni.com/pwm-timer2-avr-modo-rapido/

[2] Hoja de datos del microcontrolador ATmega1280/1281/2560/2561