Proyecto 1. Secuencia Led (Assembler)

Universidad
del Cauca

Autores:

Miguel Tapia

Esteban Arciniegas

Brenda Samboni

Santiago Martinez

Juan Molineros

Profesor:

Ing. Cristian Heidelberg Valencia Payan

Universidad del Cauca

Facultad de ingeniería electrónica y telecomunicaciones

Ingeniería electrónica y telecomunicaciones

Quinto Semestre (5)

Popayan,03 de Marzo, 2025

Introducción.

En el proyecto se trabaja la construcción y programación de una secuencia Led, usando el lenguaje de programación Assembler, trabajaremos con el PIC18F4550 y 4 Leds para desarrollar lo solicitado para el proyecto.

Esquemático:

Usaremos los pines RA0, RA1, RA2 y RA3, para el control digital de nuestros Leds, los puertos VSS para conectar a tierra el microcontrolador y los puertos VDD y VPP para alimentarlo con una fuente de corriente continua de 5V, usaremos 4 resistencias de 10 Ohms conectadas en serie con cada uno de los leds, obteniendo el siguiente esquemático:

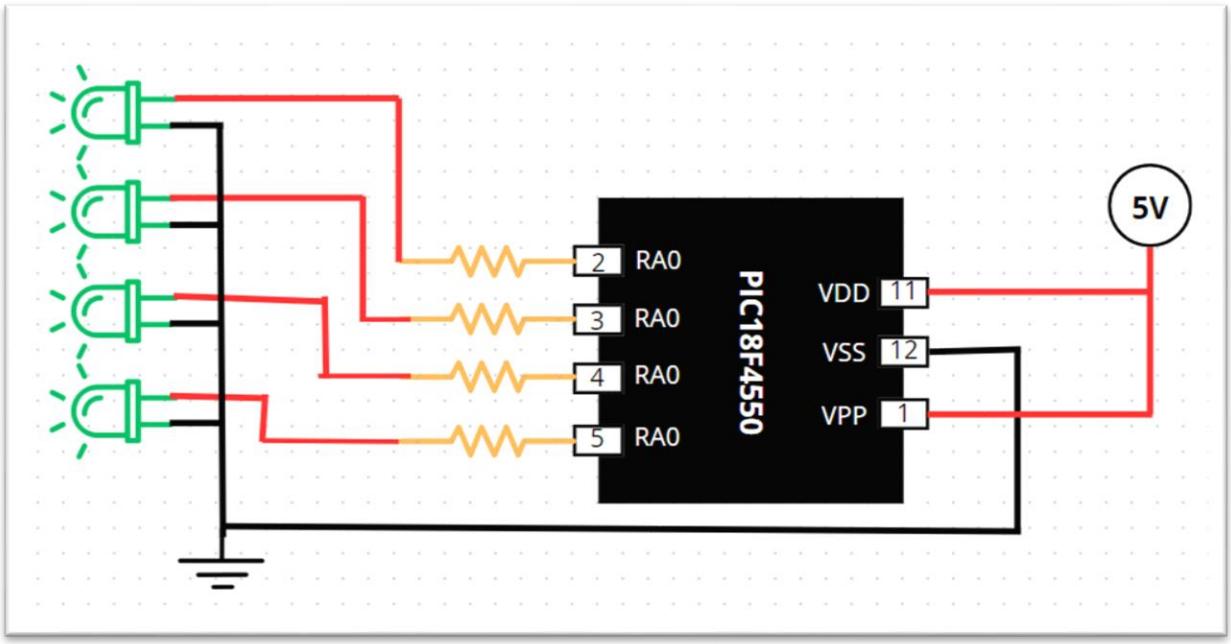


Diagrama funcionamiento:

El diagrama de flujo que representa el funcionamiento deseado del circuito es un bucle que llama a diferentes operaciones, que nos muestra nuestra secuencia de led, es el siguiente enlace:

https://drive.google.com/file/d/1rpa_7QxDAIrB1Y0VmrEDszXiB3srPLwa/view?usp=sharing

Código:

Se uso el entorno de desarrollo MPLAB, a continuación, se mostrarán partes específicas del código y tendrán una explicación breve de su funcionalidad:

Parte 1: Declaramos el PIC el cual será usado, además de configuraciones iniciales del microcontrolador, forzamos a que la memoria inicie en la posición 0x00, creamos variables en espacios específicos de la memoria, seguido de la sección de inicio en donde configuramos nuestra señal de reloj al igual que el modo de operación de los pines RA

```
#include "p18f4550.inc"


  CONFIG  FOSC = INTOSC_HS
  CONFIG  WDT = OFF
  CONFIG  LVP = OFF

  ORG 0x00

  var1 Equ 0x21
  var2 equ 0x22
  var3 equ 0x23
  GOTO Inicio

Inicio:
    MOVLW 0x72
    MOVWF OSCCON
    CLRF TRISA
    CLRF LATA
```

Parte 2: en esta sección es donde se encuentra el funcionamiento de nuestra secuencia Led, en el cual se manda señales de trabajo a pines específicos y se hace llamados a sub procesos reiterativos en el código.

```
main:

    BSF LATA, RA0   ;enciendo RA0
    BSF LATA, RA1   ;enciendo RA1
    CALL medio_seg  ;llamo a un retardo de medio segundo
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CALL medio_seg  ;llamo a un retardo de medio segundo
    BSF LATA, RA2   ;enciendo RA0
    BSF LATA, RA3   ;enciendo RA1
    CALL medio_seg  ;llamo a un retardo de medio segundo
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    BSF LATA, RA0   ;enciendo RA0
    BSF LATA, RA2   ;enciendo RA1
    CALL medio_seg  ;llamo a un retardo de medio segundo
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CALL on_off     ;llamo al alternador
    CLRF LATA       ;pongo toda la fila de RA en 0
    CALL medio_seg  ;llamo al retardo de un segundo
    GOTO main       ;vuelvo al inicio del ciclo
```

Parte 3: tenemos un proceso en el cual se alternas los valores establecidos en todos los leds y vuelve al lugar en donde fue llamado dicho proceso después de su ejecución.

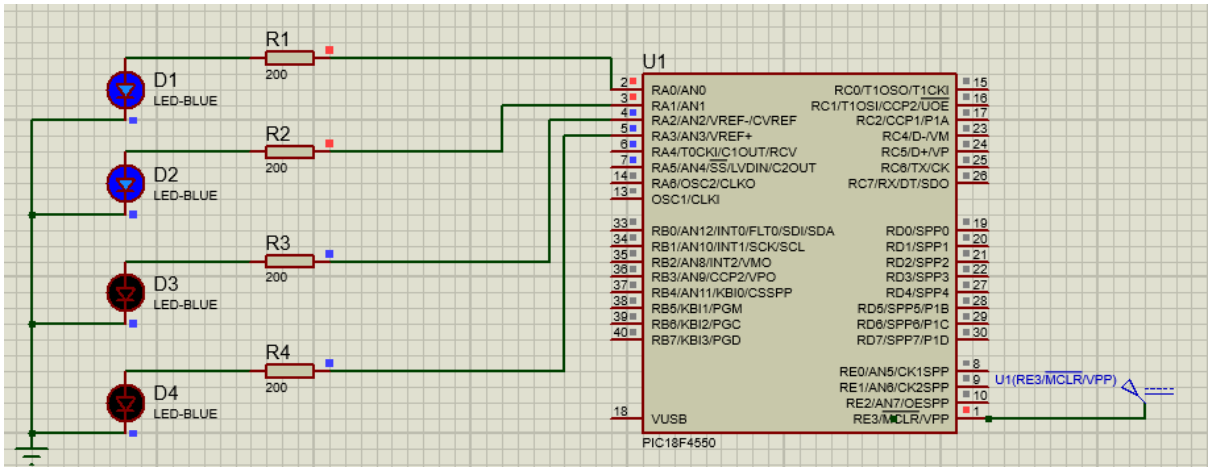
```
on_off:

    BTG LATA, RA0    ;alterno el estado de RA3
    BTG LATA, RA1    ;alterno el estado de RA3
    BTG LATA, RA2    ;alterno el estado de RA3
    BTG LATA, RA3    ;alterno el estado de RA3
    CALL medio_seg   ;llamo a un retardo de medio segundo
    RETURN
```

Parte 4: tenemos un proceso el cual nos sirve como retardo entre operación y operación, consumiendo cierta cantidad de ciclos de reloj para que haya un delay de alrededor de 0.5 segundos cada vez que se llame a este proceso, luego de ser llamado retorna a dicho lugar para seguir con los siguientes pasos de la secuencia de Leds

```
medio_seg:
    MOVLW 20
    MOVWF var1          ; Toma el valor que tiene almacenado

loop1:

    MOVLW 255           ; Le damos otro valor a w para la iteracion
    MOVWF var2          ; Se lo asiganamos a la variable

loop2:

    MOVLW 255           ; Le damos otro valor a w para la iteracion
    MOVWF var3          ; Se lo asiganamos a la variable

loop0:

    NOP                 ; Ciclo vacio
    DECFSZ var3         ; Decrementa 255 hasta ser cero
    GOTO loop0          ; Si ret3 no ha llegado a cero se repite el ciclo
    DECFSZ var2         ; Cada deremento de ret2 pasan 255 iteraciones de ret3
    GOTO loop2          ; Si ret2 aun no llega a cero se repite desde el loop 2
    DECFSZ var1         ; Cada iteracion de ret pasan 255x30 de los loops pasados
    GOTO loop1          ; Si aun no es cero se repide desde el loop 1
    RETURN              ; Devuelve a donde se llamo a la subrutina
    END
```

Pruebas simuladas y físicas:

Luego de desarrollar todo el código y que compilara exitosamente, se hizo una simulación del circuito con el fin de verificar de ante mano si su funcionamiento era el deseado:



Al final se montó el circuito en el laboratorio, corroborando su correcto funcionamiento, se muestra el video en el siguiente link:

https://drive.google.com/file/d/1YY3zxpOnH-IdSqbUC5JrKHl8_3_waTcH/view?usp=sharing

GitHub:

Todos los avances y proyectos individuales fueron registrados en el siguiente GitHub

https://github.com/juanmolineros-08/microcontroladores251/tree/main