**ARQUITECTURA DE COMPUTADORAS**

**Practica 03. Control de un Servomotor**

**OBJETIVO:**

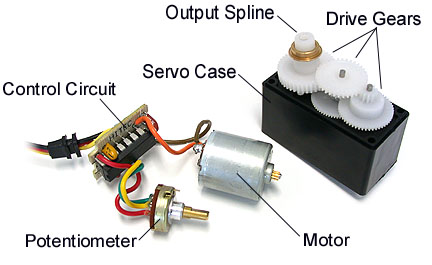
Identificar y realizar la conexión de un servomotor, la generación de la señal PWM para ubicar el rotor del servomotor en la posición (ángulo) deseado.

**MATERIALES Y EQUIPO:**

|  |  |
| --- | --- |
| Microcontrolador 16F877A con cristal (16 MHz) y capacitores (22 pF) |  |
| Protoboard (mínimo de 400 puntos) |  |
| 1 Resistencia de 10Kohm |  |
| Cables de conexión (UTP o Dupont) |  |
| Entrenadora Digital (IDL-800) |  |
| Grabadora de Microcontroladores (Master PROG) con su cable USB y el software de grabación. | Resultado de imagen de master prog mercadolibre |
| Servomotor SG90 | Resultado de imagen para servo sg90 |

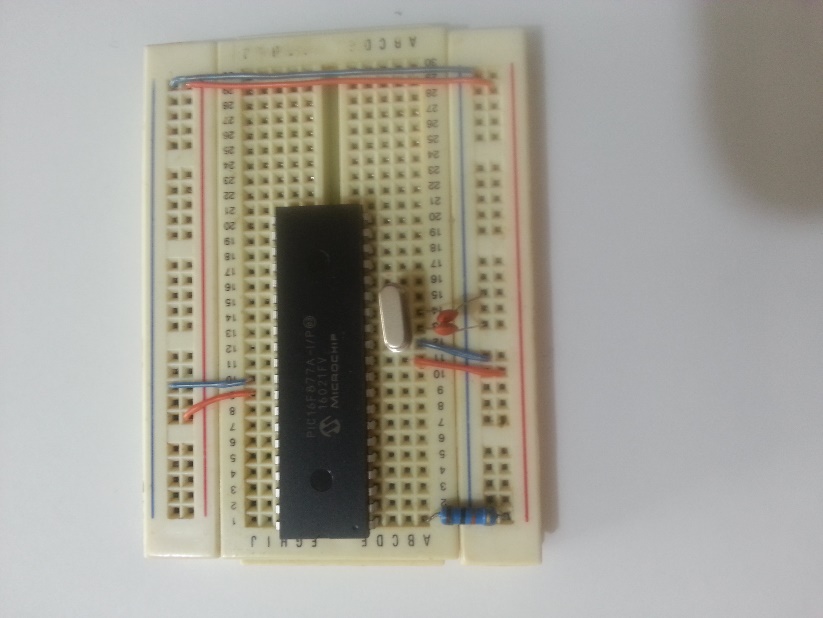
**INTRODUCCIÓN:**

En esta práctica trabajamos con un servomotor y aprendimos a moverlo a diferentes ángulos usando una señal PWM. A diferencia de un motor normal, el servo se puede colocar en posiciones específicas, como 0°, 90° o 180°. Con el PIC16F877A generamos las señales necesarias para que el servo gire hasta el punto que le indiquemos, lo que sirve para entender cómo controlar movimientos de manera más precisa.



**PROCEDIMIENTO:**

1. Realice la conexión básica del microcontrolador. Conecte en orden los pines del puerto C a los led del entrenador digital.



2. Haga un proyecto nuevo (practica03) codifique lo siguiente:

Practica 03: Control de Servo

\* El servomotor funciona con una señal PWM. La mayoría trabaja en una frecuencia de 50 Hertz, asi las señales

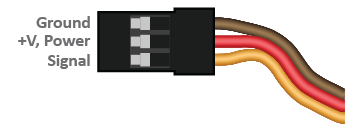
\* tienen un periodo de 20 milisegundos.

+ Para una correcta manipulación de un servo hay que revisar la Hoja de Datos a fin de conocer el ancho de pulso especifico

\* En esta práctica se implementa el micro servo Tower Pro SG90 con ancho de pulso de 500-2400 us

\* PIC16F877A 16Mhz

\* Conexión del servo: el cable rojo es la alimentación (positivo) del servo, debe conectarse a 5 volt, max 7v. de preferencia a una alimentación separada de la que alimenta al PIC; el cable café es el negativo y va conectado a GND de la entrenadora; y por último el cable naranja (señal) conectado al pin RB0 de PIC.

\*

\*/

short conta;

void main() {

TRISB=0B00000000;

PORTB=0;

while(1){

conta=0; //posiciona el servo a 180°

while(conta<=50){ //El ciclo repite 50 veces el pulso siguiente (50Hz)

PORTB.f0=1; //se envía una señal alta

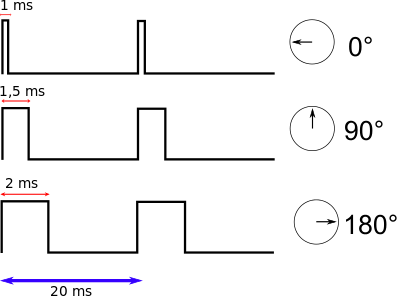
delay\_us(2000); // la señal dura 2 milisegundos alta

PORTB.f0=0; // se envía una señal baja

delay\_us(18000); // la señal dura 18 milisegundos baja

conta++;} // en total el pulso dura la suma de los tiempos anteriores, es decir 20 milisegundos

Delay\_ms(2000); //Permanece 2 segundos en la posición anterior



conta=0;

while(conta<=50){ //posiciona el servo a 90°

PORTB.F0=1;

delay\_us(1250);

PORTB.F0=0;

delay\_us(18750);

conta++;}

Delay\_ms(2000);

conta=0;

while(conta<=50){ //posiciona el servo a 0°

PORTB.f0=1;

delay\_us(500);

PORTB.f0=0;

delay\_us(19500);

conta++;}

Delay\_ms(2000);

}

}

3. Compilar y grabar el proyecto.

4. Realice otro proyecto y configure los puertos necesarios para manipular un servomotor, iniciar en la posición 0° e ir incrementando de posición en 10° hasta llegar a 180° y posteriormente decrementar en 20° hasta llegar a 0° (dos segundos de retardo en cada cambio de posición). Repetir la secuencia indefinidamente.

Codigo:

/\*

\* Control de Servo Motor con PIC16F877A

\* RB0 - Señal PWM para servo

\*/

short conta;

void main() {

TRISB = 0x00; // Puerto B como salida

PORTB = 0x00; // Inicializar PORTB en 0

while(1) {

// Posición 180° - Pulso de 2ms

conta = 0;

while(conta <= 50) {

PORTB.F0 = 1; // Señal HIGH

Delay\_us(2000); // 2ms HIGH

PORTB.F0 = 0; // Señal LOW

Delay\_us(18000); // 18ms LOW

conta++;

}

Delay\_ms(2000); // Espera 2 segundos

// Posición 90° - Pulso de 1.25ms

conta = 0;

while(conta <= 50) {

PORTB.F0 = 1;

Delay\_us(1250); // 1.25ms HIGH

PORTB.F0 = 0;

Delay\_us(18750); // 18.75ms LOW

conta++;

}

Delay\_ms(2000);

// Posición 0° - Pulso de 0.5ms

conta = 0;

while(conta <= 50) {

PORTB.F0 = 1;

Delay\_us(500); // 0.5ms HIGH

PORTB.F0 = 0;

Delay\_us(19500); // 19.5ms LOW

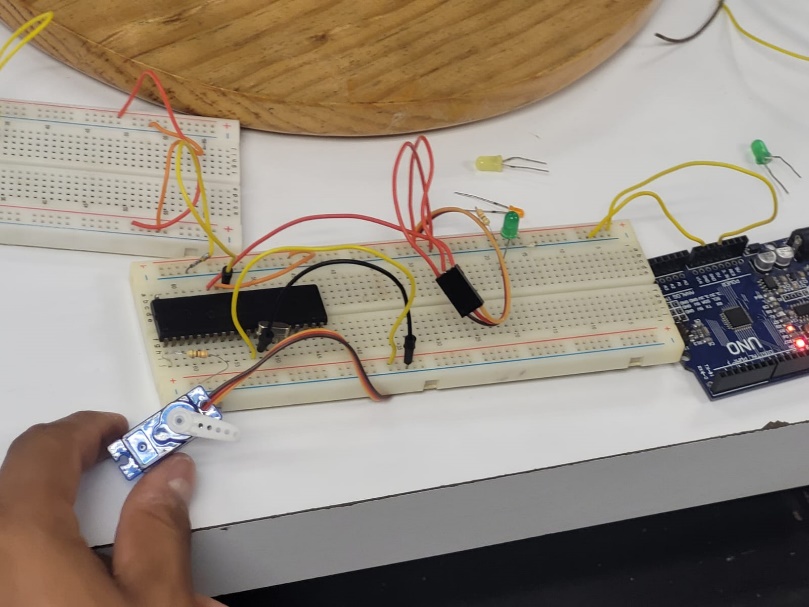
conta++;

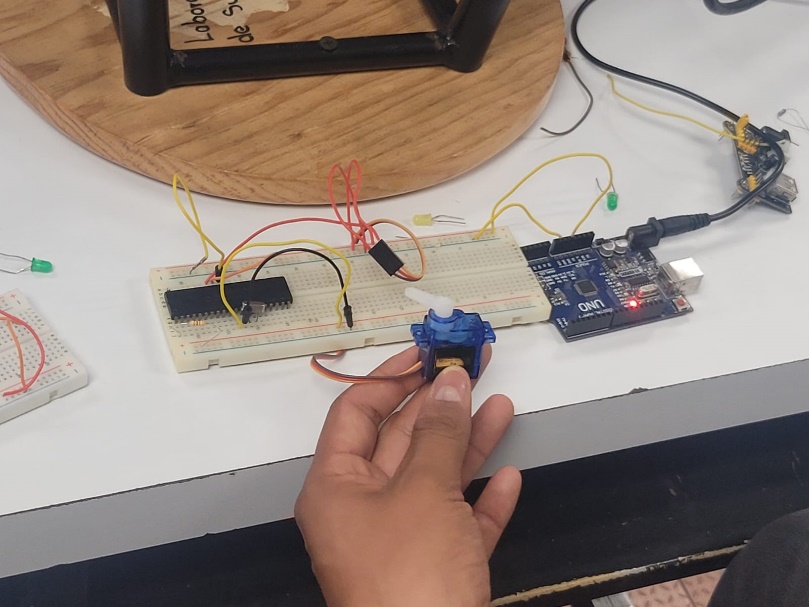
}

Delay\_ms(2000);

}

}





Conclusión:

La práctica nos ayudó a ver cómo un microcontrolador puede controlar el movimiento de un servo con solo cambiar el tiempo de la señal. Pudimos moverlo a varias posiciones y entender mejor cómo funciona la señal PWM. Es una buena base para proyectos donde necesitemos controlar motores o movimientos más complejos.