**ARQUITECTURA DE COMPUTADORAS**

**Practica Semaforo**

**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 |
| Leds | Visualización de salidas con LEDs |

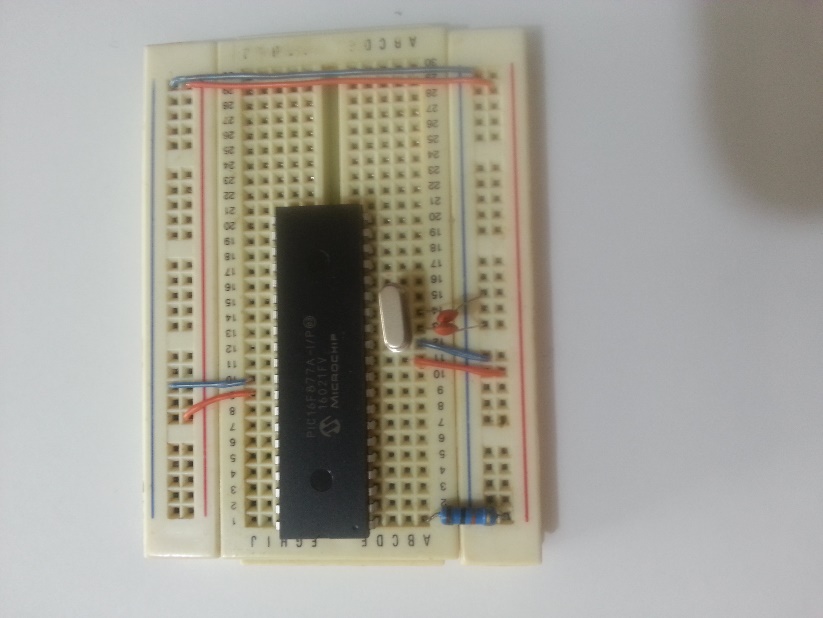
**INTRODUCCIÓN:**

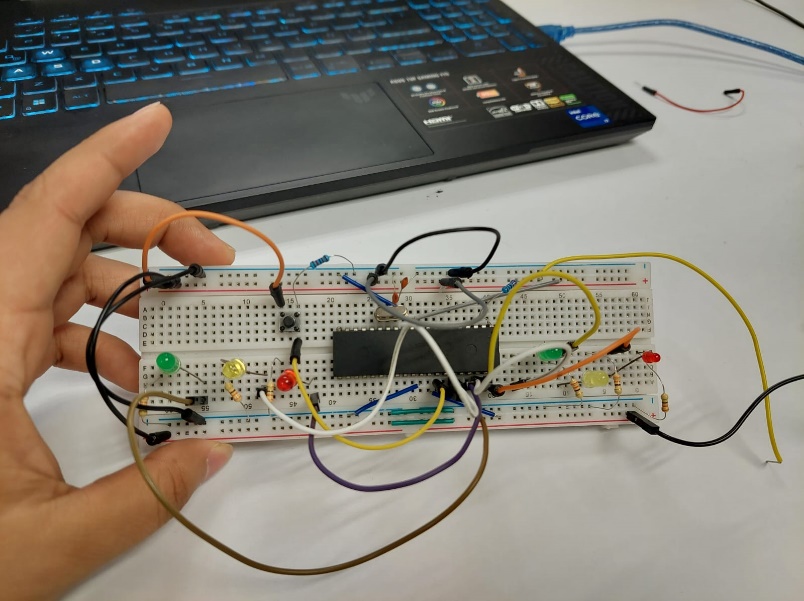
En esta práctica trabajamos con la simulación de un semáforo utilizando un microcontrolador y varios LEDs conectados en la protoboard. La idea principal fue aprender a controlar las salidas digitales del microcontrolador para encender los focos (rojo, amarillo y verde) en un orden específico, parecido a como funciona un semáforo real en la calle.

Con esta actividad se busca entender cómo se pueden programar y conectar los componentes básicos para lograr un sistema secuencial sencillo. Además, se reconoce la importancia de las resistencias, el uso correcto de la protoboard y el papel que juegan los pulsadores como entradas.

**PROCEDIMIENTO:**

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





2. Codigo:

void main() {

TRISB = 0b00000001; // RB0 como entrada (pulsador), el resto salidas

PORTB = 0x00; // Inicializar todos los pines en 0

while(1){

if( PORTB.F0 == 1 ){ // --- MODO NORMAL ---

// Semáforo 1 en ROJO, Semáforo 2 en VERDE

PORTB.F7 = 1; // Rojo S1

PORTB.F6 = 0; // Amarillo S1

PORTB.F5 = 0; // Verde S1

PORTB.F3 = 0; // Rojo S2

PORTB.F2 = 0; // Amarillo S2

PORTB.F1 = 1; // Verde S2

Delay\_ms(3000); // 3 seg en verde

// Semáforo 2 en AMARILLO

PORTB.F1 = 0;

PORTB.F2 = 1;

Delay\_ms(1000); // 1 seg en amarillo

PORTB.F2 = 0;

// Semáforo 1 en VERDE, Semáforo 2 en ROJO

PORTB.F7 = 0;

PORTB.F5 = 1;

PORTB.F3 = 1;

PORTB.F1 = 0;

Delay\_ms(3000);

// Semáforo 1 en AMARILLO

PORTB.F5 = 0;

PORTB.F6 = 1;

Delay\_ms(1000);

PORTB.F6 = 0;

}

else{ // --- MODO NOCTURNO ---

// Ambos en AMARILLO intermitente

PORTB.F7 = 0; // Rojo S1 apagado

PORTB.F6 = 1; // Amarillo S1 encendido

PORTB.F5 = 0; // Verde S1 apagado

PORTB.F3 = 0; // Rojo S2 apagado

PORTB.F2 = 1; // Amarillo S2 encendido

PORTB.F1 = 0; // Verde S2 apagado

Delay\_ms(500);

// Apagar ambos amarillos

PORTB.F6 = 0;

PORTB.F2 = 0;

Delay\_ms(500);

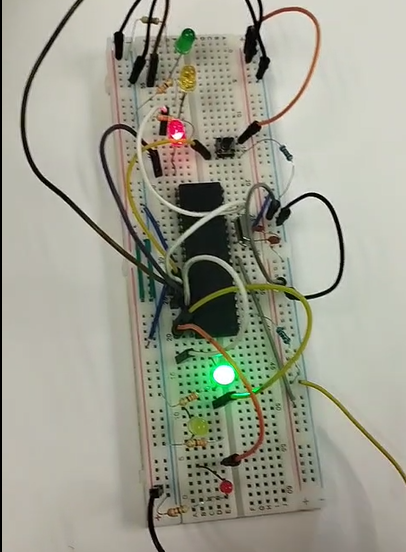
}

}

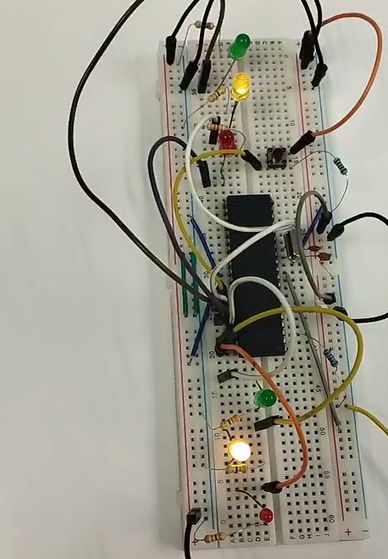
}

3. Compilar y grabar el proyecto.

Semáforo modo normal



Semáforo modo nocturno



Conclusión:

Se comprendió cómo un microcontrolador puede controlar de forma sencilla un sistema real, siempre y cuando se le programe de la manera correcta. También vimos que detalles como el buen uso de resistencias y el orden del cableado hacen la diferencia para que el circuito funcione.

Además, al programar el semáforo no solo se implementó el ciclo normal, sino también un modo nocturno donde únicamente parpadea el LED amarillo, tal como ocurre en la vida real cuando baja el flujo vehicular. Esto ayudó a entender mejor cómo se pueden manejar diferentes condiciones en un mismo programa y que un solo circuito puede adaptarse a más de una situación.