UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACIÓN DE PROYECTOS PARA INGENIERÍA ELECTRÓNICA

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

JOSUE DAVID PALACIOS MARROQUÍN
ASESORADO POR EL ING. CARLOS EDUARDO GUZMÁN SALAZAR

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, SEPTIEMBRE DE 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Inga. Elvia Miriam Ruballos Samayoa
VOCAL IV	Br. Walter Rafael Véliz Muñoz
VOCAL V	Br. Sergio Alejandro Donis Soto
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. Julio César Solares Peña
EXAMINADOR	Ing. José Antonio de León Escobar
EXAMINADOR	Ing. Marvin Marino Hernández Fernández
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACIÓN DE PROYECTOS PARA INGENIERÍA ELECTRÓNICA

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería de Mecánica Eléctrica, con fecha 30 de octubre del 2012.

Josue David Palacios Marroquin

Guatemala, 15 de Julio de 2013

Señor
Coordinador Área de Electrónica
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería
Universidad de San Carlos de Guatemala

Estimado Coordinador

Por este medio hago de su conocimiento que he finalizado la revisión del trabajo de graduación del estudiante Josué David Palacios Marroquín, titulado "PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACION DE PROYECTOS PARA INGENIERIA ELECTRONICA". Considerando que el mismo cumple a cabalidad los objetivos propuestos al momento de su aprobación.

Por lo que, lo remito a su persona para que sirva continuar con el tramite respectivo indicando además que, tanto el autor como el suscrito en calidad de ASESOR NOMBRADO, somos responsables del contenido del trabajo de graduación.

Atentamente,

Carlos Guzmán Salazar
ASESOR

CARLOS GUZMAN SALAZAR
Ingeniero Electricista
Col. No. 2762

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Ref. EIME 56. 2013
Guatemala, 17 de JULIO 2013.

Señor Director
Ing. Guillermo Antonio Puente Romero
Escuela de Ingenierìa Mecànica Elèctrica
Facultad de Ingenierìa, USAC.

Señor Director:

Me permito dar aprobación al trabajo de Graduación titulado: PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACIÓN DE PROYECTOS PARA INGENIERÍA ELECTRÓNICA, del estudiante Josué David Palacios Marroquín que cumple con los requisitos establecidos para tal fin.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente,
ID Y ENSEÑAD A TODOS

Ing. Carlos Eduardo Guzmán Salazar
Coordinador Área Electrónica

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

sro

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

REF. EIME 57. 2013.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área, al trabajo de Graduación del estudiante; JOSUÉ DAVID PALACIOS MARROQUÍN titulado: PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACIÓN DE PROYECTOS PARA INGENIERÍA ELECTRÓNICA, procede a la autorización del mismo.

Ing. Guillermo Antonio Puente Romero

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 19 DE AGOSTO 2,013.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 658.2013

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **PROPUESTA DE DISEÑO DE PLACA ENTRENADORA DE MICROCONTROLADORES PROGRAMABLES PIC PARA EL DESARROLLO Y ELABORACIÓN DE PROYECTOS PARA INGENIERÍA ELECTRÓNICO,** presentado por el estudiante universitario **Josué David Palacios Marroquín,** autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Murphy Olympo Paiz Recinos
Decano

Guatemala, 24 de septiembre de 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

/gdech

ACTO QUE DEDICO A:

Dios	Quien es el dador de toda fuerza, gracia y conocimiento, razón principal de mi existir a quien debo mi vida y dedicación.
Mis padres	David Palacios Castañeda y Carmela Marroquín, por ser la columna de mi formación personal y académica; a quienes agradeceré toda mi vida; el apoyo que nos dieron a mis hermanos y a mí, transcenderá generaciones; gracias a ellos que decidieron romper esa barrera, Dios me los bendiga siempre.
Mi novia	Zulian Azucena García Elías, por su apoyo incondicional en todo momento; una de mis fuentes más grandes de apoyo e inspiración.
Mis hermanos	Abdiel, Kevin y Marvin Palacios Marroquín, con quienes nos hemos apoyado siempre y lo seguiremos haciendo.

AGRADECIMIENTOS A:

Mis amigos de la Facultad	Eduardo José Miralles, Luis Carlos Urízar, Álvaro Luis Godoy, David Ovalle, Edy Aguilar, Julio Estuardo Chan, Javier Rosales, Mario Morales y muchos otros, con quienes nos apoyamos en los cursos y proyectos.
Mi asesor	Por apoyarme con su conocimiento en mi trabajo de graduación y depositar su confianza en mi persona para realizar este tema.
La Universidad de San Carlos de Guatemala	Mi casa de estudios; incluyendo también al pueblo de Guatemala, quienes contribuyeron con mi desarrollo profesional.
Facultad de Ingeniería	Por ser una importante influencia en mi carrera y la institución académica a quien debo mi título.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
IC	Circuito integrado
hz	Hertz, frecuencia
hr	Hora
KHz	Kilohertzio (1000 Hertz)
kΩ	Kilohmios
MHz	Megahertz (10000 Hertz)
µF	Microfaradios
mm	Milímetro
min	Minuto
Ω	Ohm, unidad de resistencia eléctrica
RL	Relevador o relé
R	Resistencia
seg	Segundo
SW	*Switch*
GND	Tierra *(ground)*
V	Voltios

GLOSARIO

Circuito integrado *E*s una pastilla pequeña de material semiconductor, de algunos milímetros cuadrados de área, sobre la que se fabrican circuitos electrónicos, generalmente mediante fotolitografía y que está protegida dentro de un encapsulado de plástico o cerámica.

Conversor A/D Dispositivo electrónico capaz de convertir una entrada analógica de voltaje a un valor binario.

Compilador Es un programa informático que traduce un programa escrito en un lenguaje de programación a otro, generando un programa equivalente que la máquina será capaz de interpretar.

Display Dispositivo electrónico compuesto por 8 segmentos emisores de luz, que es utilizado para representar valores de numéricos encendiendo secuencialmente sus luces.

Frecuencia Es una magnitud que mide el número de repeticiones por unidad de tiempo de cualquier fenómeno o suceso periódico.

Oscilador eléctrico Circuito integrado creado para obtener un sistema de oscilación que sea estable y periódico, manteniendo

	una frecuencia y una forma de onda constante. Para ello se aprovecha el proceso natural de oscilación amortiguada, que poseen los circuitos compuestos por elementos capacitivos o inductivos o ya sea inducida por un cristal de cuarzo.
PCB	En español quiere decir placa de circuito impreso, que es una superficie construida por pistas de cobre sobre una base de material no conductor que interconecta distintos componentes electrónicos.
PIC Simulator IDE	Programa dedicado a la simulación y programación de microcontroladores PIC, creado por la empresa Oshonsoft.
Pulso Eléctrico	Es un nivel de voltaje instantáneo que puede tener un periodo y frecuencia predeterminado, que en términos digitales se representa como un uno lógico.
RISC	En inglés: *reduced instruction set computing*; en cir computador con conjunto de instrucciones reducidas, el cual corresponde a la familia de controladores a los que el PIC pertenece, por las instrucciones de tamaño fijo que se presentan en un reducido número de formatos.

RESUMEN

Los microcontroladores programables PIC son dispositivos que han sido creados para solucionar tareas simples a través de la electrónica de bajo costo, que su vez son el preámbulo para los controladores programables de mayor nivel, que básicamente se centran en crear proyectos electrónicos que ayudan a realizar tareas automáticas pregrabadas en estos, con ayuda de un lenguaje de programación relativamente simple, con una cantidad de instrucciones limitadas pero suficientes.

Este trabajo de graduación ha sido creado para formular una propuesta de placa electrónica para realizar proyectos electrónicos de una forma más eficiente y simple, para así no tener que realizar una placa diferente para cada proyecto, sino que se reutiliza el mismo hardware que se ha creado, evitando gastos innecesarios y los problemas que aparecen cada vez que se crea una nueva placa electrónica, tales como los falsos contactos, pistas defectuosas y defectos de diseño, para crear una placa perfeccionada para este fin.

OBJETIVOS

General

Proponer el diseño de una placa entrenadora de microcontroladores PIC de bajo costo, como herramienta para el desarrollo de proyectos de la carrera de Ingeniería Electrónica.

Específicos

1. Presentar las propiedades básicas de los microcontroladores programables PIC, para entender el funcionamiento de ellos y obtener un concepto general de este dispositivo.

2. Presentar un diseño de placa entrenadora que ayude al estudiante a ser más práctico, eficiente y ordenado, a la hora de realizar el diseño general de cualquier proyecto electrónico.

3. Proponer un entorno de didáctico programación, que contenga herramientas prácticas de simulación, para realizar el proyecto de una forma más precisa.

4. Proponer un procedimiento para la creación de placas de circuito impreso, que sea más económica y con pistas de cobre más definidas sobre la placa, para evitar los defectos de impresión y falsos contactos entre dispositivos.

5. Presentar una serie de ejemplos prácticos que ayuden la asimilación del código de programación que interconecta las instrucciones programadas en el PIC, con cada dispositivo conectado a él.

INTRODUCCIÓN

El presente trabajo de graduación ha sido desarrollado para solventar algunas debilidades a las que se enfrentan los estudiantes de las carreras universitarias que pertenecen a la Escuela de Mecánica Eléctrica.

Se realiza un enfoque preciso en el curso de Electrónica 3, que es el primer curso que se centra en introducir al estudiante en la electrónica digital; pero aún así, se presentan debilidades en el tópico de los microcontroladores PIC, que al mismo tiempo de ser un tema nuevo, la falta de herramientas o un documento especializado, son unos de los problemas más grandes para la asimilación de estos temas, ya que son considerados de “cultura general” para cualquier ingeniero eléctrico o electrónico.

En la parte inicial se presentan conceptos básicos necesarios para comprender qué es un microcontrolador y su funcionamiento, así como las partes principales del mismo, dando a entender un panorama general sobre este dispositivo.

Después se presenta la propuesta de placa entrenadora de microcontroladores PIC 16F877 – 16F877A nombrada JP11, junto con la descripción de los dispositivos necesarios para poder construirla y la explicación de todos los circuitos que la conforman, desde el energizado del PIC hasta la conexión de los dispositivos de entrada y salida como teclado, pantalla LCD y dispositivos electromecánicos al microcontrolador.

En seguida, se muestra el programa de simulación PIC Simulator IDE, que será una herramienta con la que se hará introducir al tema de la programación de microcontroladores programables PIC. En este capítulo también se incluye una la explicación de los comandos, sintaxis, palabras reservadas y las partes del simulador que se utilizarán para realizar los proyectos con PIC.

Se continuará con la presentación del procedimiento para realizar cualquier placa electrónica de circuito impreso y consejos prácticos para llevar a cabo la creación de la placa electrónica presentada en el capítulo 2.

Luego se presenta una serie de ejemplos prácticos que incluyen la explicación de cada línea de programación, que describen los procedimientos y programas, desde lo más básico como leer el ADC del PIC, presentación de texto en la pantalla LCD y reconocer valores de frecuencia, hasta la implementación de un proyecto completo, interconectando cuatro motores DC, al microcontrolador.

Por último, se muestra la explicación sobre cómo realizar el proceso de grabación de los programas al microcontrolador por medio de la placa electrónica Mini QL-2006, a través del programa QL-Progen.

1. MICROCONTROLADORES

Los PIC son una familia de microcontroladores tipo RISC derivados del PIC1650, creado por la Compañía General Instrument; el nombre dado a los microcontroladores PIC es un diminutivo del nombre original PICmicro que quiere decir *peripheral Interface controller* (control de interfaz periférico)

1.1. Arquitectura central

- Arquitectura Harvard.
- El contador de programa está también relacionado dentro del espacio de datos, y es posible escribir en él.
- Posee una pila de hardware para almacenar instrucciones de regreso de funciones.
- Contiene un solo acumulador puede ser utilizador para cualquier instrucción.
- Se pueden utilizar las posiciones de la RAM como registros de origen y/o destino de operaciones aritméticas, lógicas, entre otras.
- Tiene una cantidad reducida de instrucciones de longitud fija.

Un microcontrolador está conformado de varias partes que están internas a su encapsulado de las cuales se tiene:

- El procesador donde se realizan el análisis, cálculo y direccionamiento de los procesos.
- Los buses donde se transportan los datos y las instrucciones.
- Memorias de almacenamiento de programas y datos
- Periféricos de conexión de conexión de dispositivos de expansión
- Puertos de entradas y salidas

1.2. Estructura de un microcontrolador

La arquitectura de un procesador determina cómo será distribuida la memoria y la información sobre las memorias.

Los microprocesadores utilizan la arquitectura Harvard, porque de esta manera se pueden controlar más versátilmente el tamaño de los buses de datos y de direccionamiento, con el único inconveniente que se necesitan más pines del CPU para interconectar de forma independiente las memorias de datos y de programa.

Hay que tener en cuenta que la ventaja es que los buses o canales que transportan esta información transcurren por separado, provocando que el procesador funciones más velozmente, caso contrario con la arquitectura Von Newman que por un único canal transporta tanto datos como direccionamiento.

En los microcontroladores PIC, la interconexión de ambas memorias es más fácil con esta arquitectura, porque están en la misma estructura que en este caso es el encapsulado del circuito integrado.

En el mismo se podrán encontrar los distintos estilos utilizados para la realización completa del formato de la tesis.

1.3. Memoria

Una de las características principales de la memoria de los microcontroladores es que siempre está adentro del encapsulado y asimismo no tiene grandes capacidades solo de apenas unos kilobits, porque las instrucciones a realizar usualmente no requieren poca memoria; estos contienen cuatro tipos diferentes de memorias tales como la memoria RAM, ROM, EPROM y OTP.

- La memoria RAM está destinada al almacenamiento de información temporal que será utilizada por el procesador para realizar cálculos u otro tipo de operaciones lógicas.

- La memoria ROM es un tipo de memoria volátil que es borrada cuando se vuelve a grabar sobre ella; en ella se almacenan las instrucciones del programa que va a desarrollar el programa.

- OTP (*one time programmable*) es un tipo de memoria similar a la ROM, solo que esta solo puede ser grabada una única vez.

- EPROM (*erasable programmable read only memory*) es un tipo de memoria volátil que debe exponerse a luz ultravioleta para que sea

borrada y así poder grabar de nuevo en ella, a diferencia de las memorias tipo EEPROM, que se pueden borrar eléctricamente.

1.4. Periféricos

Los periféricos de un microcontrolador están ubicados en los pines del encapsulado del PIC; los cuales serán descritos como puertos del microcontrolador PIC, estos se encuentran identificados por subíndices para que puedan ser localizados más fácilmente en el diseño cuando se quiera interconectar elementos externos al PIC y se declaren los puertos que serán utilizados en la programación, como por ejemplo cuando se declaran los puertos “E” de PIC como entrada de señal externa, estos pueden ser encontrarse sencillamente al verificar la hoja de datos del PIC.

1.5. Puertos de entradas y salidas de propósito general

Los periféricos de propósito general están distribuidos en segmentos de 8 pines del microcontrolador; estos son necesarios cuando se necesita accionar relevadores, motores, leds o cualquier otro dispositivo electrónico que pueda interpretar un 1 o 0 lógico, tomando como 1 a un valor aproximadamente de 5 V saliente del PIC y como 0 lógico, a un valor 0 V saliente del PIC (microcontrolador).

Cuando un periférico es configurado como puerto de entrada, este recibe niveles lógicos de voltaje para determinar cuándo un puerto es activado externamente al PIC o desactivado; en esta configuración se reconoce 5 V como 1 y 0 V como 0.

1.6. Temporizadores y contadores

Los microcontroladores tienen pines asignados para el conteo de pulsos y para conectar un temporizador; comúnmente se utilizan los cristales para realizar esta tarea de temporizarlo; estos pulsos son necesarios para la implementación de relojes, contadores, y medidores de frecuencia; cuando se asigna un pin del microcontrolador como contador, este recibe pulsos; los mismos son contabilizados por el PIC y se les asigna valor para poder analizarlos en la programación.

Por otra parte los microcontroladores tienen 1 o 2 pines para conectar el cristal al PIC; estos pulsos son como el corazón del microcontrolador que hace que este funcione; existen microcontroladores que tienen cristal interno por lo que se provoca un ahorro de pines para conectar el cristal externo.

1.7. Conversor A/D (analógico/digital)

Los conversores analógico/digital son los periféricos con mayor aplicación para los microcontroladores porque permiten asignarle un valor digital a una señal analógica menor a 5 V; este valor digital puede tomar resoluciones de 8 a 10 bits en un microcontrolador básico.

Una de las aplicaciones que se da a estos periféricos en la placa a que se hace referencia, es la implementación del teclado que se encuentra conectado a un diferencial que envía diferente niveles de voltaje al puerto A0 del PIC, que con apoyo de un segmentos, de programación, se analizan estos niveles que cambian, dependiendo del botón presionado para identificar la acción que solicita el usuario.

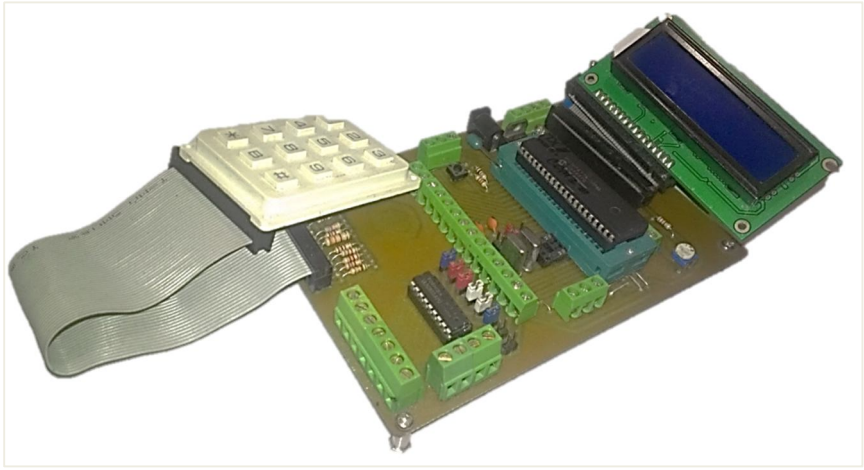
1.8. Comparadores

Compuestos de amplificadores operacionales, son utilizados para comparar dos señales analógicas y dar como salida un valor lógico de 1 o 0 dependiendo del resultado de comparación; el microcontrolador tiene la capacidad de tomar la lectura de voltaje de entrada de sus puertos analógicos que almacenan esta información en memoria; mediante esta medición se podrá tomar decisiones a través de su programación realizando funciones como activar o desactivar dispositivos por medio de cualquier señal de salida o simplemente mostrar la información en la pantalla LCD conectada al PIC.

2. PRESENTACIÓN DE PLACA ENTRENADORA JP11 PARA PIC16F877 – PIC16F877A

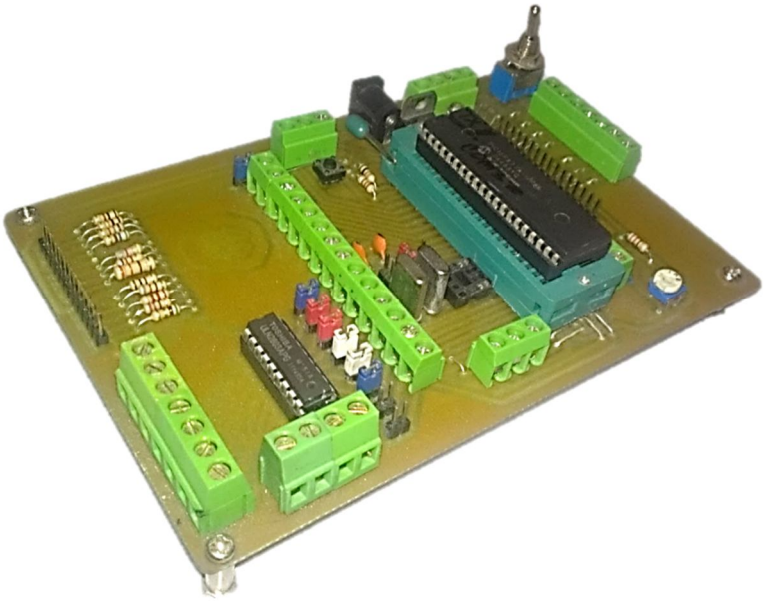
La placa entrenadora con el nombre JP11 ha sido diseñada para poder realizar la mayor cantidad de proyectos posibles y al mismo tiempo solventar los conocimientos necesarios sobre el tema de los microcontroladores para los estudiantes de ingeniería electrónica y eléctrica; esta placa se presenta como una herramienta para incursionar en el nodo de unión entre software y hardware que puede proporcionar soluciones desde las más simples y complejas al menor costo; a continuación se presenta una vista general sobre la placa entrenadora que será explicada a detalle, sección por sección en los siguientes incisos.

Figura 1. **Placa JP11**



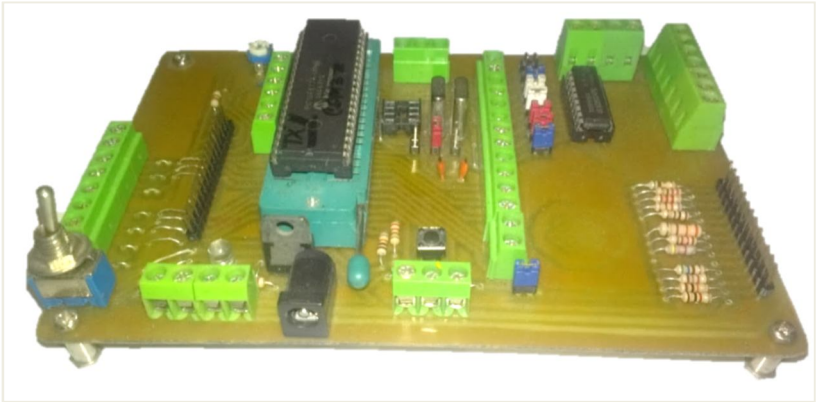
Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 2. **Placa JP11 sin teclado y LCD**



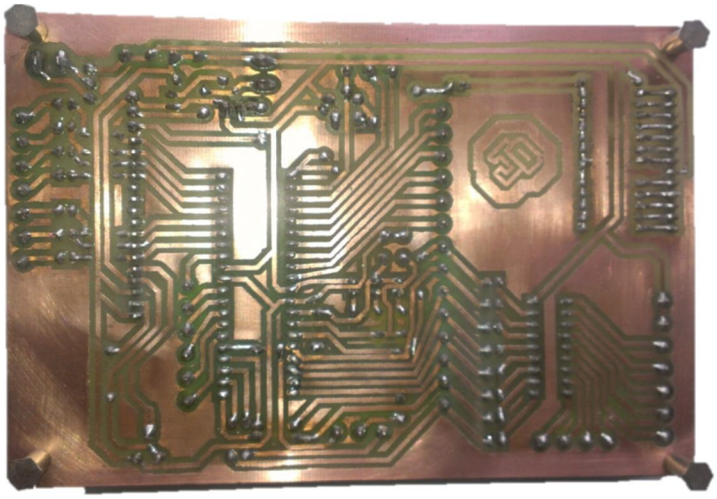
Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 3. **Anverso placa JP11**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 4. **Vista inferior de la placa**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

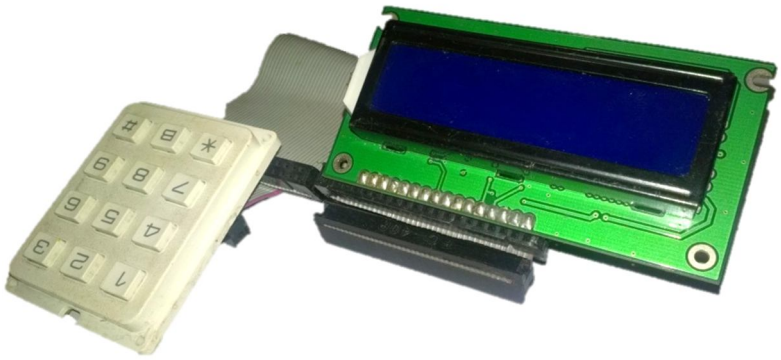
En la placa se pueden conectar diferentes dispositivos adicionales tales como motores de paso, bancos de leds, motores DC y pantallas LCD 2x16 tanto en verde como en azul.

Figura 5. **Dispositivos adicionales**

Fuente: taller propio, Ciudad de Guatemala

La placa contiene 2 series de pines IDE que han sido destinados para conectar un teclado matricial como dispositivo de entrada, y pantallas LCD 2x16 como dispositivo de salida, agregando versatilidad a los proyectos desarrollados.

Figura 6. **Teclado de nueve dígitos y LCD**

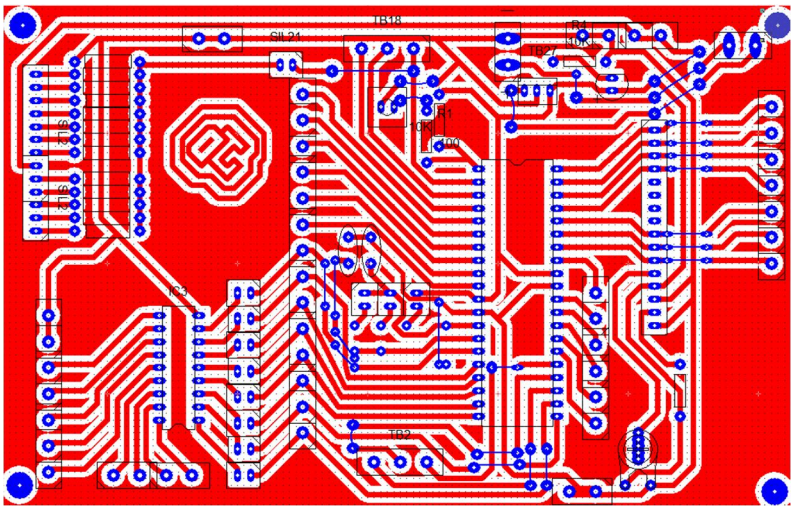


Fuente: taller propio, Ciudad de Guatemala

2.1. Diagramas esquemático de la tarjeta impresa

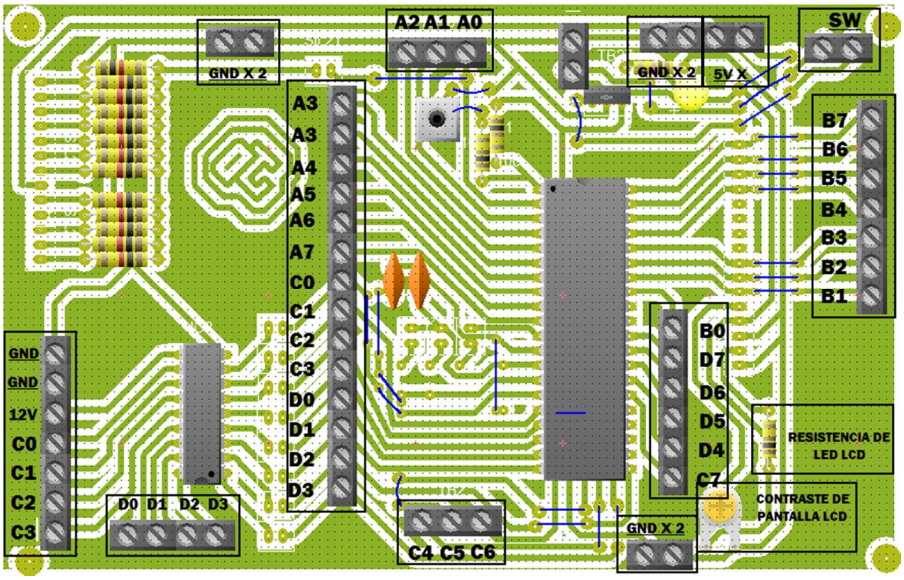
El diagrama PCB (tarjeta de circuito impreso) de esta placa ha sido diseñado en el programa PCB Wizard, que es uno de los programas más livianos y completos para poder empezar a aprender realizar proyectos electrónicos; este diagrama irá adjunto al proyecto de tesis como archivo *.pcb en el disco y también impreso en acetato para poder fotocopiarlo, tomando en cuenta las indicaciones en capítulos posteriores para la realización correcta de la placa.

Figura 7. **Vista normal del circuito**



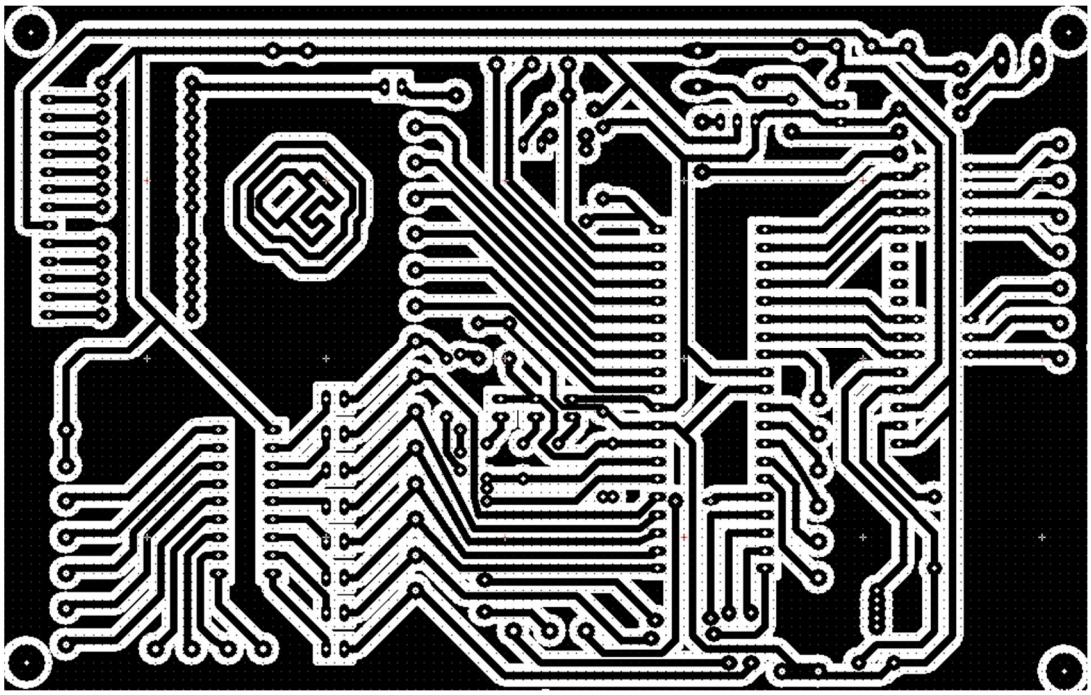
Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 8. **Vista realista del circuito**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 9. **Vista *artwork* del circuito**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

2.2. Energizado de un circuito para un microcontrolador PIC16F877A

Para el energizado del microcontrolador debe de tener en cuenta una serie de dispositivos electrónicos pasivos como resistencias, reguladores de voltaje de 5 voltios, diodos LED que indicará cuando esté apagado o encendido el circuito, botones de presión (*push botton*) y *switch* de dos estados; para la placa de circuito impreso que se presenta en este documento; se implementó el siguiente circuito de energizado debiendo tomar en cuenta que no hay una sola forma de energizar el microcontrolador, pero esa es la que mejor se aplica al circuito, dado a que cumple con las necesidades básicas y necesarias para todo proyecto.

Para realizarlo se utilizan los dispositivos siguientes conectados a las terminales indicadas en el diagrama esquemático de conexión del microcontrolador y los dispositivos pasivos que serán descritos a continuación.

2.3. Cargador de teléfono móvil de 5 V a 6 V

Se utilizará un cargador de los que comúnmente usan los teléfonos móviles, porque estos ya tienen integrado un transformador de 120 VAC a aproximadamente 5 V corriente directa; es necesario que el cargador sea de 5.2 V, a 6 V porque el regulador de voltaje 7805 que se presentará más adelante, consume cierta cantidad de corriente y si se tiene un cargador de 5 V o menos, este entregará 4.9 V o menos voltaje al PIC, y en consecuencia no entrará en funcionamiento porque al ser menor el voltaje, la corriente no es suficiente para el microcontrolador.

Figura 10. **Cargador 5 V**

Fuente: taller propio, Ciudad de Guatemala

2.4. *Switch* interruptor de dos estados

Para interrumpir la energía eléctrica que se suministra al microcontrolador se colocará un *switch* de dos estados que irá conectado en serie de la terminal positiva del cargador, directamente a la terminal de entrada el regulador de voltaje 7805.

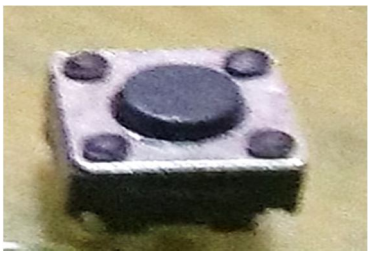
Figura 11. **Switch**

Fuente: taller propio, Ciudad de Guatemala

2.5. Botón Interruptor (*push botton*)

El *push botton* será necesario para conectarlo con la función de restablecer el funcionamiento de PIC (*reset*) que iniciará todos los procesos del microcontrolador, cuando este sea presionado.

Figura 12. **Botón interruptor**

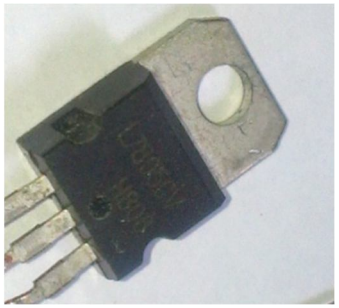


Fuente taller propio, Ciudad de Guatemala

2.6. Regulador de 5 voltios 7805

El regulador de voltaje 7805 ayudará a regular voltaje positivo que suministrará el cargador de teléfono móvil; la forma de utilizar este circuito integrado es que se conecta a la terminal de entrada al cargador ubicada en la patilla 1 del 7805, con la terminal positiva del cargador en común a la tierra del cargador con el pin 2 del 7805, dando como resultado que se suministren 5 V regulados casi perfectos del pin 3 del 7805, este voltaje regulado es el que será inyectado a nuestro microcontrolador.

Figura 13. **Regulador de voltaje**

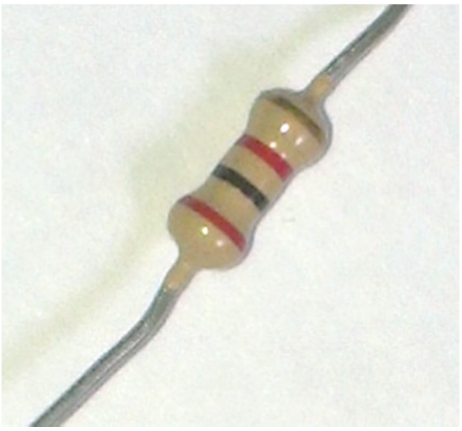


Fuente: taller propio, Ciudad de Guatemala

2.7. Resistencias de 330 Ω, 10 KΩ y 220 Ω

Las resistencias serán necesarias para regular la corriente que se suministra al circuito; una resistencia de 10 KΩ, como se mostrará en el diagrama esquemático, será necesaria para cuando al ser presionado el botón sw1 del diagrama lleve al restablecimiento del PIC (*reset*); se utilizará una resistencia de 330 Ω para regular la corriente del diodo LED y una de 220 Ω para regular la corriente del microcontrolador.

Figura 14. **Resistencias**

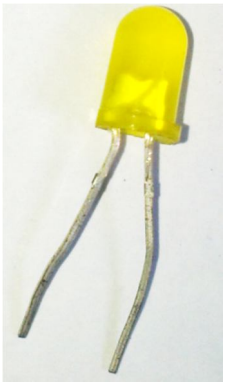


Fuente: taller propio, Ciudad de Guatemala

2.8. Diodo emisor de luz (LED)

El diodo LED tendrá una función tan importante como simple, la cual es indicar cuándo está encendido el circuito, siendo polarizado con una resistencia de 330 Ω en serie al mismo.

Figura 15. **LED**



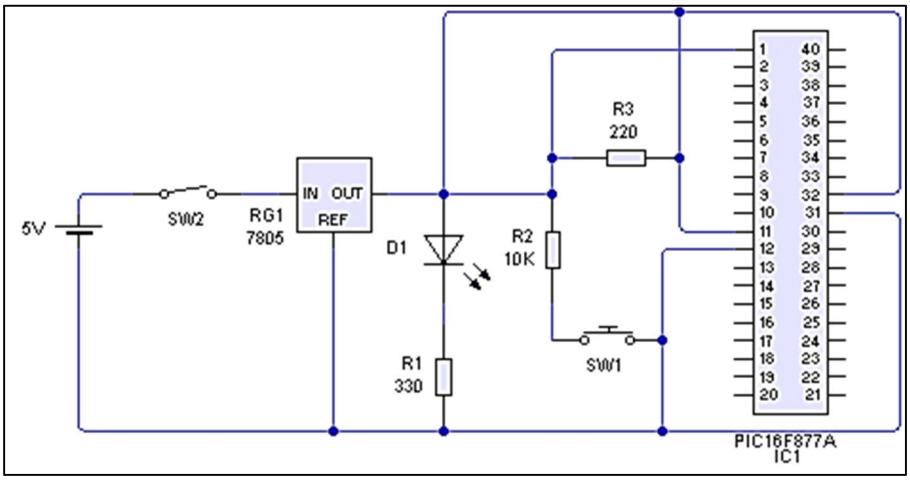
Fuente: taller propio, Ciudad de Guatemala

2.9. Diagrama energizado del PIC16F877A

A continuación se presenta el diagrama esquemático de energizado de la placa, en la cual será necesario disponer los dispositivos tales como el cargador de teléfono móvil que será la fuente de voltaje para el PIC y los demás dispositivos.

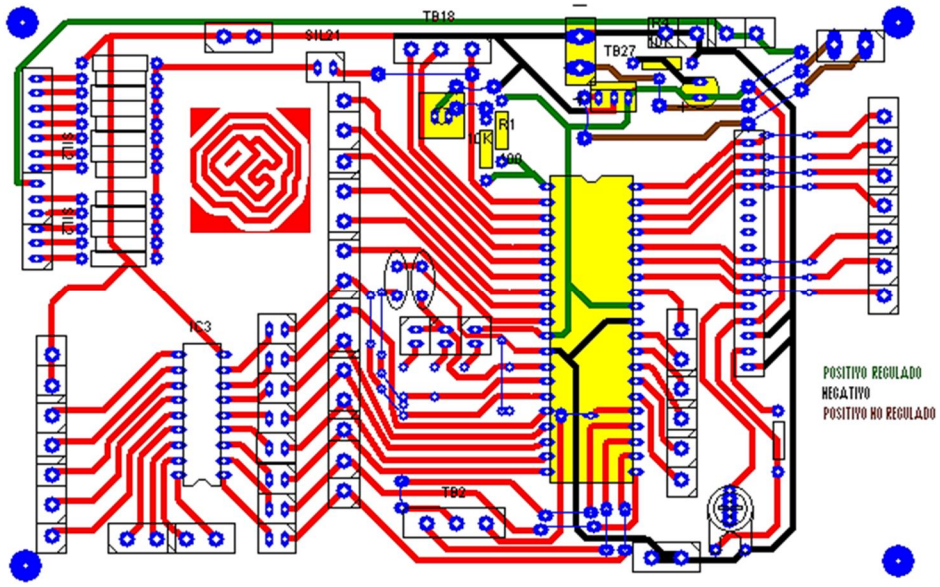
Este voltaje tiene que ser regulado por un circuito integrado 7805 que en su salida suministra 5 V exactos; dicho voltaje será distribuido a través de las resistencias de polarización del PIC y el LED que indica el encendido del circuito; una de las recomendaciones esenciales para poder energizar el circuito adecuadamente es que la fuente o cargador de móvil tiene que tener un voltaje de salida no mayor a los 6 V, pero no menor a 5.12 V, para que el regulador de voltaje consuma solo lo necesario y no se recaliente, o el caso contrario que el voltaje sea muy exacto o menor, causando que la resistencia interna que posee el regulador no permita suministrar los 5 V netos al PIC.

Figura 16. **Energizado de placa**



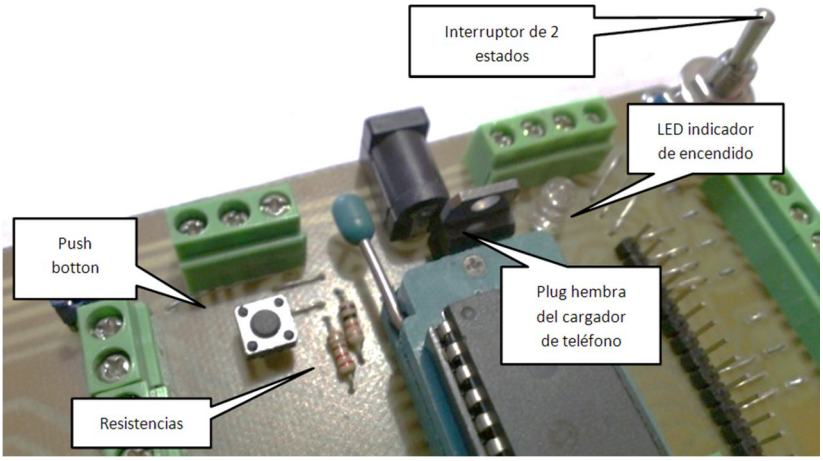
Fuente: elaboración propia, con programa de diseño de circuitos eléctricos Livewire.

Figura 17. **Diagrama energizado de voltaje**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 18. **Placa energización voltaje**



Fuente: taller propio, Ciudad de Guatemala

2.10. Circuito de interconexión de cristal - microcontrolador

Para esta parte del circuito se utilizarán 2 capacitores de 22 microfaradios conectados a tierra y asimismo a los pines número 13 y 14 del microcontrolador; la polarización del cristal y los capacitores es irrelevante porque los capacitores son cerámicos y el cristal no tienen polarización en cualquiera de sus presentaciones y frecuencias; para esta placa se utilizarán cristales de 4, 10 y 20 KHz.

2.11. Dos capacitores cerámicos

Los 2 capacitores de 22 microfaradios serán conectados uno a cada terminal del cristal de 2 patillas y los 2 capacitores deben ser conectados a tierra.

Figura 19. **Capacitores cerámicos**

Fuente: taller propio, Ciudad de Guatemala

2.12. Cristal de cuarzo

Son elementos de material piezoeléctrico que tienen la propiedad de generar oscilaciones al ser polarizados; se utilizarán de 2 y 4 terminales porque en la placa del microcontrolador se dejará la opción de seleccionar qué cristal es conveniente utilizar, ya sea de 4 Mhz, 10 Mhz y 20 Mhz (4 terminales).

Los 2 tipos de cristal tienen una conexión diferente, el de 2 terminales necesita 2 capacitores cerámicos de 22 µF para funcionar, mientras que el de 4 terminales solo debe ser energizado para funcionar, la conexión será visualizada en el diagrama esquemático que se presentará más adelante.

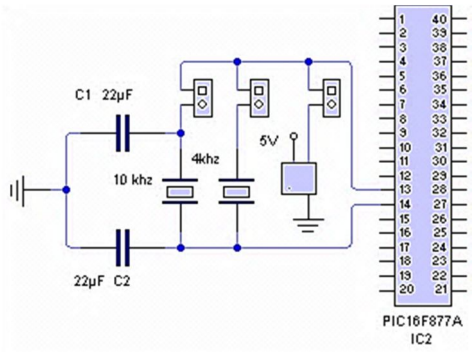
Figura 20. **Cristal de cuarzo**

Fuente: http://www.steren.com.mx/catalogo/. Consulta: 20 de mayo de 2013.

2.13. Diagrama esquemático de cristal de PIC

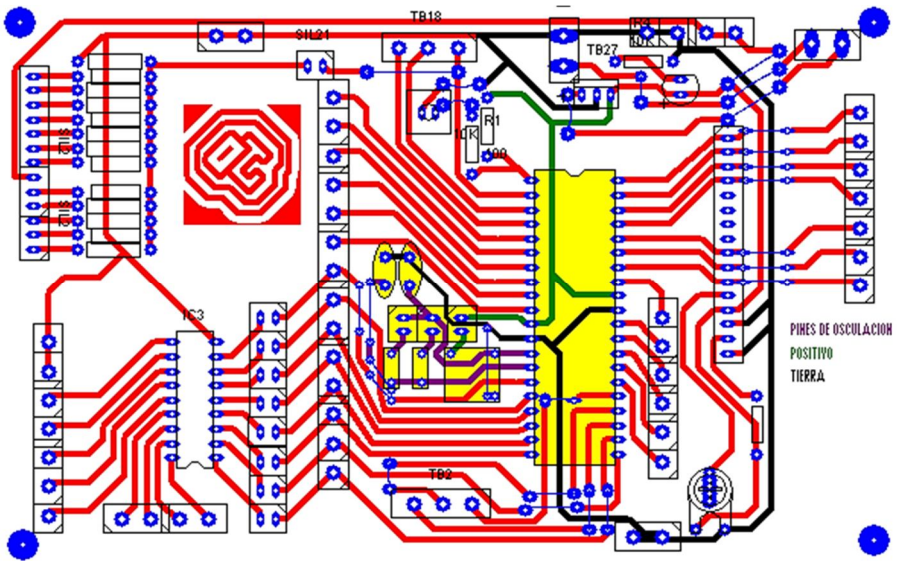
A continuación se describen las variantes de este diagrama.

Figura 21. **Diagrama esquemático de cristal - PIC**



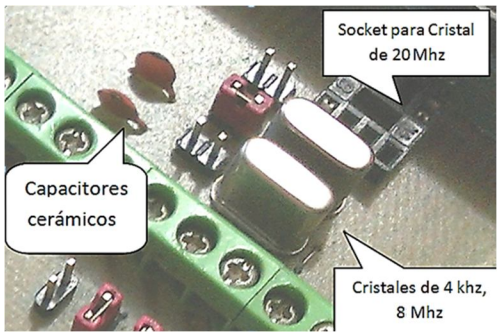
Fuente: elaboración propia, con programa de diseño de circuitos eléctricos Livewire.

Figura 22. **Diagrama PCB de cristal – PIC**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Figura 23. **Ubicación física de los cristales**



Fuente: taller propio, Ciudad de Guatemala

2.14. Circuito de interconexión del microcontrolador, el amplificador de corriente ULN2803 y distintos dispositivos de salida que utilizan altas cantidades de corriente

En la siguiente imagen se muestra el circuito esquemático que será de utilidad para interconectar un circuito integrado de amplificación de corriente ULN2803, que utiliza la configuración de 8 pares Darlington que contiene 18 pines, de los cuales los pines 9 y 10 son para el energizado del CI, pin 9 para tierra y pin 10 para 12 V; soportados por el ULN2803; de esta fuente individual solo se debe conectar en común al controlador de tierra, porque los 12 voltios de la fuente secundaria serán para polarizar distintos dispositivos electromecánicos que necesitan cantidades significativamente mayores de corriente que las proporcionadas por el microcontrolador.

Estos dispositivos son tales como motores DC, solenoides, relevadores y motores de paso (*steppers*), los cuales se conectan al ULN2803 a los pines 11 al 18 que serán como tierra virtual y el común de cada dispositivo electromecánico, será la terminal positiva de la fuente de 12 V.

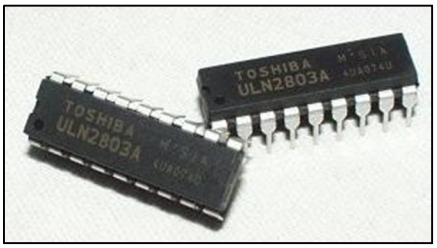
Los pines de entrada del 1 al 8 se deben interconectar al microcontrolador que emitirá la señal que accionará el ULN2803.

En total el ULN2803 contiene 8 *buffers* de corriente interconectados en línea, por ejemplo: pin 1 con pin 18, pin 2 con pin 17 y así sucesivamente, hasta el pin 8 con pin 11; los cuales se pueden conectar en paralelo para sumar más corriente para dispositivos que lo requieran (es decir los pines 18 y 17 en común a un motor DC que exija más corriente para accionarlo; asimismo, se deben encender al mismo tiempo los pines 1 y 2, para activarlos con el microcontrolador)

2.15. Amplificador de corriente ULN2803 (circuito integrado)

El circuito integrado ULN2803 es un arreglo de ocho pares Darlington; se utiliza esta configuración para obtener una amplificación de corriente y lograr energizar elementos electromecánicos como relevadores, solenoides, motores DC y motores *steppers* para poder llevar a cabo acciones físicas programadas a través del microcontrolador; a continuación se tendrá una imagen del circuito integrado y la hoja de datos básica para su conexión**.**

Figura 24. **Circuito integrado ULN2803**

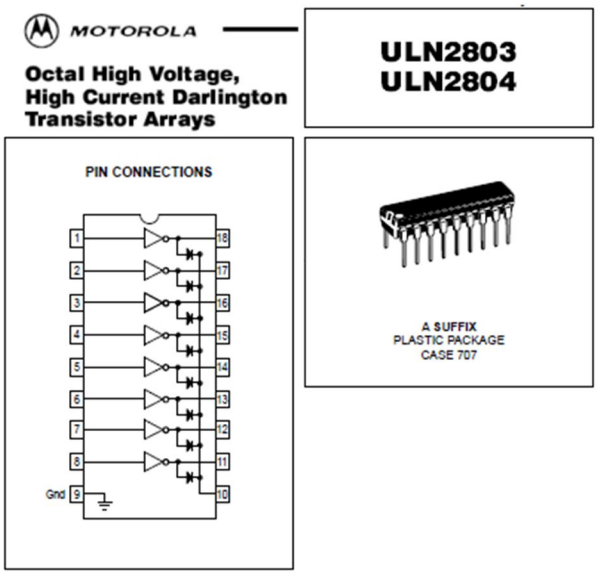


Fuente: taller propio, Ciudad de Guatemala

2.16. Circuito esquemático de interconexión del microcontrolador con el C.I. ULN2803

En el diagrama de la siguiente figura se presentan en su interior los 8 *buffers* negados que posee este circuito integrado; estos indican que la amplificación de la corriente tiene una dirección que transcurre al contrario de la indicada; lo quiere decir que se tomarán estas terminales como “tierras” o terminal de menor potencial, y los dispositivos tienen que ir interconectados en común a los 12 V de la fuente de voltaje para que puedan funcionar; por ejemplo, cuando se aplique un 1 lógico a la terminal 1 del ULN2803, inmediatamente se cierra el circuito desde el dispositivo que se encuentra conectado a la terminal 18 a la fuente de voltaje alimentación adicional de 12 V , provocando que se accione.

Figura 25. **ULN2803A motorola**

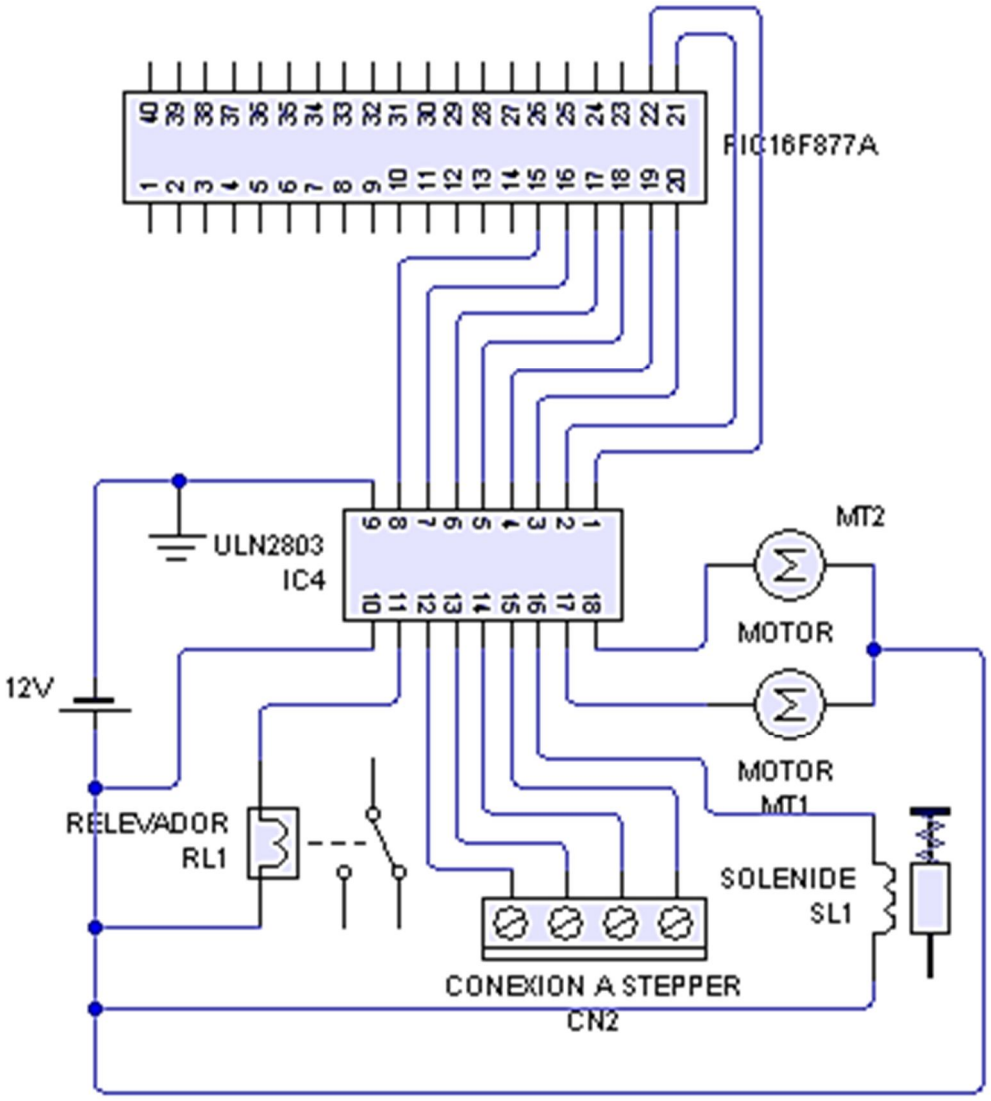


Fuente: http://pdf.datasheetcatalog.com/datasheets/90/366828_DS.pdf.

Consulta: 11 de mayo 2013.

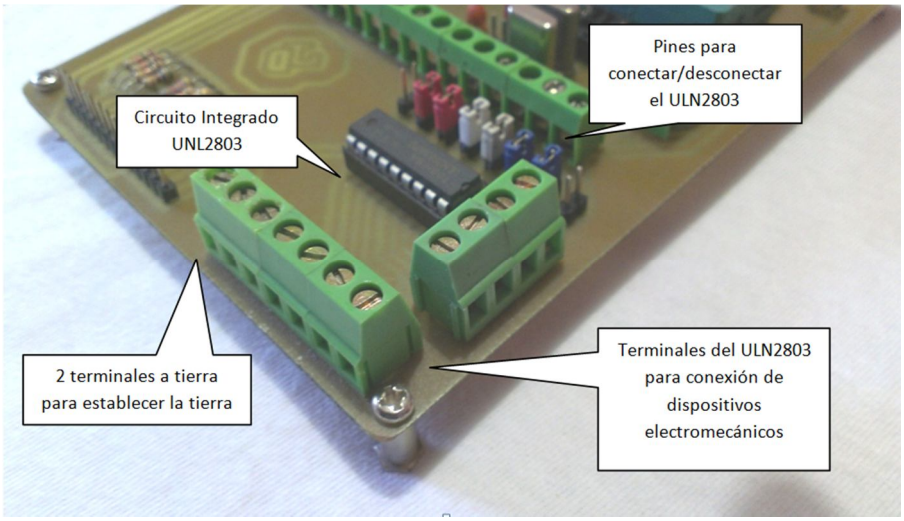
En la siguiente figura se visualiza la interconexión de distintos dispositivos electromecánicos como motores DC, motores de paso, relés y solenoides que son los dispositivos que usualmente son interconectados al ULN2803.

Figura 26. **Diagrama esquemático ULN2803 - PIC**



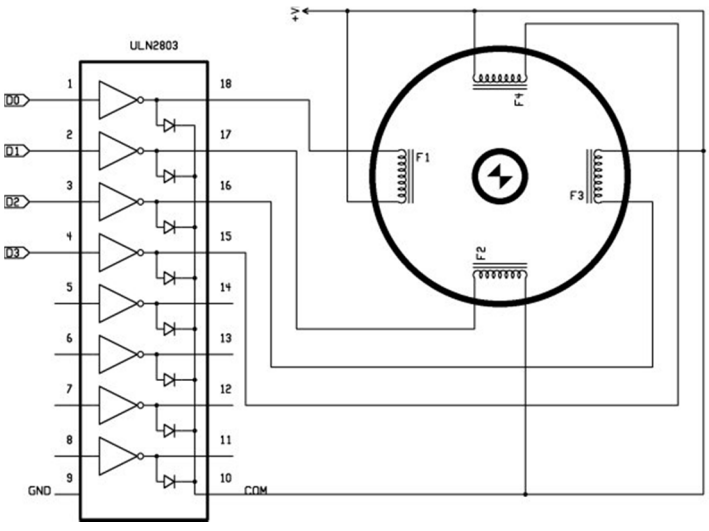
Fuente: elaboración propia, con programa de diseño de circuitos eléctricos Livewire.

Figura 27. **Ubicación física del ULN2803 y terminales**



Fuente: taller propio, Ciudad de Guatemala

Figura 28. **Diagramas de motores *stepper* - ULN2803**



Fuente: http://www.datasheetcatalog.com/datasheets_pdf/U/L/N/2/ULN2803.shtml, Consulta: 11 de marzo 2013.

2.17. Circuito de Interconexión de teclado matricial al pin A0 del PIC

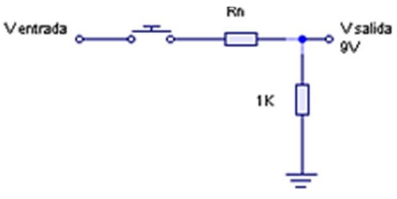
El teclado matricial será conectado al PIC de una forma en la que optimizará el uso de las patillas del microcontrolador, específicamente conectado a la patilla A0 del PIC; la idea en sí es crear una forma de utilizar los botones del teclado como si fueran botones de presión comunes (*push botton*), pero con la diferencia que utilizarán todos los botones conectados entre sí hacia la patilla A0 del PIC, en común a todos los botones.

Entonces ¿cómo hará para que el PIC reconozca cuál botón está siendo presionado?; para esto se utilizará el concepto de diferencia de voltaje, porque lo que se hará es crear una red de resistencias por cada botón, como se muestra en la siguiente imagen, de modo que todas harán una diferencia de voltaje con la resistencia Rn para el ejemplo de la imagen, teniendo las siguientes ecuaciones para la diferencia de voltaje siendo el “V salida” el que llega al PIC y es reconocido por la configuración analógica / digital del PIC; este voltaje es el que diferenciará cada botón del teclado, siendo dependiente de Rn, el cual se variará para que dé en total 10 voltajes distintos de 1 V a 5 V de rango y “V entrada” son los 5 V del regulador 7805 que energiza el PIC.

Figura 29. **Diferencia de potencia de entrada a puerto A0**

$$\frac{V_{entrada} - V_{salida}}{Rn} = \frac{V_{salida}}{1k}$$

$$\frac{(V_{entrada} - V_{salida})\ 1k}{Rn} = V_{salida}$$



Fuente: elaboración propia.

2.18. Comparación de valores de resistencia Rn versus bits de entrada

Se elaborará una tabla de comparación de los valores de resistencia Rn con las resistencias asignadas a cada botón y el valor en bits, que se reconoce el microcontrolador PIC.

Las resistencias tienen que generar diferentes niveles de voltaje en valores que van desde 0 V hasta 5 V, porque el analógico digital de PIC leerá cada uno de estos valores y les asignará un valor en bits que comprende desde 0 hasta 1024 bits; estos datos son los que se manipularán en la programación para cada uno de los botones del teclado matricial, ayudando así a optimizar el uso de las patillas del PIC, siendo únicamente una utilizada y el puerto A0, el cual se definirá posteriormente explicando la programación del teclado.

Tabla I. **Tecla presionada versus resistencia y bits de muestra**

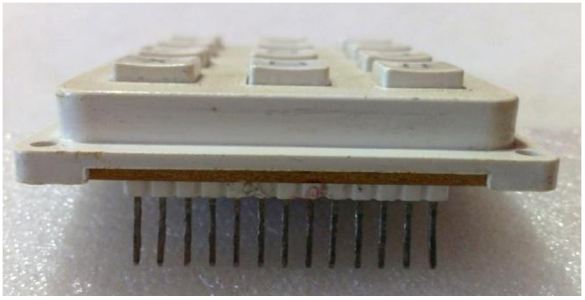
Botón del teclado matricial	Resistencia asignada un Ω	Valor en bits definido por el PIC por cada valor de voltaje recibido
1	10 k	92
2	4.3 k	189
3	2.2 k	310
4	1.8 k	356
5	1.2 k	449
6	800	552
7	560	627
8	330	750
9	220	817
B	100	914
#	51	963
--	1 kΩ resistencia en común del divisor-- de voltaje	

Fuente: elaboración propia.

Figura 30. **Teclado matricial utilizado para el proyecto**

Fuente: taller propio, Ciudad de Guatemala

Figura 31. **Pines de teclado matricial**



Fuente: taller propio, Ciudad de Guatemala

Orden de la asignación de los pines de los botones a las resistencias, según la posición normal del teclado visto de frente, como en la figura presentada.

Tabla II. **Resistencia por patilla del teclado**

1	4	7	x	2	5	8	Fuente	B	6	3	9	#
10k Ω	1.8k Ω	560 Ω	--	4.3 kΩ	1.2 kΩ	330 Ω	5V	100 Ω	2.2 kΩ	2.2 kΩ	220 Ω	51 Ω

Fuente: elaboración propia.

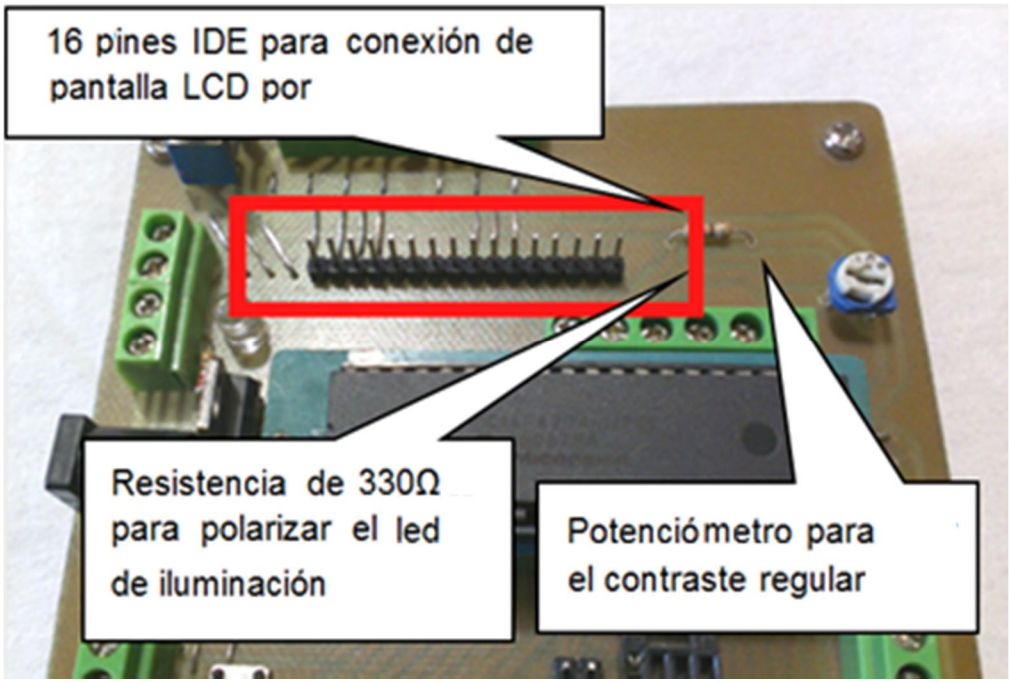
2.19. Circuito de interconexión de la pantalla LCD 2x16 con el microcontrolador

La interconexión de la pantalla LCD 2x16 con el microcontrolador debe utilizar el circuito esquemático que se presenta en la siguiente figura; se utilizará además la conexión de los puertos del PIC, para los cuales será necesario apartar el puerto B desde los pines 34 al 40 correspondientes a los pines B1 al B7; también se conectará una resistencia de 330 Ω al pin 15 y 13 (como se muestra en el circuito) de la pantalla LCD.

Esta resistencia se utilizará para energizar el diodo LED que ilumina el texto en la pantalla; también se utilizará un potenciómetro de 1 kΩ para regular el contraste del texto en la pantalla que irá conectado del pin número 11 al central del potenciómetro (para esta placa se utilizará un potenciómetro de superficie para adaptarlo a la placa más eficientemente); consecuentemente, la resistencia, el potenciómetro y el pin 13 de la pantalla van conectados a los 5 V que salen del regulador de voltaje 7805 de la placa y al final los pines 14 y 16 y el tercer pin del potenciómetro se conectan a tierra común.

En la siguiente figura se presenta la ubicación de los pines para la pantalla LCD que se encuentran sobre la placa entrenadora, y la resistencia que polarizará el LED interior de la pantalla.

Figura 32. **Pines de conexión de pantalla LCD**

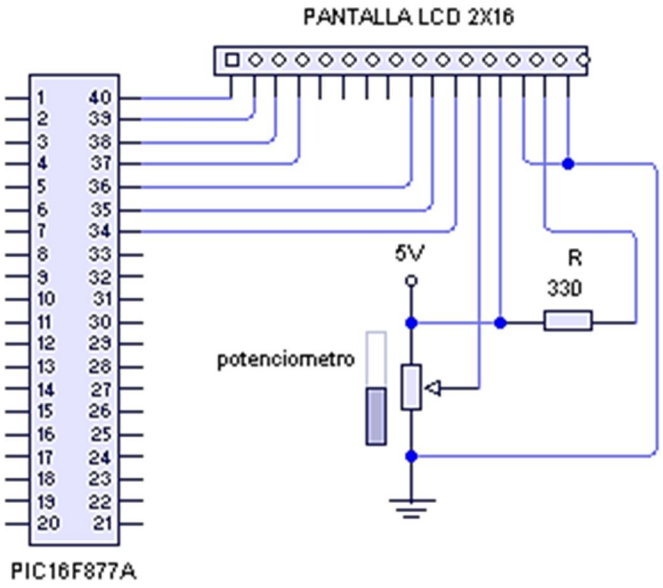


Fuente: taller propio, Ciudad de Guatemala

Figura 33. **Resistencia del LED y potenciómetro de contraste**

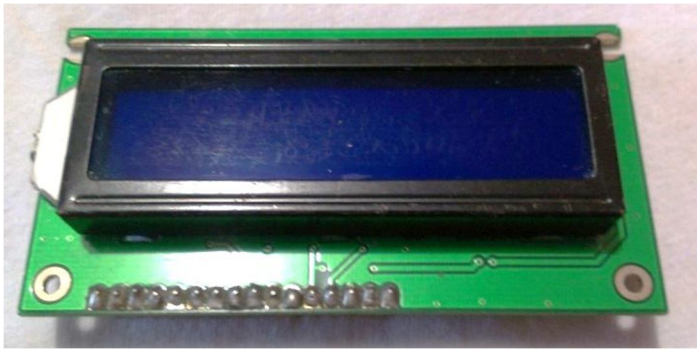
Fuente: taller propio, Ciudad de Guatemala

Figura 34. **Conexión LCD, resistencia y potenciómetro**



Fuente: elaboración propia, con programa de diseño de circuitos eléctricos Livewire.

Figura 35. **Pantalla LCD 2x16 de color azul**

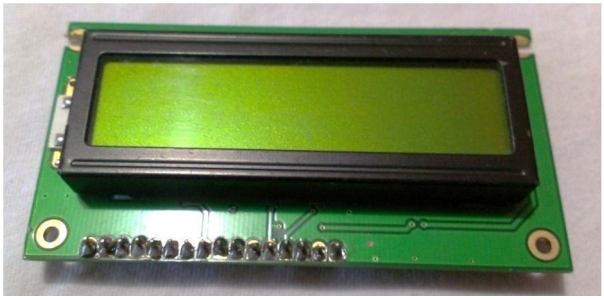


Fuente: taller propio, Ciudad de Guatemala.

2.20. Pantalla LCD 2x16 color verde

Se debe poner en corto la resistencia de 330 Ω, para que se visualice la luz de la pantalla; de lo contrario será muy difícil ver los caracteres en la misma.

Figura 36. **Pantalla LCD de color verde**



Fuente: taller propio, Ciudad de Guatemala

Figura 37. **Conexión LCD, resistencia en corto**

Fuente: taller propio, Ciudad de Guatemala

3. PROGRAMACIÓN DE PIC EN PROGRAMA PIC SIMULATOR IDE

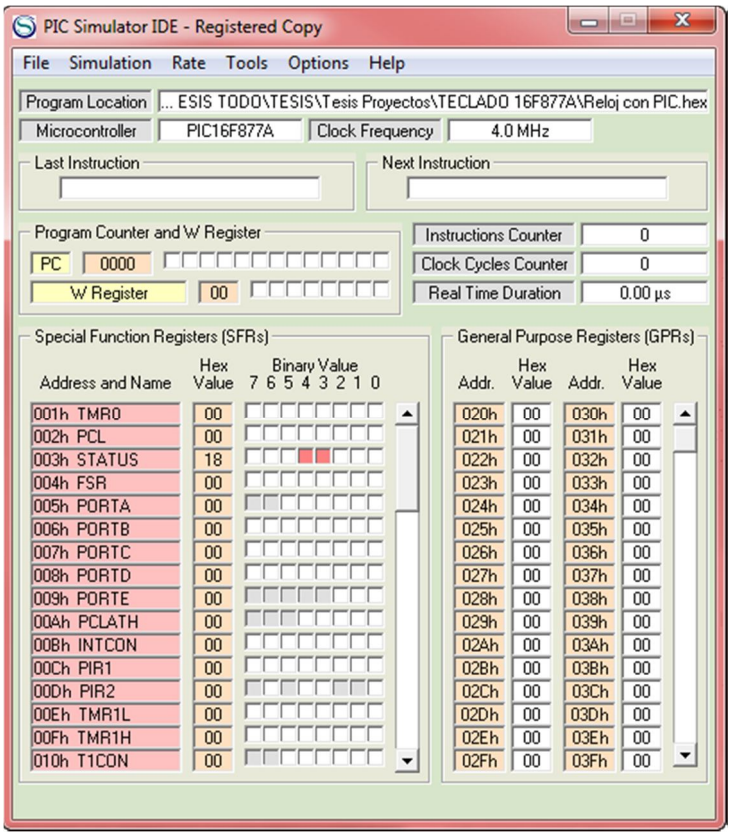
PIC Simulator IDE es un programa creado por Vladimir Soso de la empresa Oshonsoft; esta aplicación es considerada como uno de los programas más didácticos, amigables y fácil de usar para el aficionado y estudiante que está incursionando en la programación y desarrollo de proyectos con microcontroladores PIC; la placa de microcontroladores PIC propuesta para este documento ha sido elaborada para poder realizar nuestras pruebas sobre este simulador; la distribución de los puertos por defecto están de acuerdo con este programa, asimismo la distribución de los pines en el puerto B para controlar la pantalla LCD también han sido adecuados para este fin.

El simulador consta de herramientas para la prueba de periféricos que pueden ser interconectados al PIC como pantallas LCD, teclados matriciales, motores *stepper*, bancos de LEDs, *displays* de 7 segmentos, y un editor de texto que soporta lenguajes de programación como PIC Basic

3.1. Descripción de la pantalla principal

En la figura siguiente se visualiza la ventana inicial de PIC Simulator IDE; en ella se ven las aplicaciones principales para la programación del PIC, contando con una barra de menú, selector de microcontrolador y de frecuencia de reloj, visualización de las instrucciones del programa, compilado de lenguaje hexadecimal paso a paso, de donde se está ejecutando el programa.

Figura 38. **Pantalla principal PIC Simulator IDE**



Fuente: elaboración propia.

3.2. Barra de menú

Esta sección del programa contiene los menús para poder manipular el archivo raíz del programa, herramientas de simulación para dar inicio, parar o finalizar la simulación, selector de la velocidad a la que se ejecuta el programa, herramientas tales como visualización del microcontrolador, programador y demás periféricos del PIC, opciones para editar las propiedades del programa y del microcontrolador y un menú de distintos manuales de ayuda para el usuario.

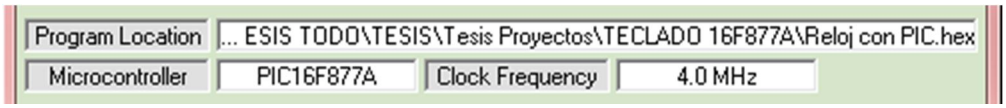
Figura 39. **Menús del programa**

Fuente: elaboración propia.

3.3. Subsecciones de la pantalla principal

En la casilla "*program location*" se visualiza la ubicación actual del programa que está cargado al simulador, dando doble *click* en la ubicación del programa; este se actualiza y verifica el programa en caso de que hayan habido modificaciones, en la casilla "*microcontroller*" se tiene el modelo del PIC sobre el cual se está programando y en la "*clock frecuency*" está la frecuencia del cristal, con la que se está trabajando el PIC ya grabado sobre la placa física sobre la que irá montado el proyecto.

Figura 40. **Sección de pantalla principal del programa**



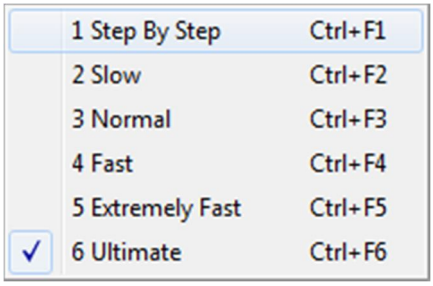
Fuente: elaboración propia.

3.4. Menú *rate*

En la barra de menú está el menú "*rate*" en el cual se tiene las velocidades sobre las que se compilará el simulador para poder ver la instrucción que se está ejecutando y a diferentes velocidades, que van desde

ver la simulación instrucción por instrucción con la opción "*step by step*" hasta la velocidad más alta que es "*Ultimate*"

Figura 41. **Menú *rate***



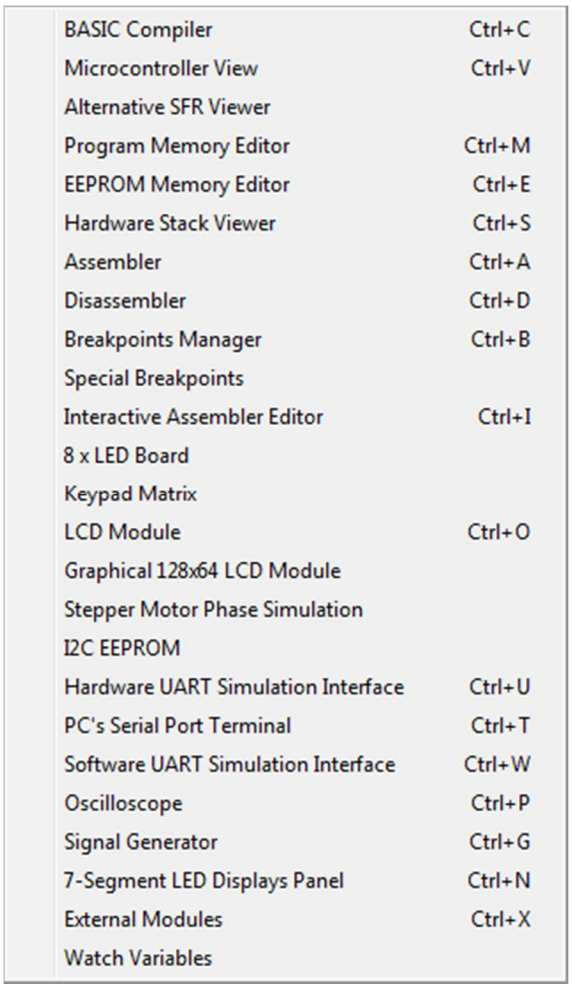
Fuente: elaboración propia.

3.5. Barra de herramientas

En el menú *tools* se encuentran distintas herramientas que servirán de apoyo para poder visualizar la compilación en tiempo real, la simulación por completo del programa a compilar.

La siguiente figura presenta el menú de la sección *tools;* aquí se pueden encontrar herramientas tales como ventanas secundarias que servirán para poder realizar simulaciones más precisas y gráficas sobre los dispositivos que se pueden interconectar al PIC, tales como osciloscopios, módulo de pantalla LCD, visor de variables, compilador Basic, vista de los puertos del microcontrolador, set de 8 LEDs, teclado matricial, entre otras herramientas.

Figura 42. **Menú *tools* (herramientas)**

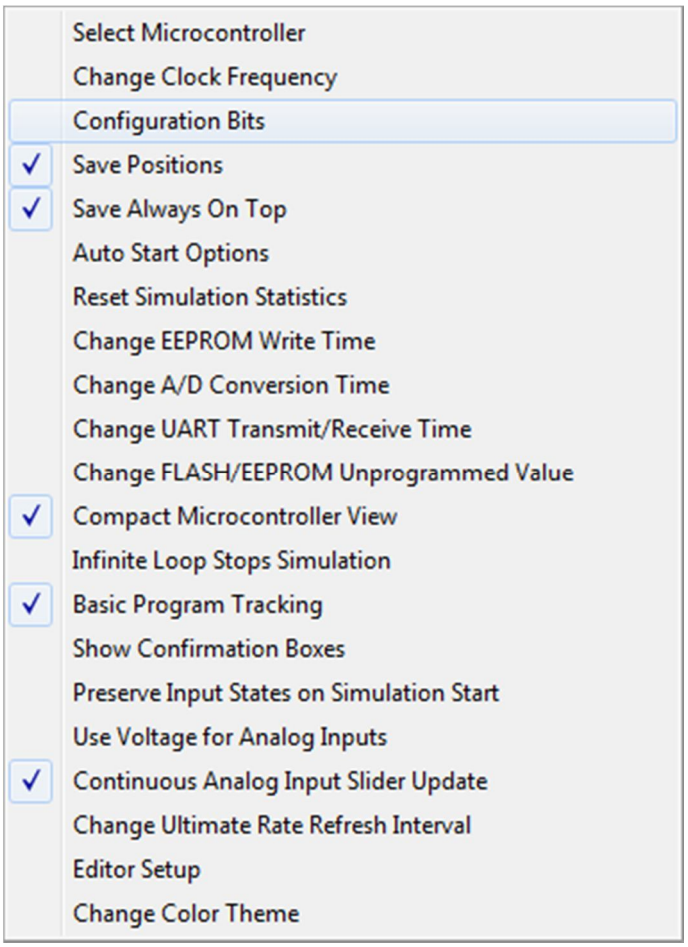


Fuente: elaboración propia.

3.6. Barra de opciones

En el menú opciones se podrán editar las propiedades del simulador como: seleccionar el microcontrolador, dar *reset* los datos de la simulación, cambiar el tiempo de escritura de la EEPROM, las propiedades del conversor analógico/digital y UART e incluso cambiar el color del tema.

Figura 43. **Barra opciones**



Fuente: elaboración propia.

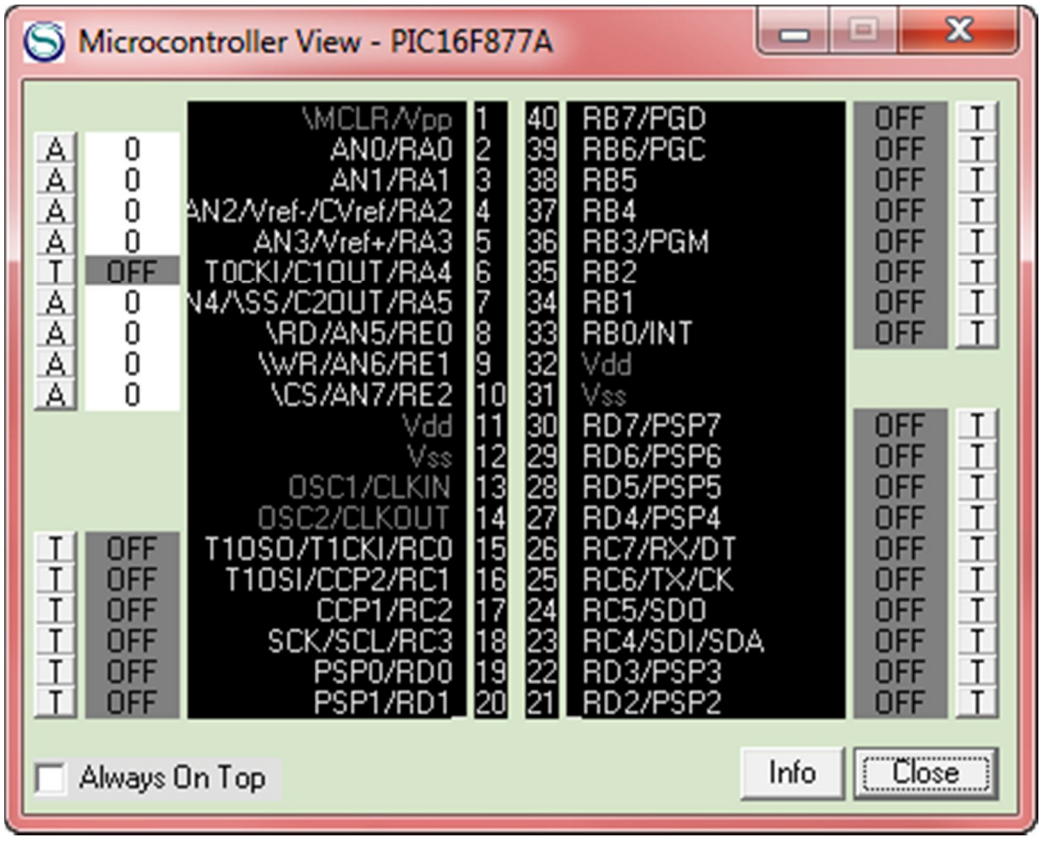
3.7. Herramientas más útiles del simulador

En esta sección se presentarán las distintas ventanas que posee el simulador, que son herramientas de ayuda para poder realizar pruebas de la programación en tiempo real.

3.7.1. Microcontroller view

La vista del microcontrolador no es útil para poder ver cuando un puerto es activado o desactivado en el PIC, cambiando de color verde a gris, dependiendo del estado en el que se encuentre el puerto y se pueden utilizar los botones de cada puerto para activarlos, cuando se configuran como entrada estos puertos o variar la entrada analógica de entrada en los puertos A del PIC.

Figura 44. **Microcontroller view**

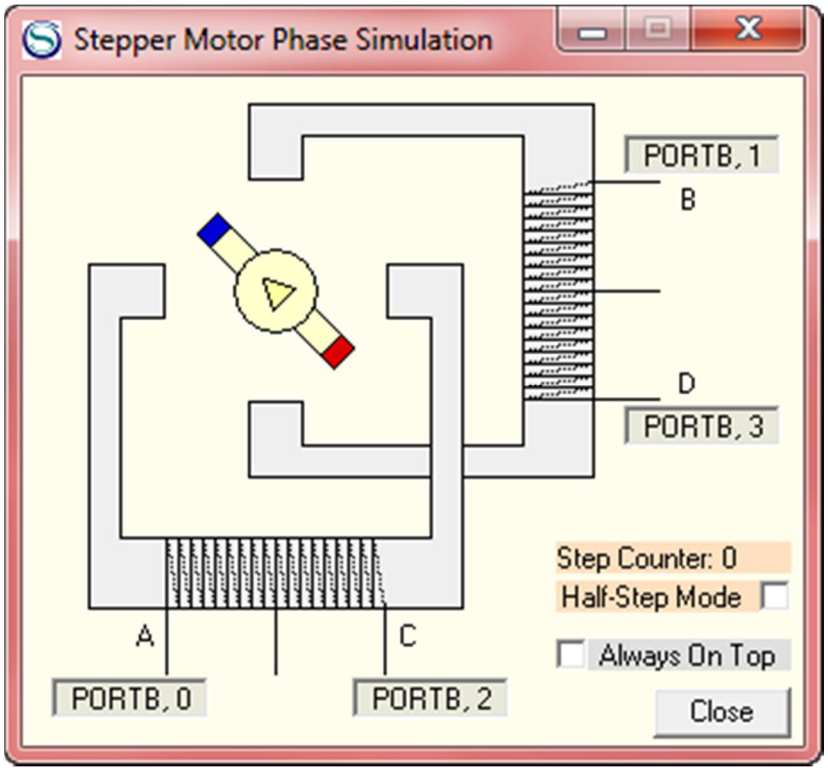


Fuente: elaboración propia.

3.7.2. Stepper motor phase simulation

Esta herramienta de simulación se utiliza para programar motores *stepper* unipolares polarizando 4 puertos de par, en par consecutivamente para provocar la rotación del eje del motor; en las terminales de los devanados de los solenoides graficados hay etiquetas editables donde se selecciona el puerto al que se quiere asignar el *motor stepper;* esta ventana presentará gráficamente la simulación del giro del rotor, dependiendo de la polarización generada por el programa del PIC.

Figura 45. **Stepper motor phase simulation**

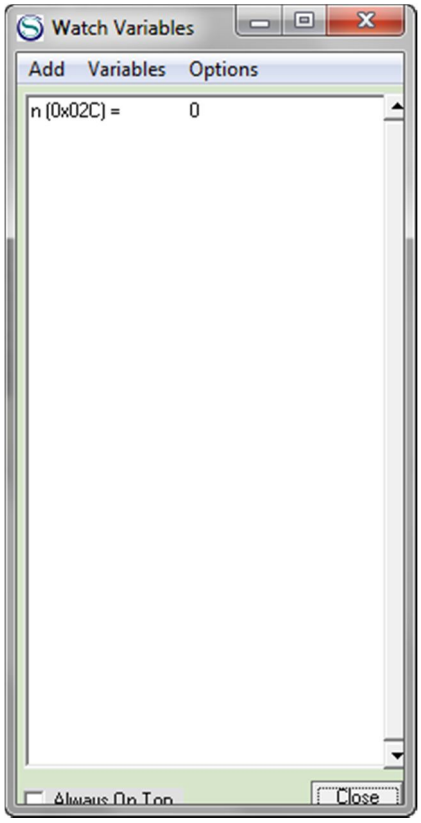


Fuente: elaboración propia.

3.7.3. Watch variables

En esta ventana aparecen las variables declaradas y el valor que tienen tanto inicialmente como en todo su proceso; en su defecto es recomendable declarar las variables y asignarles un valor igual a cero para que no haya conflictos en las mismas y que no adquieran un valor aleatorio que pueda afectar el proceso del programa compilado en el PIC; estos valores se verán reflejados en la sección blanca de la ventana donde se encuentran las variables seguidas del signo igual su valor actual.

Figura 46. **Watch variables**

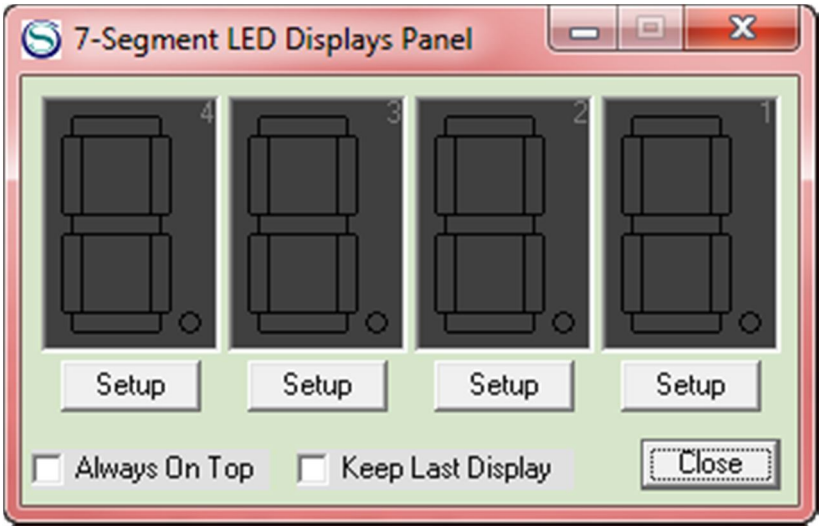


Fuente: elaboración propia.

3.7.4. 7- segment LED displays panel

Se presentan 4 *displays* de 7 segmentos, los cuales pueden ser configurados para los puertos del PIC, activándolos como si fueran leds, encendiendo consecutivamente los leds adecuados para que aparezca la representación del número.

Figura 47. **Segment LED displays panel**



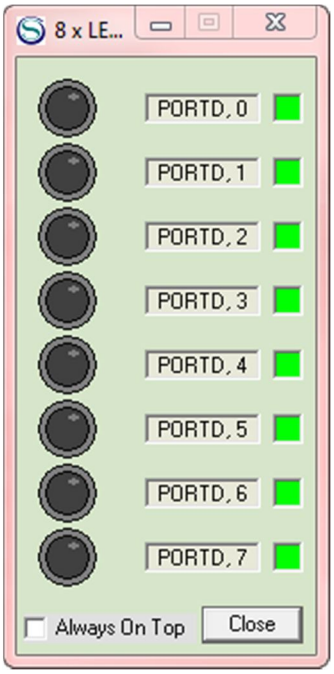
Fuente: elaboración propia.

3.7.5. 8 x LED board

La ventana siguiente representa 8 leds que pueden se conectados a cualquier puerto de PIC, con la opción de editar los colores de los leds y el puerto al que están siendo conectados; esta herramienta es muy útil al visualizar si los puertos que se necesita activar se cambian de acuerdo con lo preestablecido en el programa que realiza el estudiante.

Un problema muy común es que no se declaran adecuadamente los puertos como salida (TRISB.0 = 0 que quiere decir que se está declarando el puerto B0 como salida del PIC).

Figura 48. **8 x LED Board**

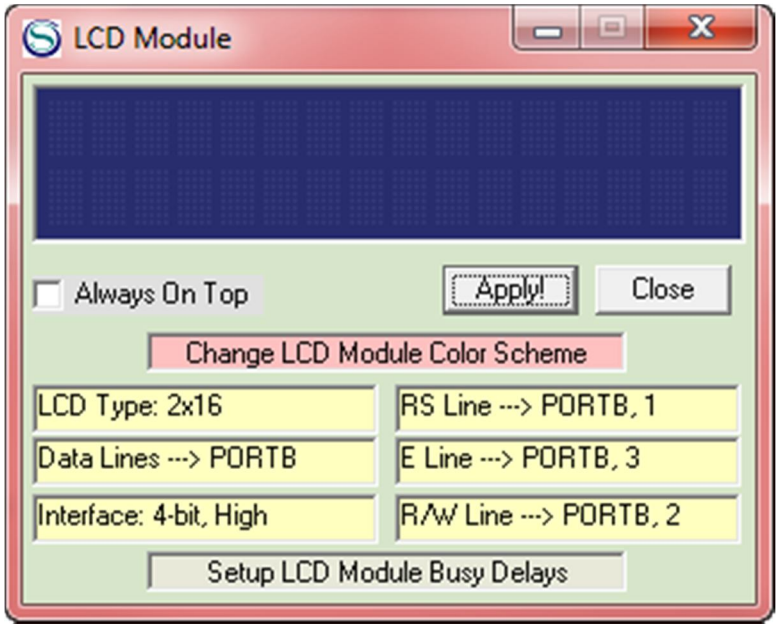


Fuente: elaboración propia.

3.7.5.1. LCD module

El módulo de pantalla LCD del simulador viene con los puertos configurados en el puerto B del PIC igual que la placa entrenadora diseñada para este trabajo de graduación. Para evitar que sea necesaria la reconfiguración de los puertos, se recomienda utilizar las configuraciones predeterminadas por el compilador, que se encuentra configurado en el puerto B del PIC; esto si usa el diseño de placa proporcionado.

Figura 49. **LCD Module**

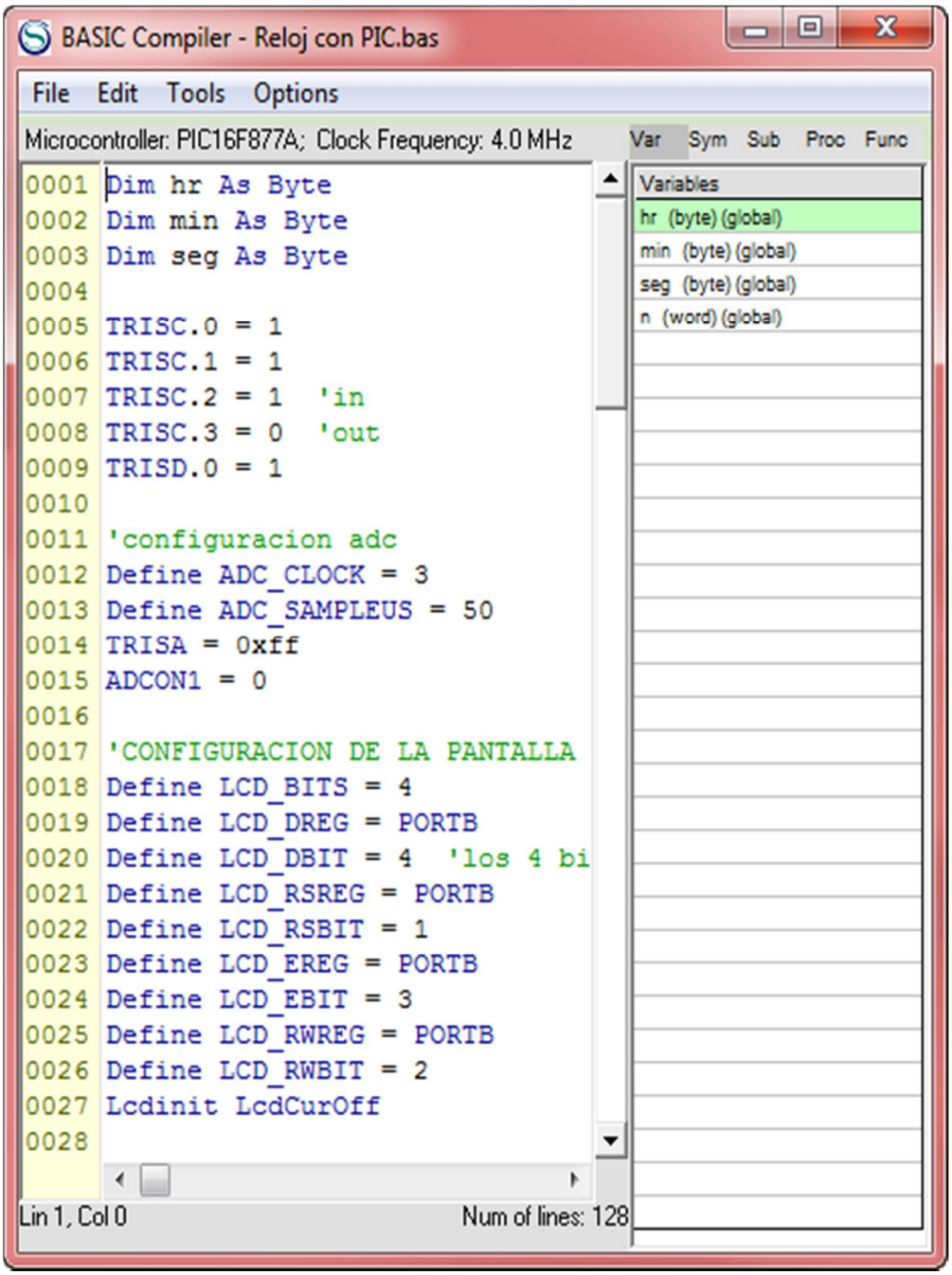


Fuente: elaboración propia.

3.7.5.2. Basic Compiler

La ventana *Basic Compiler* es aquella en la cual se puede desarrollar el programa con las instrucciones que el microcontrolador compilará en el proyecto; en el lado derecho de la ventana se tiene una sección que proporciona el valor actual que tienen las variables declaradas, así como los procesos dados a conocer en el programa; el formato que se puede ver en el *Basic Compiler* brinda una forma más sencilla de visualizar el programa, resaltando en azul las palabras que están reservadas por el compilador; en negro, las palabras escritas por el programador y en verde, anteponiendo un apóstrofe, aparecerá en color verde el texto que se quiera colocar como comentario del compilador.

Figura 50. **Compilador *basic***



Fuente: elaboración propia.

3.8. Instrucciones para la programación en PIC Simulator IDE

A continuación se presentan una breve librería de código de programación, conceptos, palabras reservadas e instrucciones para poder programar en el compilador de lenguaje PIC basic.

3.8.1. PIC basic

Para empezar a entender el funcionamiento y comportamiento de los microcontroladores con la ayuda de PIC Simulator IDE, que ofrece la herramienta PIC Basic, que es un compilador enfocado en el lenguaje Basic muy flexible, didáctico y de sintaxis sencilla, debe conocerse cómo se programa un microcontrolador.

Primero se verán las sintaxis básicas para poder realizar casi cualquier proyecto de electrónica y posteriormente se verán ejemplos de programación explicados detalladamente, que se podrán poner en práctica fácilmente, una vez se tengan todas las herramientas necesarias, de las cuales la principal es la placa entrenadora en la que se enfoca este documento, que fácilmente se adapta a la tarea de simulación y prueba de los PIc16F877A en este simulador específico.

3.8.2. Variables

En el lenguaje Basic así como en cualquier otro tipo de lenguaje de programación, son necesarias las variables que al igual que en álgebra son como recipientes que pueden adoptar cualquier tipo de valor, al que le sea asignado.

Estos valores pueden ser usualmente números, letras, palabras y el contenido de estas variables puede ser modificado cuantas veces lo necesite el programa en el cual han sido declaradas.

A continuación se describen los cuatro tipos de variables que se están disponibles y la sintaxis que soporta el simulador PIC Simulator IDE; las variables tienen que ser declaradas al inicio del programa para que sea reservado un espacio suficiente de memoria para cada una de ellas a dentro del microcontrolador.

3.8.2.1. Tipos de variables

Las variables que se van a utilizar en el programa tienen que ser seleccionadas dependiendo del uso que se les quiera dar; por ejemplo en ocasiones cuando solo se necesita definir si una variable es 1 o 0, que quiere decir que si esa variable está encendida o apagada, simplemente se declara como tipo bit; si se necesita una variable que contenga más información simplemente se puede incrementar su capacidad, declarándola como tipo byte hasta tipo Long, que logra almacenar hasta 4 bytes; a continuación se presentan los 4 tipos de variables que se pueden utilizar con los microcontroladores PIC:

- Long: almacena 4 bytes de longitud y números enteros de 0 a 4,294,967,295.

- Word: almacena 2 bytes de longitud y números enteros de 0 a 65,535.

- Byte: almacena 1 byte de longitud y números enteros de 0 a 255.

- Bite: almacena 1 bit de longitud ya sea 0 o 1.

DIM: esta instrucción es necesaria para declarar la variable para que se pueda utilizar. Ejemplos:

Figura 51. **Definición de variables**

```
Dim contador As Bit
Dim motor As Byte
Dim elevador As Byte
Dim calculador As Long
```

Fuente: elaboración propia.

3.8.3. Puertos

Los puertos del microcontrolador pueden ser activados para que sean puertos de entrada o de salida, dependiendo de la necesidad del programa; el comportamiento de los puertos son como las variables tipo byte que pueden analizar valores de 0 a 225 números enteros, lo cual es muy útil cuando se utiliza el puerto analógico/digital del PIC configurado como puerto de entrada, o simplemente también pueden ser configurados para presentar valores de entrada o de salida 1 o 0 (encendido o apagado).

3.8.3.1. Declaración de los puertos

Al declarar los puertos del PIC como puertos de entrada se utiliza el comando “TRIS” seguido del inciso del puerto que se quiere declarar; el signo igual y 1 si se utilizará como puerto de entrada o 0 si se usará como puerto de salida.

Un consejo para poder recordar cuando se usa el 0 o el 1 es tomar la analogía de que el número 0 es la inicial de la palabra *out* (salida) y el número 1 como la inicial de palabra *In* (entrada); esto para poder recordarlo más facilmente; por ejemplo se puede declarar TRISA.0 = 1, esto quiere decir que se toma el puerto A0 como puerto de entrada.

Otro ejemplo sería si se declara TRISC = %0000000 se está diciendo que el puerto C está siendo tomado por completo y además como puertos de salida del PIC; a continuación más ejemplos en la siguiente figura.

Figura 52. **Declaración de los puertos**

```
'Declaramos el puerto a número 1 como salida (out = 0).
TRISA.1 = 0
'Declaramos el puerto B número 2 como entrada (in = 1).
TRISB.2 = 1

'En binario:
'Declaramos el puerto E completo como entrada.
TRISE = %11111111

'Declaramos el puerto D completo como salida.
TRISD = %00000000

'Declaramos e puerto C mitad entrada y mitad salida.
TRISC = %11110000

'Duando usamos la forma TRISC = %00000000 estamos
'diciendo que TRISC = %7 6 5 4 3 2 1 0, cada uno
'de los números es un puerto del grupo C del PIC16f877A
'asignar un estado de encendido/apagadoa los puertos del pic.

'Declaramos el puerto A número 1 como apagado = 0V.
PORTA.1 = 0

'Declaramos el puerto B número 2 como encendido aprox. 5V.
PORTB.2 = 1

'En binario;
'Declaramos el puerto E completo como encendido.
PORTE = %11111111

'Declaramos el puerto D completo como apagado.
PORTD = %00000000

'Declaramos e puerto C mitad encendido y mitad apagado.
PORTC = %11110000
```

Fuente: elaboración propia.

3.8.4. Symbol

Una forma de simplificar la programación es el uso de nombres simbólicos mediante el uso de la herramienta Symbol, que funciona almacenando un segmento de código en una cadena que al momento de que el programa sea compilado, PIC Simulator IDE busca y remplaza esos símbolos con el código asignado y después convierte dicho código a hexadecimal, que es almacenado en el microcontrolador. Ejemplo:

Figura 53. ***Symbol***

```
PORTC.1 = 1

'En el puerto C número 1 esta conectado el motor
'DC que se activa al aparecer este 'código, para
'simplificarlo usamos el Símbolo MOTOR1 de la.
'siguiente manera:

simbol motor1 = PORTC.1
motor1 = 1  'Declaramos el puerto C numero 1 encendido.
```

Fuente: elaboración propia.

De esta manera se simplifica la comprensión del código porque se sabe que se está encendiendo el motor DC en el puerto C número 1, antes declarado en el símbolo; y se puede usar este símbolo las veces que sea necesario.

3.8.5. Abreviaturas y tipos de numeración

Cuando se declaran valores constantes se escribe directamente el valor en décimas; si se antepone 0x o H, se está escribiendo en hexadecimal y si se antepone % al número, se está escribiendo en binario.

Figura 54. **Abreviaturas**

```
'Ejemplo de asignacion de nombres a las
'constantes anteponemos Const a la variable.

cost pi = 314  'Hacemos constante  314 a PI.

'Ejemplo de asignacion de valores decimales
'y hexadecimales y binarios a variables.

Dim x As Bit  'Declaramos x como bit.
Dim Y As Byte  'Declaramos y como byte.
x = True  'Es lo mismo que X = 1.
y = 0x55  'Asignamos valor a "Y" como hexadecimal.
Y= %01010101   'Asignamos valor a "Y" como binario.
```

Fuente: elaboración propia.

3.8.6. Goto

Esta instrucción es útil para transportar el flujo del mensaje a otro punto del código donde está la etiqueta seguida de “:”; este tipo de instrucción es muy útil para poder realizar un tipo de bucle controlado.

Figura 55. **Goto**

```
'Etiqueta a donde se quiere que se salte
'el flujo del programa.
proceso:

…..........................

'Se escribe GOTO previo a la etiqueta a
'donde se requiere el salto.

Goto proceso
```

Fuente: elaboración propia.

3.8.7. Operaciones aritméticas y lógicas

El simulador puede compilar 5 diferentes tipos de operaciones aritméticas básicas las cuales son: suma, resta, multiplicación, división (cociente), módulo aritmético y raíz cuadrada; los microcontroladores no pueden manipular variables decimales que no sean números enteros; por eso cualquier operación que sea con números decimales serán tomados como su valor entero. Ejemplo:

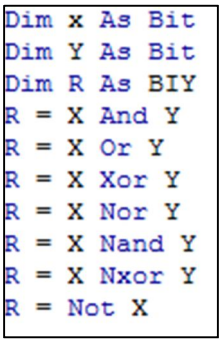
Figura 56. **Operaciones aritméticas**

```
'Declaramos num1 como palabra.
Dim num1 As Word

'Declaramos num2 como palabra.
Dim num2 As Word

'Declaramos num3 como palabra.
Dim num3 As Word

'Asignamos 15 a num1
num1 = 15

'Asignamos num1 = 15 y restamos 6 = 9.
num2 = num1 - 6

'Forma de realizar operaciones.
num3 = (9 + 2 * num2) - (125 / num1)

'Operamos raíz cuadrada de num1 = 3.
num1 = Sqr(num2)
```

Fuente: elaboración propia.

PIC Simulator IDE puede realizar también operaciones lógicas que en total son 7 las disponibles; de las cuales solo se puede realizar una sola por instrucción; a continuación se presentan algunos ejemplos.

Figura 57. **Operaciones lógicas**



```
Dim x As Bit
Dim Y As Bit
Dim R As BIY
R = X And Y
R = X Or Y
R = X Xor Y
R = X Nor Y
R = X Nand Y
R = X Nxor Y
R = Not X
```

Fuente: elaboración propia.

3.9. Configuraciones del PIC

El simulador ofrece opciones para configurar los distintos dispositivos que se conectarán al PIC a través de sus puertos; las más utilizadas en los proyectos son el ajuste del analógico/digital y la pantalla LCD, que son dispositivos de entrada y salida; en ellos se podrá analizar aritmética y lógicamente las diferentes señales que el PIC debe interpretar y traducir en información legible en la pantalla.

3.9.1. Programación para pantalla LCD

La configuración de la pantalla LCD 2x16 será realizada para que esta sea interconectada en el puerto B del PIC, así como se presentó previamente en algunos diagramas, la programación y explicación se presenta a continuación.

Figura 58. **Programación de pantalla LCD**

```
Define ADC_CLOCK = 3 'configuracion adc
Define ADC_SAMPLEUS = 10

'CONFIGURACIÓN DE LA PANTALLA LCD
Define LCD_BITS = 4
Define LCD_DREG = PORTB
Define LCD_DBIT = 4  'los 4 bits altos
Define LCD_RSREG = PORTB
Define LCD_RSBIT = 1
Define LCD_EREG = PORTB
Define LCD_EBIT = 3
Define LCD_RWREG = PORTB
Define LCD_RWBIT = 2
Lcdinit LcdCurOff

Dim an0 As Word

TRISA = 0xff  'colocamos PORTA completo como entrada
ADCON1 = 0  'colocamos el PORTA oomo entradas analógicas


loop:  'Inicio del ciclo llamado "loop"
   Adcin 0, an0
   Lcdcmdout LcdClear  'Limpia la LCD display.
   Lcdout "Entrada analoga "  'Texto en la primer línea.
   Lcdcmdout LcdLine2Home  'Coloca el cursor empezando
                           'en la linea 2.

   Lcdout "Valor: ", #an0  'En la segunda línea segido de # presenta
                           'el valor de la variable an0 en pantalla.

   WaitMs 1  'Espera un milisegundo para pasar a
             'la siguiente instruccion

Goto loop  'se salta a el inicio del ciclo llamado "loop"
```

Fuente: elaboración propia.

3.9.2. Definición de caracteres

PIC Simulator IDE proporciona la posibilidad de poder crear caracteres propios del usuario, dependiendo de la necesidad que se tenga, ya sea creando línea de la pantalla, figuras para crear una animación sencilla, o los símbolos que se puedan realizar dentro de un cuadro de 5 x 8; en la siguiente tabla se verá cómo se pueden crear caracteres a través de un simple código que se puede definir con numeración binaria; la segunda columna numerada, representa un carácter, las casillas que contienen un 1 verde van dando forma al carácter con las dimensiones mencionadas, en la pantalla LCD caben 16 en línea.

Tabla III. **Definición de caracteres, ejemplo 1**

	Numeración binaria					Suma binaria
Decimal por casilla	16	8	4	2	1	31
	1	1	1	1	1	31
	1	0	0	0	1	15
	0	1	0	1	0	10
	0	0	1	0	0	4
	0	0	1	0	0	4
	0	1	0	1	0	10
	1	0	0	0	1	15
	1	1	1	1	1	31
Dos sintaxis para definir los caracteres en el compilador	Para este caso se asigna el número 5 al carácter para ser llamado.					

```
Lcddefchar 5, 31, 15, 10, 4, 4, 10, 15, 31
Lcddefchar 5, %11111, %10001, %01010, %00100, %00100, %01010, %10001, %11111
```

Fuente: elaboración propia.

Tabla IV. **Definición de caracteres, ejemplo 2**

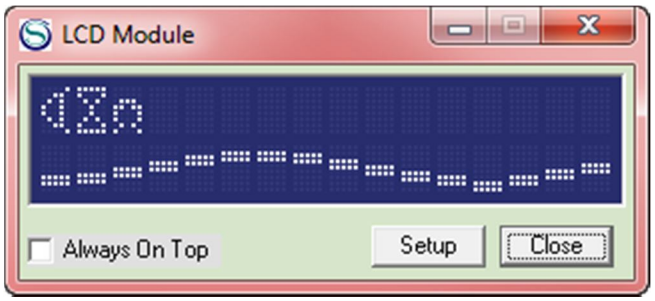
	Numeración binaria					Suma binaria
Decimal por casilla	16	8	4	2	1	31
	0	0	0	0	1	1
	0	0	1	1	0	6
	0	1	0	1	0	10
	1	0	0	1	0	18
	1	0	0	1	0	18
	0	1	0	1	0	10
	0	0	1	1	0	6
	0	0	0	0	1	1
Dos sintaxis para definir los caracteres en el compilador						
	Para este caso se asigna el número 6 al carácter para ser llamado.					

```
Lcddefchar 6, 1, 6, 10, 18, 18, 10, 6, 1
Lcddefchar 6, %00001, %00110, %01010, %10010, %10010, %01010, %00110, %00001
```

Fuente: elaboración propia.

Los caracteres que se vieron en las tablas anteriores, se observarán de la misma forma en la figura siguiente, que es una imagen del simulador de la pantalla LCD, siendo representados en los dos primeros caracteres de la primera fila de la pantalla, únicamente como demostración gráfica de lo escrito.

Figura 59. **Caracteres en el simulador**



Fuente: elaboración propia.

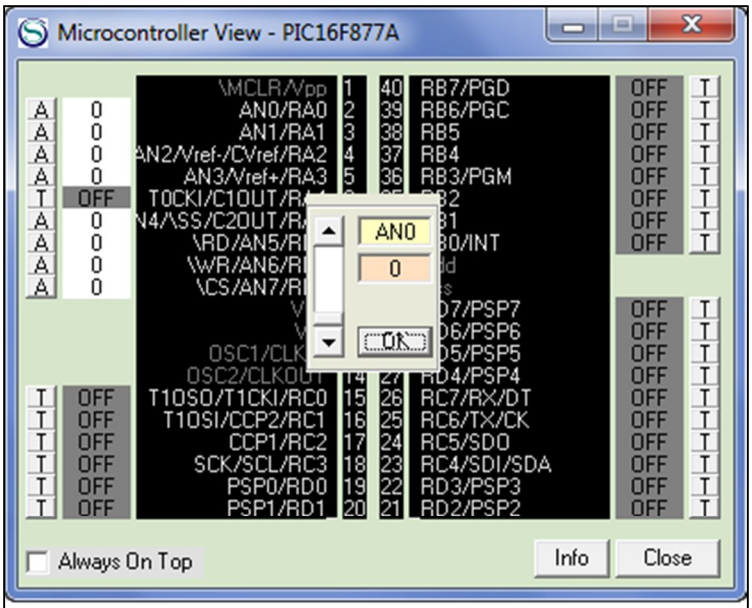
3.10. Configuración del ADC

Esta herramienta del PIC es muy útil para poder medir niveles de voltaje entrantes al PIC, los cuales deben de ser menores a 5 voltios; de ser mayor a esa la cantidad se corre el peligro de quemar el puerto asignado o bien el circuito integrado; esta herramienta es útil si se usa algún circuito de despliegue de niveles diferentes de voltaje, los cuales puedan ser identificados por el microcontrolador y dependiendo de estos valores, el PIC debe tomar decisiones de qué acción va a realizar.

Para la placa entrenadora que se presenta en este trabajo de graduación será necesario este segmento de programación, para poder identificar qué tecla se está presionando en el teclado matricial que se diseñó por medio de un circuito, que varíe el nivel de voltaje, dependiendo de la resistencia que esté en serie al botón que está siendo presionado; a continuación el segmento de programación necesario para configurar el ADC del PIC.

A continuación se muestra la ventana *Microcontroller view*, en ella se ha seleccionado el puerto A0 del PIC previamente configurado como ADC; este permite variar su nivel de bits desde 0 a 1024 bits; valor que será registrado por el PIC cuando este reciba la orden de adquirir el dato por medio del código.

Figura 60. **ADC del PIC**




```
'Configuracion del ADC.
'Definición de la frecuencia de reloj.
Define ADC_CLOCK = 3
'Defimociόm del periodo de muestreo.
Define ADC_SAMPLEUS = 50
ADCON0 = 14

'variables

Dim adc_cero As Word  'Definimos la
'variable adc_cero
```

Fuente: elaboración propia.

4. FABRICACIÓN DE PLACAS DE CIRCUITO IMPRESO

Para poder trabajar eficientemente con los microcontroladores se tiene que adquirir cierta habilidad para fabricar las propias placas, tanto las auxiliares como la placa principal del microcontrolador; para el siguiente procedimiento se utilizará el método conocido como el “de la plancha”, pues se utilizará una plancha convencional casera para planchar ropa; para transferir el tóner de una impresión láser o fotocopia de una placa de circuito de un acetato sobre una placa de baquelita o fibra de vidrio.

Este circuito puede ser diseñado en diferentes programas de creación de circuitos impresos “PCB” *(printed circuit board)* tales como PCB Wizard, Ultiboard o Proteus; el diseño de esta placa entrenadora de PICs fue realizado en PCB Wizard.

4.1. Elementos necesarios para realizar placas de circuito impreso

A continuación se muestran los dispositivos necesarios para poder llevar a cabo la creación de la placa propuesta para implementación de proyectos electrónicos:

4.1.1. Plancha para ropa

Se utilizará para calentar el tóner del impreso del acetato sobre la cara de cobre de la placa de fibra de vidrio o baquelita; el calor hace que el tóner se derrita y pueda ser transferido al cobre sin mayor esfuerzo.

Hay que tomar en cuenta que las planchas de vapor no son muy buenas para esta función, dado a que el calor no se genera de forma homogénea en toda la superficie del acetato, por los agujeros que tiene la plancha en su base; por lo que es más recomendable usar las clásicas de base plana.

Figura 61. **Plancha para ropa**

Fuente: elaboración propia.

4.1.2. Impresora láser o fotocopiadora

Para la impresión del acetato se necesitará impresión por tóner, por lo que se utilizará una fotocopiadora o impresora láser con el valor más alto de tóner y contraste sobre la impresión, para que la película de tóner sobre el acetato sea lo más gruesa posible, para realizar la transferencia de la impresión a la superficie de cobre de la forma más eficiente.

Figura 62. **Impresora láser**

Fuente: http://www.ucontrol.com.ar/wiki/images/9/90/Impresora.jpg.
Consulta: 12 de noviembre de 2012.

4.1.3. Cloruro férrico

El cloruro férrico es una sustancia que ayudará a atacar la superficie de cobre para eliminar todas las partes que no se necesiten; en sí la superficie de cobre que no esté protegida por la película de cobre o marcador permanente (se usará marcador permanente en caso de que sean necesarios retoques en la impresión) serán desintegradas por esta sustancia, para la cual solo es necesario ¼ de litro.

Figura 63. **Cloruro férrico**

Fuente: elaboración propia.

4.1.4. Taladro manual

Para poder hacer los agujeros donde serán insertados los dispositivos electrónicos es necesario un taladro manual para poder hacer estos trabajos de forma precisa y afinar el aspecto de la placa electrónica; por lo regular los kits de taladros manuales los hay de marca Dremel o Trupper, los cuales tienen accesorios compatibles y adaptadores para poder acondicionar brocas de 1 mm y 0.75 mm para los agujeros, discos y cilindros de lija y para afinar los bordes y dispositivos para pulir.

Figura 64. **Taladro manual**

Fuente: elaboración propia.

4.1.5. Esponja verde o de lana de acero

Para limpiar la placa y transferir la impresión de tóner y llevar a cabo el trabajo de soldadura, es necesario que se limpie la superficie con este tipo de material que es más suave que una lija y más áspero que la tela para quitar la suciedad y la grasa que se queda en la placa al manipularla con las manos; es necesario limpiarla para quede totalmente limpia, para evitar imperfecciones en las pistas del circuito.

Figura 65. **Esponja artificial**

Fuente: elaboración propia.

4.1.6. Recipiente plástico

Cuando ya se tiene la placa impresa llega el momento de bañarla en el cloruro férrico; para esto se necesita un recipiente que sea lo sufrientemente grande, como para que quepa la placa y la se pueda manipular con las manos.

Figura 66. **Recipiente plástico**

Fuente: elaboración propia.

4.1.7. Broca de 1/32'' y 1/64''

La broca de 1/32'' se usa para realizar agujeros donde se insertan los dispositivos como las terminales de 3 pines y circuitos integrados de 3 pines, como los reguladores de voltaje y transistores; la de 1/64'' es para hacer agujeros para insertar resistencias, capacitores, *sockets* de circuitos integrados como PICs, ULN2803 y leds que necesitan un agujero de menores dimensiones para ser insertados.

Figura 67. **Brocas**

Fuente: elaboración propia.

4.1.8. Placa de cobre

La placa de cobre es la base de la tarjeta electrónica; en ella se impregnará el diseño del circuito impreso en el acetato, que posteriormente será disuelto en cloruro férrico.

Figura 68. **Placa de cobre**

Fuente: http://personal.telefonica.terra.es/web/mamelytote/LED/led13-01.jpg.
Consulta: 14 de noviembre de 2012.

4.1.9. Marcador permanente punta fina

El marcador será de utilidad para retocar y perfeccionar la impresión de las pistas sobre la placa de cobre, debido a que en ocasiones no todas las pistas son transferidas en su totalidad; la tinta permanente funciona de la misma manera que el tóner, porque protege el cobre contra los efectos del cloruro férrico.

Figura 69. **Marcador permanente**

Fuente: http://www.audiocosas.es/principiantes/pcb/rotu.jpg.
Consulta: 16 de noviembre de 2012.

4.1.10. *Thinner* (adelgazador o diluyente)

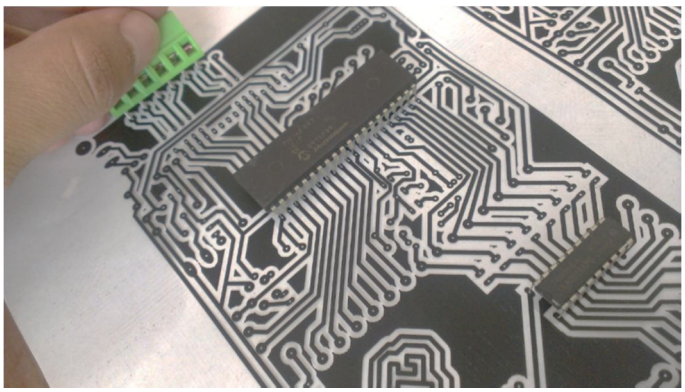
El *thinner* es un solvente químico derivado de componentes del petróleo que es utilizado para diluir pintura, que en este caso será de mucha utilidad para limpiar la placa de cobre, antes de imprimir el circuito impreso en el acetato y después cuando el cloruro férrico ha hecho su trabajo de disolver el cobre que no será de utilidad en el diseño de la placa.

4.1.11. Impresión del circuito en acetato

Cuando se tiene el diseño del circuito terminado, ya sea que se haya realizado con algún programa como PCB Wizard, a mano, sobre un papel o descargado de internet, se deben de tener en cuenta varias indicaciones previas a la impresión en acetato.

Una de las indicaciones es verificar que la impresión concuerde con la escala real, por lo cual es preferible verificar que los dispositivos electrónicos encajen perfectamente en el diseño, sobreponiéndolos sobre papel bond impreso para no desperdiciar acetatos; aunque de ser sumamente necesario, se puede recuperar el acetato impreso en tóner, limpiándolo con *thinner* y cuando ya haya sido removido todo el tóner, se procede a limpiarlo completamente con un trozo de tela seca, hasta que quede brillante de nuevo, para que no haya problemas a la hora de imprimirlo de nuevo, como si fuera una acetato virgen.

Figura 70. **Comparación de tamaño con CI**

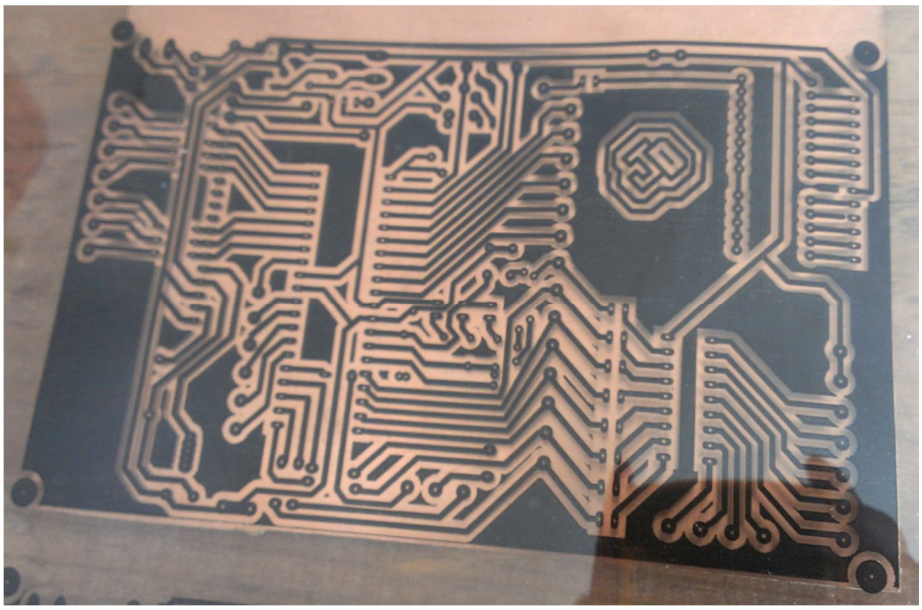


Fuente: elaboración propia.

Otra indicación necesaria para tener una óptima impresión del circuito sobre la placa, es verificar que la orientación es adecuada, porque se da el caso que el circuito está invertido en el momento de ubicar los componentes; para verificarlo simplemente, la impresión en acetato tiene que quedar igual (sin inversión a los ojos) a como se diseñó; cuando la impresión de la hoja está hacia arriba y cuando se ponga sobre la placa de cobre para plancharla, esta debe estar invertida a los ojos.

También a la hora de hacer la impresión definitiva en acetato, se debe verificar en las propiedades de impresión, que la opción de impresión económica esté desactivada, así como elevar el contraste a un nivel alto, porque se necesita una cantidad elevada de tóner en la impresión del acetato, ya que será transferida al cobre por medio de calor.

Figura 71. **Orientación de acetato**



Fuente: elaboración propia.

Cuando no se tiene impresora láser en casa, se puede fotocopiar el diseño en papel sobre acetato en una fotocopiadora; los cuidados que hay que tener es que la impresión no tenga imperfecciones como la escala a la que dicho diseño será impreso, porque no todas las fotocopiadoras o impresoras láser son iguales o cuando uno lleva el circuito en un archivo PDF, también puede generar variaciones de la escala.

Otros problemas son los rayones en blanco o rayones negros, manchas y líneas que pueden afectar al circuito, creando falsos contactos o cortos imprevistos; de no tener otra opción más que usar la mala impresión, se tiene que imprimir así como está y retocar la placa una vez esté ya impresa sobre el cobre.

Se puede utilizar una cuchilla o cualquier otro metal con punta para remover los excesos de tóner y un marcador permanente para retocar los puntos donde no llegó el tóner y así proteger ese tramo para que no sea disuelto por el cloruro férrico; después de haber verificado todas estas imperfecciones, se procede con el baño de cloruro férrico de la placa.

Se recomienda el uso del acetato por su bajo costo, comparado con el material termotransferible, alta calidad de recepción de la impresión y resistencia al calor; aunque hay que tener cuidado con el calor de la plancha, porque dependiendo de la calidad del acetato, este será más o menos resistente porque de ser una temperatura demasiado elevada, el acetato se deformará derritiéndose, provocando imperfecciones en la placa; la marca de acetatos más recomendada es Xerox.

4.2. Transferencia de la impresión del circuito al cobre

Se debe cortar la placa de cobre virgen a las dimensiones del diseño del circuito para no generar desperdicio de materiales; para esto se hará uso de las herramientas del taladro manual o una sierra delgada para cortar metal; con una lija se pueden remover estillas o imperfecciones en los bordes de la placa. El siguiente paso es la limpieza de la placa, para lo que se necesitará hacer uso de la esponja para lavar platos con la cual se raspará con fuerza la superficie de la placa hasta eliminar todo el óxido que haya en ella.

El *thinner* será de ayuda, ya que por sus propiedades químicas es muy eficiente para remover residuos de goma, grasa y otras suciedades que afecten la transferencia del tóner.

Ahora se procede con la impresión del tóner sobre el cobre; ya con la pieza de placa totalmente limpia, se colocará la impresión de acetato sobre la placa y con la plancha, a una temperatura de ¾ de la temperatura máxima; se procede a planchar hasta que la impresión se funda hacia la placa, y verificar que la misma se ha realizado, levantando con cuidado una de las esquinas del acetato.

Dependiendo de la plancha y del acetato, puede que la impresión dure de 10 segundos a 5 minutos; solo la práctica logrará la perfección de los impresos al cobre.

Con el acetato removido y la impresión realizada en su totalidad sobre la placa de cobre se procede al baño de cloruro férrico de la placa, para lo cual se introduce ¼ de litro de cloruro y la placa impresa dentro del recipiente plástico; se agita el cloruro, hasta que el cobre no cubierto por tóner o tinta permanente, sea desintegrado y removido de la placa.

4.3. Perforación de la placa impresa

Una vez se tenga completamente desintegrado el cobre que no fue cubierto por el tóner o tinta permanente, se deben hacer las perforaciones para introducir los dispositivos electrónicos que posteriormente serán soldados con estaño.

Se tienen dos tipos de broca, la de 1 mm o 1/32'' y la de 0.5 mm o 1/64''; la primera será útil para hacer perforaciones como los pines del *socket* del PIC o del ULN2803, las terminales verdes que se dirigen a los puertos y el regulador de voltaje 7805; la más pequeña de 0.5 mm o 1/64'' será útil para perforar los agujeros de las resistencias y puentes; se puede colocar una tabla debajo para evitar dañar la superficie de la mesa donde se estará perforando.

5. PROYECTOS DIDÁCTICOS CON PIC16F877A

En este capítulo se realizará una serie de programas básicos con programación, utilizándolos como ejemplos con base en PIC Basic compilado en el programa PIC Simulator IDE; en sí se hará el enfoque en los programas más elementales que utilizan al máximo las cualidades de la placa entrenadora descrita, y al mismo tiempo que se desarrolla el programa, se explicará cada uno de los aspectos del mismo desde el compilador en los cometarios color verde, que aparecerán en las figuras con código en lenguaje PIC basic.

Para explicar de una forma más didáctica los siguientes ejemplos de programación, se utilizarán imágenes del compilador; será de esta manera que se aprovechará el formato del texto que se presenta en la ventana que ofrece PIC Simulator IDE, en el cual se visualizará lo siguiente:

- El número de fila en la columna amarilla del lado izquierdo.

- Texto verde precedido de una apóstrofe denotará que el mismo se trata de un comentario, explicación o aclaración del programa que no será compilado.

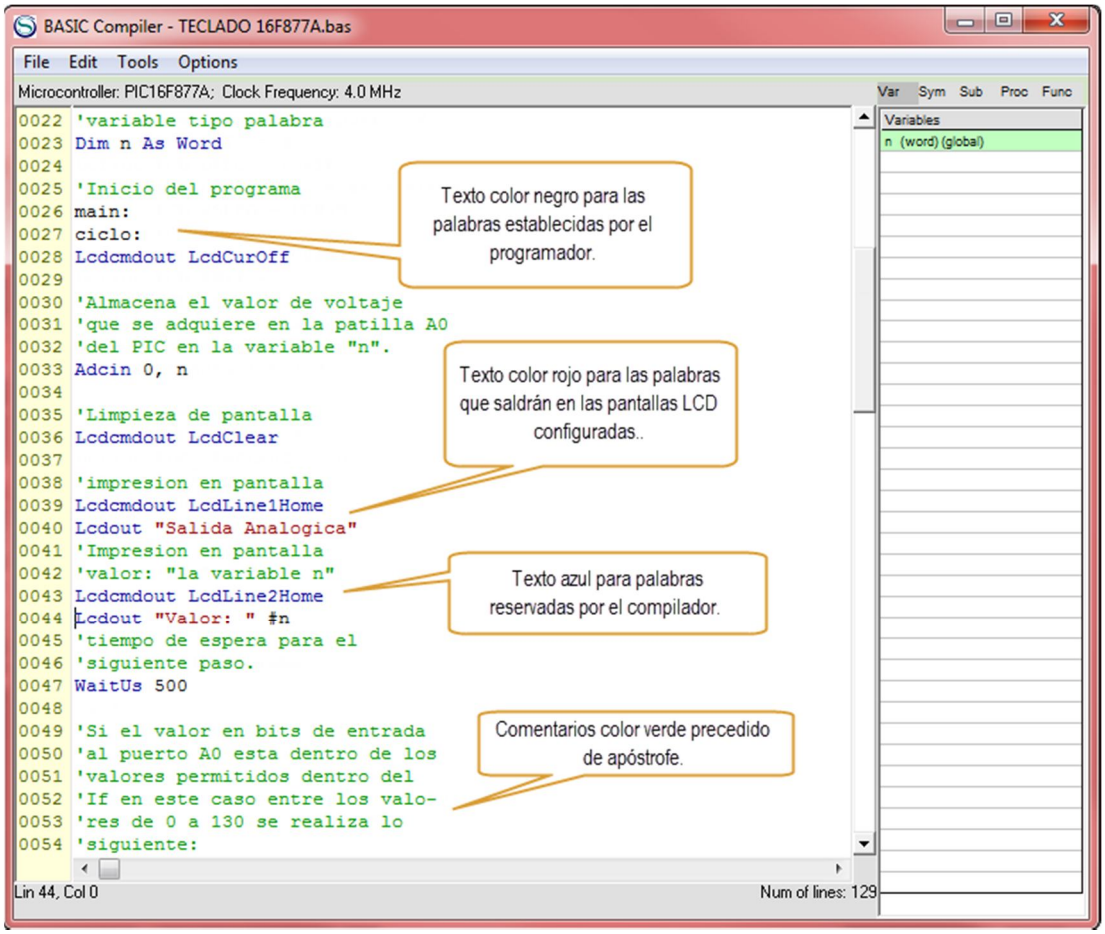
- El texto de color azul denotará que se trata de palabras reservadas por el compilador como GOTO, If, And, Then, Lcdcmout Lcdline1Home, etc.

- El texto en color negro corresponde al contenido o variables establecidas por el programador, como el nombre de las variables y números.

- El texto color rojo se refiere al que aparecerá en las pantallas configuradas LCD.

En la siguiente imagen se verá gráficamente lo descrito anteriormente.

Figura 72. **BASIC Compiler**



Fuente: elaboración propia.

5.1. Ejemplo No. 1: lector del ADC.bas

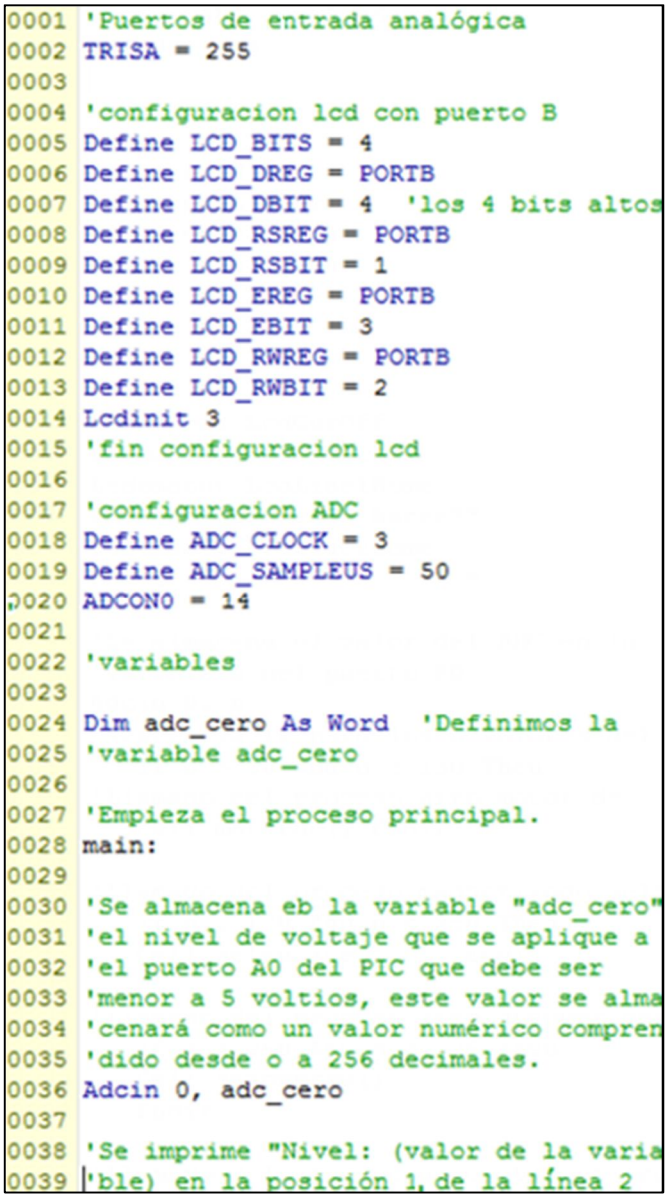
El siguiente programa está dedicado para la lectura analógica de los puertos A del PIC; todos tienen esta propiedad, la cual es muy útil para poder recoger mediciones variables y transformarlos en valores de 0 a 1024 bits, que podrán ser analizados por medio de esta programación; en los próximos ejemplos se verá cómo se utiliza esta herramienta, para multiplexar los valores de voltaje que variarán dependiendo del botón que se presione.

A la salida de cada uno de los botones, se implementa un diferencial de voltaje distinto que por medio de la programación pueden conocerse, pero se hará más el enfoque en el siguiente programa; por ejemplo, simplemente se recogerá el dato analógico que se está aplicando al puerto en ese momento y esta información será presentada como una cantidad de bits determinada.

Siempre entre el rango de 0 a 1024 bits, la estructura del programa es la siguiente:

- Configuración de la pantalla LCD
- Definición de variable
- Configuración de los puertos para que adquieran los valores del ADC
- Programa principal iniciando con un bucle
- Absorción del dato por medio de programación
- Presentación del programa en la pantalla LCD

Figura 73. **Programa lector ADC.bas 1/2**



```
0001 'Puertos de entrada analógica
0002 TRISA = 255
0003
0004 'configuracion lcd con puerto B
0005 Define LCD_BITS = 4
0006 Define LCD_DREG = PORTB
0007 Define LCD_DBIT = 4  'los 4 bits altos
0008 Define LCD_RSREG = PORTB
0009 Define LCD_RSBIT = 1
0010 Define LCD_EREG = PORTB
0011 Define LCD_EBIT = 3
0012 Define LCD_RWREG = PORTB
0013 Define LCD_RWBIT = 2
0014 Lcdinit 3
0015 'fin configuracion lcd
0016
0017 'configuracion ADC
0018 Define ADC_CLOCK = 3
0019 Define ADC_SAMPLEUS = 50
0020 ADCON0 = 14
0021
0022 'variables
0023
0024 Dim adc_cero As Word  'Definimos la
0025 'variable adc_cero
0026
0027 'Empieza el proceso principal.
0028 main:
0029
0030 'Se almacena eb la variable "adc_cero"
0031 'el nivel de voltaje que se aplique a
0032 'el puerto A0 del PIC que debe ser
0033 'menor a 5 voltios, este valor se alma
0034 'cenará como un valor numérico compren
0035 'dido desde o a 256 decimales.
0036 Adcin 0, adc_cero
0037
0038 'Se imprime "Nivel: (valor de la varia
0039 'ble) en la posición 1, de la línea 2
```

Fuente: elaboración propia.

Figura 74. **Programa lector ADC.bas 2/2**

```
0039 'ble) en la posición 1 de la linea 2
0040 'de la pantalla LCD.
0041 Lcdcmdout LcdLine2Pos(1)
0042 Lcdout "Nivel:      " #adc_cero
0043
0044 'Se espera medio segundo a que
0045 'se cambie de valor.
0046 WaitMs 500
0047
0048 'Se limpia la pantalla LCD con el
0049 'siguiente codigo.
0050 Lcdcmdout LcdLine2Clear
0051
0052 'Continua el ciclo infinito hiendo
0053 'a en "main" del programa.
0054 Goto main
0055
0056 'Fin del programa.
0057 End
```

Fuente: elaboración propia.

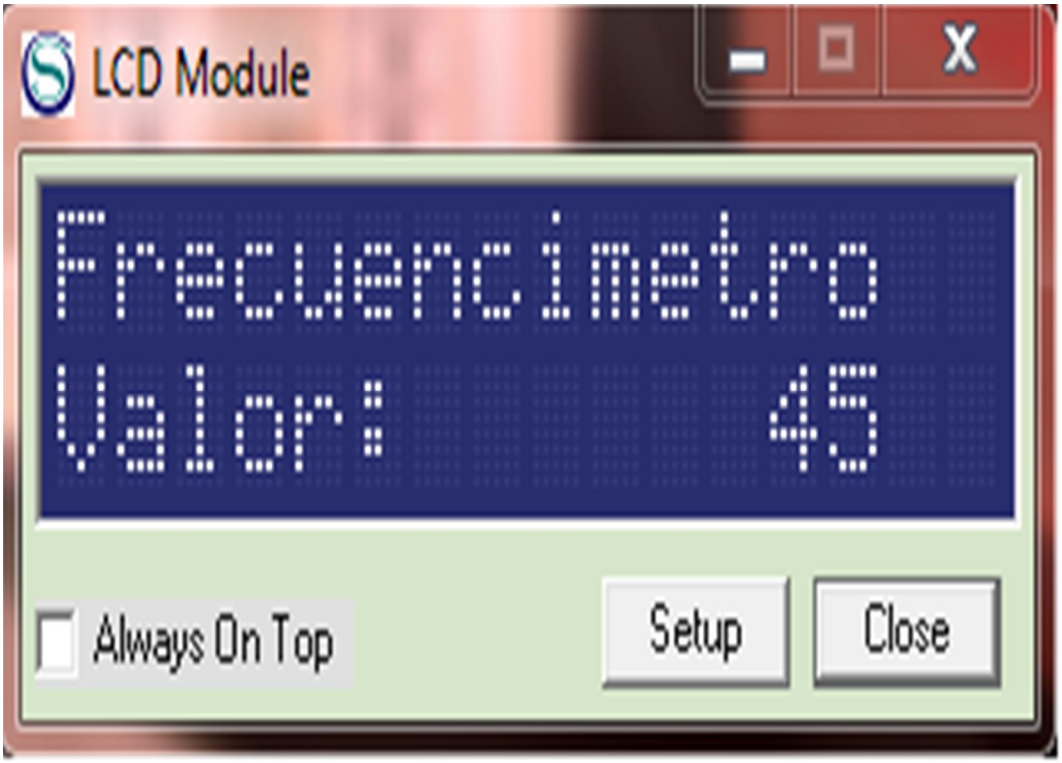
5.2. Ejemplo No. 2: frecuencímetro

Con este ejemplo se logrará medir la frecuencia de un pulso de entrada que será conectado al puerto D0 del PIC; para poder realizar la prueba de este programa, se puede conectar la salida de un circuito oscilador 555 a este puerto; es preferible que este circuito oscilador sea de frecuencia variable y así se verá la frecuencia a la que oscila en un periodo de 1 segundo; la estructura de este programa es la siguiente:

- Definición del puerto D0 como entrada
- Configuración de la pantalla
- Definición de variable frecuencia

- Inicio de programa principal
- Impresión “frecuencímetro” en la primer línea de la pantalla LCD
- Conteo de pulsos en un segundo
- Impresión en la pantalla en la línea 2 del valor de la frecuencia medida
- Retorno al ciclo

Figura 75. **Pantalla LCD “frecuencímetro”**



Fuente: elaboración propia.

Figura 76. **Frecuencímetro.bas**

```
0001 'Puertos de Entrada analógica.
0002 TRISD.0 = 1
0003
0004
0005 'Configuracion lcd con puerto B.
0006 Define LCD_BITS = 4
0007 Define LCD_DREG = PORTB
0008 Define LCD_DBIT = 4
0009 Define LCD_RSREG = PORTB
0010 Define LCD_RSBIT = 1
0011 Define LCD_EREG = PORTB
0012 Define LCD_EBIT = 3
0013 Define LCD_RWREG = PORTB
0014 Define LCD_RWBIT = 2
0015 Lcdinit 3
0016 'Fin configuracion lcd.
0017
0018 'Variables.
0019 Dim frecuencia As Word
0020
0021 main:
0022 'Borrar el cursor de pantalla.
0023 Lcdcmdout LcdCurOff
0024
0025 'Impresión en pantalla del texto
0026 '"Frecuencimetro".
0027 Lcdcmdout LcdLine1Pos(1)
0028 Lcdout "Frecuencimetro"
0029 'Loop para el conteo constante de
0030 'frecuencia.
0031 ciclo3:
0032 'Definicion de modo de conteo,
0033 'predeterminado = 2
0034 Define COUNT_MODE = 2
0035
0036 'Sintaxis de la medicion de frecuencia
0037 'Count "Puerto contador", durante 1 seg,
0038 'Variable que almacena la medición.
0039 Count PORTD.0, 1000, frecuencia
0040
0041 'Impresion en pantalla del valor medido.
0042 Lcdcmdout LcdLine2Pos(1)
0043       Lcdout "Valor:       " #frecuencia
0044 WaitMs 1000
0045 'Regreso al inicio del ciclo.
0046 Goto ciclo3
0047 End
0048
```

Fuente: elaboración propia.

5.3. Ejemplo No. 3: reconocimiento de teclado matricial

El siguiente programa demuestra el procedimiento que se ha diseñado para poder utilizar el teclado matricial que se conecta a la placa y que se comunica con el PIC por medio del puerto A0. El teclado enviará un nivel de voltaje distinto y dependiendo del botón presionado, será leído por el ADC del PIC e interpretado como un valor de 0 a 1024 bits.

Estos valores serán discriminados por los “IF” de la programación y según el rango que cumpla con las condiciones de los “IF”, entonces imprimirá en pantalla “presionó la tecla (n)”. Este proceso será enciclado para que se puedan probar todos los botones. En el programa se verán escritos los siguientes grupos de programación:

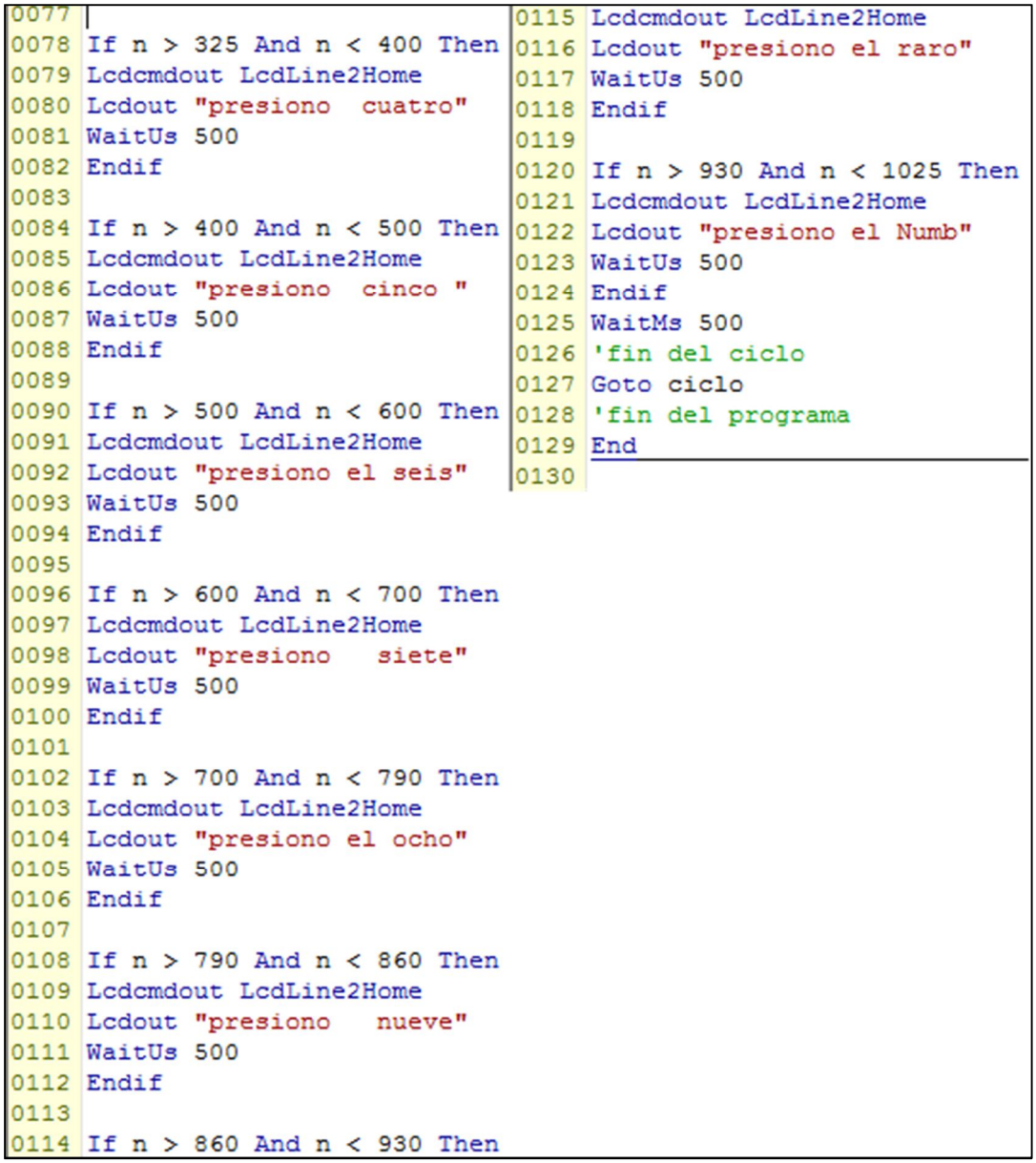
- Configuración de la pantalla LCD
- Configuración del ADC
- Definición de variables
- Programa principal
- Secciones de IF por botón presionado
- Bucle denotado con el nombre “ciclo”

Figura 77. **Teclado 16F877A.bas ½**

```
0001 'Configuracion lcd con puerto B
0002 Define LCD_BITS = 4
0003 Define LCD_DREG = PORTB
0004 Define LCD_DBIT = 4   '4 bits alto
0005 Define LCD_RSREG = PORTB
0006 Define LCD_RSBIT = 1
0007 Define LCD_EREG = PORTB
0008 Define LCD_EBIT = 3
0009 Define LCD_RWREG = PORTB
0010 Define LCD_RWBIT = 2
0011 Lcdinit 3
0012 'Fin configuracion lcd
0013
0014 'configuracion adc
0015 Define ADC_CLOCK = 3
0016 Define ADC_SAMPLEUS = 50
0017 TRISA = 0xff
0018 ADCON1 = 0
0019
0020
0021 'Se define "n" como
0022 'variable tipo palabra
0023 Dim n As Word
0024
0025 'Inicio del programa
0026 main:
0027 ciclo:
0028 Lcdcmdout LcdCurOff
0029
0030 'Almacena el valor de voltaje
0031 'que se adquiere en la patilla A0
0032 'del PIC en la variable "n".
0033 Adcin 0, n
0034
0035 'Limpieza de pantalla
0036 Lcdcmdout LcdClear
0037
0038 'impresion en pantalla
0039 Lcdcmdout LcdLine1Home
0040 Lcdout "Salida Analogica"
0041 'Impresion en pantalla
0042 'valor: "la variable n"
0043 Lcdcmdout LcdLine2Home
0044 Lcdout "Valor: " #n
0045 'tiempo de espera para el
0046 'siguiente paso.
0047 WaitMs 500
0048
0049 'Si el valor en bits de entrada
0050 'al puerto A0 está dentro de los
0051 'valores permitidos dentro del
0052 'If, en este caso entre los valo-
0053 'res de 0 a 130 se realiza lo
0054 'siguiente:
0055 If n > 0 And n < 130 Then
0056 'En la linea 2 posicion 0, imprime
0057 'en pantalla "presiono el uno"
0058 'que fue previamente asociado a
0059 'la tecla que se esta presionando
0060 'del teclado matricial.
0061 Lcdcmdout LcdLine2Home
0062 Lcdout "presiono el uno "
0063 WaitUs 500
0064 Endif
0065
0066 If n > 130 And n < 250 Then
0067 Lcdcmdout LcdLine2Home
0068 Lcdout "presiono el dos "
0069 WaitUs 500
0070 Endif
0071
0072 If n > 250 And n < 325 Then
0073 Lcdcmdout LcdLine2Home
0074 Lcdout "presiono el tres"
0075 WaitUs 500
0076 Endif
```

Fuente: elaboración propia

Figura 78. **Teclado 16F877A.bas 1/2**



```
0077
0078 If n > 325 And n < 400 Then
0079 Lcdcmdout LcdLine2Home
0080 Lcdout "presiono  cuatro"
0081 WaitUs 500
0082 Endif
0083
0084 If n > 400 And n < 500 Then
0085 Lcdcmdout LcdLine2Home
0086 Lcdout "presiono  cinco "
0087 WaitUs 500
0088 Endif
0089
0090 If n > 500 And n < 600 Then
0091 Lcdcmdout LcdLine2Home
0092 Lcdout "presiono el seis"
0093 WaitUs 500
0094 Endif
0095
0096 If n > 600 And n < 700 Then
0097 Lcdcmdout LcdLine2Home
0098 Lcdout "presiono   siete"
0099 WaitUs 500
0100 Endif
0101
0102 If n > 700 And n < 790 Then
0103 Lcdcmdout LcdLine2Home
0104 Lcdout "presiono el ocho"
0105 WaitUs 500
0106 Endif
0107
0108 If n > 790 And n < 860 Then
0109 Lcdcmdout LcdLine2Home
0110 Lcdout "presiono   nueve"
0111 WaitUs 500
0112 Endif
0113
0114 If n > 860 And n < 930 Then
0115 Lcdcmdout LcdLine2Home
0116 Lcdout "presiono el raro"
0117 WaitUs 500
0118 Endif
0119
0120 If n > 930 And n < 1025 Then
0121 Lcdcmdout LcdLine2Home
0122 Lcdout "presiono el Numb"
0123 WaitUs 500
0124 Endif
0125 WaitMs 500
0126 'fin del ciclo
0127 Goto ciclo
0128 'fin del programa
0129 End
0130
```

Fuente: elaboración propia.

5.4. Ejemplo No. 4: programa de animación en pantalla LCD

Ahora podrá apreciarse un ejemplo sencillo del tipo de animaciones que se pueden realizar con simple programación, siempre implementándolo en la pantalla LCD que está configurada en el puerto B del PIC, con código que lleva una sintaxis que emplea numeración binaria para poder comprender la posición en que se están ubicando los pixeles, en los caracteres que pueden ser predefinidos por el programador, y al mismo tiempo ubicando los caracteres en la pantalla por medio del código “Lcdcmdout Lcdlinepos (posición)”. Las siguientes imágenes son muestras de las animaciones posibles en el siguiente programa.

Figura 79. **Animaciones en LCD**

Fuente: elaboración propia.

Figura 80. **Animacion.bas 1/4**

```
0001 'Configuracion lcd con puerto B.
0002 Define LCD_BITS = 4
0003 Define LCD_DREG = PORTB
0004 Define LCD_DBIT = 4  'los 4 bits altos
0005 Define LCD_RSREG = PORTB
0006 Define LCD_RSBIT = 1
0007 Define LCD_EREG = PORTB
0008 Define LCD_EBIT = 3
0009 Define LCD_RWREG = PORTB
0010 Define LCD_RWBIT = 2
0011 Lcdinit 3
0012 'Fin configuracion lcd.
0013
0014 'Cconfiguracion ADC
0015 Define ADC_CLOCK = 3
0016 Define ADC_SAMPLEUS = 50
0017 ADCON0 = 14
0018
0019 'Variables.
0020 main:
0021
0022 'Llama proceso barras de animacion.
0023    Call bar()
0024 loop:
0025 'Llama proceso instroduccion.
0026    Call intro()
0027 'Llama proceso animacion apagado.
0028    Call apagando()
0029 Goto loop
0030 End
0031 'En el siguiente proceso se definen las barras
0032 'que serán necesarias para poder realizar la
0033 'animación en la pantalla.
0034 Proc bar()  'Se llama al proceso.
0035 Lcdcmdout LcdCurOff  'Quitamos cursor intermitente.
0036
0037 'Definimos los caracteres siendo "0" el simbolo de resistencia "ohm"
0038 Lcddefchar 0, %000000, %000000, %01110, %11011, %10001, %10001, %01010, %10001
```

Fuente: elaboración propia.

Figura 81. **Animacion.bas 2/4**

```
0039 'Definimos los caracteres como barras horizonteles.
0040 Lcddefchar 1, 0, 0, 0, 0, 0, 0, 31, 31
0041 Lcddefchar 2, 0, 0, 0, 0, 0, 31, 31, 0
0042 Lcddefchar 3, 0, 0, 0, 0, 31, 31, 0, 0
0043 Lcddefchar 4, 0, 0, 0, 31, 31, 0, 0, 0
0044 Lcddefchar 5, 0, 0, 31, 31, 0, 0, 0, 0
0045 Lcddefchar 6, 0, 31, 31, 0, 0, 0, 0, 0
0046 Lcddefchar 7, 31, 31, 0, 0, 0, 0, 0, 0
0047
0048 'Se imprime en pantalla la palabra tesis con barras.
0049 Lcdcmdout LcdLine1Pos(1)
0050 Lcdout 1, 2, 3, "***tesis**", 3, 2, 1
0051 'Se imprime en pantalla 2013 con barras.
0052 Lcdcmdout LcdLine2Pos(1)
0053 Lcdout 1, 2, 3, 4, 5, 6, "2013", 6, 5, 4, 3, 2, 1
0054 WaitMs 1500
0055 'Se limpia la primer línea de la pantalla.
0056 Lcdcmdout LcdLine1Clear
0057
0058 End Proc
0059 'En este proceso se grafica una línea curva con
0060 'ayuda de los vectores que se predefine en la
0061 'sección de barras.
0062 Proc intro()
0063 Lcdcmdout LcdCurOff
0064 Lcdcmdout LcdLine1Pos(1)
0065 Lcdout 0
0066 WaitMs 150
0067 Lcdcmdout LcdLine1Pos(1)
0068 Lcdout " ", 2
0069 WaitMs 150
0070 Lcdcmdout LcdLine1Pos(2)
0071 Lcdout " ", 3
0072 WaitMs 150
0073 Lcdcmdout LcdLine1Pos(3)
0074 Lcdout " ", 4
0075 WaitMs 150
0076 Lcdcmdout LcdLine1Pos(4)
0077 Lcdout " ", 5
0078 WaitMs 150
0079 Lcdcmdout LcdLine1Pos(5)
0080 Lcdout " ", 6
0081 WaitMs 150
0082 Lcdcmdout LcdLine1Pos(6)
0083 Lcdout " ", 7
0084 WaitMs 150
0085 Lcdcmdout LcdLine1Pos(7)
0086 Lcdout " ", 6
0087 WaitMs 150
0088 Lcdcmdout LcdLine1Pos(8)
0089 Lcdout " ", 5
0090 WaitMs 150
0091 Lcdcmdout LcdLine1Pos(9)
0092 Lcdout " ", 4
0093 WaitMs 150
0094 Lcdcmdout LcdLine1Pos(10)
0095 Lcdout " ", 3
0096 WaitMs 150
0097 Lcdcmdout LcdLine1Pos(11)
0098 Lcdout " ", 2
0099 WaitMs 150
0100 Lcdcmdout LcdLine1Pos(12)
0101 Lcdout " ", 1
0102 WaitMs 150
0103 Lcdcmdout LcdLine1Pos(13)
0104 Lcdout " ", 2
0105 WaitMs 150
0106 Lcdcmdout LcdLine1Pos(14)
0107 Lcdout " ", 3
0108 WaitMs 150
0109 Lcdcmdout LcdLine1Pos(15)
0110 Lcdout " ", 4
0111 WaitMs 150
0112 Lcdcmdout LcdLine1Pos(16)
0113 Lcdout " "
0114 WaitMs 150
```

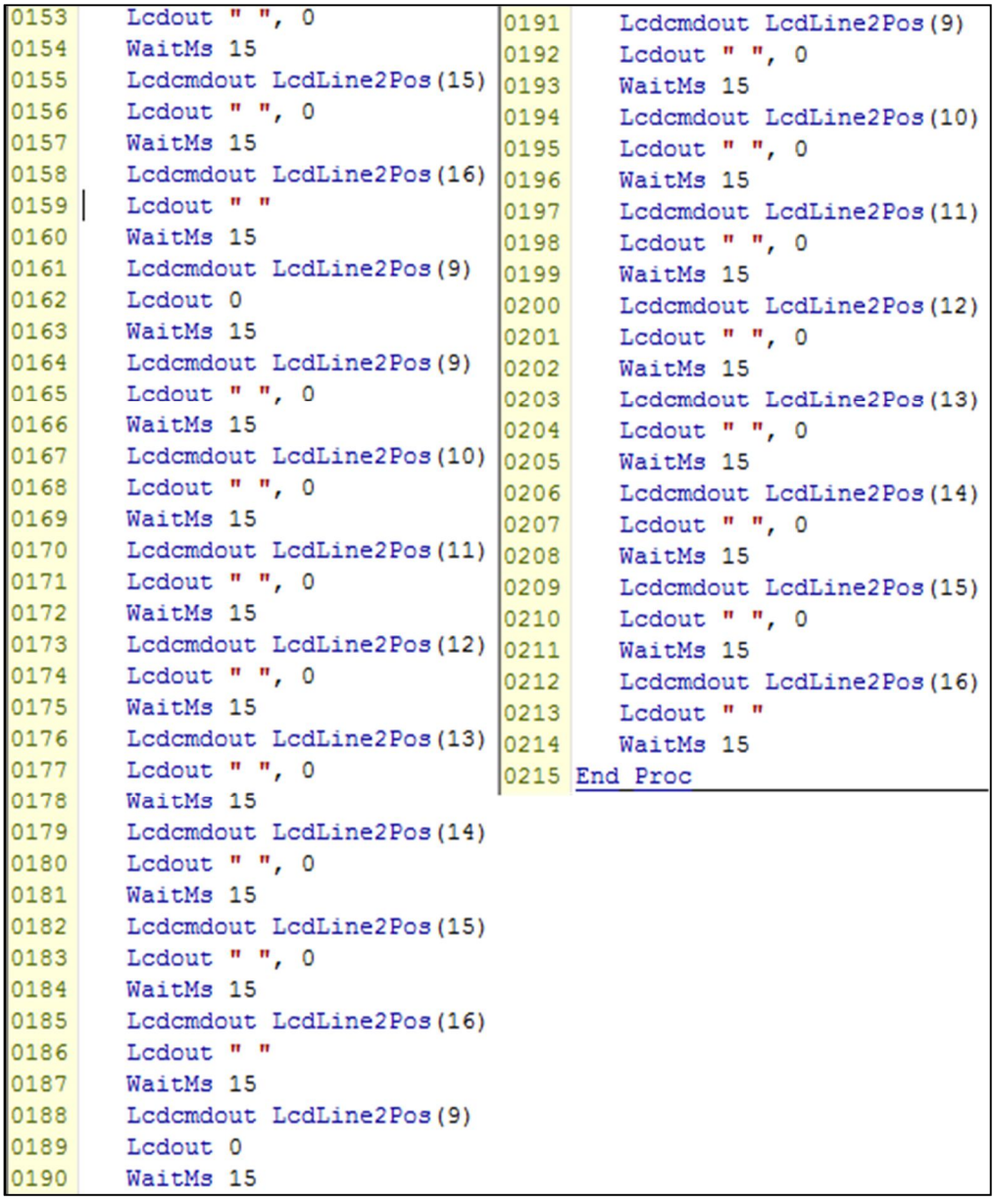
Fuente: elaboración propia.

Figura 82. **Animacion.bas 3/4**

```
0115
0116 End Proc
0117 'Se define una segunda animación,la cual sera primero
0118 'la presentación del tecto "Tesis*********" seguido de
0119 'un juego de garras, terminando con el símbolo omega
0120 'moviéndose a partir de la posición 9 de la línea 2.
0121 Proc apagando()
0122 Lcdcmdout LcdLine1Pos(1)
0123 Lcdout 0, "TESIS**********"
0124 Lcdcmdout LcdLine2Pos(1)
0125 Lcdout 1, 2, 3, 4, 5, 6, 7, 6, 5, 4, 3, 2, 1, 2, 3, 4
0126 WaitMs 20
0127
0128 Lcdcmdout LcdLine2Pos(1)
0129 Lcdout "  DEMOSTRACIÓN  "
0130 'Se posiciona el carácter "0"
0131 'aumentando una posición e
0132 'iniciando en la posición 9 de
0133 'la línea 2 de  la pantalla LCD
0134    Lcdcmdout LcdLine2Pos(9)
0135    Lcdout 0
0136    WaitMs 15
0137    Lcdcmdout LcdLine2Pos(9)
0138    Lcdout " ", 0
0139    WaitMs 15
0140    Lcdcmdout LcdLine2Pos(10)
0141    Lcdout " ", 0
0142    WaitMs 15
0143    Lcdcmdout LcdLine2Pos(11)
0144    Lcdout " ", 0
0145    WaitMs 15
0146    Lcdcmdout LcdLine2Pos(12)
0147    Lcdout " ", 0
0148    WaitMs 15
0149    Lcdcmdout LcdLine2Pos(13)
0150    Lcdout " ", 0
0151    WaitMs 15
0152    Lcdcmdout LcdLine2Pos(14)
```

Fuente: elaboración propia.

Figura 83. **Animacion.bas 4/4**



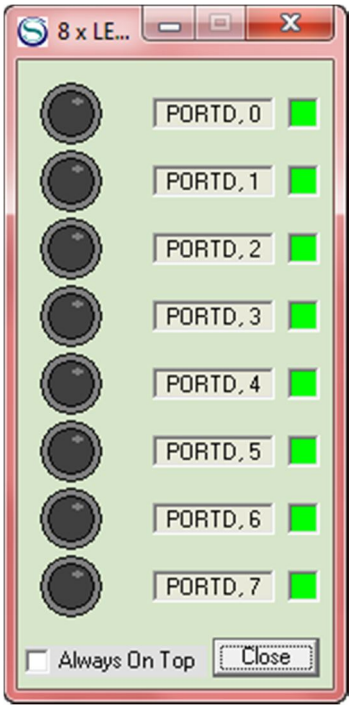
```
0153    Lcdout " ", 0
0154    WaitMs 15
0155    Lcdcmdout LcdLine2Pos(15)
0156    Lcdout " ", 0
0157    WaitMs 15
0158    Lcdcmdout LcdLine2Pos(16)
0159    Lcdout " "
0160    WaitMs 15
0161    Lcdcmdout LcdLine2Pos(9)
0162    Lcdout 0
0163    WaitMs 15
0164    Lcdcmdout LcdLine2Pos(9)
0165    Lcdout " ", 0
0166    WaitMs 15
0167    Lcdcmdout LcdLine2Pos(10)
0168    Lcdout " ", 0
0169    WaitMs 15
0170    Lcdcmdout LcdLine2Pos(11)
0171    Lcdout " ", 0
0172    WaitMs 15
0173    Lcdcmdout LcdLine2Pos(12)
0174    Lcdout " ", 0
0175    WaitMs 15
0176    Lcdcmdout LcdLine2Pos(13)
0177    Lcdout " ", 0
0178    WaitMs 15
0179    Lcdcmdout LcdLine2Pos(14)
0180    Lcdout " ", 0
0181    WaitMs 15
0182    Lcdcmdout LcdLine2Pos(15)
0183    Lcdout " ", 0
0184    WaitMs 15
0185    Lcdcmdout LcdLine2Pos(16)
0186    Lcdout " "
0187    WaitMs 15
0188    Lcdcmdout LcdLine2Pos(9)
0189    Lcdout 0
0190    WaitMs 15
0191    Lcdcmdout LcdLine2Pos(9)
0192    Lcdout " ", 0
0193    WaitMs 15
0194    Lcdcmdout LcdLine2Pos(10)
0195    Lcdout " ", 0
0196    WaitMs 15
0197    Lcdcmdout LcdLine2Pos(11)
0198    Lcdout " ", 0
0199    WaitMs 15
0200    Lcdcmdout LcdLine2Pos(12)
0201    Lcdout " ", 0
0202    WaitMs 15
0203    Lcdcmdout LcdLine2Pos(13)
0204    Lcdout " ", 0
0205    WaitMs 15
0206    Lcdcmdout LcdLine2Pos(14)
0207    Lcdout " ", 0
0208    WaitMs 15
0209    Lcdcmdout LcdLine2Pos(15)
0210    Lcdout " ", 0
0211    WaitMs 15
0212    Lcdcmdout LcdLine2Pos(16)
0213    Lcdout " "
0214    WaitMs 15
0215 End Proc
```

Fuente: elaboración propia.

5.5. Ejemplo No. 5: programa de secuencia de LEDs

Al programar una secuencia de encendido de LEDs se verá cómo se puede encender uno de ellos y apagar el resto simultáneamente con apoyo de la declaración de encendido/apagado en forma binaria, con la sintaxis PORTC=%00000111, siendo los LEDs que estén conectados al PIC por el puerto C, encendiendo los puertos 0, 1, 2, y apagando el resto; para lo cual, a la hora de simularlo, se puede hacer uso de las herramientas de simulación de PIC Simulador IDE, tal como sucede en el 8 Led board, que se visualiza en la siguiente imagen.

Figura 84. **8 LED board**



Fuente: elaboración propia.

Figura 85. **Secuencia de LEDS.bas 1/2**

```
'Definimos el puerto C como
'puerto de salida.
TRISC = 0

'configuracion lcd con puerto B
Define LCD_BITS = 4
Define LCD_DREG = PORTB
Define LCD_DBIT = 4  'los 4 bits altos
Define LCD_RSREG = PORTB
Define LCD_RSBIT = 1
Define LCD_EREG = PORTB
Define LCD_EBIT = 3
Define LCD_RWREG = PORTB
Define LCD_RWBIT = 2
Lcdinit 3
'fin configuracion lcd

'El programa presentara en en
'la pantalla LCD el numero de
'LED que se esta iluminando y
'al mismo tiempo se activara el
'puerto adecuado para encender
'el LED simultaneamente.

'Inicio del programa principal.
main:

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdCurOff

'Empieza el ciclo infinito.
loop:
'Se limpia la linea 2 de la LCD
Lcdcmdout LcdLine2Clear

'En la linea 2 de la LCD se impri-
'me el numero de LED que se ilumina.
Lcdcmdout LcdLine2Pos(1)
Lcdout "1"

'Se activa el puerto C0 del PIC
'encendiendo el LED 1.
PORTC = %00000001

'Esperamos 100 milisegundos.
WaitMs 100

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdLine2Pos(2)
Lcdout "2"
PORTC = %00000010
WaitMs 100

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdLine2Pos(3)
Lcdout "3"
PORTC = %00000100
WaitMs 100

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdLine2Pos(4)
Lcdout "4"
PORTC = %00001000
WaitMs 100

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdLine2Pos(1)
Lcdout "1"
PORTC = %00010000
WaitMs 100

Lcdcmdout LcdLine2Clear
Lcdcmdout LcdLine2Pos(5)
Lcdout "5"
PORTC = %00100000
WaitMs 100
```

Fuente: elaboración propia.

Figura 86. **Secuencia de LEDS.bas 2/2**

```
0077
0078 Lcdcmdout LcdLine2Clear
0079 Lcdcmdout LcdLine2Pos(6)
0080 Lcdout "6"
0081 PORTC = %01000000
0082 WaitMs 100
0083
0084 Lcdcmdout LcdLine2Clear
0085 Lcdcmdout LcdLine2Pos(7)
0086 Lcdout "7"
0087 PORTC = %01000000
0088 WaitMs 100
0089
0090 Lcdcmdout LcdLine2Clear
0091 Lcdcmdout LcdLine2Pos(8)
0092 Lcdout "8"
0093 PORTC = %10000000
0094 WaitMs 100
0095
0096 Lcdcmdout LcdLine2Clear
0097 Lcdcmdout LcdLine2Pos(7)
0098 Lcdout "7"
0099 PORTC = %01000000
0100 WaitMs 100
0101
0102 Lcdcmdout LcdLine2Clear
0103 Lcdcmdout LcdLine2Pos(6)
0104 Lcdout "6"
0105 PORTC = %00100000
0106 WaitMs 100
0107
0108 Lcdcmdout LcdLine2Clear
0109 Lcdcmdout LcdLine2Pos(5)
0110 Lcdout "5"
0111 PORTC = %00010000
0112 WaitMs 100
0113
0114 Lcdcmdout LcdLine2Clear
0115 Lcdcmdout LcdLine2Pos(4)
0116 Lcdout "4"
0117 PORTC = %00001000
0118 WaitMs 100
0119
0120 Lcdcmdout LcdLine2Clear
0121 Lcdcmdout LcdLine2Pos(3)
0122 Lcdout "3"
0123 PORTC = %00000100
0124 WaitMs 100
0125
0126 Lcdcmdout LcdLine2Clear
0127 Lcdcmdout LcdLine2Pos(2)
0128 Lcdout "2"
0129 PORTC = %00000010
0130 WaitMs 100
0131
0132 Lcdcmdout LcdLine2Clear
0133 Lcdcmdout LcdLine2Pos(1)
0134 Lcdout "1"
0135 PORTC = %00000001
0136 WaitMs 100
0137
0138 'Se retorna a Loop.
0139 Goto loop
0140
0141 'Fin del programa.
0142 End
```

Fuente: elaboración propia.

5.6. Ejemplo No.6: reloj con PIC

Los microcontroladores se pueden encontrar en casi cualquier objeto que utiliza una cantidad mínima de tecnología al mejor precio; con el PIC se puede crear un reloj que tendrá la precisión de un reloj de cuarzo, porque está siendo temporizado por uno; este reloj simplemente funciona manipulando las variables correspondientes a la hora, minuto y segundo; cuando los segundos llegan a 59 regresa a 0; cuando los minutos llegan a 59 regresan a 0, y cuando las horas llegan a 23, también regresan a 0; con base en esta lógica, se implementa el siguiente programa.

Figura 87. **Reloj en LCD**

Fuente: elaboración propia.

Figura 88. **Reloj con PIC.bas 1/4**

```
0001 Dim hr As Byte
0002 Dim min As Byte
0003 Dim seg As Byte
0004
0005 TRISC.0 = 1
0006 TRISC.1 = 1
0007 TRISC.2 = 1  'in
0008 TRISC.3 = 0  'out
0009 TRISD.0 = 1
0010
0011 'configuracion adc
0012 Define ADC_CLOCK = 3
0013 Define ADC_SAMPLEUS = 50
0014 TRISA = 0xff
0015 ADCON1 = 0
0016
0017 'CONFIGURACION DE LA PANTALLA LCD
0018 Define LCD_BITS = 4
0019 Define LCD_DREG = PORTB
0020 Define LCD_DBIT = 4  'los 4 bits altos
0021 Define LCD_RSREG = PORTB
0022 Define LCD_RSBIT = 1
0023 Define LCD_EREG = PORTB
0024 Define LCD_EBIT = 3
0025 Define LCD_RWREG = PORTB
0026 Define LCD_RWBIT = 2
0027 Lcdinit LcdCurOff
0028
0029
0030 Dim n As Word
0031 'Raiz del prgama
0032 hr = 0
0033 seg = 0
0034 min = 0
0035 main:
0036
0037 Call ajuste()  'procedimientos de ajuste de hora
0038 Lcdcmdout LcdClear
```

Fuente: elaboración propia.

Figura 89. **Reloj con PIC.bas 2/4**

```
0039 Call clock()  'procedimeinto de reloj
0040 Lcdcmdout LcdClear
0041 End
0042 Proc ajuste()
0043
0044 While PORTC.1 = 0
0045       Lcdcmdout LcdLine1Pos(1)
0046       Lcdout "Ajuste de hora.."
0047       Lcdcmdout LcdLine2Pos(1)
0048       Lcdout #hr, " :", #min, " :", #seg, "        "
0049
0050
0051 Adcin 0, n
0052
0053 If n > 75 And n < 130 Then  '1
0054    hr = hr + 1
0055    Endif
0056
0057 If n > 130 And n < 250 Then  '2
0058    min = min + 1
0059    Endif
0060
0061 If n > 250 And n < 325 Then  '3
0062    seg = seg + 1
0063    Endif
0064
0065 If n > 325 And n < 400 Then  '4
0066    hr = hr - 1
0067    Endif
0068
0069 If n > 400 And n < 500 Then  '5
0070    min = min - 1
0071    Endif
0072
0073 If n > 500 And n < 600 Then  '6
0074    seg = seg - 1
0075 Endif
0076
```

Fuente: elaboración propia.

Figura 90. **Reloj con PIC.bas 3/4**

```
0077 WaitMs 500
0078 Wend
0079 End Proc
0080 Proc clock()
0081 While hr < 24
0082
0083 If PORTD.0 = 1 Then
0084 Call ajuste()
0085 Endif
0086
0087 If min = 60 Then  'vuelve cero minutos
0088 min = 0
0089 Endif
0090
0091 If seg = 60 Then  'vuelve cero segundos|
0092 seg = 0
0093 Endif
0094
0095 If hr = 24 Then  'vuelve cero horas
0096 hr = 0
0097 Endif
0098       Lcdcmdout LcdLine1Pos(1)
0099       Lcdout "Reloj de PIC    "
0100
0101
0102 If min < 60 And seg < 60 Then
0103    Lcdcmdout LcdLine2Pos(1)
0104    Lcdout #hr, " :", #min, " :", #seg, "           "
0105 Endif
0106
0107 If hr = 3 Then
0108 PORTC.3 = 1
0109 Endif
0110
0111 If seg < 60 Then  'aumenta 1 a segundos
0112 seg = seg + 1
0113 Endif
0114
```

Fuente: elaboración propia.

Figura 91. **Reloj con PIC.bas 4/4**

```
0115
0116 If seg = 60 Then  'aumenta uno a minutos
0117 min = min + 1
0118 Endif
0119
0120 If min = 60 Then  'aumenta uno a horas
0121 hr = hr + 1
0122 Endif
0123
0124 If PORTD.0 = 1 Then
0125 Call ajuste()
0126 Endif
0127 WaitMs 1000
0128 Wend
0129 End Proc
```

Fuente: elaboración propia.

5.7. Ejemplo No.7: programa 4 motores + leds

La versatilidad de los PICs es lo que los hace tan prácticos, útiles y didácticos, permitiendo interconectar distintos dispositivos como motores *stepper*, motores DC y bancos de LEDs; en fin, cualquier dispositivo que pueda ser útil para realizar cualquier tipo de proyecto de nivel medio y alto. En el siguiente ejemplo se explicará cómo se puede realizar la interconexión de distintos dispositivos electromecánicos al PIC, a través de un circuito integrado ULN2803, que está incluido en la placa explicada previamente y que servirá para amplificar la corriente saliente del PIC.

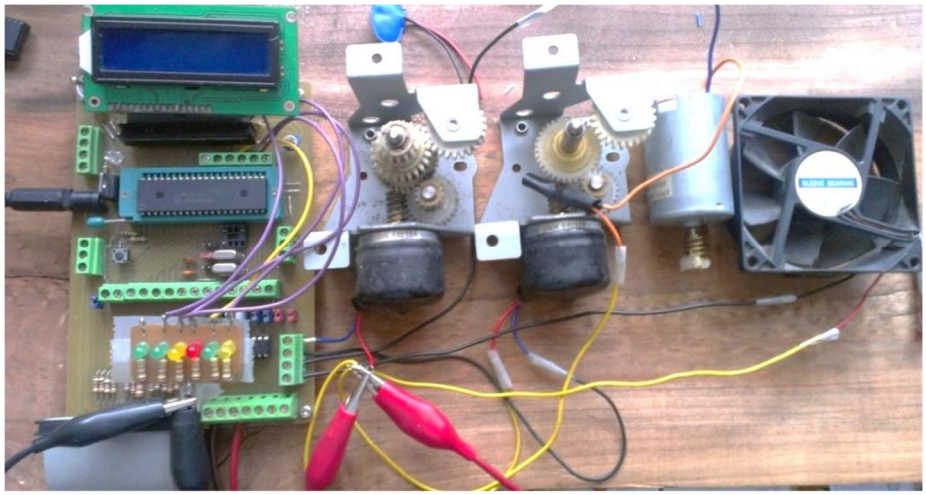
El programa presenta la siguiente estructura:

- Configuración de la pantalla LCD
- Definición de los puertos D y C como puertos de salida

- Configuración del ADC
- Definición de variables
- Inicialización de los puertos D y C
- Programa principal
- Menú
    - Ciclo de selección de opción
        - Motor DC + número, presionar el botón que corresponda al botón 1, 2, 3, 4 ó Leds.

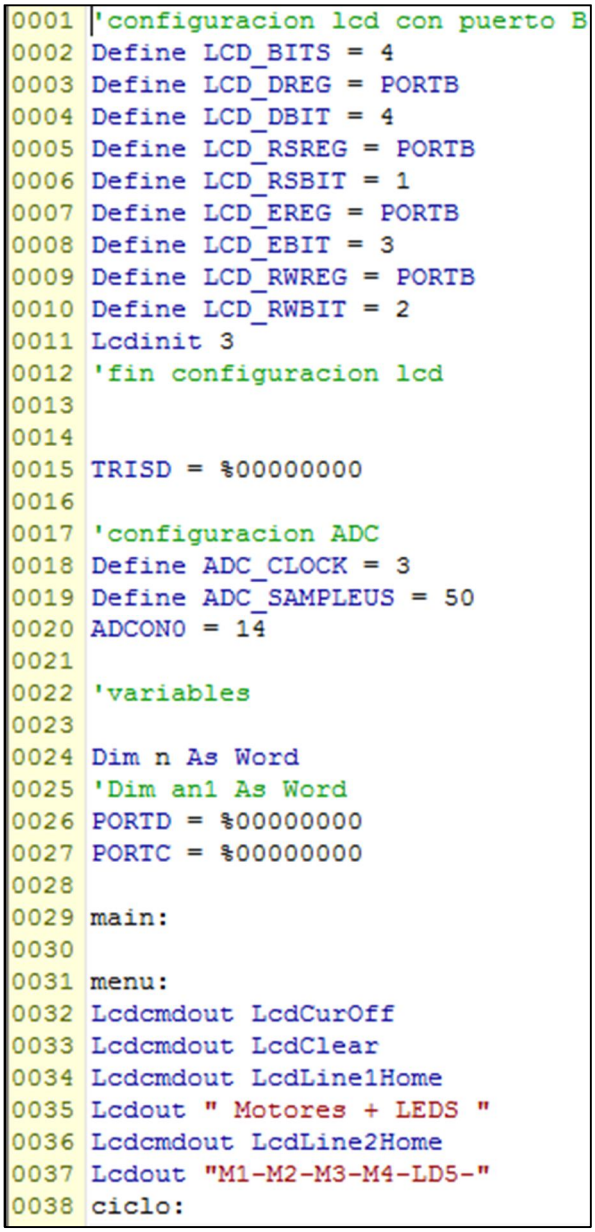
- Fin del programa principal
- Proceso Motordc1()
- Proceso Motordc2()
- Proceso Motordc3()
- Proceso Motordc4()
- Leds

Figura 92. **Lo que se podría conectar**



Fuente: elaboración propia.

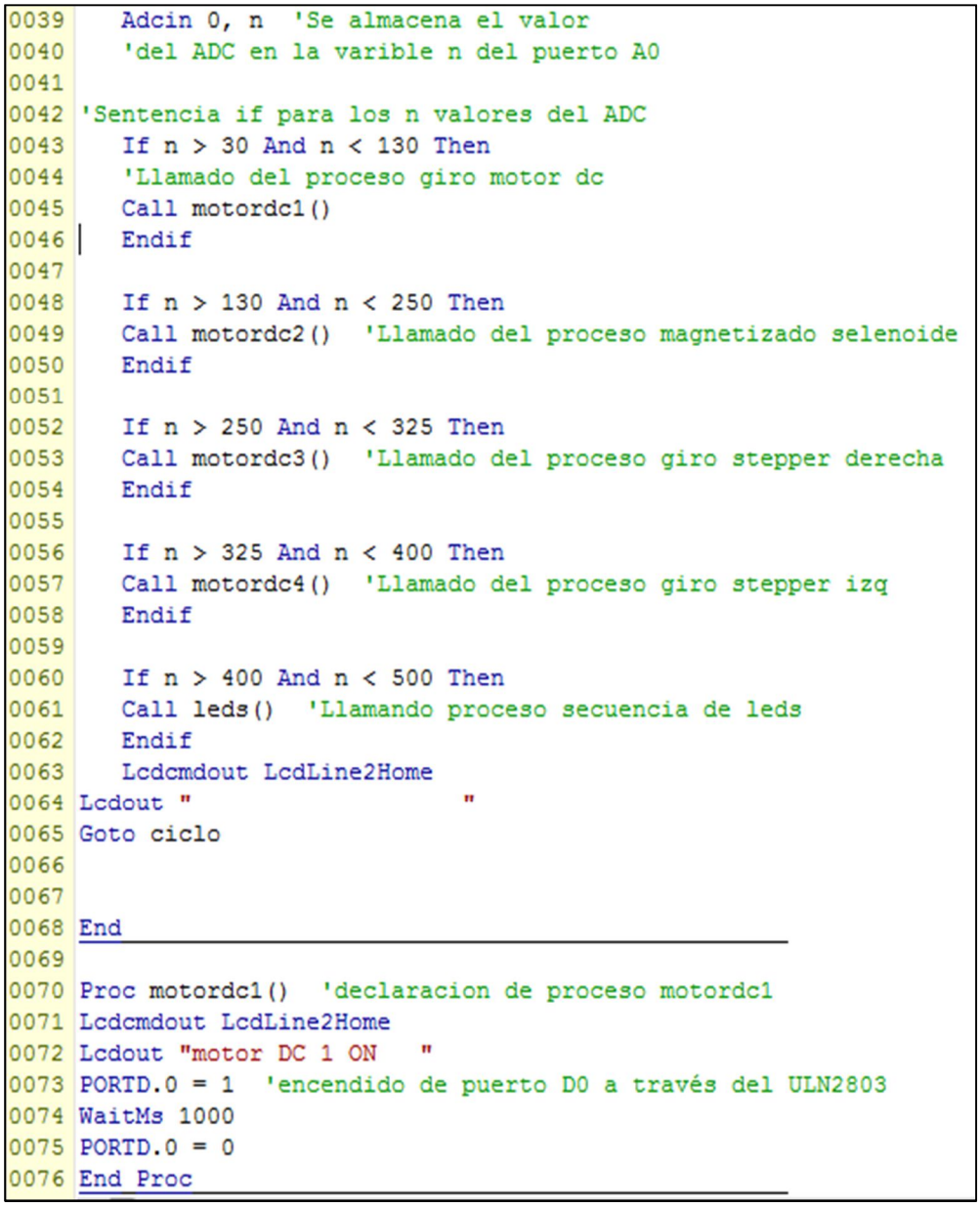
Figura 93. **4 motores + LEDs.bas 1/2**



```
0001 'configuracion lcd con puerto B
0002 Define LCD_BITS = 4
0003 Define LCD_DREG = PORTB
0004 Define LCD_DBIT = 4
0005 Define LCD_RSREG = PORTB
0006 Define LCD_RSBIT = 1
0007 Define LCD_EREG = PORTB
0008 Define LCD_EBIT = 3
0009 Define LCD_RWREG = PORTB
0010 Define LCD_RWBIT = 2
0011 Lcdinit 3
0012 'fin configuracion lcd
0013
0014
0015 TRISD = %00000000
0016
0017 'configuracion ADC
0018 Define ADC_CLOCK = 3
0019 Define ADC_SAMPLEUS = 50
0020 ADCON0 = 14
0021
0022 'variables
0023
0024 Dim n As Word
0025 'Dim an1 As Word
0026 PORTD = %00000000
0027 PORTC = %00000000
0028
0029 main:
0030
0031 menu:
0032 Lcdcmdout LcdCurOff
0033 Lcdcmdout LcdClear
0034 Lcdcmdout LcdLine1Home
0035 Lcdout " Motores + LEDS "
0036 Lcdcmdout LcdLine2Home
0037 Lcdout "M1-M2-M3-M4-LD5-"
0038 ciclo:
```

Fuente: elaboración propia.

Figura 94. **4 motores + LEDs.bas 2/3**



```
0039    Adcin 0, n  'Se almacena el valor
0040    'del ADC en la varible n del puerto A0
0041
0042 'Sentencia if para los n valores del ADC
0043    If n > 30 And n < 130 Then
0044    'Llamado del proceso giro motor dc
0045    Call motordc1()
0046    Endif
0047
0048    If n > 130 And n < 250 Then
0049    Call motordc2()  'Llamado del proceso magnetizado selenoide
0050    Endif
0051
0052    If n > 250 And n < 325 Then
0053    Call motordc3()  'Llamado del proceso giro stepper derecha
0054    Endif
0055
0056    If n > 325 And n < 400 Then
0057    Call motordc4()  'Llamado del proceso giro stepper izq
0058    Endif
0059
0060    If n > 400 And n < 500 Then
0061    Call leds()  'Llamando proceso secuencia de leds
0062    Endif
0063    Lcdcmdout LcdLine2Home
0064 Lcdout "                    "
0065 Goto ciclo
0066
0067
0068 End
0069
0070 Proc motordc1()  'declaracion de proceso motordc1
0071 Lcdcmdout LcdLine2Home
0072 Lcdout "motor DC 1 ON    "
0073 PORTD.0 = 1  'encendido de puerto D0 a través del ULN2803
0074 WaitMs 1000
0075 PORTD.0 = 0
0076 End Proc
```

Fuente: elaboración propia.

Figura 95. **4 motores + LEDs.bas 3/3**

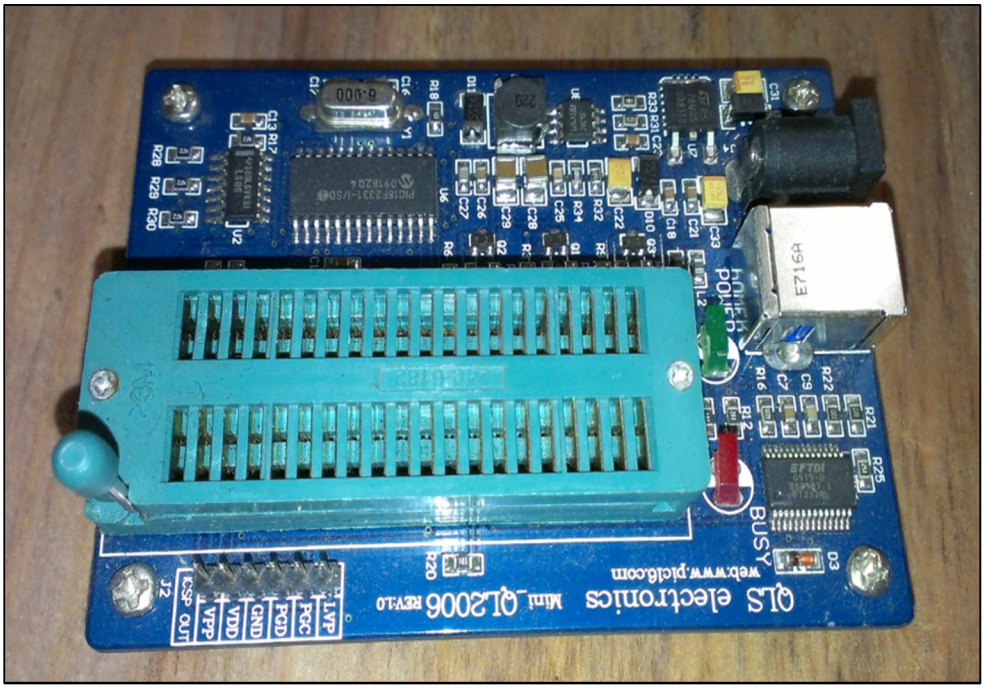
```
0077
0078 Proc motordc2()  'declaracion de proceso motordc2
0079 Lcdcmdout LcdLine2Home
0080 Lcdout "motor DC 2 ON   "
0081 PORTD.1 = 1  'encendido de puerto D1 a través del ULN2803
0082 WaitMs 1000
0083 PORTD.1 = 0
0084 End Proc
0085
0086 Proc motordc3()  'declaracion de proceso motordc3
0087 Lcdcmdout LcdLine2Home
0088 Lcdout "motor DC 3 ON   "
0089 PORTD.2 = 1  'encendido de puerto D2 a través del ULN2803
0090 WaitMs 1000
0091 PORTD.2 = 0
0092 End Proc
0093
0094 Proc motordc4()  'declaracion de proceso motordc4
0095 Lcdcmdout LcdLine2Home
0096 Lcdout "motor DC 4 ON   "
0097 PORTD.3 = 1  'encendido de puerto D3 a través del ULN2803
0098 WaitMs 1000
0099 PORTD.3 = 0
0100 End Proc
0101 Proc leds()
0102    Lcdcmdout LcdLine2Home
0103    Lcdout "leds encendidos "
0104 PORTD = %00010000  '= PORTD4 = 1 y el resto apagado
0105 WaitMs 100
0106 PORTD = %00100000  '= PORTD5 = 1 y el resto apagado
0107 WaitMs 100
0108 PORTD = %01000000  '= PORTD6 = 1 y el resto apagado
0109 WaitMs 100
0110 PORTD = %10000000  '= PORTD47= 1 y el resto apagado
0111 WaitMs 100
0112 PORTD = %00000000  '= todo el puerto D apagado
0113 WaitMs 100
0114 End Proc
```

Fuente: elaboración propia.

6. GRABACIÓN DE PROGRAMAS UTILIZANDO QUEMADORA DE PIC'S MINI QL2006

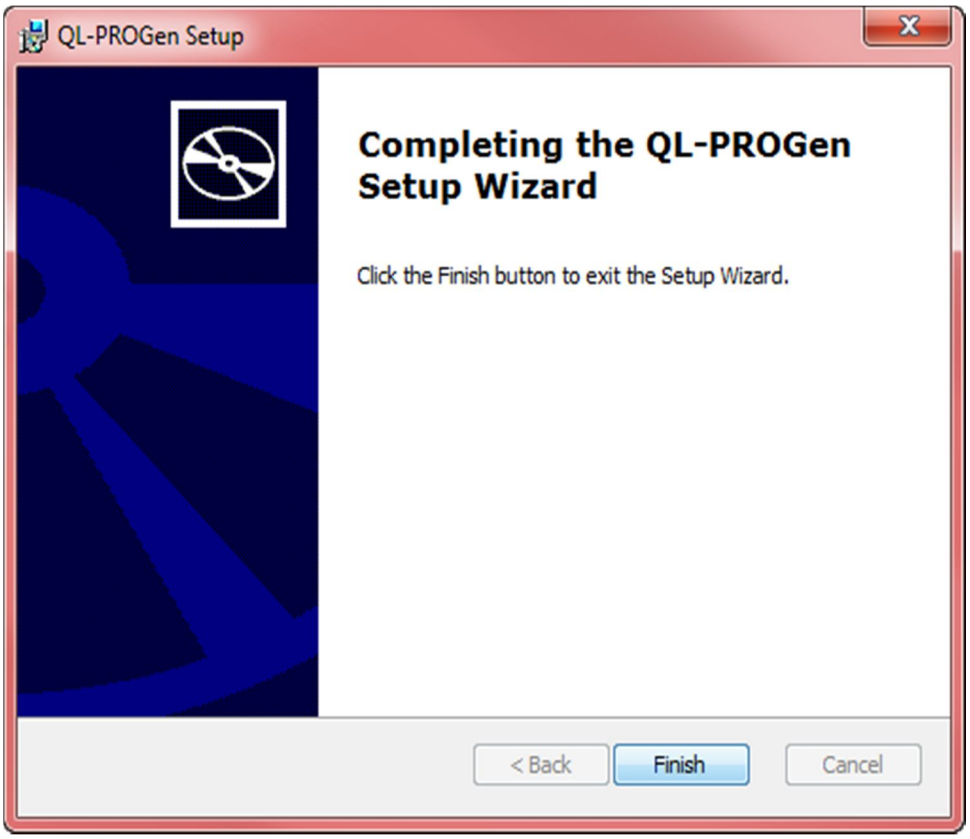
La programadora de PICS Mini QL2006 graba en lenguaje *assembler* en una gran cantidad de microcontroladores, incluyendo el PIC16F877A, a través del puerto USB de cualquier computadora, y soporta sistemas operativos desde Windows XP hasta Windows 7.

Figura 96. **Programadora de PICS**



Fuente: elaboración propia.

Figura 97. **Fin de la instalación**



Fuente: elaboración propia.

Una vez finalizada la instalación del programa, se procede a la aplicación de las versiones de controladores más avanzadas para los puertos USB que están adjuntos en las carpetas, los cuales soportan la tarjeta programadora.

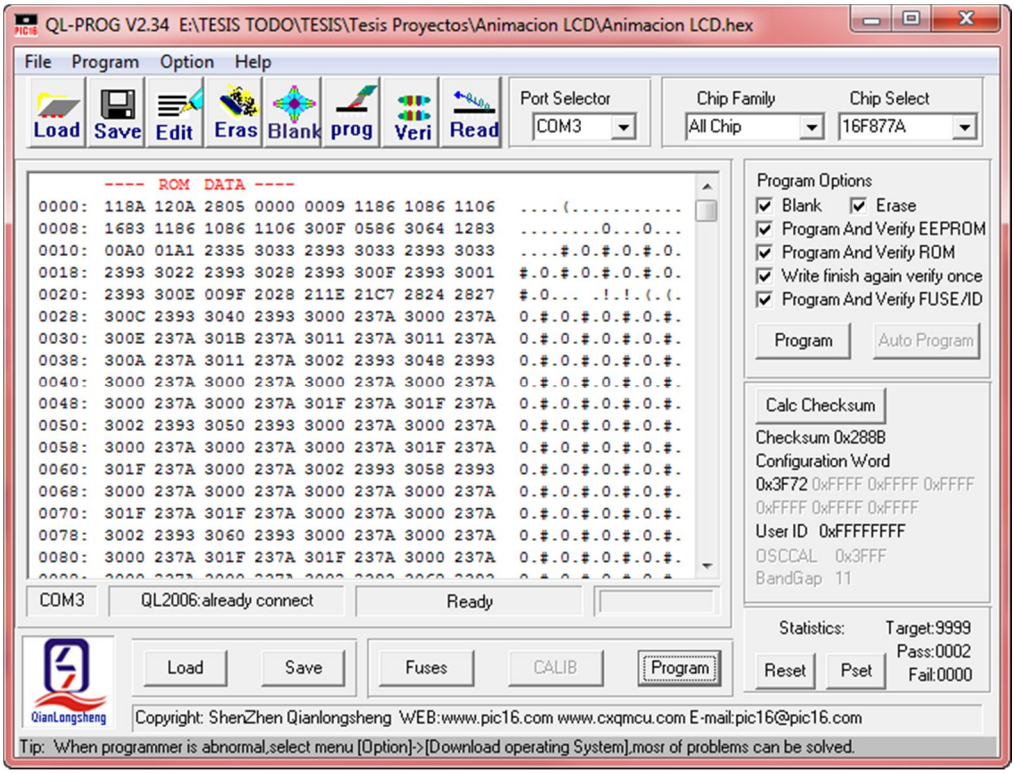
6.1. Menú principal de QL-PROG

En la primera vista del programa de grabación se encuentran todas las opciones que se pueden modificar, dependiendo del PIC, del cristal utilizado, del puerto conectado y las opciones avanzadas que se quieran modificar.

Se visualizan las opciones siguientes, para poder realizar la grabación:

- *Load:* para cargar el archivo hexadecimal del programa realizado.
- *Fuses:* aquí se puede cambiar el tipo de cristal que se está utilizando.
- *Program:* con este botón se da inicio a la grabación sobre el PIC ubicado sobre la placa.
- *Chip select:* despliega un menú con los PIC's soportados por el progrma.

Figura 98. **Primer vista del programa**

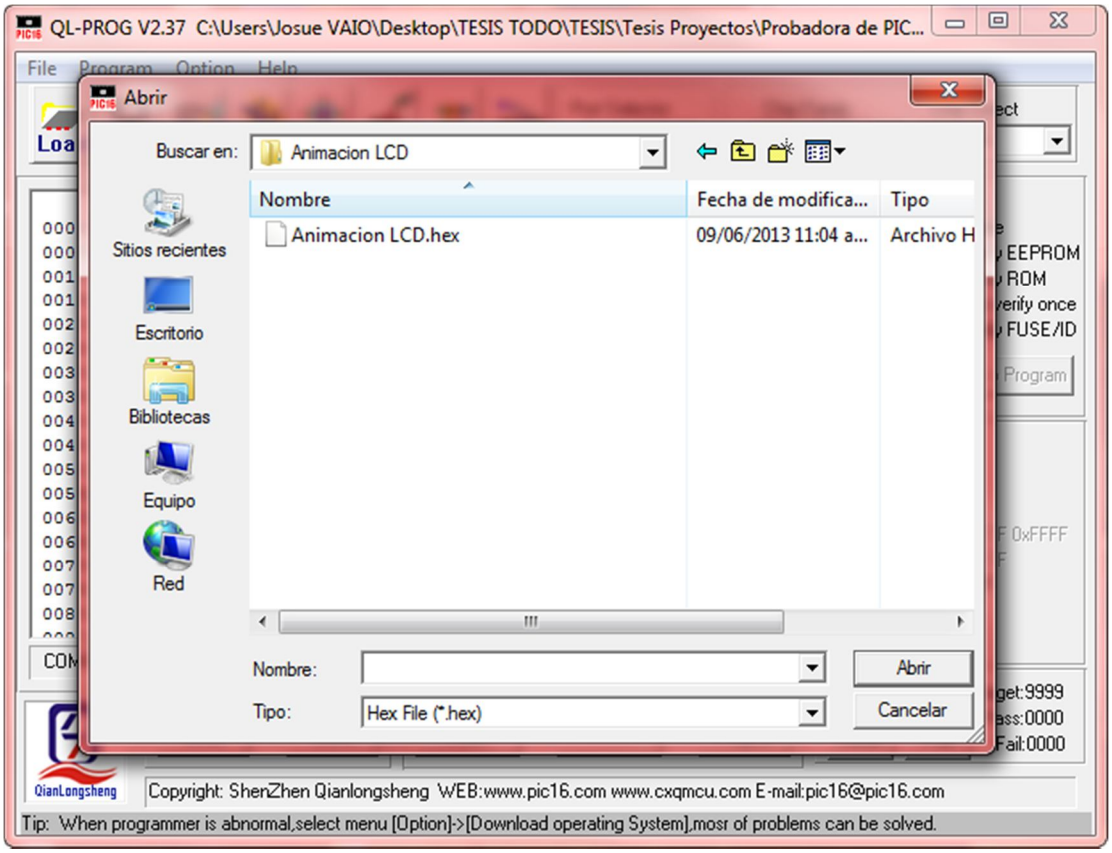


Fuente: elaboración propia.

6.2. Procedimiento para grabar el programa al PIC

Se debe cargar el programa hexagesimal al programa principal.

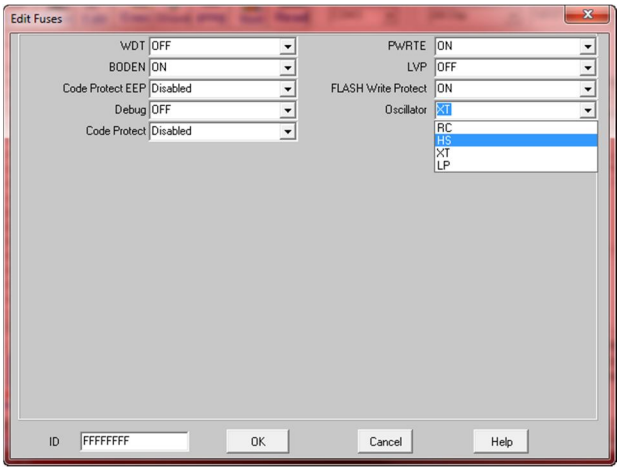
Figura 99. **Cargando programa desde *load***



Fuente: elaboración propia.

Se modifica la opción de cristal tipo “HS” en la ventana Fuses para realizar la grabación, si se usan cristales externos al PIC.

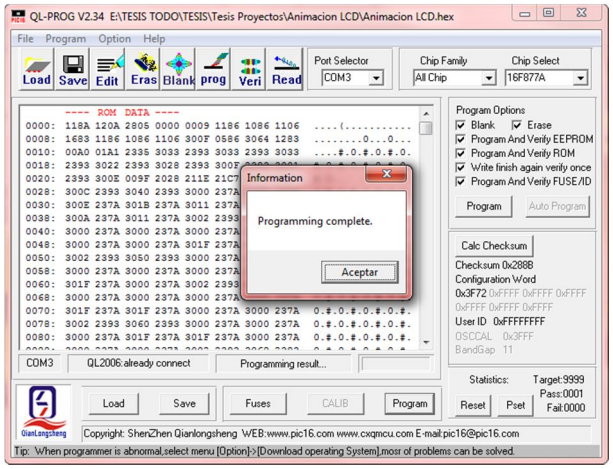
Figura 100. **Edit fuses**



Fuente: elaboración propia.

Presionar el botón “Program” y esperar a que finalice el proceso.

Figura 101. **Fin de la programación del PIC**



Fuente: elaboración propia.

CONCLUSIONES

1. Se propone una placa entrenadora diseñada para que el estudiante pueda realizar proyectos electrónicos de bajo costo y agilizar el aprendizaje de los microcontroladores PIC, con base en los conocimientos recopilados.

2. Una de las características más importantes de los microcontroladores programables PIC es que la cantidad de dispositivos que se pueden conectar a estos es sumamente amplia; las señales recibidas pueden transformarse en información necesaria, que puede ser interpretada por medio de programación. Esto le da utilidad a los circuitos integrados.

3. La placa entrenadora de microcontroladores propuesta, presenta la opción de ser mejorada y expandida, por lo que se adjunta adentro del disco compacto el archivo para PCB Wizard, para su edición e impresión; este diseño tiene una distribución de las terminales de los puertos del microcontrolador que tiene el orden de la misma forma que el circuito integrado PIC16F877/A, para una mejor interconexión de los dispositivos a la placa.

4. El método impresión de circuito es adecuado para realizar circuitos con pistas de cobre que están demasiado próximas unas con otras; por lo que cual se ha propuesto este procedimiento para la creación de la placa JP11, debido a que es un proceso que permite trasladar por completo una impresión computarizada a la placa de cobre, a través del acetato impreso a tóner.

RECOMENDACIONES

1. Verificar que el circuito integrado ULN2803 sea polarizado adecuadamente, ya que es alimentado por 12 V; esto puede provocar que se sobrecaliente y estalle, aumentando el costo final del proyecto.

2. Evitar uno de los problemas más frecuentes y sencillos que hay cuando se interconectan circuitos adicionales al PIC; hay que conectar todas las tierras en común para cerrar el circuito y que la corriente transcurra por cada dispositivo.

3. El método de impresión del circuito impreso a través de acetato puede complicarse cuando no se genera la temperatura adecuada, porque esto puede derretir y deformar el acetato cuando la temperatura es demasiado alta o no ser lo suficientemente caliente para que el tóner se desprenda y se impregne en la placa de cobre; por lo que hay que realizar varias pruebas previas a la impresión final, ya que otra causa de malas impresiones es utilizar planchas de vapor, los agujeros en la plancha de metal pueden hacer que el calor no se transmita de forma uniforme al acetato y la placa de cobre, provocando que no todas las pistas sean impresas.

4. El software con el que se ha diseñado la placa entrenadora y programado los proyectos de los ejemplo, han sido seleccionados con fines didácticos por su sencillez y fácil comprensión; pero es indispensable explorar otras alternativas que contienen más aplicaciones que permiten complementar aún más el campo de los microcontroladores

PIC, como los programas de diseño de circuitos impresos producidos por National Instrument que son Ultiboard y Multisim u otros programas adicionales a PIC Simulator IDE para programación de PICs, como el desarrollado por la empresa microelectrónica llamada PICBasic, el cual ha sido creado para la placa de pruebas de esa misma empresa, pero tiene comandos más especializados para el uso de interrupciones y librerías más completas necesarias, dependiendo del proyecto a realizar.

BIBLIOGRAFÍA

1. Biblioteca de circuitos y proyectos. *Simple machines forum Ucontrol.* [en línea]. <http://www.ucontrol.com.ar/forosmf /circuiteca-labibliotecade-circuitos-y-proyectos-de-ucontrol/>. [Consulta: mayo de 2013].

2. Biblioteca de códigos de programación, Argentina. *Lenguaje de programación PICBasic.* [en línea] <http://www.todopic.com.ar /foros/index. php?topic=14917.0>. [Consulta: enero de 2013].

3. Blog de temas de ingeniería, Colombia. *Ingeniero Forigua, diseño de circuitos impresos.* [en línea]. <http://ingeniero-forigua .blogspot .com /2009/02/diseno-de-circuitos-impresos-mediante. html>. [Consulta: enero de 2013].

4. *Data sheet* UNL2803-04. *Catálogo de hojas de datos*. [en línea]. <http://www.datasheetcatalog.com/datasheets_pdf/U/L/N/2/ULN2803.shtml>.[Consulta: marzo de 2013].

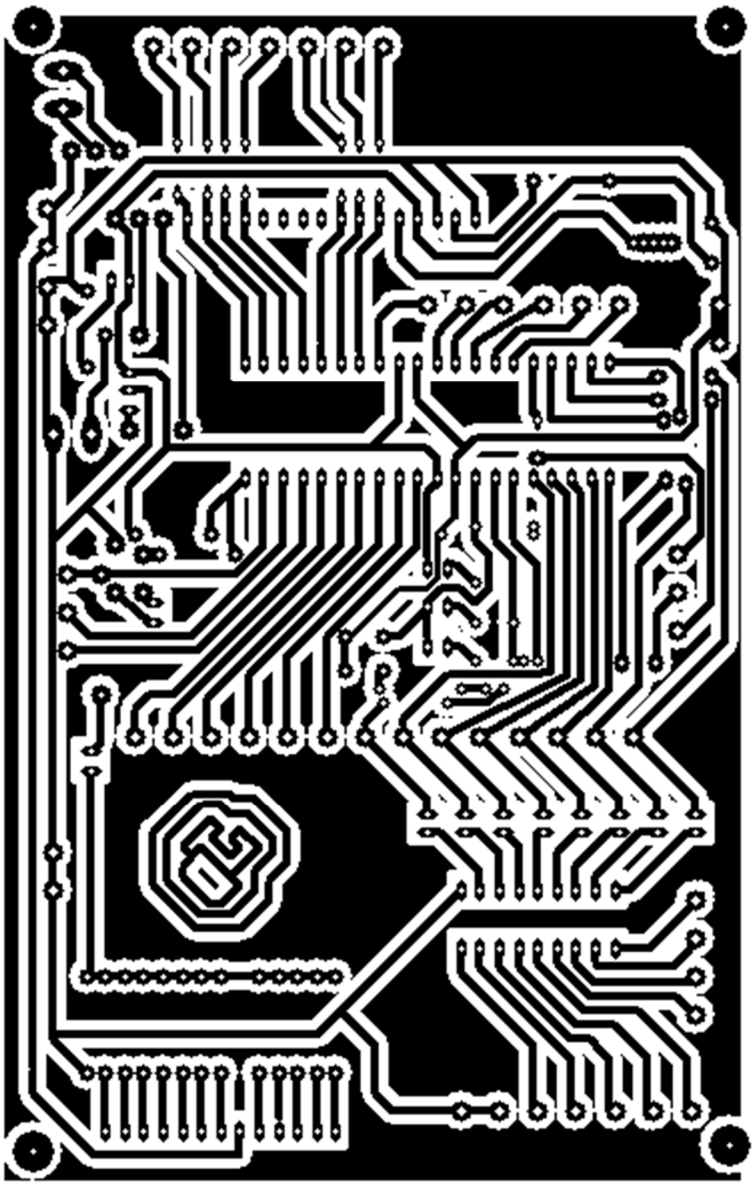
5. Ecu Red Cristal de cuarzo, Cuba. *Enciclopedia sobre temas de electrónica y conocimiento ingenieril.* [en línea]. <http://www.ecured.cu/ index.php/Cristal _de_cuarzo>. [Consulta: diciembre de 2013].

APÉNDICES

Apéndice 1. **Impresión en acetato de la placa JP11**

Indicaciones: en la siguiente página está la impresión en papel del circuito PCB de la placa entrenadora de PICS; cuando se imprima la fotocopia debe ser al 100% sin ajuste de página con la impresión de tóner más elevado.

Apendice 1a. Placa JP11

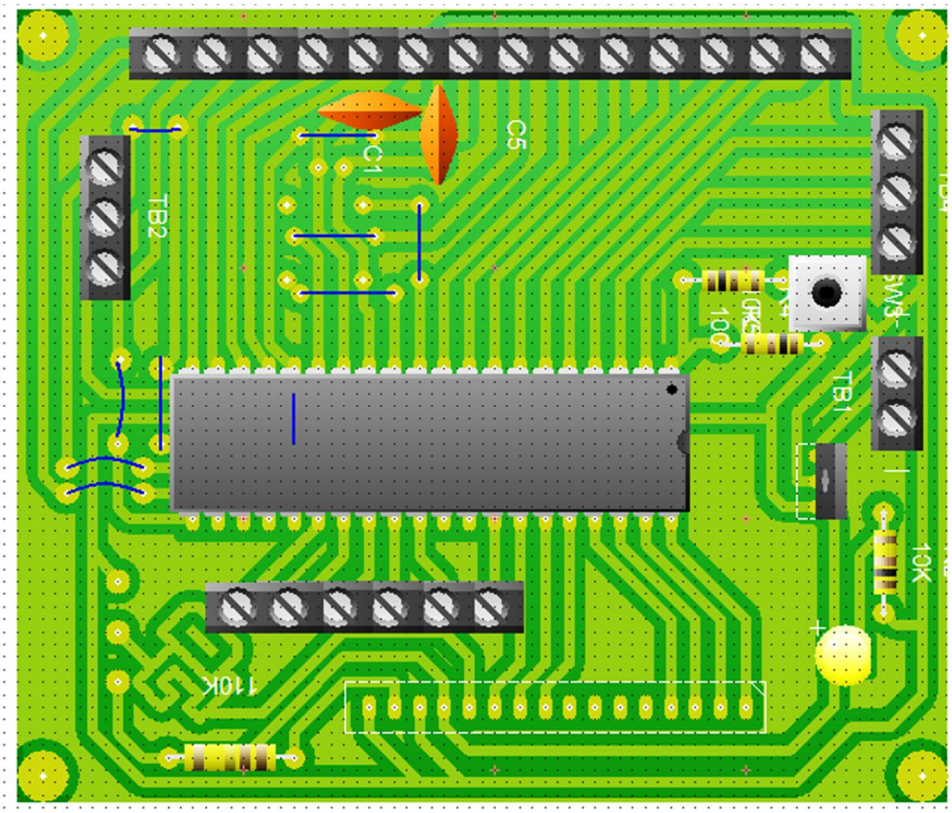


Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Apéndice 2. **Versión corta de la placa entrenadora**

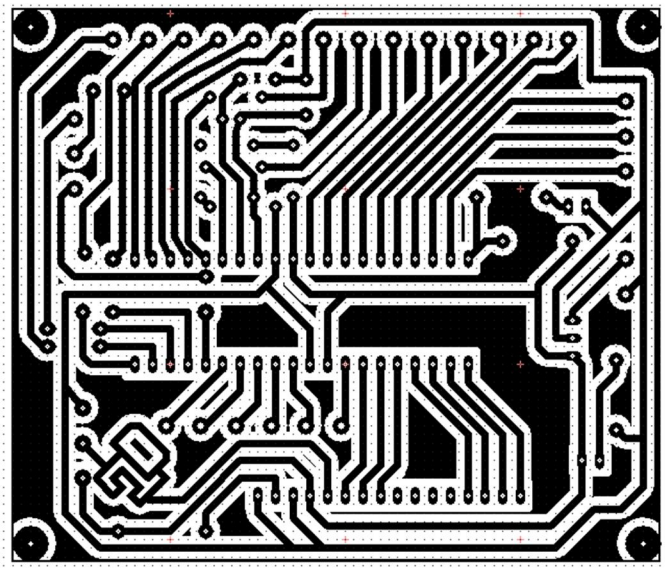
Vista realista de la placa corta entrenadora de PICs, los archivos están incluidos en el disco de tesis.

Apéndice 3a. **Vista normal**



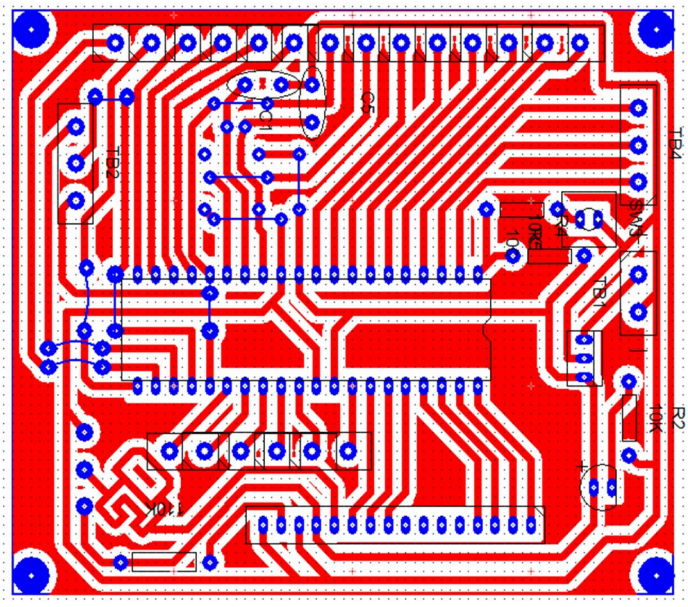
Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

Apéndice 2b. **Vista pcb**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

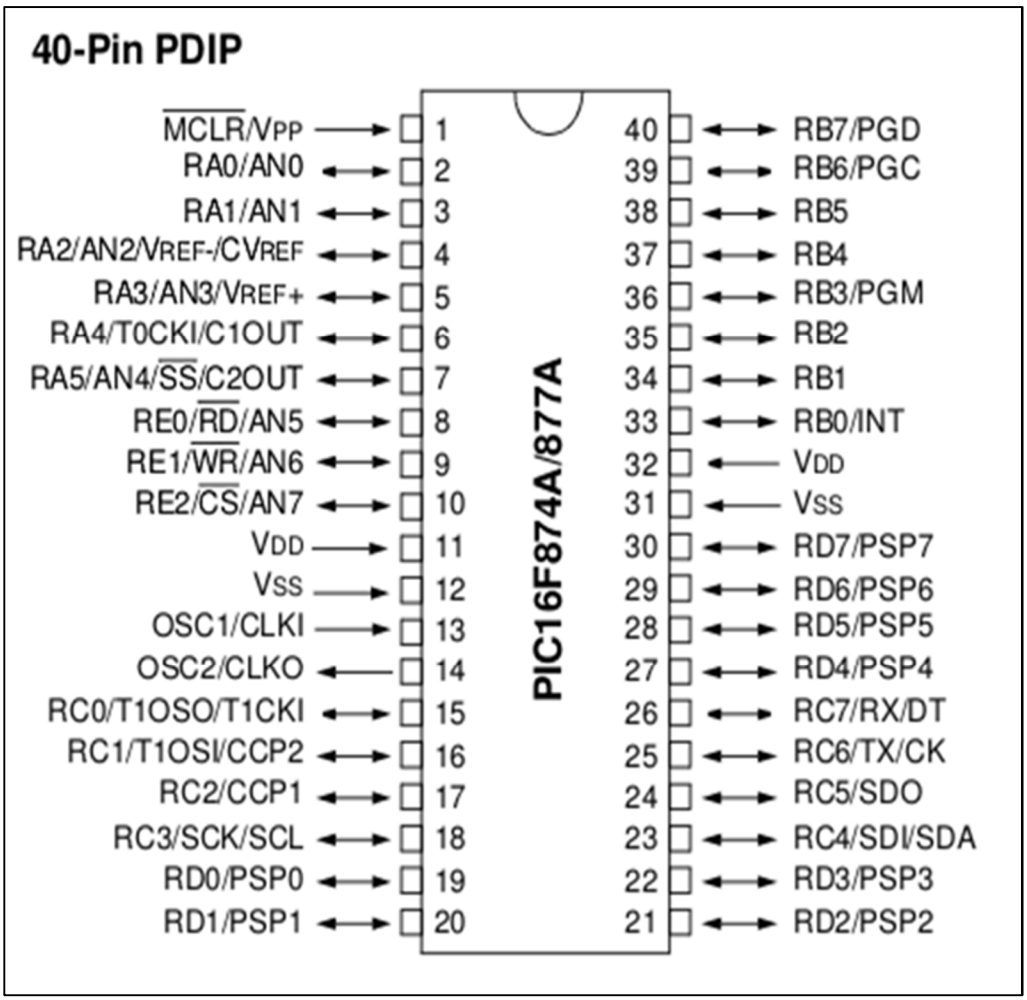
Apéndice 2c. **Vista normal**



Fuente: elaboración propia, con programa de diseño de tarjetas impresas PCB Wizard.

ANEXO

Anexo 1. **PIC16F877A**



Fuente: http://www.futurlec.com/Pictures/PIC16F874A-877A.gif.

Consulta: diciembre de 2,012.