

<p><i>EDU-PIC</i> Módulo 16F84 Manual del Usuario</p>
--

Indice :

1. Introducción	2
2. Descripción general	2
3. Definición de los pines en el 16F84	4
4. Programador de la memoria FLASH	5
5. Oscilador	5
6. Arquitectura del microcontrolador 16F84	5
7. Memoria	6
8. Registros especiales	7
8.1 Registros PCL y PCLATH	8
8.2 Registro de Status	8
8.3 Registro de Opción	8
9. Registro temporizador /contador TMR0	9
10. Interrupciones del Sistema	9
11. Programación de la EEPROM	10
12. Funciones especiales	11
12.1 Registro de configuración	11
12.2 Oscilador	11
12.3 Power on reset POR	11
12.4 Watch Dog WDT	12
12.5 SLEEP	12
12.6 Code protect	12
13. Puertos digitales	12
13.1 Leds y microswitches	13
13.2 Relevador	13
13.3 Header 16x (conector a teclado 4 x 4)	14
13.4 Conector a LCD	15
14. El set de instrucciones	15
14.1 Operandos	17
14.2 Formato de las instrucciones	18
14.3 Manejo de tablas	18
15. Puesta en marcha	19
16. Información Técnica	23
16.1 Características generales	23
16.2 Lay out	24
16.3 Diagrama electrónico	25
16.4 Lista de componentes	26
16.5 Contenido del disco CD	27
Apéndice 1. Proyectos:	
Teclado matricial , LCD y Real Time Clock	27

1. Introducción :

Deseamos felicitarlo por la adquisición del kit **EDU-PIC**. La filosofía de diseño de este módulo se centró en maximizar el número de funciones disponibles al usuario y disminuir el costo mediante el uso de componentes de uso extensivo. **EDU-PIC** emplea el procesador **16F84** de Microchip. El módulo integra el hardware y software necesarios para programar el 16F84, vía un cable serial conectado a la PC, o bien puede ejecutar los programas ya cargados en la memoria flash del chip, en modo AUTORUN.

El módulo **EDU-PIC** se ofrece en forma de KIT junto con los dispositivos auxiliares de hardware y software necesarios para su inmediata puesta en marcha : eliminador de baterías, cable serial, disco CD de aplicación que incluye el presente manual de operación. El software proporcionado en el disco de aplicación funciona para las diferentes plataformas WINDOWS, versiones 98, 2000, Milenium y XP. Cuenta con la herramienta **MPLAB**, que integra las funciones de editor, ensamblador, simulador, y compilador para el PIC 16F84, así como con la herramienta **WINPIC**, para programar el circuito 16F84 desde la PC.

Le recomendamos ir directamente al capítulo 15, página 19 : “**Puesta en Marcha**”, para tener su sistema operando en cuestión de minutos.

Este manual se acompaña de la especificación técnica y diagramas electrónicos completos del módulo, así como de numerosas ilustraciones y explicación detallada de cada uno de sus interfaces.

Por favor llame al tel. 56 53 58 01 para aclarar cualquier duda sobre su **EDU-PIC**. Puede también enviar sus emails a : **puntoflotante@hotmail.com**

Juan Bosco Martínez, Punto Flotante, S.A., enero de 2009

2. Descripción General.

El sistema **EDU-PIC** fue diseñado para su uso en una amplia gama de aplicaciones : como kit para el aprendizaje del microcontrolador 16F84, como módulo central en equipos de instrumentación ; en aplicaciones de robótica, comunicaciones y redes de control de acceso, o incorporado en sistemas de control industrial.

El diagrama general de la tarjeta EDUPIC se muestra en la siguiente figura:

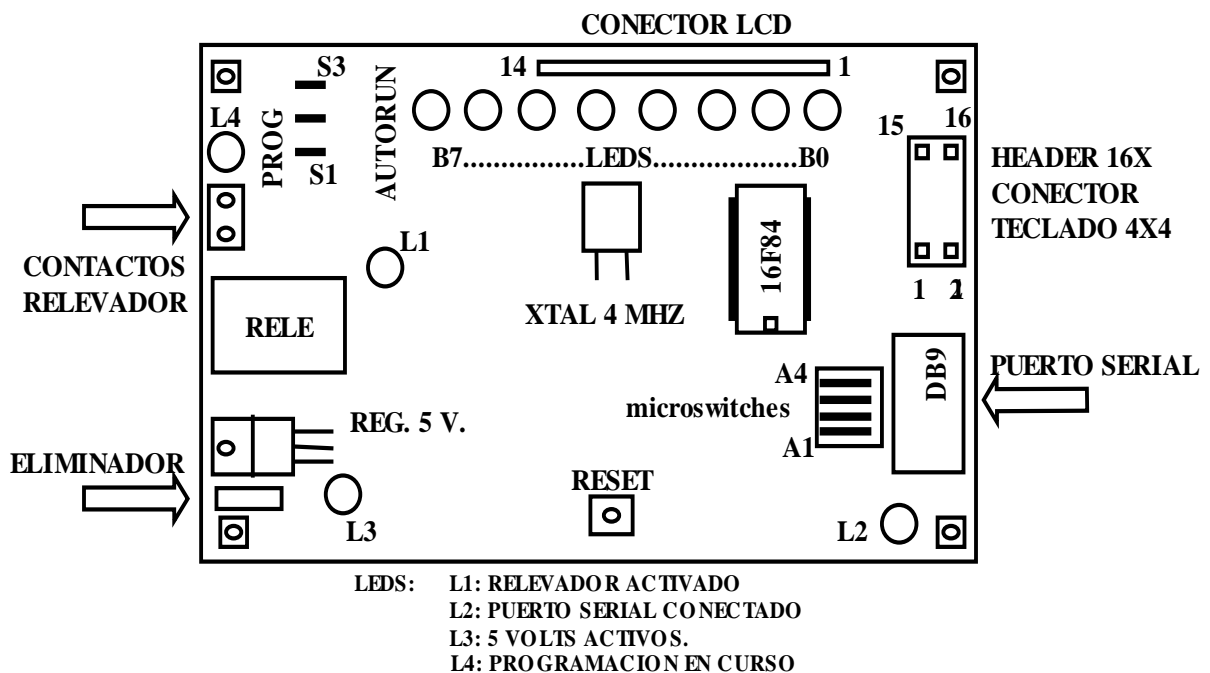


FIGURA 1

Se hace uso de la versión de 18 pines del microcontrolador 16F84, con encapsulado “dual in line”, el cual contiene, 1024 bytes de memoria FLASH, 64 bytes de EEPROM, y 68 bytes de RAM. El diseño de PIC hace un uso extensivo de los puertos e interfaces ofrecidas por el microcontrolador para ofrecer al usuario un sistema con un máximo de opciones disponibles.

La tarjeta consta de dos partes: **el programador** del 16F84, que está diseñado con base en 4 transistores, capacitores y diodos, y **el microcontrolador 16F84** con sus interfaces de leds y microswitches conectados a sus puertos. El programador es controlado por señales en el puerto serial desde la computadora PC, mediante el esquema conocido como ICSP, “In Circuit Serial Programming”, que es el estándar creado por Microchip para la programación de sus dispositivos. El ICSP usa 3 señales del 16F84 para su programación: RB5, RB6 y la señal MCLR. Los jumpers S1, S2 y S3 conectan a éstas 3 señales con el programador cuando su posición es de “PROGRAMAR”. La tarjeta EDU-PIC contiene las siguientes funciones y dispositivos en su **hardware**:

- Microcontrolador 16F84, funcionando con un cristal de 4 Mhz.
- 1K bytes de memoria FLASH, 64 bytes de EEPROM, 68 bytes de RAM y 1 temporizador.
- Tecnología CMOS con muy bajo consumo, en funcionamiento normal, 2 miliamperes @ 5 volts.
- Arquitectura Harvard, con un set de instrucciones RISC, de solamente 35 instrucciones.
- *Programador del 16F84 integrado a la tarjeta*, conector DB9 para puerto serial, para programación desde una PC.
- Puerto de salida de 8 bits con leds conectados como testigos para facilitar pruebas por parte del usuario..
- Entradas para 4 señales digitales con microswitches conectados para facilitar las pruebas.
- Salida para un relevador de baja potencia, integrado a la tarjeta, para la activación de dispositivos externos.
- Conector de 14 pines para conexión a LCD.
- Conector de 8 pines para teclado matricial.
- Conector Header de 16x, para conexión a interfaces externas.
- Circuito Watch Dog programable para evitar que el microcontrolador se salga de operación.
- Modo de operación de bajo consumo SLEEP, con un consumo virtual de 0 (<1 ua).
- Temporizador para la generación de retrasos, reloj de tiempo real ó contador de eventos
- Sistema de interrupciones, generadas desde varios dispositivos, entre ellos el temporizador y la EEPROM.
- Opción de protección de código CODE PROTECTION para evitar posible copia del firmware del circuito.
- En cuanto a su alimentación, EDU-PIC puede activarse mediante un eliminador de baterías externo, el cual se conecta al módulo y alimenta a un regulador de 5 volts integrado a la tarjeta, o bien puede funcionar en forma autónoma por medio de una batería estándar “cuadrada” de 9 volts.

El diagrama de bloques general del hardware del 16F84 se muestra en seguida:

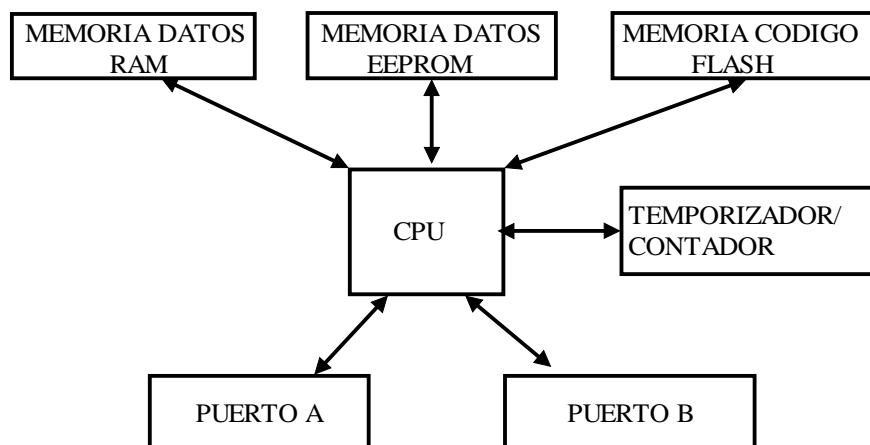


FIGURA 2

SOFTWARE :

- Se proporciona junto con el sistema, un disco CD al usuario. Se incluye en dicho disco: el programa **MPLAB**, el cual incluye un programa editor, ensamblador, simulador y compilador de C. Se incluye también el programa **WINPIC** para la programación de la tarjeta a través del puerto serial, El módulo **EDU-PIC** se conecta mediante un cable al puerto

serial COM1 de una computadora PC, haciendo de ésta una poderosa herramienta de desarrollo. El software anteriormente descrito, puede ser ejecutado desde una PC con cualquier plataforma WINDOWS desde WINDOWS 98 hasta el WINDOWS XP.

DOCUMENTACION :

- El usuario recibe en el mismo disco CD, el manual del usuario, en donde se incluye información completa sobre el sistema, incluyendo diagramas electrónicos, así como una explicación detallada del funcionamiento de cada una de sus interfaces. En el disco CD mencionado, se incluyen los archivos PDF con las “data sheets” completas del chip 16F84 y sus interfaces, y manuales de los programas descritos anteriormente. Asimismo, el usuario podrá estudiar también un programa tutorial con animaciones gráficas, que le permitirá comprender mejor la arquitectura del chip 16F84. Finalmente, el disco CD incluye también una carpeta con programas de prueba para el sistema.

3. Definición de los pines en el 16F84.

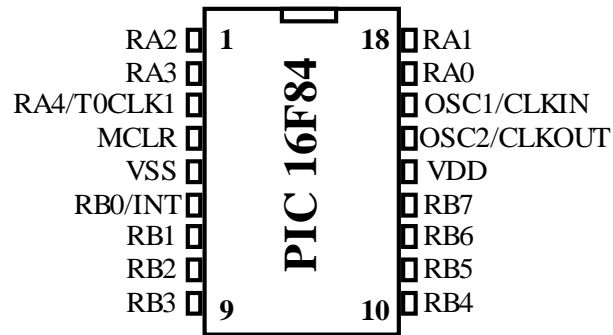


FIGURA 3

Algunos de los pines tienen varias funciones, en la tabla que sigue se describe cada uno de ellos:

Pin	Nombre	Tipo	Funciones
16	OSC1/CLKIN	Input	Señal 1 del Cristal oscilador. En modo RC, señal proveniente del oscilador RC.
15	OSC2/CLKOUT	Output	Señal 2 del Cristal oscilador. En modo RC, se genera una señal de salida (CLKOUT), que tiene ¼ de la frecuencia de la señal en OSC1 y determina el ciclo de instrucción.
4	MCLR	Input	Señal de reset externa. En modo de programación, recibe una señal de 12 a 13 volts.
17	RA0	Bi	Señal de entrada o salida.
18	RA1	Bi	Señal de entrada o salida.
1	RA2	Bi	Señal de entrada o salida.
2	RA3	Bi	Señal de entrada o salida.
3	RA4/T0CLK1	Bi	Señal de entrada o salida. También puede programarse para ser la entrada de reloj para el registro TMR0.
6	RB0/INT	Bi	Señal de entrada o salida. También puede programarse para funcionar como interrupción externa.
7	RB1	Bi	Señal de entrada o salida.
8	RB2	Bi	Señal de entrada o salida.
9	RB3	Bi	Señal de entrada o salida.
10	RB4	Bi	Señal de entrada o salida.
11	RB5	Bi	Señal de entrada o salida.
12	RB6	Bi	Señal de entrada o salida. En modo de programación, es una entrada de reloj para sincronizar los datos seriales.
13	RB7	Bi	Señal de entrada o salida. En modo de programación. Es una señal con datos en formato serial.
5	Vss	5 V.	Alimentación del circuito.
14	Vdd	GND	Tierra del circuito.

4. Programador de la memoria FLASH:

La tarjeta EDU-PIC cuenta con un programador para la memoria FLASH del circuito 16F84. Este programador usa el estándar ICSP (In Circuit Serial Programming) de Microchip para la transferencia de datos, a través del puerto serial COM1 ó COM2 de una computadora PC. El estándar ICSP hace uso de las siguientes señales en el 16F84, las cuales, durante el ciclo de programación, tienen las funciones que se señalan:

MCLR/Vdd: esta señal es usada como voltaje de programación y puede variar entre 13 volts y tierra. Normalmente es la señal de RESET general para el 16F84.

RB6: es la señal de reloj para sincronizar los datos. Normalmente es el bit 6 del puerto B.

RB7: es la señal de datos. Normalmente es el bit 7 del puerto B.

Por medio de 3 jumpers (S1, S2, S3), estas 3 señales son conectadas o aisladas de los pines del 16F84. Cuando se opera el microcontrolador en modo de programación y pruebas (junto con el programa WINPIC), entonces los 3 jumpers permanecen en la posición “PROGRAMAR”. Cuando ya se tiene el programa de aplicación funcionando, entonces los 3 jumpers ahora se cambian a la posición “AUTORUN”, no interfiriendo ninguna de las señales de control del ICSP con la operación del 16F84. En el disco que acompaña a este sistema, se incluye la especificación del estándar ICSP, en formato PDF.

El hardware de este programador opera junto con el software denominado WINPIC. Gracias a este programa, es posible no solamente transferir y programar archivos desde el puerto serial de una computadora PC, hacia la memoria FLASH del microcontrolador, sino que también es posible ejecutar y probar los programas que se están desarrollando, sin necesidad de mover los jumpers de su posición de “PROGRAMAR” o desconectar el cable serial de la computadora PC.

5. Oscilador.

El 16F84 puede funcionar con un oscilador RC ó bien un cristal. En el caso de la tarjeta **EDU-PIC**, un oscilador cristal de 4 MHz se encuentra ya instalado. El oscilador principal es dividido entre 4 para formar los pulsos Q1, Q2, Q3, y Q4, estos 4 pulsos hacen un ciclo de máquina. En el siguiente diagrama se muestra el diagrama básico de operación del reloj, en donde se divide el oscilador principal en 4 ciclos, por cada ciclo de máquina. El circuito impreso de su tarjeta, tiene también opción para la instalación de un oscilador del tipo RC (componentes R6 y C7). Es importante señalar que solamente una de estas opciones puede quedar instalada en su tarjeta y es además necesario hacer la inicialización correcta en el 16F84, indicado en el programa fuente (.ASM) el tipo de oscilador que habrá de usarse.

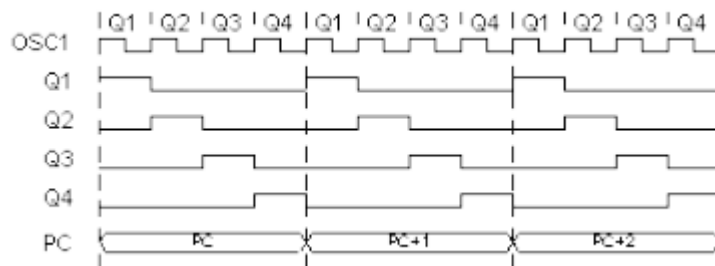


FIGURA 4

6. Arquitectura del microcontrolador PIC 16F84

El PIC16F84 pertenece al tipo de procesadores con arquitectura Harvard, es decir, la memoria de datos y de código separadas, y arquitectura RISC (Reduced Instruction Set Computer). El microcontrolador cuenta con los siguientes elementos: memoria de programa de 1K del tipo FLASH, programable y borrable eléctricamente, 64 bytes de memoria EEPROM para el almacenamiento de datos permanentes en memoria, 68 bytes de RAM., dos puertos de entrada-salida, el puerto A con 5 señales y el B con 8 señales, en total 13 señales de entrada salida. Adicionalmente, el microcontrolador cuenta con un temporizador de 8 bits, el cual al activarse incrementa su cuenta cada 4 pulsos del reloj principal, generando una señal de interrupción al llegar a la cuenta prefijada.

El CPU puede procesar un total de 35 instrucciones. Cada una de las instrucciones está contenida en una palabra de 14 bits y se ejecutan en un ciclo de instrucción, con excepción de las instrucciones de JUMP, BRANCH y CALL. Lo anterior es debido al esquema de “pipeline” usado en arquitecturas HARVARD y que permiten al procesador realizar el FETCH y el EXECUTE simultáneamente con excepción de las instrucciones de salto referidas. En el siguiente diagrama se muestra la ejecución del programa con un sistema tipo “pipeline”. Obsérvese que en todos los ciclos de reloj, se hace el fetch y execute simultáneamente, con

excepción del ciclo TCY4, en donde se deshecha (flush) la instrucción número 4 y se continúa con la instrucción 5, llamada por la subrutina..

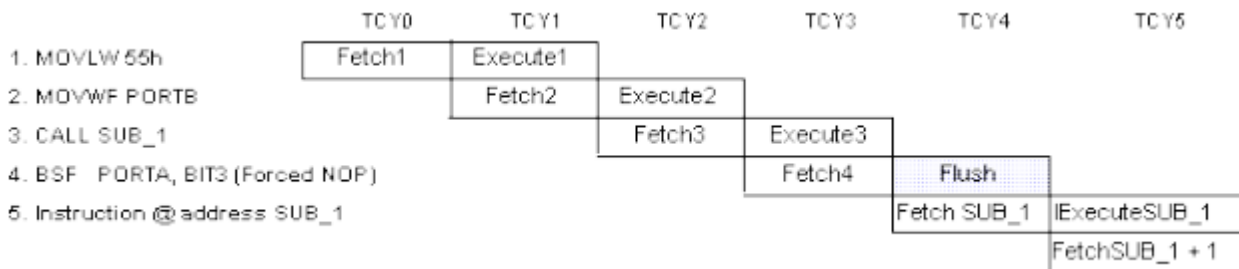


FIGURA 5

El microcontrolador 16F84 contiene los siguientes registros principales: el **registro W**, de 8 bits, que es el único acumulador del procesador, y el **registro PC** (program counter) que es un apuntador de 10 bits y que direcciona a la siguiente localidad de memoria de código que habrá de leerse y ejecutarse.

También existe una **PILA ó STACK**, que se usa para el manejo de las instrucciones de CALL, RETURN, RETFIE, RETLW. Es una pila de 8 niveles que se encuentra en una memoria independiente de la memoria de programa y código, y allí se almacenan y recuperan las direcciones de retorno después de los llamados a subrutina. Es importante señalar que, dado que se trata de una pila de solo 8 localidades, solo pueden anidarse hasta 8 llamados a subrutinas o interrupciones dentro del programa.

7. Memoria

Como se explicó anteriormente, la arquitectura HARVARD permite el uso de 2 bloques de memoria distintos con dos buses de datos separados para la memoria de **programa y de datos**. La memoria de programa es del tipo FLASH, con capacidad de programar y borrar hasta 10,000 veces, y cuenta con 1024 localidades de 14 bits, con direcciones de la 0000H a la 03FFH.

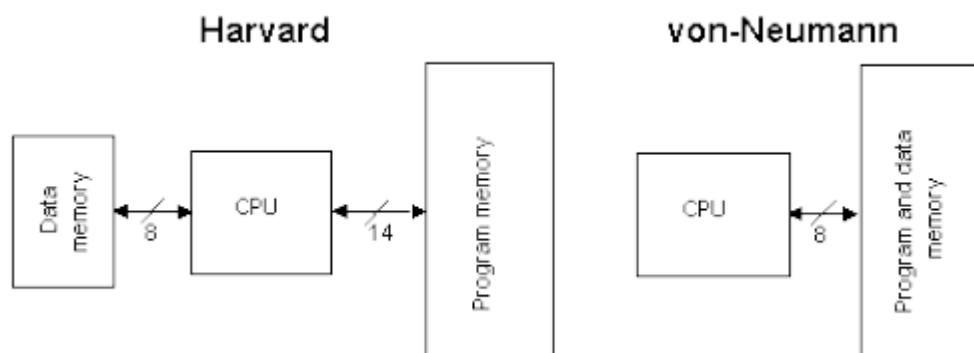
Las direcciones 0 y la 4 de esta memoria son usadas para los vectores de RESET e interrupción respectivamente, es decir, después de RESET, al contador del programa apunta hacia la dirección 0 y después de la ocurrencia de una interrupción (que se encuentre habilitada), entonces el contador del programa apuntará a la dirección 4.

La memoria de datos es de 8 bits, e incluye 68 localidades de RAM, y 64 localidades de EEPROM (con capacidad de programar y borrar hasta 10 millones de veces).

La memoria de datos RAM se encuentra mapeada debajo de los registros especiales del 16F84 (ver capítulo que sigue) en la dirección 0CH hasta la 4FH (el mapa se repite desde la 8CH a la CFH) y puede ser accesada a través de las instrucciones de MOVWF ó MOVF en forma directa o indirecta. Si se emplea direccionamiento indirecto, entonces deben de emplearse los registros especiales FSR (como apuntador de la localidad) e INDR (como destino de la lectura o almacenamiento previo del dato a escribirse)

La memoria EEPROM no puede ser accesada en forma directa, sino solo en forma indirecta a través de los registros EEADR, EEDATA, EECON1 y EECON2 y siguiendo un procedimiento especial para lectura y escritura de tal forma de evitar modificaciones accidentales del contenido de la memoria. La forma de accesar esta memoria se explica en el capítulo 10.

En el siguiente diagrama, se muestran los dos tipos de arquitectura usados en computadoras: la arquitectura Harvard y la Von Neumann.



8. Registros especiales.

File Address		File Address
00h	Indirect addr. ⁽¹⁾	80h
01h	TMR0	81h
02h	PCL	82h
03h	STATUS	83h
04h	FSR	84h
05h	PORTA	85h
06h	PORTB	86h
07h	—	87h
08h	EEDATA	88h
09h	EEADR	89h
0Ah	PCLATH	8Ah
0Bh	INTCON	8Bh
0Ch		8Ch
	88 General Purpose Registers (SRAM)	Mapped (accesses) in Bank 0
4Fh		CFh

FIGURA 7

Existen 2 bancos de registros denominados SFR (Special Function Registers), que permiten al programador acceder a algunas funciones del microcontrolador. En la tabla de abajo se detalla cada registro de estos dos bancos de memoria, ubicados en la RAM, desde la localidad 00H hasta la 0BH y de la 80H a la 8BH. El banco se selecciona mediante el bit RP0 del registro de STATUS. Si RP0 =0 se selecciona el banco 0 y si RP0=1, entonces se selecciona el banco 1. Muchos de los registros se encuentran repetidos en ambos bancos, por lo cual es indistinto leer o escribir sobre ellos desde cualquiera de los 2 bancos.

La función de cada uno de los registro es la siguiente:

INDF	REGISTRO QUE CONTIENE EL DATO LEIDO USANDO DIRECCIONAMIENTO INDIRECTO (ver FSR)
TMR0	REGISTRO QUE CONTIENE EL VALOR DEL CONTADOR/ TEMPORIZADOR (8 BITS)
OPTION REG	REGISTRO QUE PERMITE EL CONTROL DEL CONTADOR/ TEMPORIZADOR, DE LA INTERRUPCION EXTERNA Y DE LAS RESISTENCIAS DE PULL UP DEL PUERTO B.
PCL	PARTE BAJA DEL CONTADOR DEL PROGRAMA (8 BITS).
STATUS	GUARDA EL ESTADO DE LAS BANDERAS C (CARRY), DC (HALF CARRY), Z (ZERO), PD (POWER DOWN), TO (TEMPORIZADOR), RP0 (SELECTOR DE BANCO), RP1 (NO SE USA), RP2 (NO SE USA).
FSR	REGISTRO APUNTA USADO PARA EL DIRECCIONAMIENTO INDIRECTO DE LA MEMORIA DE DATOS. EL DATO, UNA VEZ LEIDO, SE ALMACENA EN EL REGISTRO INDF.
PORTA	PUERTO A
TRISA	REGISTRO DE SELECCIÓN DE BITS DE ENTRADA O SALIDA DEL PUERTO A
PORTB	PUERTO B
TRISB	REGISTRO DE SELECCIÓN DE BITS DE ENTRADA O SALIDA DEL PUERTO B.
EEDATA	ALMACENA EL VALOR LEIDO DE LA EEPROM, DE LA LOCALIDAD A DONDE APUNTA EEADDR.
EECON1	REGISTRO DE CONTROL HABILITA LECTURA Y ESCRITURA DE EEPROM.
EEADDR	APUNTA QUE ALMACENA LA DIRECCIÓN QUE HABRA DE LEERSE EN LA EEPROM

EECON2	REGISTRO DE CONTROL DE ESCRITURA. PROTEJE CONTRA ALTERACIONES INDESEADAS DEL CONTENIDO DE LA EEPROM.
PCLATH	PARTE ALTA DEL CONTADOR DEL PROGRAMA. 2 BITS, QUE JUNTO CON LOS 8 BITS DEL PCL, FORMAN LA DIRECCION COMPLETA CON LA CUAL PUEDE DIRECCIONARSE 1024 LOCALIDADES, DE LA 0000H A LA 03FFH. PCLATH PUEDE TAMBIEN VERSE COMO EL REGISTRO QUE CONTIENE LA PAGINA DE 256 BYTES EN DONDE HABRA DE DIRECCIONARSE LA MEMORIA.

8.1 Registros PCL y PCLATH:

En general estos registros son manipulados cuando se emplean tablas de datos (Look Up Tables).

PCLATH (PC Latch), puede modificarse a través de la instrucción MOVWF, pero su ejecución solo almacena el dato y **no modifica en forma inmediata** la parte alta del contador del programa y por lo tanto no produce ningún salto en el flujo del programa.

El registro **PCL** puede ser afectado por las instrucciones MOVWF ó ADDWF. Su ejecución **modifica directamente la parte baja del contador del programa y carga también el registro PCLATH en la parte alta**, e induce por tanto un salto inmediato a otra localidad. En resumen, al modificar PCL, debe de tenerse cuidado previamente de inicializar también correctamente PCLATH, pues de lo contrario el programa efectuará un salto a una localidad en una página no deseada.

8.2 Registro de Status:

En seguida se muestra un diagrama del registro de STATUS
REGISTRO STATUS (DIRECCION 03H, 83H)

IRP	RP1	RP0	TO	PD	Z	DC	C
Bit 7							Bit 0

- los bits 0, 1 y 2 son el CARRY, HALF CARRY Y ZERO, son banderas que se activan con un valor igual a 1, cuando el resultado de una operación o instrucción genera un carry, un half carry o un valor igual a cero respectivamente.
- El bit 3 se llama POWER DOWN y su valor es de 1 después de una instrucción CLRWD (CLEAR WATCH DOG TIMER) ó bien después de encender el sistema (POWER UP). El valor es de 0 después de ejecutar la instrucción SLEEP.
- El bit 4, se llama "TIMER OUT" tendrá un valor de 1 después de POWER UP, CLRWD ó SLEEP y tendrá un valor de 0 si el WDT (WATCH DOG TIMER) activa su señal de alarma.
- El bit 5, RP0, controla el banco de registros especiales SFR que habrá de accesarse. Si el bit tiene un valor de 1, se accesa el banco 1 y si tiene el valor de 0, se accesa el banco 0.
- Los bits 6 y 7 no se encuentran implementados.

8.3 Registro OPTION :

Este registro controla varias **funciones del temporizador** (bits 0..5), **de la interrupción externa** (bit 6), así como las **resistencias de PULL UP del puerto B** (bit 7). En seguida se muestra un diagrama del registro de opción.

REGISTRO OPTION (DIRECCION 81H)

RBPU	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0
Bit 7							Bit 0

- los bits 0, 1 y 2, toman un valor del 0 al 7 binario, y programan el divisor del temporizador y del WATCH DOG TIMER, de acuerdo a la siguiente tabla:

PS2	PS1	PS0	DIVISOR TMR0	DIVISOR WDT
000			1:2	1:1
001			1:4	1:2
010			1:8	1:4
011			1:16	1:8
100			1:32	1:16
101			1:64	1:32
110			1:128	1:64
111			1:256	1:128

- el bit 3, determina si el valor anterior se asigna al temporizador o al WDT. Si el valor es de 1, se asigna al WDT, si el valor es de 0, se asigna al temporizador.
- El **bit 4**, determina si el contador del temporizador se incrementa con el flanco ascendente (1) o con el flanco descendente (0) de la señal del pin 3, (RA4/T0CK1) del chip 16F84.
- El **bit 5** determina si la fuente de incremento del temporizador es la transición en el pin RA4/T0CLK1 (1) o el clock interno que maneja el ciclo de instrucción CLKOUT (0).
- El bit 6, determina, cuando su valor es de 1, que la interrupción externa se genera con el flanco ascendente del pin 6 del 16F84 (RB0/INT). Cuando su valor es de 0, entonces la interrupción se genera con el flanco descendente de la misma señal.
- El bit 7 determina, cuando su valor es de 1, que las resistencias de PULL UP en las salidas del puerto B estarán DESHABILITADAS. Si su valor es de 0, entonces dichas resistencias están HABILITADAS.

9. Registro temporizador/contador TMR0:

El registro TMR0 puede operar como un contador de los pulsos provenientes del bit RA4/T0CLK1 o como un temporizador. El modo de funcionamiento se selecciona con el bit 5 del registro de OPTION.

El bit 5 de OPTION debe de ponerse en 1 si se selecciona el **modo contador**. Al mismo tiempo, el bit 4 determina, como se explicó arriba, si la cuenta en el registro TMR0 se incrementa con el flanco ascendente o descendente del bit externo RA4/T0CLK1.

Cuando se selecciona el **modo temporizador**, entonces el bit 5 del registro de OPTION debe de ponerse en un 0. En este modo de operación, el registro TMR0 funciona junto con un PREESCALADOR. Este preescalador puede programarse para dividir la cuenta de ciclos de instrucción, entre el valor seleccionado en el registro OPTION (en los bits PS0, PS1 y PS2), de acuerdo a la tabla mostrada en la sección 7.2. En total, se pueden generar períodos de espera de hasta un máximo de 256 x 256 ciclos de instrucción ó 65,536 microsegundos = 65.5 milisegundos (operando a 4 Mhz).

Si el usuario desea manejar el registro con base en el sistema de interrupciones, la interrupción TMR0 se genera cuando el registro pasa de una valor de FFH a 00H. El mecanismo de operación de las interrupciones, usa los bits 2 y 5 del registro INTCON y se explica en el capítulo siguiente. Debe de tomarse en cuenta que si el procesador se encuentra en el modo SLEEP, entonces la interrupción TMR0 no despertará al procesador, ya que es deshabilitada durante ese modo.

10. Interrupciones del sistema.

El chip 16F84 cuenta con 4 posibles fuentes de interrupción asociadas a la ocurrencia de alguno de los siguientes eventos:

- La interrupción externa en el pin RB0/INT del chip, con flanco ascendente o descendente.
- El overflow en el temporizador del chip.
- Cualquier cambio de nivel en los pines RB4...RB7
- Cuando se ha completado la escritura de un dato en la EEPROM.

El vector de inicio de la subrutina de atención a interrupciones es la dirección 0004H. Después de la ocurrencia de una interrupción que se encuentre habilitada, el programa efectuará automáticamente un llamado a subrutina hacia esa dirección. Si se usan interrupciones dentro de su programa, es necesario entonces, en el programa principal, en la dirección 0000H, que es la dirección de inicio después de RESET, efectuar un salto a una localidad que se encuentre después de la subrutina de atención a las interrupciones del sistema.

El registro INTCON controla la habilitación y deshabilitación de estas interrupciones del sistema. Sus bits tiene las funciones que se indican enseguida.

REGISTRO INTCON (DIRECCION 0BH, 8BH)

GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF
Bit 7							Bit 0

- el **BIT 0** es una **bandera** que se pone en valor 1, si alguno de los bits del puerto RB4...RB7 cambió de valor y en 0 si ninguno de estos bits cambió su valor.
- el **BIT 1** es una **bandera** que se pone en 1, si se activa la interrupción externa (señal RB0/INT) del 16F84 y tomará un valor de 0 si no se activa dicha señal. En el registro de OPCION debe programarse si la interrupción se genera con al flanco ascendente o descendente.

- el **BIT 2** es una **bandera** que se pone en 1, si el contador del temporizador del circuito sufre un overflow, es decir, excede su cuenta máxima. Y en 0 si dicho contador no excede su cuenta máxima.
- en el **BIT 3**, deberá escribirse un valor de 1 para **habilitar** la interrupción de los bits RB4...RB7 (ver bit 0) y de 0 para deshabilitar dicha interrupción.
- en el **BIT 4** deberá escribirse un valor de 1 para **habilitar** la interrupción externa (ver bit 1) y de 0 para deshabilitar dicha interrupción.
- en el **BIT 5** deberá escribirse un valor de 1 para **habilitar** la interrupción del temporizador (ver bit 2) y de 0 para deshabilitar dicha interrupción.
- En el **BIT 6** deberá escribirse un valor de 1 para **habilitar** la interrupción de “escritura de un dato en la EEPROM completado” y un valor de 0 para deshabilitar dicha interrupción. El bit 4 del registro EECON1 es la bandera correspondiente que maneja la interrupción y se pone en un valor de 1, cuando está activa.
- el **BIT 7** corresponde al **habilitador GLOBAL** de las interrupciones (GIE). Debe de tener un valor de 1 para habilitar todas la interrupciones y de 0 para deshabilitarlas.

Cuando se genera una interrupción, el bit GIE es automáticamente puesto en 0 para impedir que se generen nuevas interrupciones. El contador del programa se carga con la dirección 0004H y la dirección de retorno es almacenada en el STACK o pila. Una vez que la subrutina de interrupción está ejecutándose (a partir de la dirección 0004H), la fuente de la interrupción puede ser determinada a través de un poleo en los bits 0, 1 y 2 del registro INTCON y bit 4 del registro EECON1. Dentro de la subrutina de atención a las interrupciones, debe también de escribirse un 0 en el bit de bandera correspondiente, para impedir que la misma interrupción vuelva a activarse una y otra vez.

11. Programación de la EEPROM:

La memoria EEPROM residente en el chip 16F84, posee 64 bytes. Dichas localidades pueden ser leídas o bien, puede escribirse en ellas durante la ejecución de un programa. Sus direcciones son de la 0 a la 3F H. El acceso a esta memoria es a través de **direccionamiento indirecto** utilizando 4 de los registros SFR (special function registers), que son los siguientes: EECON1, EECON2, EEDAT, y EEADR.

EEDAT guarda el dato que habrá de escribirse en la EEPROM, mientras que **EEADR** guarda la dirección. **EECON1** es un registro cuyos bits permiten al usuario habilitar o deshabilitar la lectura y escritura en la EEPROM. **EECON2** es un registro de control usado para evitar escrituras accidentales, de tal manera que deben escribirse en dicho registro los valores 55H y AAH, antes de iniciar un ciclo de escritura. En seguida se muestran los bits de control del registro EECON1:

REGISTRO EECON1 (DIRECCION 88H)

-	-	-	EEIF	WRERR	WREN	WR	RD
Bit 7							Bit 0

- para leer la EEPROM, debe de escribirse un 1 en el **BIT 0** del registro. Este bit tomará automáticamente el valor de 0 después de haberse efectuado la lectura del dato.
- Para iniciar el ciclo de escritura en la EEPROM, debe escribirse un 1 en el **BIT 1** del registro. Una vez terminado el ciclo de escritura, el bit tomará automáticamente el valor de 0.
- El **BIT 2** habilita o deshabilita el ciclo de escritura. Si vale 1, se habilita y si vale 0, se deshabilita.
- El **BIT 3**, es una bandera que anuncia, después de un ciclo de escritura, que ésta fue completada en forma errónea ó exitosa. Si el valor es 1, existió un error y si el valor es de 0, entonces la escritura fue completada sin error.
- El **BIT 4** es una **bandera de interrupción** asociada a la escritura en la EEPROM. Si el valor es de 1, el ciclo de escritura terminó. Si el valor es de 0, el ciclo de escritura no ha iniciado ó no ha concluído.
- Los bits 5, 6 y 7, no se usan.

Para leer o escribir en la EEPROM, debe seguirse la siguiente secuencia de instrucciones. Supongamos que deseamos leer el dato de la dirección 1AH de la EPROM, y almacenarlo en el registro W.

```
LECTURA:   BCF      STATUS,RP0      ;ELIJE EL BANCO 0 DE LOS REGISTROS ESPECIALES
            MOVLW   1AH        ;PREPARA LA DIRECCIÓN
            MOVWF   EEADR      ;ESCRIBE VALOR EN EL REGISTRO EEADR
            BSF     STATUS,RP0  ;ELIGE EL BANCO 1
            BSF     EECON1,RD   ;HABILITA EL BIT 0 (RD) DEL REGISTRO EECON1
            BCF     STATUS,RP0  ;HABILITA EL BANCO 0
            MOVF    EEDATA,W    ;LEE EL DATO EN EL REGISTRO EEDATA EN W,
                                ;USANDO DIRECCIONAMIENTO INDIRECTO.
```

Ahora supongamos que deseamos escribir en la dirección 2BH, el dato 1AH, previamente almacenado en W. El 16F84 cuenta con un mecanismo para proteger a la EPROM contra escrituras accidentales, mediante el cual deben primero escribirse los valores 55H y AAH en el registro EECON2 para habilitar el ciclo de escritura.

```

ESCRITURA:  BSF      STATUS,RP0      ;HABILITA EL BANCO 1
              BCF      INTCON,GIE      ;DESHABILITA INTERRUPCIONES
              BSF      EECON1,WREN     ;HABILITA ESCRITURA EN EEPROM
              MOVLW    55H             ;PREPARA SECUENCIA DE SEGURIDAD
              MOVWF    EECON2         ;ESCRIBE PRIMER DATO DE SECUENCIA
              MOVLW    AAH             ;SEGUNDO DATO
              MOVWF    EECON2         ;ESCRIBE SEGUNDO DATO DE SECUENCIA
              BSF      EECON1,WR      ;INICIA CICLO DE ESCRITURA
              BSF      INTCON,GIE      ;HABILITA INTERRUPCIONES
  
```

12. Funciones especiales:

12.1 REGISTRO DE CONFIGURACION:

El 16F84 cuenta con un registro de configuración, que solamente puede accesarse durante el ciclo de programación del chip. Su dirección es la 2007H. Los bits de configuración controlan la operación de diversas funciones, como el oscilador, la protección de código, el WATCH DOG, y el POWER ON RESET.

En seguida se muestran los bits de este registro:

REGISTRO CONFIGURACION (DIRECCION 2007H)

CP	CP	CP	CP	CP	CP	CP	CP	CP	CP	PWRTE	WDTE	FOSC1	FOSC0
Bit 13											Bit 0		

los bits 1 y 0: sirven para seleccionar el modo de operación del oscilador.

CLAVE EN BITS 1-0	TIPO DE OSCILADOR	FRECUENCIA DE OPERACION
00	CRISTAL LP	32 KHZ...200 KHZ
01	CRISTAL XT	100 KHZ...4 MHZ
10	CRISTAL HS	4 MHZ...20 MHZ
11	TIPO RC RESISTENCIA CAPACITOR	

bit 2: sirve para habilitar el WDT (watch dog timer). Si el valor es de 1, habilita el WDT, si es de 0, lo deshabilita.

bit 3: sirve para habilitar el power up timer PWRT, si el bit vale 1, lo deshabilita y si vale 0, lo habilita.

bits 4...13: sirven para habilitar la protección de código o deshabilitarla. Si los valores de éstos bits son de 1, la protección queda deshabilitada y si son de 0, entonces la protección es habilitada y no puede leerse la memoria de código o de programa.

12.2 OSCILADOR

El circuito 16F84 puede funcionar con un oscilador proveniente de un cristal, o bien con oscilador simple proveniente de un arreglo de resistencia-capacitor. En la tarjeta EDU-PIC, ya se encuentra instalado el cristal de 4.0 Mhz (CRISTAL XT). Además, deben de programarse correctamente los bits del registro de configuración, de acuerdo a la tabla anotada arriba. Recuerde que este registro solo puede modificarse al momento de escribirse el programa fuente .ASM.

12.3 POWER ON RESET

El circuito 16F84 cuenta con un arreglo RC interno, independiente del oscilador principal, que genera el pulso de reset en el momento que se conecta la energía a Vdd (además, el pin 4 del 16F84, MCLR debe estar conectado a 5 volts). Adicionalmente, al seleccionar en el registro de configuración la opción power up timer, PWRT=0, se mantiene el pulso de reset activo hasta después de 72 ms después de haber conectado la energía. En el caso que se esté usando un oscilador de cristal, se genera automáticamente un retraso adicional de 1024 pulsos de reloj, antes de que el pulso de reset termine. Estos retrasos permiten la estabilización del cristal antes de que el microcontrolador inicie su operación. Después de un reset, el microcontrolador inicia en la dirección 0000H la ejecución de su programa.

12.4 WATCH DOG TIMER

El WDT, es un circuito de vigilancia que permite generar un pulso de **reset automático** en caso de que el 16F84 se salga de operación por alguna inestabilidad en el voltaje de alimentación en su fuente de poder ó alguna falla en la ejecución del programa. La función es sumamente importante para **evitar que el sistema necesite intervención manual** externa para dar reset al procesador. El WDT funciona como un contador de eventos cada 18 ms, el cual genera un reset al sistema cuando la cuenta llega a un máximo y genere un **TIMEOUT**.

La programación del WDT, **debe de activarse desde el registro de configuración**,. Esto se hace escribiendo en el bit 2, WDTE=1. La dirección del registro de configuración, es la 2007H. Debe recordarse que el registro de configuración no puede accesarse desde el programa ejecutable del microcontrolador, sino directamente debe modificarse el valor en el programa fuente .ASM el cual debe ensamblarse junto con el nuevo programa que se envíe a la memoria flash del chip. Para ello, se cuenta con una directiva en el programa ensamblador que realiza esta función. En seguida se muestra un ejemplo:

```
_config _XT_OSC & _WDT_ON & _PWRTE_ON
```

Además, desde el programa ejecutable, el bit 3 del registro OPTION, debe de programarse como PSA=1, para asignar el valor del preescalador al WDT. Adicionalmente, en los bits PS0, PS1, PS2 del registro OPTION debe escribirse, desde el programa, un valor entero del 0 al 7. Cualquier valor diferente a 0, eleva el período de activación del WDT a 18 milisegundos, multiplicado por 2 elevado a ese valor, de acuerdo a la tabla mostrada en la figura. Por ejemplo, si el valor de los bits PS0, PS1 y PS2 es de 5, el período de TIMEOUT será de $18\text{ms} \times 32 = 576 \text{ ms}$. El TIMEOUT máximo para el WDT es de 2.3 segundos.

Una vez que el WDT está activado, a través de la instrucción CLRWDT, se reinicia desde 0 su período de activación. Entonces dicha instrucción debe de ejecutarse regularmente dentro de la malla principal en el programa, con un período que debe de ser MENOR al TIMEOUT programado para el WDT. Cuando por alguna causa de malfuncionamiento del 16F84 el programa se sale de su operación normal y por consecuencia la instrucción CLRWDT no se ejecuta, entonces, al llegar a un máximo la cuenta en el WDT (TIMEOUT), el circuito genera un RESET que reinicia la operación del 16F84.

12.5 SLEEP

El 16F84 cuenta con una función que le permite operar en un modo de muy bajo consumo, por ejemplo en el caso de un sistema con alimentación de energía solar ó pilas. Si se tiene una aplicación en la cual el microcontrolador no desempeña ninguna función útil hasta la ocurrencia de alguna interrupción, puede abatirse el consumo promedio del circuito a niveles cercanos a 0 ma (1 uA). La función de SLEEP se habilita con la instrucción del mismo nombre. A partir de su ejecución, los circuitos del oscilador maestro cesan de funcionar, siendo de esta forma el consumo de corriente de casi cero. Solamente la ocurrencia de alguna interrupción externa en el pin RB0/INT, la interrupción por algún cambio en los niveles de las entradas en el puerto B, la interrupción proveniente de la EEPROM, ó bien un reset en el pin MCLR del 16F84 puede restaurar la operación normal del circuito. Antes de entrar al estado de SLEEP, debe de inhibirse la operación del WDT para evitar que éste reactive al circuito a través de su reset automático.

12.6 CODE PROTECT

El microcontrolador 16F84 cuenta con una opción para evitar, de ser necesario, que alguna persona pueda copiar el código del programa contenido en la memoria FLASH del chip. Si usted desea proteger su programa entonces deberá, antes de programar el chip, poner en valor =0 los bits del 4 al 13 del registro de configuración. Sin embargo, debe de tenerse cuidado de no manipular indebidamente estos bits, ya que, una vez habilitado el modo "CODE PROTECT" será imposible acceder de nuevo el código almacenado en la memoria FLASH. También es importante señalar que un chip que ha sido protegido, no puede ser leído, pero sí puede ser borrado y reprogramado de nuevo.

13. Puertos digitales :

El sistema 16F84 cuenta con dos puertos digitales, el puerto A, con 5 bits disponibles y el puerto B con 8 bits disponibles. Ambos puertos son bidireccionales, ésto es, pueden programarse como entradas o como salidas, de acuerdo a unos registros de dirección de datos, llamados "TRIS", en el caso del puerto A es "TRISA" y del puerto B es "TRISB". En la tarjeta EDUPIC, le han sido conectados entradas con 4 microswitches para el puerto A y salidas de 8 LEDS para el puerto B, así como un relevador conectado al pin RA0. La asignación de funciones en cada uno de los bits, se muestra en la siguiente tabla. Por favor tome nota de que cada entrada y salida se encuentra también disponible en el HEADER de 16X en la tarjeta, de tal manera que el usuario pueda conectar dispositivos externos.

PUERTO	PIN EN HEADER 16x	FUNCION
PUERTO A		
RA0	PIN 1	ACTIVA/DESACTIVA RELEVADOR
RA1	PIN 3	MICROSWITCH A1, CONTROL LCD
RA2	PIN 5	MICROSWITCH A2, CONTROL LCD
RA3	PIN 7	MICROSWITCH A3
RA4	PIN 9	MICROSWITCH A4
PUERTO B		
RB0	PIN 2	LED B0, TECLADO Y1
RB1	PIN 4	LED B1, TECLADO Y2
RB2	PIN 6	LED B2, TECLADO Y3
RB3	PIN 8	LED B3, TECLADO Y4
RB4	PIN 10	LED B4, TECLADO X1
RB5	PIN 12	LED B5, TECLADO X2
RB6	PIN 14	LED B6, TECLADO X3
RB7	PIN 16	LED B7, TECLADO X4

Antes de poder escribir y leer de los puertos, es necesario primero programar qué bits serán entradas y salidas, usando las siguientes instrucciones:

```

PUERTOA    BSF      STATUS,RP0      ;ELIJE EL BANCO DE REGISTROS ESPECIALES 1
           MOVLW   0x1E              ;RA0=SALIDA, RA1..RA4=ENTRADAS
           MOVWF  TRISA              ;PROGRAMA LA DIRECCIÓN DE LOS BITS

PUERTOB    BSF      STATUS,RP0      ;ELIGE EL BANCO DE REGISTRO ESPECIALES 1
           MOVLW   0x00              ;RB0..RB7=SALIDAS
           MOVWF  TRISB

```

13.1 LEDS Y MICROSWITCHES:

Una vez inicializados los puertos de la forma mostrada, se puede desde el programa escribirse en los LEDS o leer desde los microswitches, considerando los diagramas electrónicos que se muestran enseguida. El objetivo de los LEDS y los microswitches es dar al usuario la posibilidad de realizar emulaciones de sensores digitales y salidas para la activación de actuadores. Todas las señales de los puertos están disponibles en el conector header de 14x, para su conexión a interfaces externas.

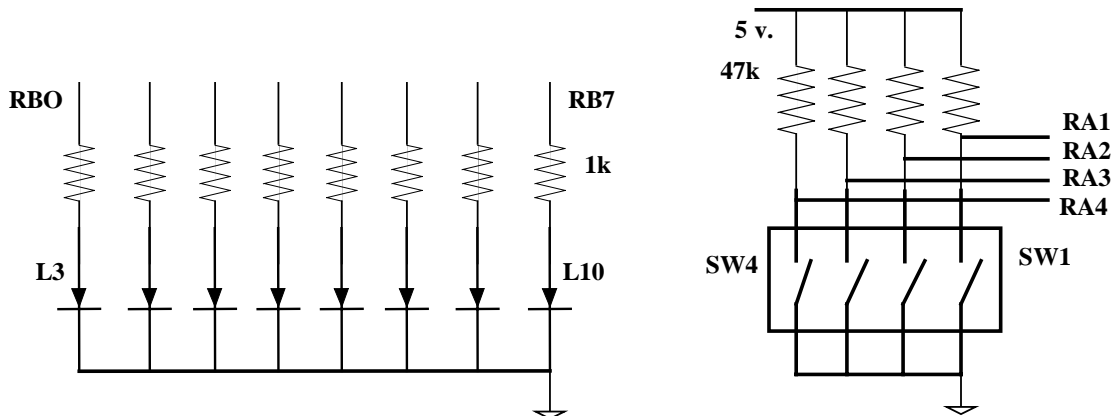


FIGURA 8: DIAGRAMA DE LOS LEDS Y LOS MICROSWITCHES

13.2 RELEVADOR

Mediante el manejo del bit RA0 del puerto A, el procesador 16F84 puede activar un relevador integrado en la tarjeta EDU-PIC. Los datos nominales de este relevador son : un polo un tiro, activación con 12 volts DC y contactos de 127 VCA @ 10

Amperes. Este relevador puede ser usado ya sea como un sensor digital ABIERTO- CERRADO para alertar a otros dispositivos del estado de alguna alarma, o bien como actuador para activar dispositivos externos como focos, válvulas, solenoides, motores, etc.

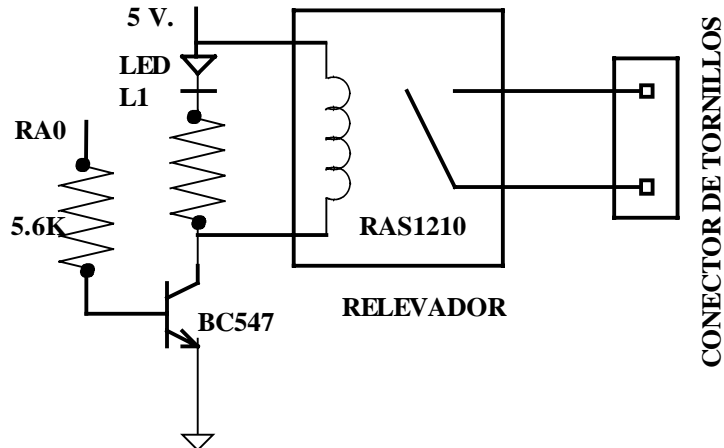


FIGURA 9: CONEXIÓN DEL RELEVADOR

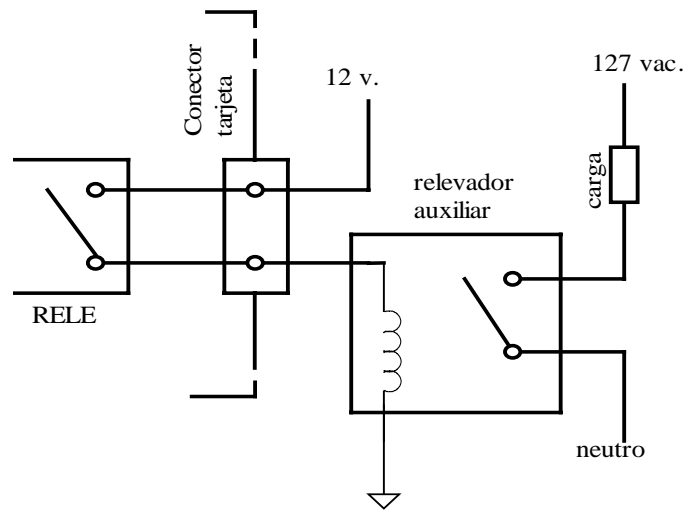


FIGURA 10: CONEXIÓN DE UN RELEVADOR AUXILIAR

13.3 HEADER DE 16 X. (CONECTOR PARA UN TECLADO MATRICIAL).

Las señales de los puertos descritas en el subcapítulo de arriba, como se explicó, están disponibles en un conector header 16x. En seguida se muestra el diagrama de conexiones de dicho conector. Por favor tome nota de que el orden de los pines es diferente a los de un circuito integrado, siendo una fila de pines nones y la otra de pines pares. Los pines del puerto B están disponibles en toda una hilera del header, para conectar en forma inmediata un teclado matricial de 4 x 4 (ver apéndice 1)

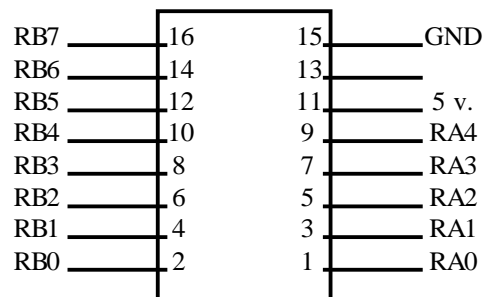


FIGURA 11

13.4. CONECTOR A LCD:

La tarjeta EDUPIC ofrece un conector estándar para la conexión de un dispositivo LCD. Este conector dispone de 14 señales, mostradas en la tabla de abajo. **Por favor consulte el apéndice 1 de este manual para información completa sobre la conexión de su LCD.** Es importante señalar que las señales RA1, RA2, RB0...RB7 (un total de 10 señales) se encuentran también disponibles en el conector HEADER 16x. Es posible usar el conector estándar de 14 pines para conectar un LCD y, en paralelo, conectar un teclado matricial al header de 16X, a pesar de que ambos dispositivos comparten las señales RB0...RB7, gracias a que pueden operar en tiempos distintos. Por favor vea los programas de prueba para el LCD y teclado matricial (apéndice 1).

LCD	EDUPIC	FUNCION	LCD	EDUPIC	FUNCION
1		Tierra	8 DB1	RB1	DATOS
2		5 volts.	9 DB2	RB2	DATOS
3 INT		Control de Intensidad	10 DB3	RB3	DATOS
4 RS	RA1	0=comando 1=datos	11 DB4	RB4	DATOS
5 R/W	TIERRA	0=escribir en LCD 1=leer	12 DB5	RB5	DATOS
6 EN	RA2	Enable modo pulso	13 DB6	RB6	DATOS
7 DB0	RB0	DATOS	14 DB7	RB7	DATOS

14 El set de instrucciones.

Existen un total de 35 instrucciones. Todas las instrucciones son palabras de 14 bits, divididas en dos partes: el código de operación y el operando. Los operandos, es decir los destinatarios de la operación a realizarse pueden ser bytes o bits de memoria o registros. De esta forma se puede hablar de instrucciones “orientadas a bytes” u “orientadas a bits”.

Todas las instrucciones, con excepción de las que modifican el contenido del contador del programa (como son los saltos y llamados a subrutina) se ejecutan en un ciclo de instrucción, es decir, 4 ciclos de reloj. Para un sistema funcionando a 4 Mhz, cada instrucción se ejecuta en 1 microsegundo. Si la instrucción modifica el contenido del contador del programa, entonces el tiempo de ejecución es de 2 ciclos de instrucción ó 2 microsegundos. El grupo de 35 instrucciones es el siguiente:

MOVE GROUP

movf	f,d	move f
movwf	f	move w to f
movlw	k	move literal to w
clrf	f	clear f
clrw		clear w
swapf	f,d	swap nibbles in f

ARITHMETIC GROUP

addwf	f,d	add w and f
addlw	k	add literal to w
subwf	f,d	subtract w from f
sublw	k	subtract w from literal
incf	f,d	increment f
incfsz	f,d	increment f, skip if 0
decf	f,d	decrement f

decfsz **f,d** **decrement f, skip if 0**

LOGIC GROUP

andwf **f,d** **and w and f**
andlw **k** **and literal to w**
iorwf **f,d** **inclusive or w and f**
iorlw **k** **inclusive or literal to w**
xorwf **f,d** **exclusive or w and f**
xorlw **k** **exclusive or literal to w**
comf **f,d** **complement f**
rlf **f,d** **rotate left f, through carry**
rrf **f,d** **rotate right f, through carry**

BIT GROUP

bcf **f,b** **bit clear in f**
bsf **f,b** **bit set in f**
btfsc **f,b** **bit test in f, skip if clear**
btfss **f,b** **bit test in f, skip if set**

CONTROL GROUP

clrwdt **clear watchdog timer**
sleep **go into sleep mode**
nop **no operation**

BRANCH GROUP

goto **k** **goto address**
call **k** **call subroutine**
return **return from subroutine**
retlw **k** **return with literal in w**
retfie **return from interrupt**
incfsz **f,d** **increment f, skip if 0**
decfsz **f,d** **decrement f, skip if 0**
btfsc **f,b** **bit test in f, skip if clear**
btfss **f,b** **bit test in f, skip if set**

14.1 OPERANDOS:

Los operandos pueden asignarse con las letras F, W, B, K, D. Cada letra tiene el significado siguiente:

F: designa alguna localidad de memoria (file register), entre 00H y 0CH ó 80H y 8CH de alguno de los dos bancos de los llamados “registros especiales”, o bien, alguna de las 68 localidades de memoria RAM, ente la 0C y la 4F.

W. designa el acumulador del 16F84.

B. designa alguno de los 8 bits del registro especial o localidad de memoria elegido.

K. designa una constante ó una dirección.

D. designa el destino de la operación. Si D=0, el destino es el registro W. Si D=1, entonces el destino es el registro ó localidad de memoria F.

Directivas del programa ensamblador:

Dentro del archivo del programa fuente, es decir del programa escrito en lenguaje ensamblador, se puede, con ayuda de la directiva EQU, definir previamente algunos valores para la facilidad de su identificación. Usualmente algunas de las definiciones son las siguientes:

W	EQU	H'0000'	PORTA	EQU	H'0005'
F	EQU	H'0001'	PORTB	EQU	H'0006'
STATUS	EQU	H'0003'	TRISA	EQU	H'0085'
RP1	EQU	H'0006'	TRISB	EQU	H'0086'
RP0	EQU	H'0005'	PCLATH	EQU	H'000A'

Estas declaraciones, junto con las del resto de los registros, así como los bits individuales de cada registro, se almacenan en un solo archivo que se denomina 16f84.inc y que es parte de las librerías ya incluídas en el programa MPLAB, del cual se habla más adelante. Entonces, es suficiente escribir, dentro del programa fuente en lenguaje ensamblador, la directiva:

include <p16f84.inc>

Y de esta forma, el usuario podrá escribir su programa en forma mucho más entendible, puesto que usará ahora los nombres de los registros y los bits y no solamente números.

Una vez establecidas estas equivalencias, podemos poner algunos ejemplos con instrucciones. Observe que, en todas ellas es posible a veces usar las equivalencias o bien escribir directamente el valor numérico. Por ejemplo:

<u>Etiqueta</u>	<u>Instrucción</u>	<u>Operando</u>	<u>Forma general:</u>
EJEMPLO1	BCF	STATUS,RP0	BCF F,D

Resultado: Bit Clear F. Pon en cero el bit RP0 del registro STATUS.

EJEMPLO2	BCF	3,5
----------	-----	-----

Resultado: mismo que en el ejemplo anterior, pero usando ahora las constantes directamente al escribir la instrucción. Observe como el hecho de escribir directamente palabras como “STATUS” en lugar del número 3, facilitan mucho la comprensión.

EJEMPLO3	BTFSS	STATUS,RP1	BTFSS F,B
----------	-------	------------	-----------

Resultado: Bit Test F, Skip if Set. Si el bit RP1 del registro designado es cero, ejecuta la siguiente instrucción, si el bit es 1, entonces no ejecuta la siguiente instrucción, pero sí la que sigue a ésta.

EJEMPLO4	ADDLW	3AH	ADDLW K
----------	-------	-----	---------

Resultado: Add literal to W. Suma el registro W con la constante 3AH. El resultado lo pone en W.

EJEMPLO5	DECFSZ	20H,W	DECFSZ F,D
----------	--------	-------	------------

Resultado: Decrement F, Skip if Zero. Decrementa el valor de la localidad 20H. El resultado lo almacena en el registro W. Si el resultado es 0, no ejecuta la siguiente instrucción, sino la inmediata después de ésta. Si el resultado es diferente de 0, entonces ejecuta la siguiente instrucción.

EJEMPLO6	DEC	2AH,W	DEC F,D
----------	-----	-------	---------

Resultado: Decrementa la localidad 2AH, el resultado lo pone en el registro W.

EJEMPLO7 ANDWF 2B,F ANDWF F,D

Resultado: hace la operación lógica AND entre la localidad 2B y el registro W. El resultado lo pone en la localidad 2B.

EJEMPLO8 BTFSC 35H,3 BTFSC F,B

Resultado: Bit Test F, Skip if Clear. Hace una prueba sobre el bit 3 de la localidad 35H. Si el valor es 0, no ejecuta la siguiente instrucción, pero sí la siguiente. Si el valor es 1, entonces ejecuta la siguiente instrucción.

14.2 FORMATO DE LAS INSTRUCCIONES:

Todas las instrucciones llevan alguno de los siguientes formatos, dependiendo de la función que desempeñen: Instrucciones orientadas a byte. Instrucciones orientadas a bit. Instrucciones de manejo de constantes (en la literatura de Microchip, las constantes se denominan “literals”) y, finalmente instrucciones de salto CALL y GOTO.

Enseguida, se muestra el formato para cada tipo de instrucción. Las palabras son de 14 bits. Obviamente el trabajo de decodificación de cada formato es realizado por el programa ensamblador, razón por la cual el programador no requiere de decodificar manualmente cada instrucción..

INSTRUCCIONES ORIENTADAS A BYTES: 13 7 0

OPCODE	d	f	(file register)
--------	---	---	-----------------

Si d=0, destino es W, si d=1, destino es f

INSTRUCCIONES ORIENTADAS A BITS: 13 9 8 7 0

OPCODE	b	(bit)	f	(file register)
--------	---	-------	---	-----------------

b selecciona el bit del registro f, (valor de 0 a 7)

INSTRUCCIONES MANEJO DE CONSTANTES: 13 8 7 0

OPCODE	k (literal)
--------	-------------

k es la constante en la instrucción.

INSTRUCCIONES CALL Y GOTO: 13 11 10 0

OPCODE	k (literal)
--------	-------------

k es la dirección inmediata en 11 bits.

14.3 MANEJO DE TABLAS:

Es importante recordar que el microcontrolador 16F84 opera con una arquitectura HARVARD. Lo anterior hace que el manejo de tablas sea distinto a procesadores con arquitectura VON NEUMANN, en donde la memoria de código y datos es compartida, y entonces, a través de direccionamiento indirecto es posible recuperar los datos de la tabla. En el caso de la arquitectura Harvard, la tabla se encuentra en la memoria de código (programa) y debe entonces de manejarse como parte del programa ejecutable. La forma de resolverlo es como se indica en el siguiente ejemplo. La tabla forma parte de una subrutina que en este ejemplo lleva, predictiblemente, la etiqueta TABLA.

Supongamos que deseamos crear una tabla de 4 datos. Y que estos 4 datos son los caracteres ASCII de las letras H,O,L y A. Dentro del programa principal, existirá una instrucción de “call” que llama a la etiqueta que está al inicio de la tabla, como se muestra en el listado de abajo.

Una vez que el el contador del programa queda posicionado al inicio de la tabla, usamos la instrucción addwf, para sumar a la parte baja del contador del programa, PCL, un número del 1 al 4, que debe de estar almacenado en el registro W desde antes de la ejecución de la instrucción de “call”. Al ejecutarse dicha instrucción (addwf), el contador del programa queda posicionado en la localidad deseada de la tabla.

Ahora, mediante la instrucción “retlw” (return from subroutine with literal in w), regresamos al programa principal, pero el registro W, contiene ahora el dato que aparece a la derecha de la instrucción (alguno de los códigos ASCII de la H,O,L ó A), y los cuales forman parte de la tabla.

Es decir que el registro W contiene, antes de la instrucción de “call”, el desplazamiento (offset) deseado sobre la dirección de inicio de la tabla, (que en el ejemplo, puede ser un valor del 1 a 4). Aquí es muy importante señalar que en la suma (addwf), se está afectando la parte baja PCL (8 bits) y también la alta (2 bits) del contador del programa, la cual se carga con el

contenido de PCLATH. Es necesario entonces cargar también el registro PCLATH para que contenga el valor de la página en donde se encuentra la tabla, que pueden ser los valores 0, 1 2 ó 3.

```

...main...
movlw 1           ;en este ejemplo, la tabla se encuentra en la página 1.
movw,f PCLATH    ;carga el número de página en PCLATH
.....          ;en esta zona del programa, debe de inicializarse W con el desplazamiento.
call TABLA
.....          ;en ésta línea regresa la subrutina TABLA con el dato de la tabla en W.
.....
TABLA addwf PCL   ;ésta instrucción suma a PCL, el contenido de W, y carga PCLATH en la parte al
                ;ta del contador del programa.
retlw 'H'       ;regresa de la subrutina con el código ASCII de la "H" en W.
retlw 'O'
retlw 'L'
retlw 'A'

```

15. Puesta en marcha:

PASO 1: REVISION DE COMPONENTES Y PRUEBA INICIAL DE LA TARJETA EDUPIC:

Revise por favor que su kit incluya lo siguiente: tarjeta EDU-PIC, eliminador de baterías, cable serial, y disco CD de aplicación. Para poder usar el disco de aplicación, usted necesita una computadora con Windows98 o bien un sistema operativo más reciente, incluyendo XP, con una unidad lectora de CD, y un puerto serial con conector DB9. Mueva el selector del eliminador de baterías a 7.5 volts y conéctelo a su tarjeta EDUPIC.

Tome como auxiliares para la realización de este punto, a la **figura 12 y la figura 17 (foto) al final del manual**. Los jumpers en EDUPIC S1, S2, S3, deben de estar en la posición AUTORUN. Debido a que existen dos señales de control del LCD compartidas con los microswitches, para la realización de esta prueba, los **SWITCHES A1 y A2 deben de estar en OFF (ABIERTOS)**, es decir, **en su posición más cercana al puerto serial**. Una vez alimentada la tarjeta, el programa cargado de fábrica en la memoria FLASH del 16F84, que escribe un mensaje en el LCD, debe de funcionar en forma inmediata, encendiendo y apagando los LEDS en forma secuencial. No es necesario que el módulo LCD esté conectado. Si el programa funciona correctamente, puede usted continuar con la puesta en marcha.

PASO 2: INSTALACION DE LOS PROGRAMAS DEL DISCO DE SOPORTE:

Genere una carpeta denominada EDUPIC en el escritorio de su PC. Ahora inserte su disco de soporte EDU-PIC, en su unidad lectora de CD de su computadora. Abra el contenido del disco y verifique su contenido. Copie, desde la unidad lectora del CD, todos los archivos hacia la carpeta EDUPIC.

Hay dos archivos comprimidos: MPLAB660.zip y, dentro de la carpeta WINPIC, winpicpr.zip y por esta razón es que también se anexan en el disco los archivos auxiliares para descomprimir (WINZIP, PKREADER). El resto de los archivos son información didáctica, programas tutoriales o manuales de algunos dispositivos, programas y estándares empleados. Se incluye también el Acrobat Reader para poder leer los archivos en formato .pdf. En el archivo "readme" existe una lista completa de los archivos contenidos en el disco CD y su aplicación.

Ahora ejecute el archivo comprimido MPLAB660.zip y, una vez descomprimido, ejecute su programa de instalación. Ponga por favor atención en las siguientes ventanas que aparecerán durante la instalación:

Título de la ventana:	Usted elige:	
SOFTWARE LICENSE	I ACCEPT	NEXT
SELECT DESTINATION	MPLAB IDE	NEXT
SHORTCUTS?	YES	NEXT
ADD TO DESKTOP?	YES	NEXT
VIEW READ ME FILES?	NO	NEXT

Después de esta ventana, el programa trata de buscar algún dispositivo de hardware compatible con MPLAB, como son programadores y emuladores, a través del puerto USB, para instalar sus drivers. Como no es el caso, simplemente ignore y cierre todas las ventanas que aparezcan con respecto a estos drivers para USB, hasta que aparezcan las siguientes ventanas:

INSTALATION COMPLETED FINISH

Aquí la máquina es reiniciada para completar la instalación. Una vez que se complete el proceso de inicio, aparecerá automáticamente en el escritorio un ícono de acceso directo de MPLAB IDE.

MPLAB660 es un programa de desarrollo que integra funciones de edición, ensamblado y simulación de programas e incluso puede integrar un compilador de C. Una vez que la etapa de desarrollo se ha completado en MPLAB660, los archivos ejecutables .HEX generados, son usados por otro programa llamado WINPIC para programar la memoria FLASH del procesador 16F84 en la tarjeta EDU-PIC a través del cable serial de la computadora.

Para terminar con la instalación de programas, abra ahora la carpeta WINPIC, en donde encontrará el programa comprimido winpicpr.zip. Abra este archivo y elija la opción “extract” de todos los archivos para ser instalados en el mismo subdirectorio WINPIC. Dé click a la opción “yes to all” si aparece un mensaje de WINZIP preguntando por el reemplazo de los archivos. Haga un acceso directo al escritorio del archivo winpic.exe.

PASO 3: QUITAR PROTECCION DE SOLO LECTURA A LOS ARCHIVOS.

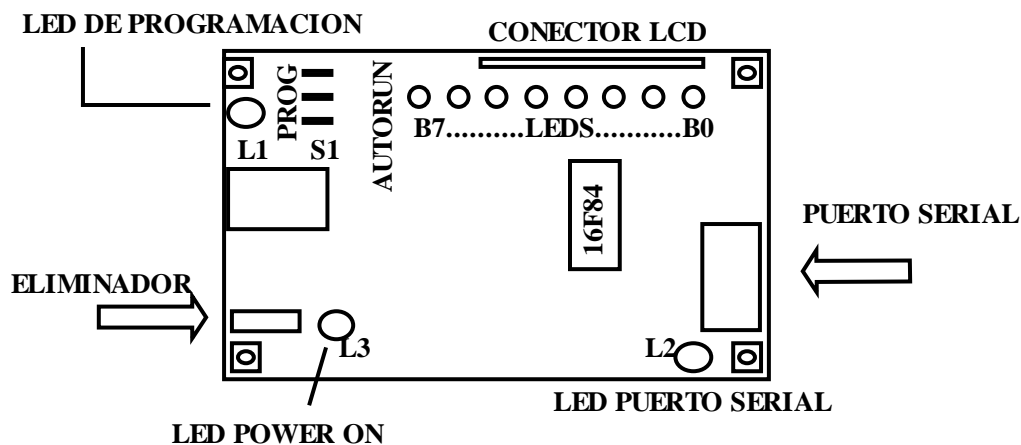
Antes de poder editar y ensamblar cualquier programa en MPLAB, es necesario quitar la protección de solo lectura de los archivos de prueba que se encuentran en el subdirectorio EDUPIC\archivos16F84. Estos archivos traen esa protección debido a que fueron copiados directamente de un CD de solo lectura.

Abra el subdirectorio “archivos16F84”, y observe los archivos allí contenidos. Ahora quite la protección de solo lectura en los archivos mediante el siguiente procedimiento: con la carpeta “archivos16F84” abierta, seleccione “edit” y luego “seleccionar todo”. Una vez que todos los elementos aparezcan sombreados, dé click en la parte derecha del mouse sobre cualquiera de los íconos y seleccione la opción “propiedades”. Quite allí la selección de la opción “solo lectura”, dejando el cuadrado en blanco. Luego dé click en “aplicar” y “cerrar”. De esta forma todos los archivos quedarán ahora disponibles para su edición en MPLAB. Si omite este paso, MPLAB no podrá editar ni ensamblar ningún archivo, y enviará mensajes de error al intentarlo.

Los archivos en la carpeta “archivos16F84” son programas de prueba para los diferentes dispositivos de la tarjeta EDUPIC. Los nombres son los siguientes: contador, demof84, led, microsw, lcdmensaje, lcd1, lcd2, LCD 16x1, LCD16X1 V.2, LCD16X1 V3, LCD16X1 V4, teclado1, teclado2, teclado3, teclado4, RTC, RTC V2, RTC V3. Las terminaciones son:

Terminación .ASM que son los archivos fuente. Estos archivos son solamente de texto y están escritos en lenguaje ensamblador para el microcontrolador 16F84. **Terminación .LST** que son los archivos ensamblados y que incluyen el texto fuente, las localidades de memoria con su código de máquina y los mensajes de error. Los archivos con **terminación .HEX** son archivos objeto listos para su transferencia a la tarjeta EDUPIC a través del programa WINPIC. Existe también un archivo de nombre **16f84.inc**, el cual contiene las directivas del ensamblador necesarias para escribir rápidamente un programa haciendo referencia a los registros y a los bits individuales de los registros del 16F84. Más abajo se explica con más claridad el uso de este archivo.

Cada vez que un archivo .ASM es ensamblado por MPLAB, se crean automáticamente 4 nuevos archivos. Por ejemplo, si el archivo contador.ASM es ensamblado, al final del proceso aparecerán en el mismo subdirectorio los siguientes archivos: contador.ASM, contador.COD, contador.ERR, contador.LST y contador.HEX. Si aparece algún mensaje de error durante el proceso de ensamblado, el usuario deberá revisar el archivo .LST para encontrar el número de línea en donde ocurrió el error, para después corregirlo en el archivo .ASM y volver a repetir el ensamblado.



PASO 4: CONEXIÓN DE LA TARJETA A LA COMPUTADORA:

Por favor identifique las componentes indicadas en la figura 12 para la tarjeta EDU-PIC, en particular, el microcontrolador 16F84, el conector del eliminador de baterías y el conector del cable serial. Observe los letreros de **PROGRAMAR** y **AUTORUN** en la tarjeta. Ambos indican la posición que deben de tener los 3 jumpers ya sea para trabajar en modo de programación del 16F84, o bien para correr el programa previamente almacenado.

Coloque los 3 jumpers en su posición más cercana al letrero **PROGRAMAR**. Ahora conecte el eliminador de baterías (con el selector del voltaje en 7.5 volts) y el cable serial a los conectores indicados. El conector DB9 del lado opuesto del cable serial, debe de conectarse al COM1 ó COM2 de su computadora. El led que se encuentra al lado del conector del eliminador, deberá encenderse. El LED que se encuentra junto al conector DB9 se enciende con algunos sistemas operativos y con otros no.

PASO 5: CONFIGURAR EL PROGRAMA WINPIC Y HACER LA TRANSFERENCIA DEL ARCHIVO.

Ejecute el programa WINPIC desde su escritorio. Para configurarlo, seleccione la solapa “device,config” y allí la opción “part”=“16F84A” y luego seleccione “oscillator”=“XT, XTAL”, “code protection”=“off”, “power up timer”=“disabled”, y “watch dog enable”=“disabled”. En la solapa “interface”, “interface type”=“JDM programmer”, “on port”=“COM1”.

Observe que en la tarjeta existen 3 jumpers: S1, S2 y S3. Ponga S1, S2 y S3 en posición de “**PROGRAMAR**”.

Nota importante: los jumpers S1, S2 y S3 no deben moverse de su posición de “**PROGRAMAR**” mientras la tarjeta esté conectada al puerto serial de la computadora PC, y únicamente deberán de moverse hacia la posición “**AUTORUN**”, cuando la tarjeta esté funcionando en modo autónomo, ya que, en esa posición, las señales RB6 y RB7, así como MCLR quedan completamente desconectadas de los circuitos del programador de la tarjeta (ver diagrama electrónico del sistema).

En seguida, trabajando en la solapa “CODE”, elija la opción “file”, “load and program device” y luego elija el path C:\EDUPIC\archivos16f84. Allí elija el archivo “contador.hex”. El archivo llena los datos en el buffer de WINPIC y al mismo tiempo se transfiere a la memoria FLASH del 16F84 en la tarjeta EDUPIC. Una vez finalizado la transferencia sin errores, el programador anuncia “program finished no errors”. En caso de escribir un mensaje de error, revise las conexiones y la posición de los jumpers.

PASO 6: EJECUTAR EL PROGRAMA EN EL MODULO EDUPIC.

Seleccione la opción “device”=“reset/disconnect ICSP/Go” para ejecutar el programa. Inmediatamente, deberá observar una cuenta binaria en los leds de su tarjeta. También puede oprimir la tecla F9 para hacer que el programa se ejecute. Siguiendo el mismo procedimiento, cargue y ejecute los programas “demof84.hex” (corrimiento de los leds), “microsw.hex” (lectura de los microswitches y activación del relevador con la combinación 1001). Observe que en todos los casos, no es necesario realizar el borrado previo de la memoria FLASH, sino que ésta es borrada automáticamente antes de ser reprogramada.

PASO 7: EJECUCION DEL PROGRAMA EN MODO AUTORUN.

El programa ya cargado en la tarjeta EDU-PIC puede ser ahora ejecutado en modo autorun. Para esto, desconecte el cable serial de la tarjeta y mueva los jumpers S1, S2, S3, hacia la posición “**AUTORUN**”. Dé reset a la tarjeta y el programa se ejecutará ahora en modo autorun. En seguida, el usuario aprenderá a editar, ensamblar, simular y modificar un programa.

PASO 8: CONFIGURAR MPLAB. EDITAR, ENSAMBLAR Y SIMULAR UN PROGRAMA EN LA PC:

MPLAB es un programa que integra numerosas funciones adicionales a las que se cubre en este manual. Desde MPLAB se puede, por ejemplo compilar archivos 16F84 en lenguajes de alto nivel. En el disco CD que acompaña a EDUPIC, se incluye el archivo de instalación del programa PICCLITE, que es un compilador ANSI de C para el 16F84 y que es compatible con MPLAB. Sin embargo, el presente manual, no incluye la puesta en marcha de este compilador dentro de MPLAB. Se deja al usuario el estudio de dicho lenguaje y su utilización dentro del MPLAB. En el subdirectorio “archivos16F84”, hay también un archivo de prueba escrito en lenguaje “C” llamado “led.C”. Por favor acuda al “help” de MPLAB para información completa de todas sus capacidades.

Abra desde el escritorio el archivo ejecutable MPLAB IDE. Para configurarlo, elija la opción “configure”, “select device”. Allí abra la opción “16F84”. Ahora señale la opción “file”, “open” y después elija el path C:\EDUPIC\archivos16f84. Allí elija el archivo “demof84.asm”. El listado del archivo fuente demof84.asm es el siguiente:

```

; File DEMOF84.ASM
; Assembly code for PIC16F84 microcontroller

; Enciende leds del puerto B con un patrón de rotación
; Oscilador Cristal de 4 Mhz. Cada LED enciende durante 0.5 seg.

; Configuración del CPU
; 16F84, Oscilador cristal 4 Mhz,
; watchdog timer off, power-up timer on)
; incluye archivo p16f84.inc

processor 16f84
include <p16f84.inc>
__config _XT_OSC & _WDT_OFF & _PWRTE_ON
J      equ   H'1F'      ; localidad 1F es J
K      equ   H'1E'      ; localidad 1E es K

; Programa

org    0                ; origen de memoria flash =0

movlw  B'00000000'     ; w := 00000000
tris   PORTB           ; puerto B=0 (bits son salidas)
movlw  B'00000001'     ;
movwf  PORTB           ; port B =1
bcf    STATUS,C        ;carry=0
mloop: rlf    PORTB,f   ;rotate left PORT B (incluye carry)
movlw  D'200'         ;modificar este valor a 4 para realizar simulación
movwf  J                ; J := w
jloop: movwf  K          ; K := w
kloop: decfsz K,f       ; K = K-1, skip next if zero
goto   kloop
decfsz J,f             ; J = J-1, skip next if zero
goto   jloop
goto   mloop

end

```

Este es un programa cuya función es encender uno a uno los leds del puerto B con un patrón de rotación. Observe cada instrucción y familiarícese con cada una de las directivas usadas, en particular:

include <p16f84.inc> permite incluir dentro del programa, y como si fuera parte del mismo, al archivo p16f84.inc, el cual ya contiene numerosas directivas con equivalencias de los registros, y los bits usados por el 16f84.

__config _XT_OSC & _WDT_OFF & _PWRTE_ON permite programar el registro de configuración del PIC, en la dirección 2007H. En este caso queda programado con el valor 3FF3H.

J equ H'1F' permite usar la variable J en lugar del valor hexadecimal asociado a su derecha a través de la directiva EQU, para facilidad al programar.

K equ H'1E' permite usar la variable K en lugar del valor hexadecimal asociado a su derecha a través de la directiva EQU, para facilidad al programar.

Una vez con el archivo en la pantalla de su computadora, encuentre la línea en donde se encuentra la instrucción de carga al registro W que se encuentra con comentarios en *letra inclinada*. Con el objeto de escalar la velocidad de simulación, que es muy lenta en comparación al tiempo real, modifique el valor “200” y ponga un “4”. Ahora, elija la opción “project” y después “quickbuild”. Esta opción realiza el ensamblado del programa y produce como salidas, entre otros, los archivos: demof84.lst, y

demof84.hex. El archivo .LST contiene el archivo fuente y los código de máquina. El archivo .HEX, contiene solamente el código de máquina que habrá de almacenarse en el 16F84, en este caso, en la tarjeta EDU-PIC.

FORMATO DEL ARCHIVO .HEX:

El archivo demof84.hex producido en el ejemplo anterior es el siguiente:

```
:100000000306600013086000310860D32309F00FC
:0C0010009E009E0B09289F0B0828052865
:02400E00F33F7E
:00000001FF
```

Es importante, antes de pasar a explicar el formato del archivo, aclarar que la longitud de las instrucciones en el 16F84 es de 14 bits, es decir, que cada localidad de memoria ocupa 2 bytes de almacenamiento en un archivo. Originalmente, el formato .HEX fue diseñado para computadoras con localidades de memoria de solo 8 bits de longitud, de tal forma que el valor de las localidades de memoria en éste archivo se encuentra multiplicado por 2.

El primer byte de cada línea, es un valor que corresponde al número de bytes (en hexadecimal) de información existentes en dicha línea. Por ejemplo, en la primera línea, hay un 10, lo cual significa que habrá 16 bytes de información. En la segunda línea hay un 0C, lo que significa que habrá 12 bytes de información. En la tercera un 02, ó 2 bytes de información en esa línea.

En seguida sigue la dirección de memoria inicial del bloque en donde habrán de almacenarse dichos bytes. En este ejemplo, vemos un 0000 en la primera línea, y un 0010 en la segunda. *Mucha atención aquí:* en la segunda línea, la dirección 0010, ó 16 decimal, corresponde a la dirección número 8 del sistema 16F84 por lo que se explicó en el primer párrafo. En la tercera línea, la dirección 400E, en realidad corresponde a la 2007 en la memoria del PIC 16F84.

El siguiente byte es un 00 en todas las líneas y es un byte reservado en el formato, pero sin información útil.

Los siguientes bytes en cada línea corresponden a la información que habrá de almacenarse en memoria. En la primer línea son 16, en la segunda 12 y en la tercera, solo 2.

Al final de la línea está un byte que es un check sum que sirve para verificación, y corresponde al byte menos significativo de la suma binaria de todos los bytes anteriores en esa línea.

Ahora elija la opción “debugger” y “select tool” y luego “MPLAB SIM”, de esta manera se cargará automáticamente la herramienta de simulación del procesador, con la cual usted podrá observar la ejecución del programa paso a paso. En cada instrucción usted puede revisar el contenido de registros y memoria mediante la opción “View”, la cual le permite ver los registros o localidades de memoria seleccionados.

Dentro de la opción “View”, elija “file registers” para que aparezca una ventana con los registros, e inicie la simulación oprimiendo F6. Avance la simulación oprimiendo repetidamente la tecla F7 y observe el contenido del registro 06, (que es el puerto B) que es en donde están conectados los LEDS. Con la tecla F6, avance paso a paso. El registro pasará del valor hexadecimal 01 al 02, 04, 08, 10, 20,40, 80 que es el patrón de corrimiento de los leds. Observe asimismo el contenido de las localidades 0EH y 0FH que se usan como contadores para los retardos.

ESCALAMIENTO DEL TIEMPO EN EL SIMULADOR CON RESPECTO AL TIEMPO REAL:

La velocidad de simulación es mucho menor a la velocidad del programa corriendo en tiempo real en la tarjeta EDU-PIC. Por esta razón es que en el programa cargado en MPLAB, en la subrutina de retraso debe de modificarse el valor decimal de 200 en el registro W, poniendo un 4 en su lugar. De lo contrario tomaría mucho tiempo hacer la simulación de una secuencia completa de corrimientos. Una vez concluida la simulación, este valor deberá reemplazarse de nuevo por un valor de 200 para cargar el programa en la tarjeta, y obtener un retraso de 250 ms entre cada corrimiento en tiempo real.

Para finalizar este punto, reemplace, como se explicó, el valor ‘4’ por ‘200’ y vuelva a ensamblar el programa mediante los comandos “project” y “quickbuild”. Una vez ensamblado el programa sin errores, es posible transferir el archivo ejecutable a su tarjeta EDUPIC a través del programa WINPIC, según se explicó en los pasos anteriores.

16. Información técnica:

16.1 CARACTERISTICAS GENERALES:

Procesador: microcontrolador PIC16F84, cristal de 4 Mhz, con tecnología CMOS de bajo consumo.

Arquitectura: Harvard, con la memoria de código (14 bits) y de datos (8 bits) separadas. Procesamiento "pipeline".

Tecnología: RISC (reduced instruction set computer), con 35 instrucciones, con 14 bits de longitud de palabra.

Memoria: 1K localidades (14 bits) de FLASH, 68 localidades (8 bits) de RAM, 64 localidades (8 bits) de EEPROM.

Capacidad de lectura /escritura: hasta 10,000 ciclos en la memoria flash y hasta 10,000,000 en la EEPROM.

Puertos digitales: puerto A de 5 bits, puerto B de 8 bits, un total de 13 bits programables como entradas o como salidas.

Resistencias de pull up: disponibles en puerto B, cuando está programado como entradas.

Salidas digitales a LEDs : el puerto B con 8 bits conectados a LEDs.

Capacidad de salidas: cada bit de salida puede tomar ("sink"), ó generar ("source"), hasta 25 miliamperes.

Entradas digitales a microswitches: un total de 4, en el puerto A.

Salida a relevador: relevador modelo RAS-1210, contactos de 127 V @ 1 A. Salida a conector de tornillos.

Header: de 16 contactos, con todas las señales de entrada y salida disponibles, tierra y 5 v.

Conector para teclado: 8 señales incluídas en el header 16x, para teclado matricial de 4 x 4.

Conector para LCD: de 14 contactos, estándar y listo para la conexión de un LCD.

Funciones adicionales: power-on reset, temporizador, watch dog, code protection, sleep (bajo consumo).

Temporizador/contador de eventos: de 8 bits, con preescalador de 8 bits.

Interrupciones: 4 fuentes de interrupción: una externa del pin RBO/INT, una de overflow del temporizador, una más por el cambio en los niveles de las entradas RB4...RB7, y una última por escritura completa en la EEPROM.

Dimensiones: 10 cms x 7.5 cms, tarjeta de fibra de vidrio, thru hole.

Consumo: normal < 2 ma @ 5v. y 4 Mhz., en SLEEP mode solo 15 uA.

Fuente de poder: eliminador de baterías de 300 ma. @ 6 v. ó 7.5 v.

Programador del chip 16F84: integrado en la tarjeta, la programación se realiza desde una computadora PC por puerto serial, empleando el freeware WINPIC.

Software para desarrollo: MPLAB (freeware de Microchip), que incluye editor, ensamblador, simulador y compilador. WINPIC, programador de la memoria flash del 16F84.

16.2. LAY OUT

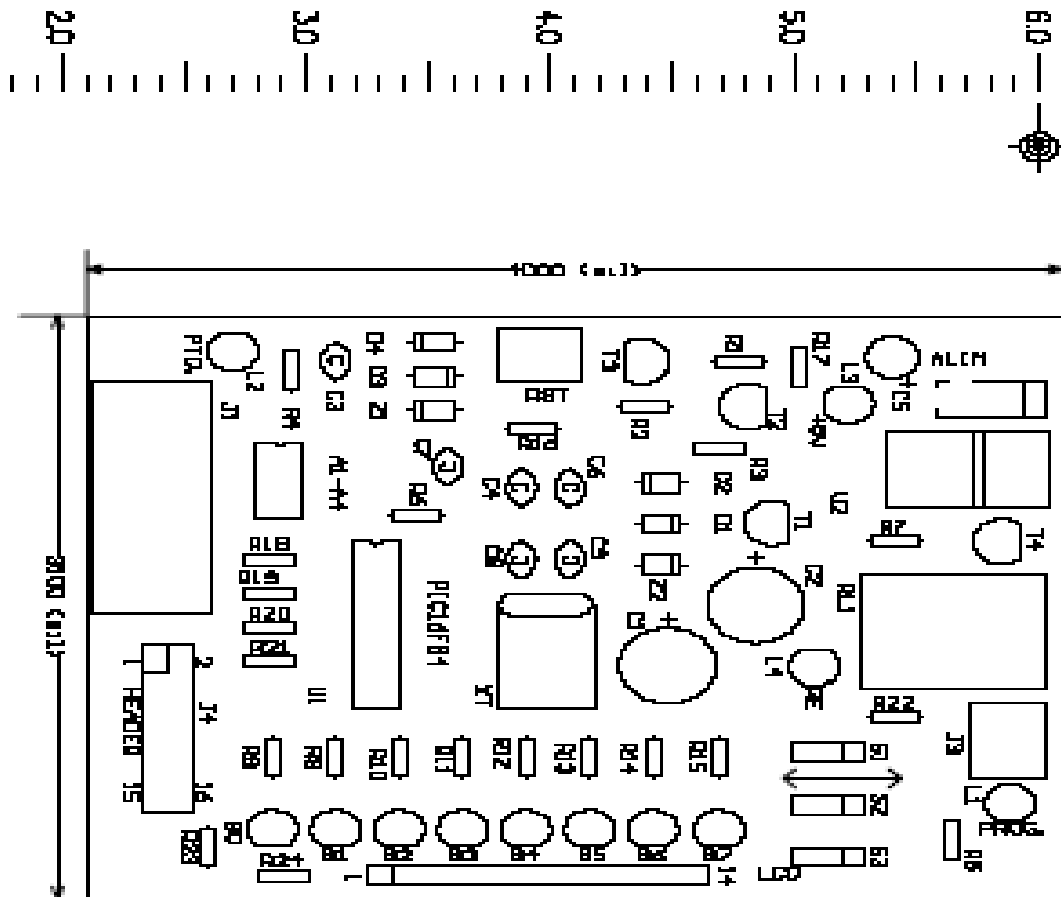
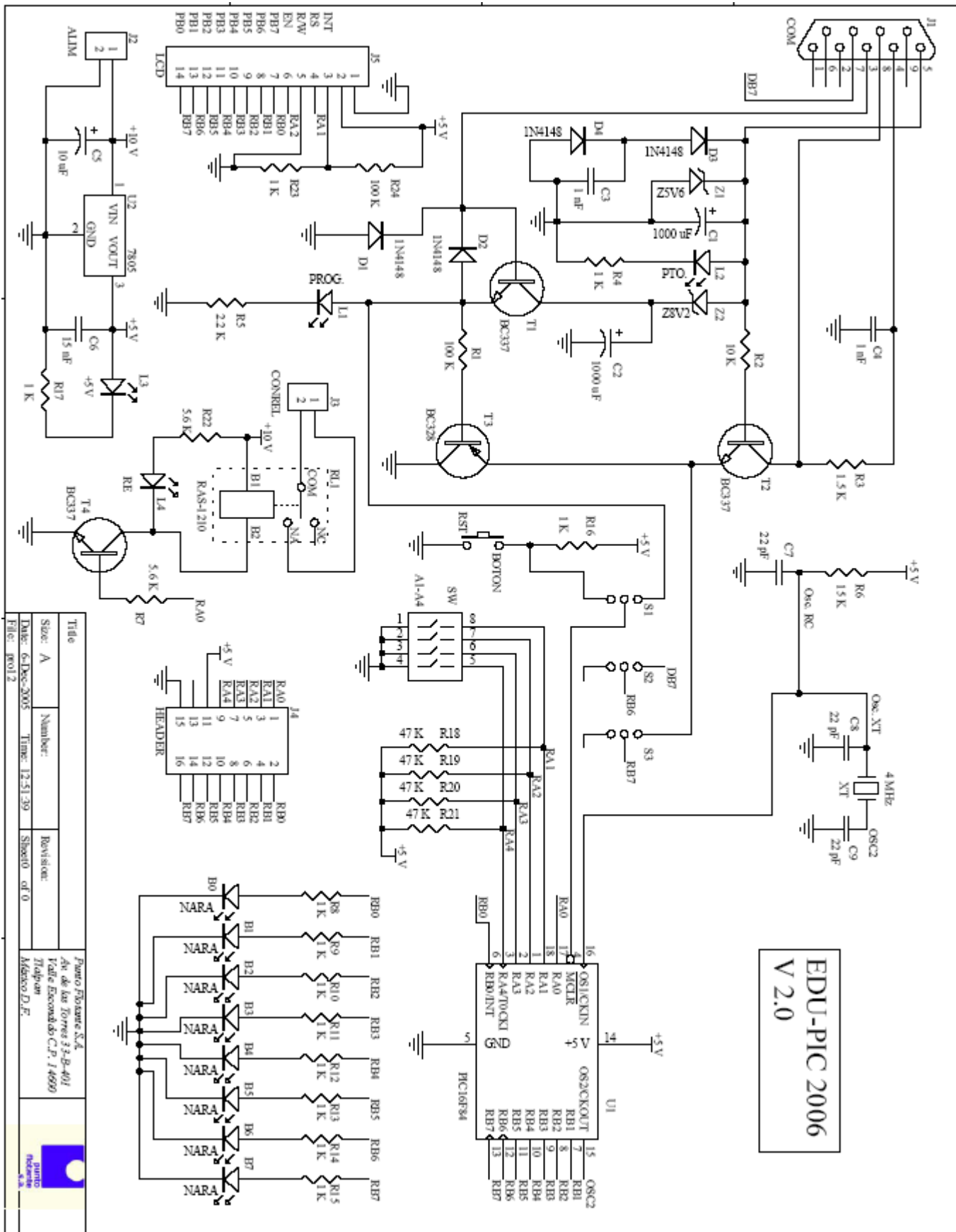


FIGURA 13

16.3 DIAGRAMA ELECTRONICO:



16.4 LISTA DE COMPONENTES:

Referencia	Descripcion	Tipo	Huella	Cant.	P/U	Subtotal
B0,B1,B2,B3,B4,B5,B6,B7	leds anaranjados	NARA	LUZ	8		
C2	Capacitor electrolítico	470 uf	CELEG	1		
C1	Capacitores Elec.	1000 uF	CELEG	1		
C3, C4	Capacitores Cer.	1 nF	CAPI	2		
C5	Capacitor Elec.	10 uF	CELCH	1		
C6	Capacitor Cer.	10 nF	CAPI	1		
C7, C8, C9	Capacitores Cer.	22 pF	CAPI	3		
D1, D2, D3, D4	Diodos	1N4148	DINU	4		
J1	Conector DB9 para impreso	500-020	DB9H	1		
J2	Conector para eliminador	ALIM	COCO	1		
J3	Conector de tornillos 2X	TRT-02	TORA	1		
J4	Conector ángulo 8X (teclad)	HEADER		1		
J5	Conector 14 pines	F-36S	CP14	1		
L1	led amarillo	PROG.	LUZ	1		
L2	led verde	PTO.	LUZ	1		
L3	led rojo	+5V	LUZ	1		
L4	led anaranjado	RE	LUZ	1		
R1, R24	resistencias 1/2 w	100 K	RES	2		
R2	resistencia 1/2 w	10 K	RES	1		
R3	resistencia 1/2 w	1.5 K	RES	1		
R5	resistencia 1/2 w	2.2 K	RES	1		
R6	resistencia 1/2 w	15 K	RES	1		
R7, R22	resistencias 1/2 w	5.6 K	RES	2		
R4, R8 a R17, R23	resistencias 1/2 w	1 K	RES	12		
R18, R19, R20, R21	resistencias 1/2 w	47 K	RES	4		
RL1	relevador	RAS-1210	REL	1		
RST	botón de reset	AU-101	BOT	1		
S1, S2, S3	conector jumper 2 pines	PUENTE	CP3	3		
puentes	tres jumpers	GMJ-2		3		
SW	microswitches 4P	DIP-4P	BAS8	1		
base	para circuito integrado	8 patas		0		
T1, T2, T4	Transistores NPN	BC337	TR92	3		
T3	Transistor PNP	BC327	TR92	1		
U1	Microcontrolador PIC	16F84A-04P	BAS18	1		
base	para circuito integrado	18 patas		1		
U2	Regulador de voltaje + 5 V	7805	REG	1		
XT	Cristal	4 MHz	XTAL	1		
Z1	Diodo zener 5.6 v.	Z5V6	DINU	1		
Z2	Diodo zener 8.2 v.	Z8V2	DINU	1		
tarjeta	circuito impreso	EDU-PIC		1		
fuelle	eliminador de baterias	300 ma.ELI-035		1		
DISCO CD	disco CD			1		
cable	cable serial macho-hembra	DB9 A DB9		1		
empaque	caja de empaque			1		
bolsa antiestática				1		
ensamblado de tarjeta (1 hora)				1		
Módulo LCD 16 x 1	1 renglón			1		
teclado	hexadecimal			1		

16.5 Contenido del disco CD de aplicación:

Los siguientes son los archivos incluidos en este disco:

1. Curso básico de PICS (Documento html), debe de ejecutarse con IE6.
2. Curso básico de PICS (carpeta, contiene archivos de soporte del archivo del punto 1)
3. Programas de prueba 16F84 (carpeta)
4. WINPIC (carpeta, programador para el sistema 16F84)
5. WINZIP (carpeta, programa para descomprimir archivos ZIP).
6. X14 architecture (carpeta, presentación para flash player 6.0)
7. Manual 16F84 (documento PDF)
8. Watch Dog 16F84 (documento PDF)
9. Manual PICCLITE compiler (documento PDF)
- 10.MPLAB Simulator (documento PDF)
- 11.PIC 16F84 EEPROM (documento PDF)
- 12.PIC 16F84 instruction set (documento PDF)
- 13.PIC Microcontrollers (documento PDF)
- 14.PIC Serial programming ICSP (documento PDF)
- 15.PIC Book (documento PDF)
- 16.EDUPIC SPEC 2006 (documento PDF)
- 17.KIT EDUPIC SPEC 2006 (documento PDF)
- 18.EDUPIC Manual 2006 (documento PDF)
- 19.MPLAB IDE 6.60 (archivo ZIP, debe descomprimirse e instalarse)
- 20.PICCLITE Setup (ANSI C Compiler,archivo de instalación ejecutable)
- 21.AdBdRdr60 (archivo ajecutable, de instalación del Adobe Reader)
- 22.Pk Reader (archivo ejecutable, de instalación)

Apéndice 1: Proyectos

Conexión a teclado matricial , LCD y Real Time Clock

Conexión a un teclado matricial:

Si se requiere conectar un teclado a su sistema, por favor haga uso de la siguiente información. El ejemplo que se ilustra, es para un teclado matricial de 16 teclas de la marca GREYHILL, modelo 86JB2-203. **No existe un estándar** para los conectores de los teclados, de tal forma que cada modelo exige un hardware de conexión y una subrutina de manejo distintos.

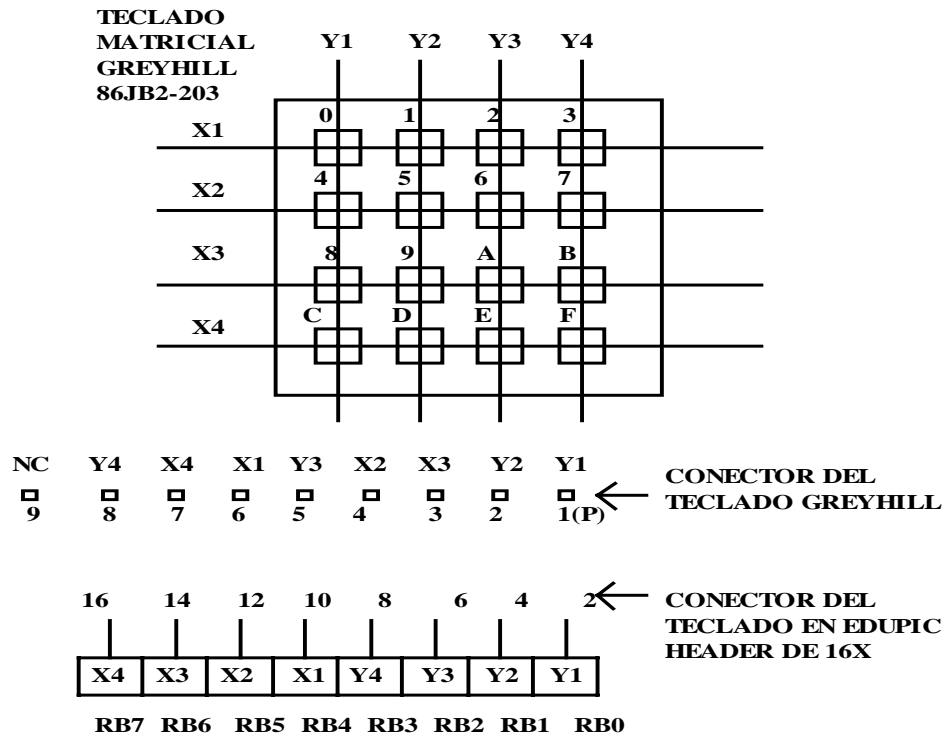


FIGURA 15 : DIAGRAMA DEL TECLADO Y SU INTERFAZ CON EDUPIC

El teclado viene organizado con 4 renglones (X1...X4) y 4 columnas (Y1...Y4), de tal forma de que cada tecla queda identificada por la intersección de un renglón y una columna. En la figura se muestran los diagramas del teclado, así como de la asignación de señales en el puerto B, en el conector de EDUPIC y en el conector del teclado.

La técnica de programación para detectar qué botón fue oprimido, es escribir en los bits del puerto B en forma secuencial un “CERO” en las columnas Y1, Y2, Y3, Y4, y leer cada vez los renglones X1, X2, X3, X4. Cuando una tecla es oprimida, la lectura en alguno de los renglones será “CERO” y el código de 8 bits X-Y hallado se convierte en el código ASCII de la tecla oprimida mediante una tabla. La tabla de abajo muestra esta relación:

<u>Código en B</u>	<u>Hexa</u>	<u>Tecla</u>	<u>Código en B</u>	<u>Hexa</u>	<u>Tecla</u>
B7.....B0			B7.....B0		
11101110	EE	“0”	10111110	BE	“8”
11101101	ED	“1”	10111101	BD	“9”
11101011	EB	“2”	10111011	BB	“A”
1110 0111	E7	“3”	10110111	B7	“B”
11011110	DE	“4”	01111110	7E	“C”
11011101	DD	“5”	01111101	7D	“D”
11011011	DB	“6”	01111011	7B	“E”
11010111	D7	“7”	01110111	77	“F”

PROGRAMAS DE PRUEBA PARA EL TECLADO: Mediante la tabla de arriba, el programa que controla al teclado pasa de la parte de detección a otra que identifica el código ASCII de la tecla oprimida. Por favor abra los archivos que se encuentran en el disco de aplicación, en el subdirectorio “archivos16F84”. El nombre de estos archivos es: “teclado1”, “teclado2”, “teclado3” y “teclado4”. Lea en cada archivo la función que desarrolla. La secuencia del 1 al 4 está hecha para facilitar la comprensión del control del teclado. “teclado4” funciona con el LCD conectado a su puerto LCD en el cual muestra la tecla oprimida correspondiente.

CONEXIÓN DEL TECLADO A EDUPIC: usando los diagramas de las figuras 1, 13 y 15, se conectan las señales de los renglones y columnas del teclado a los pines pares: 2, 4, 6, 8, 10, 12, 14, 16 en el HEADER 16X del sistema EDUPIC. Recuerde que este diagrama está hecho específicamente para el teclado del modelo mostrado, de tal forma que si se tiene un teclado distinto, entonces el diagrama de conexiones cambiará. Tome nota de la forma en que están numerados los pines del HEADER, ya que siguen una convención distinta a la de los circuitos integrados, estando los pines pares en una fila y los nones en la otra.

Conexión a un LCD:

A diferencia del teclado, los fabricantes del display de cristal líquido (“Liquid Cristal Display”) LCD, han estandarizado sus señales en un conector de 14 pines, así como sus comandos de control para el manejo del mismo.

El LCD es actualmente el circuito más barato y confiable para mostrar datos en un proceso de monitoreo y control. Su interfaz con los controladores se realiza a través de un conector de 14 pines, cuya configuración es respetada por la mayoría de los fabricantes.

En el LCD se pueden mostrar datos como la hora y la fecha, así como valores de variables tales como nivel, presión, gasto, temperatura, etc. El LCD puede también emplearse para programar parámetros internos del sistema, de acuerdo a su aplicación o para mostrar al usuario las opciones del sistema mientras lo opera.

En la interfaz de 14 pines, 8 son señales de datos. Estos datos se manejan en códigos ASCII y se escriben en la memoria del LCD en forma secuencial. A través de estas mismas señales pueden escribirse también comandos. En las siguientes tablas se ilustran, en la primera, la distribución de señales de la interfaz y en la segunda algunos de los comandos más comúnmente empleados.

El módulo LCD lleva integrado a sus circuitos una memoria ROM conocida como “generador de caracteres” que habrá de generar los patrones de puntos de los caracteres en la pantalla.

También tiene una RAM interna que almacena los caracteres en forma secuencial y los exhibe en el módulo LCD.

Todas las señales de datos y control llevan niveles TTL (0 a 5 volts), con excepción de la señal de control de intensidad, en el cual hay que aplicar tierra para la mayor intensidad y 5 volts para la menor. En EDUPIC, este voltaje ya viene preajustado con un divisor de resistencias formado por R26 y R25, de 1K y 100K para dar la intensidad adecuada.

El diagrama general de conexiones del display se muestra en la figura 9. En su aplicación más simple, se desea solo escritura al LCD y el pin 5 se conecta permanentemente a tierra. La forma de escribir datos en el display es la siguiente :

- se mandan comandos de inicialización al display, con la señal RS=0 (en estado bajo). Los comandos típicos son los 4 mostrados en la tabla. Para mandar el comando, se escribe su código en los 8 bits DB0 a DB7 y se le da un pulso BAJO a la señal EN. Esta señal debe estar NORMALMENTE ALTA y se pueden manejar pulsos de 50 ms. Es importante esperar a que los datos estén estables antes de aplicar el pulso.

PIN	FUNCION	PIN	FUNCION
1	Tierra	8	DB1
2	5 volts.	9	DB2
3 INT	Control de Intensidad	10	DB3
4 RS	0=comando 1=datos	11	DB4
5 R/W	0=escribir en LCD 1=leer	12	DB5
6 EN	Enable modo pulso	13	DB6
7	DB0	14	DB7

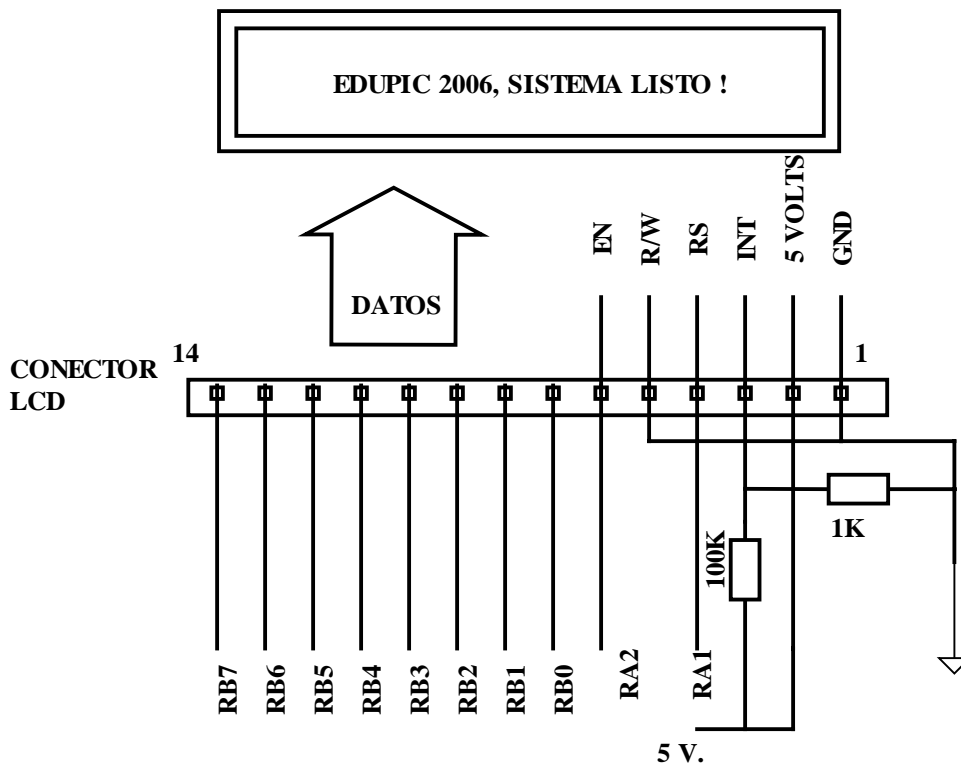


FIGURA 16 : CONEXIÓN DEL MODULO LCD

COMMAND	FUNCION	HEXA
CLEAR	LIMPIA DISPLAY	01H
HOME	POSICIONA CURSOR AL INICIO DEL LCD	03H
CURSOR	MUESTRA CURSOR CON PARPADEO	0FH
8 BITS	SELECCIONA INTERFAZ DE 8 BITS PARA TRANSFERENCIAS DE DATOS	38H
RENGLON1	SELECCIONA ESCRITURA EN EL PRIMER RENGLON DEL LCD	80H
RENGLON2	SELECCIONA ESCRITURA EN EL SEGUNDO RENGLON DEL LCD	C0H

- con RS=1, se escribe los datos ASCII en las 8 líneas DB0 a DB7, con el mismo procedimiento señalado arriba. Los caracteres van apareciendo en el mismo orden en el que se mandan. Para un LCD de dos renglones, cada renglón debe de manejarse con un comando independiente. Para escribir en el primer renglón, debe previamente de enviarse el comando 80H y para el segundo renglón, un C0H.
- Las señales de control RS y EN, se manejan por medio de los pines RA1 y RA2 del 16F84.

PROGRAMAS DE PRUEBA PARA EL LCD: existen dos tipos de displays LCD:

LCD 16 X 2 con dos renglones de 16 caracteres cada uno. Se proporcionan los programas de prueba “LCD1” ,“LCD2”, “LCD MENSAJE” y “LCD 16 X 2” (escribe un mensaje y hace scrolling hacia la izquierda), para la prueba del dispositivo. Dichos programas se encuentran en el subdirectorio “archivos16F84”.

LCD 16 X 1 es un dispositivo LCD con un renglón de 16 caracteres, pero electrónicamente está dividido en dos secciones de 8 caracteres. Cada una de estas secciones se maneja como si fuese un renglón de 8 caracteres. Para la prueba de estos LCD, se tienen los archivos “LCD 16 X 1”, “LCD16X1 V2”, “LCD16X1 V3” (mensaje de 16 caracteres con scrolling) y “LCD16X1 V4” (mensaje de 256 caracteres con scrolling hacia la izquierda), también en el subdirectorio “archivos16F84”.

CONEXIÓN DEL LCD AL SISTEMA EDUPIC: para conectar un LCD a su sistema, por favor conecte cada pin en EDUPIC con su correspondiente en el LCD, del 1 al 14. Tome como auxiliares a los diagramas de las figuras 13 y 16.

NOTA IMPORTANTE: Las señales de control correspondientes a RA1 y RA2 están compartidas por el puerto del LCD y los microswitches A1 y A2. Entonces es muy importante que, al operar el LCD, los **SWITCHES A1 y A2 estén en OFF, ABIERTOS**, es decir, en su posición más cercana al puerto serial. De lo contrario, el LCD no funcionará, debido a que sus señales de control estarán bloqueadas por los microswitches.

Real Time Clock

El reloj de tiempo real es una herramienta sumamente útil en los sistemas de microcontrol, por ejemplo en la implementación de aplicaciones como temporizadores industriales, en los cuales se deben activar o desactivar ciertos dispositivos en ciertas horas, o en los sistemas de control de acceso en los cuales se deben detectar eventos y conocer la hora en la que ocurrieron.

Se emplea el temporizador TMR0 y el preescalador para generar ciclos de interrupción cada 65.536 milisegundos. Contadores adicionales generados por software, realizan la cuenta de segundos, decenas de segundos, minutos, decenas de minutos, horas y decenas de horas. Una vez que la cuenta alcanza 23:59:59 y se genera la cuenta de un segundo adicional, entonces la hora pasa automáticamente a 00:00:00. Dado que el sistema descrito funciona por interrupciones, es posible añadir al programa principal propuesto, rutinas de control adicionales para resolver un problema específico sin afectar el funcionamiento de la base de tiempo

Si el usuario dispone de un display LCD de 16 x 1, entonces podrá hacer uso del programa desarrollado para hacer que en éste se muestre la hora, minutos y segundos del día. Por favor cargue desde WINPIC (refiérase al capítulo de “Puesta en Marcha”), el archivo RTC V3.HEX hacia su sistema EDU-PIC. El archivo original comienza con la hora 00:00:00.

Si desea poner la hora actual, modifique el archivo RTC V.3.ASM, usando MPLAB, localizando la subrutina “seto” y escriba allí los contenidos deseados de: segundos, decenas de segundos, minutos, decenas de minutos, horas y decenas de horas. En seguida aplique el comando “quickbuild” y después desde WINPIC vuelva a cargar el programa en EDU-PIC.

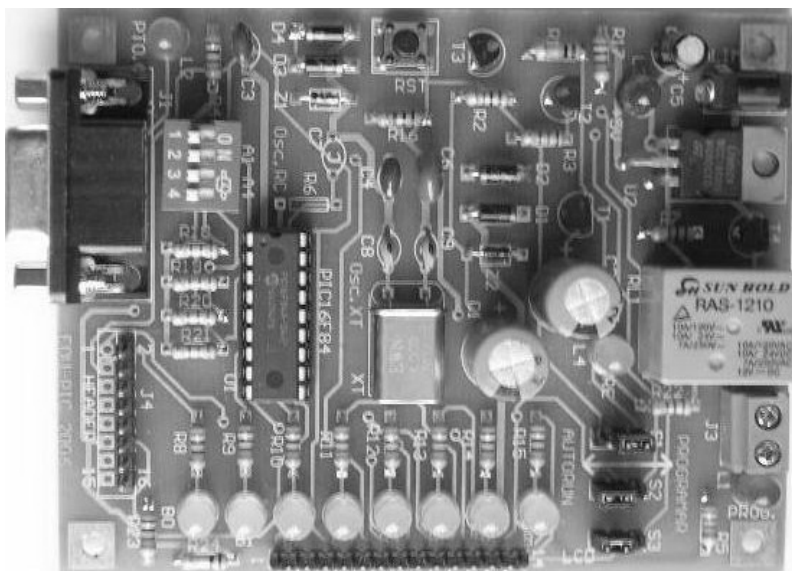


FIGURA 17