

SEMÁFORO SEMICONTROLADO UTILIZANDO TRIACS

RESUMEN: En el presente informe se describe el procedimiento que se llevó a cabo para el diseño y montaje de un semáforo semicontrolado con TRIAC's, con todos sus componentes, el lenguaje utilizado en la programación del PIC y, además, el código fuente explicado detalladamente y los dataste de los mismos dispositivos utilizados.

Palabras clave: TRIAC, opto acoplador, BT136, DIAC.

OBJETIVOS

- Desarrollar un proyecto de la vida real con aplicación útil que funcione inteligentemente y se pueda reprogramar.
- Mediante el uso de pocos recursos económicos y de hardware lograr cumplir con el primer objetivo de manera eficaz y eficiente.
- Aplicar elementos de potencia y de lógica en la solución del problema.

INTRODUCCIÓN

En la electrónica una de las más importantes e interesantes áreas con una amplia difusión hoy día es la automatización y el control no sólo porque hoy en día tiene una aplicación en casi todos los campos sino porque permite crear una cantidad ilimitada de modelos al igual que una ilimitada variedad de aplicaciones prácticas. Se puede utilizar en el hogar, en el colegio, en la industria, en el comercio, en la investigación, en fin, en todos los campos del conocimiento y del quehacer humano puede haber una aplicación de la automatización y el control.

Además, en nuestro caso, nos permite una aproximación a la programación y a la aplicación práctica de los conocimientos obtenidos teóricamente en clase, puesto que mediante un micro se puede hacer aún más fácil la aplicación de estos conceptos de control.

CONDICIONES DE DISEÑO Y FUNCIONAMIENTO

Antes de entrar en la etapa de diseño tuvimos en cuenta que nuestro proyecto cumpliera con las siguientes especificaciones:

- Debe estar alimentado de la red eléctrica.
- Reprogramable.
- Deberá soportar condiciones de uso real, como voltaje de la red y temporizaciones.
- Se deberán usar lámparas de potencia y los dispositivos necesarios para operar esta parte del circuito, ya sean tiristores, TRIAC's o cualquier otro elemento adecuado.
- Los tiempos deben ser reales.
- El circuito de control y lógica debe ser eficiente, sencillo de programar, de fácil consecución sus elementos y económico.
- El diseño del programa debe ser eficiente y utilizar conceptos planteados en la teoría. Además debe tener se en cuenta el uso de memoria.
- El lenguaje a utilizar debe permitir hacer ajustes fáciles al código en cualquier etapa del proyecto.
- La alimentación de este circuito de control se debe obtener también de la red.

MÉTODO DE DISEÑO

Al comenzar a trabajar en el proyecto lo primordial fue definir el proyecto, luego los materiales a utilizar y después el lenguaje a utilizar para programar el PIC.

Después de tener definido el proyecto, nuestros siguientes pasos fueron los siguientes:

- Definir el tipo de semáforo a diseñar: peatonal, doble salida, sencillo. Esto con el fin de definir los elementos necesarios para su construcción.
- Escoger el tipo de TRIAC's y opto acopladores a emplear, teniendo en cuenta la corriente y el modo de operación, si transistorizados o con DIAC.
- Elegir entre los diferentes tipos de MOC30XX el que se adaptase a las condiciones de diseño nuestras.
- Elegir los reguladores, puente o zéner a utilizar para proporcionar Vcc al micro y al circuito de la lógica.
- Escogimos entonces el PIC, que no es otro más que el que utilizamos durante el semestre en las diferentes aplicaciones.
- Definimos el lenguaje a utilizar para el proyecto.

JUSTIFICACIÓN DE ELECCIÓN

- Escogimos un semáforo para vehículos sencillo de un solo sentido (aunque en la programación se dejó la posibilidad de habilitar otro desde el mismo PIC en sentido contrario), por la facilidad del diseño y por el costo de los materiales. También, con el fin de dejar el semáforo como elemento de educación para los niños de un preescolar de la ciudad se pensó que era más viable hacerlo lo más simple posible sin obviar nada de los componentes electrónicos tanto de control como de programación.
- Se escogió el TRIAC BT136 por su capacidad de soportar niveles altos de

tensión (hasta 600V) y una corriente considerable (4 A) ya que deben ir conectados directamente a la red. Se decidió utilizar un opto acoplador con detector de cruce por cero para hacer la transición entre estados menos brusca.

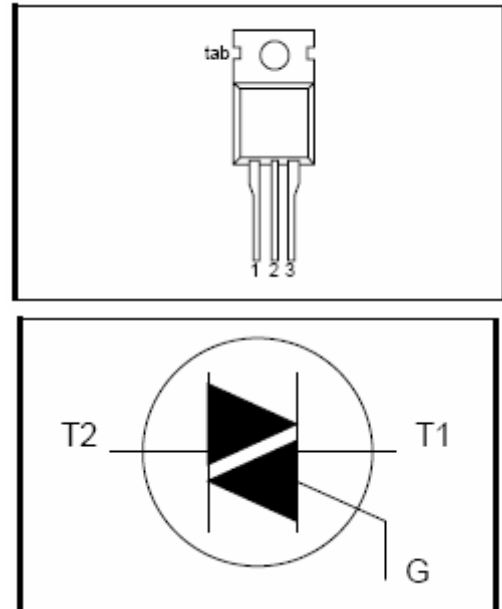


FIGURA 1. TRIAC utilizado en el proyecto (BT136) y distribución de pines.

- Se utilizó el MOC3041 no sólo porque tiene una capacidad de soportar el voltaje AC hasta 115V, lo cual es suficiente para nuestra aplicación, sino por su detector de cruce por cero y su configuración con snubber para recortar los picos de corriente y proteger el circuito de control del de potencia.

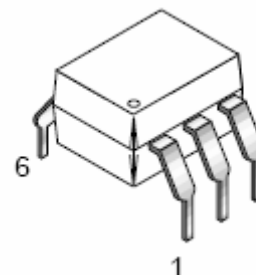


FIGURA 2. Presentación del encapsulado del MOC3041 con detector de cruce por cero.

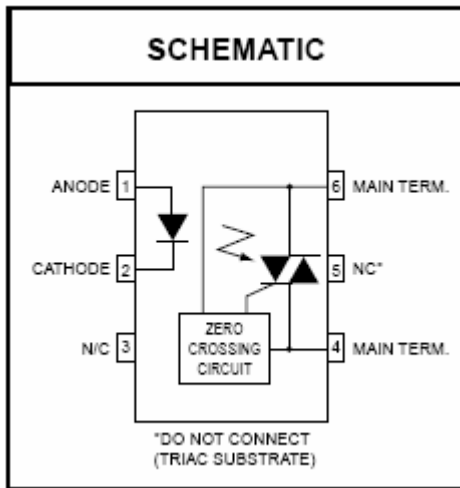


FIGURA 3. Distribución interna de componentes y asignación de pines del MOC3041.

- d. Elegimos una configuración de fuente con un puente de potencia y luego sólo con un diodo zéner y una resistencia de potencia para brindarnos el V_{cc} para el PIC. Esto lo hicimos utilizando los siguientes cálculos y teniendo en cuenta que el rizado era mínimo, lo cual no afectaba la operación del micro:

$$R = \frac{(V_{in} - V_{sal})}{I_{total} + I_{zener}}$$

$$R = \frac{(170 - 5.1)}{14mA + 5mA}$$

$$R = 8678.94\Omega$$

$$Potencia \Rightarrow P = I^2 * R$$

$$P = (19mA)^2 * 8678.94$$

$$P = 3.2W$$

- e. La corriente del circuito de control la medimos utilizando una fuente normal

y aplicamos las fórmulas del zéner. Utilizamos un zéner de 5.1 y una resistencia de potencia de 8.2K y 5W, para asegurarnos.

- f. Utilizamos el PIC 16F877A no sólo porque es el que hemos venido utilizando durante todo el semestre, sino porque es el de más fácil consecución, mejor precio y que presenta las mejores prestaciones dentro de su rango.

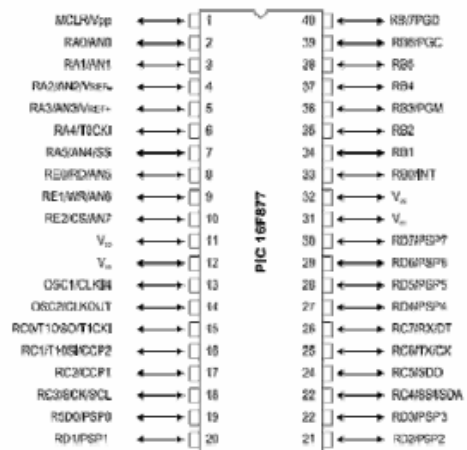


Figura 4. PIC 16f877A.

- g. Por último, la elección del lenguaje a utilizar para programar el PIC, fue crítica puesto que en la materia hemos venido trabajando en lenguaje C y es muy fácil trabajar con él por su alto nivel. Sin embargo, como debíamos estar haciendo modificaciones constantes sobre las temporizaciones y otras variables del programa, optamos por el ASSEMBLER que nos permite acceder más fácilmente a estas.

DIAGRAMA DE BLOQUES

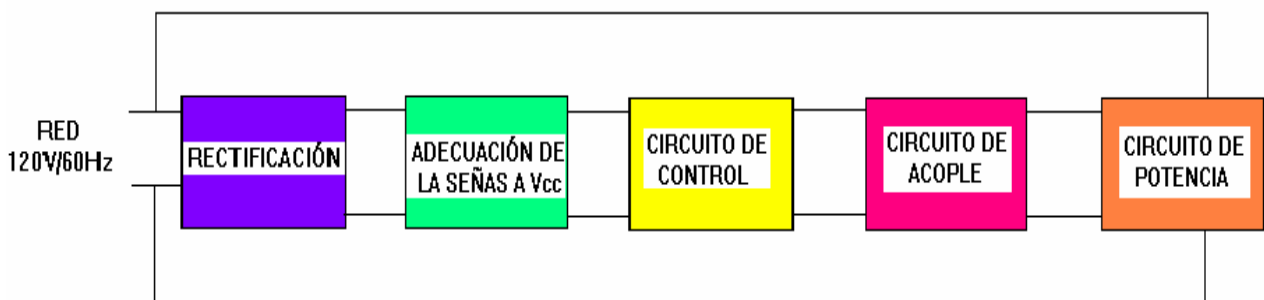
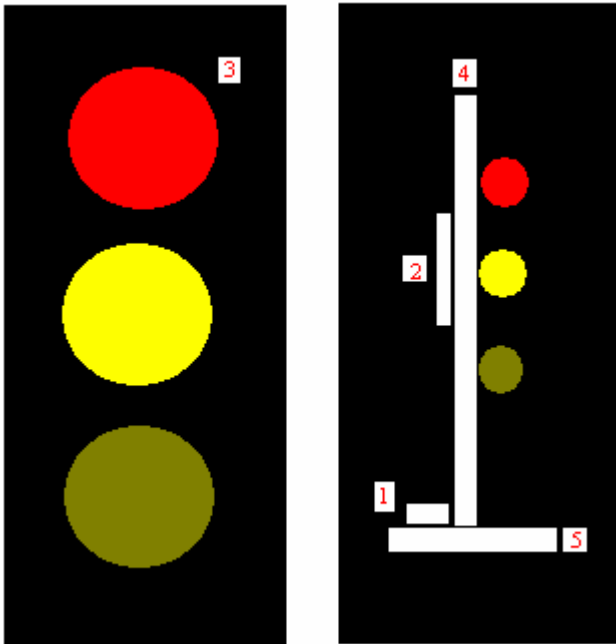


DIAGRAMA SEMÁFORO



1. PIC 16F877A, BT136 (TRIAC), MOC3041 (opto acoplador).
2. Puente de potencia, resistencia de potencia, zéner 5.1., regleta.
3. Lámparas de 120V/100W.
4. Asta central.
5. Soporte y anclaje.

CONCLUSIONES

- Nos pudimos dar cuenta que es muy fácil realizar casi cualquier programa o montaje utilizando tanto lenguaje C como el ASSEMBLER, sabiendo de dónde partir y cuáles serán los objetivos propuestos.
- La compilación de todos los códigos deben pasar invariablemente a ASSEMBLER pero es mucho más fácil programar en lenguaje C, aunque a veces para afinar detalles se puede ir al código básico y realizarlo mejor, como en nuestro caso.
- A la hora del montaje encontramos dificultades especialmente a la hora de calibrar los disparos de los TRIAC's con los opto acopladores aún utilizando los componentes más idóneos para el caso.
- El análisis y perfecta medición de los parámetros de potencia fue de vital importancia para el diseño de la fuente zéner y para la protección del circuito de control, aunque el opto acoplador debería también cumplir esa función.
- Nos queda con marcado interés la conclusión casi obvia de tener en cuenta los imprevistos a la hora de realizar cualquier montaje pues, aunque la electrónica digital es en sí bastante precisa al contrario de la analógica, siempre hay inconvenientes y retrasos, generalmente debidos al factor humano.
- También como en estos proyectos se combinan factores analógicos, mecánicos y digitales, esta confluencia hace aún más crítico el tiempo de preparación de los proyectos. Por ejemplo para el presente la falla en un TRIAC y un MOC y su no tan fácil consecución en Ibagué para reponerlos.
- Por último, al ir desarrollando el proyecto nos dimos cuenta de que hay muchísimas aproximaciones a un mismo problema y que es necesario

tener en cuenta sobre todos dos aspectos: mínimo número de componentes y costos. Teniendo en cuenta estos factores se puede llevar a cabo un trabajo eficiente, sin tener en cuenta estos ítems se puede llevar a cabo un trabajo sólo

eficaz. También en el presente proyecto se tuvo muy en cuenta la accesibilidad al programa para realizar cambios rápidos en las temporizaciones.


```
T2
  MOVLW    B'00000010'
  MOVWF    PORTA
  MOVLW    B'11111101'
  MOVWF    PORTE
  CALL     RETARDO2
```

```
T3
  MOVLW    B'00000100'
  MOVWF    PORTA
  MOVLW    B'11111011'
  MOVWF    PORTE
  CALL     RETARDO1
```

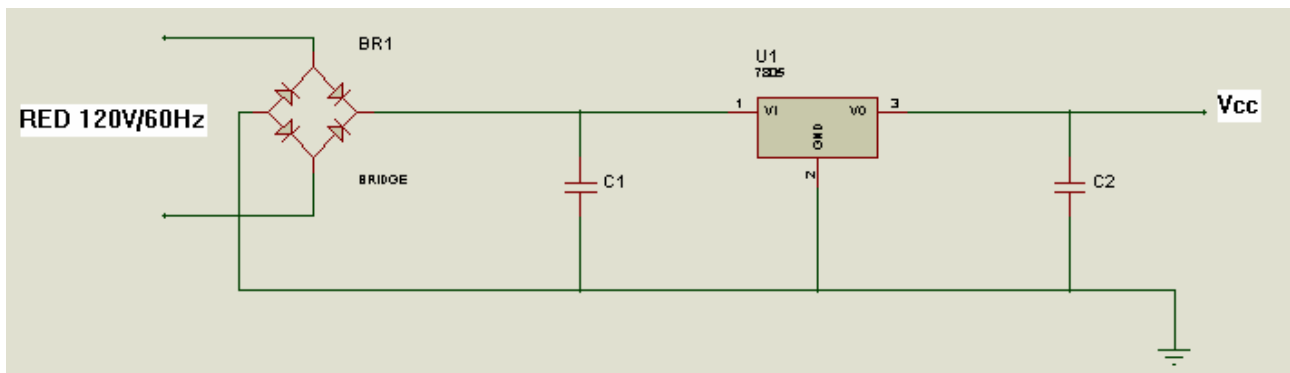
```
T4
  MOVLW    B'00000010'
  MOVWF    PORTA
  MOVLW    B'11111101'
  MOVWF    PORTE
  CALL     RETARDO2
  GOTO    T1
  END
```

```
;
```

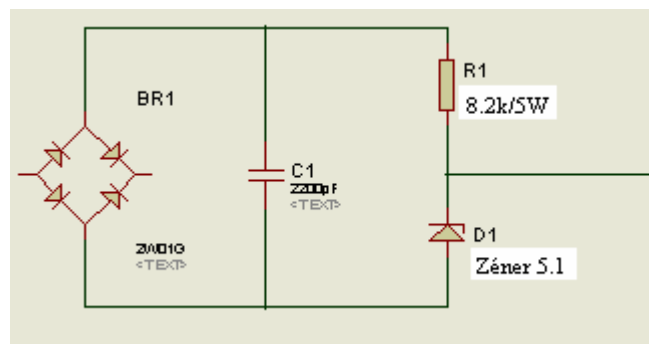
END

ANEXO 2. CIRCUITOS MONTADOS

CIRCUITO DE RECTIFICACIÓN Y ADECUACIÓN DE LA SEÑAL (REGULACIÓN)

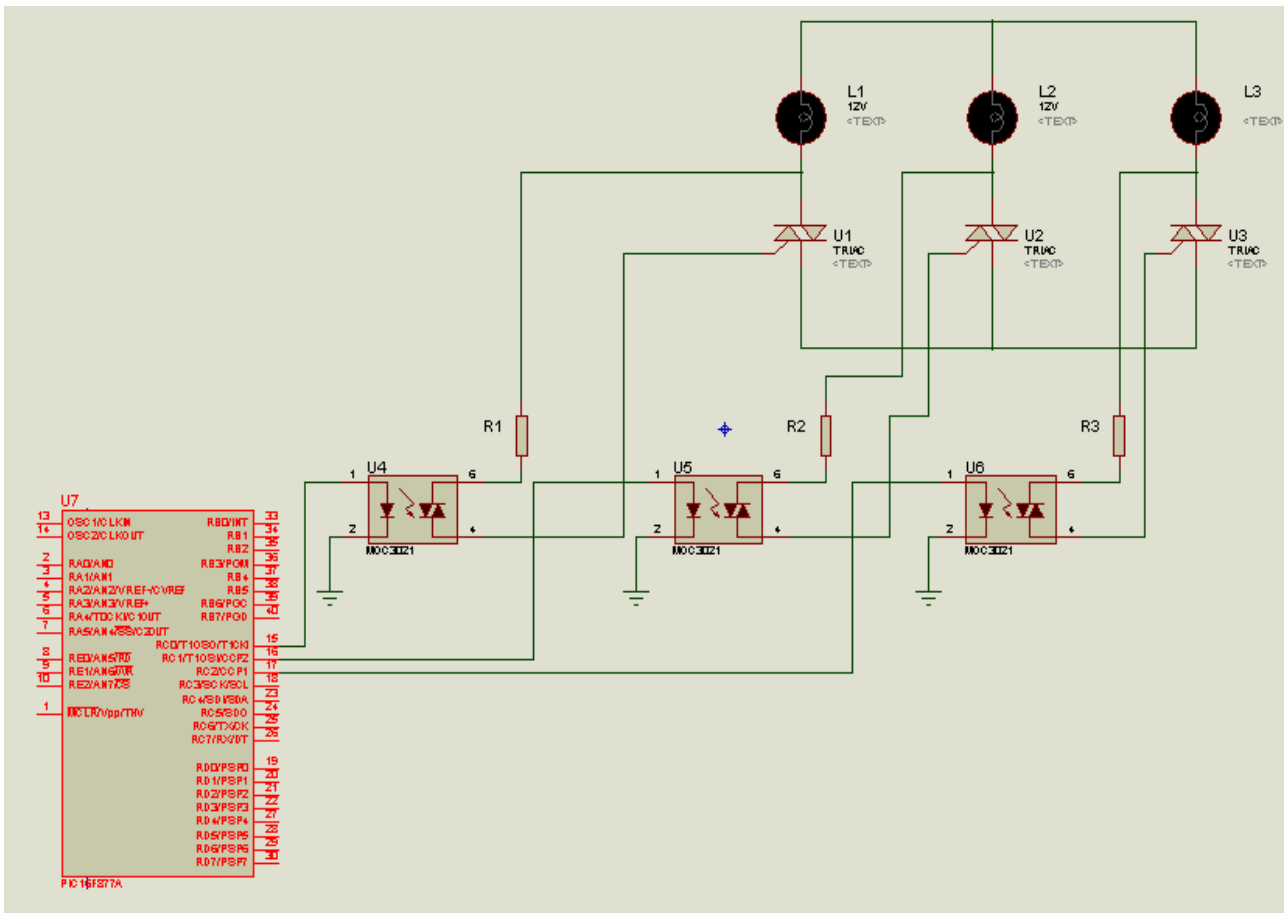


En este circuito luego se prescindió del LM7805 y se ubicó en su lugar un zéner y una resistencia de potencia que proveían el Vcc del micro sin importar el rizo que era de 0.05% (Tolerable). El circuito que montamos finalmente quedó de la siguiente manera:

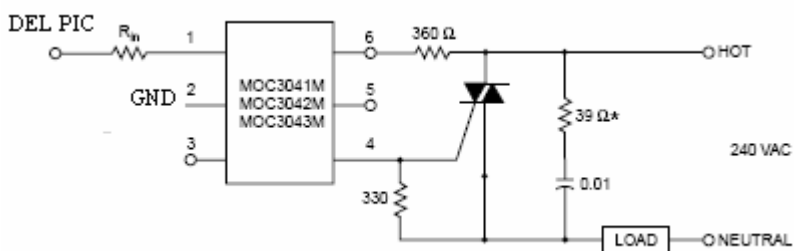


Zéner usado.

CIRCUITOS DE CONTROL, ACOPLE Y POTENCIA



Los pines de salida del PIC fueron 2 (RA0), 3(RA1) y 4(RA2).



$$R_{in} = V \text{ del PIC} / I_{MOC}$$

Tanto el voltaje que sale del PIC como la corriente del MOC, se obtuvieron de los parámetros del DATASHEET de cada uno.

$$R_{in} = 5V / 15mA$$

$$R_{in} = 330 \text{ Ohmios}$$

