INSTITUTO TECNOLÓGICO SUPERIOR AERONÁUTICO

ESCUELA DE TELEMÁTICA

OPTIMIZACION DEL LABORATORIO DE MICROCOMPUTADORES ITSA MEDIANTE LA CONSTRUCCIÓN MÓDULOS ENTRENADORES PARA PRÁCTICAS EN MICROCONTROLADORES PIC, QUEMADOR - PROGRAMADOR DE MEMORIAS Y ELABORACIÓN DE GUÍAS DE LABORATORIO.

POR:

CBOS: BENAVIDES RAMOS JORGE ANDRES

CBOS: BAUTISTA MACHUCA EDWIN RAMIRO

Tesis presentada como requisito parcial para la obtención del titulo de:

TECNÓLOGO EN TELEMÁTICA

2002

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por los Srs. CBOS. Edwin Ramiro Bautista Machuca y CBOS. Jorge Andrés Benavides Ramos, como requerimiento parcial a la obtención del titulo de TECNÓLOGOS EN TELEMÁTICA.

Latacunga, Octubre del 2002.

_____

Ing. Wilson Vinueza

DEDICATORIAS

El presente trabajo esta dedicado en primer lugar a Dios, a mi Madre, a mi Padre y a toda mi familia ya que ellos me han apoyado en mis alegrías, y desengaños, logrando hacer realidad mis mejores sueños

EDWIN BAUTISTA

Este esfuerzo va dedicado a mis padres que me dieron todo el apoyo posible para poder alcanzar este logro , a Dios por brindarme la salud y vida que poseo para poder seguir adelante con las metas, sobrepasando barreras, obstáculos y en el largo camino de la vida y así ascender un peldaño mas en la vida.

BENAVIDES JORGE

AGRADECIMIENTO

Nuestro sentimiento de agradecimiento esta dirigido a todas aquellas personas que queremos y respetamos, porque aquellas personas se han sacrificado día a día demostrándonos, que aun cuando la vida no vale nada, que aun cuando todo se ha perdido te queda el futuro

Tanto de manera especial a nuestro inmortal Instituto Tecnológico Superior Aeronáutico a todos sus profesores así como a nuestro asesor, Ing. Wilson Vinueza, por el esfuerzo prestado para sacar adelante este proyecto .

INDICE

CAPITULO V: CONCLUSIONES Y RECOMENDACIONES

INDICE DE TABLAS.

INDICE DE FIGURAS.

INTRODUCCIÓN

Para mejoras del ITSA más que todo de los laboratorios de microcomputadores y conjuntamente con ellos el aprendizaje de los microcontroladores que quizá es uno de los componentes mas versátiles que existen , ya que su aplicaciones están limitadas tan solo por la imaginación. Cada elemento hoy en día se puede encontrar hasta en el hogar así por ejemplo los microcontroladores están presentes en cafeteras eléctricas, televisores, hornos de micro ondas, etc.

Aunque todavía existen muchas ramas a las que esta innovación tecnológica no ha sido aplicada, de ahí la necesidad de construir el Entrenador K-148 como un medio de aprendizaje y conocimiento para la aplicación de los microcontroladores que como se mencionó es una herramienta de vital importancia para enfrentar el reto que impone la tecnología actual y para satisfacer un mercado que requiere un personal idóneo en el área del diseño y desarrollo. Pero la mejor y quizá la única manera de aprender es realizando prácticas y aplicaciones reales con las cuales el alumno desarrollará la habilidad necesaria.

CAPITULO I
EL PROBLEMA.

1.1.-Planteamiento del Problema.

Conociendo que en el laboratorio de microcomputadores del ITSA necesitan realizar prácticas con las diferentes familias de microcontroladores existentes en el mercado y en el ámbito tecnológico, se ha optado por optimizar dicho laboratorio implementando Módulos Entrenadores de la familia PIC y además se podrá realizar el grabado y borrado de memorias mediante el Quemador – Programador de Memorias con sus respectivas guías de laboratorio.

1.2.- Objetivos

1.2.1.- Objetivo General.

Optimizar el Laboratorio de Microcomputadores ITSA mediante la construcción de Módulos Entrenadores para prácticas en Microcontroladores PIC, Quemador - Programador de Memorias y Elaboración de Guías de Laboratorio.

1.2.2.- Objetivos Específicos.

- Ayudar en el avance Tecnológico del Laboratorio de Microcomputadores del ITSA para mejorar la calidad de enseñanza - aprendizaje del alumnado.
- Construir un Entrenador PIC Avanzado para prácticas en el Laboratorio de Microcomputadores.
- Realizar prácticas en circuitos con un modelo de la Familia MICROCHIP y elaboración de guías de laboratorio.

1.3.- Justificación.

Conociendo que en el laboratorio de microcomputadores del ITSA necesita realizar prácticas con las diferentes familias de microcontroladores existentes en el mercado y en el

ámbito tecnológico, se ha optado por optimizar dicho laboratorio implementando Módulos Entrenadores de la familia PIC y además en el mencionado laboratorio se podrá realizar el grabado y borrado de memorias mediante el Quemador – Programador de Memorias.

Consientes de la necesidad que se puede suplir con los microcontroladores, con la convicción de que la mejor metodología es"APRENDER HACIENDO", esperamos como resultado de este Proyecto que los siguientes estudiantes se introduzcan fácil y rápidamente en el maravilloso mundo de los microcontroladores y se preparen para asumir el reto tecnológico que se impone cada día los nuevos avances de la ciencia.

CAPITULO II

MARCO TEORICO

Introducción

2.1.- Generalidades.

El desarrollo de cada nuevo dispositivo electrónico trae consigo técnicas de diseño diferentes, por lo general, más simples. En los años sesenta, para construir un reloj digital se necesitaba acoplar un buen número de circuitos lógicos como contadores, divisores, decodificadores y redes combinatorias. Al mismo tiempo, el diseñador debía poseer muy buenos conocimientos sobre cada uno de los elementos.

A partir de 1970, el panorama de la electrónica cambió radicalmente con la aparición del microprocesador. Vendría la época de oro del Z-80, el 8085, el 6800 y otros microprocesadores utilizados como elementos centrales en aparatos de control, y se consolidarían las técnicas de integración, el estudio de las memorias, la programación en lenguaje de máquina y la adaptación de periféricos de todo tipo.

En 1980, aproximadamente, los fabricantes de circuitos integrados dieron a conocer un nuevo chip llamado microcontrolador, el cual contenía toda la estructura de un microcomputador, es decir, unidad de proceso (CPU), memoria RAM, memoria ROM y circuitos de entrada/salida. Este se concibió como un dispositivo programable que puede ejecutar un sinnúmero de tareas y procesos.

Desde este momento, el diseño de productos electrónicos cambió radicalmente, circuitos lógicos, manejo de periféricos, temporizadores y estructura de computadores, todo programable y alojado en un solo integrado; es decir, un pequeño computador para todas las aplicaciones.

Los dispositivos PIC son microcontroladores RISC baratos y capaces para ejecutar hasta 5Mips. Tan solo dispone de 32 comandos de programación, y en los modelos mas básicos, de 14 líneas de Entradas y/o Salidas, totalmente programable. Uno de los más utilizados es el PIC16F84, que es un dispositivo borrable eléctricamente, ideal para los experimentadores, se programa fácilmente con un protocolo que utiliza solo tres líneas de control, aunque no es el más barato.

2.2.- Los Microcontroladores.

Los microcontroladores son circuitos integrados programables, capaces de ejecutar las órdenes o secuencias que están grabadas en su memoria. Cada uno de estos está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica dentro del ordenamiento del mismo y a su vez permiten obtener configuraciones diferentes. Esto se pueden diferenciar según el tamaño y cantidad de sus elementos básicos y características especiales. En esta sección nombramos algunas de las partes básicas a manera de información para establecer comparaciones más adelante.

Memoria ROM (Memoria de sólo lectura)

Memoria RAM (Memoria de acceso aleatorio)

Líneas de entrada/salida (I/O). También llamados puertos, se utilizan para conectar los elementos externos al microcontrolador

Lógica de control. Coordina la interacción entre los demás bloques

Componentes especiales en un microcontrolador.

En muchas ocasiones, se requiere algo más que los pines de entrada y salida para controlar algún proceso, como se muestra en la figura 2.1 Pensando en ello los fabricantes de microcontroladores han adicionado algunos componentes especiales en algunos de sus modelos. Las posibilidades son amplísimas y el usuario puede escoger la que más le convenga.

- Algunos microcontroladores tienen conversores análogo a digital (A/D), en caso de que se requiera medir señales no digitales, como por ejemplo temperatura, voltaje, luminosidad, etc.

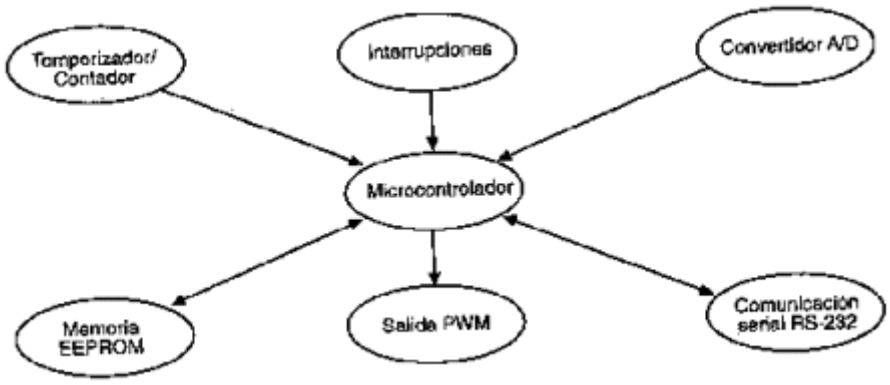


Figura 2.1 Funciones adicionales en los microcontroladores

- Si se requiere medir períodos de tiempo entre eventos, generar temporizaciones o salidas con frecuencia específica, algo muy común en electrónica, se puede disponer de uno o varios temporizadores programables (timers).
- Cuando se necesita establecer comunicación con otro microcontrolador o con un computador se puede disponer de una interfase serial RS-232.
- Para desarrollar una aplicación donde los datos no se alteren a pesar de que se retire la alimentación, se puede usar un microcontrolador con memoria EEPROM, que es un tipo de memoria ROM que se puede programar o borrar eléctricamente sin necesidad de circuitos especiales.
- Para quienes utilizan salidas PWM (modulación por ancho de pulso) en el control de motores DC o cargas resistivas, existen microcontroladores que pueden ofrecer varias de ellas.

- Cuando se requiere atender eventos en tiempo real o se tienen procesos que no dan espera, se debe utilizar la técnica llamada de «Interrupciones». Cuando una señal externa activa una línea de interrupción, el microcontrolador deja de lado la tarea que se encuentra ejecutando para atender una situación especial y luego puede regresar a continuar con la labor que estaba desempeñando.

2.2.1.- Tipos o Familias.

Existen en el mercado varias marcas reconocidas como las principales, dadas sus características, difusión y usos en productos de consumo masivo. Entre ellas están Motorola, Intel, Philips, National y Microchip.

Familia Intel 8051.

Esta familia de microcontroladores de 8 bits contiene varias referencias, cada una de ellas acondicionada para aplicaciones específicas. Todas las versiones existentes tienen la misma CPU, memoria RAM, temporizadores, puertos paralelos y entradas/salidas de tipo serial.

El 8051 tiene 4 Kbytes de memoria ROM que debe programarse durante el proceso de fabricación del circuito integrado. En el 8751, la memoria ROM se ha reemplazado por una memoria EPROM que el usuario puede programar y borrar con luz ultravioleta.

Tabla 2.1 Microcontroladores de la familia Intel 8051

Referencia	Memoria ROM	Memoria RAM	Timers	
8051	4K		128	2
8052	8K	256		3
8031			Externa	
128	2			

8032			Externa	
256	3			
8751			4K (Eprom)	128
	2			

El 8031 es un caso especial; no tiene memoria ROM interna y por lo tanto, la memoria de programa se debe colocar externamente. Para comunicarse con la memoria externa, el micro debe usar tres de los cuatro puertos paralelos de entrada/salida.

Familia Motorola 6805.

Es una de las más utilizadas en el mundo. Ha sido optimizada para aplicaciones de control especializado, en lugar de procesamiento de datos, y forma parte de dispositivos de producción masiva como juguetes, equipos de video, impresoras, modems, electrónica automotriz y electrodomésticos. Cada año aparecen nuevos modelos que reemplazan a los anteriores. Todos los dispositivos básicos están construidos a partir de la misma CPU de 8 bits y tiene RAM, ROM, puertos de entrada/salida y temporizadores; algunos tienen, además, puertos seriales, convertidores análogo a digital y memorias EEPROM o EPROM.

Tabla 2.2. Microcontroladores de la familia Motorola 6805

Referencia	Memoria ROM	Memoria RAM	Timers	Otros
68HC05C4		4K		
176	1			
68HC0508		8K		
176	1			
68HC05C2		2K		
176	1			
68HC705C4		4K (Eprom)	176	
	1			

68HC705K1 504

32 Y 64 1

68HC05BM 2K

176 1 mejoradoconverA/D

Familia Microchip 16CXX.

Está formada por una amplia variedad de componentes con diferentes tamaños de memoria, diferentes velocidades, diferentes tipos de encapsulado y diferente número de pines de entrada/salida. El conjunto de instrucciones es de sólo 35, por eso se dice que es un microcontrolador de tipo RISC (Reduced Instruction Set Computer - Computador con Set de Instrucciones Reducido) a diferencia de las anteriores familias que utilizan la tecnología CISC (Computador con Set de Instrucciones Complejo).

Tabla 2.3. Microcontroladores de la familia PIC16CXX de Microchip

Referencia	Memoria ROM	Memoria RAM	Pines I/O	Otros
16C54		512		32
16C57	12	2K		80
16C61	20	1K		48
	12			Interrupción

16C71 1K 48

12 Conver.A/D

16C84 1K(EEPROM) 48

12

12C509 1K 48

6 Encaps 8pines

Esta familia sobresale por estar muy difundida actualmente a nivel mundial; la mayoría de revistas de electrónica en el mundo la usan para sus proyectos y le dedican artículos periódicamente. Su flexibilidad, configuraciones para todas las necesidades y bajo costo, la hacen muy atractiva para los consumidores a gran escala y para los estudiantes o diseñadores independientes.

Familia Microchip 16F84.

El PIC16F84 es un microcontrolador con la memoria de programa de tipo FLASH, lo que representa gran facilidad en el desarrollo de prototipos y en su aprendizaje ya que no se requiere borrarlo con la luz ultravioleta como las versiones EPROM sino, permite reprogramarlo nuevamente sin ser borrado con anterioridad.

2.2.2.- Ventajas y Desventajas.

Ventajas

- Familia Intel 8051 la programación en este microcontrolador viene dada desde la fabricación.
- La Familia Motorola 6805 se utiliza en la mayor parte de producción masiva, como son equipos de video, impresoras, electrodomésticos, etc.
- La Familia Microchip 16CXX tiene un conjunto de 35 instrucciones y es un microcontrolador de tipo RISC (Reduced Instruction Set Computer - Computador con Set de Instrucciones Reducido)
- La Familia Microchip 16F84 es un microcontrolador con memoria de programa tipo flash (no se requiere borrarlo con anterioridad para su reprogramación), bajo consumo de potencia, además es completamente estático, esto quiere decir que el reloj puede detenerse y los datos de la memoria no se pierden.

Desventajas

- La Familia Intel 8051 su memoria EPROM se programa con luz ultravioleta. Dentro de este existe el 8031donde su memoria ROM no es interna por lo que se debe colocar externamente.+
- La Familia Motorola 6805 no se utiliza para procesamiento de datos, constantemente aparecen nuevas versiones nuevos modelos por lo que no se puede estandarizar de esta manera quedando obsoletos los anteriores.
- La Familia Microchip 16CXX tiene una memoria EPROM que utiliza luz ultravioleta para su programación.
- La Familia Microchip 16F84 fabricado en tecnología CMOS por que es muy costoso.

2.2.3.- Arquitectura Interna.

Arquitectura interna del PIC16C84.

Este término se refiere a los bloques funcionales internos que conforman el microcontrolador y la forma en que están conectados, por ejemplo la memoria EE-PROM (de programa), la memoria RAM (de datos), los puertos, la lógica de control que permite que todo el conjunto funcione, etc.

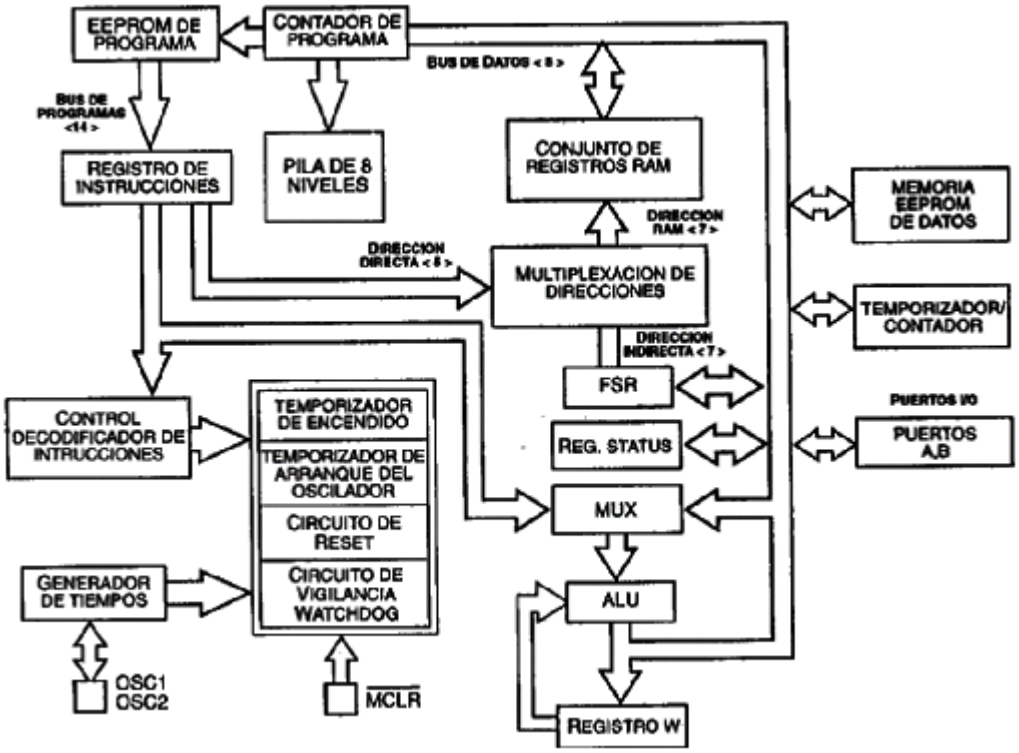


Figura 2.2. Arquitectura interna del PIC16C84

La figura 2.2 muestra la arquitectura general del PIC16C84; en ella se pueden apreciar los diferentes bloques que lo componen y la forma en que se conectan. Se muestra la conexión de los puertos, las memorias de datos y de programa, los bloques especiales como el watchdog, los temporizadores de arranque, el oscilador, etc.

Todos los elementos se conectan entre sí por medio de buses. Un bus es un conjunto de líneas que transportan información entre dos o más módulos. Vale la pena destacar que el PIC 16C84 tiene un bloque especial de memoria de datos de 64 bytes del tipo EEPROM.

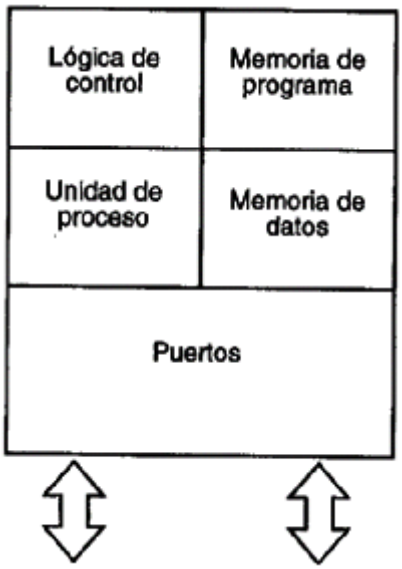


Figura 2.3. Arquitectura simplificada del PIC16C84

El PIC16F84 se basa en la arquitectura Harvard, en la cual el programa y los datos se pueden trabajar desde memorias separadas, lo que posibilita que las instrucciones y los datos posean longitudes diferentes. Esta misma estructura es la que permite la superposición de los ciclos de búsqueda y ejecución de las instrucciones, lo cual se ve reflejado en una mayor velocidad del microcontrolador

2.2.4.- Circuitos de experimentación.

Circuito de experimentación con el PIC16C84.

El entrenador K-061 de CEKIT es un circuito muy versátil que contiene varios circuitos útiles para empezar a experimentar con el PIC, en la figura 2.4. se muestra su diagrama en bloques.

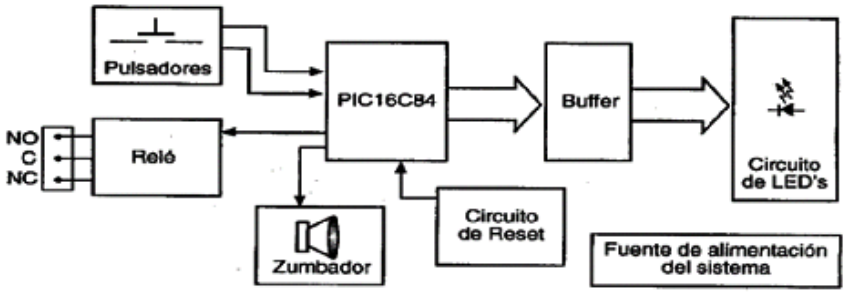


Figura 2.4. Diagrama en bloques del entrenador K-061 de CEKIT

El entrenador incluye un circuito de reset, la fuente de alimentación, simulación de entradas con pulsadores, un relé para manejo de cargas de potencia, una salida con zumbador y salidas con LED's. Estas últimas pasan a través de un buffer con el integrado ULN2803 para mejorar la capacidad de corriente y aislar e independizar los circuitos cuando se desea extender el puerto B del microcontrolador hacia otra tarjeta o un protoboard.

Descripción.

El módulo es completamente autónomo, solo necesita conectarse a la línea de alimentación de 120/220 VAC y disponer de un microcontrolador PIC16C84 programado previamente. La fuente de alimentación está incluida en el circuito; además se dispone de un interruptor de encendido y apagado.

El entrenador ha sido construido de la forma más sencilla posible. Se dispone en el de 8 LED's, un zumbador, dos teclas y un relé, con lo cual se ocupa los 12 pines de entrada/salida del microcontrolador. Los LED's se manejan a través de un integrado ULN2803; este es un buffer que permite obtener una buena iluminación. Para encenderlos, el microcontrolador debe dar una señal positiva en el pin correspondiente, ya que el buffer tiene internamente un transistor NPN que invierte el pulso.

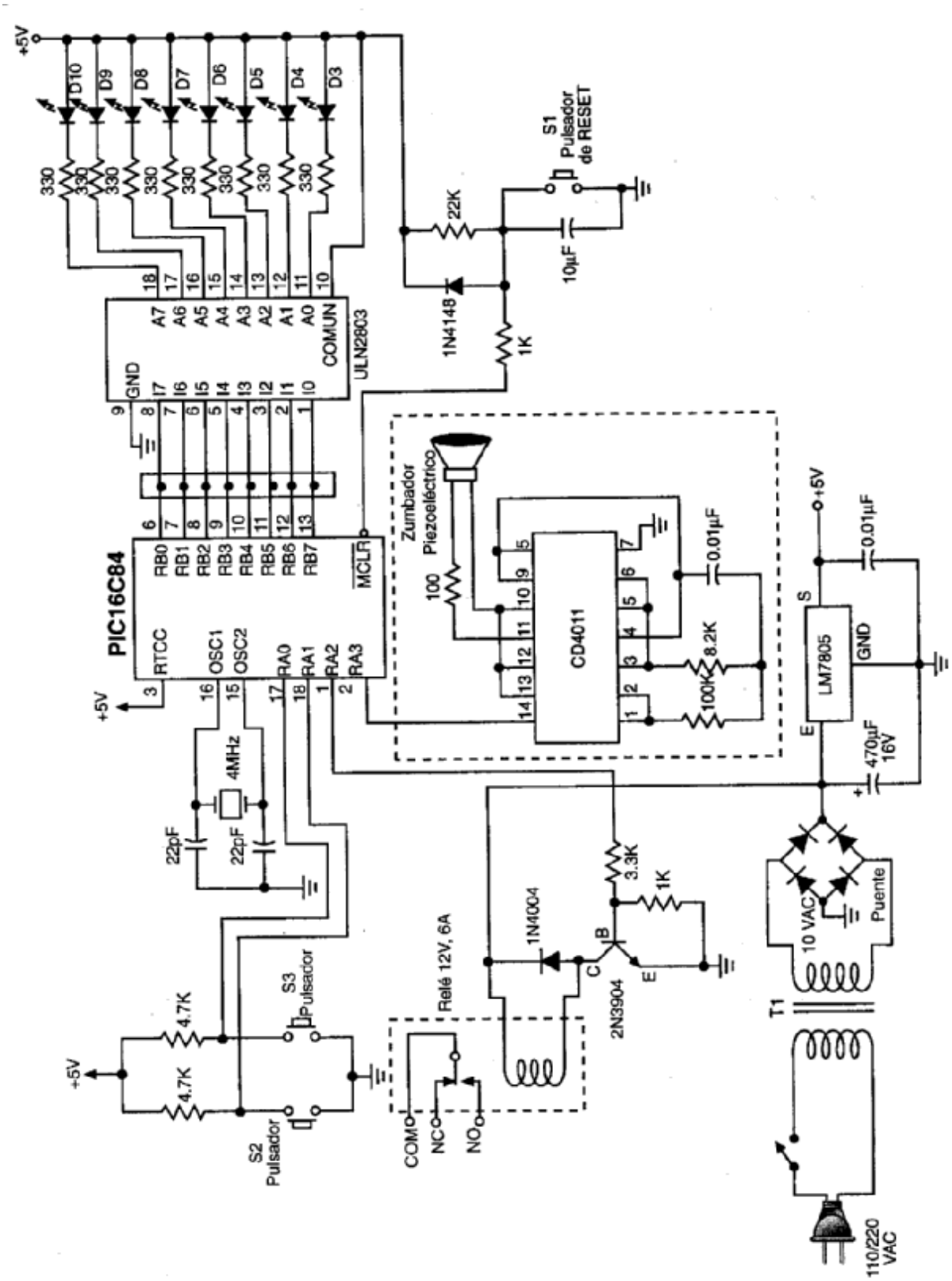


Figura 2.5. Circuito del entrenador K-061

- El zumbador se encuentra listo con su oscilador y todos los componentes necesarios; el microcontrolador sólo debe entregar una señal positiva para encenderlo y éste debe emitir su sonido.
- Las teclas o pulsadores que se pueden leer tienen una resistencia conectada al positivo de la fuente, normalmente se lee una señal lógica positiva y al oprimir la tecla se lee una señal lógica negativa.
- El relé se maneja usando un transistor NPN. Entonces para activarlo, el microcontrolador debe entregar una señal positiva.

Existe en el circuito impreso del entrenador un conector de 9 pines donde se tiene disponible todo el Puerto B del microcontrolador y también la tierra del circuito. Este sirve para que el usuario del sistema pueda trabajar con esas señales en otra tarjeta de aplicación.

Utilizando este entrenador, se puede realizar una gran cantidad de experimentos y ejercicios sin necesidad de gastar tiempo en el ensamble de los circuitos en protoboard o en un circuito impreso.

2.2.5.- Proyectos con Microcontroladores.

Cuando se desea construir un aparato utilizando un microcontrolador como elemento central, se debe tener en cuenta varios aspectos: Para empezar, se debe conocer los alcances del proyecto, esto con el fin de escoger el microcontrolador más adecuado.

Se debe conocer las características físicas del microcontrolador escogido, tales como las funciones que desempeña cada uno de los pines, los voltajes que estos pueden soportar, los niveles de corriente que pueden soportar, etc.

Por otro lado, la parte de software o de programación también exige un buen conocimiento de las instrucciones del microcontrolador, de las herramientas que se necesita para escribir el programa y de la forma como éste se pasa a la memoria del integrado. Herramientas como el computador, donde se escribe el programa, y un circuito programador o quemador para grabarlo en la memoria del chip.

En proyectos avanzados también se debe tener conocimientos específicos de la aplicación, como las conversiones análogos a digital, las interfaces y protocolos de comunicaciones, etc.

2.3.- El Microcontrolador PIC

2.3.1.- Introducción

El PIC16F84 es un microcontrolador con la memoria de programa de tipo FLASH, lo que representa gran facilidad en el desarrollo de Prototipos y en su aprendizaje ya que no requiere borrarlo con la luz ultravioleta como las versiones EPROM sino, permite reprogramarlo nuevamente sin ser borrado con anterioridad, por está razón, utilizaremos este microcontrolador para desarrollo de nuestro proyecto de tesis.

2.3.2.- Características Técnicas.

Pines y funciones.

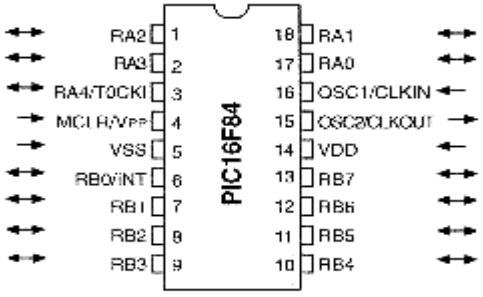


Figura 2.6. Diagrama de pines del PIC16F84

El PIC16F84 es un microcontrolador de Microchip Technology fabricado en tecnología CMOS, su consumo de potencia es muy bajo y además es completamente estático, esto quiere decir que el reloj puede detenerse y los datos de la memoria no se pierden.

El encapsulado más común para el microcontrolador es el DIP (Dual In-line Pin) de 18 pines, propio para usarlo en experimentación. La referencia completa es 16F84-04/P, para el dispositivo que utiliza reloj de 4 MHz. Sin embargo, hay otros tipos de encapsulado que se pueden utilizar según el diseño y la aplicación que se quiere realizar. Por ejemplo, el encapsulado tipo *surface mount* (montaje superficial) tiene un reducido tamaño y bajo costo, que lo hace propio para producciones en serie o para utilizarlo en lugares de espacio muy reducido, la figura 2.7 muestra los tipos de empaque que puede tener el integrado.

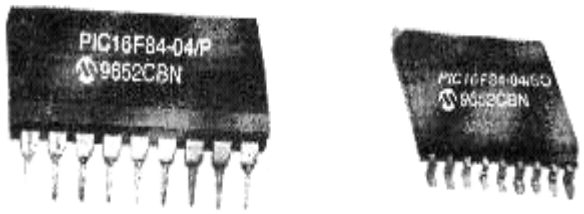


Figura 2.7 Tipos de encapsulado

Puertos del microcontrolador.

Los puertos son el puente entre el microcontrolador y el mundo exterior. Son líneas digitales que trabajan entre cero y cinco voltios y se pueden configurar como entradas o como salidas.

El PIC16F84 tiene dos puertos. El puerto A con 5 líneas y el puerto B con 8 líneas, figura 2.8. Cada pin se puede configurar como entrada o como salida independiente programando un par de registros diseñados para tal fin. En ese registro un "O" configura el pin del puerto correspondiente como salida y un " 1" lo configura como entrada.

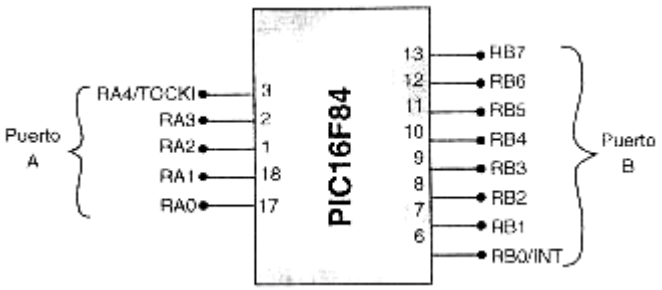


Figura 2.8 Puertos del PIC16F84

El puerto B tiene internamente unas resistencias de pull up conectadas a sus pines (sirven para fijar el pin a un nivel de cinco voltios), su uso puede ser habilitado o deshabilitado bajo control del programa. Todas las resistencias de pull-up se conectan o se desconectan a la vez, usando el bit llamado RBPU que se encuentra en el registro (posición de memoria RAM) llamado OPTION. La resistencia de pull-up es desconectada automáticamente en un pin si este se programa como salida. El pin RBO/INT se puede configurar por software para que funcione como interrupción externa, para configurarlo se utilizan unos bits de los registros INTCON y OPTION.

El pin RA4/TOCKI del puerto A puede ser configurado como un pin de entrada/ salida del temporizador /contador. Cuando este pin se programa como entrada digital, funciona como un disparador de Schmitt (Schmitt trigger), puede reconocer señales un poco distorsionadas y llevarlas a niveles lógicos (cero y cinco voltios). Cuando se usa como salida digital se comporta como colector abierto, por lo tanto, se debe poner una resistencia de pull-up (resistencia extrema conectada a un nivel de cinco voltios). Como salida, la lógica es inversa: un "O" escrito al pin del puerto entrega en el pin un" 1" lógico. Además, como salida no puede manejar cargas como fuente, sólo en el modo sumidero.

Como este dispositivo es de tecnología CMOS, todos los pines deben estar conectados a alguna parte, nunca dejarlos al aire porque se puede dañar el integrado. Los pines que no se

estén usando se deben conectar a la fuente de alimentación de +5V, como se muestra en la figura 2.9

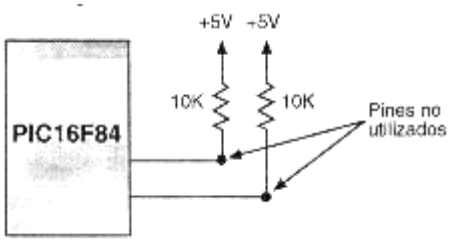


Figura 2.9 Los puertos no utilizados se deben conectar a la fuente.

La máxima capacidad de corriente de cada uno de los pines de los puertos en modo sumidero (sink) es de 25 mA y en modo fuente (source) es de 20 mA. La máxima capacidad de corriente total de los puertos es:

	PUERTO A	**PUERTO B**
Modo sumidero	80 mA	150 mA
Modo fuente	50 mA	100 mA

El consumo de corriente del microcontrolador para su funcionamiento depende del voltaje de operación, la frecuencia y de las cargas que tengan sus pines. Para un reloj de 4 MHz el consumo es de aproximadamente 2 mA; aunque este se puede reducir a 40 microamperios cuando se está en el modo sleep (en este modo el micro se detiene y disminuye el consumo de potencia). Se sale de ese estado cuando se produce alguna condición especial que veremos más adelante.

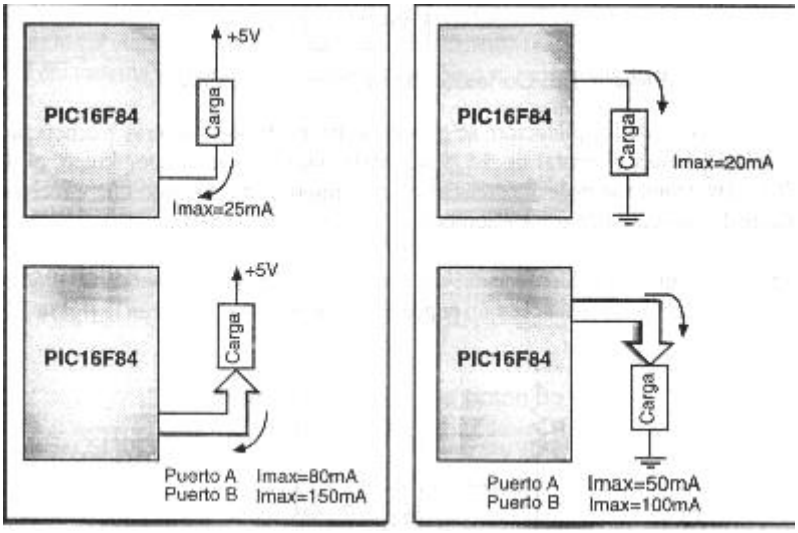


Figura 2.10 Capacidad de corriente del PIC16F84.

El oscilador externo.

Todo microcontrolador requiere un circuito externo que le indique la velocidad a la que debe trabajar. Este circuito, que se conoce como oscilador o reloj, es muy simple pero de vital importancia para el buen funcionamiento del sistema. El PIC16F84 puede utilizar cuatro tipos de oscilador diferentes. Estos tipos son:

- **RC**. Oscilador con resistencia y condensador.
- **XT.** Cristal.
- **HS.** Cristal de alta velocidad.
- **LP.** Cristal para baja frecuencia y bajo consumo de potencia.

En el momento de programar o "quemar" el microcontrolador se debe especificar que tipo de oscilador se usa. Esto se hace a través de unos fusibles llamados "fusibles de configuración".

El tipo de oscilador que se sugiere para las prácticas es el cristal de 4 MHz, porque garantiza mayor precisión y un buen arranque del microcontrolador. Internamente esta

frecuencia es dividida por cuatro, lo que hace que la frecuencia efectiva de trabajo sea de 1 MHz, por lo que cada instrucción se ejecuta en un micro segundo. El cristal debe ir acompañado de dos condensadores y se conecta como se muestra en la figura 2.11.

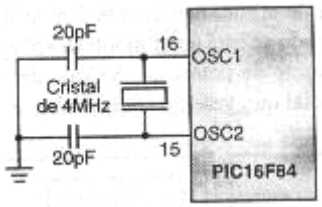


Figura 2.11Conexión de un oscilador a cristal

Dependiendo de la aplicación, se pueden utilizar cristales de otras frecuencias; por ejemplo se usa el cristal de 3.579545 MHz porque es muy económico, el de 32,768 KHz cuando se necesita crear bases de tiempo de un segundo muy precisas. El límite de velocidad en estos microcontroladores es de 10 MHz.

Si no se requiere mucha precisión en el oscilador y se quiere economizar dinero, se puede utilizar una resistencia y un condensador, como se muestra en la figura 2.12

.

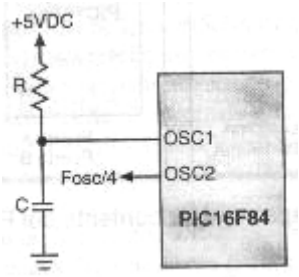


Figura 2.12. Conexión de un oscilador RC

Reset.

En los microcontroladores se requiere un pin de reset para reiniciar el funcionamiento del sistema cuando sea necesario, ya sea por una falla que se presente o porque así se halla diseñado el sistema. El pin de reset en los PIC es llamado MCLR (master clear).

El PIC16F84 admite diferentes tipos de reset:

- Al encendido (*Power On Reset*)
- Pulso en el pin MCLR durante operación normal.
- Pulso en el pin MCLR durante el modo de bajo consumo (modo sleep)
- El rebase del conteo del circuito de vigilancia (watchdog) durante operación normal.
- El rebase del conteo del circuito de vigilancia (watchdog) durante el modo de bajo consumo (sleep).

El reset al encendido se consigue gracias a dos temporizadores. El primero de ellos es el OST *(Oscillator Start-Up Timer*: Temporizadores encendido del oscilador), orientado a mantener el microcontrolador en reset hasta que el oscilador del cristal es estable. El segundo es el PWRT *(Power-üp Timer*: Temporizador de encendido), que provee un retardo fijo de 72 ms (nominal) en el encendido únicamente, diseñado para mantener el dispositivo en reset mientras la fuente se estabiliza. Para utilizar estos temporizadores, sólo basta con conectar el pin MCLR a la fuente de alimentación, evitándose utilizar las tradicionales redes RC externas en el pin de reset.

El reset por MCLR se consigue llevando momentáneamente este pin a un estado lógico bajo, mientras que el watchdog WDT produce el reset cuando su temporizador rebasa la cuenta, o sea que pasa de OFFh a OOh. Cuando se quiere tener control sobre el reset del

sistema se puede conectar un botón como en la figura 2.13.

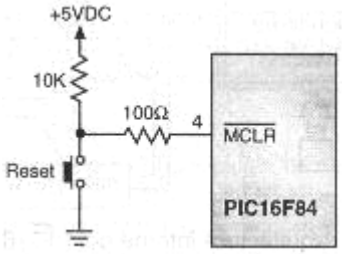


Figura 2.13 Conexión del botón de reset

2.3.3.- Arquitectura interna.

Arquitectura.

Este término se refiere a los bloques funcionales internos que conforman el microcontrolador y la forma en que están conectados, por ejemplo la memoria FLASH (de programa), la memoria RAM (de datos), los puertos, la lógica de control que permite que todo el conjunto funcione, etc.

La figura 2.14 muestra la arquitectura general del PIC16F84, en ella se pueden apreciar los diferentes bloques que lo componen y la forma en que se conectan. Se muestra la conexión de los puertos, las memorias de datos y de programa, los bloques especiales como el watchdog, los temporizadores de arranque, el oscilador, etc.

Todos los elementos se conectan entre sí por medio de buses. Un bus es un conjunto de líneas que transportan información entre dos o más módulos. Vale la pena destacar que el PIC16F84 tiene un bloque especial de memoria de datos de 64 bytes del tipo EEPROM, además de los dos bloques de memoria principales que son el de programa y el de datos o registros.

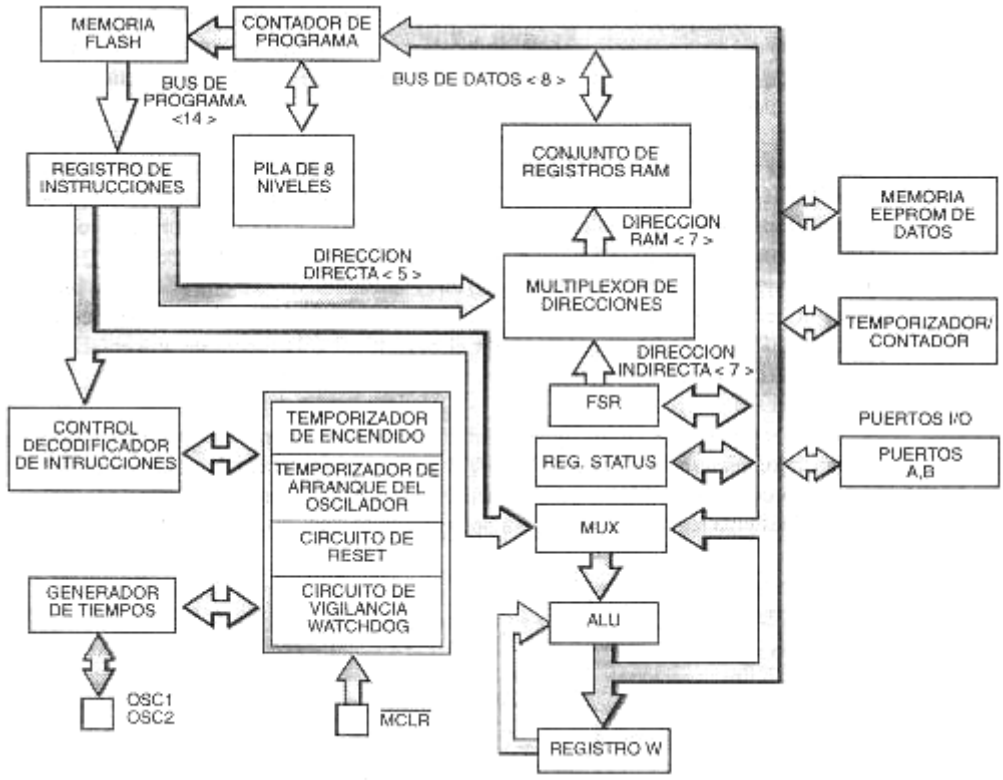


Figura 2.14 Arquitectura interna del PIC16F84

El PIC16F84 se basa en la arquitectura Harvard, en la cual el programa y los datos se pueden trabajar desde memorias separadas, lo que posibilita que las instrucciones y los datos posean longitudes diferentes. Esta misma estructura es la que permite la superposición de los ciclos de búsqueda y ejecución de las instrucciones, lo cual se ve reflejado en una mayor velocidad del microcontrolador.

Memoria de programa.

Es una memoria de 1 Kbyte de longitud con palabras de 14 bits. Como es del tipo FLASH se puede programar y borrar eléctricamente, lo que facilita el desarrollo de los programas y la experimentación. En ella se graba, o almacena, el programa o códigos que el microcontrolador debe ejecutar. Dado que el PIC16F84 tiene un contador de programa de 13 bits, tiene una capacidad de direccionamiento de 8K x 14, pero solamente tiene

implementado el primer 1K x 14 (0000h hasta 03FFh). Si se direccionan posiciones de memoria superiores a 3FFh se causará un solapamiento con el espacio del primer 1K. En la figura 2.15 se muestra el mapa de la memoria de programa.

Vector de reset.

Cuando ocurre un reset al microcontrolador, el contador de programa se pone en ceros (000H). Por esta razón, en la primera dirección del programa se debe escribir todo lo relacionado con la iniciación del mismo.

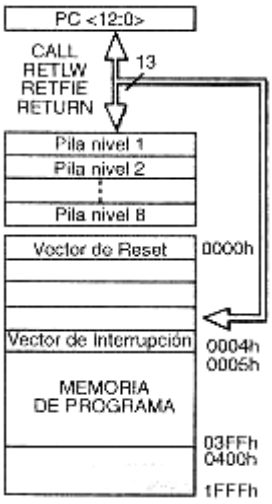


Figura 2.15 Mapa de la memoria de programa

Vector de interrupción.

Cuando el microcontrolador recibe una señal de interrupción, el contador de programa apunta a la dirección 04H de la memoria de programa, por eso, allí se debe escribir toda la programación necesaria para atender dicha interrupción.

Registros (Memoria RAM).

El PIC16F84 puede direccionar 128 posiciones de memoria RAM, pero solo tiene implementados físicamente los primeros 80 (04F en hexadecimal). De estos los primeros 12 son registros que cumplen un propósito especial en el control del microcontrolador y los 68 siguientes son registros de uso general que se pueden usar para guardar los datos temporales de la tarea que se está ejecutando, figura 2.16. Los registros están organizados como dos arreglos (páginas) de 128 posiciones de 8 bits cada una (128 x 8); todas las posiciones se pueden accesar directa o indirectamente (esta última a través del registro selector FSR). Para seleccionar que página de registros se trabaja en un momento determinado se utiliza el bit RPO del registro STATUS. A continuación haremos una descripción de los registros:

OOh o INDO: Registro para direccionamiento indirecto de datos.

Este no es un registro disponible físicamente; utiliza el contenido del FSR y el bit RPO del registro STATUS para seleccionar indirectamente la memoria de datos o RAM del usuario; la instrucción determinará que se debe realizar con el registro señalado.

Olh o TMRO. Temporizador /contador de 8 bits.

Este se puede incrementar con una señal extrema aplicada al pin RA4/TOCKI o de acuerdo a una señal interna proveniente del reloj de instrucciones del microcontrolador. La ruta de incremento del registro se puede determinar por medio de un preescalador, localizado en el registro OPTION. Como una mejora con respecto a sus antecesores, se le ha agregado la generación de interrupción cuando se rebasa la cuenta (el paso de OFFh a OOh).

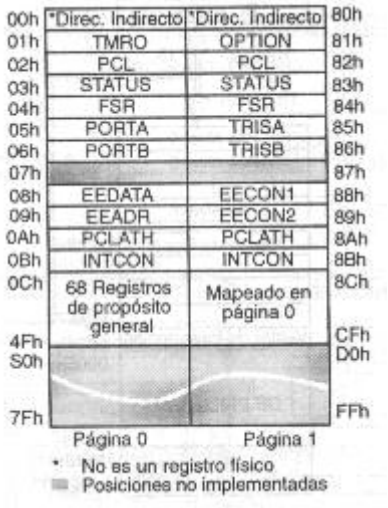


Figura 2.16. Registros del PIC16F84

02h o PCL: Contador de programa.

Se utiliza para direccionar las palabras de 14 bits del programa del usuario que se encuentra almacenado en la memoria ROM; este contador de programas es de 13 bits de ancho, figura 2.17. Sobre el byte bajo, se puede escribir o leer directamente, mientras que sobre el byte alto, no. El byte alto se maneja mediante el registro PCLATH (OAh). A diferencia de los PIC de primera generación, el 16F84 ante una condición de reset inicia el contador de programa con todos sus bits en "cero". Durante la ejecución normal del programa, y dado que todas las instrucciones ocupan sólo una posición de memoria, el contador se incrementa en uno con cada instrucción, a menos que se trate de alguna instrucción de salto.

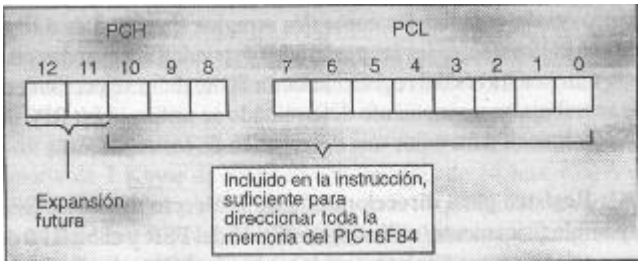


Figura 2.17. Contador de programa (13 bits)

En una instrucción CALL o GOTO, los bits PC<10;0> se cargan desde el código de operación de la instrucción, mientras que los bits PC<11:12> lo hacen desde el PCLATH<4:3>- Como solamente el primer 1K de memoria está implementado, el código de operación de la instrucción puede contener la dirección destino, eso quiere decir que se pueden hacer saltos y llamados a subrutinas sin necesidad de tener en cuenta la paginación de memoria de programa.

En otras instrucciones donde PCL es el destino, PC<12:8> se carga directamente desde el PCLATH<4:0>, por ejemplo en el caso de la instrucción ADDWF. Esto se debe tener en cuenta cuando se desea hacer lectura de tablas usando el comando: ADDWF PC,1 , en este caso se debe tener en cuenta que la tabla debe estar comprendida dentro de un solo bloque de 256 bytes (0-255, 256-511, etc.)

03h o STATUS: Registro de estados.

Contiene el estado aritmético de la ALU, la causa del reset y los bits de preselección de página para la memoria de datos. La tabla 2.4 muestra los bits correspondientes a este registro. Los bits 5 y 6 (RPO y RP1) son los bits de selección de página para el direccionamiento directo de la memoria de datos; solamente RPO se usa en los PIC16F84. RP1 se puede utilizar como un bit de propósito general de lectura/escritura. Los bits TO y PD no se pueden modificar por un proceso de escritura; ellos muestran la condición por la cual se ocasionó el último reset.

Tabla 2.4. Registro de estados

Registro: STATUS

IRP	RP1	RPO	TO	PD	Z	DC	C

BIT 7
Dirección: 03h
Condición de reset. 000??Xxx

IRP: Selector de página para direccionamiento indirecto
Este BIT no se utiliza efectivamente en el PIC16F84, por lo que se pude utilizar como un BIT de propósito general.
RP1, 0: Selectores de página para direccionamiento directo
Solamente RPO se utiliza en el PIC16F84. RP1 se pude utilizar como un BIT de propósito general.
TO: Time out Bit del finalización del temporizador.
Se coloca en 0 cuando el circuito de vigilancia watchdog finaliza la temporización.
PD: Power Down o Bit de bajo consumo. Se coloca en 0 por la Instrucción SLEEP.
Z: Zero o bit de cero. Se coloca en 1 cuando el resultado de operación Lógicas o aritmética es cero.
DC: Digit Carry o Bit de acarreo de dígito. En operaciones aritméticas se Activa cuando hay acarreo entre el bit 3 y el 4.
C: Carry o Bit de acarreo. En instrucciones aritméticas se activa cuando Se presenta acarreo desde el bit más significativo del resultado.

04h o FSR: Registro selector de registros.

En asocio con el registro INDO, se utiliza para seleccionar indirectamente los otros registros disponibles. Mientras que los antecesores del PIC16F84 sólo poseían 5 bits activos, en este microcontrolador se poseen los 8 bits. Si en el programa no se utilizan llamadas indirectas, este registro se puede utilizar como un registro de propósito general.

Para entender mejor el funcionamiento de este registro veamos un programa simple que borra el contenido de la memoria RAM, empleando direccionamiento indirecto.

Tabla 2.5. Programa simple que borra el contenido de la memoria RAM

```
MOVLW   20      ; inicializa el puntero en la memoria RAM
MOVWF   FSR     ; que se va a borrar
NEXTCLRF INDO   ; borra el registro indexado (es decir el que
                ; está siendo direccionado por el FSR).
INMCF   FSR, R  ; incrementa el puntero
BTFSS   FSR, 5  ; pregunta si ya acabó el banco de memoria
GOTO    NEXT    ; sigue borrando los registros que faltan
```

continúa……

05h o PORTA: Puerto de Entrada /Salida de 5 bits. Este puerto, al igual que todos sus similares en los PIC, puede leerse o escribirse como si se tratara de un registro cualquiera. El registro que controla el sentido (entrada o salida) de los pines de este puerto está localizado en la página 1, en la posición 85h y se llama TRISA.

06h o PORTB: Puerto de entrada /salida de 8 bits. Al igual que en todos los PIC, este puede leerse o escribirse como si se tratara de un registro cualquiera; algunos de sus pínes tienen funciones alternas en la generación de interrupciones. El registro de control para la configuración de la función de sus pines se localiza en la página 1, en la dirección 86h y se llama TRISB.

08h o EEDATA: Registro de datos de la EEPROM. Este registro contiene el dato que se va a escribir en la memoria EEPROM de datos o el que se leyó de ésta.

09h o EEADR: Registro de dirección de la EEPROM. Aquí se mantiene la dirección de la EEPROM de datos que se va a trabajar, bien sea para una operación de lectura o para una de escritura.

OAh o PCLATH: Registro para la parte alta de la dirección. Este contiene la parte alta del contador de programa y no se puede acceder directamente.

OBh o INTCON: Registro para el control de interrupciones. Es el encargado del manejo de las interrupciones y contiene los bits que se muestran en la Tabla 2.6.

81h u OPTION: Registro de configuración múltiple. Posee varios bits para configurar el preescalador, la interrupción externa, el timer y las características del puerto B. Los bits que contiene y las funciones que realiza este registro se muestran en la tabla 2.7. El preescalador es compartido entre el MTRO y el WDT; su asignación es mutuamente excluyente ya que solamente puede uno de ellos ser preescalado a la vez.

85h o TRISA: Registro de configuración del puerto A. Como ya se mencionó, es el registro de control para el puerto A. Un "cero" en el bit correspondiente al pin lo configura como salida, mientras que un "uno" lo hace como entrada.

86h o TRISB: Registro de configuración del puerto B. Orientado hacia el control del puerto B. Son válidas las mismas consideraciones del registro anterior.

Tabla 2.6 Registro INTCON

Registro: INTCON

GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

BIT 7
Dirección: 0Bh
Condición de reset. 0000000Xb

GIE: Global Interrupt Enable o habilitador general de interrupciones.
0: deshabita todas las interrupciones
1: Habilita las interrupciones.
EEIE: EEPROM Write Interrupt Enable o Habilitación de interrupción por escritura de la EEPROM
0: la deshilabilita
1: la habilita
TOIE: TMRO Interrupt Enable o habitación de interrupción del Temporalizado TMRO.
0: la deshabilita
1: la habilita
INTE: INT Interrupt Enable o habitación de la interrupción INT
0: la deshabilita
1: la habilita
RBIE: RBIF Interrupt Enable o habitación de la interrupción RBIF
0: la deshabilita
1: la habilita
TOIF: TMRO Overflow Interrupt Flag o bandera de la interrupción por Sobrefujo del TMRO. Se coloca en 1 cuando el TMRO pasa de Of. A 00h; ésta debe ser puesta a 0 por programa.
INTF: INT INterrupt Flajo bandera de interrupción INT: Se coloca en 1 Cuando la interrupción INT (RB<0>) ocurre ésta debe ser puesta a 0 por el programa.
RBIF: RB Port Change Interrupt Flag o Bandera de interrupción por cambio En el puerto B.
Se coloca en 1 cuando una de las entradas RB<7:4> cambia; éste debe ser puesta a 0 por programa.

EEPROM de datos.

Este es el registro de control de la memoria de datos y sólo destina cinco bits para ello, los más bajos; los tres bits superiores permanecen sin implementar. En la tabla 2.8. se muestran las funciones de estos bits.

89h o EECON2: Registro auxiliar para control de la memoria EEPROM de datos.

Registro que no está implementado físicamente en el microcontrolador, pero que es

necesario en las operaciones de escritura en la EEPROM de datos; ante cualquier intento de lectura se obtendrán "ceros".

OCh a 4Fh: Registros de propósito general. Estas 68 posiciones están implementadas en la memoria RAM estática, la cual conforma el área de trabajo del usuario; a ellas también se accede cuando en la página 1 se direccionan las posiciones 8Ch a CFh.

Tabla 2.7 Registro OPTION

Registro: OPTION

RBPU	INTEDG	GRTS	RTE	PSA	PS2	PS1	PSO

BIT 7 bit 0
Dirección: 81h
Condición de reset. 11111111b

RBPU: PortB Pull-up Enable o habilitador de pull-up del puerto B.
0: Habilita las pull-UPS internas
1: las deshabilita
INTEDG: INT Interrupt Edge Select o Selector de flanco de la interrupción INT
0: flanco de bajada
1: flanco de subida
RTS: TMRO Signal Source o Fuente de señal de TMRO.
0: ciclo de intrucciones interno (temporalizador)
1: transmisión en el pin RA4/TOCK (contador)
RTE: TMRO Signal Edge o Flanco de la señal TMRO
0: incremento de transición de bajo a alto
1: incremento de transición de alto a bajo
PSA: Prescaler Assignment o Asignación del preescalador
0: TRMO (contador/temporizador)
1: WDT (circuito de vigilancia)
PS2, 1,0: Prescaler Value o valores del preescalador

Valor	TMRO	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

Esto se ha diseñado así para evitar un excesivo cambio de páginas en el manejo de la RAM del usuario, agilizando los procesos que se estén llevando acabo y descomplicando la labor del programador.

Registro de trabajo W.

Este es el registro del trabajo principal, se comporta de manera similar al acumulador en los microprocesadores. Este registro participa en la mayoría de las instrucciones.

Pila (Stack).

Estos registros no forman parte de ningún banco de memoria y no permite el acceso por parte del usuario. Se usan para guardar el valor del contador del programa cuando se hace un llamado a una subrutina o cuando se atiende una interrupción; luego, cuando regresa a seguir ejecutando su tarea normal, el contador de programa recupera su valor leyéndole nuevamente desde la pila. El PIC16F84 tiene una pila de 8 niveles, esto significa que se puede anidar 8 llamados a subrutina sin tener problemas.

Tabla 2.8 registro EECON1

Registro: EECON1

U	U	U	EEIF	WRERR	WREN	WR	RD

BIT 7 bit 0
Dirección: 88h
Condición de reset. 0000X000b

U: Unimplemented No implementadas.
Estos bites se leen como ceros
EEIF: EEPROM Write Completion Interrupt Flag o Bandera de finalización
De la escritura. Se coloca en "1" cuando finaliza con éxito la escritura
En la EEPROM de datos; se debe colocar en "0" por programa. El bit
De habilitación correspondiente es el EEIE, localizado en el registro
INTCON (0B<6>).
WRERR: Write Error Flag o Bandera de error de escritura. Se coloca en 1
Cuando la operación de escritura termina prematuramente,
Debido a cualquier condición de reset
WREN Write Enable o habilitación de escritura. Si se coloca en "0" no
No permite las operaciones de escritura; en 1 las habilita.
WR: Write Control o Control de escritura. Al colocarse en "1" inicia
Un ciclo de escritura. Este bit sólo es puesto a "0" por Hardware,
Una vez la escritura termina.
RD: Read Control o Control de lectura.
Al colocarse en "1" se inicia una lectura de la EEPROM de datos,
La cual toma un ciclo del reloj de instrucciones. Este bit solo se
limpia (se coloca en "0") por hardware, al finalizar la lectura de la
posición de la EEPROM.

Características especiales .

Algunos elementos que forman parte de los PIC no se encuentra en Microcontroladores de otros fabricantes o simplemente representa alguna ventaja o facilidad a la hora de hacer un diseño. Veamos una breve descripción de las más significativas:

Circuito de vigilancia (Watchdog Timer).

Su función es restablecer el programa cuando este se a perdido por fallas en la programación o por alguna razón externa. Es muy útil cuando se trabaja en ambientes con mucha interferencia o ruido electromagnético. Está conformado por oscilador RC que se encuentra dentro del microcontrolador.

Este oscilador corre de manera independiente al oscilador principal. Cuando se habilita su funcionamiento, dicho circuito hace que el microcontrolador sufra un reset cada determinado tiempo (que se puede programar entre 18ms y 2 segundos). Este reset lo puede evitar el usuario mediante una instrucción especial del microcontrolador (**CLRWDT:** borrar el conteo del Watchdog), la cual se debe ejecutar antes de que se termine el periodo nominal de dicho temporizador. De esta manera, si el programa se ha salido de su flujo normal, por algún ruido o interferencia extrema, el sistema se reiniciará (cuando se acabe el tiempo programado y no se haya borrado el contador) y el programa puede restablecerse para continuar con su funcionamiento normal.

En las primeras prácticas no se utiliza el circuito de vigilancia para facilitar el trabajo; por eso, en el momento de programar el microcontrolador se debe seleccionar en los fusibles de configuración "watchdog timer OFF". Más adelante veremos algunos ejemplos que ilustran su funcionamiento y la manera de utilizarlo.

Temporizador de encendido (Power-up Timer) .

Este proporciona un reset al microcontrolador en el momento de conectar la fuente de alimentación, lo que garantiza un arranque correcto del sistema. En el momento de grabar el micro se debe habilitar el fusible de configuración Power-up Timer", para ello se debe seleccionar la opción "ON". Su tiempo de retardo es de 72 milisegimdos.

Modo de bajo consumo (sleep).

Esta característica permite que el microcontrolador entre en un estado pasivo donde consume muy poca potencia. Cuando se entra en este modo el oscilador principal se detiene, pero el temporizador del circuito de vigilancia (watchdog) se reinicia y empieza su conteo nuevamente. Se entra en ese estado por la ejecución de una instrucción especial (llamada SLEEP) y se sale de él por alguna de las siguientes causas: cuando el microcontrolador sufre un reset por un pulso en el pin MCLR, porque el watchdog hace que se reinície el sistema o porque ocurre una interrupción al sistema.

Interrupciones.

Este microcontrolador incluye el manejo de interrupciones, lo cual representa grandes ventajas. El PÍC16F84 posee cuatro fuentes de interrupción a saber:

Interrupción externa en el pin RBO/INT

- Finalización del temporizador/contador TMRO
- Finalización de escritura en la EEPROM de datos
- Cambio de nivel en los pines RB4 a RB7

El registro OBh o INTCON contiene las banderas de las interrupciones INT, cambio en el puerto B y finalización del conteo del TMRO, al igual que el control para habilitar o deshabilitar cada una de las fuentes de interrupción, incluida la de escritura en memoria EEPROM. Sólo la bandera de finalización de la escritura reside en el registro 88h (EECON1<4>).

Si el bit GIE (Global Interrupt Enable) se coloca en O, deshabilita todas las interrupciones. Cuando una interrupción es atendida, el bit GIE se coloca en O

automáticamente para evitar interferencias con otras interrupciones que se pudieran presentar, la dirección de retomo se coloca en la pila y el PC se carga con la dirección 04h.

Una vez en la rutina de servicio, la fuente de la interrupción se puede determinar examinando las banderas de interrupción. La bandera respectiva se debe colocar, por software, en cero antes de regresar de la interrupción, para evitar que se vuelva a detectar nuevamente la misma interrupción.

La instrucción RETFIE permite al usuario retomar de la interrupción, a la vez que habilita de nuevo las Interrupciones, al colocar el bit GIE en uno. Debe tenerse presente que solamente el contador de programa es puesto en la pila al atenderse la interrupción; por lo tanto, es conveniente que el programador tenga cuidado con el registro de estados y el de trabajo, ya que se pueden producir resultados inesperados si dentro de ella se modifican.

Interrupción externa. Actúa sobre el pin RBO/INT y se puede configurar para activarse con el flanco de subida o el de bajada, de acuerdo al bit INTEDG (OPTION<6>).

Cuando se presenta un flanco válido en el pin INT, la bandera INTF (INTCON<1>) se coloca en uno. La interrupción se puede deshabilitar colocando el bit de control INTE (INTCON<4>) en cero. Cuando se atiende la interrupción, a través de la rutina de servicio, INTF se debe colocar en cero antes de regresar al programa principal. La interrupción puede reactivar al microcontrolador después de la instrucción SLEEP, si previamente el bit INTE fue habilitado.

Interrupción por finalización de la temporización. La superación del conteo máximo (OFFh) en el TMRO colocará el bit TOIF en uno (INTCON<2>). El bit de control respectivo es TOIE (INTCON<5>).

Interrupción por cambio en el puerto RB. Un cambio en los pines del puerto B <7:4> colocará en uno el bit RBIF (INTCON<0>). El bit de control respectivo es RBIE (INTCON<3>).

Interrupción por finalización de la escritura. Cuando la escritura de un dato en la EEPROM finaliza, se coloca en 1 el bit EEIF (EECON1<4>). El bit de control respectivo es EEIE (ÍNTCON<6>)

Memoria de datos EEPROM.

El PIC 16F84 tiene una memoria EEPROM de datos de 64 posiciones (Oh a 3Fh), de 8 bits cada una. Este bloque de memoria no se encuentra marcado en ningún banco, el acceso a esas posiciones se consigue a través de dos registros de la RAM:

- El registro EEADR (posición 09), que debe contener la dirección de la posición de la EEPROM a ser accesada
- El registro EEDATA (posición 08), que contiene el dato de 8 bits que se va a escribir o el que se obtuvo de la última lectura.

Adicionalmente, existen dos registros de control: el EECON1 (88h), que posee cinco bits que manejan las operaciones de lectura/ escritura y el EECON2 (89h), que aunque no es un registro físico, es necesario para realizar las operaciones de escritura.

La lectura toma un ciclo del reloj de instrucciones, mientras que la escritura, por ser controlada por un temporizador incorporado, tiene un tiempo nominal de 10 milisegundos,

este tiempo puede variar con la temperatura y el voltaje. Cuando se va a realizar una operación de escritura, automáticamente se hace primero la operación de borrado. El número típico de ciclos de borrado/ escritura de la EEPROM de datos es de 1.000.000.

Fusibles de configuración.

El PIC 16F84 posee cinco fusibles, cada uno de los cuales es un bit. Estos fusibles se pueden programar para seleccionar varias configuraciones del dispositivo: tipo de oscilador, protección de código, habilitación del circuito de vigilancia y el temporizador al encendido. Los bits se localizan en la posición de memoria 2007h, posición a la cual el usuario sólo tiene acceso durante la programación del microcontrolador.

Cuando se programa la protección de código, el contenido de cada posición de la memoria no se puede leer completamente, de tal manera que el código del programa no se puede reconstruir. Adicionalmente, todas las posiciones de memoria del programa se protegen contra la reprogramación.

Una vez protegido el código, el fusible de protección sólo puede ser borrado (puesto a 1) si se borra toda la memoria del programa y la de datos.

Las pull-ups internas.

Cada uno de los pines del puerto B tiene un débil elemento pull-up interno (250 uA típico); este elemento es automáticamente desconectado cuando el pin se configura como salida. Adicionalmente, el bit RBPU (OPTION<7>) controla todos estos elementos, los cuales están deshabilitados ante una condición de reset. Estos elementos pull-up son especialmente útiles cuando el microcontrolador va a colocarse en el modo de bajo consumo, ya que ayudan a no tener las entradas flotantes, significando una reducción en el consumo de corriente.

El conjunto de instrucciones.

Estas se clasifican en orientadas a registros, orientadas al bit y operaciones literales y de control. Cada instrucción es una palabra de 14 bits, dividida en un código de operación (el cual especifica la orden a ejecutar) y uno o más operandos sobre los que se actúa. En el Manual del Usuario se encuentra la lista completa de instrucciones, la cual incluye ejemplos y explicaciones. Como se puede observar allí, en total son 35, las cuales tardan un ciclo de máquina, a excepción de los saltos, que toman dos ciclos.

2.3.4.- Circuitos de experimentación.

El proyecto del entrenador K-148 se diseñó pensando en utilizar al máximo la capacidad de los microcontroladores PIC16F84, 16C71 y 12C508. Aunque, puede ser usado por otras referencias que tengan pines compatibles. Este dispositivo posee todos los elementos necesarios para realizar las prácticas y ejercicios propuestos.

Pensando en facilitar su aprendizaje y aprovechar al máximo las ventajas de dichos dispositivos, se diseñó el sistema o tablero de entrenamiento, el cual incluye todos los circuitos necesarios para la implementación de las comunicaciones seriales, el manejo de los puertos, las conversiones análogos a digital, etc. Adicionalmente, para experimentar con el PIC16F84 sólo se requiere el circuito programador de microcontroladores (CK-175 de CEKIT) y el entrenador PIC avanzado (K-148).

Bloques funcionales del entrenador.

La principal aplicación de este aparato es servir como elemento de soporte a los alumnos que desean aprender el manejo de dichos dispositivos. Está conformado por un conjunto de circuitos, independientes entre sí, para que el estudiante pueda realizar los más variados experimentos y proyectos sencillos (para los que se inician), o de cierta

complejidad (para los más experimentados). Cada bloque tiene listas las conexiones a la fuente de alimentación y a los diferentes elementos que lo componen, de esta forma, sólo se requiere conectar las entradas y/o salidas al sitio que corresponda. Dichas conexiones se hacen fácilmente empleando alambre telefónico, ya que se implemento un sistema de conectores que permiten hacer esta labor fácilmente.

El PIC16C84

El PIC16C84 es un microcontrolador de la familia Microchip, totalmente compatible con el PIC16F84. Su principal característica es que posee memoria "EEPROM" en lugar de memoria Flash, pero su manejo es igual. Con respecto al PIC16F84, este microcontrolador presenta dos diferencias:

- La memoria de datos tiene menor tamaño, aquí se tienen 32 registros de propósito general (el mapa de memoria de datos llega hasta 2FH).

- En el momento de programar el microcontrolador, el fusible de selección del temporizador de arranque (Power Up Timer) trabaja de forma inversa, es decir, si en el PIC16F84 se selecciona la opción "Low" para activarlo, en el PIC 16C84 se debe seleccionar "High".

2.4.- Lenguajes Ensambladores.

Este microcontrolador ha sido reemplazado de forma gradual por el PIC16F84, por lo tanto, los diseños que lo utilicen como elemento de control deben ser actualizados. Aunque, como se ve, es un proceso casi transparente.

El programa es lo que hace funcionar al microcontrolador; las tareas que se ejecutan y las secuencias que se realicen dependen de él. Las instrucciones que componen un

programa se escriben en un editor de texto normal (ej. el editor del MS-DOS) y son propias de cada familia de microcontroladores.

Para convertir estas instrucciones, que son comprensibles para el programador en un código que pueda ser entendido por el microcontrolador, se utiliza un programa llamado ensamblador. Utilizaremos en adelante un programa ensamblador de Microchip Tecnology llamado MPASM, el cual es fácil de manejar, flexible, de libre distribución, la detección de errores es simple y tiene la ventaja de ser el más usado en el mundo. Las aplicaciones de los manuales de Microchip y los artículos publicados en las principales revistas, lo utilizan como base.

El programa con las instrucciones comprensibles por el programador se llama código fuente, y el que genera el ensamblador y se utiliza para programar el microcontrolador se llama código objeto.

Para empezar a escribir los programas del PIC, se deben copiar todos los archivos que están en el disquete del curso al disco duro del computador. Cree un directorio llamado «CURSOPIC» y luego copie todo del disquete. El procedimiento es el siguiente:

1. Sitúese en el directorio raíz del disco duro «C:\>».
2. Cree el directorio escribiendo: «C:\>md cursopic» , luego oprima «Enter».
3. Copie los archivos del disquete: «C:\>copy a :*.* c:cursopic».
4. Para trabajar entre al directorio: «C:\>cd cursopic» , luego oprima «Enter».
5. En la pantalla debe aparecer: «C:\CURSOPTC>».

Ahora está listo para escribir los programas, usar el ensamblador, programar los microcontroladores y entrar al maravilloso mundo de los PIC's.

2.4.1.- Código Fuente.

El programa debe cumplir unas normas simples para ser aceptado por el ensamblador, como por ejemplo el tipo de microcontrolador usado, la dirección de inicio del programa en el micro, las etiquetas, las instrucciones, etc.

Para entender más fácil este concepto vamos a escribir un programa corto y explicaremos paso a paso todo lo que este contiene, tabla 2.9. El ejercicio consiste en encender cuatro LED's que están conectados en el puerto B del microcontrolador que hace parte del circuito entrenador.

Si en el computador estamos en el directorio de trabajo «cursopic», se debe escribir la siguiente línea:

«C:\CURSOPIC>edit prueba.asm» , y luego oprimir «Enter».

De esta forma, entramos en el editor de texto del MS-DOS y podemos escribir las instrucciones correspondientes. El programa se escribe en cuatro columnas como se ve en el ejemplo. Asegúrese de «guardar» o «salvar» cada vez que vaya a salir del editor.

Tabla2.9. Ejemplo de cómo escribir un programa

```
;=================================================================

list  p=16F84

;-----------------------------------------------------------------------------------------------

    ptob equ 06                                    ;el puerto b está en la
                                                   dirección 06 de la memoria RAM
;-----------------------------------------------------------------------------------------------

reset       org   0                                ;el vector de reset es lo dirección 00 de la

                                                   ;memoria de programa
```

```
Inicio      movlw 0                 ;carga el registro w con 00
            tris  ptob              ;copia el valor de w al registro que programa
                                    ;el sentido de los pines del puerto B. En
                                    ;este caso los pone como salidas
            movlw 0f                ;carga el registro w con 0f
            movwf ptob              ;carga el puerto B con el valor 0f que estaba
                                    ;en el registro w

ciclo nop                           ;el programa se queda en este ciclo hasta que
                                    ;sufra  un reset

goto  ciclo
end
;-------------------------------------------------------------------------------------------------
;En el momento de quemar el microcontrolador se debe seleccionar los fusibles de configuración siguientes:
;oscilador                  XT
;watchdog                   OFF
;código de protección       OFF
;power-up timer             ON
```

Las reglas básicas para escribir un programa para un microcontrolador PIC tienen que ver con palabras o comandos especiales, signos de puntuación, tabulaciones para escribir en columnas, etc. A continuación, haremos una descripción de cada una de ellas y veremos que tipos de líneas de instrucciones o de comandos se pueden usar. Como se ve, cada línea

de programa tiene cuatro campos o columnas; cada una contiene diferente información y recibe un nombre específico, a saber:

Etiqueta	**Operación**	**Operandos**	**Comentario**
ej.: inicio	movlw	O	;carga el registro w con O

Las tabulaciones se utilizan en el editor de texto para separar el contenido del programa en columnas. El programa ensamblador entiende este formato y entonces puede proceder a ensamblarlo. La longitud del tabulador (8, 9010 espacios) no es crítica, el ensamblador reconoce cuando se pasa de una columna a otra.

Etiquetas.

Son nombres simbólicos que se asignan a una dirección de la memoria. Por ejemplo, cuando se escribe la palabra inicio, se está asignando el símbolo inicio a un valor propio del microcontrolador, que es la dirección de memoria correspondiente donde quedará grabada esa instrucción.

Esto es de gran utilidad porque no es necesario trabajar con los números reales de las direcciones físicas, sino que se puede trabajar con algunos símbolos o palabras que son de muy fácil recordación, sin importar en donde esté ubicado. Las reglas para definir etiquetas son muy simples:

- Deben empezar en la primera posición de la columna uno.
- Pueden incluir caracteres alfanuméricos, línea de subrayado (_) e interrogación.
- Su longitud no debe exceder de 31 caracteres.

Operación.

Es la instrucción del microcontrolador que se ejecuta. Todo el conjunto de instrucciones del PIC16F84.

Operandos.

Son los registros o cantidades sobre los que se realizan las operaciones o instrucciones. Puede ser un registro de la memoria de datos o un valor constante que comúnmente se conoce como «literal». Cuando se utilizan dos operandos, el primero es el operando fuente y el segundo el operando destino.

Literal.

Es una constante, usualmente un número hexadecimal. Ejemplo:

```
start    movlw    00
```

En este caso, el literal (número) es el 00. Los literales son valores que se definen usando la instrucción movlw y otras que se verán más adelante.

Comentario.

El ensamblador ignora esta línea en el momento de generar el código objeto, pero para la persona que hace el programa es muy importante ya que ahí puede escribir la idea o la explicación de lo que está haciendo en esa línea del programa.

Punto y coma (;).

Es un delimitador; hace que el ensamblador ignore la línea de texto que se encuentra a la derecha de él. Se usa para escribir comentarios acerca de la instrucción particular que se tiene y de la acción que está ejecutando el microcontrolador.

2.4.2.- Clases de líneas en un programa.

Como ya vimos qué formato tiene una línea de programa, veremos qué tipo de líneas pueden escribirse. Se pueden usar cuatro tipos diferentes, la de igualdad (equ), la de origen (org), la de instrucción normal y la final (end), figura 2.10

Tabla2.10. Tabulaciones y tipos de línea

	Etiqueta	**Símbolo de igualdad**	**Número hexadecimal o dirección de memoria**	**Comentarios**
Igualdad:	_____	equ	_____	_____ ;
	Etiqueta	**Símbolo de origen**	**Dirección hexadecimal**	**Comentarlos**
Origen:	_____	org	_____	_____ ;
	Etiqueta	**Instrucción**	**Registro o número literal**	**Comentarios**
Programa:	_____	_____	_____	; _____
	No se usa	**Símbolo de línea final**	**No se usa**	**Comentarios**
Final:		end		____;___

Igualdades

Se utilizan para asignar un nombre o símbolo a un valor específico (hexadecimal). Por ejemplo cuando se hace:

```
ptob    equ    06
```

Se está asignando el nombre ptob al valor 06, que corresponde a la dirección física de el puerto B en la memoria RAM del microcontrolador.

De esta forma, cuando se quiere hacer referencia a un pin o a todo el puerto B no hay necesidad de buscar en la hoja de datos del micro cual es su dirección física, sino que se puede hacer referencia a ptob, el ensamblador así lo entenderá. Además, para la persona que escribe el programa, es mucho más sencillo.

También se puede usar para asignar nombres a un número cualquiera. Por ejemplo:

```
r09         equ         09
min         equ         01
```

Se asigna el nombre r09 al registro 09 de la memoria RAM del microcontrolador,. De esta forma, siempre que se refiera a dicha posición, se puede hacer a r09.

Por ejemplo :

```
movlw      01
 movwf      r09
```

En este caso se cargó el acumulador con el valor 01 y luego se pasó a la posición o registro de memoria de la dirección 09, que está representado por el nombre r09.

En un programa se podría tener la siguiente línea :

```
bsf         r09,min
```

En ella se hace referencia al bit número 1 del registro de memoria 09.

Origen.

«ORG» es una directiva del ensamblador que se usa para definir el sitio (de la memoria) donde se desea escribir alguna porción del programa en particular. Ejemplo:

```
org          O
```

En este caso se indica que las instrucciones que vienen a continuación quedarán a partir de la dirección de memoria 00 en el microcontrolador. La línea origen se puede usar para varios fines :

Al especificar el vector de reset del microcontrolador, para escribir la instrucción que le indica al micro que debe ir al inicio del programa, así:

```
reset       org         O
goto        inicio
```

En este caso, cuando se resetea el micro, el contador de programa apunta a la dirección O y entonces ejecuta la instrucción que le dice que vaya a la parte de inicio del programa, que puede estar escrita algunas líneas más abajo.

En el vector de interrupción, ejemplo:

```
ínter       org         04
movlw       05
```

Cuando el microcontrolador detecta una señal de interrupción, el contador de programa apunta a la dirección 04 de la memoria. Entonces, allí se debe escribir la porción de programa que se encarga de atender dicho requerimiento de interrupción. En el ejemplo, la primera instrucción que se escribe y queda grabada en la dirección 04 es:

```
movlw       05
```

Final.

Se usa para decirle al ensamblador que el programa ha terminado; se usa la directiva end.

Formato de un programa

En vista de las descripciones anteriores, podemos resumir el formato de un programa de la forma en que muestra a continuación.

```
                    ;==================================
Encabezado          list p=16F84
                    ;----------------------------------------------------
Igualdades
generales           ptoa equ 05
                    ptob equ 06
                    ;----------------------------------------------------
Otras
igualdades
                    ;----------------------------------------------------
Programa
Final               end
                    ;==================================
```

Encabezamiento del programa.

La información al comienzo del programa se llama encabezado, ejemplo:

```
list        p = l6F84
```

«List» es un comando del ensamblador que indica algunas directrices al inicio del programa. Por ejemplo, la directriz p = 16F84, indica cuál tipo de microcontrolador se va a grabar con el programa que se está escribiendo.

2.4.3.- Archivos y Formatos de Ensamblados.

Cuando se escribe un programa en el editor del DOS, se debe crear un archivo con extensión .asm, como en el ejemplo anterior: prueba. asm. El ensamblador sólo acepta archivos con esa extensión.

Cuando se ensambla un programa, se generan automáticamente varios archivos que tienen el mismo nombre del archivo fuente, pero con extensiones diferentes, a saber:

Un archivo con extensión. lst (listado), que es muy similar al archivo. asm, pero con algunas modificaciones como la inclusión de los números de las líneas, entre otras. Un archivo con extensión .err (error), que contiene una lista donde se especifica si el programa escrito tiene errores. Si los tiene, dice en que línea y de que tipo.

Un archivo con extensión. hex (hexadecimal), que es el código objeto hexadecimal con el cual se puede grabar el microcontrolador.

Si se desea, también se puede generar un archivo con extensión .obj; este es un código objeto diferente del .hex, pero que en algunos quemadores de PIC se puede utilizar.

2.4.4.- Como Ensamblar un programa.

Cómo ensamblar un programa.

Para ver el funcionamiento del programa ensamblador se utiliza el programa **prueba.asm** que se ha usado como ejemplo . Los pasos son los siguientes:

Asegúrese que el programa sea escrito en forma idéntica al original. El archivo debe grabarse con el nombre **prueba.asm.**

Escriba el comando «C:\CURSOPIC>MPASM» , luego oprima «Enter».

Debe aparecer un pantallazo como el que se muestra en la figura 2.18

Ubíquese en la casilla SOURCE FILENAME, y presione con el mouse la tecla BROWSE para obtener el listado de los archivos **asm.** del directorio.

Seleccione el archivo llamado prueba.asm; cuando esté seleccionado oprima «Enter».

Figura 2.18. Segundo pantallazo del ensamblador

En la primera casilla de la pantalla se verá el nombre del archivo, pase a escoger el tipo de microcontrolador, de la misma forma que se escoge el archivo. Seleccione el 16F84.

Igualmente seleccione en la opción de generar el archivo o lista de errores. Para ensamblar seleccione la tecla ASSEMBLE.

Luego de ensamblado, aparecen en la pantalla unas líneas que dicen si se encontraron o no errores de sintaxis en el programa y si hay mensajes de advertencia o no. Si los hay, debe revisar el archivo .err donde muestra que tipo de mensajes se tienen y en que línea del programa se encuentra el dato que causó dicho mensaje.

Los mensajes de advertencia no son causa de falla en el programa, se pueden referir a cosas de menor importancia. Los mensajes de error se pueden referir a una instrucción escrita en forma incorrecta, un carácter no reconocido por el ensamblador, etc.

En el archivo .err se puede verificar en que número de línea del programa fuente está el error. Nuevamente entre al editor de texto y revise el listado del programa prueba-asm. Desplace el cursor hasta dicha línea y corríjalo; luego vuelva a ensamblar.

De no presentarse errores, puede observar el archivo .lst y el archivo .hex. Con este último, se va a programar el microcontrolador.

2.4.5.- Como Programar el Microcontrolador PIC.

Este proyecto de tesis está basado en el programador de microcontroladores PIC figura 2.19.

Programador.

Este es un aparato que puede programar los PICs de 18 pines de la segunda generación; estos incluyen los microcontroladores PIC 16C6X, 16C7X, el 16C8X y los microcontroladores PIC12C50X de 8 pines. Este programador se conecta al puerto paralelo de cualquier computador PC o compatible.

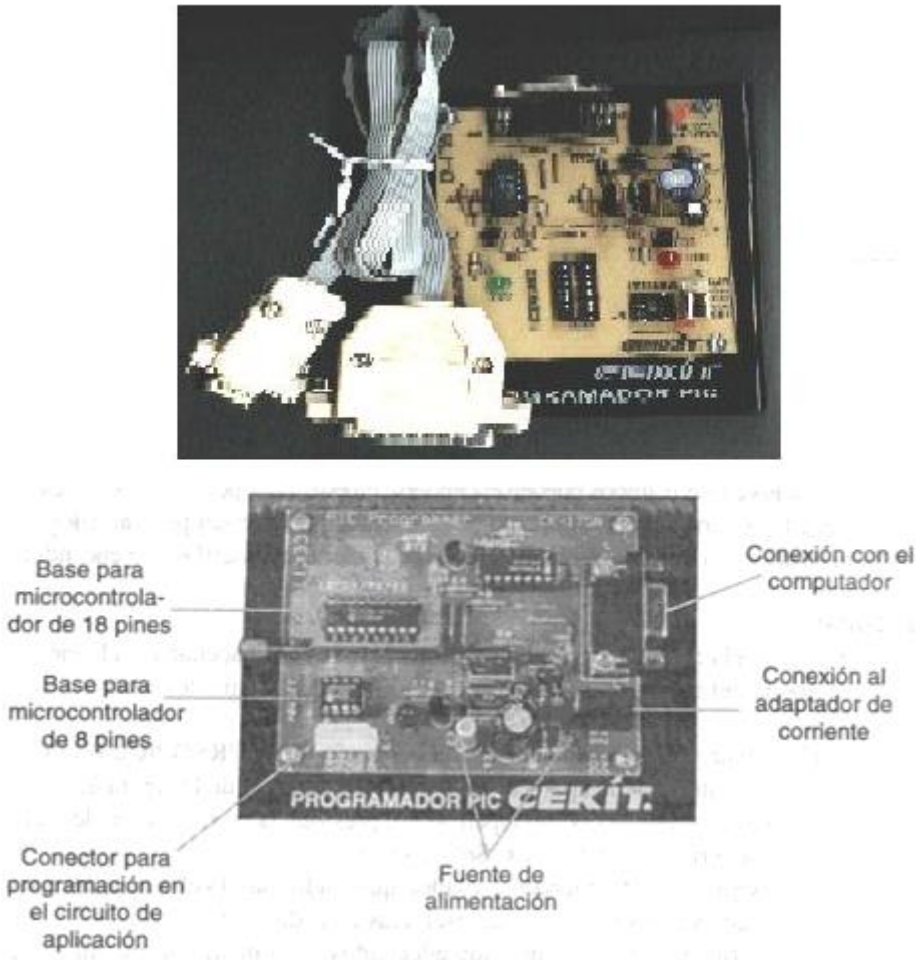


Figura 2.19. Programador de microcontroladores PIC

Cómo guardar el programa en la memoria del microcontrolador.

Asegúrese que el socket del programador esté vacío, de tal manera que el microcontrolador no se encuentre en éste hasta que el software del programador sea ejecutado.

Conecte el programador al puerto paralelo de un computador PC, utilizando el cable suministrado, el cual tiene un conector macho tipo «D» de 25 pines que irá al computador y un conector macho tipo «D» de 9 pines, que irá al programador. En la figura 2.20 se muestra el diagrama de conexiones de este cable.

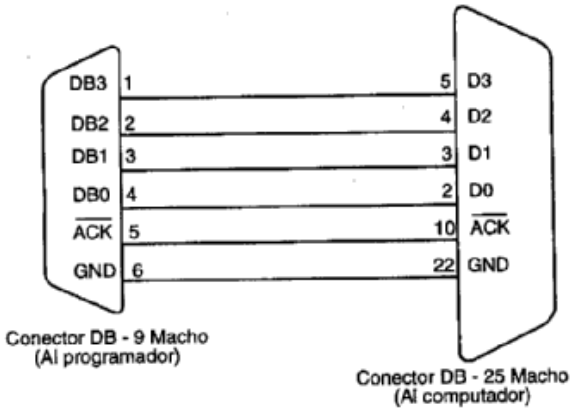


Figura 2.20. Conexión entre el puerto paralelo del computador y el programador.

Conecte el programador al adaptador de pared (este deberá tener un voltaje entre 15 y 20 voltios si es DC, y un voltaje entre 12 y 15 voltios si es AC), y conecte éste a la red pública de AC (110V/60Hz ó 220V/50Hz, dependiendo del país).

En este momento, el programador estará preparado para programar los microcontroladores IC descritos.

Nota: Uno de los LEDs o ambos pueden encenderse en este paso, pero no debe ser causa de alarma; ellos se apagarán al momento de ejecutar el software del programador. No inserte ni remueva un microcontrolador cuando cualquiera de los LEDs esté encendido.

Operación.

Una vez se tiene el programa con extensión .hex listo para almacenarse en la memoria de programa del microcontrolador, se deben seguir los siguientes pasos:

Ejecute el software del programador con el comando «C:\CURSOPIC\>PROG» , luego oprima «Enter»; debe aparecer un pantallazo como el de la figura 2.21.
Con el mouse seleccione la opción Open, si no, presione la combinación de teclas <ALT> O para abrir su archivo .hex. Seleccione el archivo.

Luego debe elegir el tipo de microcontrolador apropiado usando el mouse del computador o las teclas que se encuentran resaltadas en cada opción.

Seleccione los fusibles de programación adecuados de la misma forma que en el punto anterior.

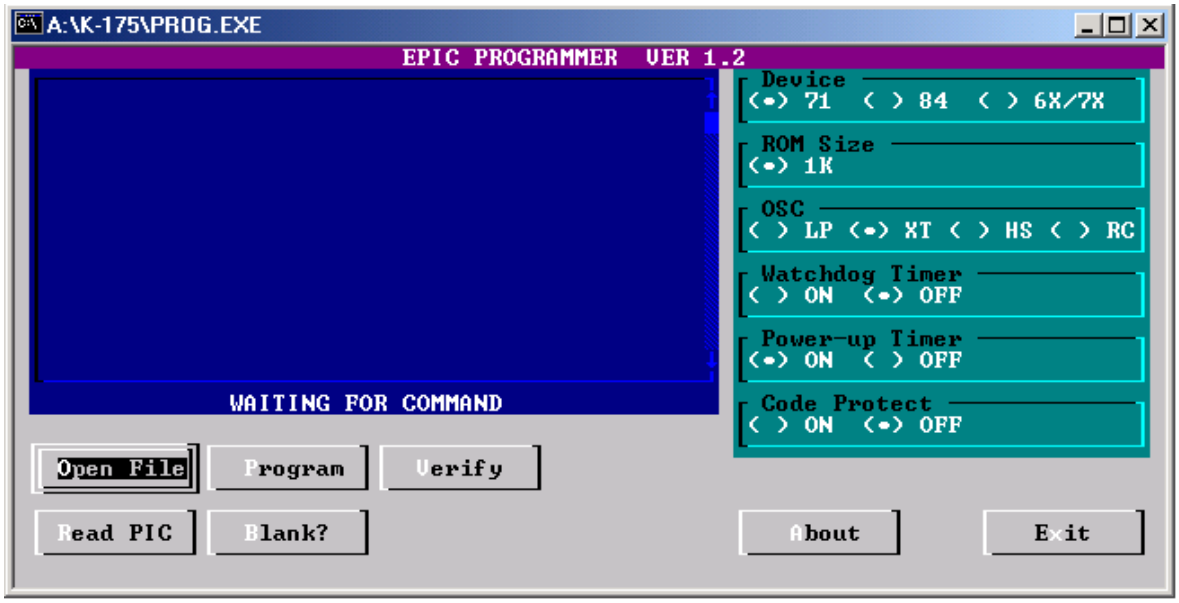


Figura 2.21. Pantallazo del software programador de PIC

Inserte un microcontrolador PIC16F84 en el socket del programador, cuidando que la posición es la correcta (el pin 1 debe coincidir con el punto), y con el mouse seleccione la opción Program, si no, presione las teclas <ALT> P para programarlo.

En el caso de aparecer un mensaje que dice que el micro no está borrado, escoja la opción de seguir adelante con el proceso de grabación. Esta es una ventaja del PIC 16F84: no es necesario borrarlo antes de programarlo.

Luego de programado, retire el chip con cuidado y póngalo en su circuito, el cual debe estar desconectado de la fuente de alimentación.

2.5.- Instrucciones del microcontrolador PIC.

Instrucciones del PIC16F84.

El PIC16F84 responde a una serie de instrucciones o códigos que se deben grabar en su memoria de programa. En total son 35; también estamos usando dos adicionales, que según Microchip son obsoletas, pero las usamos por presentar mayor facilidad en el aprendizaje.

Tabla 2.11. Conjunto de instrucciones del PIC16F84.

Operaciones orientadas a Registros				
Nemotécnico	Operación	Código de Operación	Estados afectados	
		msb lsb		
ADDWF f, d	Sumar W y f	00 0111dfff ffff	C,DC,Z	
ANDWP f,d	AND entre W y f	00 0101dfff ffff	Z	
CLRF f	Limpiar f	00 00011fff ffff		Z
CLRW	Limpiar w	00 00010XXX XXXX		z
COMF f, d	Complementar f	00 1001dfff ffff	Z	
DECF f,d	Decrementar f	00 0011dfff ffff	Z	
DECFSZ f, d	Decrementar f, saltar si cero	00 1011dfff ffff		
INCF f, d	Incrementar f	00 1010dfff ffff	Z	
INCFSZ f, d	Incrementar f, saltar si cero	00 1111dfff ffff		
IORWF f, d	OR entre W y f	00 0111dfff ffff	Z	
MOVF f, d	Mover f	00 1000dfff ffff		Z
MOVWF f	Mover W a f	00 00001fff ffff		
NOP	No operación	00 0000OXX0 0000		
RLF f, d	Rotar a la izquierda a través del carry	00 1101dfff ffff		C
RRF f.d	Rotar a la derecha a través del carry	00 1100dfff ffff		C
SUBWF f, d	Restar W de f	00 0010dfff ffff	C,DC,2	

SWAPF f, d	Intercambiar nibbles de f	00 1110dfff ffff	
XORWP f, d	OR exclusiva entre W y f	00 0110dfff ffff	Z
Operaciones orientadas a bits			
BTFSC f, b	Probar bit b de f, saltar si es cero	01 10bbbfff ffff	
BTFSS f, b	Probar bit b de f, saltar si es uno	01 11bbbfff ffff	
Operaciones literales y de control			
ADDLW k	Sumar literal k a W	11 111xkkkk kkkk	C,DC,Z
ANDLW k	AND entre k y W	11 1001kkkk kkkk	Z
CALL k	Llamar subrutina	10 0kkkkkkk kkkk	
CLRWDT	Limpiar WDT	00 00000110 0100	TO,PD
GOTO k	Salta a dirección k	10 1kkkkhkk kkkk	
IORLW k	OR entre k: y W	11 1000kkkk kkkk	Z
MOVLW k	Cargar a W con literal k	11 00xxkkkk kkkk	
RETFIE	Retornar de interrupción	00 00000000 1001	
RETLW k	Retornar y cargar a W con k	11 01xxkkkk kkkk	
RETURN	Retornar de subrutina	00 00000000 1000	
SLEEP	Ir al modo de bajo consumo	00 00000110 0011	TO,PD
SUBLM k	Restarle k a W	11 110xkkkk kkkk	C,DC,2
XORLW k	OR exclusiva entre k y w	11 1010kkkk kkkk	Z

Las instrucciones TRIS y OPTION no aparecen en el listado, pero se pueden usar para realizar algunas funciones de manera más sencilla, al menos durante el nivel básico. TRIS

sirve para configurar los puertos, es decir, para decir si un pin trabaja como entrada o como salida. OPTION sirve para programar algunas funciones especiales.

2.5.1.- Estructura de Instrucciones.

Las instrucciones del PIC 16F84 se dividen en tres grupos diferentes: Orientadas a los registros, orientadas a los bits y de control. Todas se ejecutan en un ciclo de instrucción, es decir, en un microsegundo cuando se tiene un cristal de 4 MHz, excepto las instrucciones de salto que ocupan dos ciclos de instrucción. Cada una consta de 14 bits que están divididos en dos partes:

El código de operación y el operando.

bit 13 bit0

Código de operación Operando (s)

El código de operación especifica el tipo de instrucción (sumar, restar, cargar, probar bit, etc.) y siempre es igual. El operando es la parte variable de la instrucción y especifica un dato, una etiqueta o dirección de salto, el número o nombre del registro con el que se efectúa la operación, la posición del bit que se desea chequear o ajustar a un valor o el destino donde se guardará el resultado de la operación.

2.5.2.- Tipos de Instrucciones.

Instrucciones orientadas a los registros (o al byte que contienen):

En ellas la letra "f" representa uno cualquiera de los 48 registros de memoria RAM y la letra "d" representa cual es el destino del resultado de la operación.

Si d = 0 el resultado de la operación se carga en el registro de trabajo W; si d = 1 el resultado se guarda en el registro "f " especificado por la instrucción. Ejemplo:

INCF f,d podría ser **INCF r06,0**

En este caso el contenido del registro 06 se incrementa una vez y ese nuevo valor se guarda en el registro W, el contenido del registro r06 permanece intacto.

Instrucciones orientadas a un bit de los registros:

En ellas la letra "b" representa un bit de el registro "f " que lo contiene. Ejemplo:

BCF f,b podría ser BCF r09,2

En este caso el bit número 2 de el registro r09 se pone en cero; los otros bits permanecen intactos.

Instrucciones de control y manejo de literales:

En ellas la letra "k" representa un dato constante (literal) o una dirección de memoria para un salto (etiqueta). Ejemplo 1:

GOTO k podría ser GOTO 00 ó GOTO reset

En este caso se le dice al contador de programa del microcontrolador que va a la dirección de memoria 00. Ejemplo 2:

MOVLW k podría ser MOVLW 05

En este caso el registro de trabajo W se carga con el número 05.

Otras definiciones.

Nemotécnico es el nombre que se le da a la instrucción del microcontrolador que está escrita de manera entendible por la persona que hace el programa (ej. MOVLW).

La letra "x" representa una condición que no importa.

Las letras: "Z, C, DC" representan los bits del registro de estados (banderas).

Tabla 2.12. tabla de equivalencias en los sistemas de numeración

Decimal	Hexadecimal	Binario
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Esta tabla nos sirve para comprender mejor las operaciones o estados que se obtienen en cada una de las instrucciones, ya que cada una de ellas va acompañada por un ejemplo, en el cual uno o varios registros se ven afectados y cambian de valor. Así, el lector puede convertir el número obtenido luego de la operación, en su equivalente binario, para observar bit a bit, el tratamiento que cada instrucción hace a los registros o valores constantes que en ella intervienen.

2.5.3.- Descripción de las Instrucciones.

Para facilitar el estudio de las instrucciones y hacer claridad en su operación, existe una descripción detallada de cada una, explicando la forma de escribirla en el programa y la forma en que esta afecta los registros o bits involucrados en ella, Además, se incluyen ejemplos que muestran en forma clara los resultados que se obtienen con el uso de cada una especificada en el Anexo A.

2.5.4.- Programación Básica.

Rutinas de programación.

Las rutinas son uno de los recursos más valiosos cuando se trabaja en programación; ellas permiten que los programas sean más simples, debido a que el programa principal se disminuye cuando algunas tareas se escriben en forma de programas pequeños e independientes.

En programación, una rutina se define como un procedimiento independiente (un bloque), que realiza una labor específica y a la cual se puede llamar desde cualquier parte del programa principal. Dado que cada rutina realiza una labor en particular, como encender un LED o establecer una base de tiempo de un segundo, por ejemplo, el programador, cuando está seguro de su funcionamiento, puede mantenerla almacenada y disponible en un «banco» o «librería», para utilizarla en cualquier programa, sin volver a escribir las líneas que realicen lo deseado: sólo necesitará copiar el bloque de la rutina o tener el archivo disponible para cuando se realice la compilación del programa; posteriormente, deberá hacer el llamado al procedimiento adecuado, en el instante preciso.

Para la elaboración de cualquier rutina, es necesario conocer más a fondo las instrucciones que la forman y cuales son las condiciones o estados que se modifican ante cada una de ellas. Los estados, banderas o flags son bits que se afectan por algunas operaciones lógicas o aritméticas y la consulta de estos se utiliza para tomar decisiones dentro del programa. Debemos tener presente que en el microcontrolador PIC 16F84, los estados se encuentran localizados en el registro 03, llamado STATUS. Los primeros tres bits de este registro (O al 2) son los estados sobre los cuales debemos ejercer una vigilancia, para determinar así los siguientes pasos a ejecutar.

El bit menos significativo (bit 0) es el correspondiente al estado del Carry o acarreo, el siguiente (bit 1) corresponde al DC o acarreo de dígito y el tercero de ellos (bit 2) es el correspondiente al estado Z o cero.

Cuando escribimos el programa en un editor de textos, para posteriormente utilizar el ensamblador , debemos recordar que, en el formato de las instrucciones primero se escribe el registro fuente y a continuación el registro destino de la operación que se está realizando. Se llama registro fuente aquel registro que se utiliza como origen de la información y registro destino aquel en el cual se almacena el resultado de la operación.

Si deseamos que el destino sea el registro W, podemos optar por escribir en la parte correspondiente, W ó 0; si deseamos que se haga en el mismo registro fuente, podemos escribir 1 u omitirlo. Por ejemplo, las siguientes instrucciones son equivalentes:

comf 10,w equivale a comf 10,0

xorwf 15 equivale a xorwf 15,1

La suma.

Esta operación se implementa en los microcontroladores PIC por una sola instrucción: ADDWF; en la Tabla 2.13 se observan sus detalles. Lo más importante para considerar en todos los casos son los estados afectados ya que, como lo habíamos mencionado, ellos son los indicadores que se deben utilizar para tomar decisiones en el programa.

Tabla 2.13. Detalles de la suma en los microcontroladores PIC

Instrucción:	ADICIÓN
Sintaxis:	ADDWF f,d
	Coa. Op. Dest. Fuente
Codificación:	0001 n d fffff
Operación:	(f)+ (W) -d
Estados afectados:	C, DC, Z
Descripción:	El contenido del registro W se agrega
	al contenido del registro f. Si 'd' es O
	el resultado se almacena en el registro W
	;si 'd' es 1, se hará en el registro f.

Uno de los aspectos que conviene recalcar en esta instrucción, es que el estado del carry no es considerado para los resultados de la operación; se afecta, pero no interviene. Consideremos los ejemplos que se muestran en la Tabla 2.14, y observemos la forma en la cual se afectan los estados y el registro destino en cada una de las instrucciones.

Tabla 2.14. Ejemplos de la suma

Antes de la instrucción			Instrucción	Después de la instrucción				
Fuente		W		Fuente	W	Z	DC	C
1	1001 0011	0101 1000	ADDWF Fuente .W	1001 0011	111010110	0	0	0
2	10101011	00100101	ADDWF Fuente	0110 1101	00100101	0	1	0
3	10101011	01101100	ADDWF Fuente .O	1011 1001	0001 01110	1	1	1
4	11001101	0011 0011	ADDWF Fuente .1	0000 0000	0011 00111	1	1	1

En estos ejemplos, las cantidades se expresan en binario, para visualizar más fácilmente las operaciones bit a bit. Los resultados, ubicados en el destino de la operación, se han resaltado para comprender mejor la forma en que las instrucciones manejan este parámetro.

En las operaciones el carry (C) se coloca a 1 cuando la suma supera el valor más alto que puede contener un byte (OFF hex o 225 decimal); en caso contrario permanece en O, indicando que no se presentó sobreflujo o acarreo.

El acarreo de dígito (DC) se activa cuando la suma de los nibbles (partes inferiores o 4 bits menos significativos de los bytes) supera el valor que este puede contener (OF hex o 15 decimal). El estado Z (Zero o cero) se activa cuando la operación da como resultado O en el registro destino, como se muestra en el ejemplo 4.

La resta.

De manera similar a la suma, esta operación se implementa con una sola instrucción: SUBWF; en la Tabla 2.15 se observan sus detalles más importantes.

Tabla 2.15. Detalles de la resta en los microcontroladores PIC

Instrucción: RESTA
Sintaxis: SUBWF f,d
Cód. Op. Dest. Fuente
Codificación: 000010 d fffff
Operación: (f) - (W) - d
Estados afectados: C, DC, Z
Descripción: El contenido del registro W se resta
del contenido del registro f. Si 'd' es O
el resultado se almacena en el registro W;
si “d' es 1, se hará en el registro f.

Consideremos los ejemplos de la tabla 2.16 y observemos la forma en la cual se afectan los estados y el registro destino ante cada una de las instrucciones. De nuevo, las cantidades están expresadas en binario y los resultados, ubicados en el destino de la operación, se han resaltado para visualizar cómo las instrucciones manejan este parámetro..

Tabla 2.16. Ejemplos de la resta

Antes de la instrucción			Instrucción	Después de la instrucción				
Fuente		W		Fuente	W	Z	DC	C
1	1011 1001	0100 0010	SUBWF Fuente, W	1011 1001	01110111	0	1	1
2	1011 1001	01001100	SUBWF Fuente	0110 1101	01001100	0	0	1
3	1011 1001	11100011	SUBWF Fuente, “	1011 1001	11010110	0	1	0
4	1011 1001	1011 1001	SUBWF Fuente, 1	0000 0000	10111001	1	1	1

En las sustracciones, el carry o acarreo (C) se coloca en 1 cuando el valor absoluto del registro de trabajo W (sustraendo) es menor que el valor absoluto del registro fuente (minuendo); por lo tanto, es un indicador que el resultado es un número positivo. Si por el contrario, el carry resultante es O, indica que el resultado es un número negativo, ya que el minuendo era menor que el sustraendo.

El acarreo de dígito (DC), mientras tanto, se coloca en 1 cuando el valor absoluto del nibble menos significativo del registro de trabajo W es menor que el nibble correspondiente del registro fuente y se pondrá en O en caso contrario. Este estado es muy útil en rutinas de conversión de números binarios a números BCD, como se verá más adelante.

Por último, el estado Z sólo se activará cuando ambas cantidades sean iguales y el resultado de la operación, por lo tanto, sea cero, siendo éste el valor que se almacena en el registro destino.

La rotación.

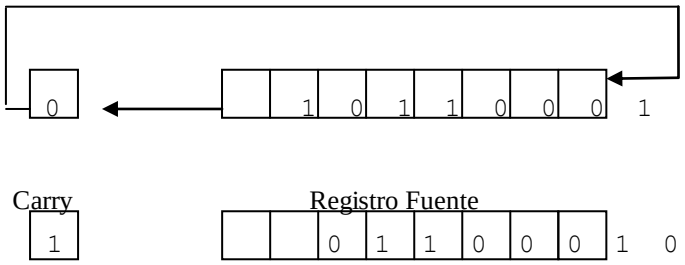
En los microcontroladores PIC se encuentran dos instrucciones de rotación: una hacia la izquierda y otra hacia la derecha; ambas incluyen el carry dentro de la operación. Ya que la diferencia entre las dos rotaciones está sólo en el sentido en que se realiza, sólo mostraremos los detalles de una de ellas, Tabla 2.17.

Tabla 2.17. Rotación a la izquierda

Instrucción:	ROTACIÓN A LA IZQUIERDA
Sintaxis:	RLF f,d
	Cód. Op. Dest. Fuente
Codificación:	001101 d ffffi
Operación:	f<n>-d<n+1>; f<7>-C, C-d<0>;
Estados afectados:	C
Descripción:	El contenido del registro 'f se rota un bit hacia la izquierda a través del carry. Si 'd' es O, el Resultado Se almacena en el registro W; si 'd' es 1,se hará en el registro f.

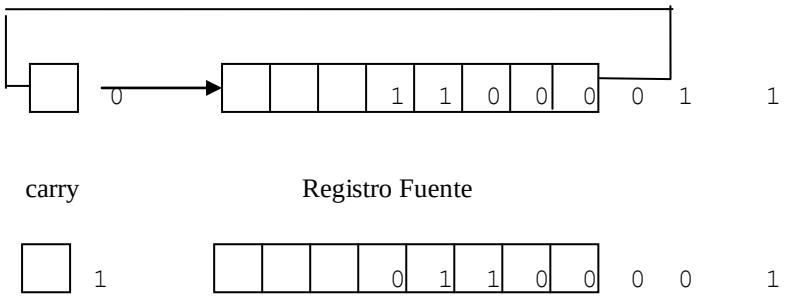
INSTRUCCIÓN RLF

INSTRUCCIÓN RLF



Registro destino después de la rotación

INSTRUCCIÓN RRF



Registro destino después da la rotación

2.6.- Estudio de Componentes

2.6.1.- Teclado Matricial.

Este teclado tiene una matriz de 4 x 4, con los caracteres desde 0 hasta F (hexadecimal). En el conector del mismo se encuentran marcadas las filas como F0, F4, F8 y FC. Las columnas se denominan C0, C1, C2, C3.La numeración de las filas y de las columnas corresponden al primer carácter, ya sea de la fila o de la columna.

Las columnas del teclado tienen conectados unas resistencias de pull-up. Estas sirven para fijar un nivel alto cuando no se este oprimiendo ninguna tecla. Como ejercicio, es conveniente que el estudiante identifique por su propia cuenta los pines del teclado, utilizando un multìmetro, esto con el fin de familiarizarse con sus conexiones.

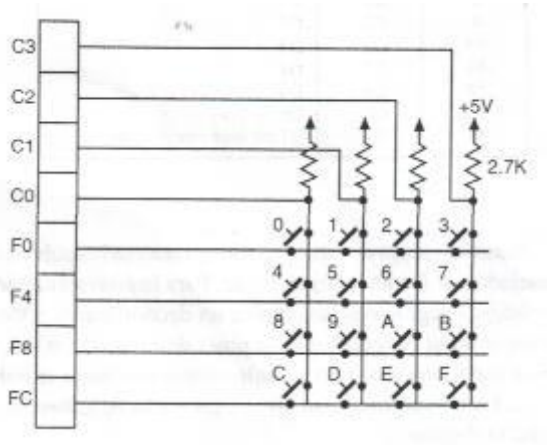


Figura 2.22 teclado matricial.

2.6.2.- Display de 7 Segmentos de 4 Dígitos.

El sistema dispone de 4 displays de cátodo común , conectados sobre el mismo bus de datos y con los cátodos manejados de forma independiente. Para la conversión de datos que entrega el microcontrolador al código 7 segmentos, se emplea un decodificador 9368.

Todos los segmentos del display se encuentran conectados a los pines de salida del 9368, por lo tanto el digito que se desea mostrar este momento, se debe seleccionar mediante la habilitación del display correspondiente. Esto se hace conectando un nivel lógico alto en base del transistor NPN que maneja el cátodo de dicho display. Los 4 dígitos que se va ha mostrar, se debe conectar desde el microcontrolador a los pines marcados D, C, B y A del integrado 9368. EL de mayor peso es el D, y el de menor peso es el A, los pines marcados D1 a D4 corresponden a la base de los transistores que habilitan los displays.

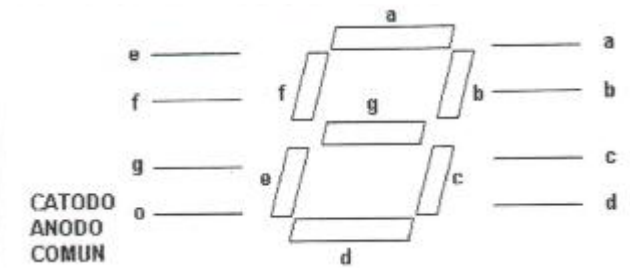


Figura 2.23 Display de Cátodo / Ánodo Común

2.6.3.- Display Alfanumérico.

Módulo de cristal líquido LCD.

Este elemento permite darle mucha funcionalidad y elegancia a los proyectos, además es muy fácil de manejar con un microcontrolador. En este circuito se tienen todas las conexiones necesarias para su funcionamiento, incluyendo un potenciómetro para controlar el brillo de la pantalla.

Si se utiliza un módulo LCD que tiene todos los pines en línea, se puede conectar directamente al circuito impreso sin necesidad de hacer puentes. Si por el contrario, el módulo tiene los pines en un arreglo de dos filas de 7 pines cada uno, se deben conectar los puentes de alambre que están marcados en el circuito impreso. Normalmente, esta última clase tiene además luz posterior (backlight), se debe entonces conectar los dos pines del extremo derecho del módulo y la resistencia de 10 ohm (1/2W) que se encuentra junto al potenciómetro de ajuste del brillo.

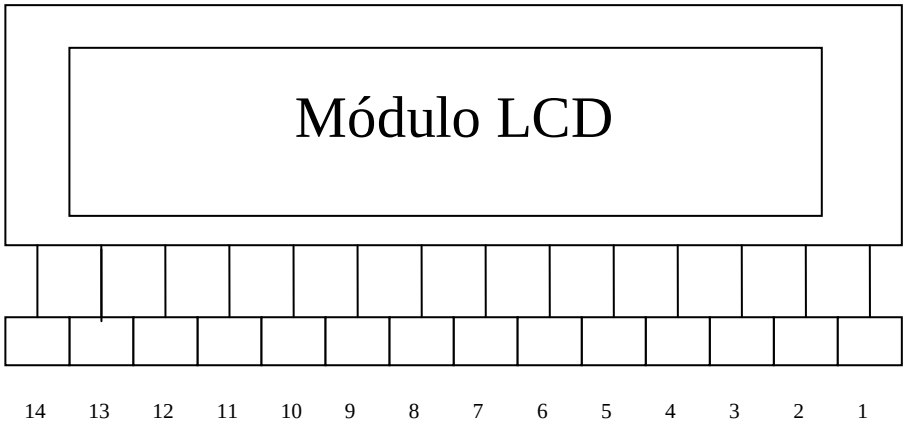


Figura 2.24. Módulo de Cristal Líquido

Tabla 2.18. Características de los pines del módulo LCD

Terminal	Símbolo	Nombre y Función
1	VSS	Tierra, OV
2	Vdd	Alimentación, +5V
3	Vo	Ajuste de voltaje de contraste
4	RS	Selección Control/Dato
5	R/E	Lectura/ Escritura en LCD
6	E	Habilitación
7	DO	DO bit menos significativo
8	D1	D1
9	D2	D2
10	D3	D3
11	D4	D4
12	D5	D5
13	D6	D6
14	D7	D7 bit más significativo

2.6.4.- La comunicación serial RS232.

Es una de las más utilizadas para implantarla solo se requiere un circuito integrado MAX 232 y 5 condensadores de 10 micro faradios (TANTALIO), los pines de comunicación se denominan Tx y Rx

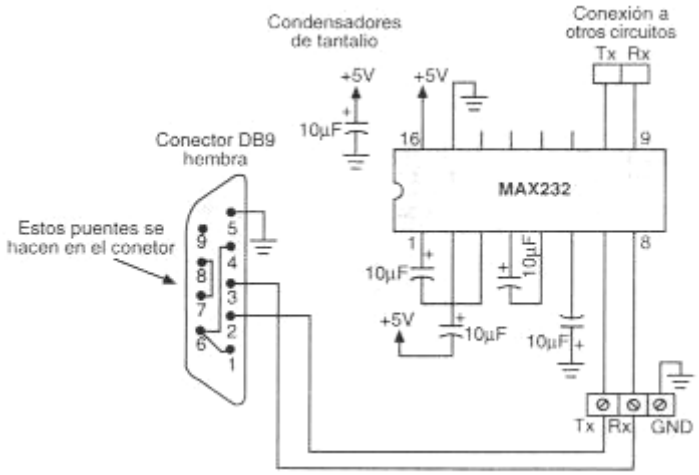


Figura 2.25. Comunicación Serial RS232

2.6.5.- Potenciómetro Digital.

Este circuito integrado contiene un potenciómetro de estado sólido que se comporta de manera similar a los potenciómetros electromecánicos. Los dos extremos de la resistencia variables esta conectados uno al positivo a la fuente de alimentación y el otro a la tierra , de esta forma el cursor puede variar entre 0 y 5 voltios. El control del valor de la resistencia se hace utilizando 3 pines : cs, u/d inc . este sistema es ideal para hacer conversiones de D/A.

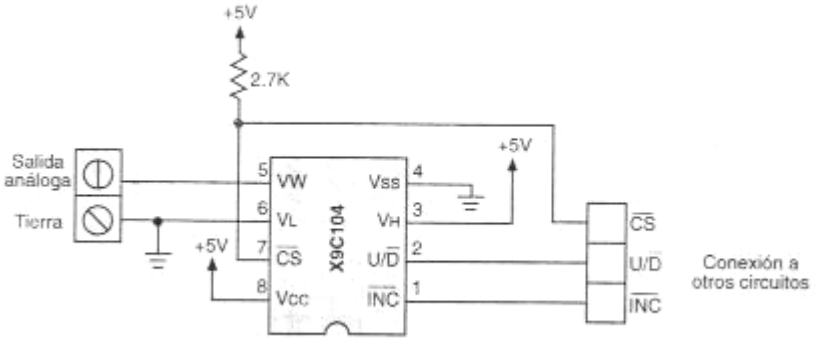


Figura 2.26. Potenciómetro Digital

2.6.6.- Memoria EEPROM Serial.

Memoria serial 24LCOX.

La conexión de esta memoria con el microcontrolador se hace utilizando la interface serial I^2C. El selector de dos posiciones sirve para manejar el pin WP wríte protect) de la memoria, el cual, si está a un nivel lógico alto (+5V), sólo permite leer en la memoria (read) y si está en un nivel lógico bajo (OV), permite leer y escribir (read/ write).

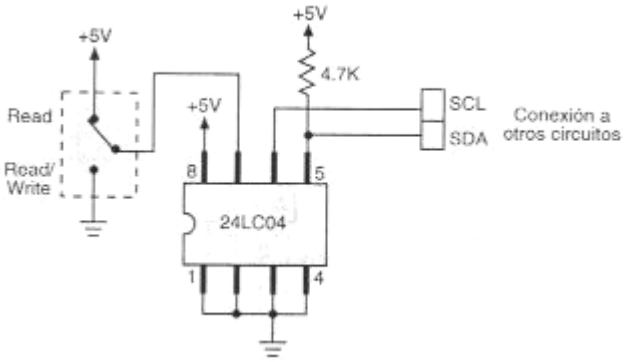


Figura2.27. Memoria Serial 24LCOX

Entradas análogas.

Una de las características más importantes del PIC16C71 (y otros de Microchip), es el convertidor análogo a digital. Para comprobar su funcionamiento, el entrenador incluye dos fuentes de señal análoga, la primera es un potenciómetro conectado entre +5V y tierra, cuyo cursor se puede conectar directamente al microcontrolador (desde el punto marcado Al). Otra de las posibles aplicaciones de este potenciómetro es servir como voltaje de referencia para el convertidor análogo a digital, mientras que la otra entrada análoga se utiliza como fuente de señal.

La segunda fuente de voltaje análogo es un sensor de temperatura LM35, el cual presenta una variación de 10 mV/°C. El sensor se ha conectado en forma de sonda para que sea posible medir temperaturas justo sobre la fuente de calor. La conexión al circuito principal se hace mediante un conector de tres pines mostrado en la Figura 2.28. también se puede ver la forma en que se conectan los cables.

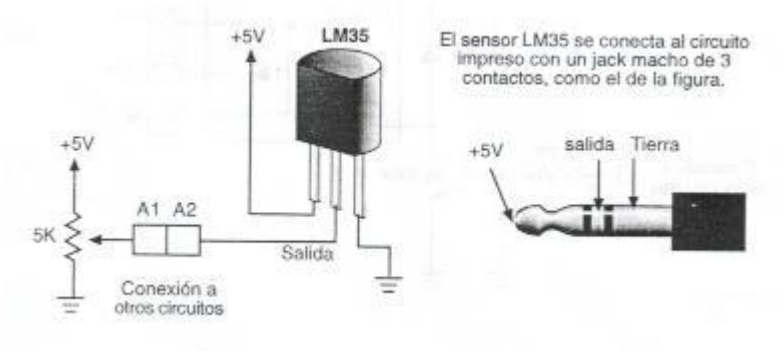


Figura2.28. Entradas Análogas

2.6.7.- Sonda de Temperatura.

Para medir la temperatura emplearemos un LM35. Este sensor tiene tres pines : Alimentación, tierra y la salida análoga. Este dispositivo presenta en su salida una variación de 10mV/ºC, por lo tanto, el valor de la temperatura se puede obtener directamente, sin necesidad de hacer modificaciones al dato obtenido.

CAPITULO III

DESARROLLO

3.1.- Estudio de Alternativas.

3.1.1 Planteamiento de la Alternativa.

- Familia Intel 8051
- Familia Motorola 6805
- Familia Microchip 16CXX
- Familia PIC16F84 ò Familia Microchip 16F8**4**

3.1.2 Análisis y Evaluación de Alternativas.

- **Familia Intel 8051** son microcontroladores de 8 bits, tienen un mismo CPU, memoria RAM , Temporizadores, puertos paralelos y entradas / salidas de tipo serial, 4kbytes de memoria ROM. Se programa durante el proceso de fabricación del microcontrolador.

- **Familia Motorola 6805** en lugar de procesar datos forma parte de impresoras, módems, y electrodomésticos. tienen un mismo CPU de 8bytes , memorias RAM, ROM ,puertos de entrada / salida, temporizadores, puertos seriales convertidores analógico / digital, memorias EPROM Ò EEPROM

- **Familia Microchip 16CXX** utiliza solo 35 instrucciones, es un microcontrolador de tipo RISC (Reduced Instruction Set Computer - Computador con Set de Instrucciones Reducido) a diferencia de las anteriores familias que utilizan la

tecnología CISC (Complex Instruction Set Computer - Computador con Set de Instrucciones Complejo).

- **El PIC16F84** es un MÍcrochip de tecnología CMOS, su consumo de potencia es muy bajo y además es completamente estático

El encapsulado más común para el microcontrolador es el DIP (Dual In-line Pin) de 18 pines, propio para usarlo en experimentación. La referencia completa es 16F84-04/P, para el dispositivo que utiliza reloj de 4 MHz.

3.1.3.- Ventajas Desventajas.

Ventajas.

- tiene dos puertos. El puerto A con 5 líneas y el puerto B con 8 líneas
- Cada pin se puede configurar como entrada o como salida independiente programando un par de registros diseñados para tal fin.
- En ese registro un "O" configura el pin del puerto correspondiente como salida y un " 1" lo configura como entrada.
- El puerto B tiene internamente unas resistencias de pull - up conectadas a sus pines.
- El PIC16F84 puede utilizar cuatro tipos de oscilador diferentes.
- El PIC16F84 admite diferentes tipos de reset
- Todos los elementos se conectan entre sí por medio de buses.
- El PIC16F84 tiene un bloque especial de memoria de datos de 64 bytes del tipo EEPROM
- Tiene arquitectura Harvard,

Desventajas.

- No tiene un convertidor análogo digital
- No brinda muchas facilidades para proyectos muy pequeños en comparación con el PIC12C508.
- Tiene tecnología C-MOS.

3.1.4.- Selección de la Alternativa.

- El PIC16F84 es un microcontrolador con la memoria de programa de tipo FLASH, lo que representa gran facilidad en el desarrollo de Prototipos y en su aprendizaje ya que no requiere borrarlo con la luz ultravioleta como las versiones EPROM sino, permite reprogramarlo nuevamente sin ser borrado con anterioridad.
- Presenta bajo consumo de potencia, además es completamente estático, esto quiere decir que el reloj puede detenerse y los datos de la memoria no se pierden.
- Es fácil su manera de funcionamiento debido a que tiene tan solo 35 instrucciones y dos instrucciones más que son obsoletas, pero sirven para entender de una forma mas didáctica su programación.
- .Es compatible con otras familias en espacial el Microchip 16C84.

3.2.- Requerimientos Técnicos

3.2.1.- Teclado Matricial.

Nos proporcionan algunas técnicas que puedan ayudar a optimizar los diseños.

El multiplexaje, que se define como una forma de compartir secuencialmente el tiempo para que dos o más señales se puedan transmitir a la vez por un mismo medio conductor, es sin duda una gran herramienta (y en ocasiones la única) para conseguir un mejor

aprovechamiento de un dispositivo. Nosotros la utilizaremos para la lectura de teclados y la visualización de información a través de displays de siete segmentos.

Manejo de teclados.

Inicialmente consideremos la implementación de un teclado sencillo, el cual consta básicamente de 8 interruptores (dipswitch), tal como se muestra en la figura 3.1, en donde a cada pin del puerto B del microcontrolador corresponde una determinada tecla. Cuando estas teclas no están presionadas, el pin correspondiente estará conectado a un nivel lógico alto, en cambio cuando alguna de ellas se presiona, el pin correspondiente se conectará a un nivel lógico bajo; en este teclado por lo tanto se lee "ceros". Un aspecto que vale la pena tener en cuenta es que si el microcontrolador tiene elementos pull-ups internos, las resistencias que se muestran pueden eliminarse, simplificando el circuito.

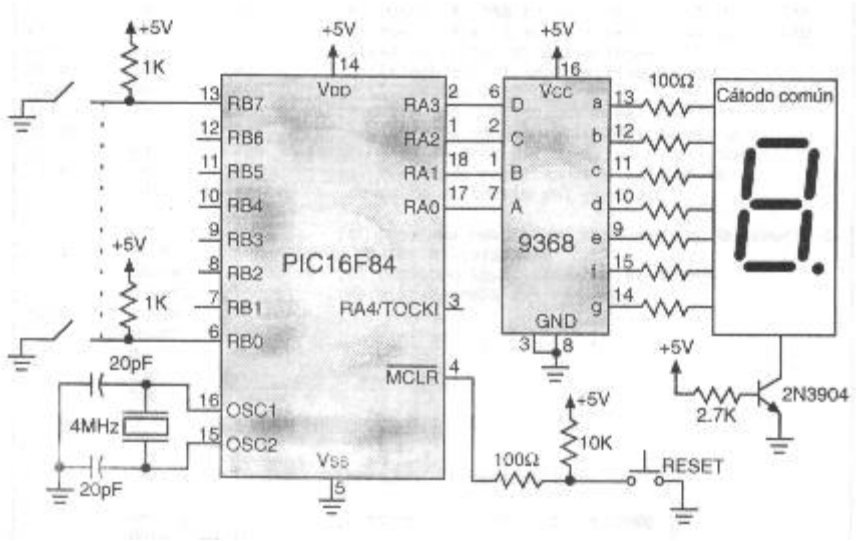


Figura3.1 Lectura de un teclado sencillo.

Cuando las líneas de entrada/salida empiezan a escasear, debemos pensar en rediseñar este teclado, optimizándolo. La figura 3.2 muestra una alternativa para este teclado, observe que para las mismas 8 teclas se tienen sólo 6 líneas de entrada/salida (nos hemos ahorrado 2 líneas, las cuales pueden ser aprovechadas para otros propósitos igualmente importantes); en la figura 3.3 se muestra un teclado de 16 elementos, precisando sólo 8 líneas de entrada/salida (necesitaríamos 8 líneas más si hubiésemos seguido el principio de

diseño inicial).Estos dos últimos teclados tienen algo en común, están organizados matricialmente y para manejarlos se requiere el multiplexaje.

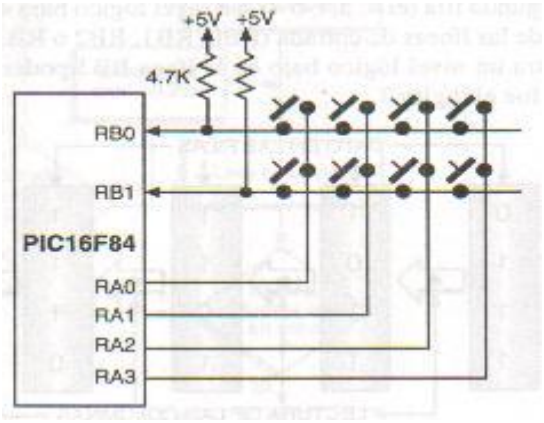


Figura 3.2. Teclado matricial de 2 filas x 4 columnas

Consideremos la figura 3.3, que muestra el teclado matricial de 4 filas por 4 columnas. En este caso, las líneas del microcontrolador correspondientes a las filas se han configurado como salidas y las correspondientes a las columnas como entradas.

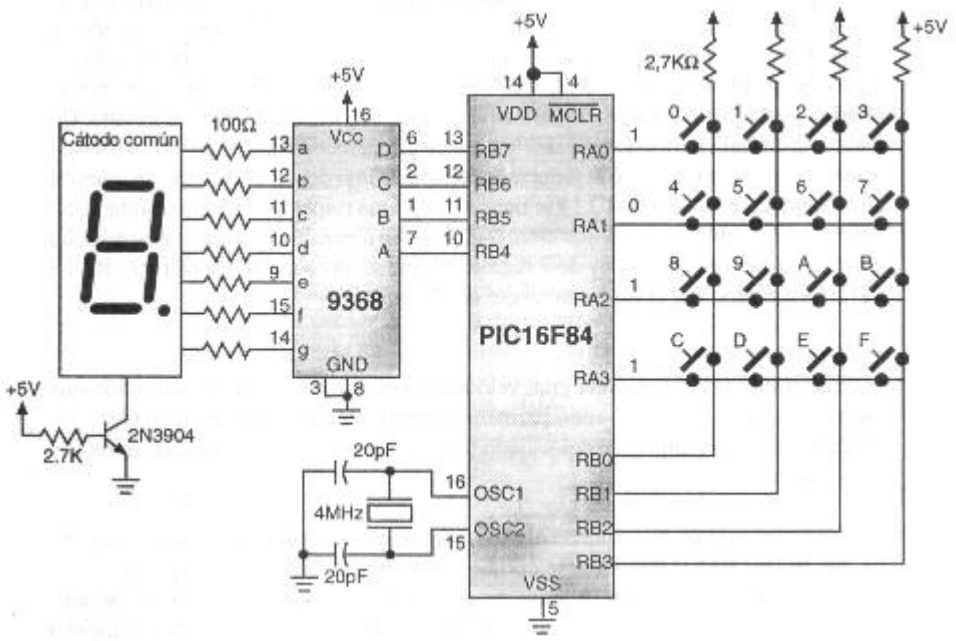


Figura 3.3. Teclado matricial de 4 filas x 4 columnas

Como se puede observar, normalmente, las líneas de entrada permanecen en un nivel lógico alto, gracias a los elementos pull-up (resistencias de 2.7K). La clave para manejar este tipo de teclados consiste en enviar por las líneas de salida sólo un cero por vez; por ejemplo si enviamos un cero por la línea RA1, cuando oprimimos una tecla de la segunda fila (el 4, 5, 6 ó 7), un nivel lógico bajo se reflejará en el pin correspondiente de las líneas de entrada (RBO, RB1, RB2 o RB3 respectivamente); así, si se encuentra un nivel lógico bajo en la línea RB3 podemos concluir que la tecla presionada fue el dígito 7.

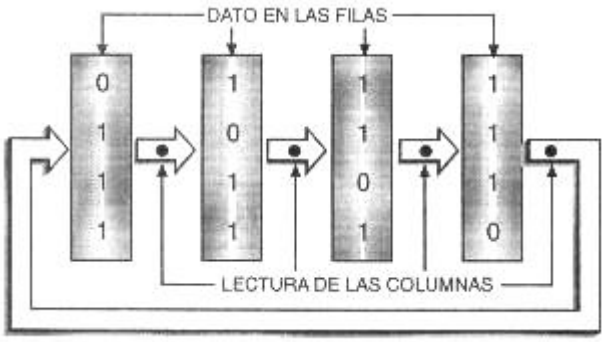


Figura 3.4. Secuencia para la lectura de un teclado matricial

Si queremos explorar todo este teclado, bastará con rotar el cero circularmente, de tal manera que solamente un cero se encuentre en las filas del teclado, cuando se realiza las lecturas de las líneas de entrada (las columnas) como se muestra en la figura 3.4. Cuando el cero llegue a la fila más significativa del teclado, debe reingresar en la Próxima ocasión por la menos significativa, reiniciando la exploración del teclado.

Otro aspecto que no se puede olvidar, son los rebotes causados por la pulsación de una tecla. Cuando una tecla se oprime, sus contactos actúan como resortes, y la unión eléctrica no es estable; se generan una serie de uniones y desuniones mecánicas durante un intervalo significativo de tiempo. Estos rebotes pueden dar lugar a que, en una aplicación real, el programa los interprete como si se hubieran generado muchas pulsaciones, si es que no se

toman los correctivos del caso. Para ello existen soluciones de hardware y software, consideramos más interesantes las segundas ya que simplifican el diseño. Allí, la solución más obvia es que después de la detección de la tecla pulsada se genere un retardo en la lectura del teclado, de tal manera que se ignoren los contactos subsiguientes debidos a los rebotes.

Experimentalmente, se encuentra que un retardo aceptable tiene un valor comprendido entre los 100 a 125 ms; tiempos más pequeños pueden todavía interpretar los rebotes y tiempos más largos pueden tornar demasiado lento un teclado. En ocasiones, conviene también pensar en el tipo de usuarios de un sistema, ya que hay algunos de ellos que tienen la tendencia a mantener oprimida una tecla un tiempo más largo del común de las personas, lo que mal controlado en el programa puede dar lugar a un funcionamiento incorrecto del sistema.

3.2.2.- Display de 7 Segmentos de 4 Dígitos.

Los display de siete segmentos son un elemento muy útil en el diseño de aparatos electrónicos, por ejemplo, cuando se requiere visualizar el dato proveniente de un conteo, de temporización, el estado de una máquina, etc.

Para manejar el display utilizaremos un decodificador 9368, que es el compatible del tradicional 7448, pero decodifica de binario a hexadecimal, es decir que puede mostrar los caracteres de A hasta F. En los ejercicios el microcontrolador debe encargarse de verificarse cuando el conteo llega a 9 para empezar nuevamente en 0.

Utilizaremos un display de catódo común, para aumentar su visualidad se puede conectar a un transitor NPN que le entrega una buena corriente.

Nota: Si se usa el decodificador 7448 en el lugar del 9368,en el pìn 3 se debe dejar al aire.

Para realizar un multiplexaje con 4 display consideremos la estructura de la figura 3.5. Allí se tienen cuatro displays de siete segmentos, con punto decimal; un puerto de 4 bits (o la mitad de uno de 8) controla cuatro transistores NPN, que finalmente alimentan los cátodos de cada uno de los displays, en donde el bit menos significativo controla el display de menor peso. Un puerto de 8 bits maneja cada uno de los segmentos de los displays, en donde el bit menos significativo del puerto controla el segmento a, el siguiente el b, y así sucesivamente, hasta llegar al más significativo, que controla el punto decimal del display.Las resistencias tienen como objeto limitar la corriente que fluye a través de los segmentos y que ingresa a los pines del microcontrolador.

Por la configuración, es fácil deducir que cuando se tiene un nivel lógico bajo en la base del transistor éste se comporta como un interruptor abierto y no se presenta corriente entre emisor y colector; se necesita tener un uno lógico en las bases de éstos para que el transistor se comporte como un interruptor cerrado y se presente dicha corriente. Pero aún cuando los transistores se comporten como interruptores cerrados, no se encenderá ningún display si la salida del puerto que maneja los segmentos tiene niveles lógicos bajos (ceros); para encenderlo, es necesario colocar un nivel lógico alto en el pin de salida correspondiente a cada segmento.

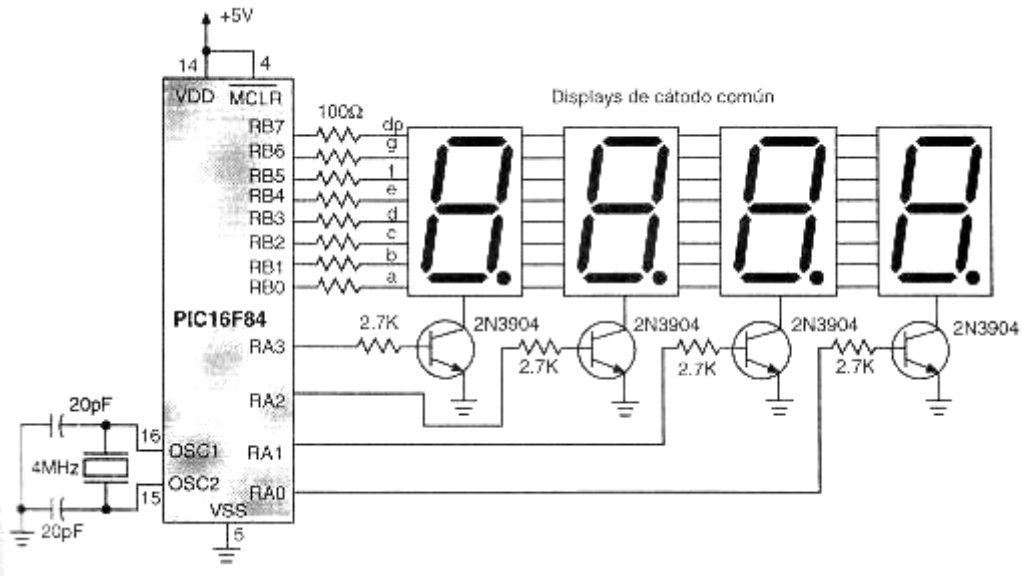


Figura 3.5. Configuración para manejo de displays de 7 segmentos

Así por ejemplo, si queremos mostrar un tres en el segundo display menos significativo debemos establecer básicamente los siguientes pasos:

- Colocar el número binario 0010 en el puerto que controla los transistores
- Enviar el número binario 01001111 por el puerto que controla los segmentos

Un buen observador encontrará que aunque los anteriores pasos consiguen el objetivo propuesto (mostrar en la pantalla el —3-, en donde la línea quiere decir espacios en blanco), este proceso puede conllevar efectos indeseados. Aquí, por ejemplo, si el puerto que controla los segmentos tenía un valor diferente del binario 00000000 o de 01001111, entre la ejecución de los dos pasos dados se mostrará en el display un número o un símbolo que es diferente del valor deseado. Por ello, es conveniente apagar momentáneamente los segmentos, de tal manera que nos permita seleccionar adecuadamente el display en cuestión y posterior a esto, enviar el dato correcto a los segmentos; por lo tanto, un paso O

que se debe agregar a este proceso es colocar el número binario 00000000 en el puerto que controla los segmentos.

Otra opción que permite obtener el mismo resultado, con un proceso diferente, sería la siguiente:

0. Colocar el número binario 0000 en el puerto que controla los transistores
1. Enviar el número binario 01001111 por el puerto que controla los segmentos
2. Colocar el número binario 0010 en el puerto que controla los transistores

Ambos procesos conllevan al mismo objetivo propuesto, eliminando la posibilidad de los efectos indeseados.

3.2.3.- Display Alfanumérico.

Módulo LCD.

Cuando se trabaja en diseño de circuitos electrónicos es frecuente encontrarse con la necesidad de visualizar un mensaje, que tiene que ver con el estado de la máquina a controlar, con instrucciones para el operario, o si es un instrumento de medida, mostrar el valor registrado. En la mayoría de los casos, recurrimos a los displays de siete segmentos, pero estos además de no mostrar caracteres alfanuméricos ni ASCII, tienen un elevado consumo de corriente y son un poco dispendiosos de manejar, cuando se requiere hacer multíplexaje.

Los módulos de cristal líquido o LCD, solucionan estos inconvenientes y presentan algunas ventajas, como un menor consumo de corriente, no hay que preocuparse por hacer multiplexaje, no hay que hacer tablas especiales con los caracteres que se desea mostrar, se pueden conectar fácilmente con microprocesadores o microcontroladores y además, los

proyectos adquieren una óptima presentación y funcionalidad. En principio, vamos a conocer las características más importantes de los módulos, luego se muestra la forma de conectarlos con el microcontrolador y se hacen programas simples para escribir mensajes en la pantalla.

Módulos de cristal líquido o LCD.

Antes de mostrar la forma de conectar estos módulos con el microcontrolador, haremos un pequeño recuento de las principales características que ellos tienen, las cuales nos servirán para entender mejor los programas y los diagramas que se muestran más adelante:

- Los módulos LCD se encuentran en diferentes presentaciones, por ejemplo (2 líneas por 16 caracteres), 2x20, 4x20,4x40, etc. La forma de utilizarlos y sus interfaces son similares, por eso, los conceptos vistos aquí se pueden emplear en cualquiera de ellos. En nuestro caso, trabajaremos con un display de 2x16, ya que es de bajo costo, se consigue fácilmente en el comercio y tiene un tamaño suficiente para la mayoría de las aplicaciones.
- La figura 3.6 muestra dos tipos de configuración de pines que se encuentran comúnmente, aunque cambian su ubicación, estos conservan las mismas funciones.

Algunos módulos LCD tienen luz posterior o "backiight", para mejorar su visualizaciòn

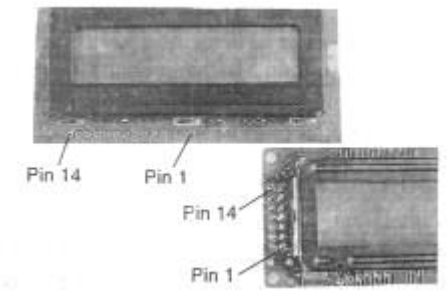


Figura 3.6. Configuración de pines de los módulos LCD

Està se maneja a través de dos pines que normalmente se conectan a +5V y a tierra. Para evitar que se presenten altas temperaturas, debido a la luz posterior, estos pines se deben manejar de manera pulsante (encendiendo y apagando), con una frecuencia de aproximadamente 60 Hz. Otra opción mucho más sencilla es utilizar una resistencia de 10 ohmios (a 1/2W) para alimentar el positivo del backlight.

- Los pines de conexión de estos módulos incluyen un bus de datos de 8 bits, un pin de habilitación (B), un pin de selección, que indica que el dato es una instrucción o un carácter del mensaje (RS) y un pin que índica si se va a escribir o leer en el módulo LCD (R/W). La figura 2.26 describe la función de cada uno de ellos.

Tabla 3.1. Función de los pines del módulo LCD

Terminal	Símbolo	Nombre y Función
1	Vss	Tierra, 0V
2	Vdd	Alimentación +5V
3	Vo	Ajuste de Voltaje de contraste
4	$\overline{RS}$	Selección Dato/Control
5	R/$\overline{W}$	Lectura/escritura en LCD
6	E	Habilitación
7	D0	D0 Bit menos significativo
8	D1	D1
9	D2	D2
10	D3	D3
11	D4	D4
12	D5	D5
13	D6	D6
14	D7	D7 Bit más significativo

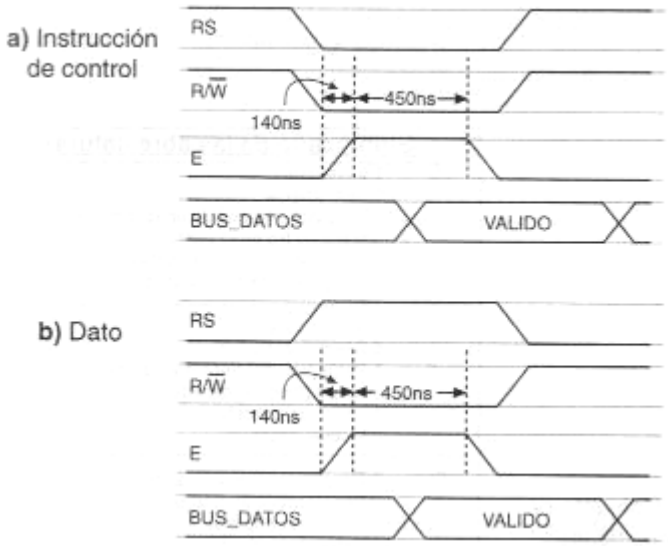


Figura 3.7. Diagrama de tiempo del módulo LCD

- Según la operación que se desee realizar sobre el módulo de cristal líquido, los pines de control E, RS y R/W deben tener un estado determinado. Además, debe tener en el bus de datos un código que indique un carácter para mostrar en la pantalla o una instrucción de control. En la figura 3.7 se muestra el diagrama de tiempos que se debe cumplir para manejar el módulo.
- El módulo LCD responde a un conjunto especial de instrucciones, estas deben ser enviadas por el microcontrolador o sistema de control al display, según la operación que se requiera. Estas instrucciones se emplean en los ejemplos que realizaremos más adelante, en ellos se explica la forma de utilizarlas. Se muestran las instrucciones del módulo.

- La interface entre el microcontrolador y el display de cristal líquido se puede hacer con el bus de datos trabajando a 4 u 8 bits. Las señales de control trabajan de la misma forma en cualquiera de los dos casos, la diferencia se establece en el

momento de iniciar el sistema, ya que existe una instrucción que permite establecer dicha configuración. Estas conexiones se explican más adelante en forma detallada.

Conjunto de instrucciones de los módulos LCD.

1/D = 1 Incrementa

= 0 Decrementa

S = 1 Desplaza el mensaje en la pantalla

= 0 Mensaje fijo en la pantalla

D = 1 Encender (activar) la pantalla

= 0 Apagar la pantalla (desactivar)

C = 1 Activar cursor

= 0 Desactivar cursor

B = 1 Parpadea carácter señalado por el cursor

= 0 No parpadea el carácter

S/C = 1 Desplaza pantalla

= 0 Mueve cursor

RL = 1 Desplazamiento a la derecha

= 0 Desplazamiento a izquierda

DL = 1 Datos de ocho bits

= 0 Datos de cuatro bits

BF = 1 Durante operación interna del módulo

= 0 Finalizada la operación interna

- Los caracteres que se envían al display se almacenan en la memoria RAM del módulo. Existen posiciones de memoria RAM, cuyos datos son visibles en la pantalla y otras que no son visibles, estas últimas se pueden utilizar para guardar

caracteres que luego se desplazan hacia la parte visible. En la figura 3.8. se muestran las direcciones de memoria visibles y no visibles, que conforman las dos líneas de caracteres del módulo.

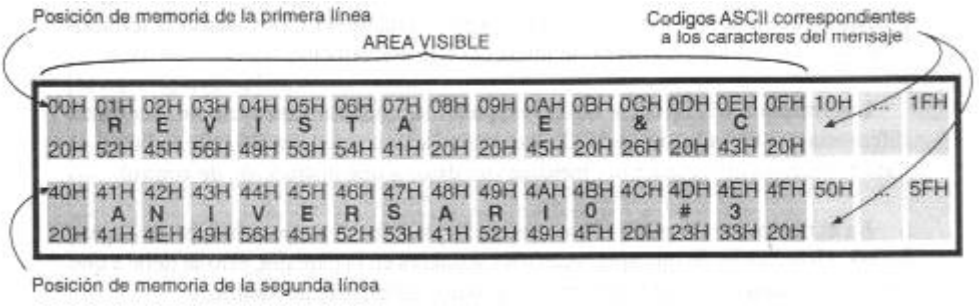


Figura 3.8. Mapa de memoria del módulo LCD

- Es importante anotar que sólo se pueden mostrar caracteres ASCII de 7 bits, por lo tanto algunos caracteres especiales no se pueden ver (se debe tener a la mano una tabla de los caracteres ASCII para conocer los datos que son prohibidos). Por otra parte, se tiene la opción de crear caracteres especiales (creados por el programador), y almacenarlos en la memoria RAM que posee el módulo.

Interface con microcontrolador a 8 bits.

El proyecto consiste en conectar el módulo de cristal líquido a un microcontrolador PIC16F84, utilizando el bus de datos a 8 bits. En este caso se emplea el PIC16F84, aunque se puede implementar con otros microcontroladores que sean compatibles. En la figura 3.9. se muestra el diagrama de conexiones para este caso.

Para estos ejercicios en particular, sólo nos interesa escribir datos en la pantalla (no hacer lectura); por lo tanto el pin de selección de lectura/escritura (R/W) en el display, se conecta a tierra. El puerto B del microcontrolador se utiliza como bus de datos, y el puerto A se encarga de generar las señales de control.

En el oscilador del PIC16F84 se emplea un cristal de 4 MHz, por lo tanto tenemos ciclos de instrucción de un microsegundo. Para el módulo LCD, se emplea un potenciómetro de 5Kohm, conectado entre +5V y tierra, para controlar el contraste de la pantalla.

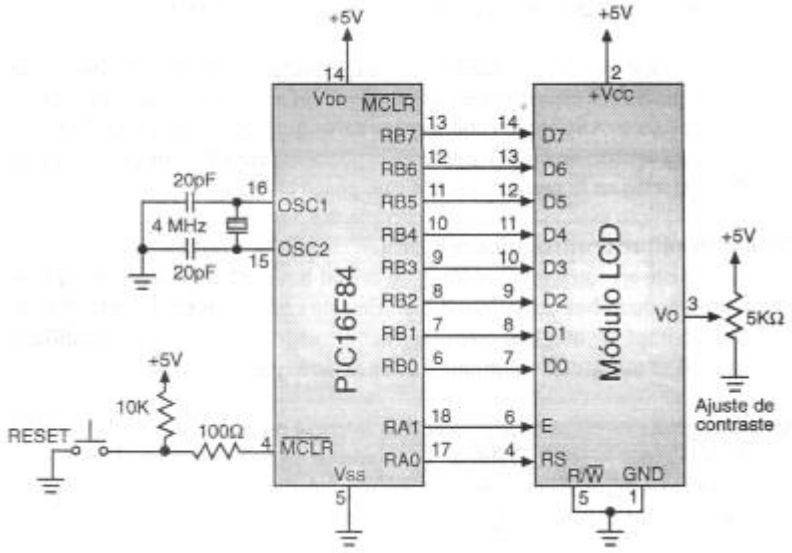


Figura 3.9 diagrama esquemático de la conexión de 8 bits entre el microcontrolador y el módulo LCD

3.2.4.- Comunicación Rs232.

Interface Serial RS-232.

El puerto serial de las computadoras, conocido también como puerto RS-232, es muy útil ya que permite la comunicación no sólo con otras computadoras, sino también con otros dispositivos tales como el mouse, impresoras y por supuesto, microcontroladores.

Existen dos formas de intercambiar información binaria: la paralela y la serial. La comunicación paralela transmite todos los bits de un dato de manera simultánea y tiene la ventaja que la transferencia es rápida, pero la desventaja de necesitar una gran cantidad de hilos o líneas, situación que encarece los costos y se agrava cuando las distancias que

separan los equipos entre los cuales se hace el intercambio es muy grande, debido a las capacitancias entre los conductores, la cual limita el correcto intercambio de datos a unos pocos metros.

La comunicación serial por su parte, transmite un bit a la vez, por lo cual es mucho más lenta, pero posee la ventaja de necesitar un menor número de líneas para la transferencia de la información y las distancias a las cuales se puede realizar el intercambio es mayor; a esto se suma que mediante dispositivos como los modem, la comunicación se pueda extender prácticamente a cualquier lugar del planeta.

Existen dos formas de comunicación serial: la sincrónica y la asincrónica. En la comunicación sincrónica, además de una línea sobre la que transfieren los datos, se necesita otra que contenga pulsos de reloj que indiquen cuando un dato es válido; la duración del bit está determinada por la duración del pulso de sincronismo. En la comunicación asincrónica, los pulsos de reloj no son necesarios y se acude a otros mecanismos para realizar la lectura/escritura de los datos; la duración de cada bit está determinada por la velocidad con la cual se realiza la transferencia de datos. En esta práctica sólo trataremos la comunicación asincrónica o asincrona.

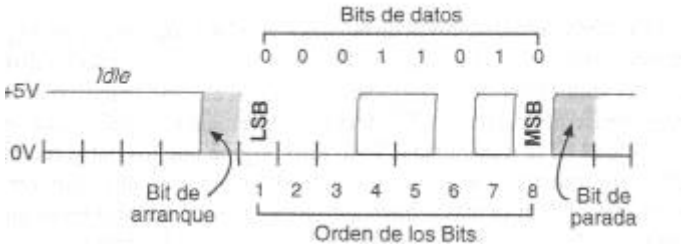


Figura 3.10. Estructura de un carácter que se transmite serialmente

La figura 3.10 muestra la estructura de un carácter que se transmite de forma asíncrona. Normalmente, cuando no se realiza ninguna transferencia de datos, la linea del transmisor es pasiva (idle) y permanece en un estado alto. Para empezar a transmitir datos, el transmisor coloca esta línea en bajo durante el tiempo de un bit, lo cual se conoce como bit de arranque (start bit) y a continuación, empieza a transmitir con el mismo intervalo de tiempo los bits correspondientes al dato (que pueden ser 7 u 8 bits), empezando por el menos significativo (LSB), y terminando con el más significativo (MSB). Al finalizar se agrega el bit de paridad (Parity) si es que está activada esta opción, y los bits de parada (Stop) que pueden ser 1 ó 2, en los cuales la línea regresa a un estado alto. Al concluir esta operación el transmisor estará preparado para transmitir el siguiente dato.

El receptor no está sincronizado con el transmisor y desconoce cuando va a recibir datos. La transición de alto a bajo de la línea del transmisor activa al receptor y éste genera un conteo de tiempo de tal manera que realiza una lectura de la línea medio bit después del evento; si la lectura realizada es un estado alto, asume que la transición ocurrida fue ocasionada por ruido en la línea; Si por el contrario, la lectura es un estado bajo, considera como válida la transición y empieza a realizar lecturas secuenciales a intervalos de un bit hasta conformar el dato transmitido. El receptor puede tomar el bit de paridad para determinar la existencia o no de errores y realizar las acciones correspondientes, al igual que los bits de parada para situaciones similares. Lógicamente, tanto el transmisor como el receptor deberán tener los mismos parámetros de velocidad, paridad, número de bits del dato transmitido y de bits de parada.

Dentro de los microcontroladores hay algunos que poseen funciones y registros especiales para las comunicaciones seriales, tales como la familia PIC16C63 o PIC16C73

de Microchip, los cuales se encargan de manejar todos los aspectos relacionados con las comunicaciones asíncronas, si previamente se han definido todos sus parámetros. Aún si el microcontrolador o microprocesador no posee la opción de las comunicaciones seriales, esta se puede implementar siempre y cuando se tenga presente la duración de cada uno de los bits en la línea. El elemento clave es detectar el bit de arranque, bien sea a través de interrupciones, o bien a través de la lectura frecuente de la línea que contiene los datos. En ambos casos, lo recomendable es que después de detectado el bit de arranque, la lectura de los bits restantes se realice en la mitad del bit, con un error permitido en cada uno de ellos del 3% del tiempo (aunque se podría extender hasta el 4%), sin que se presenten errores de lectura.

En los circuitos digitales, cuyas distancias son relativamente cortas, se pueden manejar transmisiones en niveles lógicos TTL (O - 5V), pero cuando las distancias aumentan, estas señales tienden a degradarse debido al efecto capacitivo de los conductores y su resistencia eléctrica. El efecto se incrementa a medida que se incrementa la velocidad de la transmisión. Todo esto origina que los datos recibidos no sean iguales a los transmitidos, lo que no se puede permitir en una transferencia de datos. Una de las soluciones más inmediatas en este tipo de situaciones es aumentar los márgenes de voltaje con que se transmiten los datos, de tal manera que las perturbaciones causadas se puedan minimizar e incluso ignorar.

Ante la gran variedad de equipos, sistemas y protocolos que existen surgió la necesidad de un acuerdo que permitiera que los equipos de varios fabricantes pudieran comunicarse entre sí. A principios de los años sesenta se desarrollaron varias normas que pretendían hacer compatibles los equipos, pero en 1962 se publicó la que se convirtió en la más

popular: la norma RS-232. Esta norma define la interface mecánica, las características, los pines, las señales y los protocolos que debía cumplir la comunicación serial. La norma ha sufrido algunas revisiones, como la RS-232C en 1969 y la la EIA/TIA-232E en 1991.

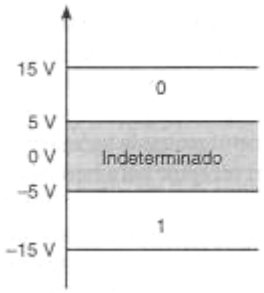


Figura 3.11. Niveles de voltaje RS-232

De todas maneras, todas las normas RS-232 cumplen básicamente con los mismos niveles de voltaje, como se puede observar en la figura 3.11:

- Un uno lógico es un voltaje comprendido entre -5V y -15V en el transmisor y entre -3V y 25V en el receptor.

- Un cero lógico es un voltaje comprendido entre 5V y 15V en el transmisor y entre 3V y 25V en el receptor.

Por lo tanto, deben existir dispositivos que permitan convertir niveles TTL a niveles RS-232 y viceversa. Los primeros dispositivos utilizados fueron los drivers MC 1488 y los receivers MC 1489 de Motorola, de los que se desarrollaron versiones mejoradas como los SN75188, SN75189 de Texas Instruments y algunos similares de otros fabricantes. Todos los dispositivos nombrados anteriormente necesitan tres voltajes diferentes para su operación cuando el equipo actúa como transmisor y receptor, lo cual no representa ningún problema en computadores tipo PC, ya que se disponen de estos voltajes en la fuente. Pero

cuando se trata de sistemas de microcontroladores, en las cuales el espacio es importante y no se puede disponer de voltajes diferentes a 5 voltios, estos circuitos integrados no se pueden utilizar. Para esto se han desarrollado alternativas muy útiles, como el integrado MAX232 que describiremos más adelante.

Se debe tener presente que la norma RS-232 fue desarrollada hace más de 30 años, época en la cual los requerimientos y las capacidades de los equipos eran diferentes. En la actualidad esta norma es un poco limitada, tanto para la distancia a la cual se puede transmitir, como para la velocidad y número de transmisores y receptores que pueden estar simultáneamente conectados. Existen otras normas para la comunicación serial, en la cual se incrementa el número de trasmisores o receptores, la velocidad de transmisión, la distancia, etc. Pero a pesar de esto, los principios lectores siguen siendo los mismos de la comunicación asincrónica y de la interface RS-232.

Aspectos prácticos de una comunicación serial.

El envío de niveles lógicos (bits) a través de cables o líneas de transmisión necesita la conversión a voltajes apropiados. En un circuito lógico o con microprocesador se trabaja con niveles de voltaje inferiores a 0.8 para representar el valor lógico O y voltajes mayores a 2.0 para representar el valor lógico 1. Por lo general, cuando se trabaja con familias TTL y CMOS se asume que un "O" es igual a cero voltios y un "1 "a+5 V.

Cuando la comunicación que se pretende hacer es muy corta, se pueden conectar directamente el transmisor y el receptor para hacer la transferencia de bits usando los mismos niveles lógicos tradicionales de O y 5 V. Pero cuando la distancia es mayor a los dos metros, la información digital se afecta notablemente por acción de la atenuación en el cable, el ancho de banda del mismo y la velocidad con que se transmita. La interface RS-

232C es una de las diferentes soluciones que hay para esta situación. Básicamente consiste en cambiar los niveles lógicos de la salida o envío de O y 5V a dos niveles de voltaje de magnitud mayor: uno positivo (+V) para representar el cero lógico y uno negativo (-V) para representar el uno. En el equipo receptor de la información se realiza el proceso contrario, los niveles positivos y negativos que lleguen se convierten a los niveles lógicos tradicionales de O y 5V, figura 3.12. Los niveles de voltaje son simétricos con respecto a tierra y son almenos de +3V para el "O" binario y -3V para el" 1". En la figura 2.37 se muestra un ejemplo de la transmisión de un carácter sobre una línea RS-232, incluyendo sus respectivos niveles de voltaje.

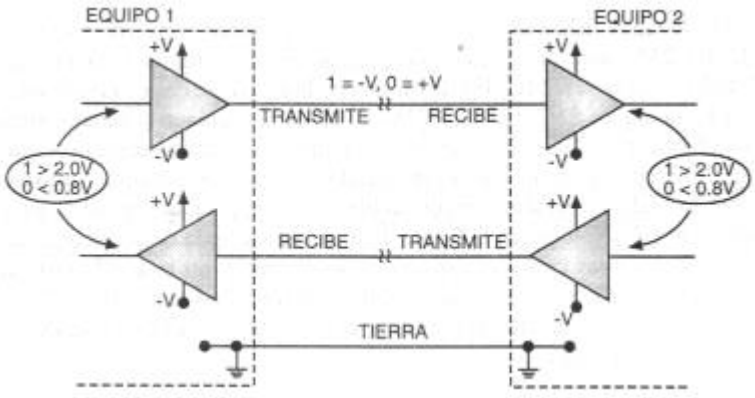


Figura 3.12. Representación de la interface RS-232

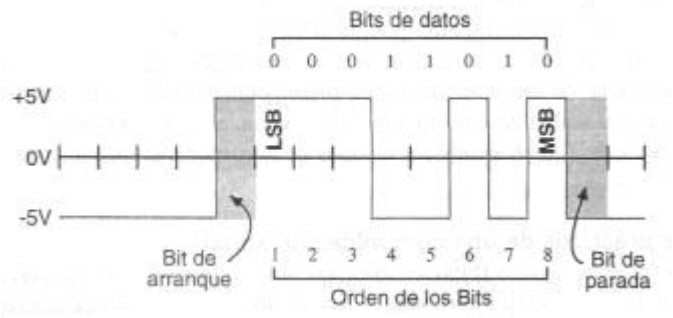


Figura 3.13. Señal presente sobre una línea RS-232

En la práctica, los niveles de voltaje los determinan las fuentes de alimentación que se apliquen a los circuitos de la interface; los niveles más comunes son desde ±12V hasta ±15V. Una interface RS-232 está compuesta por el circuito transmisor que convierte la señal de bajo voltaje del equipo lógico a los niveles de voltaje alto que se necesitan en la línea de transmisión y un receptor que realiza la función inversa. En los manuales de circuitos integrados se llama Une drivers y Une receivers, respectivamente, a los circuitos que ejecutan esta conversión de niveles de voltaje.

Por lo general, se utiliza con las interfaces RS-232 cable multipar o cable ribbon con un solo conductor como referencia de tierra. El ruido que se capta a través de la línea aún puede originar problemas. Para reducir el efecto se suele conectar un condensador en paralelo con la salida del circuito transmisor. Según las reglamentación, los estándares de la interface RS-232 permiten una separación máxima de 15 metros a una velocidad de transmisión no mayor a 9.6 kbps (kilo bits por segundo). Sin embargo, se realizan conexiones a distancias mayores sin problema alguno. En la figura 3.14. se muestran los conectores de la interface RS-232.

Transmisión de datos secundaria (STxD) 14 1Tierra sistema, blindaje (GND)

Trananision de datos (TxD)	15	2 Transmisión Datos (TxD)
Recepción de Datos secundario (SRxD)	16	3 Recepción Datos (RxD)
Reloj de Recepción	17	4 Solicitud de envío (RTS)
No usado	18	5 Listo para envío (CTS)
Solicitud de envío secundaria (SRTS)	19	6 Datos listos para envío (DSR)
Datos listos en terminal (DTR)	20	7 Tierra lógica (SIG)
Detección de calidad de señal	21	8 Detección de Portadora (CD)
Detección de tono (RI)	22	9 Reservado
Selección de rata de datos (DRS)	23	10 Reservado
Reloj de Transmisión	24	11 No usado
No usado	25	12 Detección de Portadora secundario (SCD)
		13 Solicitud de envío secundaria (SCTS)

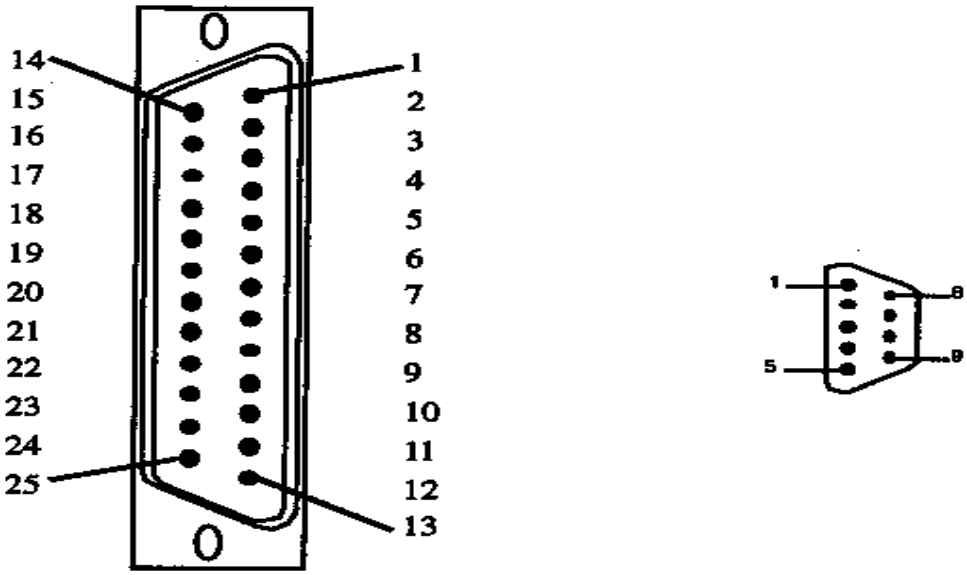


Figura 3.14. Conectores RS-232 con sus respectivos pines

N" Pin	Nombre de la señal
1	Detector de portadora (CD)
2	Recepción de Datos (RxD)
3	Transmisión de Datos (TxD)
4	Datos listos en terminal (DTR)
5	Tierra (GND)

6 Datos listos para enviar (DSR)

7 Solicitud de envío (RTS)

8 Listo para envío (CTS)

9 Detector de tono (RI)

El MAX232.

Este circuito integrado soluciona los problemas de niveles de voltaje cuando se requiere enviar señales digitales sobre una línea RS-232. El MAX232 se usa en aquellas aplicaciones donde no se dispone de fuentes dobles de ±12 voltios; ejemplo, en aplicaciones alimentadas con baterías de una sola polaridad. El MAX232 necesita solamente una fuente de +5V para su operación; un elevador de voltaje interno convierte el voltaje de +5V al de doble polaridad de ±12V.

Como la mayoría de las aplicaciones de RS-232 necesitan de un receptor y un emisor, el MAX232 incluye en un solo empaque 2 parejas completas de driver y receiver, como lo ilustra la estructura interna del integrado que se muestra en la figura 3.15. El MAX232 tiene un doblador de voltaje de +5V a +10 voltios y un inversor de voltaje para obtener la polaridad de -10V El primer convertidor utiliza el condensador Cl para doblar los +5V de entrada a +10V sobre el condensador C3 en la salida positiva V+. El segundo convertidor usa el condensador C2 para invertir +10V a -10V en el condensador C4 de la salida V-. El valor mínimo de estos condensadores los sugiere el fabricante en el recuadro de la misma figura, aunque en la práctica casi siempre se utilizan condensadores de Tantalio de 10 uF. En la tabla de la figura 3.16. se presentan algunas características de funcionamiento de este circuito integrado.

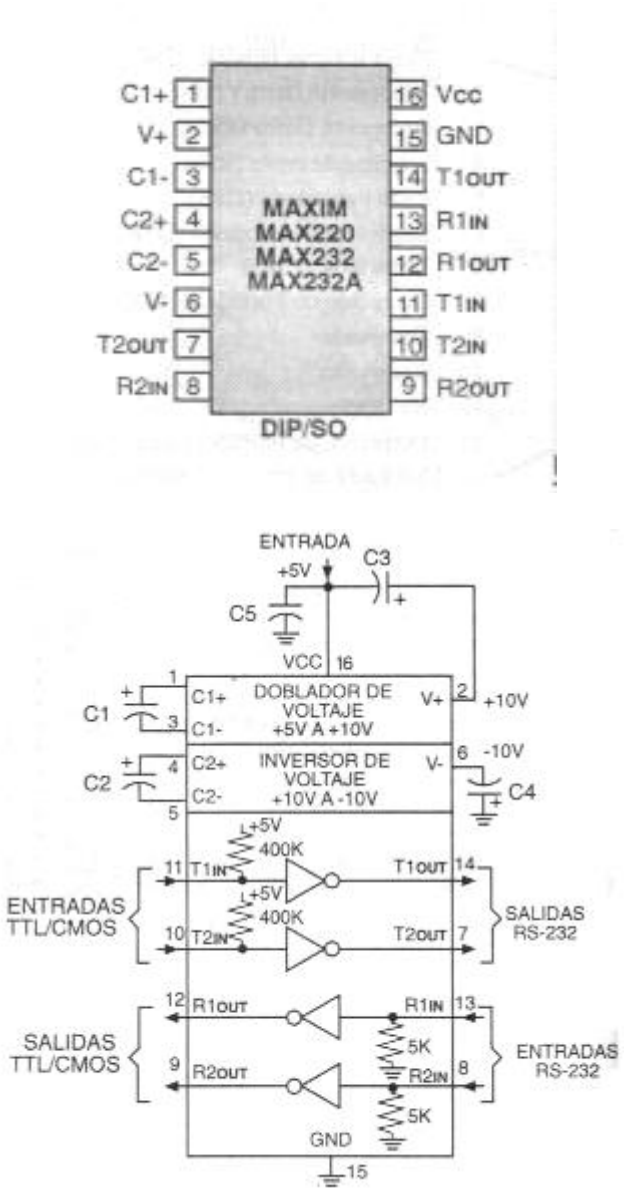


Figura 3.15. Diagrama de pines y estructura interna del MAX232

Una aplicación clásica consiste en conectar las salidas para trasmisión serial TX y RX de un microcontrolador a una interface RS-232 con el fin de intercambiar información con una computadora. La mayoría de los sistemas concentradores de datos están compuestos por sensores conectados a microcontroladores que, a su vez, vía RS-232 le comunican los datos recolectados a un computador central. El MAX232 implementa la interface con la misma fuente de alimentación de +5. voltios. En la figura 3.16 se ilustra la conexión serial de un microcontrolador a través del MAX232

Tabla 3.2. Caracteristicas del MAX232.

LIMITES:			
Fuente de alimentación	**-03 a +16V**		
Voltaje de entrada			
Tin	**-0.3V a (VCC-0.3V)**		
Rin	**+-30V**		
Voltajes de salida:			
Tout	**+-15V**		
Rout	**-0.3V a (VCC +0.3V)**		
Protección corto	**Continua**		
Dicipacion de potencia	**842 Mw**		
CARACTERÍSTICAS a Vcc= +5V, C1-C4=0.1Uf			
TRANSMISOR	**Min**	**Tip**	**Máx.**
Voltaje de salida(carga 3 Kohm)	**+-5V**	**+-8V**	
Entrada baja		**1.4V**	
Entrada alta	**2V**	**1.4V**	
Velocidad		**200Kb/s**	
RECEPTOR			
Rango de entrada			**+-30V**
Entrada baja	**0.8V**	**1.3V**	
Entrada alta		**1.8V**	**2.4V**
Resistencia de entrada	**3Kohm**	**5Kohm**	**7Kohm**

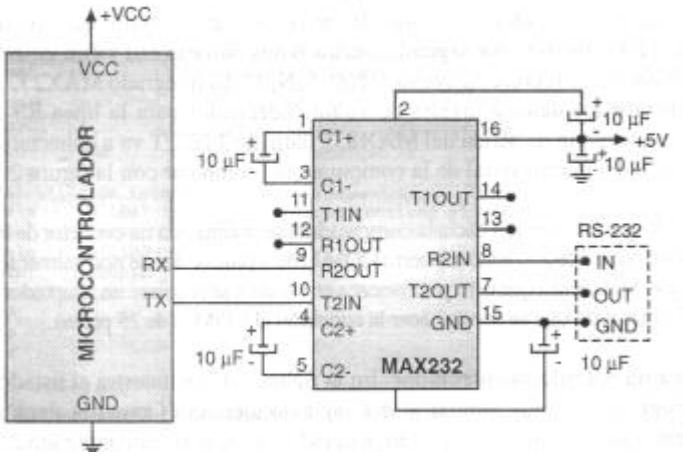


Figura 3.16. Aplicación típica del MAX232

Envío de datos seriales desde el microcontrolador hacia la computadora.

El ejercicio que vamos a realizar tiene por objeto practicar la comunicación serial y entender los principios básicos que la rigen. Consiste en hacer un contador decimal (O a 9), el cual se incrementa cada vez que se oprime un pulsador y muestra el dato del conteo en un display de 7 segmentos, a la vez que lo envía hacia la computadora para que sea mostrado en la pantalla. La comunicación entre el microcontrolador y la computadora se da en un solo sentido (del primero hacia el segundo), por lo tanto se utiliza sólo una línea de datos y el cable de tierra. En la figura 3.17 se muestra el diagrama esquemático del circuito.

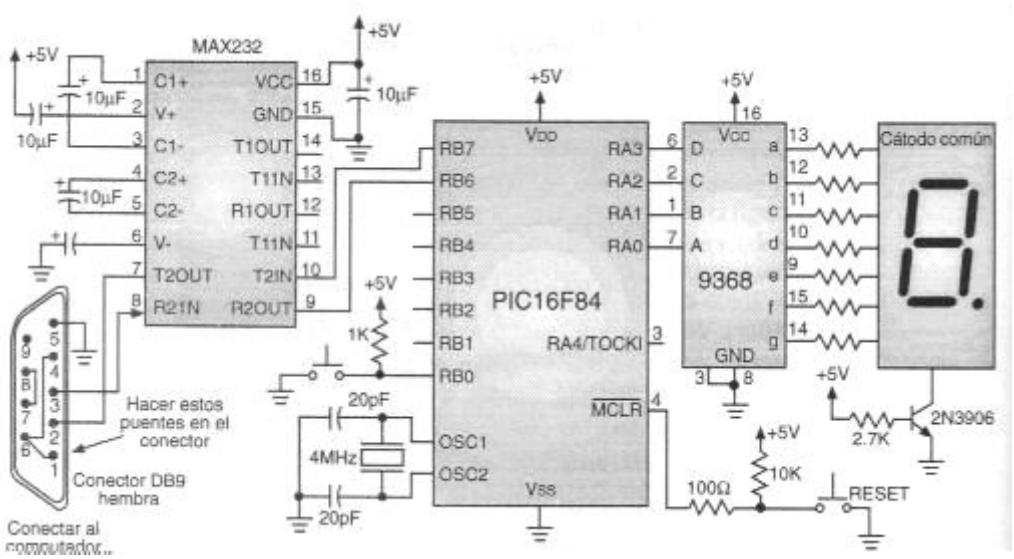


Figura 3.17. Diagrama del contador decimal que envía los datos serialmente hacia la computadora

3.2.5 Potenciómetro Digital.

Manejo de un potenciómetro digital.

Un potenciómetro digital tiene algunas ventajas sobre uno electromecánico como son: mayor precisión, mayor velocidad, menor tamaño, no existe desgaste mecánico, etc. Sus aplicaciones pueden ser: control de volumen digital, convertidor digital a análogo, control digital de una fuente variable y en general, cualquier aplicación que involucre la variación de una resistencia como medio para hacer control.

El circuito integrado del potenciómetro digital es de marca Xicor su principal ventaja es poseer internamente una memoria EEPROM donde conserva el valor de resistencia que debe mantener entre sus terminales. Esto es de gran utilidad cuando se utiliza en sistemas que no poseen baterías de respaldo al momento de cortar la alimentación.

Potenciómetro digital X9C503 de Xicor.

Es un potenciómetro de estado sólido no volátil que contiene un arreglo de 99 resistencias y cuyas uniones son accesibles por el cursor o pin central del potenciómetro.

La posición del cursor es controlada por los pines CS, INC y Ü/D; su posición entre las 99 resistencias puede ser almacenada en memoria EEPROM para no perder su valor cuando se retire la alimentación.

La alimentación del integrado (VCC) es de +5V y los extremos del potenciómetro (VH y VL) se pueden conectar entre ±5V. El hecho de tener 99 resistencias hace que el potenciómetro tenga 100 pasos, es decir que si se polariza entre O y +5V, se obtienen variaciones de 50mV en el cursor. Como se indica en la referencia del integrado (X9C 104), su resistencia total es de 100 kohm, también se consiguen en valores de 10 kohm (X9C 103) y 50kohm (X9C503). La figura 3.18 muestra el diagrama de pines del potenciómetro.

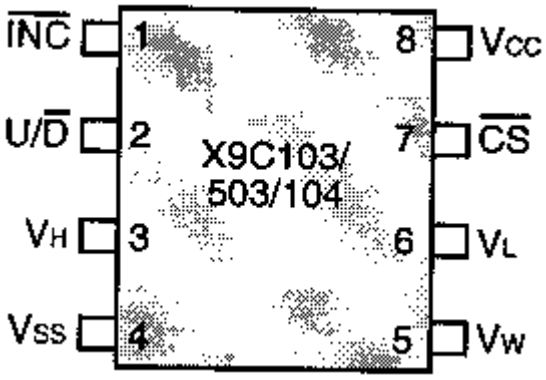


Figura 3.18. Diagrama de pines del potenciómetro digital

Operación del potenciómetro.

Como se muestra en la figura 3.19, el X9C 104 tiene 3 bloques funcionales: el primero lo componen el bloque de control, el contador y el decodificador; el segundo lo conforma la memoria no volátil y el tercero el arreglo de resistencias de salida.

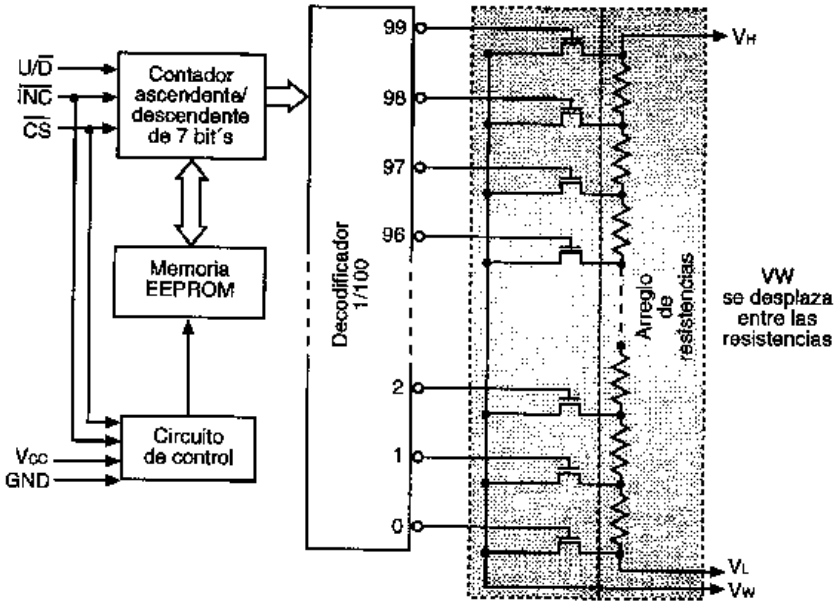


Figura 3.19. Diagrama de funcional del potenciómetro digital

El bloque de control opera como un contador ascendente/descendente (Up/Down). La salida del contador es decodifícada para seleccionar una sola salida que se conecte al cursor del potenciómetro. En condiciones normales, el contenido del contador puede ser almacenado en la memoria EEPROM.

Las entradas INC, U/D y CS controlan el movimiento del cursor a lo largo del arreglo de resistencias. El X9C104 es habilitado cuando el pin CS tiene un nivel bajo, un flanco de bajada en el pin INC hará que se incremente o decremente el contador de 7 bits (dependiendo de la polaridad del pin U/D). La salida de ese contador es decodificada para seleccionar 1 de 100 posiciones en el arreglo de resistencias.

Interface con el microcontrolador.

El manejo adecuado de las señales del potenciómetro digital impone unas reglas específicas. Para obviar este problema, se utilizó un microcontrolador de 8 pines PIC12C508.

En la figura 3.20 se muestra la unión entre el PIC y el X9C104. Se puede ver que los pines de control del potenciómetro están unidos directamente al microcontrolador, en el pin de CS se ha conectado una resistencia de 2.7kohm a fuente. Esto para garantizar que la memoria EEPROM no altere su contenido con el ruido que se genera en el momento del encendido. Los pulsadores SI (subir) y S2 (bajar) se conectan directamente a los pines del PIC, se uüliza una resistencia conectada a la fuente de alimentación para garantizar un nivel alto cuando no se oprimen las teclas.

El pin de MCLR del microcontrolador se ha conectado a la fuente de alimentación para garantizar un reset adecuado.

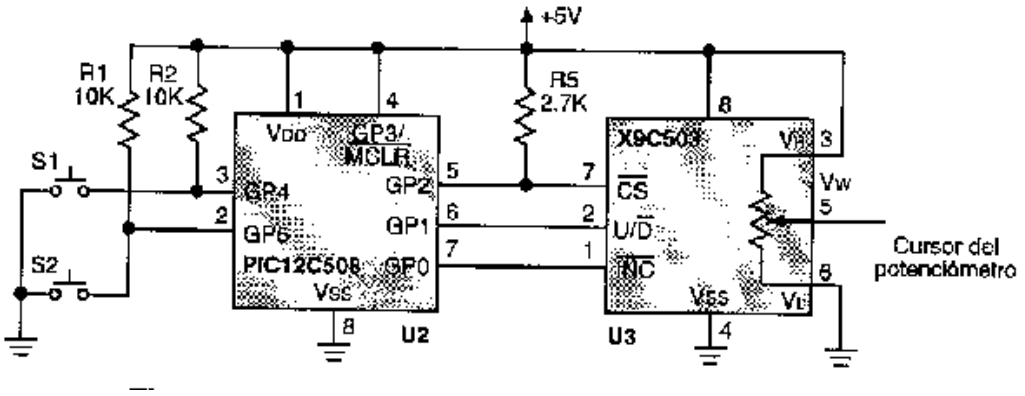


Figura 3.20. Diagrama esquemático del circuito completo

En este circuito se utiliza la principal característica del PIC12C508, que consiste en eliminar la necesidad de componentes extemos. Como se puede ver, no se ha conectado ningún tipo de oscilador ya que se utiliza la opción de oscilador interno de 4 MHz.

Los pines 3, 5 y 6 del X9C104 corresponden al potenciómetro digital propiamente dicho. Con este circuito se puede tener una resistencia variable idéntica a un potenciómetro

electromecánico, la única restricción es que los extremos del mismo no se pueden conectar a una fuente de alimentación que sobrepase los ±5V.

Software del microcontrolador. El programa que se graba en el microcontrolador solamente debe encargarse de leer los pulsadores SI y S2, de acuerdo a ellos debe generar las señales para que el potenciómetro aumente o disminuya su resistencia.

3.2.6.- Memoria EEPROM Serial.

Conexión de memorias seriales al PIC.

Las técnicas para almacenar información en medios electrónicos se perfeccionan más cada día. A diario vemos ejemplos de su utilización en nuestros hogares y oficinas, por ejemplo, en receptores de televisión, reproductores de compact disc, sistemas de control remoto, impresoras, fotocopiadoras, teléfonos celulares, etc. Una de estas tecnologías corresponde a las llamadas memorias EEPROM seriales, las cuales tienen grandes ventajas si se comparan con otras posibilidades. Entre sus principales características se cuentan:

- Se pueden conectar fácilmente con microprocesadores o microcontroladores, inclusive algunos de ellos tienen pines dedicados para esta labor.
- Transferencia de datos de manera serial, lo que permite ahorrar pines del micro para dedicarlos a otras funciones.

- Ocupan la décima parte del espacio de las memorias que trabajan en paralelo, esto permite ahorrar dinero debido al menor tamaño del circuito impreso.

- El consumo de corriente es mucho menor que en las memorias que trabajan en paralelo, esto las hace ideales para sistemas portátiles que funcionan con baterías.

El objetivo de esta práctica es mostrar los aspectos más importantes de su tecnología y enseñar conceptos básicos para su utilización en circuitos reales, se basa en las memorias que tienen comunicación a 2 hilos empleando la interface I2C, cuyas referencias más conocidas son 24LCO 1/02/04/16. La velocidad de transferencia de información para estos dispositivos es de 100 ó 400 kHz (aunque el límite lo impone el protocolo PC más no la tecnología del dispositivo). Como característica importante de este elemento se tiene la inmunidad al ruido, dado que este integrado tiene filtros en los pines de comunicación.

Memorias 24XX.

Figura 3.21. Configuración de pines de la memoria 24LCXX

Estas memorias utilizan el bus de 2 hilos para comunicarse con otros dispositivos. Dado que cumplen con el protocolo I2C, tiene un pin llamado SCL que recibe los pulsos generados por el dispositivo maestro (o sea el microcontrolador) y otro llamado SDA que maneja el flujo de datos de forma bidireccional (entrada/salida). En la figura 3.21 se muestra el diagrama de pines correspondiente a estas memorias.

Este dispositivo no requiere de un pin habilitador o chip select, ya que en este esquema la transferencia de información solo se puede iniciar cuando el bus esté libre. En este caso, como cada dispositivo tiene su dirección determinada mediante los pines AO, Al y A2; solamente responderá la memoria cuya dirección coincida con la dirección que va encabezando la trama de información. En la tabla 3.3 se muestra la capacidad de almacenamiento de estos dispositivos y las posibilidades de direccionamiento que tienen.

Tabla 3.3. Capacidad de memoria y direccionamiento de las memorias 24LCXX

Referencia	Capacidad en Kbits	Bloques internos	A0	A1	A2	Dispositivos En el bus
24LC01B,24C01	**1**	**1**	**1 ò 0**	**1 ò 0**	**1 ò 0**	**8**
24LC02B,24C02	**2**	**1**	**1 ò 0**	**1 ò 0**	**1 ò 0**	**8**
24LC04B,24C04	**4**	**2**	**X**	**1 ò 0**	**1 ò 0**	**4**
24LC08B	**8**	**4**	**X**	**X**	**1 ò 0**	**2**
24LC016B	**16**	**8**	**X**	**X**	**X**	**1**

Transferencia de la información.

Cuando el microcontrolador desea entablar comunicación con la memoria, debe enviarle una serie de bits que llevan la siguiente información:

1. Se envía el bit de arranque o start bit

2. El código 1010 (propio de estas memorias)

3- La dirección del dispositivo (A2, Al, AO)

4. Un bit que indica que se desea escribir ("O") en la memoria

Luego de esta memoria debe enviar un reconocimiento para informarle al microcontrolador que recibió la información. Dicho reconocimiento, llamado ACK, consiste en poner el bus en un nivel bajo (lo hace la memoria). Después el microcontrolador debe enviar los bits que corresponden a la posición de memoria que se quiere leer o escribir; nuevamente la memoria envía un reconocimiento. El paso siguiente depende de la operación que se vaya a ejecutar.

Si se trata de un proceso de escritura, el microcontrolador solo debe enviar el dato a ser almacenado y esperar el reconocimiento por parte de la memoria para confirmar que llegó correctamente. Si se trata de una lectura, nuevamente se debe repetir los primeros cuatro pasos, solo que en lugar de un "O" que indica escritura, se debe enviar un "1" que indica lectura. Después se espera el reconocimiento y acto seguido se puede leer el byte con el dato que estaba en la posición de memoria que se indicó anteriormente. Cuando se termina la operación, el microcontrolador debe enviar una señal de parada o stop bit. En la figura 3.22 se muestra el diagrama de tiempos correspondiente a todo el proceso descrito anteriormente.

Ejemplo de aplicación.

El ejercicio consiste en hacer un contador de O a 9 con un interruptor pulsador y un display de siete segmentos,similar al ejercicio de la figura 3.1. La diferencia radica en que el número que se muestra en el display se va a almacenar simultáneamente en una memoria 24LC01 (LC quiere decir que puede trabajar desde 2 voltios). Se va a utilizar un microcontrolador PIC16F84 (aunque se puede utilizar un 16C61 o 16C71).

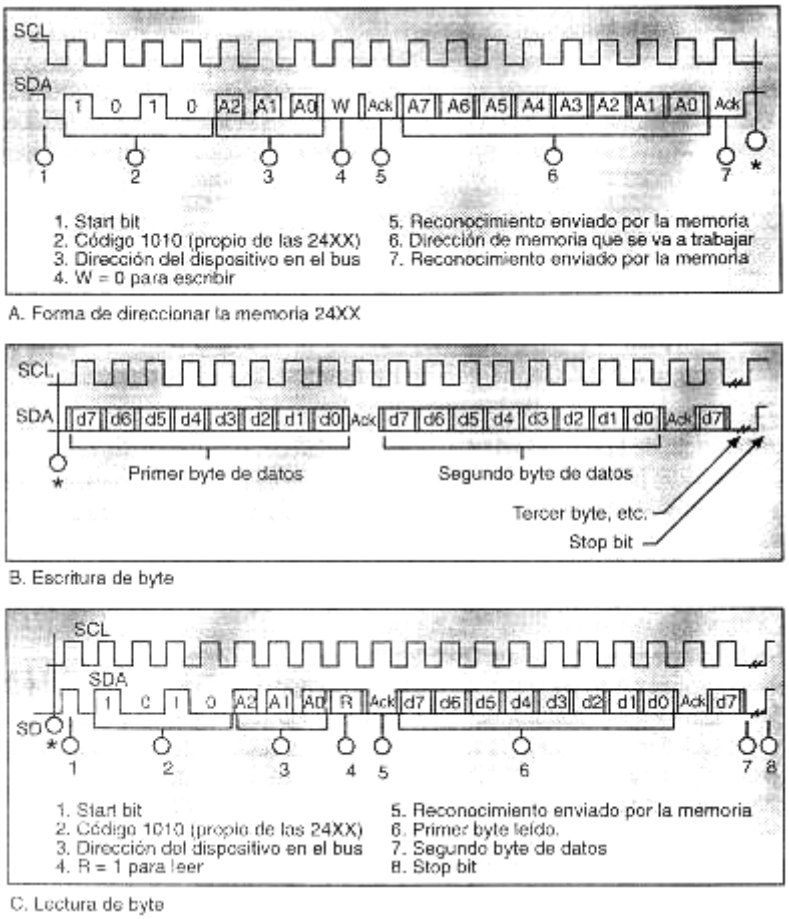


Figura 3.22. Diagrama de tiempos para la lectura y la escritura en una memoria 24LCXX

En la figura 3.23 se muestra el diagrama esquemático del circuito. En este caso los pines de dirección de la memoria se conectaron a tierra, al igual que el pin WP.

La resistencia de 4.7 kohm conectada al pin SDA es necesaria dado que dicho pin tiene salida de colector abierto (ppen collector). El display se conecta al puerto A y el pulsador al pin RBO.

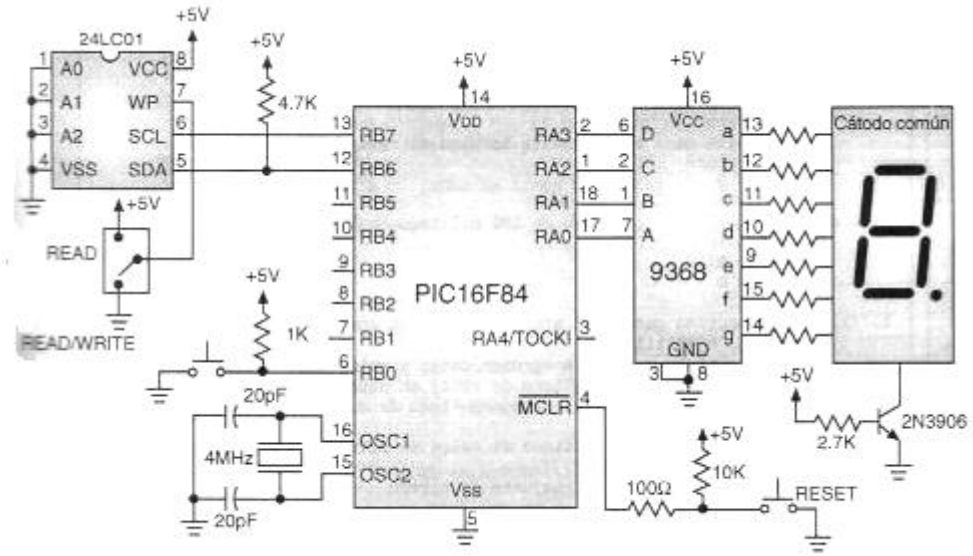


Figura 3.23. Diagrama esquemático del contador con PIC y memoria 24LC01

3.2.7.- Sonda de Temperatura.

Termómetro digital con módulo LCD.

Como ejemplo de la utilización del convertidor A/D del PIC16C71, vamos implementar un termómetro que muestra la temperatura en grados, centígrados, en la pantalla de un display o módulo LCD.

Para medir la temperatura emplearemos un LM35. Este sensor tiene tres pines: Alimentación, tierra y la salida análoga. Este dispositivo presenta en su salida una variación de 10mV/ºC, por lo tanto, el valor de la temperatura se puede obtener directamente, sin necesidad de hacer modificaciones al dato obtenido.

El LM35 puede trabajar en un rango de temperatura entre -55 y 150ºC, la fuente de alimentación positiva puede estar entre 4 y 30 voltios. Además, su precisión es de 0.5ºC. Este elemento viene en un encapsulado plástico TO-92 y tiene la misma apariencia de un transistor.

El módulo de cristal líquido se conecta al microcontrolador utilizando interfase de 4 bits. En este ejercicio, conectamos la entrada análoga al pin RA2/AN2 del microcontrolador y el voltaje de referencia del convertidor análogo a digital, se configura para que sea el voltaje de alimentación del PIC (VDD). En la figura 3.24 se muestra el diagrama esquemático del termómetro digital.

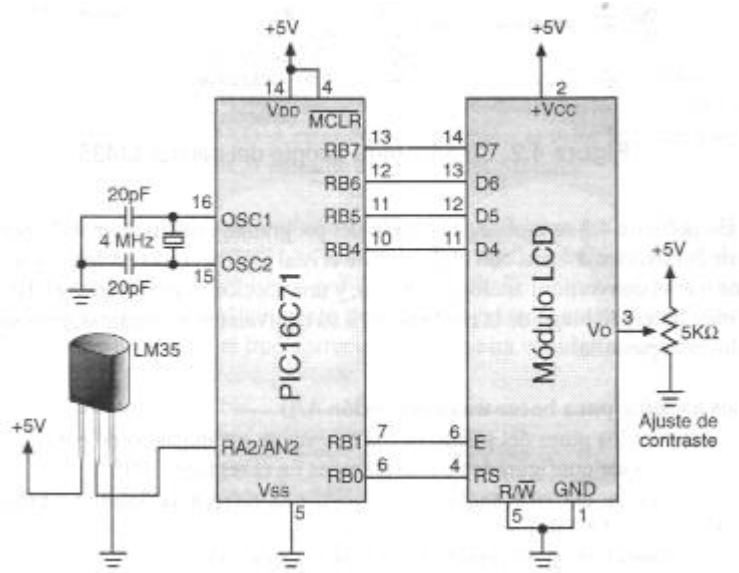


FIGURA 3.24. Diagrama esquemático del termómetro digital con PIC

La máxima lectura del convertidor análogo a digital (0FFH) se obtiene cuando la entrada análoga sea superior a 5 voltios. De tal manera, que si hacemos la división de 5 voltios entre 255 (FFH), obtenemos una relación de aproximadamente 200mV por unidad. Por ejemplo, si la entrada análoga está en 160mV (es decir 16 ºC), el valor digital de entrega el convertidor es 8 (160/20). Dado lo anterior, el valor que entrega el convertidor se debe duplicar para compensar la diferencia (variación de 10mV/ºC y resolución de lectura de 20mV). Aunque esto hace que la lectura sea siempre un número par, se puede aceptar en vista de que retrata de ejercicios netamente didácticos.

Para solucionar el problema que tenemos que duplicar el valor leido y garantizar con el resultado de la conversión corresponda a la temperatura real, se puede utilizar un ampliador operacional para amplificar la señal entregada por el sensor LM35, dicho operacional se debe configurar como amplificador no inversor d ganancia 2. De esta forma, el valor análogo adecuado en la entrada del convertidor y con la fuente de alimentación como voltaje de referencia, se puede obtener una lectura precisa y directa.

En la figura 3.25. se muestra el diagrama del circuito que utiliza el amplificador operacional, si se utiliza este montaje se puede obviar la parte del programa que duplica el valor leído (la parte sombreada en este listado).

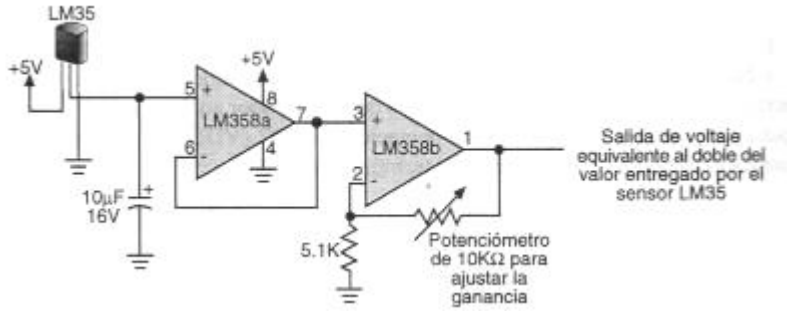


Figura 3.25 Circuito para acople del sensor LM35

3.3.- Construcción del Entrenador.

3.3.1.- Construcción del Entrenador.

El entrenador K-148 se diseñó pensando en utilizar al máximo la capacidad de los microcontroladores PIC16F84, 16C71 y 12C508. Aunque, puede ser usado por otras referencias que tengan pines compatibles. Este dispositivo posee todos los elementos necesarios para realizar las prácticas y ejercicios propuestos.

Figura 3.26. Construcción del Entrenador

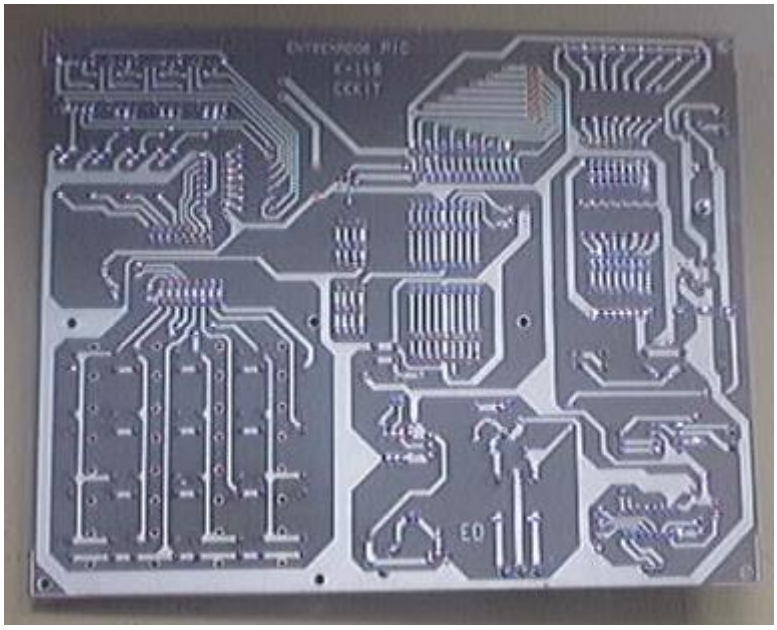


Figura 3.27. Vista Posterior del Entrenador

Para facilitar su aprendizaje y aprovechar al máximo las ventajas de dichos dispositivos, se diseñó el sistema o tablero de entrenamiento, el cual incluye todos los circuitos necesarios para la implementación de las comunicaciones seriales, el manejo de los puertos, las conversiones análogos a digital, etc. Adicionalmente, para experimentar con el PIC16F84 sólo se requiere el circuito programador de microcontroladores y el entrenador PIC avanzado (K-148).

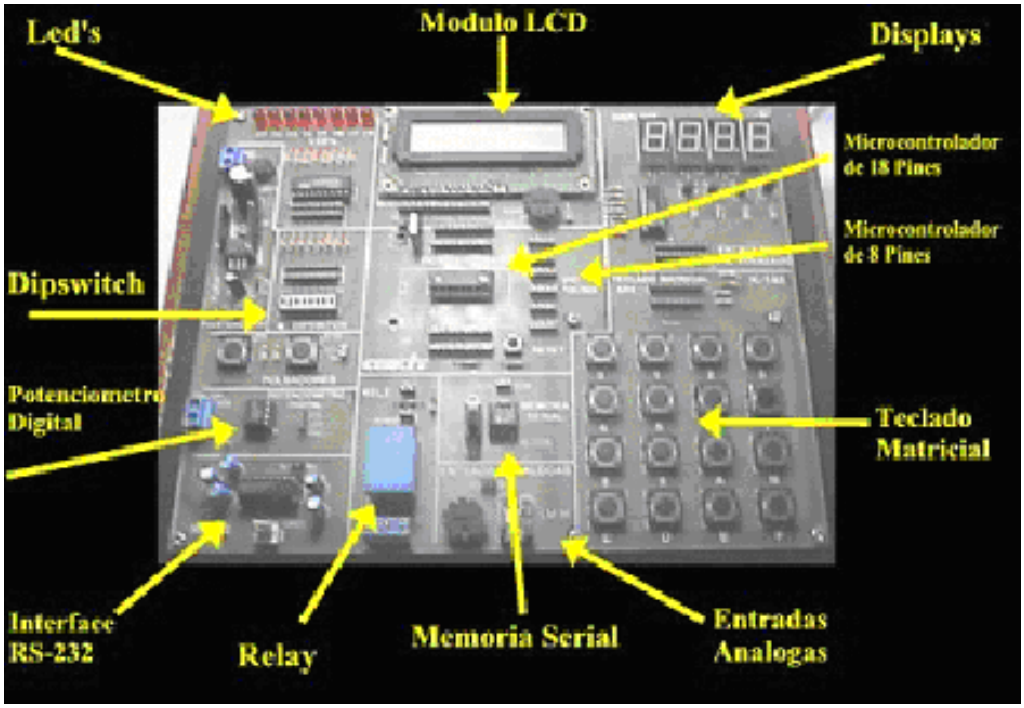


Figura 3.28. Bloques funcionales del entrenador

La principal aplicación de este aparato es servir como elemento de soporte a las personas que desean aprender el manejo de dichos dispositivos. Está conformado por un conjunto de circuitos, independientes entre sí, para que el estudiante pueda realizar los más variados experimentos y proyectos sencillos (para los que se inician), o de cierta complejidad (para los más experimentados). Cada bloque tiene listas las conexiones a la fuente de alimentación y a los diferentes elementos que lo componen, de esta forma, sólo se requiere conectar las entradas y/o salidas al sitio que corresponda. Dichas conexiones se hacen fácilmente empleando alambre telefónico, ya que se implemento un sistema de conectores que permiten hacer esta labor fácilmente. A continuación describiremos cada uno de los bloques funcionales:

Microcontroladores PIC16F84/C71/C508

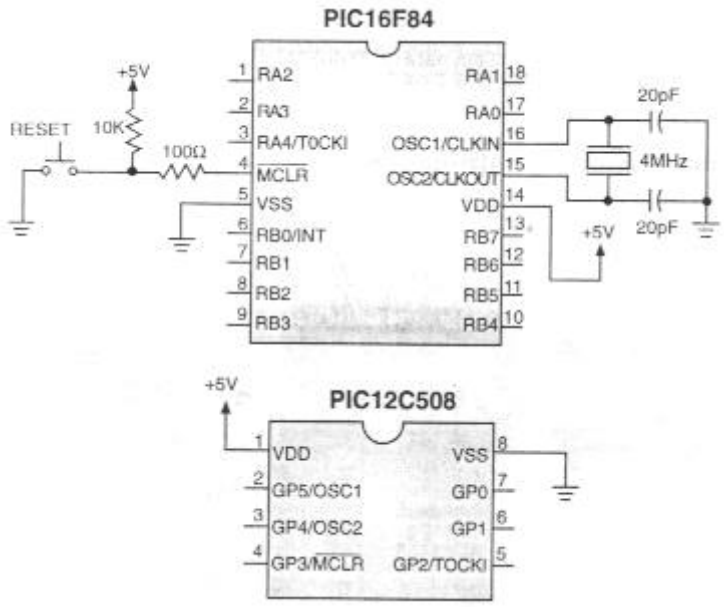


Figura 3.29. Microcontroladores PIC16F84/C71/C508

Estos microcontroladores tienen características especiales que los hacen muy versátiles y funcionales. Entre las principales se encuentran: temporizador/ contador de 8 bits, 4 fuentes de interrupción, 12 líneas de entrada/ salida (84 y 71), circuito de vigilancia (watchdog), capacidad de corriente de 20 mA por pin, Ik de memoria Flash (84) y EPROM (71), convertidor A/D de 4 canales (71), memoria EEPROM de datos (84), oscilador RC interno (508), etc.

El sistema entrenador pretende utilizar todas estas características, para lo cual se dispone de buses de expansión en los pines del microcontrolador La conexión de la fuente de alimentación. el oscilador a cristal y el circuito de reset ya se encuentran implementadas. La base de 18 pines sirve para alojar el PIC16F84 y el 16C71 (al igual que otros PIC de 18 pines), mientras que el PÍC12C508, por ser de tan reducido tamaño, tiene su propia base de 8 pines.

3.3.2.- Construcción de la Fuente.

El sistema se alimenta de la red de 110 ó 220 VAC, incluye un interruptor general y un fusible de protección. Luego de rectificar la señal, se obtiene una tensión cercana a los 12VDC que se emplea para alimentar el relevo. De igual forma, se conecta el regulador de +5V, el cual alimenta toda la parte digital del sistema.

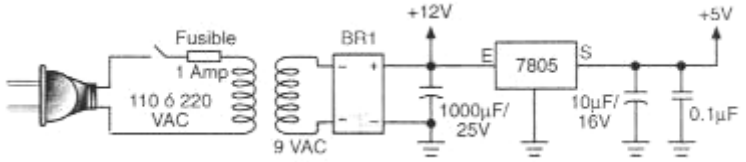


Figura 3.30. Diagrama de la Fuente de Alimentación

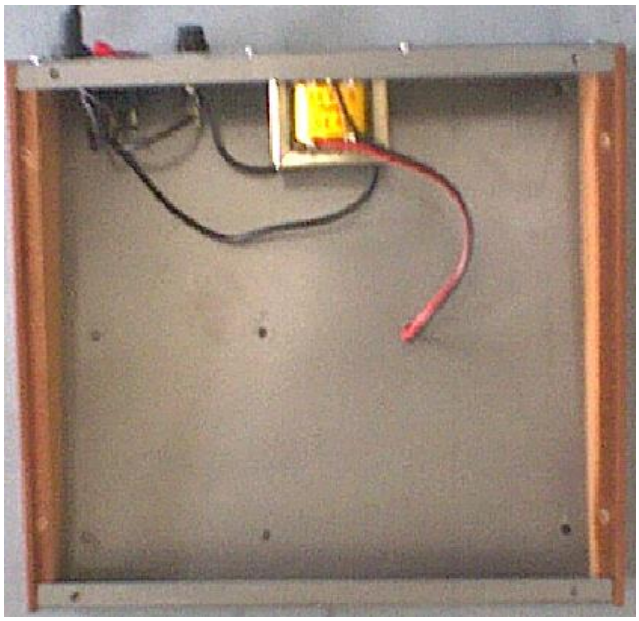


Figura 3.31. Fuente de Alimentación

3.3.3.- Quemador – Programador.

Programador por puerto paralelo.

Es un diseño programador de PICs diseñado para trabajar conectado al puerto paralelo de un PC. Se trata de un diseño modificado para soportar la lectura de datos desde el PIC.

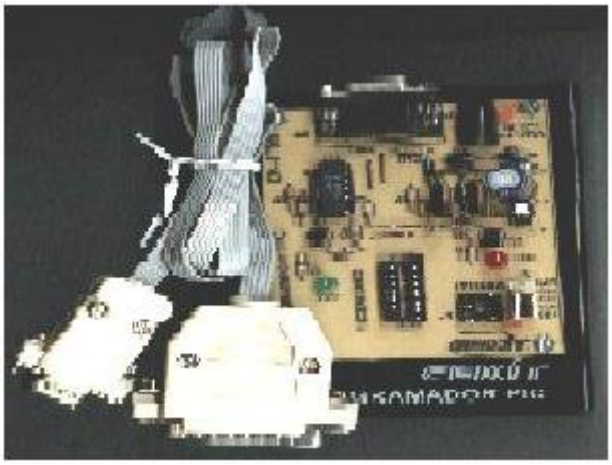


Figura 3.32. Quemador - Programador

3.3.4.- Interfaces de Conexión.

La comunicación serial RS-232 es una de las más utilizadas,

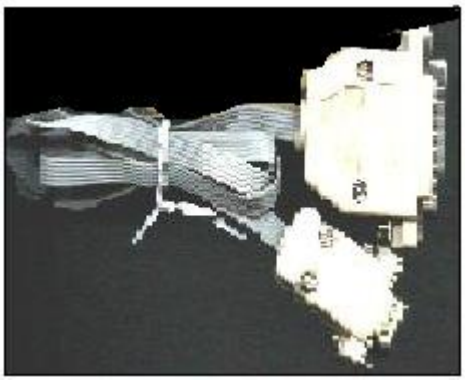


Figura 3.33. Interfaces de Conexión

3.4.- Pruebas de funcionamiento y Análisis de Resultados.

3.4.1.- Pruebas de funcionamiento.

Fuente de alimentación.- Se alimenta de la red de 110 o220 la cual entrega (5 y 12) DC

Los LEDs.- Esta conformado por 8 leds los cuales necesariamente trabajan con un buffer ULN2803(TRANSISTORES NPN) que invierten los pulsos del microcontrolador ya que recibe de los interruptores lógicos un pulso negativo (0L) cuando esta encendido el dipswitch.

Dipswitch.- cuando el interruptor se encuentra abierto (OFF) el microcontrolador leerá 1 lógico, y cerrado 0 lógico.

Relè.- este dispositivo se activará al poner un nivel lógico alto en el pin RL1.

Memoria serial.- el selector sirve para manejar WP de la memoria, si esta en el nivel alto estará en posición de lectura; y si esta en nivel bajo este leerá y escribirá.

Entradas análogas.-utiliza dos entradas una que es el potenciómetro conectado entre 5 voltios y tierra y sirve como referencia para el convertidor análogo digital; y la otra entrada es un sensor de temperatura (LM35) que genera una señal de 10mV/C

Módulo Cristal Liquido.- tiene un potenciómetro que sirve para ajustar el brillo de la pantalla.

Displays.- utilizamos 4 display de cátodo común además los cátodos se pueden manejar independiente.

Teclado matricial.-tiene una matriz de 4 x 4 con los caracteres desde 0 hasta F las filas como F0, F4 F8, FC, las columnas se denominan C0, C1, C2, C3,que incluyen resistencias interna pull up para fijar un nivel alto cuando no esté presionada ninguna tecla.

3.4.2.- Análisis de Resultados.

- Para realizar cualquier practica o ejercicio con los diferentes elementos y dispositivos del entrenador debe realizarse el estudio respectivo.

- Para la programación de los PICS, se debe seguir los pasos especificados en el manual del usuario
- Se debe recordar que los interruptores lógicos da pulso invertidos
- Cuando se trabaje con la memoria serial se debe observar la posición del selector, de acuerdo a la función ha realizar.
- Para cada práctica se debe utilizar el PICs correspondiente.

CAPITULO IV

MARCO ADMINISTRATIVO

4.1.- Cronograma.

CRONOGRAMA DE ACTIVIDADES

	FEBRERO				MARZO				ABRIL				MAYO				JUNIO				JULIO				AGOSTO				SEPTIEMBRE			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
ENTREGA DEL PERFIL										X																						
APROBACION DEL PERFIL											X																					
RECOPILACION DE INFORMACION						X	X	X	X	X	X	X																				
ESTUDIO DE ALTERNATIVAS											X	X	X																			
ADQUISICION DE MATERIALES														X	X	X	X															
ENTREGA DEL PRIMER BORRADOR																	X															
CONSTRUCCION DEL ENTRENADOR K-148															X	X	X	X	X													
VERIFICACION DEL FUNCIONAMIENTO																			X	X												
ELABORACION DE GUIAS DE LABORATORIO																				X	X	X	X	X	X	X						
ENTREGA DEL SEGUNDO BORRADOR																									X							
PRUEBAS FINALES																									X	X	X					
ENTREGA DEL BORRADOR FINAL																											X					

4.2.- Presupuesto.

MATERIALES	VALOR UNITARIO
KIT ENTRENADOR • Teclado Matricial. • Display de 7 Segmentos de 4 Dígitos. • Display Alfanumérico. • Comunicación Rs232. • Potenciómetro Digital. • Memoria EEPROM Serial. • Sonda de Temperatura	350
QUEMADOR-PROGRAMADOR • RESISTENCIAS • COMPUERTAS LOGICAS • TRANSISTORES NPN • DIODOS LET • REGULADORES DE VOLTAJE	70
CABLES Y BUSES DE DATOS • BUS DE DATOS • CABLES DE ALIMENTACIÓN	20
FUENTES DE ALIMENTACION • TRANSFORMADOR • RECTIFICADOR • FILTROS • REGULADORES DE VOLTAJE	20
VARIOS • **HERRAMIENTAS BASICAS**	30
TOTAL	490

CAPITULO V
CONCLUSIONES Y RECOMENDACIONES

5.1.- Conclusiones.

- Se optimizó el Laboratorio de Microcomputadores mediante la construcción de Entrenadores para prácticas en Microcontroladores PIC.
- Al realizar este proyecto se han realizado prácticas con el Entrenador conjuntamente con todos los componentes.
- Se realizo la entrega de un programa para el Quemador Programador.
- Realizado todo el trabajo se diseño la forma de funcionamiento del entrenador y del programa con el cual funciona el Quemador Programador que está especificado en el manual del usuario.

5.2.- Recomendaciones.

Con el fin de mejorara los laboratorios con el que cuenta el Instituto tecnológico Superior Aeronáutico y con mejoras al futuro se pone las siguientes recomendaciones:

- Realizar las prácticas en el laboratorio de microcontroladores con personal especializado con el fin de dar un correcto uso al entrenador (**PIC**), y así poder evitar el deterioro del mismo.
- Utilizar dicho entrenador de acuerdo al manual del usuario.
- Realizar más prácticas con este entrenador a fin de que el estudiante se introduzca mas fácil y rápidamente en el ámbito tecnológico de los PICs.

Bibliografía.

- [Monografías] http://www.monografias.com
- [Monografías] http://www.manuel.com
- [Google] http://www.microcontroladoresPIC.com
- [Monografías] http://www.Hobiespic.com
- Curso Básico Microcontroladores PIC del CEKIT
- Curso Avanzado de Microcontroladores PIC del CEKIT

El PIC16C71

Este microcontrolador promete resolver con ventaja estas situaciones, ya que posee un convertidor análogo a digital interno bastante interesante:

• Cuatro canales de entrada

• Tiempo de conversión mínimo de 20 |is

• Voltaje de referencia interno o extemo

• Resolución de 8 bits con precisión de ±1 LSB

• Rango de entrada análoga desde Vss hasta Vref

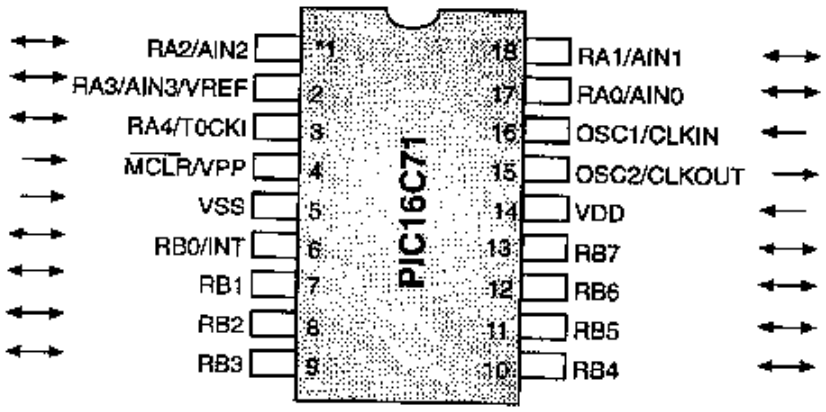


Figura 1. Diagrama de pines del PIC16C71

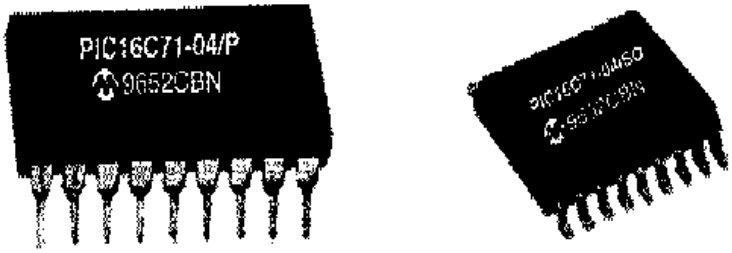


Figura 2. Tipos de encapsulado

El PIC12C50X

El PIC12C508 y el PIC12C509 son microcontroladores de 8 bits basados en la familia PIC16C5X.

Principales características del PIC12C50X:

• 8 pines en total, de los cuales 5 se pueden usar como entrada / salida (I/O)

• Únicamente 33 instrucciones.

• Velocidad de operación de 4 MHz.

• El PIC12C508 tiene una memoria de programa de 512 posiciones. El PIC12C509 tiene memoria de programa de 1024 posiciones.

• El PIC12C508 tiene una memoria RAM de 25 posiciones. El PIC12C509 tiene memoria RAM de 41 posiciones.

• Pila de 2 niveles.

• Oscilador interno de 4 MHz con calibración programable.

• Temporizador / contador de 8 bits.

• Pull-up interno en los pines de entrada / salida y en el pin de reset.

• Bajo consumo de potencia.

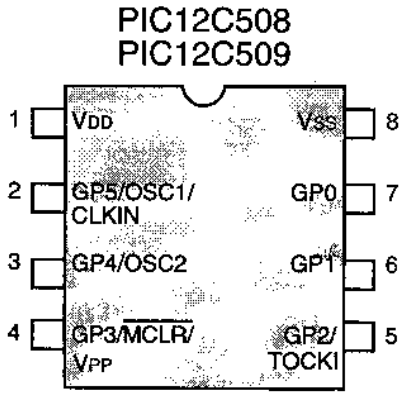


Figura 3. Diagrama de pines del PIC12C50X

Glosario de términos.

- **Bytes.-** La combinación de 8 bits.
- **Leds.-** Diodo emisor de luz.
- **Buffer.-** Un integrado que incrementa la corriente.
- **Chip.-** Es igual a un integrado.
- **Memoria de programa tipo flash.-** Sirve para el desarrollo de prototipos permitiendo reprogramarlo nuevamente sin borrarlo.
- **Pull-up.-** Resistencias internas de teclado matricial.
- **RAM.-** Memoria de acceso aliatorio.
- **ROM.-** Memoria de solo lectura.
- **Stack.-** Pila de memoria.
- **Rutinas.-** Se define como un procedimiento independiente. Al cual se puede llamar desde cualquier parte del programa.
- **ITSA.-** Instituto Tecnológico Superior Aeronáutico.
- **Pin .-** Entrada salida de un integrado.
- **PIC.-** Tipo de familia de microcontroladores.
- **Encapsulado.-** Forma compacta del integrado.
- **EEPROM.-** Memoria programable solo de lectura con un Enable.
- **ASM.-** Extensión del listado de los archivos del directorio.
- **ASSEMBLE.-** Ensamblador del microcontrolador PIC.
- **Decodificador.-** Cambia los códigos a su forma original.
- **NPN.-** Negativo, Positivo, Negativo.
- **LCD.-** Módulo de Cristal Liquido.
- **V.-** Voltios.

Hoja de datos personales 1

1.- Datos bibliográficos

Nombres: Edwin Ramiro**.**

Apellidos: Bautista Machuca.

Edad: 23 años.

Fecha de Nacimiento: 13 de Junio de 1979.

Lugar de Nacimiento: Píllaro.

Estado civil: Soltero.

2.-Estudios Realizados:

Primaria: Escuela Fiscal "Mariscal Sucre".

Secundaria: Colegio Mixto "Jorge Alvarez".

Superior: Escuela Técnica de la Aviación Civil.

Instituto Tecnológico Superior Aeronáutico.

Hoja de datos personales 2

1.- Datos bibliográficos

Nombres: Jorge Andrés**.**

Apellidos: Benavides Ramos.

Edad: 24 años.

Fecha de Nacimiento: 16 de febrero de 1978.

Lugar de Nacimiento: Ibarra.

Estado civil: Soltero.

2.-Estudios Realizados:

Primaria: Escuela Modelo "Velasco Ibarra".

Secundaria: Colegio Seminario "San Diego".

Superior: Escuela Técnica de la Aviación Civil.

Instituto Tecnológico Superior Aeronáutico.

HOJA DE LEGALIZACIÓN DE FIRMAS

ELABORADO POR

_____

Sr. CBOS. Edwin Bautista.

_____

Sr. CBOS. Benavides Jorge.

DIRECTOR DE LA ESCUELA DE TELEMÁTICA

_____

Ing. Eduardo Castillo.
MAYO.TEC. AVC.

Latacunga, Octubre del 2002