# PRACTICA N° 8

MODULADOR DE ANCHO DE PULSO POR FASE CORRECTA

DE UN MICROCONTROLADOR ATMEGA328P

## OBJETIVOS.

HABILITAR Y USAR LOS TEMPORIZADORES MEDIANTE MODULADOR DE ANCHO DE PULSO POR FASE CORRECTA.

## INTRODUCCION

El PWM se utiliza principalmente para el control de motores mediante un puente H, esto es debido a que la pérdida de par es mínimo y la velocidad se puede controlar en un rango aceptable.

El AVR tiene dos modos de PWM el rápido y el de fase correcta, para esta ocasión se realizara la práctica en modo de fase correcta. Esta proporciona una resolución alta debido a que tiene una doble pendiente, primero va de 0 hasta llegar a 255 para después descender a 0, esto en el caso del temporizador (Timer) 0 y 2. Este modo tiene el doble de frecuencia que el de pendiente simple, sin embargo la simetría de sus pendientes lo hacen optimo para el control de motores.

El PWM se origina cuando hay una comparación en la cual se igualan los registros TCNT0 y OCR0, lo que origina que se limpie la salida OC0 cuando se está ascendiendo la cuenta y pone a uno cuando se está descendiendo y existe la igualdad vea la figura 8.1, en el modo no invertido y hace lo contrario en el modo invertido. El modo PWM de fase correcta tiene una frecuencia menor que el PWM rápido por lo cual se usa más en aplicaciones de control de motores.

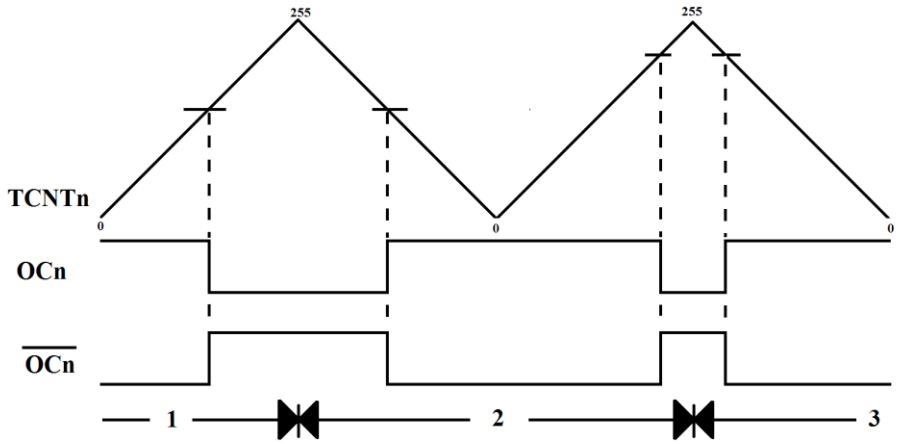


**Figura 8.1 Grafica de PWM fase correcta**

La frecuencia del PWM en el modo de fase correcta PWM puede calculada por la siguiente ecuación:

$$f_{OCnFCPWM} = \frac{f_{clk\ I/O}}{N * 510}$$

La variable $N$ representa el factor de pre-escala (1, 8, 64, 256 o 1024).

Los registros involucrados para el PWM en modo de fase correcta se muestran a continuación, cabe aclarar que la mayoría de los bits que intervienen se han descrito en las prácticas 5, 6 y 7.

**TABLA 8.1 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 |

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

| 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 ) |

## 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 |

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

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

### Registro Temporizador/Contador (TCNT0)

| BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

### Registro de Salida de Comparación (OCR0A)

| BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

### Registro de Salida de Comparación (OCR0B)

| BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

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

| BIT 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | OCF0B | OCF0A | TOV0 |

## MATERIAL Y EQUIPO

1 Protoboard
1 C.I. AVR ATMEGA 16
8 LEDs
8 resistencias de 330Ω
Fuente de Alimentación Regulada de 5 volts
Cables para conexión rápida

1 Multimetro

# DESARROLLO

## Planteamiento del problema

Se requiere escribir un programa en el cual se aumente o disminuir el ciclo de trabajo del PWM en modo de fase correcta mediante dos botones, el botón 1 se empleara para aumentar mediante la interrupción externa 0 y el botón 2 se empleara para disminuir mediante la interrupción externa 1.

### Procedimiento

1. Realizar el diagrama de flujo.
2. Incluir las bibliotecas necesarias
3. Desactivar la interrupción global
4. Definir la pre escala
5. Habilitar la interrupción por desbordamiento del temporizador 0
6. Habilitar la interrupcion global
7. En la función de interrupción sacar a puerto la variable que incrementa
8. Incrementar la variable

## Diagrama del circuito

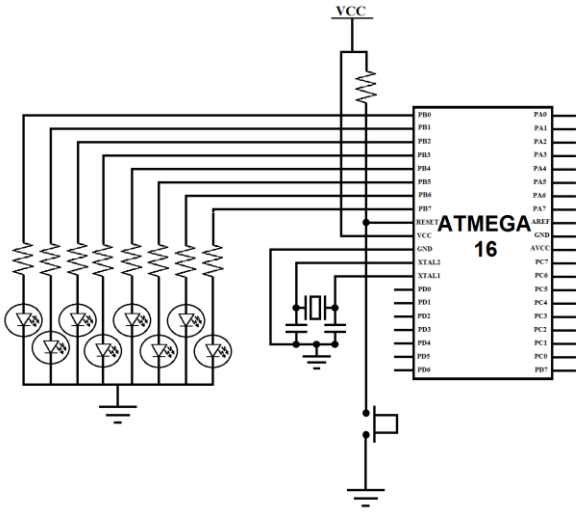


**Figura 8.1 Diagrama del circuito para la practica 8.**

## CODIGO EN LENGUAJE C.

```
# define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <avr/cpufunc.h>
int i=250;
ISR(INT0_vect, ISR_NAKED){
        i=i+10;
        OCR0A=i;
reti();
}
ISR(INT1_vect, ISR_NAKED){
        i=i-10;
        OCR0A=i;
reti();
        }

ISR(TIMER0_COMPA_vect, ISR_NAKED){
        PORTB=PINB;
        reti();
        }

int main(void)
{
        cli();
        DDRB = 0xff;
        DDRD = 0x00;
        PORTD=255;
        TCCR0A = (1<<WGM00)|(1<<COM0A1)|(1<<COM0A0);
        TCCR0B= (1<<WGM02)|(1<<CS02)|(1<<CS00);
        OCR0A = i;
        TIMSK0 = (1<<OCIE0A);

        EIMSK = (1<<INT1)|(1<<INT0);
        EICRA = (1<<ISC11)|(1<<ISC10)|(1<<ISC01)|(1<<ISC00);

        sei();

        while(1)
        {
                _NOP();
                _NOP();
        }
}
```

## ACTIVIDADES DE TRABAJO AUTÓNOMO.

1) Practica propuesta por el profesor en clase.