Reporte. Prender un Led

Unidad Académica Multidisciplinaria Mante (UAMM)

Arquitectura de Computadoras

Montoya Garza Luis Ángel

Hernández Ruiz Haydee Michelle

Rueda Martínez Alison Michelle

Silva Sánchez Yamilka Arely

7° “E”

M.C. López Piña Daniel

Ciudad Mante, Tamaulipas. Agosto del 2025

Introducción

En esta práctica, daremos nuestros primeros pasos en la programación de microcontroladores utilizando el popular PIC16F877A y el lenguaje de programación C. Aprenderemos a configurar y controlar los pines de entrada y salida de propósito general de este chip para encender un componente fundamental: un LED.

El objetivo principal es familiarizarnos con la estructura del microcontrolador y su entorno de desarrollo, lo que nos permitirá escribir un código que configure un pin específico como una salida digital y establezca su estado lógico en "alto" (HIGH) para encender el LED. Este ejercicio es el "Hola Mundo" de la electrónica programable, y dominarlo es la base para proyectos más complejos, ya que nos enseña a cómo el software interactúa y controla el hardware físico.

Desarrollo

La fase de desarrollo de esta práctica es el punto donde la teoría se convierte en acción, en un diálogo directo entre el software y el hardware. Para iniciar, el primer paso es reunir todos los componentes esenciales. No es solo el microcontrolador PIC16F877A el que hace el trabajo; se necesita una protoboard, un LED y una resistencia para limitar la corriente que lo alimenta. Además, el PIC necesita su propio pulso vital: un cristal oscilador de 16 MHz con sus capacitores de 22 pF, que funcionan como el reloj interno del sistema, dictando el ritmo de cada operación.

Código

```
void main() { //Funcion principal
    PORTC=0; //Se le asigna un LOW al puerto C conformado por los pines 15, 16, 17, 18, 23, 24, 25 y 26 del PIC.
    TRISC=0; //Se configura todo el puerto C como salida equivale a 0b00000000 (binario), 0x00(hexadecimal).
    while(1){
    PORTC=~PORTC;  //Aplica un NOT al puerto invirtiendo los valores.
    Delay_ms(1000);  //Espera un segundo
    }
}
```

void main() { ... }: Es la función principal del programa.

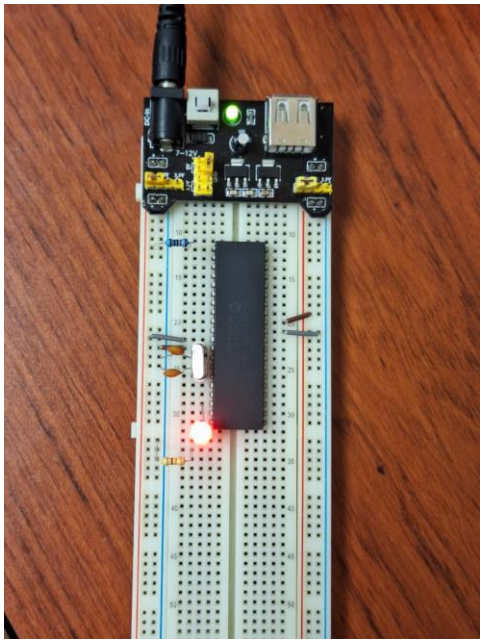
PORTC = 0; y TRISC = 0; Estas dos líneas conforman la fase de inicialización.

PORTC Es el registro que controla el estado de los pines del Puerto C.

while(1) { ... } Asegura que el programa se mantenga en ejecución constante, lo que nos permite controlar los LEDs de manera continua.

PORTC = ~PORTC; Esta es la línea que realiza la acción principal de parpadeo. El operador ~ es el NOT bit a bit.

Delay_ms(1000); Esta función crea la pausa en la ejecución del programa. El valor 1000 está en milisegundos, por lo que el programa se detiene por un segundo.



https://github.com/luisangelmg11/Arquitectura-de-Computadoras/blob/main/1.0%20Practica%20Evidencia%20Video.mp4

Conclusión

En esta práctica, se logró con éxito el objetivo de programar el microcontrolador PIC16F877A para controlar un LED. Mediante la configuración del registro TRISC, se estableció el pin del Puerto C como una salida digital, lo que permitió controlar el estado de un componente externo.

La manipulación del registro PORTC y el uso del operador NOT (~) en un bucle infinito (while(1)) generaron una inversión constante del estado del pin, creando el efecto de parpadeo. La función Delay_ms fue fundamental para introducir la pausa necesaria, haciendo que el cambio de estado fuera perceptible.

Como resultado, se verificó el control efectivo del hardware a través de software, lo que confirma una comprensión básica de los registros de control del PIC y la lógica de programación para sistemas embebidos.

Perfiles GitHub

https://github.com/luisangelmg11
https://github.com/HaydeesitaSJJH
https://github.com/alisonrueda
https://github.com/Yamixjk