

# Reporte de Diseño Proyecto 1 en L<sup>A</sup>T<sub>E</sub>X

Kevin Estuardo Munoz 15358

March 18, 2018

## **Abstract**

El proyecto 1 consiste en realizar un semáforo de cuatro vías implementando un microcontrolador PIC16F887; y el sistema implementado a escala.

## **1 Explicación del Semáforo Implementado**

### **1.1 Lineamientos y Requerimientos**

Los lineamientos para la implementación del semáforo son:

- Implementación a través de un microcontrolador, específicamente PIC16F887.
- Los tiempos deberán ser controlados a través de temporizadores del microcontrolador.
- La interfaz de usuario del semáforo deberá implementarse a través de interrupciones.
- El sistema a escala será implementado a través de focos incandescentes de 110 VAC.

Los requerimientos del proyecto son:

- La implementación debe contar con 4 modos: mañana, tarde, noche y manual.
  - Modo mañana: una vía (Zona 16) deberá tener prioridad sobre las demás.
  - Modo tarde: todas las vías deben tener la misma prioridad.
  - Modo noche: una vía (Zona 5) deberá tener prioridad sobre las demás.
  - Modo manual: los focos amarillos de las cuatro vías titilan, mientras los otros focos se mantienen apagados.

- La luz amarilla tendra un periodo de 3 segundos para los modos manana, tarde y noche; debera contar un un titileo de un periodo definido a discrecion.
- La seleccion del modo de funcionamiento debe realizarse a traves de una interfaz de usuario compuesta por 5 botones, uno para cada modo y un boton de reset.
- En el modo manual todos los focos amarillos titilan a una frecuencia de 2.5 segundos.

## 1.2 Diagrama de Timing

Los tiempos calculados cumplen con las especificacion, modo manana y noche una via tiene prioridad sobre las demas, y modo tarde la misma prioridad para todas las vias.

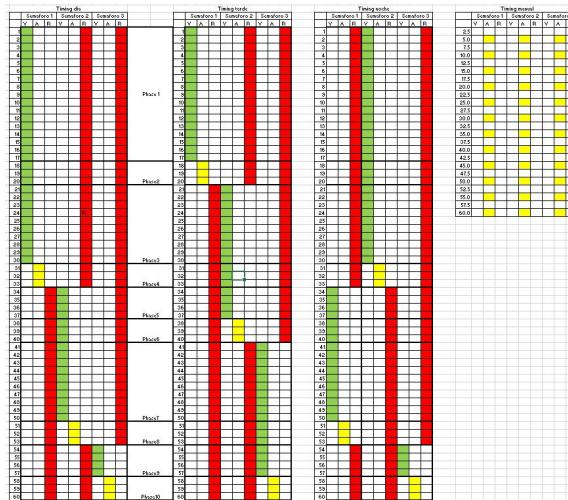


Figure 1: Diagrama de Timing

## 1.3 Fase de Potencia

Para esta fase se utilizaron optoacopladores, relays, transistores npn, diodos 1n4007, focos y flaponeras. Los optoacopladores se utilizaron para aislar el microcontrolador, los relays se utilizaron para encender los focos y controlar el voltaje AC y tierra, los transistores permiten el relay reciba la corriente necesaria para activar la bobina y los diodos se utilizaron como proteccion de flujo de corriente.

## 1.4 Implementacion Fisica

Para la implementacion fisica del semaforo se realizo un corte laser para hacer una caja de material mdf, con agujeros, para simular un semaforo de 4 vias, dentro de la cual se colocaron las flaponeras con los focos.

## 2 Diagrama de Bloques

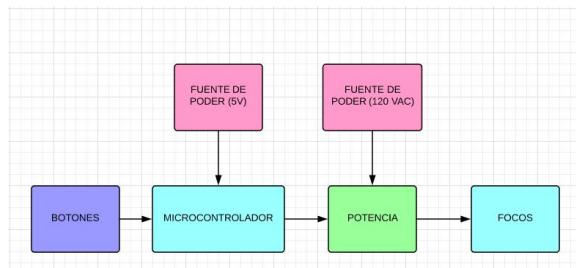


Figure 2: Diagrama de Bloques

### 2.1 Botones

Se realizo una interfaz de usuario con 5 botones, uno para seleccionar un modo, y uno para reset del sistema.

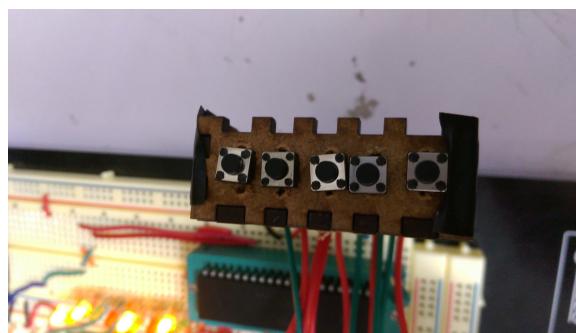


Figure 3: Interfaz con el Usuario

### 2.2 Microcontrolador

Se utilizo un microcontrolador PIC16F887 para la implementacion del codigo realizado.

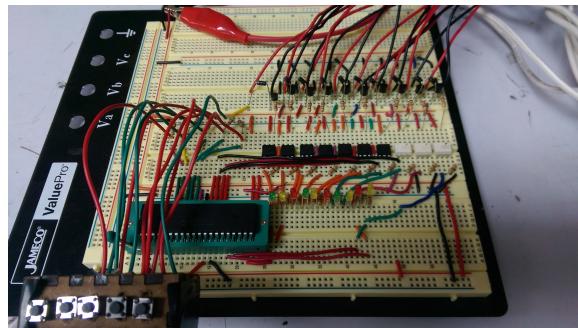


Figure 4: Circuito

### 2.3 Potencia

Se utilizaron los optoacopladores 4n35 (integrados) para aislar la parte de potencia a los focos y el microcontrolador, ya que dentro del integrado se encuentra un diodo emisor infrarrojo, y un fototransistor. La señal del microcontrolador enciende el led, el cual permite corriente entre colector y emisor en el fototransistor, y esta señal va a la base de un transistor 2n3904, el cual permite corriente entre colector (relay) y emisor (tierra). Se utilizó un relay de 120VAC activado por la señal del optoacoplador, y permite que la pata común se conecte con la pata de normalmente abierto para que el foco encienda.

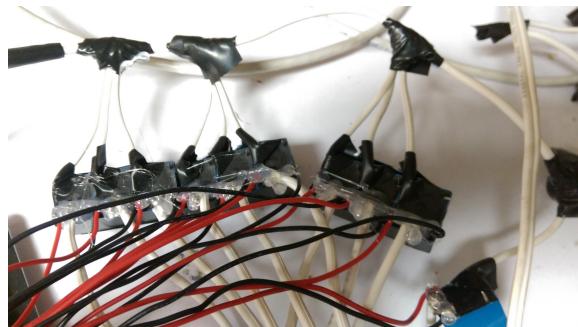


Figure 5: Potencia

### 2.4 Focos

Se utilizaron focos de 110VAC, flaponeras y cable paralelo. Para la conexión se utilizó el cable paralelo para conectar un voltaje común en una pata de todas las flaponeras, y en la otra pata de cada flaponera se conectó con cable paralelo a la pata de común del relay. La pata de normalmente abierto de los relays fue conectada con cable paralelo para la tierra de la espiga.

## **3 Firmware**

### **3.1 Oscilador**

- Se utilizo el oscilador interno del PIC, a una frecuencia de 8MHz.

### **3.2 Timer**

- Se utilizo TMR0, con prescaler de 1:128 y con N inicial igual a 61.

### **3.3 Interrupciones**

- Se utilizaron las interrupciones de TMR0 (T0IF), las interrupciones en PORTB (para RB0, RB1, RB2, RB3 Y RB4)

### **3.4 Entradas**

- Las entradas estan situadas en PORTB, especificamente en RB0, RB1, RB2, RB3 Y RB4, con modo manana, tarde, noche, manual y reset respectivamente.

### **3.5 Salidas**

- Se utilizo PORTC y PORTD como salidas digitales. Las salidas son
  - RD0 verde via 1
  - RD1 amarillo via 1
  - RD2 rojo via 1
  - RD3 verde via 2
  - RD4 amarillo via 2
  - RD5 rojo via 2
  - RD6 verde via 3
  - RD7 amarillo via 3
  - RC0 rojo via 3
  - RC1 amarillo via 4

### **3.6 Subrutinas**

- Se utilizo una subrutina DELAY, la cual al ser ejecutada realiza una serie de conetos en 1 variable, lo cual genera un espacio de tiempo en el que no se ejecuta otra instruccion.

### 3.7 Variables Importantes

- CONTROL se utilizaron los bits 0 (manana), 1 (tarde), 2 (noche) y 3 (manual) como banderas del modo en el que se encuentra el semaforo.
- PHASETIMER se utilizo como un contador, el cual se incrementa cada segundo.
- TIMERTWOFIVE se utilizo como contador, el cual se incrementa cada 2.5 segundos.
- CHANGEVAR se utilizo de la misma manera que la variable CONTROL, pero esta se cambia en las interrupciones para no afectar el ciclo actual del semaforo.

### 3.8 Pseudocodigo

- crear las palabras de configuracion.
- Crear las variables a utilizar.
- Vector de interrupciones
  - Deshabilitar interrupciones globales
  - verificar el estado del puerto de interrupcion (PORTB), y realizar el cambio del bit de estado segun el bit del puerto que cambio.
  - habilitar interrupciones globales
  - Limpiar la bandera de interrupciones del puerto.
- Realizar el setup necesario.
  - Activar interrupciones en puerto b.
  - Seleccionar frecuencia de Oscilacion.
  - Salidas como digitales.
  - habilitar los puertos como salidas o entradas.
  - Iniciar el timer 0
  - habilitar interrupciones globales
- Mover N inicial al timer 0
- Verificar el estado de bandera de interrupcion del timer 0.
- Si se activo, limpiarla y continuar.
- Si no se activo regresar a la verificacion.
- Incrementar un una variable contador.
- mover la variable contador a W.

- Restar 21 a w.
- Si se activa la bandera zero, continuar.
- Si no se activa la bandera zero, regresar a verificar bandera de interrupcion del timer 0.
- Incrementar variable contador de 1 segundo.
- Mover el contador a W, y restar el tiempo en el cual se produce un cambio de fase. (En la figura 1 se puede observar que el diagrama de timing se encuentra dividido en 11 fases, siendo cada una de ellas el momento en el que se produce un cambio de estado en los bits de salida del puerto C y D)
- Si se activa la bandera zero, continuar.
- Si no se activa la bandera zero, seguir verificando con el siguiente valor de tiempo.
- Activar los bits propios de cada fase.
- Si no se activa la bandera zero con algun valor de tiempo, regresar a verificar la bandera de interrupcion del timer 0.

## 4 Resultados Importantes



Figure 6: Implementacion

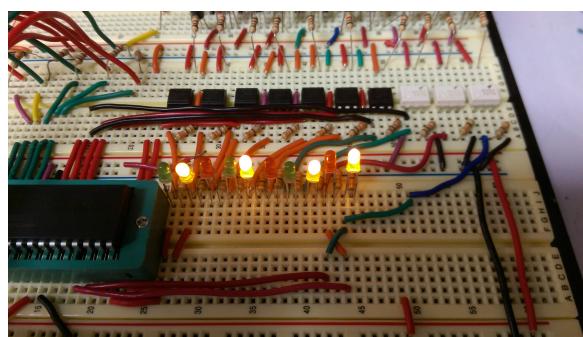


Figure 7: Funcionamiento del proyecto