Universidad Veracruzana

Facultad de ingeniería Mecánica Eléctrica,

Región Orizaba-Córdoba.

Manual del taller:

INTRODUCCIÓN AL MICROCONTROLADOR PIC18F4550

Duración:

30 horas

Facilitador:

M.C. Jesús Medina Cervantes

Ciudad Mendoza, Veracruz, México.

Junio, 2012

Introducción

La **Universidad Veracruzana** ha impulsado de forma institucional la transición a un **Modelo Educativo Integral y Flexible**, el cual está centrado en el aprendizaje de los estudiantes y orientado al desarrollo de competencias profesionales. Para hacer realidad estos cambios, es evidente la necesidad de transformar de los procesos de aprendizaje, lo que a su vez demanda de cambios compartidos y de manera sistemática por parte de los profesores para conseguir la transformación de las prácticas docentes en el aula.

Para consolidar el modelo educativo de manera que impacte de forma efectiva en el aprendizaje y formación integral de los estudiantes, la Universidad Veracruzana ha establecido un programa de **Innovación educativa**, cuya finalidad es lograr que todos los académicos de la Institución adopten una metodología de enseñanza apoyada en cuatro componentes, que son: pensamiento complejo, competencias, vinculación docencia-investigación y el uso de tecnologías de información y comunicación. Así, resulta evidente que los académicos de la Universidad Veracruzana deberán mejorar sus perfiles profesionales, así como permanecer en constante actualización pedagógica y disciplinar.

Por esta razón, los miembros del cuerpo académico UV CA-318: Investigación en Ingeniería Aplicada, perteneciente a la **Facultad de Ingeniería Mecánica Eléctrica**, han iniciado un programa que pretende atender la necesidad de actualización disciplinar de los académicos de la Facultad, en las distintas áreas de ingeniería de los PE que oferta. Por ello, se han planeado cursos o talleres de actualización disciplinar impartidos por los miembros del cuerpo académico, o en su caso, realizar la gestión pertinente para que los impartan académicos de otros cuerpos académicos, de la UV o de diferentes instituciones.

Desde el año 2010, el cuerpo académico ha impartido hasta el momento tres talleres dirigidos a los académicos de la Facultad, que son: **Introducción a Mechanical Inventor**, **Introducción a SolidWorks** y **Programación básica con Matlab**. Los primeros dos abordan aspectos de Diseño asistido por computadora para generar modelos tridimensionales de elementos mecánicos y el tercero se enfoca a la programación en software especializado para cálculos de ingeniería.

Por otro lado, en colaboración con otro cuerpo académico de la UV, se impartió el taller de **Interacción con el mundo virtual con 3Ds Max**, el cual se orientó a la creación de modelos tridimensionales elaborados con el software 3Ds Max y manipulados en un ambiente virtual mediante la aplicación vrbuild2 de Matlab.

Los talleres han permitido a los académicos de la Facultad conocer y manipular software de actualidad especializado en ingeniería. De esta manera, los académicos pueden visualizar diferentes usos y aplicaciones del software, conforme los incorporen en el quehacer docente, en el desarrollo de proyectos y en la investigación.

Ahora, el taller que se desarrollará en este periodo intersemestral se denomina **Introducción al microcontrolador PIC18F4550**.

Objetivo

Que los académicos adquieran competencias básicas en el manejo del microcontrolador PIC18F4550, así como su programación e implementación en proyectos de automatización y control, con la finalidad de que incorporen su uso como herramienta didáctica de apoyo y/o para la dirección de proyectos que promuevan el desarrollo de dispositivos experimentales electromecánicos y mecatrónicos de bajo costo para fortalecer el equipamiento de los laboratorios de la facultad.

Justificación

Se pretende que los docentes conozcan y se interesen en gran medida en el manejo y programación del microcontrolador PIC18F4550, con la finalidad de que ellos propongan y desarrollen en la Facultad proyectos para la creación de dispositivos o prototipos que se requieren para la realización de prácticas de laboratorio de las diferentes experiencias educativas prácticas que se imparten en los programas educativos de la Facultad. Con ello, se podrán reducir reducir las necesidades de equipamiento de laboratorio, en tanto que el desarrollo de equipos experimentales propios dejará un

mayor aprendizaje y adquisición de competencias tanto en los estudiantes como en los docentes.
Por otro lado, se pretende también que los académicos que imparten clases en la carrera de Mecatrónica estén familiarizados con el uso de esta tecnología tan ampliamente utilizada en la actualidad en dicho campo disciplinar, para que puedan colaborar y acompañar de mejor manera a los estudiantes en su proceso de aprendizaje.

Intención del manual

El presente manual ha sido elaborado para servir como apoyo didáctico complementario al desarrollo de las actividades del taller **Introducción al microcontrolador PIC18F4550**. Se trata de un manual básico cuya finalidad es fungir como material de consulta para la realización de los proyectos propuestos en el taller.

Se pretende que el manual sea práctico y de fácil lectura, para que el lector desarrolle con facilidad los proyectos presentados. Por otro lado, es necesario decir que este manual no pretende ser exhaustivo, por la naturaleza y cantidad de horas dedicadas para el desarrollo del taller.

Si el lector tiene algunos comentarios para mejorar este documento o ha encontrado algún error en su contenido, por favor hágalo saber al autor a la siguiente dirección electrónica: jemedina@uv.mx

Licencia

Contenido

Lista de figuras

1.INTRODUCCIÓN A LOS MICROCONTROLADORES

Los microntroladores surgieron a partir de una necesidad concreta que tuvieron ingenieros japoneses de la empresa Busicom en 1969. El requerimiento fue obtener una cantidad de circuitos integrados para calculadoras que fueran diseñados de acuerdo a sus proyectos. Estos requerimientos fueron expuestos a la empresa Intel. La solución propuesta por el jefe del proyecto, Marcian Hoff, requería que la operación del circuito integrado fuera determinada por un programa almacenado en el propio circuito (Milan, 2008). De esta manera nació el primer microcontrolador, el cual ha evolucionado rápidamente hasta nuestros días.

1.1. ¿Qué son los microcontroladores?

Los microcontroladores son circuitos integrados programables que contienen todos los elementos necesarios para desarrollar y controlar una tarea determinada. La cantidad de componentes que se integran a los microcontroladores depende del diseño de los fabricantes, sin embargo, los elementos básicos suelen ser: microprocesador, memoria RAM, memoria de programa, convertidor A/D, oscilador, puerto de comunicación, etc. Esto le ha brindado una gran versatilidad a este tipo de dispositivos y hoy en día su utilización se ha incrementado enormemente en el mundo (Palacios, Remiro, & López, 2004).

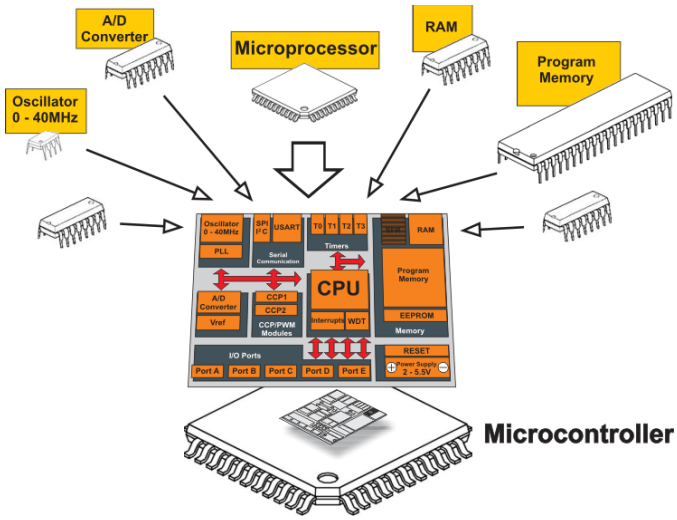


Figura 1. Componentes de los microcontroladores (Milan, 2008).

Cuando se desea trabajar en sistemas basados en microcontroladores es necesario realizar algunas consideraciones como: la cantidad de entradas/salidas que son necesarias para su operación, si se requiere convertidor A/D, si realizará algunas operaciones distintas de encender o apagar un relevador, si se requiere algún módulo especializado de comunicación, etc. Una vez que se han respondido estas preguntas básicas, se está en condiciones de elegir un microcontrolador. Sin embargo, esto no es una tarea fácil, ya que actualmente existen un buen número de marcas y gamas de microcontroladores, como son: AVR de Atmel, Freescale (antes Motorola), Basic Stamp de Parallax y PICmicro de Microchip, entre muchos otros.

De estos, los microcontroladores PICmicro de Microchip son considerados como los idóneos y más aceptados tanto para principiantes aficionados como para buena parte de profesionales. Uno de los factores principales del éxito de Microchip es que tienen la política de ofrecer la documentación y todo el software necesario sin ningún costo para el usuario; el cual puede obtener la información a partir de su página web www.microchip.com

Las ventajas que ofrecen los microcontroladores PIC son (Casacuberta, 2010):

- Existe gran variedad de familias
- Poseen herramientas de desarrollo comunes
- Existe una gran cantidad de unidades funcionales embebidas (temporizadores, USART, convertidores A/D, USB, RF, Ethernet, etc.)
- Precios competitivos
- Amplio soporte (hojas de datos, libros, información disponible en internet)

1.2. Arquitectura interna

Los microcontroladores PIC utilizan la arquitectura Harvard, lo cual significa que disponen de dos memorias independientes, una para el programa y la otra para los datos, cada una con sus respectivos buses. Esto le brinda a los microcontroladores la posibilidad de de tener acceso simultáneo a ambas memorias, así como solapar operaciones para mejorar el rendimiento (Salamanca, 2003).

Además, los microcontroladores PIC cuentan con la tecnología RISC, por lo que poseen un número reducido de instrucciones y solamente las instrucciones de carga y almacenamiento tienen acceso a la memoria de datos. Su objetivo principal hacer posible la segmentación y el paralelismo en la ejecución de instrucciones y reducir los accesos a memoria.

1.3. Hoja de datos del PIC18F4550

Los microcontroladores PIC existen en gamas de 8-bit, 16-bit y 32-bit. Dentro de la gama más simple de 8-bit se encuentra el microntrolador PIC18F4550, el cual pertenece a la familia PIC18 MCU. Sus características de memoria de programa, memoria RAM, número de entradas/salidas, número de canales analógicos y tipos de puertos de comunicación, han hecho de este PIC uno de los más utilizados para diversas aplicaciones.

Microchip ofrece la hoja de datos (data sheet) de todos sus microcontroladores de forma gratuita, las cuales se pueden descargar directamente desde su página web.

Enseguida, en la imagen 2 se presenta la descripción de pines del microcontrolador PIC18F4550.

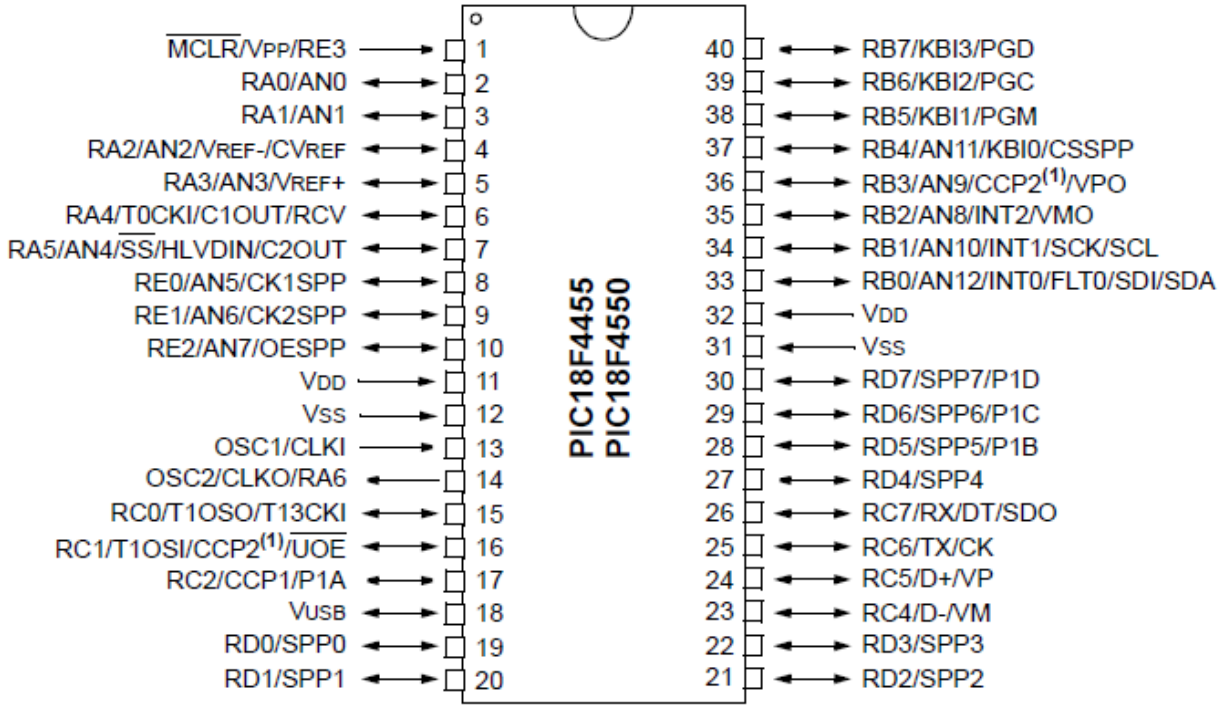


Figura 2. Esquema de pines del PIC18F4550 (Microchip, 2009).

Tabla 1. Características de la familia PIC18 (Microchip, 2009).

Features	PIC18F2455	PIC18F2550	PIC18F4455	PIC18F4550
Operating Frequency	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
Program Memory (Bytes)	24576	32768	24576	32768
Program Memory (Instructions)	12288	16384	12288	16384
Data Memory (Bytes)	2048	2048	2048	2048
Data EEPROM Memory (Bytes)	256	256	256	256
Interrupt Sources	19	19	20	20
I/O Ports	Ports A, B, C, (E)	Ports A, B, C, (E)	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers	4	4	4	4
Capture/Compare/PWM Modules	2	2	1	1
Enhanced Capture/ Compare/PWM Modules	0	0	1	1
Serial Communications	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART
Universal Serial Bus (USB) Module	1	1	1	1
Streaming Parallel Port (SPP)	No	No	Yes	Yes
10-Bit Analog-to-Digital Module	10 Input Channels	10 Input Channels	13 Input Channels	13 Input Channels
Comparators	2	2	2	2
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled
Packages	28-Pin PDIP 28-Pin SOIC	28-Pin PDIP 28-Pin SOIC	40-Pin PDIP 44-Pin QFN 44-Pin TQFP	40-Pin PDIP 44-Pin QFN 44-Pin TQFP

En la tabla 1 se presentan las características principales de la familia PIC18. En particular, se pueden observar en la última columna de la derecha las características del PIC18F4550. Este microcontrolador cuenta con 5 puertos E/S, 4 temporizadores, 20 fuentes de interrupción, comunicación serial, módulo USB, 13 canales de entradas analógicas y 2 módulos PWM.

2. COMPONENTES ELECTRÓNICOS EXTERNOS

En la realización de proyectos que involucran el control de dispositivos mediante microcontroladores siempre será necesaria la utilización de diferentes componentes electrónicos que son externos al PIC y que son necesarios para el correcto funcionamiento del proyecto realizado.

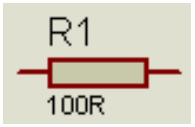
2.1. Fuente de alimentación

El microcontrolador PIC18F4550 se alimenta con 5V DC, los cuales se aplican entre los pines de alimentación V_{DD} (+) y V_{SS} (-). El consumo de corriente del dispositivo depende de las cargas conectadas al microcontrolador y de su frecuencia de trabajo. Se aconseja que se coloque un capacitor de desacoplo de 100nF lo más cerca posible de los pines de alimentación (Palacios, Remiro, & López, 2004). Una fuente DC de 5V y por lo menos 700mA será suficiente para el desarrollo de todos los proyectos presentados en este manual.

2.2. Resistor

Los resistores son elementos electrónicos que se utilizan para limitar la corriente eléctrica que fluye en un dispositivo. Son elementos compuestos de carbón y otros componentes resistivos. La corriente máxima en un resistor está limitada por la potencia máxima que puede disipar su cuerpo. Los valores más comunes encontrados son 0.25W, 0.5W y 1W. En la figura 3 se muestran ejemplos de resistores y su símbolo electrónico.

a)



b)

Figura 3. a) Resistores de 0.5W de diferentes valores, b) símbolo electrónico.

Los resistores poseen un código de colores mediante el cual se puede leer con facilidad su valor. Existen resistores con códigos de 4, 5 y 6 bandas de colores, tal como se muestra en la figura 4.

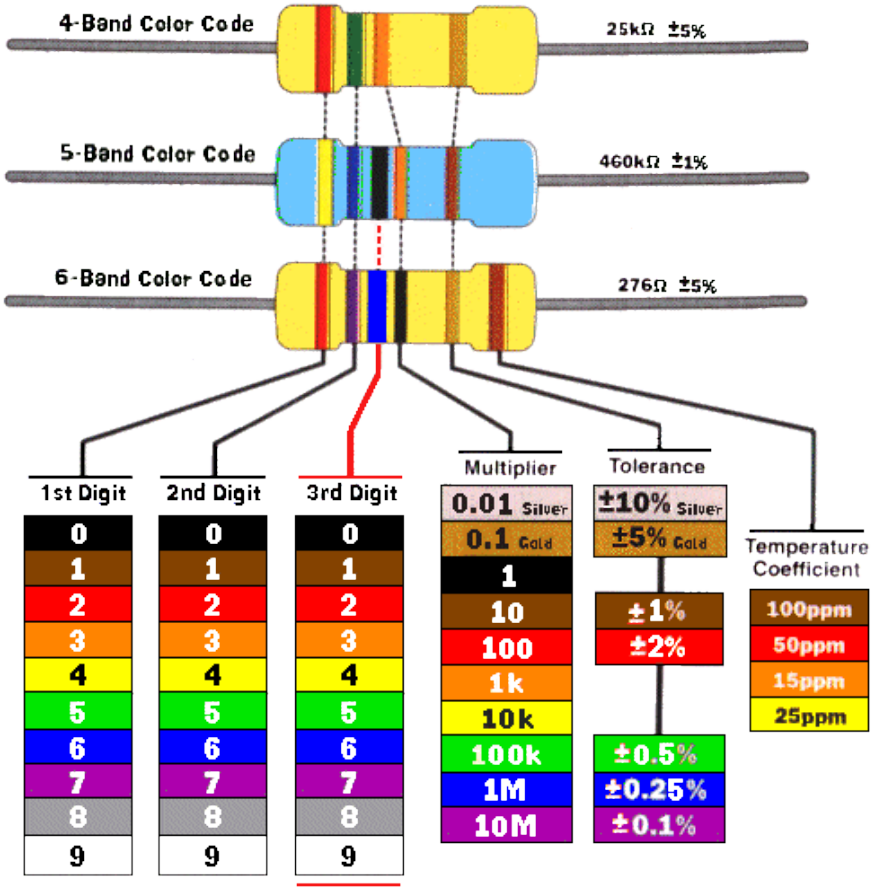
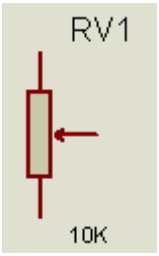


Figura 4. Código de colores para resistores de 4, 5 y 6 bandas (EDJMM9, 2011).

2.3. Potenciómetro

Los potenciómetros son resistores con valor de resistencia eléctrica variable. De esta forma se puede controlar indirectamente la intensidad de corriente eléctrica que fluye por un circuito si se conecta en paralelo, o la diferencia de potencial eléctrico si se conecta en serie. En la figura 5 se muestra un ejemplo de potenciómetros típicos y su símbolo electrónico. En este manual se hará uso de un potenciómetro para realizar la simulación de la señal de entrada de un sensor analógico.

a)

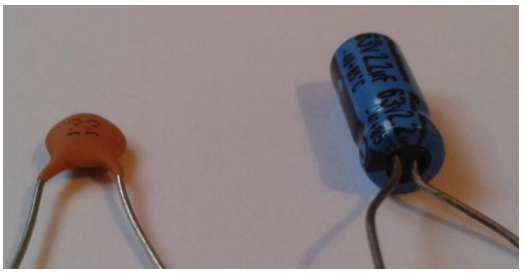


b)

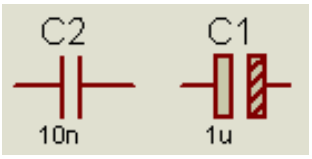
Figura 5. a) Potenciómetros de 10kΩ, b) símbolo electrónico.

2.4. Capacitor

Es un dispositivo pasivo capaz de almacenar energía en un campo eléctrico entre dos superficies o placas conductoras, separadas por un material dieléctrico o por el vacío. Los capacitores se utilizan en baterías, memorias, filtros, adaptación de impedancias, para flash en cámaras fotográficas, para mantener corriente en un circuito y evitar las caídas de tensión. Los capacitores se realizan de diferentes materiales, como: vidrio, mica, papel, cerámicos, tantalio, electrolíticos, etc. En la figura 6 se muestran ejemplos de capacitores y su símbolo electrónico.



a)

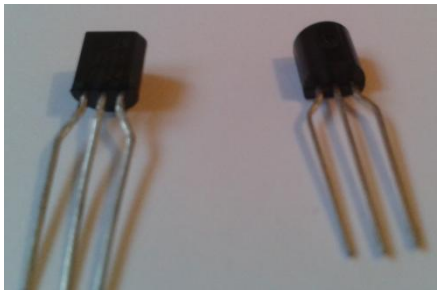


b)

Figura 6. a) capacitor cerámico y electrolítico, respectivamente, b) símbolos electrónicos.

2.5. Transistor

Son dispositivos electrónicos semiconductores que cumplen alguna función como: amplificadores, osciladores, conmutadores o rectificadores. Estos se encuentran en los aparatos electrónicos como radios, televisores, reproductores de audio y video, relojes de cuarzo, computadoras, teléfonos celulares, etc. En la figura 7 se muestran ejemplos de transistores y su símbolo electrónico.



a)

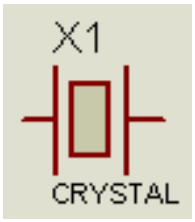
b)

Figura 7. a) Transistores, b) símbolo electrónico.

2.6. Oscilador

Los microcontroladores siempre requieren de un circuito que les indique la velocidad de trabajo. A estos circuitos se les llaman osciladores o relojes. Los osciladores generan ondas cuadradas de alta frecuencia. Existen diferentes tipos de osciladores que se pueden utilizar en los microcontroladores, como son: XT (cristal de cuarzo), RC (oscilador con resistor y capacitor), HS (cristal de alta velocidad), LP (cristal para baja frecuencia y bajo consumo de potencia) y EXTERNO (se aplica una señal de reloj externa). El tipo de oscilador que se utilizará en los proyectos presentados en el presente manual es el tipo HS (high speed resonator) de 20MHz. En la figura 8 se muestra el oscilador y su símbolo electrónico.

a)



b)

Figura 8. a) Oscilador HS de 20MHz, b) símbolo electrónico.

2.7. Botón pulsador

Los botones pulsadores permiten introducir señales digitales (niveles lógicos “0” ó “1”) hacia los microcontroladores. Es posible establecer dos arreglos diferentes para establecer el nivel nivel lógico que se desea introducir de manera inicial (cuando aún no se ha presionado el botón pulsador). Los dos tipos de arreglos se muestran en la figura 9.

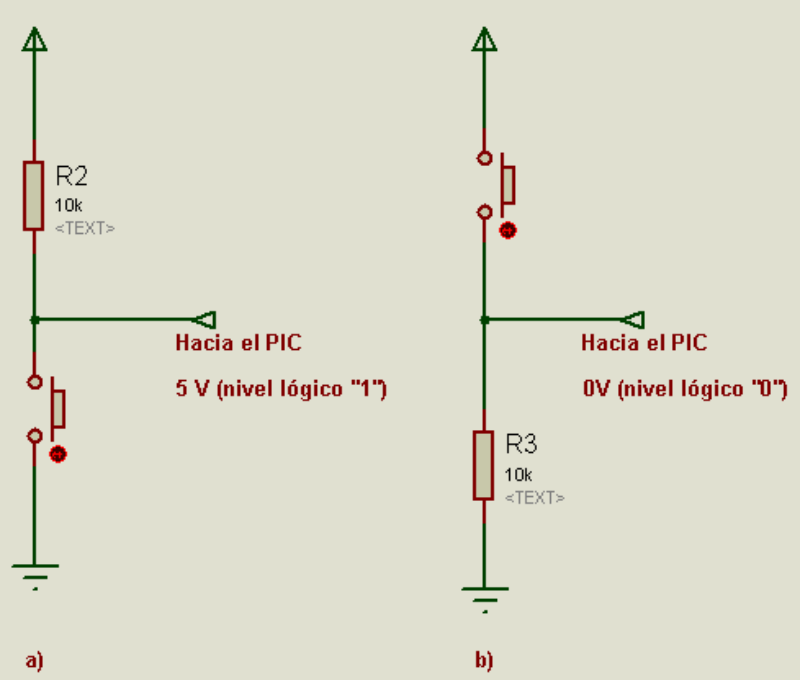


Figura 9. a) Entrada inicial con nivel lógico “1”, b) Entrada inicial con nivel lógico “0”.

2.8. Diodo

Un diodo es un componente semiconductor electrónico de dos terminales, el cual permite la circulación de corriente eléctrica solamente en un sentido. La curva característica de un diodo (I-V) consta de dos regiones, la primera por debajo de cierta diferencia de potencial, en la cual se comporta como un circuito abierto y por lo tanto no conduce. En la otra, por encima de una cierta diferencia de potencial se comporta como un circuito cerrado con una resistencia eléctrica mínima. En la figura 10 se muestra un ejemplo de un diodo y su símbolo electrónico.

a)

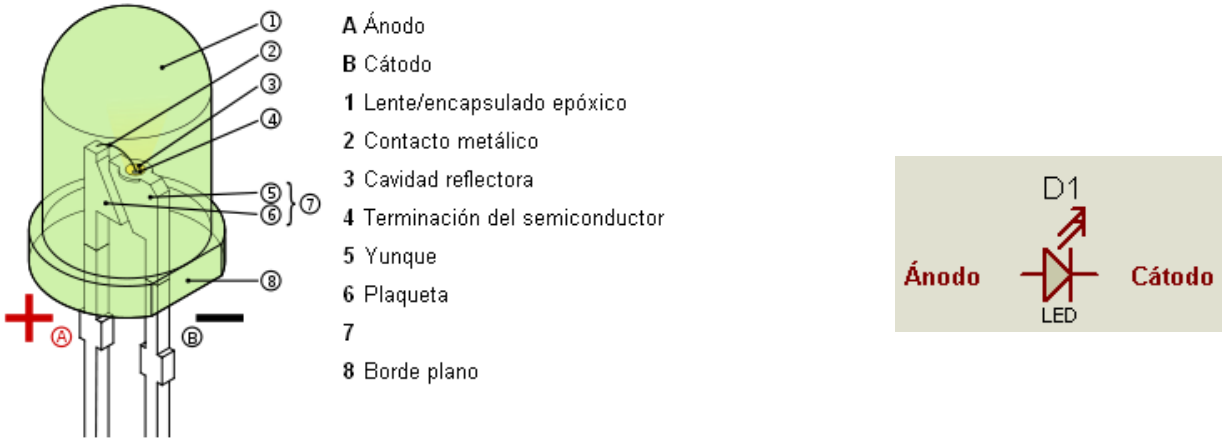
b)

Figura 10. a) Diodo, b) símbolo electrónico.

2.9. Led

Un Led (Light emitting diode) es un diodo semiconductor que emite luz. Los diodos se utilizan como indicadores en diferentes dispositivos electrónicos. Los diodos se polarizan en directo con una

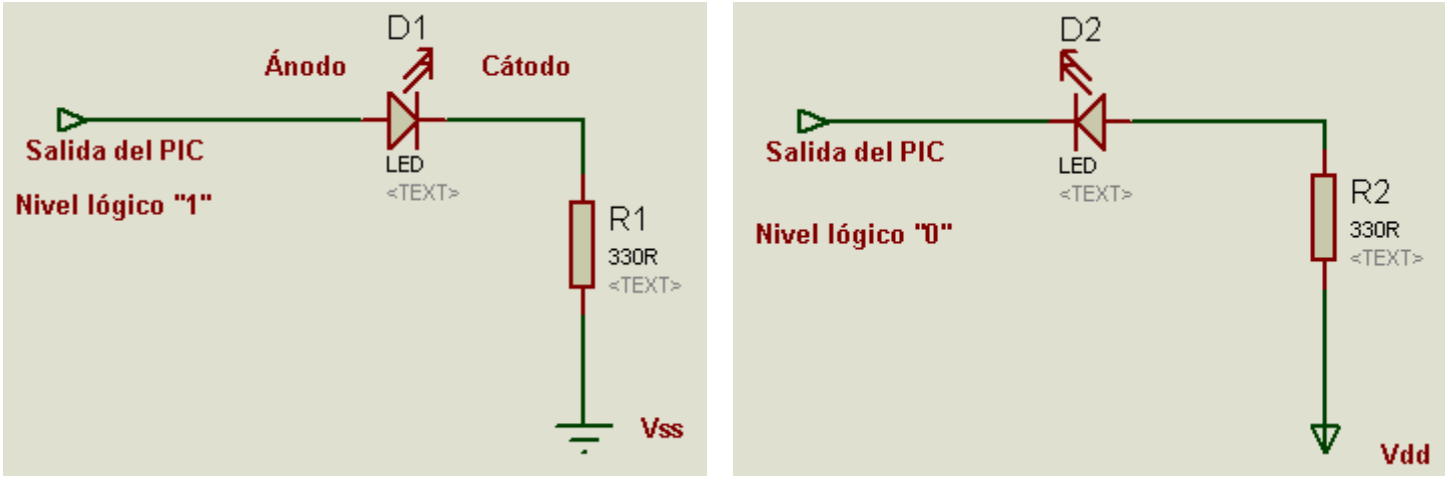
diferencia de potencial entre sus extremos de 1.2 a 2.2V y sólo requieren de 5 a 30mA para que se observe su luminosidad. En la figura 11 se muestran las partes componentes de un led y su símbolo electrónico.



a) b)

Figura 11. a) Componentes de un led, b) símbolo electrónico.

En la figura 12 se muestran dos formas posibles de conectar un led. En la figura 11a el ánodo del led se conecta directamente a una salida del microcontrolador, mientras que el cátodo se conecta en serie con un resistor de 330Ω que cierra el circuito conectado a V_{SS}. De esta manera, cuando la salida del microcontrolador envíe un nivel lógico “1” el led se iluminará. En la figura 11b el cátodo del led se conecta directamente a una salida del microcontrolador, mientras que el ánodo se conecta en serie con un resistor de 330Ω que cierra el circuito conectado a V_{DD}. De esta manera, cuando la salida del microcontrolador envíe un nivel lógico “0” el led se iluminará.



a) b)

Figura 12. a) El Led se ilumina con nivel lógico “1”, b) El led se ilumina con nivel lógico “0”.

2.10. Sensores

Un sensor es un dispositivo diseñado para captar magnitudes físicas o químicas, las cuales son transformadas en variables eléctricas. Algunas de las cantidades físicas que se pueden medir con los sensores son: temperatura, intensidad luminosa, aceleración, distancia, inclinación, presión, fuerza, torsión, humedad y pH. Los sensores se aplican en la robótica, en la industria automotriz, aeroespacial, manufactura, medicina, etc. Algunos sensores pueden trabajar de forma analógica y otros de forma digital.

En el presente manual se presenta un proyecto que involucra el uso del censor óptico CNY70, que es un sensor óptico reflexivo con salida a transistor. El sensor detecta un objeto que se encuentra a una distancia de entre 5 y 10mm. La salida del sensor es analógica y su valor de tensión depende de la cercanía del objeto. En este manual se utilizará como un sensor de no contacto digital, por lo que el sensor detectará la presencia o ausencia de un objeto frente a él, entre las distancias antes mencionadas. En la figura 13 se presenta el diagrama de conexión del sensor CNY70. En esta forma de conexión la salida del comparador LM358 enviará un nivel lógico “1” hacia el microcontrolador cuando el sensor CNY70 detecte la presencia de un objeto “claro”.

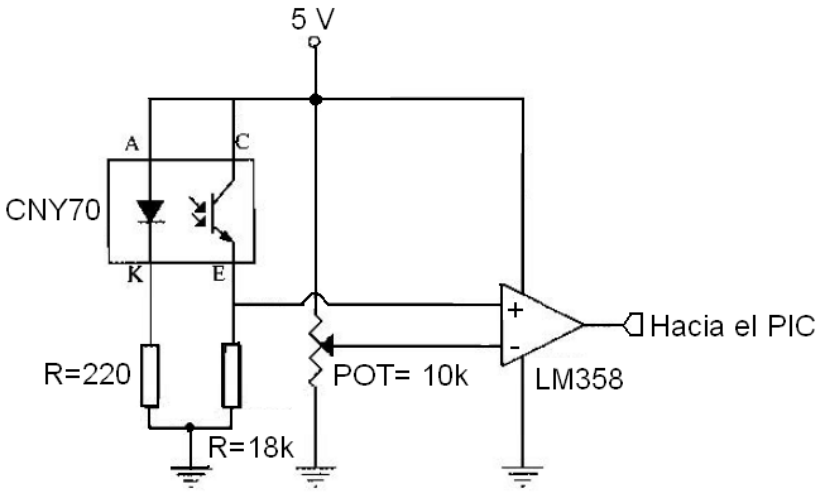


Figura 13. Sensor CNY70 en modo digital mediante un comparador (Palacios, Remiro, & López, 2004).

2.11. Actuadores

Un actuador es un dispositivo diseñado para transformar energía eléctrica, neumática o hidráulica para realizar la activación de un proceso, con la finalidad de producir un efecto en un proceso automatizado. Los actuadores eléctricos, neumáticos, hidráulicos y electrónicos se utilizan para operar dispositivos mecatrónicos. Normalmente, los actuadores hidráulicos se utilizan cuando se requiere una gran cantidad de potencia, mientras que los neumáticos se utilizan para realizar posicionamientos. Los actuadores eléctricos son más versátiles y los más usuales son: electroválvulas, resistencias calefactoras, motores eléctricos (de inducción, de corriente continua, sin escobillas, paso a paso y servomotores), bombas, compresores y ventiladores.

2.12. Relevador

El relé es un dispositivo electromecánico que hace la función de un interruptor controlado mediante un circuito eléctrico, que por medio de una bobina y un electroimán acciona uno o varios contactos que permiten cerrar o abrir otros circuitos eléctricos independientes. Los relevadores pueden controlar circuitos de salida de mayor potencia que el circuito de entrada, por lo que suele llamárseles como amplificadores eléctricos. En la figura 14 se presenta un ejemplo de relé y su símbolo electrónico.

a) b)

Figura 14. a) Relevador con bobina de 6V DC, b) símbolo electrónico.

En la figura 15 se presenta el diagrama de conexiones para el funcionamiento adecuado de un microcontrolador con salida a relevador.

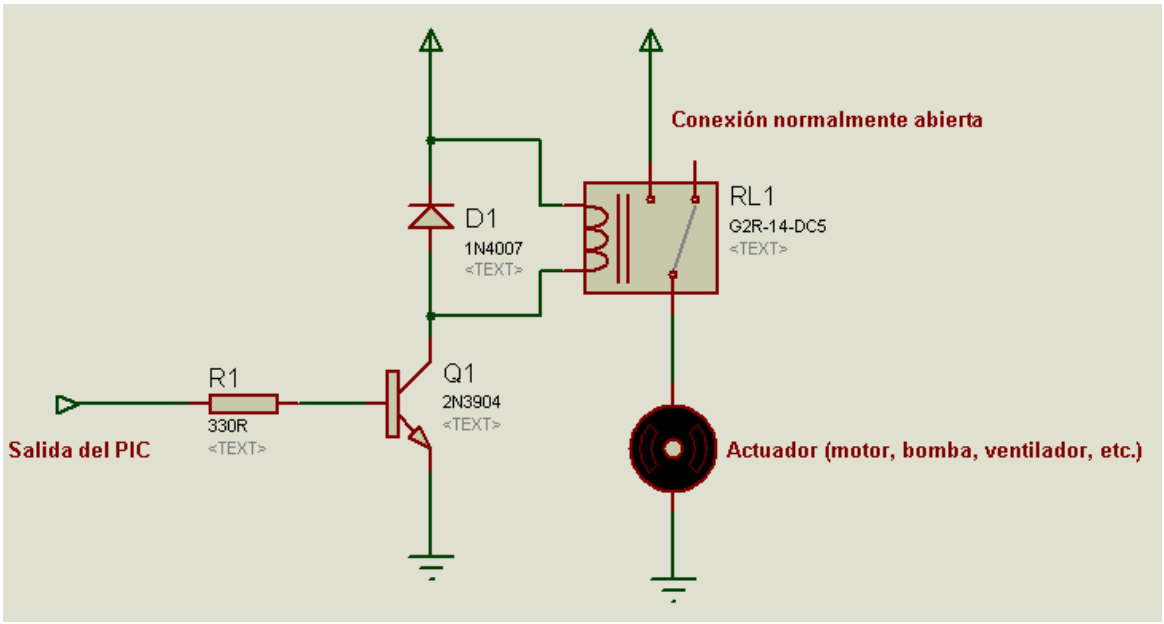


Figura 15. Conexión de salida a relevador.

2.13. Display LCD

Los display LCD permiten mostrar la información requerida mediante caracteres alfanuméricos. Existen gran variedad de displays LCD, sin embargo, en el presente manual se centrará la atención en un display LCD de 16x2 (2 líneas por 16 columnas). Este tipo de display puede ser gobernado mediante conexión de bus de 4 bits o de 8 bits. Para disminuir el número de salidas utilizadas en el microcontrolador se escogerá la conexión de bus de 4 bits. En la figura 16 se presenta un ejemplo del display LCD JHD 162A y el diagrama electrónico del LCD LM016L. Estos LCD difieren únicamente en que el LCD JHD 162A posee dos pines extras (pines 15 y 16) que sirven para proveer de luz trasera al LCD. El pin 15 se conecta en serie con una resistencia de 470Ω o de 680Ω y la resistencia se conecta a V_{DD}. El pin 16 se conecta a V_{SS}.

a) b)

Figura 16. a) Display LCD JHD 162A, b) símbolo electrónico LCD LM016L.

En la figura 17 se muestra el diagrama de un ejemplo de conexiones para el display LCD LM016L, el cual se conecta con el puerto B del PIC18F4550. Estas mismas conexiones aplican para el LCD JHD 162A.

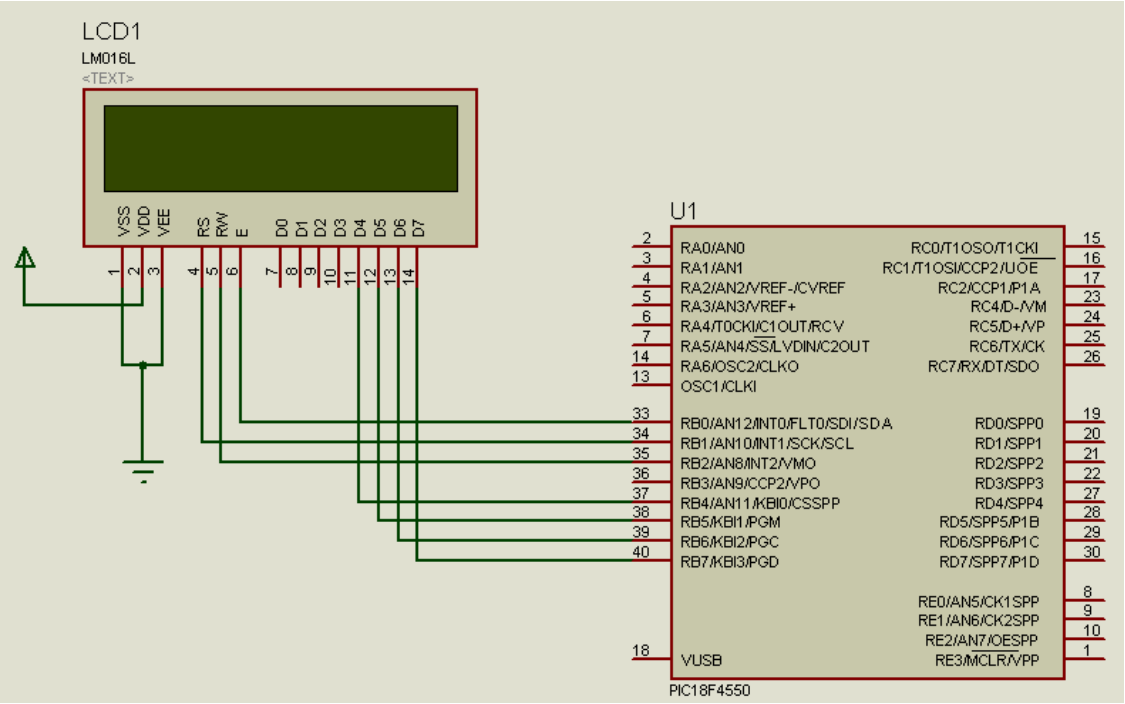


Figura 17. a) Diagrama de conexiones entre el LCD LM016L y el PIC18F4550.

2.14. MAX 232

El MAX232 es un circuito integrado que convierte las señales serie de una PC a señales adecuadas para su uso en TTL compatible con los circuitos de lógica digital de un microcontrolador y viceversa. Para realizar una conexión full duplex desde el USART del PIC, se deben conectar las señales TXD, RXD y la masa (GND), tal como se observa en la figura 18.

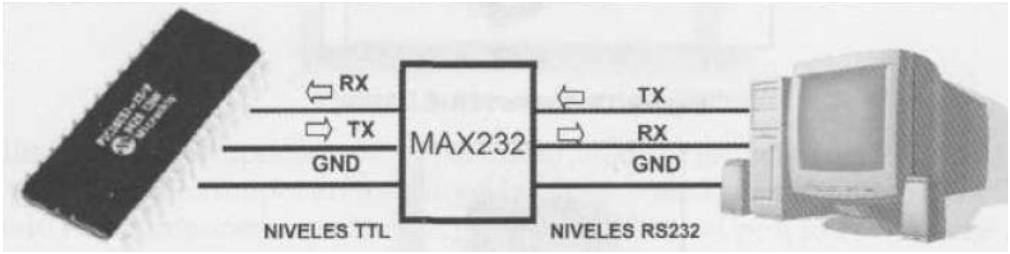
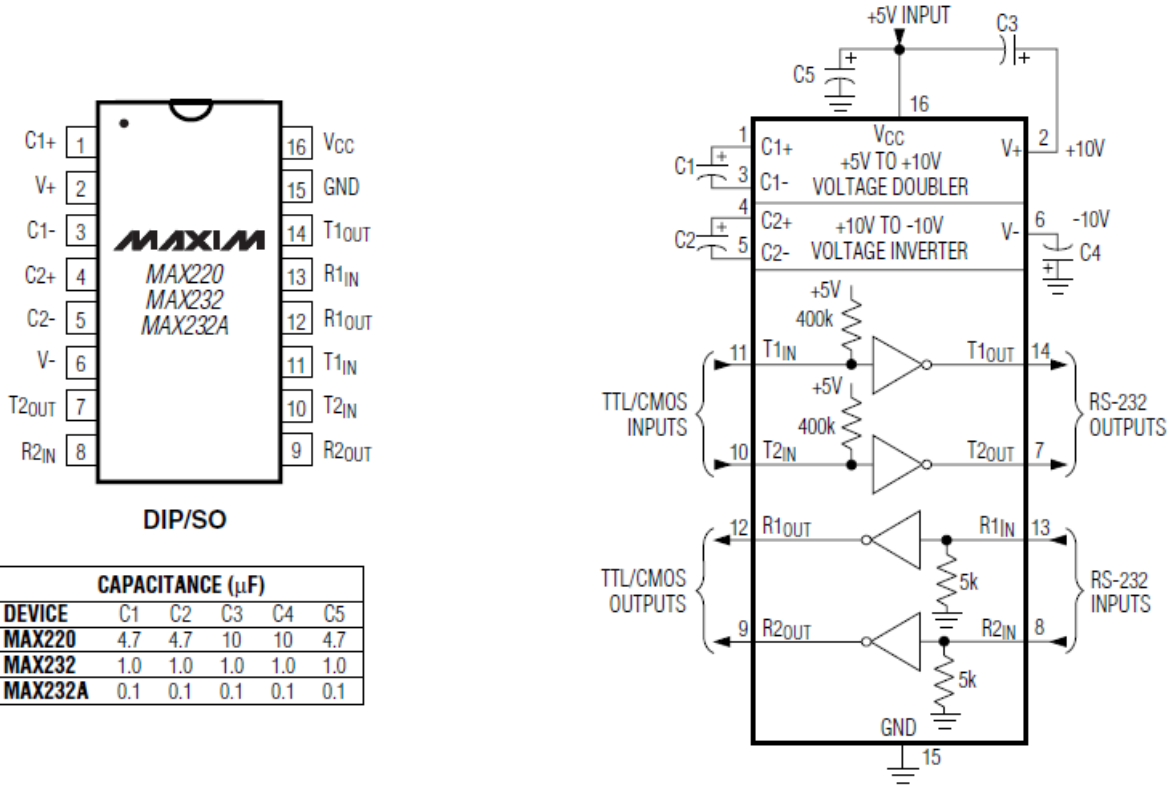


Figura 18. Ejemplo de conexión full duplex entre un PIC y una PC (García, 2008).

En el presente manual se realizará la conexión serial desde el PIC mediante un conector hembra DB9. Si la PC tiene puerto serial, la conexión es directa mediante un cable serial. Sin embargo, si se usa una laptop será necesario utilizar un convertidor serial a usb, debido a que las laptops actuales ya no poseen un puerto serie. Para que la comunicación funcione con la laptop, será necesario instalarle un controlador, llamado PL2303. En la figura 19 se muestra un diagrama de las conexiones necesarias para el MAX 232, el cual requiere cuatro capacitores electrolíticos de 1µF.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figura 19. Diagrama de conexión del MAX 232 (Maxim, 2001).

2.15. LM358N

El LM358N es un circuito que consiste en dos amplificadores operacionales independientes de alta ganancia. Estos operacionales están diseñados para operar con una sola fuente de voltaje. Las posibles aplicaciones son: la amplificación de transductores, como un amplificador operacional común que puede ser alimentado con una sola fuente positiva. Algunos sensores no proporcionan señales digitales puras y es necesario acondicionar dicha señal antes de aplicarla al microcontrolador. El LM358N es un circuito de gran utilidad cuando se desea controlar un circuito digital con señales que no lo son.

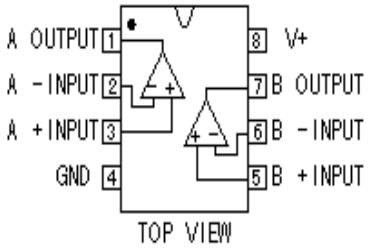


Figura 20. Diagrama de pines del amplificador operacional LM358N.

2.16. Zumbador

Es muy útil cuando se requiere indicar mediante una señal audible que ha ocurrido algún evento en particular. El tipo de zumbador a utilizar es piezoeléctrico. En la figura 21 se muestra una imagen del símbolo electrónico zumbador.

Figura 21. Símbolo electrónico del zumbador.

2.17. Programador de PICs

Para grabar un programa a un microcontrolador se requiere de un programador de PICs. El programador que se utilizará en este manual es el programador MASTER PROG, el cual se conecta a la PC mediante USB. En la figura 22 se muestra una fotografía del programador.

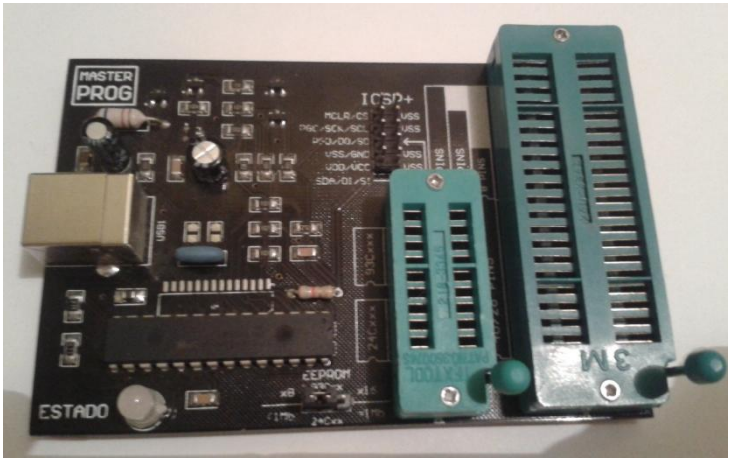


Figura 22. Programador de PICs MASTER PROG.

Para grabar un programa al microcontrolador PIC18F4550, se coloca el PIC sobre la base del programador y se asegura el PIC mediante la palanca de sujeción. Después se conecta el programador a la PC mediante el cable USB. Enseguida se da doble clic al archivo ejecutable de la interfaz del programador. Tal interfaz se muestra en la figura 23.

Figura 23. Interfaz de usuario del programador MASTER PROG.

Una vez abierta la interfaz, se da clic en el menú “Archivo” y se escoge la opción “Abrir Hex”, tal como se muestra en la figura 24.

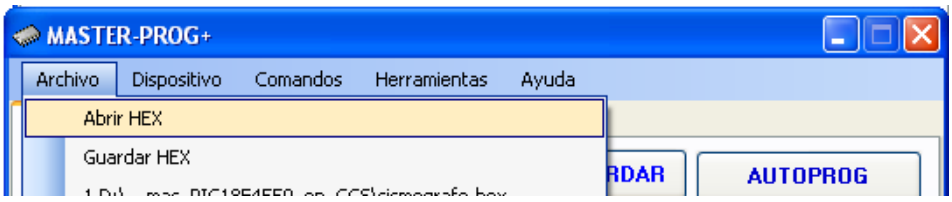


Figura 24. Se busca el archivo hexadecimal a grabar en el PIC.

Se busca el archivo hexadecimal que corresponde al programa que se desea grabar al microcontrolador. Finalmente, se presiona el botón “Escribir”. Al finalizar, la interfaz muestra el mensaje: “ Programación Correcta!” (Ver la figura 25).

Figura 25. Firmware grabado con éxito al PIC18F4550.

2.18. Protoboard

Los protoboards son una herramienta excelente para montar circuitos electrónicos de prueba, sin la necesidad de realizar conexiones permanentes. Un protoboard consiste de una cubierta de plástico que tiene pequeños huecos. En la figura 26 se muestra un protoboard, así como sus conexiones internas, vistas desde su parte trasera.

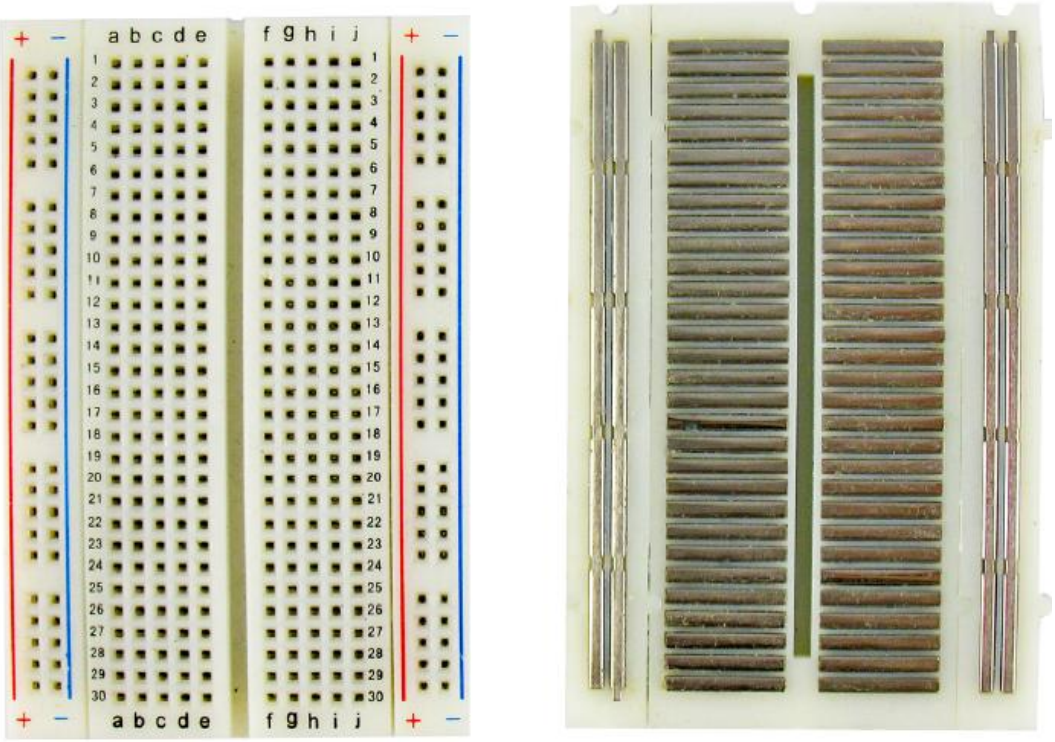


Figura 26. Vista frontal y posterior de un protoboard (Faludi, 2011).

3. PROGRAMACIÓN EN C

En este manual se utilizará el lenguaje C para realizar la programación del microcontrolador PIC18F4550.

3.1. Compilador de C

El compilador C de CCS se desarrolló específicamente para la programación de los microcontroladores PIC, por lo que dispone de una amplia colección de librerías de funciones predefinidas. Para mayor información, visite la página web www.ccsinfo.com

Un demo del compilador se puede descargar de forma gratuita de la página web antes mencionada. El demo es completamente funcional durante 45 días (requiere conexión a internet para su uso).

3.2. Estructura básica de un programa en C

Los programas en C requieren que el programador tenga en cuenta los siguientes requerimientos mínimos de estructura.

DIRECTIVAS

Estas incluyen las palabras reservadas. Sirven para configurar la forma de trabajo del microcontrolador. Estas directivas comienzan con el símbolo # y continúan con una palabra reservada específica.

DEFINICIÓN DE VARIABLES Y CONSTANTES

Se definen las variables y constantes globales que son necesarias para la el programa a desarrollar. Los tipos variables pueden ser byte, entero, flotante, lógico, caracter, etc.

FUNCIONES

Existen funciones predefinidas por el compilador CCS, sin embargo, se pueden establecer funciones propias.

FUNCIÓN PRINCIPAL

Esta función se requiere para iniciar el flujo del programa, ya que es llamada cuando inicia la ejecución de un programa.

INSTRUCCIONES
Las instrucciones determinan las acciones del programa dentro del microcontrolador.

COMENTARIOS
Estos se pueden utilizar en cualquier lugar del programa. Sirven para facilitar la lectura de las acciones que ejecuta el programa.

El programa así escrito se denomina fuente y puede estar escrito en uno o varios archivos. Para que el programa pueda ser ejecutado, se debe compilar y enlazar con todas las funciones que se necesiten. El proceso de compilar consiste en traducir el programa fuente a código máquina. La extensión de archivo que se utiliza para realizar la grabación del programa al PIC es ".hex"

3.3. Creación de un proyecto en CCS

Existen dos formas para realizar un proyecto en el compilador C de CCS. Se puede elegir la opción "PIC Wizard", en la pestaña "Project", para que el compilador guíe paso a paso la creación del proyecto, ver la figura 27. La otra forma es hacerlo de manera manual. Para esto es necesario elegir la opción "Create" en la misma pestaña "Project". En este manual se trabajará con la segunda opción, pues de esta manera se tiene el control completo del proyecto.

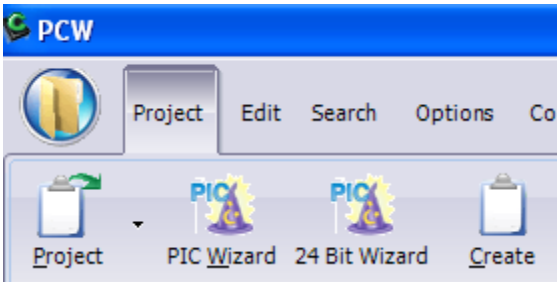


Figura 27. Creación de un proyecto en el compilador C de CCS.

Antes de crear un proyecto de forma manual, es necesario que antes ya se haya realizado el programa que se utilizará en el proyecto. Una vez que se ha creado el programa, cuando se elige la opción "Create" aparece una ventana donde se debe buscar y seleccionar el programa a utilizar y

después se debe seleccionar el microcontrolador a utilizar. Una vez realizado lo anterior, será posible compilar el programa y construir el archivo hexadecimal que se grabará en el PIC.

3.4. Entradas y salidas

En los siguientes proyectos se abordan las acciones básicas de entradas y salidas en un microcontrolador.

3.4.1. Proyecto 1. Encendido de leds.

Se emplearán dos botones pulsadores (señales de entrada) y dos leds (señales de salida). Cuando se presione un botón pulsador se encenderá un led. Cada botón pulsador controlará un solo led. Los leds permanecerán encendidos mientras sus respectivos botones permanezcan pulsados.

Los materiales necesarios son:

- 1 PIC18F4550
- 1 Oscilador de 20MHz
- 2 capacitores de 15pF
- 3 resistencias de 10kΩ
- 2 resistencias de 330Ω
- 2 botones pulsadores
- 2 leds
- Alambre calibre 22
- 1 Fuente de 5V CD

Suele ser muy conveniente que antes de realizar un montaje real del proyecto, así como la programación del microcontrolador, se realice primero una simulación del mismo, para verificar su correcto funcionamiento. Actualmente existen varios simuladores de circuitos electrónicos que nos facilitan dicha tarea. En este manual se presentan ejemplos de simulaciones realizadas con el software Proteus. Se puede descargar Proteus 7 Demo directamente de la página web de Labcenter Electronics www.labcenter.com

Para realizar la simulación se trabaja en el módulo ISIS Proteus, en el cual se buscan en sus librerías todos los elementos que se utilizarán en el circuito electrónico. Realizado esto, se acomodan uno a uno los elementos en forma conveniente dentro del espacio de trabajo. Después se realizan las conexiones adecuadas para que el circuito quede completo. Como ejemplo, en la figura 28 se presenta el diagrama esquemático correspondiente al proyecto 1.

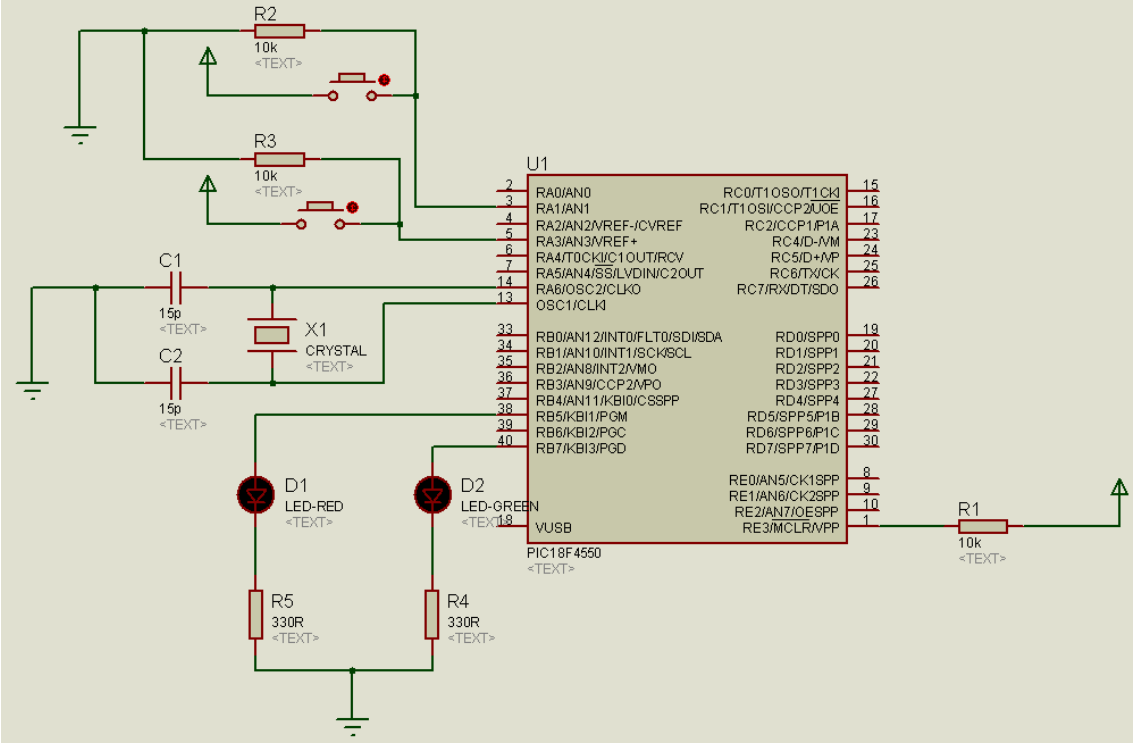


Figura 28. Encendido y apagado de leds con botones pulsadores.

Para poder realizar la simulación, es necesario que se cargue el programa elaborado en C compilado y transformado a la extensión hexadecimal. Una vez hecho lo anterior, se inicia la simulación en tiempo real del proyecto realizado. Cuando se tiene éxito en la simulación, se está en condiciones de realizar el montaje real del proyecto. En la figura 29 se presenta una fotografía del circuito electrónico montado en un protoboard.

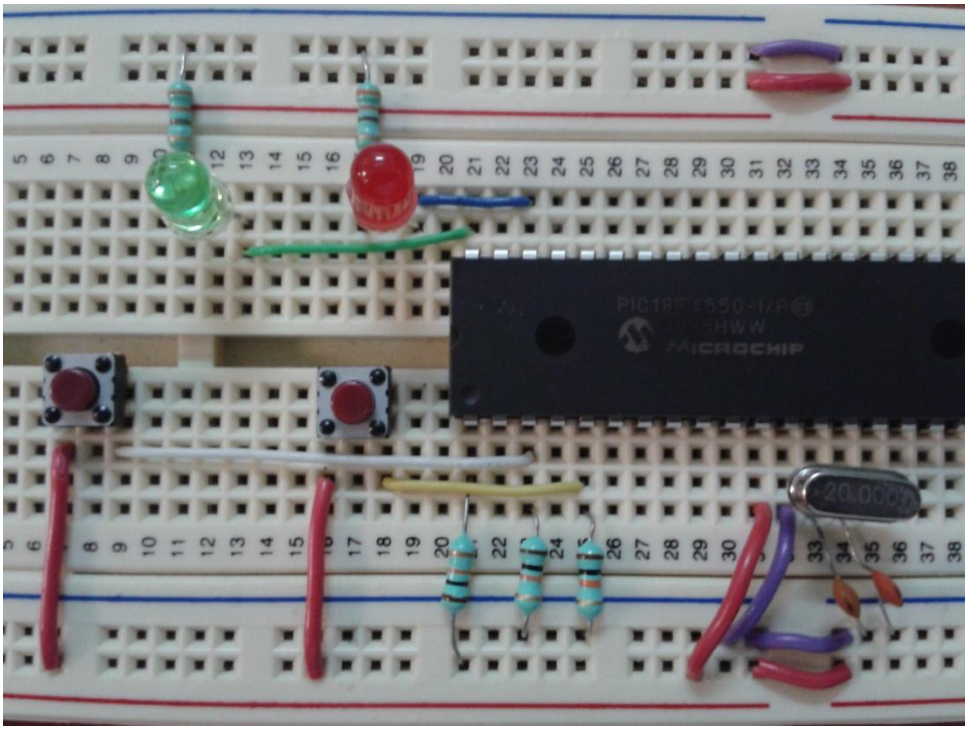


Figura 29. Montaje real del proyecto 1.

A continuación, en la figura 30 se presentan dos programas del proyecto 1. La única diferencia entre los dos programas presentados es básicamente la forma en que son configurados los puertos. El programa que se muestra a la izquierda utiliza la directiva predefinida por el compilador C de CCS Para configurar los puertos: #USE FAST_IO (puerto). Con esta directiva, la configuración de los puertos se realiza mediante la función: set_tris_puerto (valor). En el ejemplo, set_tris_a(0xFF) configura el puerto A como entradas y set_tris_b(0x00) configura el puerto B como salidas.

El programa que se muestra a la derecha realiza la configuración a través de la RAM, definiendo los registros TRISx y PORTx como BYTES situándolos en la posición correspondiente en la RAM. La directiva usada es #BYTE. Después se escribe directamente sobre los registros para configurar los puertos. En el ejemplo, #BYTE TRISA 0X0F92 define la posición en la memoria RAM del registro A, #BYTE PORTA 0X0F80 define la posición en memoria del puerto A y PORTA=0XFF configura al puerto A como entradas.

En los programas de la figura 30 la directiva #FUSES se utiliza para configurar el comportamiento de trabajo del microcontrolador. En el proyecto 1 se utiliza un oscilador de 20MHz, por lo que se utiliza el fusible HSPLL para describir a un oscilador de alta velocidad con la configuración de preescalador;

el fusible PLL5 se utiliza para establecer un preescalado de 5 a la entrada del microcontrolador para que reciba los 4Mhz que acepta; el fusible CPUDIV4 se utiliza para obtener un pos escalado de 6 para conseguir los 16MHz de frecuencia de trabajo del microcontrolador.

```
#include <18F4550.h>
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP
#fuses NODEBUG,PLL5,CPUDIV4,MCLR
#use delay(clock=16000000)
#use fast_io(a)
#use fast_io(b)

void main(void){

   set_tris_a(0xFF);
   set_tris_b(0x00);

   output_low(PIN_B5);
   output_low(PIN_B7);

   while (true){

      if (input(PIN_A1)==1)
         output_high(PIN_B7);
      else
         output_low(PIN_B7);

      if (input(PIN_A3)==1)
         output_high(PIN_B5);
      else
         output_low(PIN_B5);
   }
}
```

```
#include <18F4550.h>
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP
#fuses NODEBUG,PLL5,CPUDIV4,MCLR
#use delay(clock=16000000)

#BYTE TRISA  = 0x0F92
#BYTE TRISB  = 0x0F93
#BYTE PORTA  = 0x0F80
#BYTE PORTB  = 0x0F81

void main(void){

   PORTA = 0xFF;
   PORTB = 0x00;

   output_low(PIN_B5);
   output_low(PIN_B7);

   while (true){

      if (input(PIN_A1)==1)
         output_high(PIN_B7);
      else
         output_low(PIN_B7);

      if (input(PIN_A3)==1)
         output_high(PIN_B5);
      else
         output_low(PIN_B5);
   }
}
```

Figura 30. Programa Proyecto1.c (Dos formas distintas de configuración de puertos).

Para poder realizar la simulación, es necesario que se cargue en el simulador el programa elaborado en C compilado y transformado a la extensión hexadecimal. Para hacerlo se da doble clic en el diagrama del microcontrolador y aparece una ventana donde se busca el programa con extensión hexadecimal y después se configura la frecuencia de trabajo. Una vez hecho lo anterior, se inicia la simulación en tiempo real del proyecto realizado.

3.4.2. Proyecto 2. Semáforo.

Tomando como base el circuito del proyecto 1, ahora se agrega un led de color amarillo conectado al pin 39 del PIC18F4550 (RB6). De esta manera se realizará la simulación del funcionamiento de un

semáforo. Al energizar el sistema, el semáforo permanece apagado. Cuando se presiona el botón conectado al pin 3 del PIC (RA1) se inicia el funcionamiento del semáforo, el cual trabaja de la siguiente manera: el led verde permanece encendido durante 4 segundos, después parpadea dos veces en un intervalo de un segundo, después se enciende el led amarillo durante un segundo y finalmente el led rojo durante tres segundos, repitiéndose de manera indefinida. Para apagar el semáforo (sin resetearlo) se debe presionar y mantener presionado el botón conectado al pin 5 del PIC (RA3) antes de que el led rojo se apague. Esta forma de apagar el semáforo se utilizó para ilustrar más adelante la importancia y ventajas que ofrecen las interrupciones.

Los materiales necesarios son:

- Los materiales del proyecto 1, más
- 1 Resistencia de 330Ω
- 1 led amarillo

En la figura 31 se muestra el diagrama del circuito electrónico realizado en Proteus.

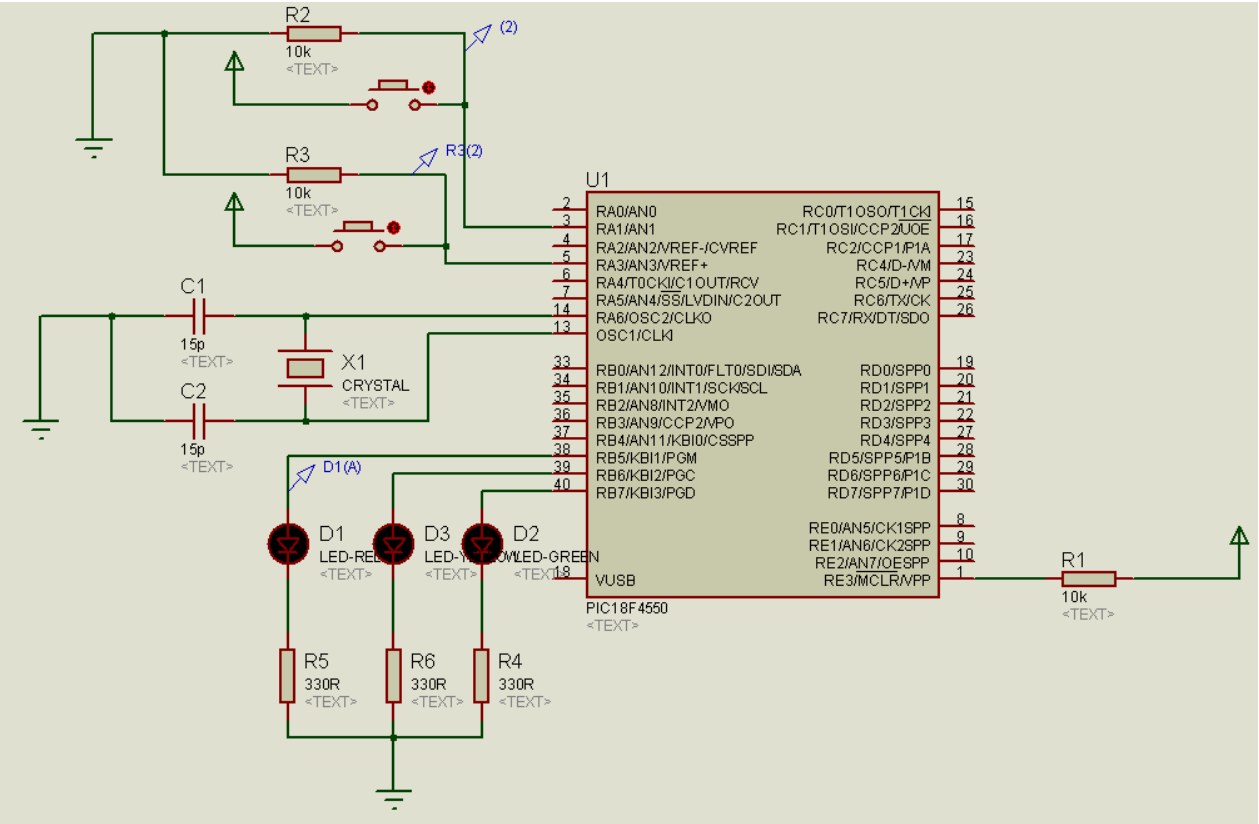


Figura 31. Circuito electrónico del semáforo.

En la figura 32 se presenta el programa del semáforo. En el programa se utiliza la directiva #USE DELAY (CLOCK=16000000), la cual le permite al microcontrolador realizar los cálculos necesarios cuando se utilizan las funciones de retardo DELAY_US y DELAY_MS.

```
#include <18F4550.h>
#fuses HSPLL,PLL5,CPUDIV4,NOPROTECT,NOWDT,MCLR
#use delay (clock=16000000)
#use fast_io(a)
#use fast_io(b)

void main (void){
   set_tris_a(0xFF);
   set_tris_b(0x00);
   output_low(PIN_B5);
   output_low(PIN_B6);
   output_low(PIN_B7);

   while(input(PIN_A1)==0);
   while (true){
      output_high(PIN_B7);
      delay_ms(4000);
      output_low(PIN_B7);
      delay_ms(250);
      output_high(PIN_B7);
      delay_ms(250);
      output_low(PIN_B7);
      delay_ms(250);
      output_high(PIN_B7);
      delay_ms(250);
      output_low(PIN_B7);
      output_high(PIN_B6);
      delay_ms(1000);
      output_low(PIN_B6);
      output_high(PIN_B5);
      delay_ms(3000);
      output_low(PIN_B5);
      if (input(PIN_A3==1))
   }

}
```

Figura 32. Programa proyecto2.c (semáforo).

Ahora se puede realizar la simulación del programa en el circuito realizado en Proteus. Cuando la simulación resulta exitosa, se puede proceder a realizar el montaje real del proyecto. En la figura 33 se presenta una fotografía del circuito electrónico montado en un protoboard.

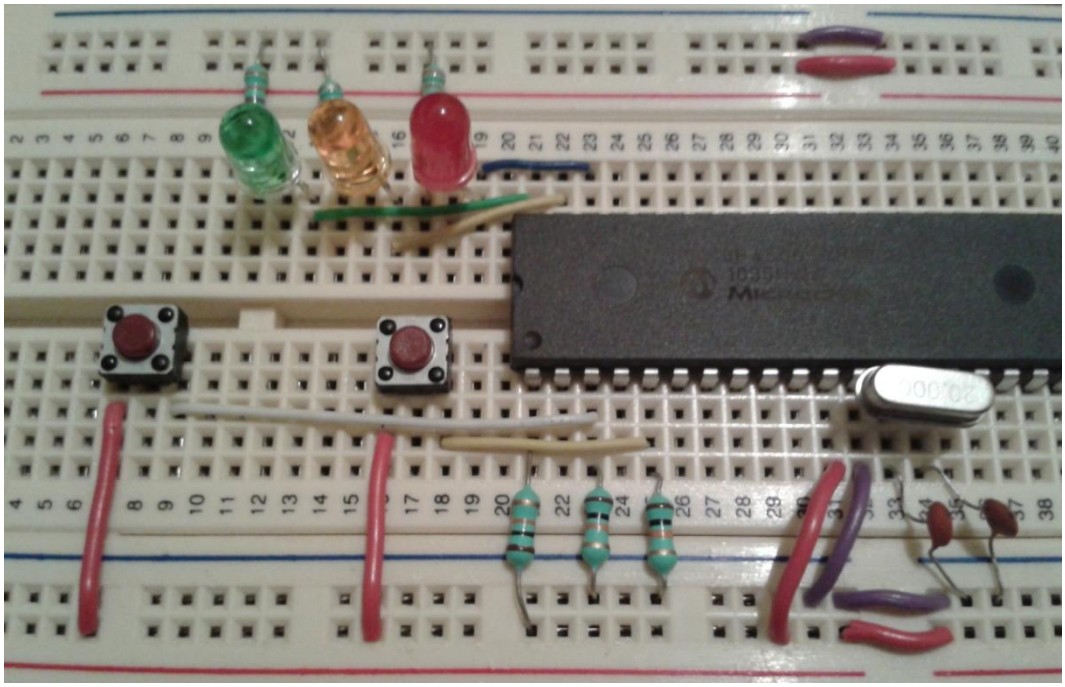


Figura 33. Montaje real del proyecto 2.

3.5. Interrupciones

Una interrupción en el microcontrolador provoca que el flujo normal de un programa se interrumpa para atender las instrucciones de la porción de programa que corresponde a la interrupción. De esta manera, las instrucciones programadas en la interrupción son atendidas de manera inmediata. En el compilador de C, la directiva que se encarga del manejo de las interrupciones es #INT_xx. En el microcontrolador PIC18F4550 se tienen 20 fuentes posibles de interrupción. El microcontrolador PIC18F4550 tiene dos vectores de interrupción, uno de alta prioridad y otro de baja prioridad. La configuración de la prioridad de las interrupciones se puede realizar con instrucciones como: #INT_xx (NORMAL, baja prioridad), #INT_xx FAST (Interrupción de alta prioridad, sin guardar registros), #INT_xx HIGH (Interrupción de alta prioridad), #INT_GLOBAL (La función del usuario se localiza en la dirección 8 para que el usuario controle la interrupción). En el proyecto 2b se presenta el uso de una interrupción normal.

3.5.1. Proyecto 2b. Semáforo con uso de interrupción.

El proyecto 2b es muy similar al proyecto 2. Sin embargo, en este proyecto se muestra el uso de una interrupción externa para hacer que se detenga el funcionamiento del semáforo. Se toma como base

el circuito del proyecto 2, sin embargo, ahora se agrega un botón pulsador conectado al pin 33 del PIC18F4550 (RB0). El botón conectado al pin 5 del microcontrolador (RA3) en el proyecto 2 no será utilizado, así que en el montaje real en protoboard se puede quitar este botón. La diferencia en el funcionamiento es que en el proyecto 2 para apagar el semáforo era necesario presionar y mantener presionado el botón conectado a RA3 antes de que se apagara el led rojo, para que cuando el led se apagara y el pic leyera la siguiente instrucción detectara que estaba presionado el botón. En este proyecto 2b se utilizará una fuente de interrupción externa (botón pulsador conectado a RB0) que hará que cambie de valor una variable global para que se apague el semáforo cuando se apague el led rojo. De esta forma, ya no es necesario mantener presionado el botón conectado al pin 5 (RA3), ahora sólo es necesario presionar en cualquier momento y de forma momentánea al botón conectado al pin 33 (RB0). El diagrama del circuito electrónico se muestra en la figura 34.

Los materiales necesarios son:

- Los materiales del proyecto 2, más
- 1 Resistencia de 10kΩ
- 1 Botón pulsador

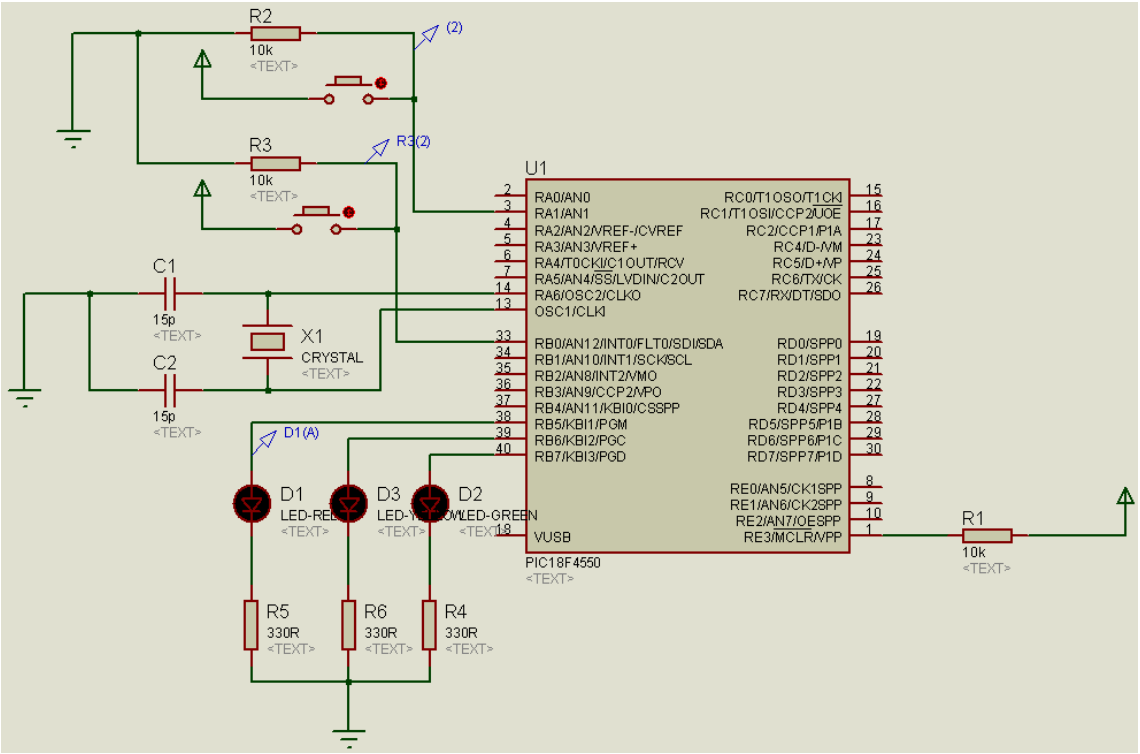


Figura 34. Circuito electrónico del semáforo (con interrupción externa).

En la figura 35 se presenta el programa del semáforo, utilizando la interrupción externa por RB0. EN el programa, se declara la variable global *int8 x=0*, cuyo valor será modificado en la función de interrupción. La función de interrupción del pin 33 del PIC (RB0) es #INT_EXT y esta se define después de las directivas y antes de la función *main*.

```
#include <18F4550.h>
#fuses HSPLL,PLL5,CPUDIV4,NOPROTECT,NOWDT,MCLR
#use delay (clock=16000000)
#use fast_io(a)
#use fast_io(b)
int8 x=0;

#int_ext
void ext_isr(){
   x=0;
}

void main (void){
   set_tris_a(0xFF);
   set_tris_b(0x01);
   output_low(PIN_B5);
   output_low(PIN_B6);
   output_low(PIN_B7);
   enable_interrupts(global);
   enable_interrupts(int_ext);
   ext_int_edge(H_TO_L);

   while (true){
      if (input(PIN_A1)==1)
         x=1;
         while(x==1){
            output_high(PIN_B7);
            delay_ms(4000);
            output_low(PIN_B7);
            delay_ms(250);
            output_high(PIN_B7);
            delay_ms(250);
            output_low(PIN_B7);
            delay_ms(250);
            output_high(PIN_B7);
            delay_ms(250);
            output_low(PIN_B7);
            output_high(PIN_B6);
            delay_ms(1000);
            output_low(PIN_B6);
            output_high(PIN_B5);
            delay_ms(3000);
            output_low(PIN_B5);
      }
   }
}
```

Figura 35. Programa proyecto2b.c (semáforo con interrupción).

Las instrucciones que permiten habilitar la interrupción externa por el pin 33 (RB0) son las siguientes: *enable_interrupts(global)*, *enable_interrupts(int_ext)* y *ext_int_edge (H_TO_L)*. Estas funciones permiten habilitar todas las interrupciones del PIC, habilitan la interrupción externa por RB0 y se utiliza el flanco de bajada para activar la interrupción, respectivamente. Es muy importante notar que en la configuración del PIC se debe definir como entrada al pin 33 (RB0) para que pueda ocurrir la interrupción, *set_tris_b(0x01)*. En la figura 36 se muestra una fotografía del montaje real del proyecto 2b en protoboard. En la fotografía aparece también conectado un botón al pin 5 del pic, pero como ya se mencionó, no se utilizará y se puede quitar del montaje.

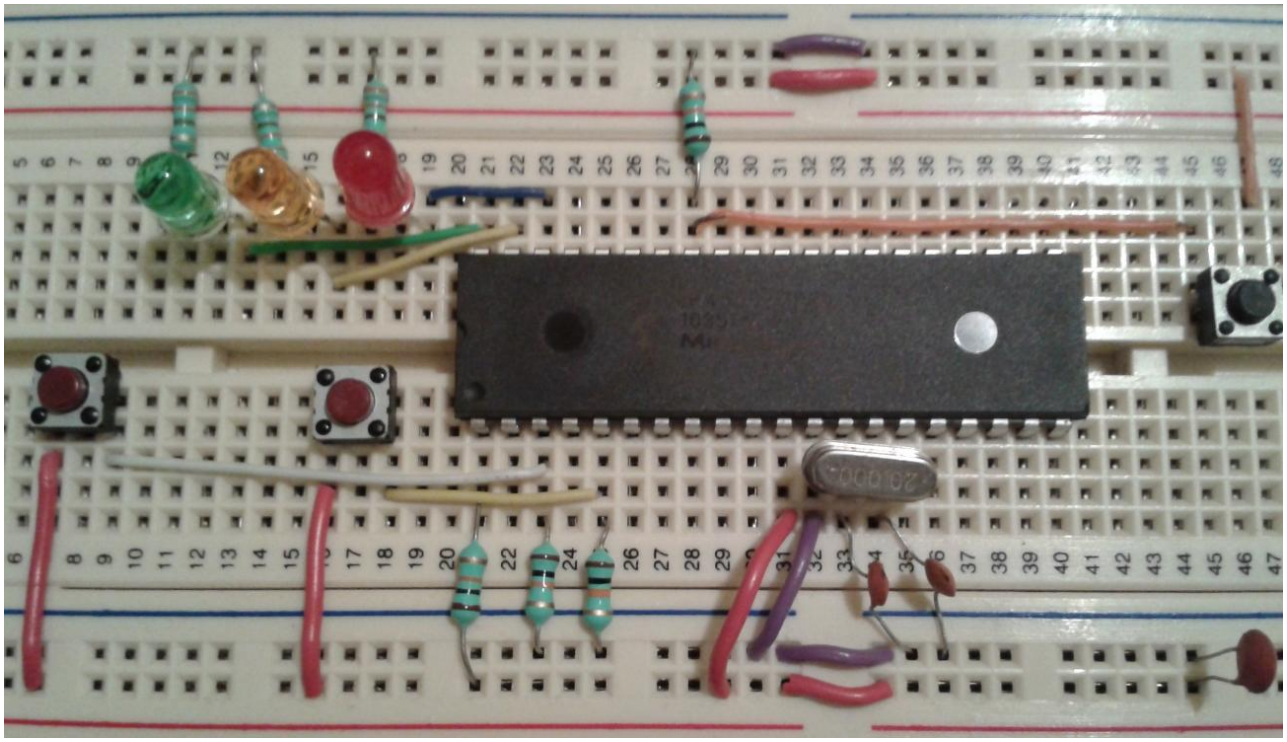


Figura 36. Montaje real del proyecto 2b.

3.6. Convertidor A/D y uso del display LCD

En esta sección se hará uso del convertidor analógico/digital que posee el PIC18F4550 para recibir una señal analógica y mostrar su valor en el display LCD. La señal analógica será leída mediante el pin 2 del PIC (RA0). El LCD se conectará al PIC mediante el puerto B, tal como se ilustra en la figura 17.

3.6.1. Proyecto 3. Lectura de una señal analógica y accionamiento de un relevador y una alarma.

En el proyecto 3 se presenta el uso del módulo A/D del microcontrolador, el uso del display LCD y el accionamiento de un relevador para activar una carga eléctrica de 110V AC.

El sistema contará con una señal analógica, proveniente de un potenciómetro de 10kΩ, que simulará un sensor de temperatura (LM35). Dicha señal será leída mediante la entrada RA0 del microcontrolador (pin 2 del PIC). Como ya se mencionó, el LCD estará conectado al puerto B. El pin 20 del microcontrolador (RD1) funcionará como salida y servirá para activar al relevador que hará funcionar un foco alimentado a 110V AC, el cual simulará un ventilador. El pin 21 (RD2) también funcionará como salida y servirá para activar un buzzer que simulará una alarma que indicará un exceso de temperatura.

El sensor de temperatura funciona linealmente bajo la siguiente función: T(ºC)=100*X. Donde X es el valor leído en voltaje. El valor máximo de lectura del sensor son 150ºC, los cuales alcanza cuando el voltaje es de 1.5V.

El sistema accionará al relevador para que active al ventilador cuando la temperatura se encuentre en un intervalo de 80ºC a 120ºC. Si se supera este intervalo, el ventilador seguirá funcionando pero se activará la alarma que indica un exceso de temperatura en el sistema. En la figura 37 se muestra el diagrama electrónico realizado en Proteus.

Los materiales necesarios son:

- 1 PIC18F4550
- 1 Oscilador de 20MHz
- 2 capacitores de 15pF
- 2 resistencias de 10kΩ
- 1 potenciómetro de 10 kΩ
- 1 transistor 2N3904
- 1 diodo 1N4007
- 1 relevador con bobina de 6V DC
- 1 buzzer

- 1 Display LCD (16x2)
- Alambre calibre 22
- 1 Fuente de 5V CD

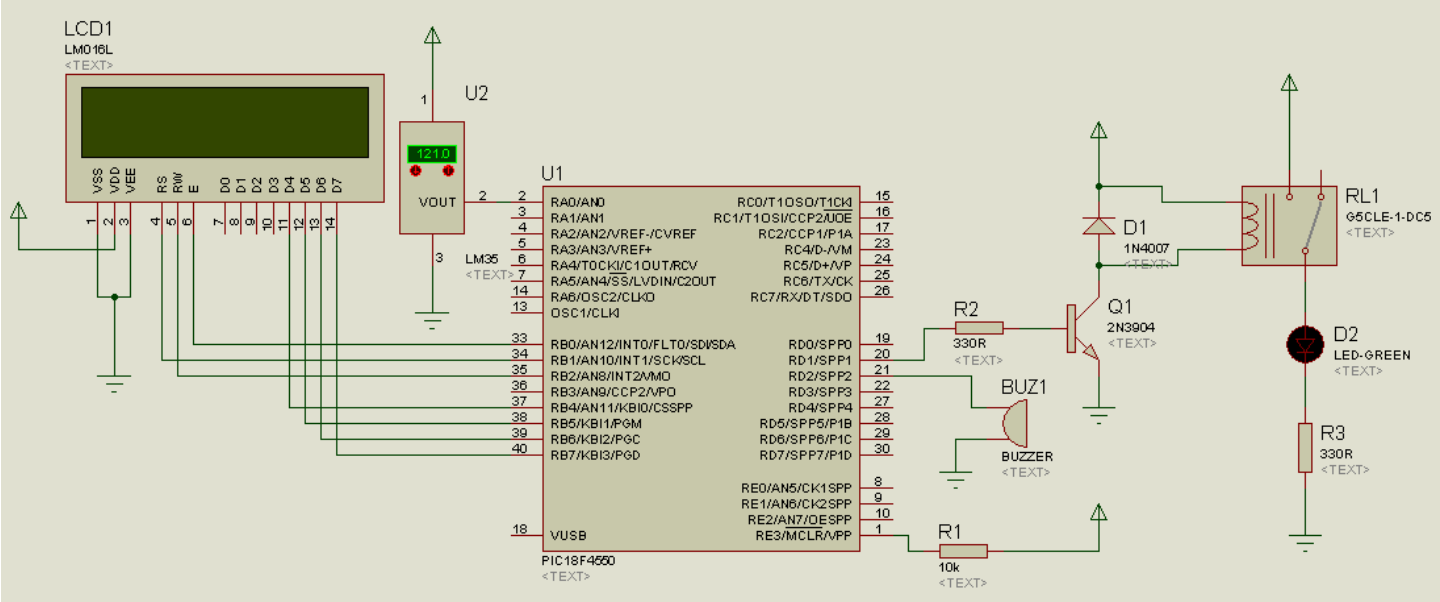


Figura 37. Circuito electrónico del proyecto 3.

En la figura 38 se presenta el código del programa del proyecto 3.

```
#include <18f4550.h>
#fuses HSPLL,NOWDT,NOPROTECT,PLL5,CPUDIV4,MCLR
#device ADC=8
#use delay(clock=16000000)
#use fast_io(a)
#use fast_io(b)
#use fast_io(d)
#define LCD_DATA_PORT getenv("SFR:PORTB")
#include <lcd.c>

unsigned int8 valor_adc;
float factor=0.01960784;
float voltaje;
float temperatura;

void main(){
   lcd_init();
   set_tris_a(255);
   set_tris_b(0);
   set_tris_d(0);
   setup_adc_ports(AN0|VSS_VDD);
   output_low(PIN_D1);
   output_low(PIN_D2);
   setup_adc(ADC_CLOCK_DIV_64);
```

```
    set_adc_channel(0);
    delay_us(50);

    while(true){
        read_adc(ADC_START_ONLY);
        while (!adc_done());
        valor_adc=read_adc(ADC_READ_ONLY);
        voltaje=factor*valor_adc;
        temperatura=100*voltaje;
        lcd_gotoxy(1,1);
        printf(lcd_putc,"%1.2f V ,",voltaje);
        lcd_gotoxy(8,1);
        printf(lcd_putc,"%3.1f C  ",temperatura);

        if ((temperatura > 80) && (temperatura <120)){
          output_high(PIN_D1);
          lcd_gotoxy(1,2);
          printf(lcd_putc,"VENTILADOR ON   ");
        }
        else{
          output_low(PIN_D1);
          lcd_gotoxy(1,2);
          printf(lcd_putc,"                ");
        }
        if (temperatura > 120){
          output_high(PIN_D1);
          output_high(PIN_D2);
          lcd_gotoxy(1,2);
          printf(lcd_putc,"¡TEMP. EXCEDIDA!");
        }
        else{
          output_low(PIN_D2);
        }
        delay_ms(200);
    }
}
```

Figura 38. Programa proyecto3.c.

En el programa se utilizan nuevas funciones dedicadas para la configuración y uso del módulo A/D. La función setup_adc_ports(AN0|VSS_VDD) sirve para este propósito. Con ella de configura el pin 2 del PIC18F4550 como una entrada para señales analógicas con voltajes de referencia de 0V y 5V. La función setup_adc_channel(0) le indica al PIC el pin por donde se realizará la lectura, mientras que la instrucción read_adc sirve para ejecutar la lectura y la conversión de la señal analógica a digital. Además, en las directivas se declara #device ADC=8 para configurar el convertidor A/D a 8 bits. En la figura 39 se muestra el montaje real del proyecto 3 en protoboard.

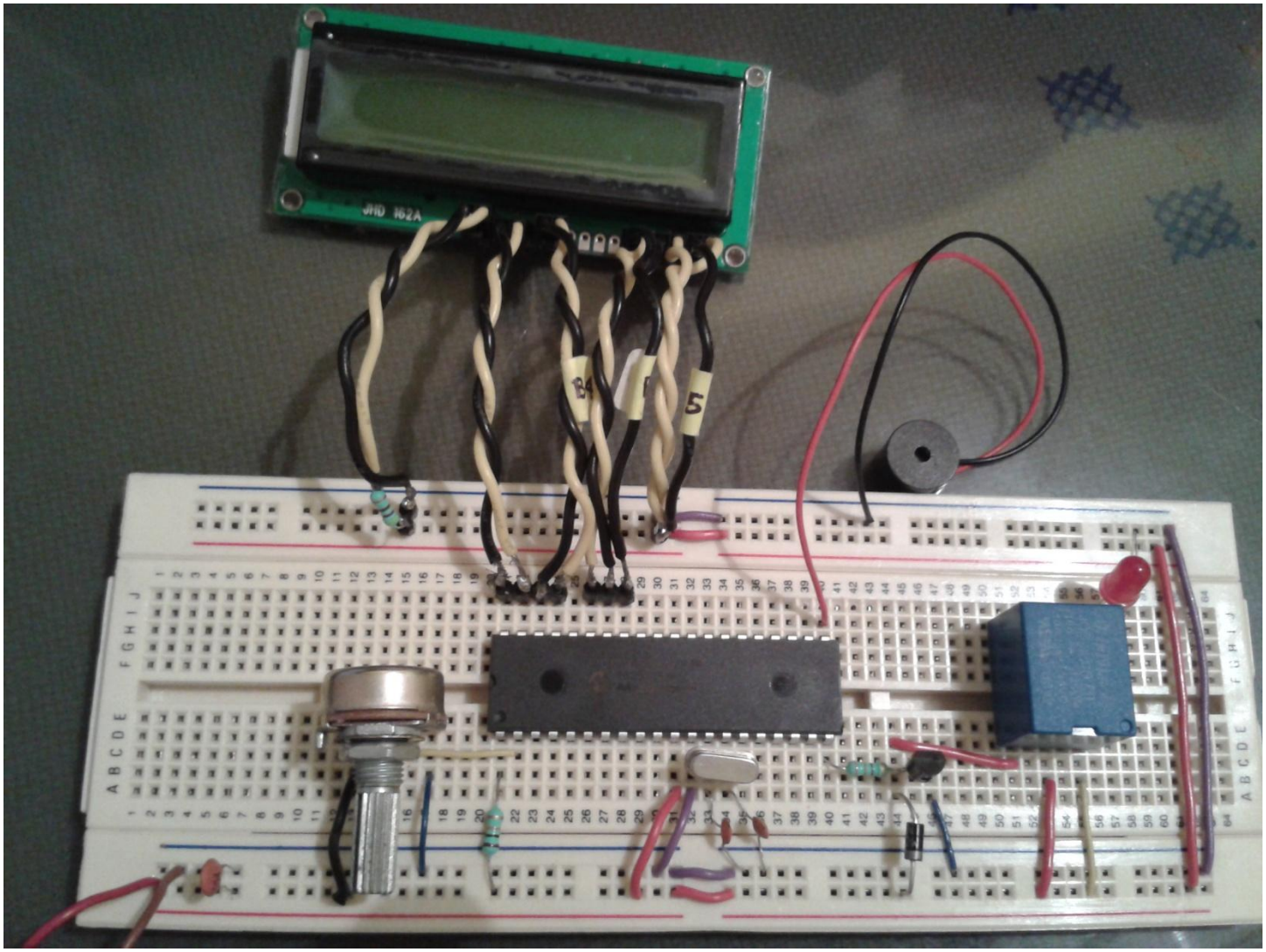


Figura 39. Montaje real del proyecto 3.

3.7. Comunicación serial

Una de las formas más sencillas para comunicar una PC con un dispositivo, como un microcontrolador, es mediante su puerto de comunicación serial utilizando el protocolo estándar RS232. La norma RS232 establece dos tipos de conectores, llamados DB-25 y DB-9. El número indica la cantidad de pines del conector. Cada pin tiene una función especificada por la norma. La longitud máxima de la distancia entre la PC y el dispositivo no debe ser superior a los 15m y la máxima velocidad de transmisión es de 20,000 baudios, para la norma RS232. En la PC para la comunicación serial, un 1 lógico está entre -3V y -15V y el 0 lógico está entre +3V y +15V, por lo que estos niveles no son compatibles con las señales TTL de los microcontroladores. Así que es necesario utilizar un módulo conversor de señales, como el MAX232 descrito en la sección 2.14 de

este manual. Más detalles acerca de la norma RS232 se encuentran en (García, 2008) y en (Palacios, Remiro, & López, 2004).

Para comunicar una PC con un microcontrolador sólo se requieren tres pines de conexión mediante el conector DB-9:

1. Transmisión de datos (TxD), pin 3
2. Recepción de datos (RxD), pin 2
3. Pin de masa (SG), pin 5

En la figura 40 se muestran ejemplos de conectores DB-9 hembras, así como el detalle de los alambres soldados a los pines 2, 3 y 5. Este tipo de conector es el que se utiliza en el microcontrolador.

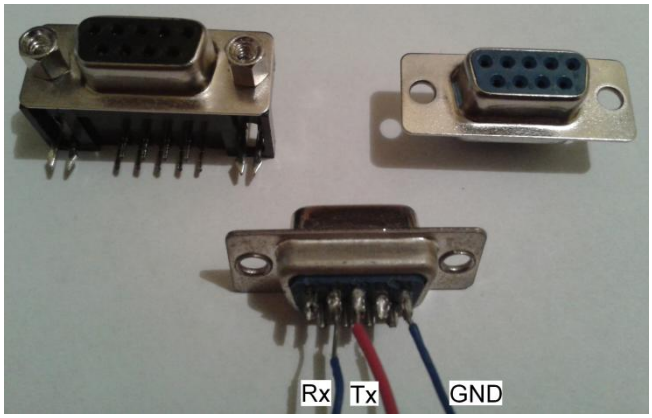


Figura 40. Conectores DB-9 hembras.

3.7.1. Proyecto 4. Comunicación serial mediante interfaz de usuario desarrollada en Matlab.

En este proyecto se demuestra el uso de la comunicación serial entre un microcontrolador y una PC o laptop. El montaje del proyecto se basa en el proyecto anterior, sólo que ahora no se utilizará el buzzer y se agregarán las conexiones necesarias para el MAX232 y el conector DB9.

El presente proyecto ilustra un sistema muy básico de control SCADA como los que se utilizan en la industria actualmente. Básicamente, el sistema del microcontrolador contará con una señal analógica proveniente de un potenciómetro de 10kΩ, la cuál será recibida constantemente por el microcontrolador en el pin 2 (AN0). Esta señal simula la entrada analógica proveniente de un sensor, que puede ser de temperatura, presión, humedad, aceleración, nivel de líquido, etc. El

microcontrolador utilizará su conversor A/D de 8 bits para obtener el valor digital de la señal analógica, el cual mostrará en el display LCD con la leyenda “DATO AN0: valor”, donde “valor” puede variar de 0 a 255. Al mismo tiempo, tal valor será enviado por el microcontrolador hacia la PC mediante el puerto serial.

En el sistema se tiene instalado un relevador que hará la función de activar una carga o actuador, el cual puede ser un foco, un led, un motor, una bomba, un ventilador, etc. El relevador está conectado al pin 20 del microcontrolador (RD1). El relevador sólo podrá ser activado cuando una señal enviada por el puerto serial desde la PC así lo indique. De esta manera, el usuario u operador, controlará la actividad del relevador desde la interfaz de usuario en su PC.

La interfaz de usuario a realizar se muestra en la figura 41, en la cual se puede ver que cuenta con 4 botones, una barra deslizable y una imagen de una lámpara, así como un área con texto indicando el nombre del puerto de comunicación serial.

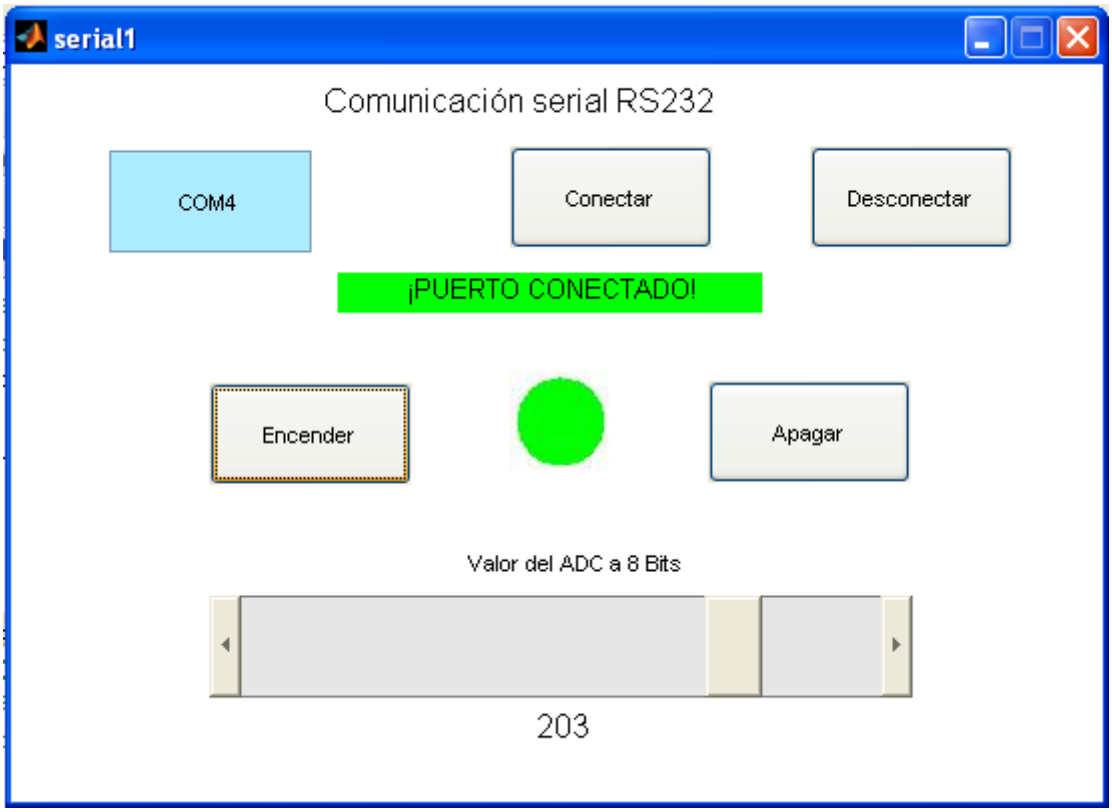


Figura 41. Interfaz gráfica de usuario del sistema SCADA.

El botón “Conectar” se utiliza para establecer la comunicación serial entre el PIC y la PC. El botón “Desconectar” se emplea para detener la comunicación. Cuando se presione el botón “Encender” la imagen de la lámpara cambiará indicando que se ha encendido y se enviará una señal por puerto serial al microcontrolador para que se funcione el relevador y se active la carga que está conectada al mismo (en este caso se encenderá un led). Cuando se presione el botón “Apagar”, la imagen de la lámpara cambia nuevamente indicando que se ha apagado y la PC enviará por el puerto serial la señal para que se desenergice el relevador y se desactive su carga (en este caso se apaga el led). En la barra deslizante se puede observar gráficamente el nivel de la señal analógica y además se observa su valor en forma textual. Con ello se demostrará la comunicación serial tipo “full duplex” entre el microcontrolador y la PC. En la figura 42 se muestra el montaje real de este proyecto en protoboard.

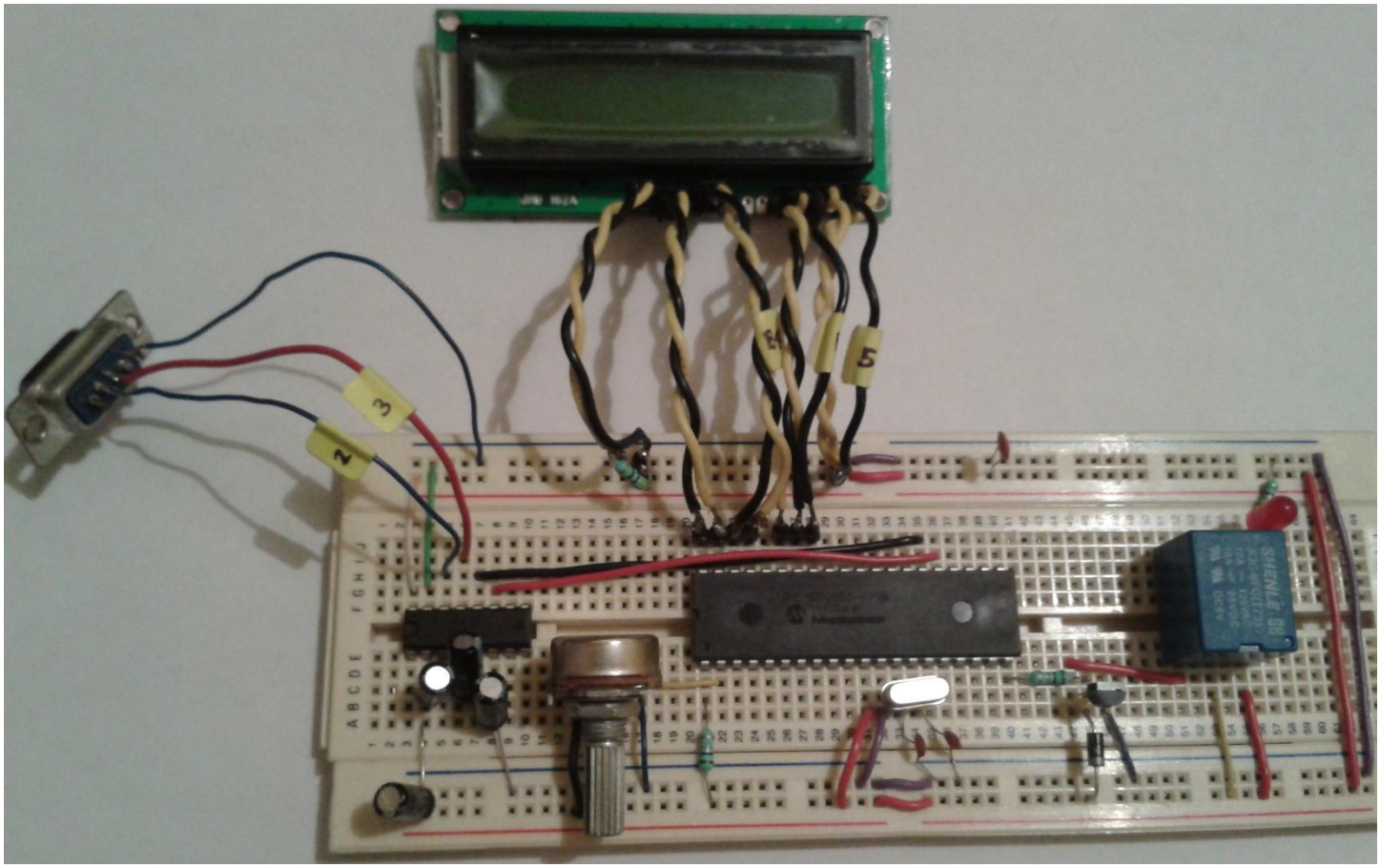


Figura 42. Interfaz gráfica de usuario del sistema SCADA.

Enseguida, en la figura 43 se presenta el programa realizado en CCS. En él se puede observar la directiva: #use rs232(baud=9600, parity=N, xmit=PIN_C6, rcv=PIN_C7, bits=8, stop=2), que es con la cual se configura el módulo de comunicación serial del PIC18F4550.

```
#include <18f4550.h>
#fuses HSPLL,NOWDT,NOPROTECT,PLL5,CPUDIV4,MCLR
#DEVICE ADC=8
#use delay(clock=16MHz)
#use rs232(baud=9600, parity=N, xmit=PIN_C6, rcv=PIN_C7, bits=8, stop=2)
#use fast_io(a)
#use fast_io(b)
#use fast_io(c)
#use fast_io(d)
#define LCD_DATA_PORT getenv("SFR:PORTB")
#include <lcd.c>

int8 valor_adc=0;
int8 valor=0;

#INT_RDA
void rda_isr(void){
   valor=getc();
   lcd_gotoxy(1,2);
   if (valor==1){
      output_high(PIN_D1);
      printf(lcd_putc," LED ENCENDIDO  ");
   }
   else{
      output_low(PIN_D1);
      printf(lcd_putc,"  LED APAGADO   ");
   }
}
void main(){
   lcd_init();
   set_tris_a(0xFF);
   set_tris_b(0x00);
   set_tris_c(0xB9);
   set_tris_d(0x00);

   output_low(PIN_D1);

   lcd_gotoxy(1,1);
   printf(lcd_putc,"COMUNIC.  SERIAL");
   delay_ms(2000);
   printf(lcd_putc,"\f");
   delay_ms(500);

   setup_adc_ports(AN0|VSS_VDD);
   setup_adc(ADC_CLOCK_DIV_64);
   set_adc_channel(0);
   delay_ms(1);

   enable_interrupts(int_rda);
   enable_interrupts(global);
```

```
    while(true){
       read_adc(ADC_START_ONLY);
       while (!adc_done());
       valor_adc=read_adc(ADC_READ_ONLY);
       lcd_gotoxy(1,1);
       printf(lcd_putc,"DATO ANO:%5u",valor_adc);
       putc(valor_adc);
       delay_ms(300);
    }
}
```

Figura 43. Programa para comunicación serial con interfaz de usuario en Matlab.

En el programa se observa también que se utiliza otra fuente de interrupción del microcontrolador, la interrupción por recepción de datos en el puerto serial. Para utilizar esta interrupción se debe utilizar la directiva #INT_RDA y dentro de la función de interrupción se escriben las instrucciones que se requieren que el microcontrolador realice, como la lectura y almacenamiento del dato serial que ha recibido. Dentro de la función “main” es necesario que se habilite este tipo de interrupción, mediante: “enable_interrupts(int_rda)”.

Ahora, para realizar la interfaz de usuario en Matlab se procede como sigue. En la ventana de comandos (Command Window) se escribe la instrucción “guide”, con lo cual se abrirá una ventana en la que es necesario indicar si se abrirá una interfaz existente o si se creará una nueva. En este caso se elegirá crear una nueva y para ello se elige la pestaña “Create New GUI” y luego “Blank GUI (Default)”. Entonces aparece la ventana donde se creará el diseño de la interfaz, tal como se muestra en la figura 44. En esa ventana se encuentran los elementos necesarios para el diseño de la GUI. Se tienen elementos como botón, caja de texto editable, caja de texto no editable, barra deslizable, menú desplegable, gráfica, tabla, etc.

Para la interfaz de usuario se utilizarán 4 botones, un elemento de gráfica, una barra deslizable, una caja de texto editable y 4 cajas de texto no editables, como se muestra en la figura 45. La caja de texto editable tendrá el tag “puerto” y en su string se escribe “COM1” para indicar el nombre del puerto serial al que se va a conectar. El botón con el string “Conectar” tiene el tag “conectar”; el botón con el string “Desconectar” tiene el tag “desconectar”; el botón con el string “Encender” tiene el tag “encender” y el botón con el string “Apagar” tiene el tag “apagar”. Al elemento gráfica se le pone el tag “lampara”. Al elemento deslizable de se pone el tag “valorslider” y a una de las cajas de texto no editable se le pone el tag “valoradc”.

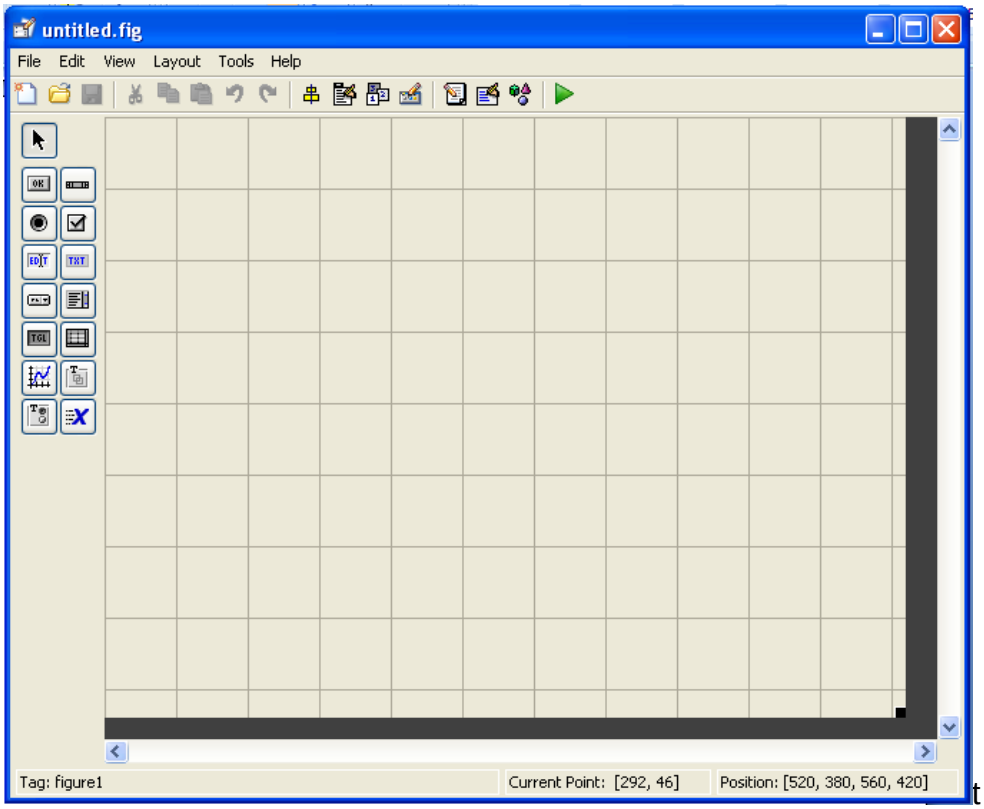


Figura 44. Ventana con herramientas para la creación de interfaz de usuario.

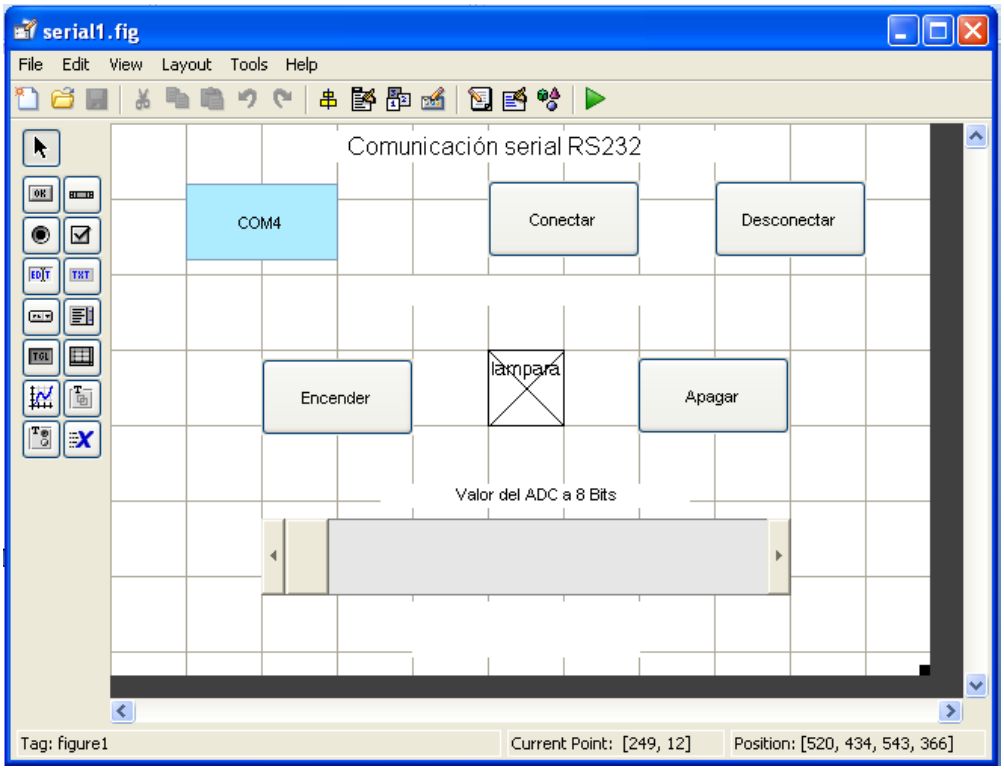


Figura 45. Elementos de la interfaz de usuario.

Enseguida, se presenta el programa de Matlab para la interfaz.

```
function varargout = serial1(varargin)
% SERIAL1 M-file for serial1.fig
% Programa para probar la transmisión de datos por puerto serial RS-232
% El programa recibe un valor analógico y envía un dato para encender o
% apagar un led.
% Programó: M.C. Jesús Medina Cervantes
% El programa funciona en conjunto con el programa CCS: serial.c
% Last Modified by GUIDE v2.5 03-Jul-2012 00:11:09
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @serial1_OpeningFcn, ...
                   'gui_OutputFcn',  @serial1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before serial1 is made visible.
function serial1_OpeningFcn(hObject, eventdata, handles, varargin)
global t led adc handles1 encendido apagado
encendido=imread('b_on.JPG');
apagado=imread('b_off.JPG');
%Se crea el timer
t = timer('TimerFcn',{@tiempo},'ExecutionMode','fixedSpacing','Period', 0.1);
% Choose default command line output for serial1
handles1=handles;
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
imshow(apagado,'Parent',handles1.lampara);
led=0;
adc=0;

% --- Outputs from this function are returned to the command line.
function varargout = serial1_OutputFcn(hObject, eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on slider movement.
function valorslider_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function valorslider_CreateFcn(hObject, eventdata, handles)
```

```
% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on button press in encender.
function encender_Callback(hObject, eventdata, handles)
global led s handles1 encendido
led=1;
imshow(encendido,'Parent',handles1.lampara);
fwrite(s,led);

% --- Executes on button press in apagar.
function apagar_Callback(hObject, eventdata, handles)
global led s handles1 apagado
led=0;
imshow(apagado,'Parent',handles1.lampara);
fwrite(s,led);

% --- Executes on button press in conectar.
function conectar_Callback(hObject, eventdata, handles)
global t s handles1
puerto_s=get(handles1.puerto,'String');
s=serial(puerto_s);
set(s,'BaudRate',9600);
set(s,'DataBits',8);
set(s,'Parity','none');
set(s,'StopBits',2);
set(s,'Flowcontrol','none');
set(s,'InputBufferSize',1);
set(s,'OutputBufferSize',1);
set(s,'Timeout',5);
fopen(s);
set(handles1.mensaje,'String','¡PUERTO CONECTADO!','BackgroundColor','Green');
start(t);

% --- Executes on button press in desconectar.
function desconectar_Callback(hObject, eventdata, handles)
global t s handles1
set(handles1.mensaje,'String','¡PUERTO DESCONECTADO!','BackgroundColor','Red');
stop(t);
fclose(s);
delete(s);
clear s;

function puerto_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function puerto_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function tiempo(hObject, eventdata) %(source,eventdata)
global s adc handles1
```

```
if(s.BytesAvailable==1)
    adc = fread(s,1);
    set(handles1.valoradc,'String',num2str(adc))
    adc=adc/255;
    set(handles1.valorslider,'Value',adc)
end
```

El diagrama electrónico del sistema realizado en Proteus se presenta en la figura 46. Como se puede apreciar, el diagrama es semejante al del proyecto 3, sólo que en este se eliminó el buzzer y se agregó el elemento COMPIM, que sirve para emular el puerto serial.

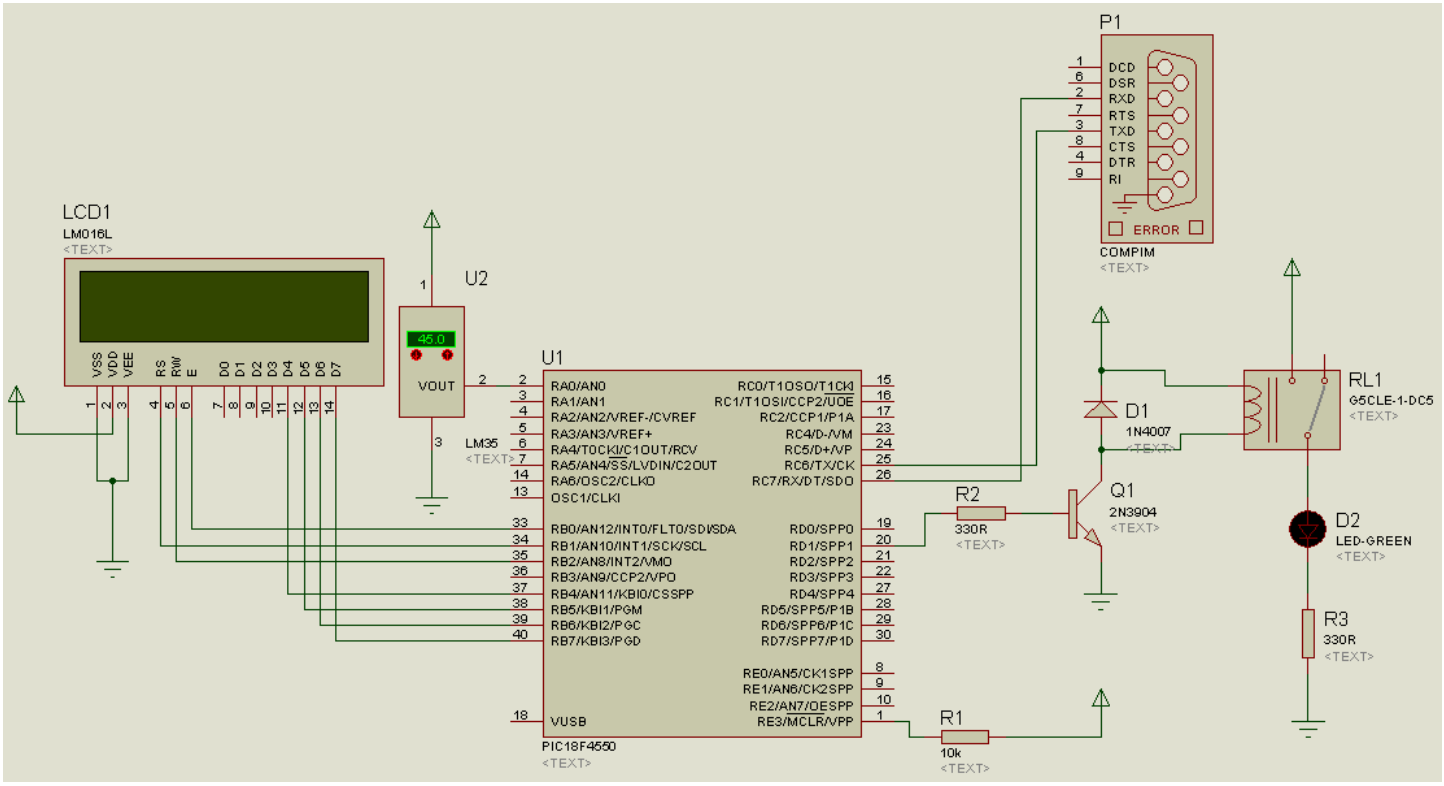


Figura 46. Diagrama electrónico del proyecto 4.

Si se desea simular el funcionamiento del sistema en Proteus es necesario cargar el programa realizado en CCS; pero además, se requiere hacer uso de puertos seriales virtuales para lograr la simulación de la comunicación entre la GUI de Matlab y el circuito realizado en Proteus. Para conseguir la simulación, se hará uso de una versión demo de un software que sirve para crear puertos virtuales en la PC, llamado “Virtual Serial Port Driver”, de la empresa “Eltima Software”. La versión demo de este driver se puede descargar desde: http://www.eltima.com/products/vspdxp/

Una vez que se han creado los puertos seriales virtuales en su PC, sólo resta realizar la simulación y si es exitosa se puede proceder a probar el montaje en el circuito real en el protoboard. En la figura 47 se muestra como ejemplo la simulación del proyecto 4.

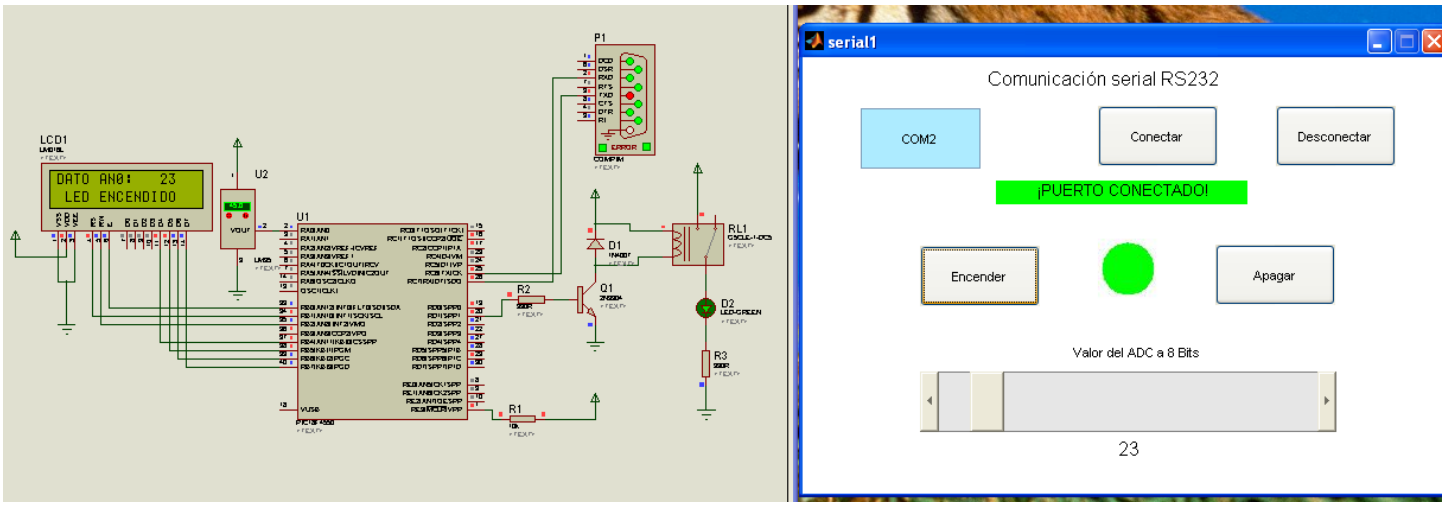


Figura 47. Simulación del sistema SCADA del proyecto 4.

4.CONCLUSIÓN

Los proyectos desarrollados en este manual fueron planteados para que el usuario se introduzca de manera sencilla en el manejo y uso de los microcontroladores con fines educativos. Las herramientas presentadas en el manual son ampliamente utilizadas en la actualidad y permiten que los aprendices se den cuenta con facilidad de los errores que cometen, con lo cual pueden aprender de ellos rápidamente.

Se espera que tanto el taller como el manual sean de utilidad a los académicos de la Facultad y que los motive para aprender más del interesante mundo de los microcontroladores. Sobretodo y como se redactó al principio, se desea que los académicos desarrollen estas competencias para que las apliquen en su quehacer docente de manera didáctica y que promuevan el desarrollo de proyectos con estudiantes, que deriven en la realización de prototipos didácticos que suplan las necesidades de equipamiento de laboratorio.

Para mayores referencias sobre programación y las funciones del compilador CCS, el autor del presente manual recomienda leer la ayuda que trae el compilador y los libros que se incluyen en la bibliografía.

Bibliografía

Casacuberta, J. (13 de Febrero de 2010). *Universidad Politécnica de Valencia, información general.* Recuperado el 15 de agosto de 2011, de http://personales.alumno.upv.es/jorcaga1/ses/archivos/microcontroladores%20pic.pdf

EDJMM9. (2011). *Electrónica y circuitos*. Recuperado el 22 de 06 de 2012, de http://www.taringa.net/comunidades/electronik/1231497/codigo-de-colores-para-resistencias.html

Faludi, R. (2011). *Building Wireless Sensor Networks.* O'Reilly Media Inc.

García, E. (2008). *Compilador C CCS y simulador PROTEUS para microcontroladores PIC.* México: Alfaomega.

Maxim. (Agosto de 2001). Maxim 5V Powered, Multichannel RS-232 Drivers/Receivers. USA.

Microchip. (2009). PIC18F2455/2550/4455/4550 Data Sheet. USA.

Milan, V. (2008). PIC Microcontrolers.

Palacios, E., Remiro, F., & López, L. (2004). *Microcontrolador PIC16F84, Desarrollo de Proyectos.* México: Alfaomega, Ra-Ma.

Salamanca, S. (Octubre de 2003). *webdearde.* Recuperado el 6 de Mayo de 2012, de http://wiki.webdearde.com/images/1/1b/Pic-gama-media.pdf