UNIVERSIDAD AUTÓNOMA AGRARIA
"ANTONIO NARRO"

DIVISIÓN DE INGENIERÍA

Diseño y Evaluación de un Software Para Microcontroladores PIC, para la digitalización de la señal de un Penetrómetro.

Por:

JOSÉ MANUEL GALLEGOS RAMÍREZ

T E S I S

Presentada como Requisito Parcial para

Obtener el Título de:

INGENIERO MECÁNICO AGRÍCOLA

Buenavista, Saltillo, Coahuila, México.

Mayo de 2004

UNIVERSIDAD AUTÓNOMA AGRARIA
"ANTONIO NARRO"

DIVISIÓN DE INGENIERÍA

Diseño y evaluación de un software para microcontroladores PIC, para la digitalización de la señal de un penetrómetro.

Por:

JOSÉ MANUEL GALLEGOS RAMÍREZ

T E S I S

Que somete a la Consideración del H. Jurado Examinador como Requisito Parcial para Obtener el Título de:

INGENIERO MECÁNICO AGRÍCOLA

Aprobada por el Comité de Tesis

Presidente del Jurado

_____

M.C. Héctor Uriel Serna Fernández

Co-Director de Tesis
Externo del INIFAP

_____

DR. Santos G. Campos Magaña

Sinodal

_____

DR. Martín Cadena Zapata

Coordinador la División de Ingeniería

_____

M.C. Luis E. Ramírez Ramos

Buenavista, Saltillo, Coahuila, México

Mayo de 2004

AGRADECIMENTOS

Te agradezco a ti, **Dios padre**, por haberme permitido concluir una meta mas en esta vida y por iluminar constantemente mi camino, y a ti **Virgen de Guadalupe** por cuidar siempre de la gente que quiero.

A mi **UAAAN** por haberme dado la oportunidad de incrementar mis conocimientos para formarme profesionalmente y por permitirme formar parte de la gran familia de los Buitres. Orgullosamente siempre serás mi ALMA MATER.

Con todo el respeto que me merece al **Ph. DR. Santos Gabriel Campos Magaña,** por todo el tiempo empleado en el desarrollo de este proyecto, por la excelente asesoria técnica y por la confianza brindada hacia mi persona.

Al **Dr. Jesús Uresti Gil**, por facilitarme las instalaciones del área de Mecanización del Campo Experimental (INIFAP), Cotaxtla Veracruz. para realizar este proyecto.

Al **M.C. Héctor Uriel Serna Fernández** por el apoyo y las facilidades brindadas en todo momento, por ser un ejemplo a seguir profesionalmente y por permitirme conocer al maestro, a la persona y al amigo, Ing. Sinceramente Gracias.

A sí también un agradecimiento sincero a los maestros del Departamento de Maquinaría Agrícola, **Al Dr. Martín Cadena Zapata, Dr. Aguinaldo García Santos, M.C Juan Antonio Guerrero Hdz, M.C. Jesús Valenzuela García, M.C. Tomás Gaytán Muñiz, Ing. Juan Arredondo Valdez, Ing. B. Elizabeth de la Peña Casas, Ing. Ramiro Luna Montoya, Ing. Rosendo González Garza.** y a todos aquellos que participaron con la aportación de sus conocimientos en el desarrollo de mi profesión.

A todos mis compañeros de la generación XCVI, de la carrera de Ing. Mecánico Agrícola, en especial a mis amigos **Iván de J. Méndez, Lupita Segundo, Francisco Rodríguez, Antonio Muñoz, Juan José Vázquez, Nadia Escamilla, Miguel Chan, Kennedy Mendoza, Roberto Mtz, Carlos Augusto, Carlos García, Octavio Cárdenas, Miguel Márquez, Ángel Mendoza, Miguel Ramírez, Alfredo Sánchez, Jorge Ruiz.** Por toda su amistad y por hacer de mi estancia en la Narro un bonito recuerdo.

A la familia **Lázaro Córdova**, en especial a mi gran amigo **David**, por sus palabras de animo que siempre me impulsaron a seguir adelante, y por los ratos buenos y malos que hemos pasado.

A la familia **Arévalo Lázaro** por abrirme las puertas de su casa y brindarme confianza, y por todo el apoyo que de ellos siempre recibí.

De igual forma agradezco a **Hipólito Beltruy y familia** por todas las atenciones brindadas.

Al todo el personal que labora en el **área de Mecanización Agrícola del INIFAP**, campo experimental Cotaxtla, y que de una u otra contribuyeron en la realización de este trabajo, en especial, **M.C. Sergio M. Jácome Maldonado**, **Sra. Irene del Ángel, Timoteo Paredes**, por ser excelentes personas.

Agradezco sinceramente a todas aquellas personas, (amigos, compañeros, conocidos), que de una u otra forma me estimularon y me brindaron su apoyo en el trayecto de mi carrera.

Al **Consejo Estatal de Ciencia y Tecnología (COECYT),** por el apoyo económico otorgado para la realización de este proyecto de Tesis.

"M i l G r a c i a s a t o d o s U s t e d e s"

DEDICATORIAS

A mis padres:

Sra. Esmeralda Ramírez Alcudia

A ti madre querida por ser el motivo principal de mi superación, por que nunca podré pagarte tus preocupaciones, desvelos y tristezas, quiero que sepas que este triunfo también es tuyo.
“Gracias Mamá”

Sr. Jesús Gallegos Naranjo

Con cariño y con todo respeto por que he contado con tu apoyo y confianza en todo momento, por que siempre has luchado por darnos lo mejor, tu esfuerzo ha valido la pena
“Gracias Papá”

A mis hermanos

Carlos, Abel, Román, Laura, Ángel, Jaime, Maria y Brenda

Un enorme agradecimiento a ustedes mis queridos hermanos por brindarme su apoyo en los momentos difíciles, por ser la familia que Dios me ha concedido y porque siempre sigamos unidos como hasta ahora.
“De todo corazón donde se encuentren que Dios les bendiga “

A todos mis familiares, por sus palabras de aliento que me impulsaron a seguir adelante, especialmente a mi tío **Manolo Ramírez Alcudia**. Por apoyarme siempre en mi preparación profesional.

A mi novia **Dalila Lázaro Córdova**, por ser tan paciente en tu espera, por darme la confianza y el apoyo necesario para seguir adelante y por formar parte importante en mi vida. “Que Dios te bendiga y te cuide siempre”.

ÍNDICE DE CONTENIDO

ÍNDICE DE CUADROS

ÍNDICE DE FIGURAS

ANEXO I

ANEXO II

ANEXO III

"PROGRAMAS UTILIZADOS"

RESUMEN

En los últimos años la maquinaria agrícola se a incrementado en tamaño, poder y rapidez, factores que han hecho que las operaciones de campo sean mas frecuentes e intensivas, realizándolas comúnmente cuando el suelo está húmedo, situación que ha causado gran daño sobre la estructura del suelo, y por ende la compactación del mismo.

El penetrómetro es el aparato que permite obtener el valor de la resistencia a la penetración del suelo, esta es la fuerza por unidad de superficie necesaria para introducir una punta cónica hasta una cierta profundidad. El diseño de penetrómetros con captadores de fuerza y de distancia conectadas a sistemas de adquisición de datos, permite conocer con gran rapidez la fuerza que opone el suelo a la entrada del cono.

La finalidad de este trabajo es realizar determinaciones de resistencia del suelo, mediante un equipo implementado al tractor donde la resistencia a la penetración y el desplazamiento pudieran ser digitalizadas con el apoyo de dos sensores, mediante la evaluación y desarrollo de un software para microcontroladores PIC-16f84 y 16F872 con dos interrupciones externas y salidas a display con memoria para descargar en una PC, ayudados del software MPLAB y de un programador para realizar la tarea de ensamblado y quemado del microcontrolador.

El sensor optoelectrónico se empleo para determinar el desplazamiento, debido a que su sensitividad es optima para generar las interrupciones que nos permiten desplegar números decimales en la pantalla del módulo LCD.

Con el dinamómetro de anillo, conectado a un amplificador generamos la entrada de señal analógica haciendo la interface al microcontrolador PIC16F872 a través del pin RA0 para hacer la conversión a digital y mostrar el resultado en el módulo

LCD, logrando con esto las bases para medir la fuerza de resistencia del suelo a la penetración.

I. INTRODUCCIÓN

El suelo es un cuerpo natural, que se encuentra sobre la superficie de la corteza terrestre, conteniendo materia viva y soportando o siendo capaz de sostener, y dar sustento a las plantas.

En los últimos años la maquinaria agrícola se a incrementado en tamaño, poder y rapidez, factores que combinados, han hecho que las operaciones de campo sean mas frecuentes y más intensivas, realizándolas comúnmente cuando el suelo está húmedo, situación que ha proporcionado gran daño sobre la estructura del suelo, y por ende una disminución en la porosidad del suelo, causando la compactación del mismo, Cañavate (1989).

La necesidad de conocer como las propiedades mecánicas de los suelos afectan el desempeño de la maquinaria agrícola, nos ha hecho prioritaria la idea de diseñar y construir un instrumento capaz de medir todas estas variables, que directa o indirectamente están relacionadas con el rendimiento del equipo agrícola (tractor-implemento).

El equipo mas difundido hasta ahora para la determinación de estas propiedades mecánicas del suelo, es sin duda el penetrómetro que la Asociación Americana de Ingeniería Agrícola ha estandarizado bajo la norma de ASAE. S 313 (ASAE, 1992), donde se especifican dos medidas de conos, ambos de 30°, uno grande de 20. 27 mm y otro pequeño de 12.83 mm de diámetro y una velocidad de penetración de 30.5 mm s^{-1} (Figura. 2.1).

Tomando como base de referencia los datos anteriores, se diseñara un equipo para ser montado al tractor sobre el enganche tripuntual, el cual a diferencia del existente por la ASAE, este será digitalizado a través del uso de sensores y un sistema de adquisición de datos que al estar combinados serán capaz de medir y registrar señales obtenidas básicamente de la resistencia del suelo, lo que nos permitirá realizar un gran número de lecturas con gran facilidad y rapidez, y conocer la oposición de un suelo al ser penetrado, así como la recopilación y el almacenamiento de la información a la hora de tomar las lecturas, a demás de que nos dará la seguridad de la precisión al momento de realizar las pruebas.

1.3 Antecedentes

Los suelos responden a las presiones aplicadas de diferentes modos, teniendo cada uno su propia relación compresión – deformación. De esta manera los diferentes investigadores alternan el uso de uno u otro método según el particular problema que en ese momento analizan. El uso del penetrómetro de cono para la exploración de la reacción mecánica del suelo a la compresión ha venido a formar parte de los métodos más usados para medir la resistencia del suelo a la penetración.

Muchos estudios han utilizado el penetrómetro para caracterizar la resistencia natural del suelo. A pesar de la diversidad de equipos y métodos utilizados, existe evidencia de que la resistencia o dureza del suelo es caracterizada adecuadamente a través de estos equipos.

Cerisola (2003), dice que el penetrómetro es el aparato que permite obtener el valor de la resistencia a la penetración del suelo. Esta es la fuerza por unidad de superficie necesaria para introducir una punta cónica hasta una cierta profundidad. El tamaño de las puntas y la velocidad con que se introducen en el suelo han sido normalizadas por la ASAE, (1992).

Payán, y Sánchez (2003), señalan que la penetrometría es una de las herramientas mas utilizadas en estudios de la calidad física del suelo. El diseño de penetrómetros con captadores de fuerza y de distancia conectadas a sistemas de adquisición de datos, permite conocer con gran rapidez la fuerza que opone el suelo a la entrada del cono. Varios autores han empleado la resistencia a la penetración como indicador de la compactación en suelos agrícolas.

Una de las mayores ventajas que presenta la utilización del penetrómetro como equipo de medición de la resistencia del suelo, es la versatilidad para ser usado en campo directamente, pudiéndose en poco tiempo realizar un gran número de lecturas, esta característica le permite ser una herramienta de diagnostico inicial sobre la condición física del suelo.

1.4 Justificación

La finalidad de este trabajo es realizar determinaciones de resistencia del suelo, mediante un equipo implementado al tractor donde la resistencia a la penetración y el desplazamiento puedan ser digitalizadas con el apoyo de dos sensores.

Por lo cual nos dimos a la tarea de presentar en este trabajo el desarrollo de un penetrómetro con un sensor para medir el desplazamiento y un dinamómetro de anillo para medir la fuerza, los cuales al estar combinados con un sistema digital tendrán la función de recopilar la información durante la prueba y almacenarla para su posterior análisis.

En la UAAAN, no contamos con un penetrómetro digital operado por la potencia hidráulica del tractor con velocidad uniforme de avance; lo que hace prioritario el diseñar y construir dicho equipo, para poder determinar posteriormente los parámetros exigidos en el empleo de implementos.

El penetrómetro digital montado al tractor, es un instrumento que se requiere para evaluar la resistencia del suelo a la penetración a diferentes

profundidades, así como las características de esfuerzo deformación, esfuerzo en un punto, esfuerzos y presiones en el suelo, que afectan a los implementos agrícolas.

Este trabajo pretende ayudar a resolver los problemas que presentan los diferentes equipos (tractor – implemento), durante las labores de los suelos agrícolas de México, así como facilitar las investigaciones posteriores sobre consumo de combustible, potencia y/o desgaste de los implementos, lo cual ayudara a determinar parámetros de diseño que permitan acondicionarlos para cada región y aumentar la producción con mayor capacidad de uso y mayor rendimiento del equipo agrícola, buscando como todo proyecto economizar la inversión del productor y obtener mayores ganancias para él.

Con la construcción de este equipo de trabajo se desea equipar el laboratorio del Departamento de Maquinaría Agrícola de la UAAAN, así como parte del instrumental de laboratorio de pruebas para el Centro de Desarrollo de Maquinaría Agrícola.

1.3 Objetivos e Hipótesis

VI. Objetivos

- Evaluar dos sistemas de sensores uno de desplazamiento y otro de fuerza a la penetración.
- Evaluación y desarrollo de un software para microcontroladores PIC- 16f84 y 16F872 con dos interrupciones externas y salidas a display con memoria para descargar en una PC.

Hipótesis:

Es posible medir los parámetros de resistencia del suelo a la penetración, con la evaluación de los sensores de desplazamiento y de fuerza, combinados con el sistema digital.

REVISIÓN DE LITERATURA

2.1 Compactación del suelo sobre el desarrollo de las plantas

El suelo es la capa superficial de la corteza terrestre que contiene minerales, materia orgánica, aire, agua y nutrientes. Los suelos agrícolas no sólo constituyen el soporte y fuente de alimentación de las plantas de cultivo, sino que además son objeto de una serie de acciones por parte de los vehículos y máquinas agrícolas. De cara a su mejor utilización, los diseñadores y usuarios han de conocer la respuesta del suelo a todo tipo de operación mecánica en la que esta implicado.

2.1.1 Compactación

Flamand (1995), menciona que la compactación es todo proceso de acción dinámica que aumenta la densidad en un suelo, al mismo tiempo que disminuye su compresibilidad.

Cañavate (1989), establece que la compactación es la variación de volumen de suelo bajo la acción de fuerzas de compresión que pueden ser de origen mecánico (paso de vehículos), o naturales (humectación – desecación, impacto de las gotas de lluvia, etc.). La cuantificación del estado de compactación se realiza basándose en los valores que toman una serie de propiedades del suelo, tales como: porosidad, índice de huecos, densidad aparente y densidad real.

2.1.2 Resistencia a la penetración

Cañavate (1989), menciona que la resistencia de un suelo a la penetración de una determinada herramienta de sondeo, constituye una variable que aglutina otras propias del suelo tales como compactación, cohesión y rozamiento interno. Nos da una idea de la dureza de ese suelo para las condiciones específicas que se encuentra en un determinado momento.

Aunque los valores obtenidos no reflejan más que un índice, que a su vez depende de la forma del elemento que se ha introducido en el suelo (placas, semiesferas, conos, etc.), siempre se puede encontrar una relación entre la resistencia a la penetración y la que opone el suelo a una acción diferente producida bien por un neumático, arrastrado o motriz, o a una determinada herramienta de trabajo de un apero agrícola.

2.2 Generalidades del penetrómetro como equipo de medición

. El *penetrómetro de cono ASAE S. 313*, tuvo su origen en la estación experimental Waterways del ejército de Estados Unidos en Vicksburg Mississipi (Jorajuria). Es un dispositivo simple, ya muy avalado por el extenso uso que muchos investigadores han hecho de él, ya que permite hacer muchas mediciones rápidamente, lo que posibilita compensar la muy alta variabilidad horizontal que los suelos agrícolas tienen respecto al parámetro resistencia a la penetración, y permite revelar datos a profundidades importantes, hasta 600 mm en la mayoría 800 mm en algunos desarrollos particulares, como el citado por Agüero *et al*. (1996), para un penetrómetro que desarrollaron para ser montado sobre el enganche tripuntual del tractor.

Según Cañabate (1989), los aparatos que se utilizan para la determinación de la resistencia a la penetración son llamados penetrómetros, y son los que mas se usan en los suelos agrícolas, los cuales están compuestos por una serie de elementos como son:

- Captador de esfuerzos (mecánico, electrónico o hidráulico).

- Cuantificador (escala numérica, indicador digital, registrador, etc.).
- Varilla soporte.
- Elemento de penetración (semiesfera, placa, punta cónica, etc.).

De entre todos los sistemas diseñados para caracterizar la resistencia a la penetración, hoy día el más utilizado es el penetrómetro propuesto por ASAE–R313 (NN1981) de punta cónica de 30°, este aparato se caracteriza por usar dos puntas cónicas definidas por el diámetro de la base del cono, de 20.27 mm y 12.83 mm (Figura. 2.1).

El penetrómetro es el aparato que permite obtener el valor de la resistencia a la penetración del suelo, esta es la fuerza por unidad de superficie necesaria para introducir una punta cónica hasta una cierta profundidad. El penetrómetro nos permite recabar una serie de información que nos ayudan al conocimiento directo de propiedades físicas e indirecto de las químicas.

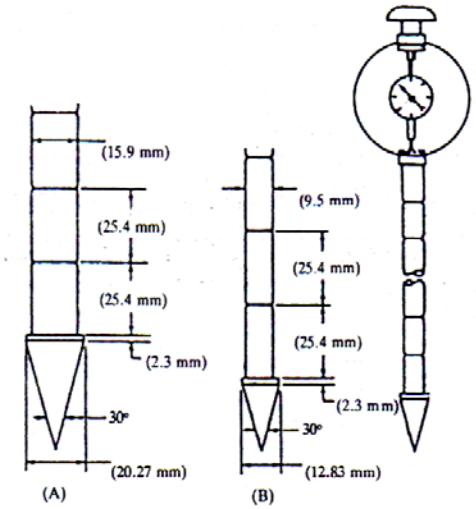


Figura 2.1 Penetrómetro normalizado por la ASAE.

2.3 Transductores

Los transductores son dispositivos que permiten la comunicación de los circuitos electrónicos con el mundo exterior y transducen, o transforman, una forma de energía a otra.

Mojica (2000), dice que un transductor es un dispositivo que ha sido diseñado para reaccionar ante un estimulo físico y proporcionar una salida que puede ser un desplazamiento o por lo regular un voltaje que posteriormente puede ser comparado y analizado dependiendo de los requerimientos del sistema donde se utiliza. Generalmente los transductores son empleados en la medición de magnitudes físicas, como por ejemplo, temperatura, presión, longitud, fuerza, etc.

Wolf y Smith (1992), definen a los transductores como dispositivos que convierten energía o información de una forma a otra. Se emplean extensamente en el trabajo de medición por que no todas las cantidades que se necesitan medir se pueden mostrar con tanta facilidad como otras. Generalmente se puede efectuar una mejor medición de una cantidad si ésta se puede convertir a otra forma que se pueda mostrar con facilidad y exactitud.

Dieck (2000), establece que los transductores de fuerza cubren toda la gama de dispositivos que convierten un esfuerzo mecánico a una cantidad eléctrica como voltaje o corriente. El transductor de fuerza más importante en el estudio de instrumentación es la galga extensométrica.

2.4 Microcontroladores

Un microcontrolador es un circuito integrado programable que contiene todos los componentes de un computador, se emplea para realizar una tarea determinada para la cual ha sido programado.

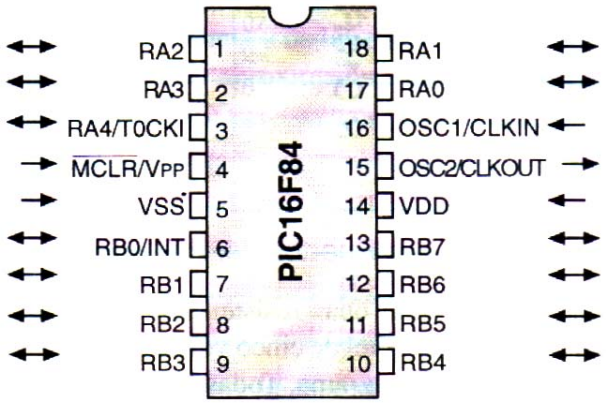
Dispone de procesador, memoria para el programa y los datos, líneas de entrada y salida de datos y suele estas asociado a múltiples recursos auxiliares.

Puede controlar cualquier cosa y suele estar incluido en el mismo dispositivo que controla.

2.4.1 PIC16F84

Es el PIC con memoria Flash más popular. Cuenta con una Memoria de Programación serial de 1024 instrucciones y 64 localidades de memoria RAM. La memoria de programación es eléctricamente borrable ya que no se requiere borrarlo con luz ultravioleta como las versiones EPROM. Internamente cuenta con un contador de tiempo. Este PIC se presenta en un Chip de dieciocho pines de los cuales 13 están disponibles como Entradas / salidas. Figura 2.2

- *Pines y funciones*



VII. Figura 2.2 Diagrama de pines del PIC16F84.

El PIC16F84 (Anexo I –1.1) es un microcontrolador de Microchip Technology, 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.

2.4.2 PIC16C71

El PIC16C71 es un microcontrolador de Microchip Technology, el cual que posee un convertidor análogo a digital con cuatro canales de entrada, lo que permite construir dispositivos controladores de reducido tamaño.

Posee un convertidor análogo a digital interno.

- Cuatro canales de entrada
- Tiempo de conversión mínimo de 20 μs
- Voltaje de referencia interno o externo
- Resolución de 8 bits
- con precisión de ±1 LSB
- Rango de entrada análoga desde Vss hasta Vref

- *Pines y funciones*

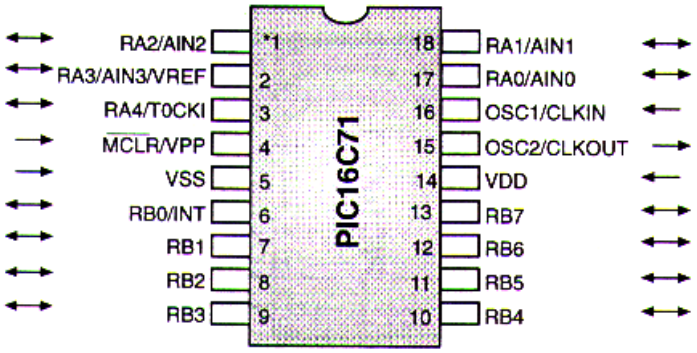


Figura 2.3 Diagrama de pines del PIC16C71.

El PIC16C71 (Anexo I – 1.2) es totalmente compatible con el PIC16F84, las características de programación son las mismas para ambos (numero de pines, puertos de entrada y salida, tipos de oscilador externo y reset,), dentro de su arquitectura, presenta los registros contenidos en la memoria RAM, para el control del convertidor analógico a digital (ADCON0 Y ADCON1), además de que los pines del puerto A aunque se lee y escribe como un registro cualquiera, realizan funciones alternas

2.4.3 PIC16F872

El PIC16F872 (Anexo I – 1.3), es un miembro relativamente nuevo de la familia del los PIC16F87X, pertenecientes a la gama media, admite interrupciones, posee comparadores de magnitudes analógicas, convertidores A/D, cuenta con 28 pines y memoria de tipo Flash.

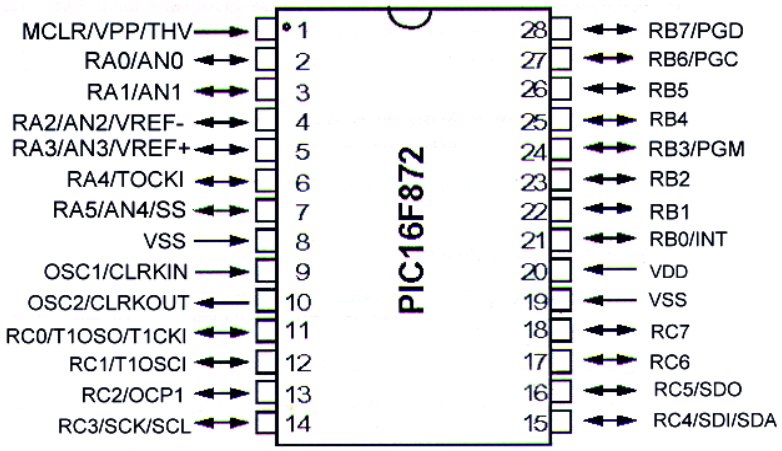


Fig. 2.4 Pines del PIC16F872.

2.5 Módulos de cristal líquido o LCD

Los módulos LCD, se emplean cuando existe 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.

- 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.

- Aunque la configuración de los pines del modulo, se sitúen en posiciones diferentes, conservan las mismas funciones. Algunos módulos LCD tienen luz posterior o “backlight”, para mejorar su visualización, está 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 con una frecuencia de aproximadamente 60 Hz. Otra opción mucho más sencilla es utilizar una resistencia de 10 ohmios (a ½ W) 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 (E), un pin de selección, que indica que el dato es una instrucción o un carácter del mensaje (RS) y un pin que indica si se va a escribir o leer en el módulo LCD (R/W). Figura 1.4.1 del anexo I muestra las funciones de cada uno de ellos.

- Según la operación que se desee realizar sobre el LCD, 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

2.5.1 Funcionamiento del LCD con el microcontrolador

El módulo LCD responde a un conjunto especial de instrucciones, estas deben ser enviadas por el microcontrolador al display, según la operación que se requiera.

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.

Los caracteres que se envían al display se almacenan en la memoria RAM del módulo para después ser enviadas al despliegue en la pantalla.

- Interface con el microcontrolador a 8 bits

Se utiliza el puerto B del microcontrolador PIC16F84 como bus de datos, y el puerto A se encarga de generar las señales control.

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

Como el PIC16F872 contiene 3 puertos en su estructura interna, para la conexión con el módulo LCD utilizaremos el Puerto C para las conexiones de las líneas DB0-DB7 y los pines, (RA0-RA2), del puerto A para generar las señales de control conectadas al RS, R/W, y E, RA4 funciona como colector abierto, pero podemos usar los pines de RA3-RA5.para conectar alguna interrupción externa como pulsadores.

- Interface con el microcontrolador a 4 bits

Para realizar esta conexión con el PIC16F84 se utilizaran los 4 pines de mayor peso del puerto B (RB4-RB7) y las señales de control (RS y E), se generaran con los dos pines de menor peso de este puerto (RB0 y RB1).

Esta misma condición se emplea para el PIC16F872, con la diferencia de que las señales de control se general por medio de los pines , (RA0-RA2), del puerto A. La figura 2.5, muestra la conexión típica de un Módulo de Cristal Liquido, con los pines del microcontrolador.

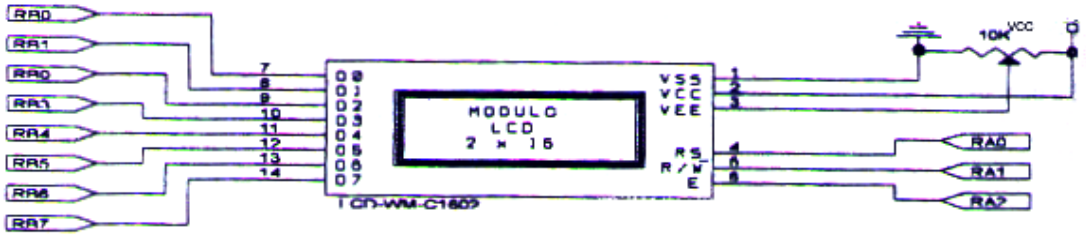


Figura 2.5 Conexión de una pantalla LCD con las líneas del PIC.

2.6 Acondicionador de señal

Más allá del transistor primitivo, el amplificador operacional es el bloque más básico para las aplicaciones analógicas. Funciones fundamentales como ganancia, aislamiento de carga, la inversión de señal, que agregando y/o substrayendo signos se llevan a cabo fácilmente con este bloque. También pueden llevarse a cabo circuitos más complejos, como el amplificador de la instrumentación, una corriente al conversor de voltaje, y filtros, por nombrar sólo unos. Sin tener en cuenta el nivel de complejidad del circuito del amplificador operacional, sabiendo el funcionamiento fundamental y conducta de este bloque ahorrarán una cantidad considerable de tiempo de diseño. Figura 2.6.

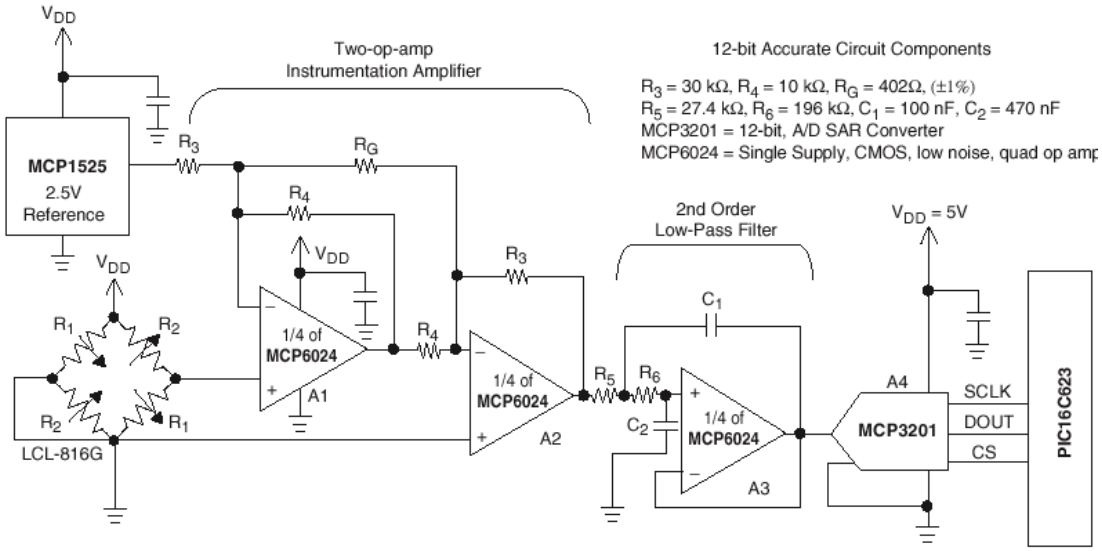


Figura 2.6 Conexión de un acondicionador de señal.

III. MATERIALES Y MÉTODOS

VIII. 3.1 Materiales

Los materiales y equipos empleados para el desarrollo y evaluación del software para la digitalización de un penetrómetro fueron los siguientes:

- Un microcontrolador PIC16F84, memoria tipo flash lo que permite que sea reprogramado, compuesto de 18 pines, 13 pines de E/S 5 del puerto A y 8 del puerto B, utiliza un oscilador de cristal de 4MHz, Cuenta con una Memoria de Programación serial de 1024 registros, 128 posiciones de memoria RAM de 8 bits cada una y 64 posiciones de memoria EEPROM de 8 bits cada una, Figura 2.2.
- Un microcontrolador PIC16F872, 2K x 14 bits de Programa de Memoria tipo flas, contiene 28 pines y tres puertos (A, B y C) para las E/S, único con 35 palabras en la memoria de instrucciones, 64 bytes de memoria de datos EEPROM, 5 canales de 10-bit, convertidor Analógico--Digital (A/D), El rango de voltaje de operación es de 2.0 a 5.5 V, Figura 2.4.
- Un manual (Curso avanzado de microcontroladores PIC CEKIT), referencia para el desarrollo de los programas (Duque, 1998).

- Componentes electrónicos, (tarjetas, integrados, diodos, leds, cables, resistencias, pulsadores, reset, capacitores, etc.), y herramienta (cautín, multímetro, pinzas, desarmadores, reguladores, etc.), utilizados en la construcción del hardware, para cada programa desarrollado.

- Una computadora personal (Toshiba, con procesador AMD – K6 (tm) 3D, memoria RAM de 32.0 MB, con capacidad de disco duro de 4.02 GB).

- El software MPLAB como editor que utiliza el ensamblador MPASM y la herramienta de simulación que utiliza se llama MPLAB-SIM, usados para redactar la estructura del programa, ensamblarla y realizar la simulación.

- Un programador o quemador PICSTART Plus (Development programmer) de Microchip Technology, conectado al puerto serial de la computadora y a una fuente de alimentación de +9 V, utilizado para grabar el programa en la memoria del chip. Figura 3.1.

- Un Módulo de Cristal Líquido, con un display de 2 líneas por 16 caracteres y de 14 pines, para visualizar el valor registrado por el sensor.

Figura 3.1. Programador PICSTART Plus.

- Un banco de pruebas para montar el PIC16F84 y ver su desempeño en cada programa, Figura 3.2.

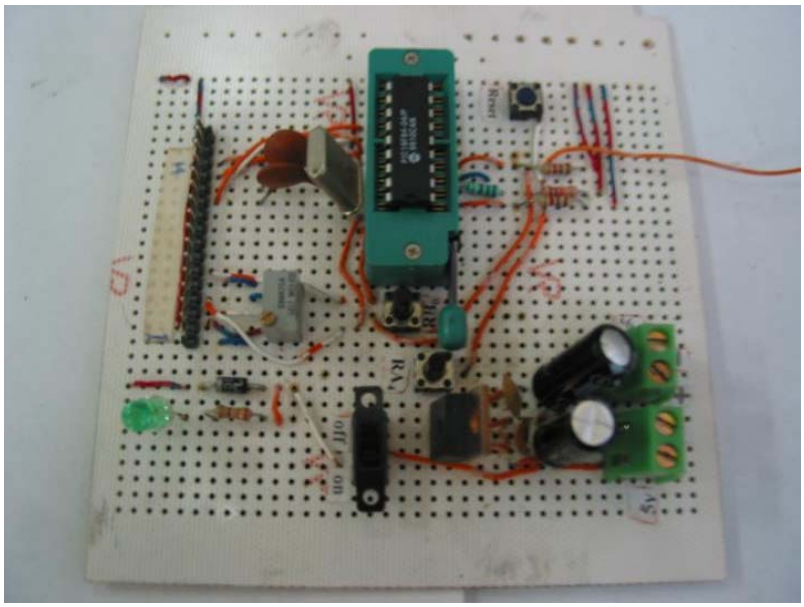


Figura 3.2 Banco de pruebas para el PIC16F84.

- Un banco de pruebas para el PIC16F872, utilizado para ver su funcionamiento en cada programa, Figura 3.3.

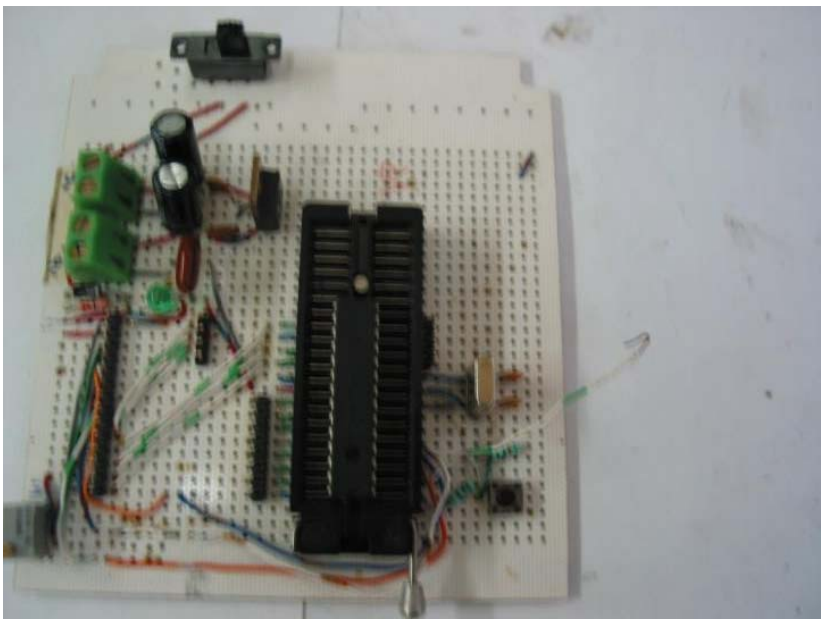
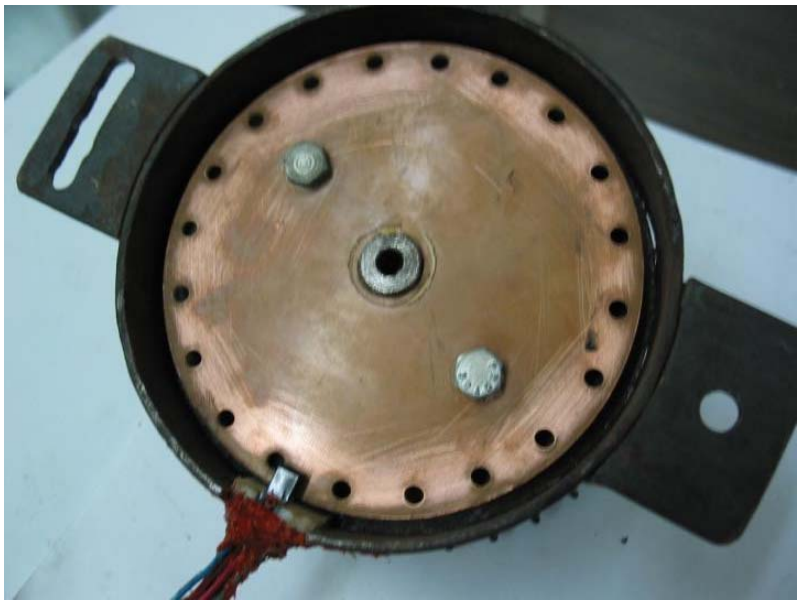


Figura 3.3 Banco de pruebas para el PIC16F872.

- Un sensor optoelectrónico para medir el desplazamiento, que detecta la señal analógica y la envía al microcontrolador, Figura 3.4.



IX. Figura 3.4 Sensor optoelectrónico.

- Un dinamómetro de anillo, para medir la fuerza. introduciendo un voltaje al microcontrolador el cual realiza la conversión de Analógico a Digital, Figura 3.5.

Figura 3.5 Dinamómetro de anillo.

- Un acondicionador de señal ML324N utilizado como filtro para estabilizar la señal enviada por el sensor y amplificar la señal del mismo.

- Un tripie con una canastilla y contrapesos de pesos conocidos, para simular la fuerza de resistencia de penetración del suelo.

3.2 Metodología

La metodología seguida para el desarrollo del software se muestra en el siguiente diagrama:

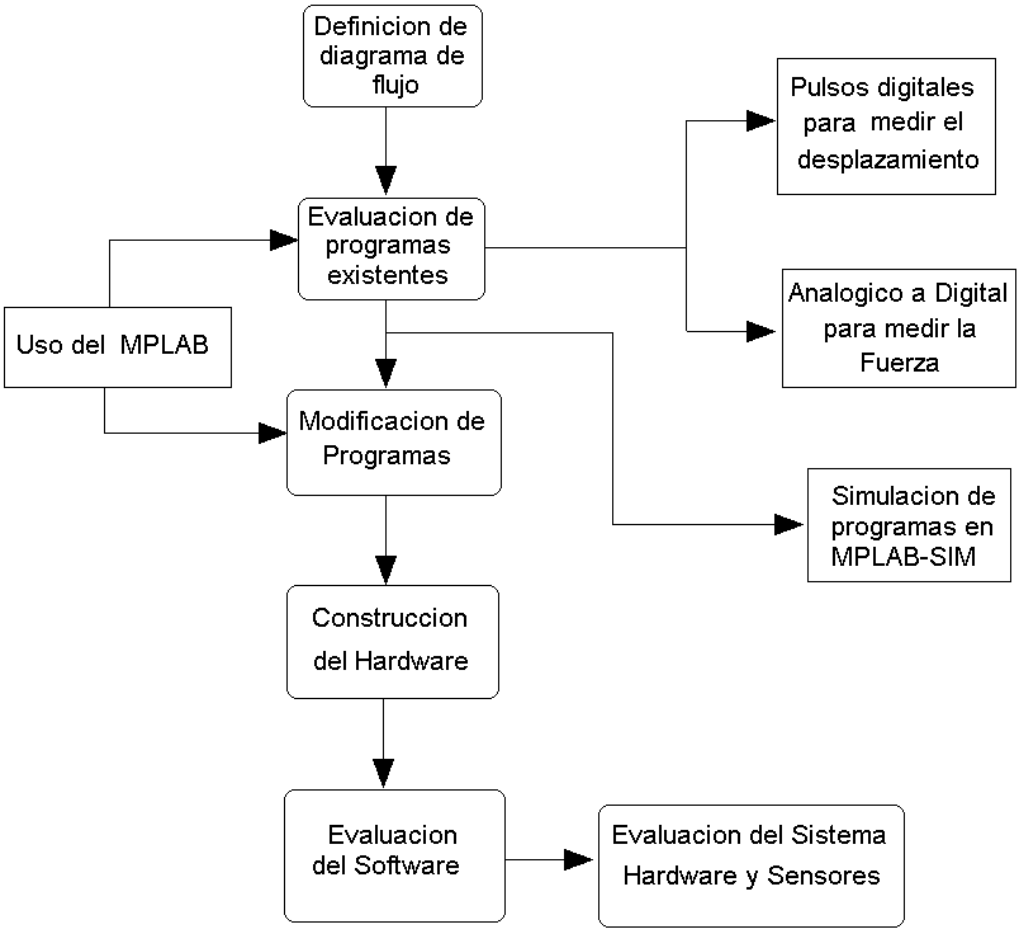


Figura 3.6 Diagrama empleado en el desarrollo del software de fuerza y desplazamiento.

Para determinar la Evaluación y desarrollo de un software para el monitoreo de fuerza y desplazamiento con microcontroladores PIC, interrupciones externas y salidas a display se inicia con la construcción del diagrama de flujo, (Figura 3.6) definiendo cada una de las subrutinas a evaluar.

3.2.1 Desarrollo digital para monitorear desplazamiento

Como introducción para tener contacto con la memoria. del microcontrolador se usó el Proyecto 1 Conexión de Led y Dipswitch del Anexo III, con este ejercicio se buscaba familiarizarnos con la estructura del programa, (la definición de los registros y la configuración de los puertos A y B como entradas ó salidas), el uso del MPlab y del quemador PICSTART Plus.

Con la ayuda del MPlab, transcribimos el cuerpo principal del software, programando los pines del puerto A como salidas y los pines del puerto B como entradas. Debido a que el ensamblador exige cierta tabulación de sus elementos, para la escritura del programa pusimos en la primera columna la definición de variables, mientras que las instrucciones las colocamos en la segunda y tercera columna y los comentarios antecedidos de un (;) en la cuarta columna, el uso de mayúsculas y minúsculas generalmente no son obligatorias pero obedecen a una serie de reglas o normas de estilo que facilita la lectura del código fuente, se procedió al ensamblado para verificar los errores y corregirlos, en caso de que los hubiera.

Usando el PICSTART Plus procedimos al quemado o grabado del programa en la memoria del microcontrolador, y se monto en una tarjeta de prueba para ver su funcionamiento, Figura 3.2, después configuramos el puerto A como entrada y B como salida, se quemo y se probó nuevamente el programa, con lo que se comprobó el uso de los puertos como entradas ó como salidas digitales.

En el Proyecto No. 2 Manejo de un módulo LCD con conexión a 4 bits Anexo III, se muestra el software utilizado como base para mostrar un mensaje en el display en donde la conexión entre el PIC y el módulo se hizo a través de los pines del puerto B (RB4-RB7) del microcontrolador y las señales de control generadas por RB0 y RB1, con lo cual dejamos libre todo el puerto A y los pines RB2 y RB3 para usarlos en otras funciones. El objetivo de este ejercicio era evaluar el funcionamiento de la rutina llamada CONTROL, que es la encargada de generar las señales y los tiempos necesarios para que exista una correcta comunicación entre el PIC y el LCD, Se construyó el circuito de la Figura 4.5 para probar el programa.

Para el tercer paso se tomo el Proyecto 2 y se le modificaron los puertos A y B, de tal forma que el pìn RB0 quedara configurado como entrada donde se conecto un pulsador que generaría las señales de entrada al microcontrolador. Los puertos se modificaron en el programa principal en la parte de inicio y en la rutina de control, debido a que las señales se generaban por el RB0. Quedando como se muestra en el Proyecto No. 3 Manejo de un LCD con los pines RA0 y RA1 para las señales de control del Anexo III y la Figura 4.7, muestra el circuito empleado.

La lógica para la lectura del desplazamiento era que necesariamente sé tenia que diseñar un programa que registrara la entrada de pulsos, primero a través de una interrupción externa y después poder acondicionarle el sensor que realizaría la misma función. Se partió de un programa que registra la entrada de pulsos a través del pin RB0 y realiza el incremento a 1 cada vez que se pulsa el botón, Proyecto No. 5 Programa para probar la Conexión de una interrupción externa del Anexo III, su diseño solo admite la entrada de 10 pulsos y cuando llega a este valor muestra en el display un número que tiene almacenado en el registro W, se redacto y se grabo el programa.

El siguiente paso fue hacerle ajustes a PULSA, con la idea de poder obtener en pantalla él número de pulsos introducidos, a la subrutina mide1 se le implemento esta instrucción, a la instrucción de XORLW le cambiamos el valor comparativo de 10 la primera vez, por 5

la segunda por y la ultima por 255. Y obtuvimos el Proyecto No. 6 Registro de 10 pulsos y despliegues en el LCD.

Después se empleo la rutina de pulsa que aparece en el Proyecto No. 4 Manejo de un display de 7 segmentos del Anexo III y se la acoplamos al Proyecto No. 6, modificándole la instrucción de XORLW para que hiciera el conteo hasta 255 pulsos, modificándose también el programa principal, para que cada vez que llegara a 255 se reiniciara el conteo automáticamente, y agregamos una instrucción para iniciar el contador en cero. Consiguiendo el propósito general del Software para medir desplazamiento, Proyecto No. 7 Contador decimal de 255 pulsos desplegados en un Módulo LCD.

Como ultimo paso se conecto el microcontrolador a través del pin RB0 con el sensor optoelectrónico. El equipo donde se encuentra el sensor consta de un engrane acoplado a un caparazón que cubre a un disco con 20 agujeros, y pegado a su pared se encuentra el sensor que genera una señal de interrupción cuando se gira el engrane y coincide un agujero del disco con el sensor, enviando al despliegue un pulso. Figura 3.7.

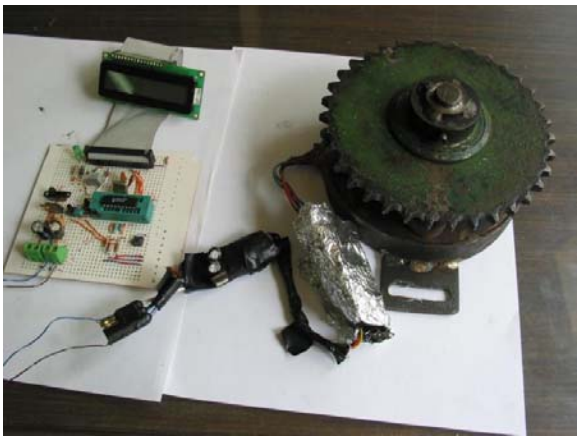


Figura 3.7 Conexión del sensor optoelectrónico al microcontrolador.

Desarrollo analógico-digital para monitorear fuerza

La otra variable a sensar es la fuerza, utilizando el PIC16F872.

Se partió de una fuente de programas bajados de Internet, iniciando con Proyecto 1.1 Conexión de Leds y un Dipswitch, Anexo III, y al igual que para el 16F84 el primer programa se utilizo con el objetivo de hacer funcionar el microcontrolador, para lo cual se ensamblo en el MPlab y se quemo, construyéndose el circuito incluido en la información, donde habría de probarse usado como banco para poner el PIC al cual se le conectaban y desconectaban los circuitos propios de cada programa. La Figura 4.14, muestra el circuito usado en este programa y la Figura 4.13 la tarjeta que contiene los Leds, los pulsadores y los cables que se conectaron al microcontrolador.

Para el segundo ejercicio empleamos el uso del módulo LCD para visualizar la información enviada por el sensor, en este caso contamos con el Proyecto 1.2 para la conexión a 8 bits entre el PIC y el LCD, del Anexo III, el cual copiamos al MPlab para ensamblarlo y después quemarlo al microcontrolador.

El circuito utilizado para este programa se muestra en la Figura 4.16 y la tarjeta que se acoplo al banco que contiene los tres pulsadores y el selector (Dipswitch), para poder utilizarlo en la Figura 4.16.

Cuando logramos encender el LCD, tomamos el Proyecto 1.3 Conversión Analógico - Digital del Anexo III, que lee la entrada de voltaje entre 0 y 5 volts y realizamos la misma operación de ensamblado y quemado, así como la construcción de la parte restante del circuito donde se encuentra la conexión del acondicionador de señal (Figura 3.8), con una ganancia de 100 a 150. El circuito empleado es el que aparece en la Figura 4.19.

Con el apoyo del multímetro, visualizamos la variación del voltaje conectando la entrada positiva del multímetro a la salida del acondicionador (pin 2) y el negativo a tierra, cuando logramos hacer variar el voltaje a través de la resistencia, conectamos el acondicionador a la entrada del microcontrolador y visualizamos la variación en el Display.

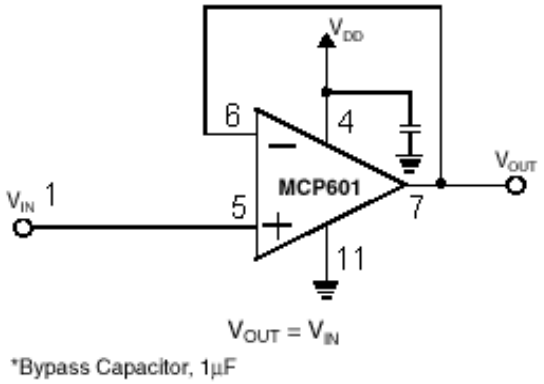


Figura 3.8 Conexión del primer acondicionador.

Como la señal no era lo suficientemente estable y las lecturas variaban muy seguido, conectamos la otra parte del acondicionador como filtro para eliminar el ruido que hacia variar el voltaje. Figura 3.9

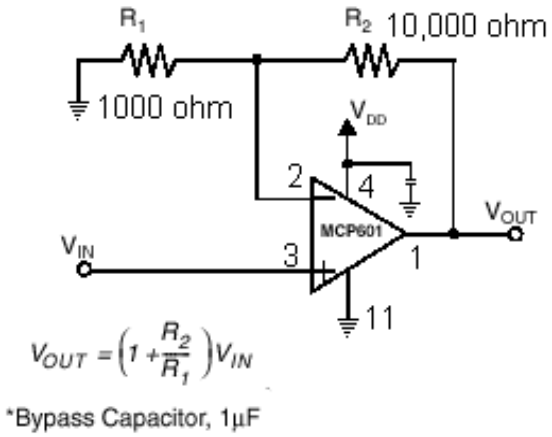


Figura 3.9 Conexión del segundo acondicionador.

Cuando logramos tener la señal estable requerida, conectamos el dinamómetro de anillos probado con anterioridad, que nos generaba la variación del voltaje cuando se le aplicaba una fuerza (tensión o compresión).

Para simular el funcionamiento del equipo montado al penetrómetro, de un extremo del dinamómetro lo colgamos a un tripie y del otro extremo le colgamos una canastilla aplicándole una fuerza de tensión y dejamos pasar 10 minutos para que se estabilizara la

señal, después utilizamos 5 pesos (los primeros 3 de 17 Kg, el cuarto de 13.5 Kg. y el quinto de 16 Kg), y se le agregó a la canastilla uno por uno dejando pasar 30 segundos entre la colocación de un peso y otro, colocados los 5 pesos dejamos reposar 1 minuto y realizamos la descarga para comparar los resultados desplegados durante el incremento. Esta prueba se repitió 3 veces. Figura 3.10.

Figura 3.10 Pruebas de simulación para obtener fuerza.

IV. RESULTADOS

Para un mejor entendimiento de los resultados obtenidos, se dividió el capitulo en dos secciones, el primero relacionado con el desplazamiento y el segundo con la fuerza y cada uno presenta la secuencia para llegar al software final, que incluye diagrama de flujo, circuito electrónico empleado, software y resultados de las pruebas.

4.1 Medidor de desplazamiento:

El sistema integral de sensores a base de pulsos y el módulo convertidor de un sistema binario a decimal a través de un microcontrolador 16F84, usando un display 2 líneas x 16 caracteres.

Los diagramas de flujos diseñados y un hardware y software para lograr que este sistema funcione son los siguientes

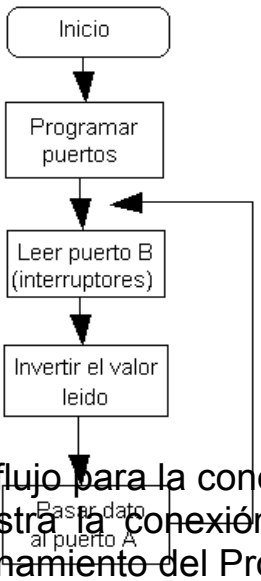


Figura 4.1 Diagrama de flujo para la conexión de los LED y Dipswitch.

La siguiente figura muestra la conexión de Led y Dipswitch con el PIC16F84, para el funcionamiento del Proyecto 1.

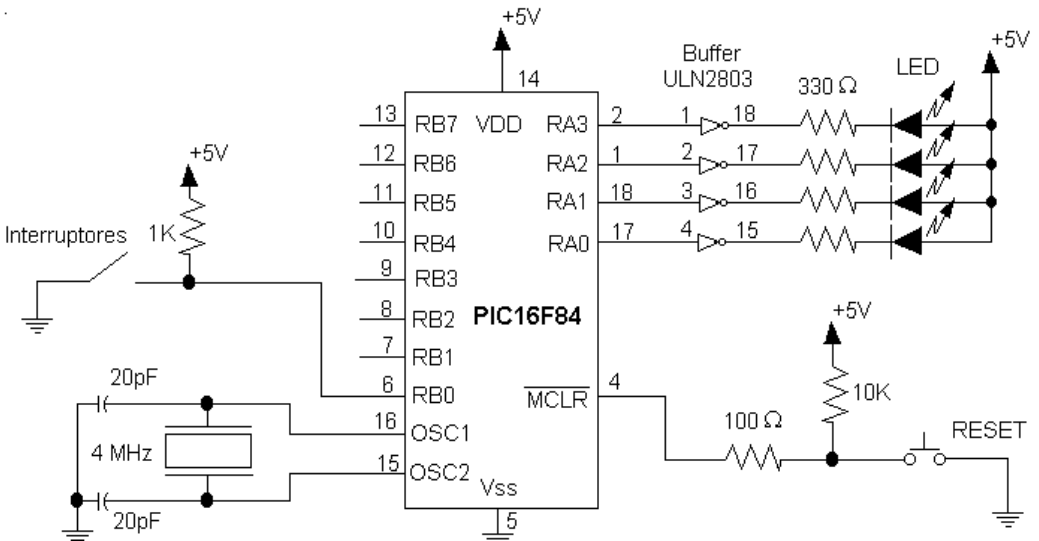


Figura 4.2 Diagrama para la conexión de los Leds.

En el Anexo III se muestra el programa respectivo, en donde se genera una interrupción externa introducida a través del pin RB0 del microcontrolador, la cual se encarga de encender o no los Leds. El hardware empleado para el encendido de los Leds se muestra en la siguiente figura, y funciona como un contador decimal del 1 al 9, al introducir el primer pulso se enciende el Led 1, después el 2,3,4 y las combinaciones de encendido de 4 y 1 generan el 5, 4 y 2 para el 6, 4 y 3 el 7, 4, 3, 1 el 8 y 4,3,2 el 9.

Figura 4.3 Hardware para empleado para visualizar el encendido de Leds.

2.- Manejo de un módulo LCD con conexión a 4 bits (Proyecto 2), el siguiente diagrama de flujo muestra la secuencia utilizada para encender el LCD.

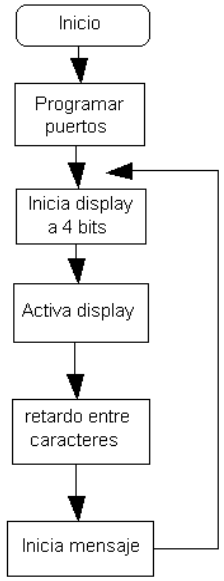


Figura 4.4 Diagrama l encendido del display a 4 bits.

El circuito electrónico usado para lograr este objetivo es el siguiente.

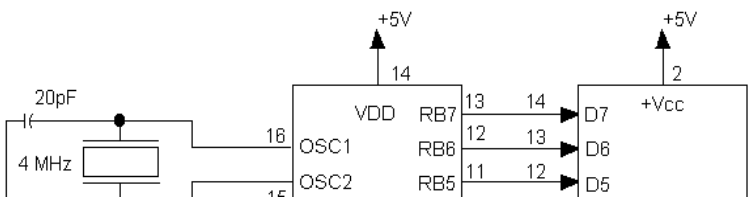


Figura 4.5 Conexión a 4 bits entre el microcontrolador y el módulo LCD.

La conexión se realizo usando un bus de datos de 4 bits, con los 4 pines de mayor peso del puerto B (RB4-RB7) del controlador y las señales de control con los dos pines de menor peso (RB0 y RB1).

El software implementado al microcontrolador, se encarga de generar las señales para mostrar el mensaje en el display, a través de la rutina de CONTROL. Dicho mensaje ocupa las dos líneas de la pantalla y se repite indefinidamente. El programa que realiza esta tarea se muestra en el Anexo III.

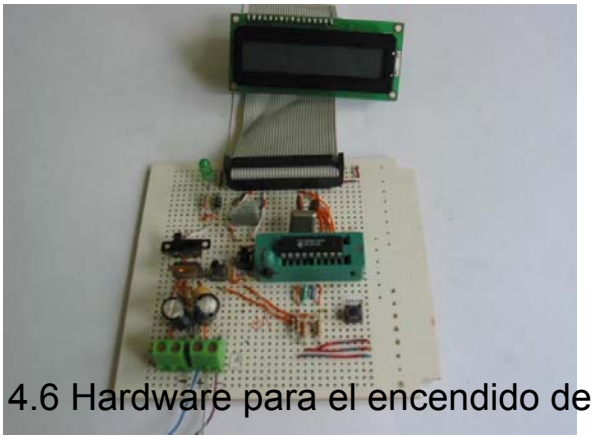


Figura 4.6 Hardware para el encendido de módulo LCD.

3.- Manejo de un LCD con los pines RA0 y RA1 para generar las señales de control (Proyecto 3), se genera a partir de la configuración de los puertos A y B por software. El diagrama de flujo se muestra en la Figura 4.4 y el circuito usado para realizar esta función en la Figura 4.7.

Por medio de la modificación del software se configuraron el pin RB0 como entrada y del (RB4 – RB7) se programaron como salidas

para conectarlos al LCD como bus de datos y por parte del puerto A los pines RA0 y RA1 fueron los que se configuraron como salidas para generar las señales de control. La modificación en el programa nos permitió dejar libre en el circuito al pin RB0, para usarlo como interrupción externa.

El hardware que presenta estas modificaciones y que se empleo como banco de pruebas de este microcontrolador se muestra en la Figura 4.8.

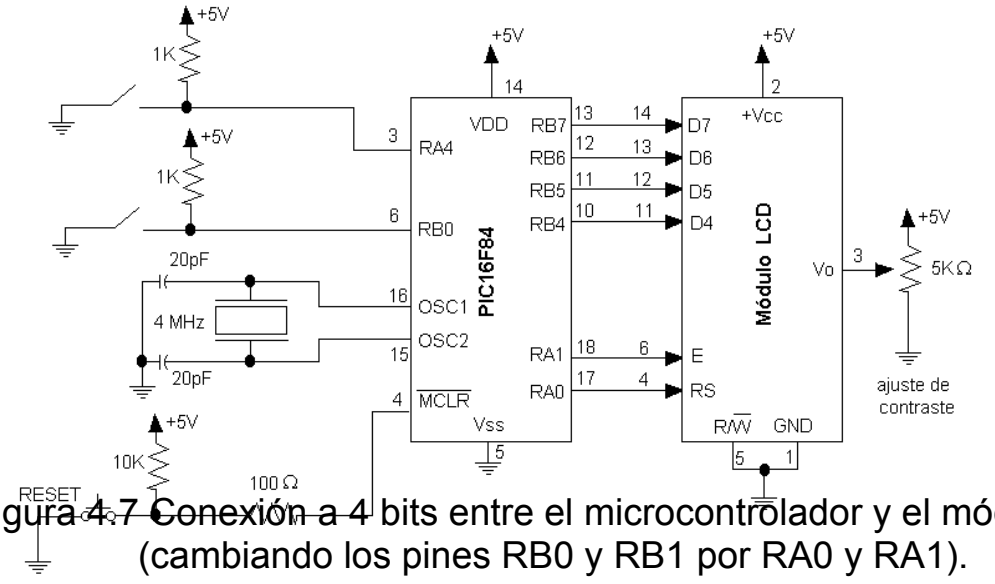


Figura 4.7 Conexión a 4 bits entre el microcontrolador y el módulo LCD (cambiando los pines RB0 y RB1 por RA0 y RA1).

El cuerpo principal del programa que hace funcionar este circuito se encuentra en el Anexo III.

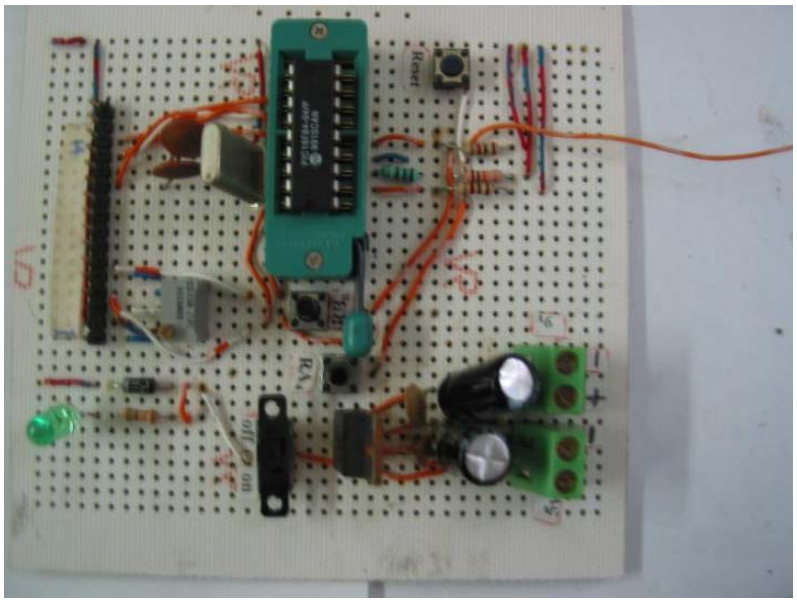


Figura 4.8 Banco de pruebas del PIC16F84.

4.- Para el manejo de un display de 7 segmentos el Proyecto 4, Anexo III, muestra la rutina de PULSA, necesaria para hacer un contador decimal a través del registro de pulsos, introducidos por un interruptor conectado al pin RB0.

5.- Programa para probar la Conexión de una interrupción externa (Proyecto 5) utilizando el circuito mostrado en la Figura 4.7, con un pulsador conectado al pin RB0 del microcontrolador, se genera una interrupción de señal que el PIC recibe y la interpreta, produciendo el despliegue de un digito en decimal.

En el Anexo III se encuentra registrado el programa que realiza el control de estas funciones.

Con este programa se registra la entrada de 10 pulsos y en ese momento despliega el numero “17” en pantalla que se encuentra en el registro W, manteniéndolo ahí indefinidamente.

Las pruebas que se realizaron donde se cambio el numero de entrada de pulsos, (en la rutina de pulsa), por 5 y 15 solo nos generaron la posibilidad de poder desplegar el numero “17” que se encuentra almacenado en el registro.

6.- Registro de 10 pulsos y despliegues en el LCD (Proyecto 6, Anexo III).
Este programa realiza el conteo de pulsos de un numero programado y lo muestra en la pantalla del módulo LCD.

Después de realizar las pruebas mencionadas en la metodología para comprobar el funcionamiento de este software se logro obtener el despliegue de un número guardado en la memoria del microcontrolador a través de la introducción de pulsos, por medio de la interrupción externa.

El circuito y el diagrama de flujo que nos indica los pasos a seguir en este proyecto son los mismos que se usaron en los apartados 3 y 5.
7.- Contador decimal de 255 pulsos desplegados en un Módulo LCD (Proyecto 7) Después de realizar todas las modificaciones

mencionadas en la metodología en los programas anteriores, obtuvimos como resultado final el programa que muestra en la pantalla del módulo LCD el despliegue de 255 pulsos en números decimales iniciando en 00 e incrementándose en 1 conforme se van introduciendo los pulsos. El circuito y el hardware empleados en este programa son los que aparecen en las Figuras 4.7 y 4.8.

El esquema que muestra el diagrama de flujo que sigue esta secuencia se muestra a continuación.

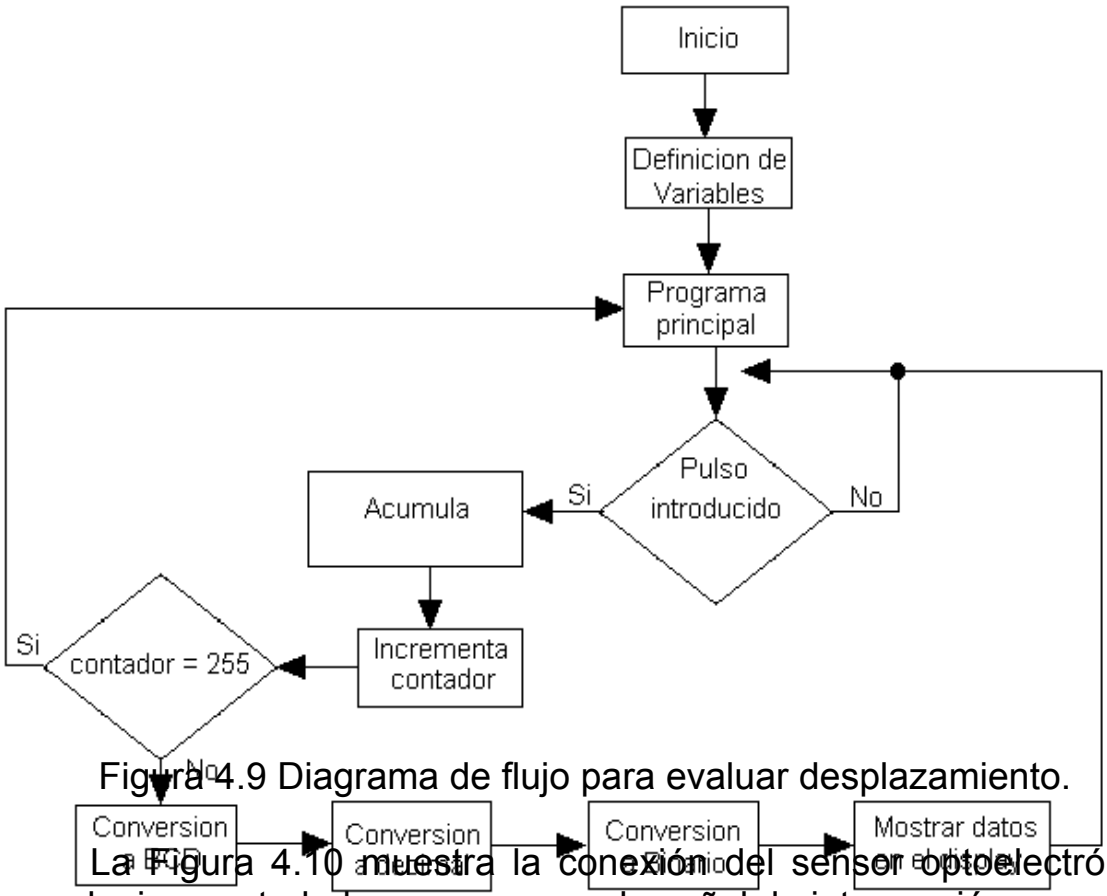


Figura 4.9 Diagrama de flujo para evaluar desplazamiento.

La Figura 4.10 muestra la conexión del sensor optoelectrónico con el microcontrolador para generar la señal de interrupción.

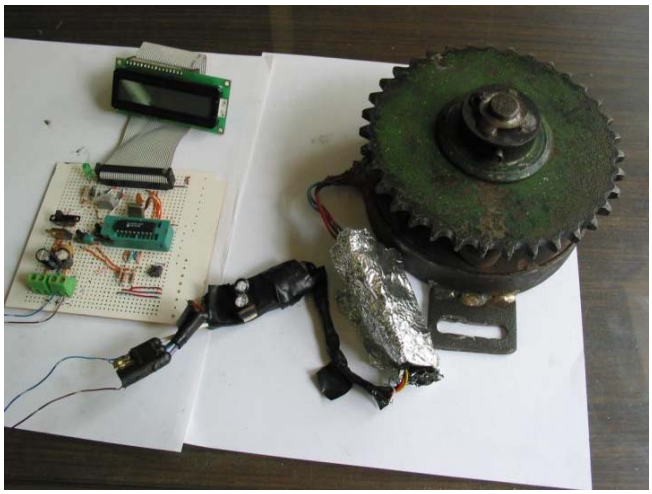


Figura 4.10 Conexión del sensor al microcontrolador.

Después de acoplarle el sensor al hardware del microcontrolador, se realizaron pruebas para la verificación del software, que fueron muy sencillas y consistieron en dar una vuelta al engrane y anotar la cantidad de pulsos generados, se realizaron 5 repeticiones de 3 vueltas cada una, para comparar si existía algún cambio en la generación de los pulsos desplegados por cada vuelta y mostrados en el Cuadro 4.1.

Cuadro 4.1 Pulsos enviados por el sensor de desplazamiento y registrados por
el software para 3 vueltas dadas al engrane.

Numero de vueltas	Numero de pulsos Repetición 1	Numero de pulsos Repetición 2	Numero de pulsos Repetición 3	Numero de pulsos Repetición 4	Numero de pulsos Repetición 5
1	20	20	21	18	20
2	40	39	41	38	42
3	63	59	61	59	62

4.2 Medidor de fuerza:

1.- Conexión de Leds y un Dipswitch (Proyecto 1.1, Anexo III) Para lograr hacer funcionar el microcontrolador se empleo la secuencia que muestra el diagrama de flujo y el circuito de la Figura 4.14, que muestra un montaje de conexión de un selector (Dipswitch) como generador de señal al microcontrolador y el encendido de 8 leds como salidas.

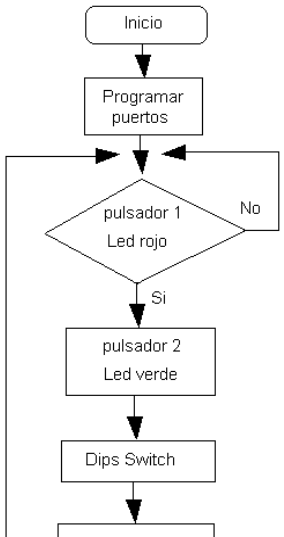


Figura 4.11 Diagrama de flujo para la conexión de los Leds y Selector (Dipswitch).

El diagrama muestra la secuencia para el encendiendo de 8 Leds a través de dos pulsadores, el primero habilita a un Led rojo y el segundo a uno verde que es el que genera la señal para que los 8 restantes se activen, dependiendo de los leds seleccionados por el Dipswitch que integra este circuito donde podemos seleccionar físicamente que Led queremos encendido ó apagado.

El hardware implementado en este proyecto se muestra en la Figura 4.13 el cual contiene la conexión de los dos pulsadores, el selector y los 8 Leds restantes.

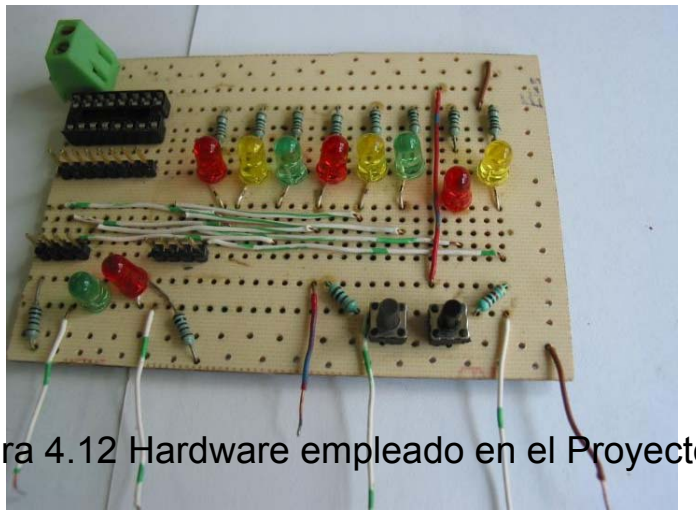


Figura 4.12 Hardware empleado en el Proyecto 1.

El circuito empleado para el desarrollo de este ejercicio es el siguiente.

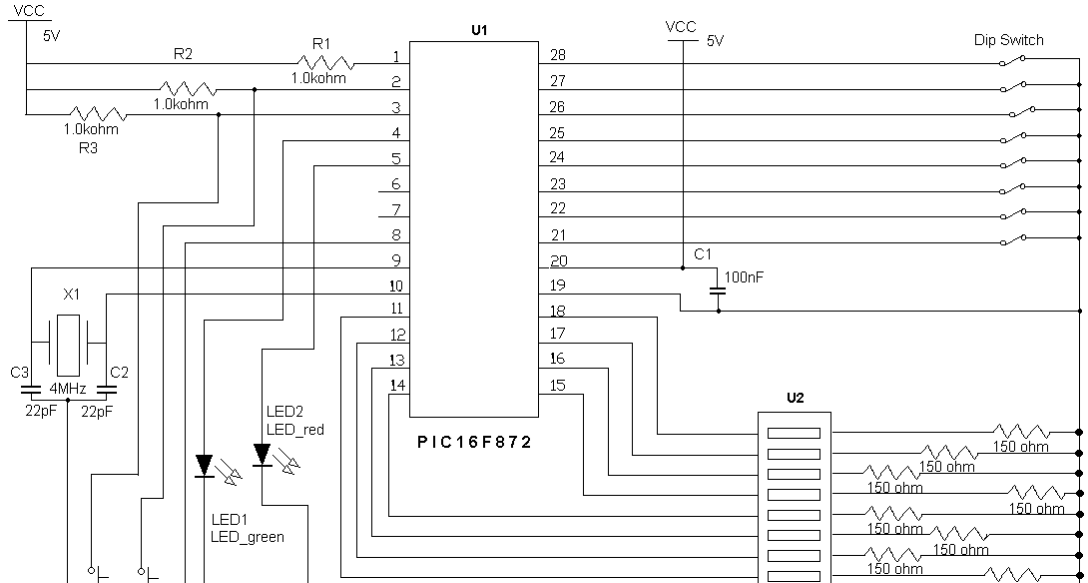


Figura 4.13 Conexión de los Leds y Dipswitch con el PIC16F872.

2.- Programa para la conexión a 8 bits (Proyecto 1.2). El diagrama muestra los pasos a seguir para generar el encendido de un LCD, conectado a 8 bits.

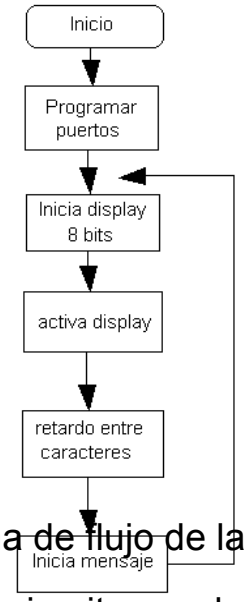


Figura 4.14 Diagrama de flujo de la conexión del módulo LCD.

Se empleo el siguiente circuito usado como banco para probar este microcontrolador generándose solo los hardware necesarios para cada programa.

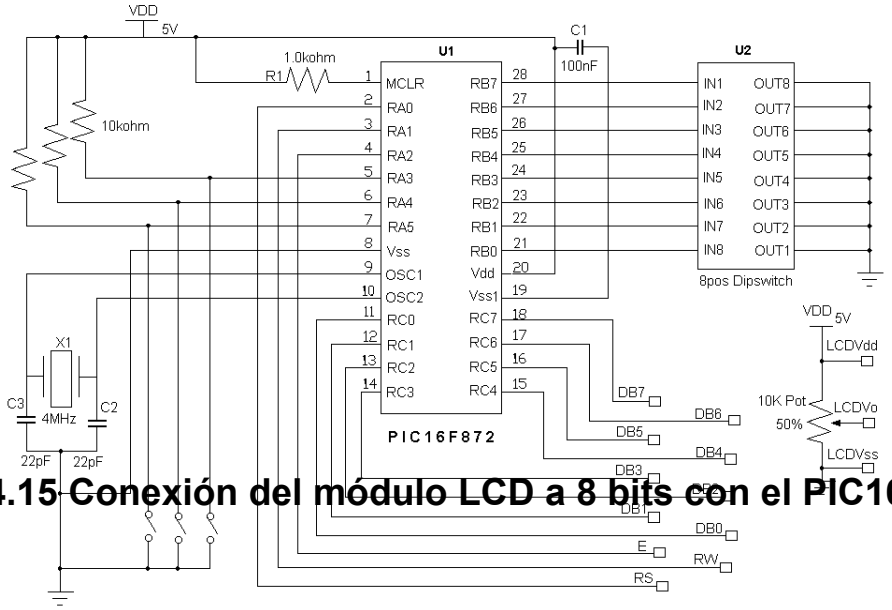


Figura 4.15 Conexión del módulo LCD a 8 bits con el PIC16F872

El circuito de la Figura 4.17 muestra el hardware con 3 pulsadores que conectado al microcontrolador generan una orden, un caracter o un retorno mostrado en el display, una vez que se activó el microcontrolador, utilizando el selector enviamos un número en binario al PIC, el cual realiza la conversión a decimal y despliega un caracter en pantalla.

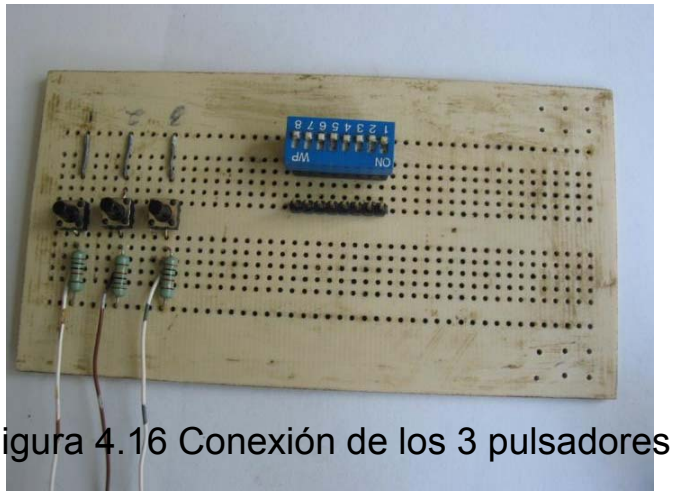


Figura 4.16 Conexión de los 3 pulsadores.

En el programa que muestra el funcionamiento de este circuito se encuentra Anexo III con lo que logramos obtener el encendido de un módulo LCD, conectado a 8 bits con el microcontrolador y generar el despliegue de caracteres a partir de la introducción de numeración binaria con el apoyo del selector.

3.- Conversión Analógico – Digital (Proyecto 1.3) Con el empleo de este ejercicio y basándonos en la secuencia que muestra el diagrama de flujo siguiente, se obtuvo la conversión de una señal analógica a digital mostrando un resultado en decimal en un módulo LCD.

En el Anexo III se localiza el cuerpo principal de este software.

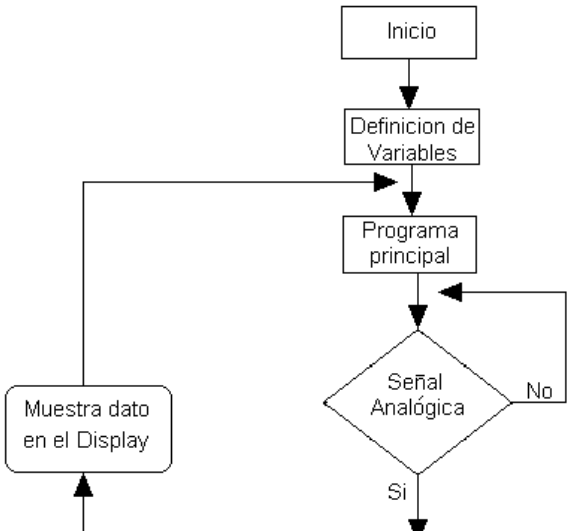


Figura 4.17 Diagrama de flujo para la conversión A/D.

El diagrama del circuito empleado se muestra en la Figura 4.19.

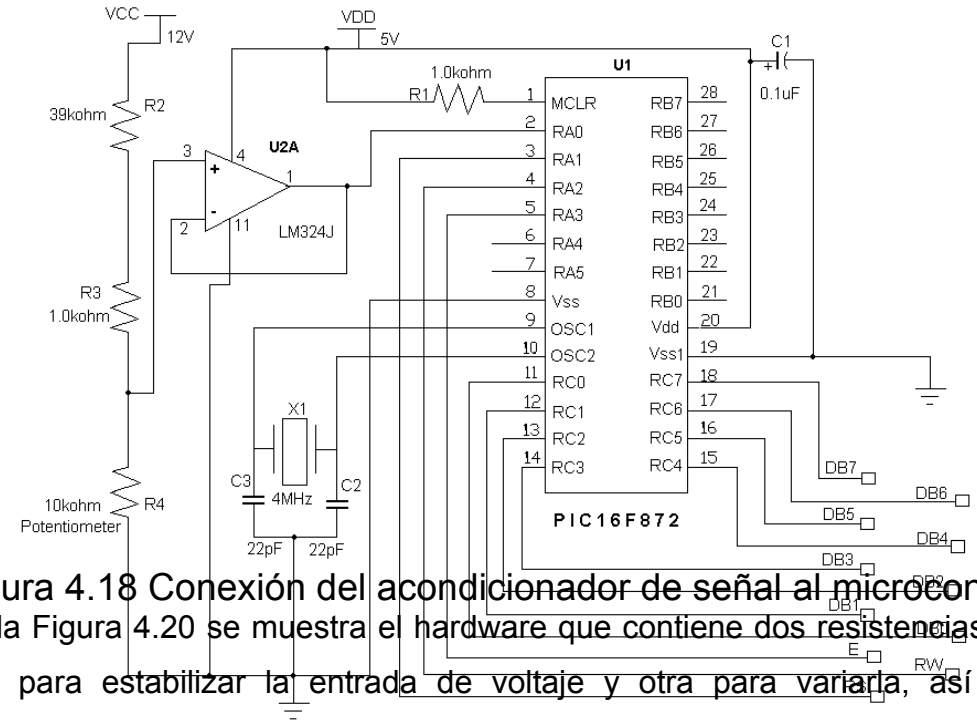


Figura 4.18 Conexión del acondicionador de señal al microcontrolador.

En la Figura 4.20 se muestra el hardware que contiene dos resistencias variables una para estabilizar la entrada de voltaje y otra para variarla, así como un amplificador para generar el acondicionamiento y la amplificación de la señal analógica.

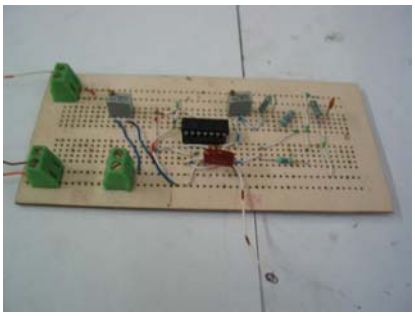


Figura 4.19 Conexión de los amplificadores.

4.3 Conexión del primer amplificador al PIC16F872

Con la conexión del primer amplificador obtuvimos una ganancia de 1-150 mV que nos generaba una diferencia de voltaje de aproximadamente 98 mV desplegado en el LCD.

En la Tabla siguiente se muestra la salida digital registrada por el sensor, al aplicarle diferentes cargas de pesos conocidos.

Cuadro 4.2. Resultados de la conexión de un amplificador.

Voltaje Inicial de referencia 490 (mV)						
Carga aplicada (Kg)	Fase 1 (mV)		Fase 2 (mV)		Fase 3 (mV)	
(para)	Carga	Descarga	Carga	Descarga	Carga	X. Descarga
17	686	588	588	588	588	588
34	785	784	686	686	686	686
51	882	882	882	882	784	784
65	980	980	980	980	882	882
80	1078	1078	1078	1078	980	980

La grafica de la Figura 4.22 muestra el voltaje enviado por el amplificador al PIC y la salida digital del microcontrolador visualizada en el LCD, (mV).

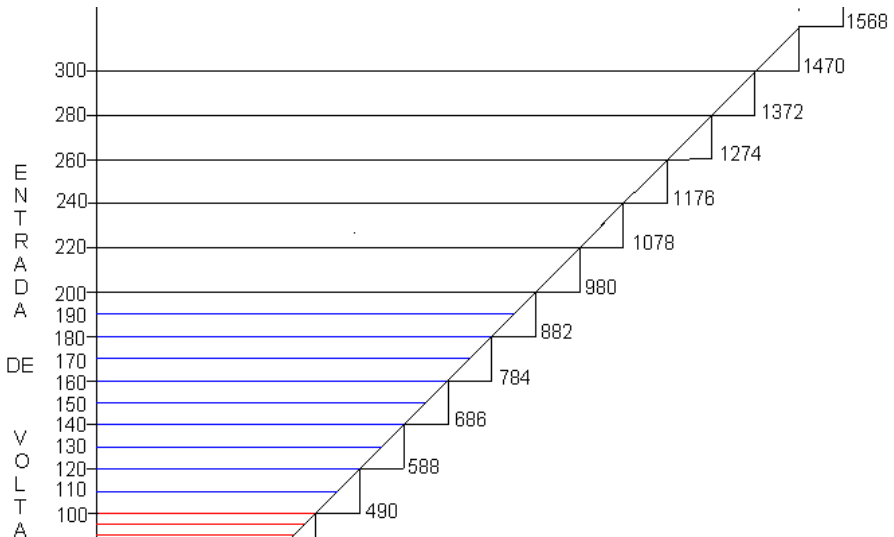


Figura 4.20 Grafica de la conversión Analógico a Digital.

Como se aprecia en la figura de la grafica anterior por cada 20 mV de incremento que envía el amplificador al microcontrolador, se obtiene una salida digital en incremento de 98 mV,

La entrada de voltaje que recibe el acondicionador es variada por la resistencia de 50 KΩ y de acuerdo a la ganancia obtenida por cada mV de entrada tendremos 150 mV de salida.

4.4 Conexión del segundo amplificador al PIC16F872

Con la conexión del segundo amplificador obtuvimos una ganancia de 1-250 mV que nos generaba una diferencia de voltaje de aproximadamente 196 mV desplegado en el LCD.

En la tabla siguiente se muestra la salida digital registrada por el sensor, al aplicarle diferentes cargas de pesos conocidos.

Cuadro 4.3 Resultados de la conexión de dos amplificadores.

Voltaje Inicial de referencia 490 (mV)

Carga aplicada (Kg)	Fase 1 (mV)		Fase 2 (mV)		Fase 3 (mV)	
(para)	Carga	Descarga	Carga	Descarga	Carga	Xl. Descarga
17	686	686	686	686	784	686
34	980	980	882	882	980	980
51	1176	1176	1176	1176	1176	1176
65	1372	1372	1372	1372	1132	1372
80	1666	1666	1568	1568	1666	1666

V. CONCLUSIONES Y RECOMENDACIONES

Mis objetivos eran diseñar un sistema con sensores y microcontroladores para la obtención de una salida digital para medir fuerza y desplazamiento.

1. Se desarrollaron el Hardware y el Software apropiados para medir fuerzas y desplazamiento, sin embargo no presentan la sensibilidad esperada.

2. Para el medidor de desplazamiento los sensores optoelectrónicos son los más apropiados, en virtud de mostrarnos una alta resolución, en su desempeño.

3. El sensor de fuerza tipo anillo evaluado, el cual se encuentra integrado por galgas extensiométricas, nos proporciona la sensibilidad, que nos permite evaluar una carga aproximada de 80 Kg.

4. Para alcanzar la sensibilidad que se requiere para el desplazamiento, se recomienda realizar modificaciones al Software en el tiempo de retardo (time delay), para lograr que detecte solo 20 pulsos en cada vuelta proporcionadas al engrane.

5. En virtud de que el programa evaluado para medir la fuerza, tiene un rango de evaluación de incremento de 20 mV, se requiere reducir el escaneo de la señal a 1 mV.

6. Debido a los problemas presentados en la medición de la señal enviada por el sensor de fuerza tipo anillo se recomienda realizar la conexión mostrada en la Figura 2.6.

VI. LITERATURA CITADA

Angulo U. J, I. A. Martínez (1999). Microcontroladores PIC, diseño practico de aplicaciones segunda edición, McGraw-Hill Interamericana de España, S.A.U.

Cooper, William D.; Helfrick, Albert D. (1991). Instrumentación electrónica moderna y técnicas de medición. México, Prentice hall Hispanoamericana, S.A.

Cañavate, J. O. (1989). Técnica de Mecanización Agraria, Madrid España, Mundi-Prensa.

Edison Duque C. (1998). Curso avanzado de Microcontroladores PIC, CEKIT (Compañía Editorial Tecnológica).

Flamand Rodríguez C. L. (1995). Introducción a la Mecánica de Suelos, Universidad Autónoma de Chapingo, Dirección General de Difusión Cultural, Dpto. de Publicaciones.

Graciano Dieck Assad. (2000). Instrumentación acondicionamiento electrónico y adquisición de datos, México, Editorial Trillas.

Hinojosa G. M. (2003). Desarrollo de un transductor de fuerza para la evaluación de implementos integrales en tractores categoría II basado en el diseño octagonal extendido, Tesis licenciatura UAAAN, Buenavista Saltillo, Coah. México.

Mojica, E.D. (2000). Evaluación de discos cortadores de residuos y abresurcos para labranza de conservación empleando equipos multiusos de tracción animal, Tesis de Licenciatura, Instituto Tecnológico de Veracruz.

Moo Y. V. M. J. (1999). Desarrollo y validación de un sistema de adquisición de datos para evaluación de implementos de labranza, Tesis maestría Universidad Veracruzana, H. Veracruz, Ver.

Stanley Wolf, Richard F.M., Smith. (1992). Guia para mediciones electrónicas y practicas de laboratorio, México, Prentice hall Hispanoamericana, S.A.

http://www.suelos.org.ar/PonenciaJorajuria.pdf

Daniel Jorajuria Collazo, Prof. Titular Departamento de Ingeniería Agrícola y Forestal; Facultad de Ciencias Agrarias y Forestales–UNLP. Penetrometría como parámetro mecánico del suelo. E-mail: dajo@ceres.agro.unlp.edu.ar

http://www.suelos.org.ar/penetrometria%20Cecilia%20Cerisola.pdf

Dra. Cecilia Cerisola. Ventajas de utilizar la Penetrometría en sencillos modelos linealizables para conocer la densidad aparente seca de un suelo. Manejo y Conservación de Suelos. Dpto. Ambiente y Recursos Naturales; Fac. Cs. Agr. Y Ftales. U.N.L.P.

http://www.chapingo.mx/terra/contenido/16/4/art303-307.pdf

De León González, F.1, Payán Zelaya, F. y S. Sánchez R. Localización de las capas compactadas en el perfil del suelo mediante la penetrometría. Identification of soil Compacted Layers Using a Cone Digital Penetrometer.

http://www.micropic.arrakis.es/marcos.htm

http://zeus.uam.mx/labre/microcontroladores.htm

www. electronicaestudio.com/articulos (**ter872.zip**)

www. agelectronica.com

ANEXO I

1.1 Características del PIC16F84

a) *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 1.1.1. Cada pin se puede configurar como entrada o como salida independiente programando un par de registros diseñados para tal fin. En ese registro un “0” configura el pin del puerto correspondiente como salida y un “1” lo configura como entrada.

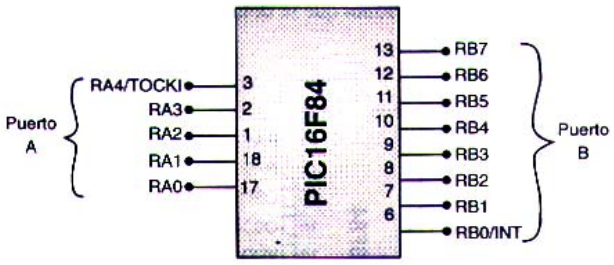


Figura 1.1.1 Puertos del PIC16F84.

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 la memoria RAM) llamado OPTION. La resistencia de pull-up es desconectada automáticamente en un pin si este se programa como salida. El pin RB0/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 o como entrada 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 externa conectada a un nivel de cinco voltios). Como salida, la lógica es inversa: un “0” escrito al pin del puerto entrega en el pin un “1” lógico.

Todos los pines deben estar conectados a alguna parte, nunca dejarlos al aire por que 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 1.1.2.

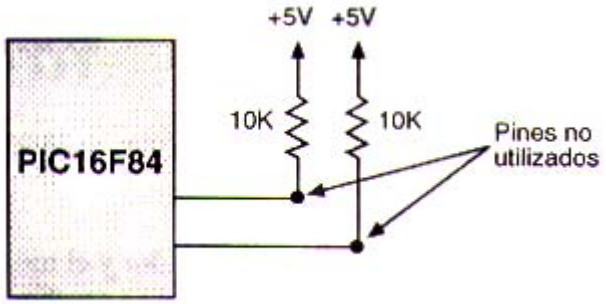


Figura 1.1.2 Los puertos no utilizados se deben conectar a la fuente.

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 en consumo de potencia). Se sale de ese estado cuando se produce alguna condición especial.

- El oscilador externo

Todo microcontrolador requiere un circuito externo que le indique la velocidad a la que debe trabajar, se les conoce como oscilador o reloj, es muy simple pero de vital importancia para el buen funcionamiento del sistema. El PIC16F84 puede utilizar 4 tipos de osciladores 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 los “fusibles de configuración”.

El tipo de oscilador que se sugiere es el de 4 MHz, por que 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 14 MHz, por lo que cada instrucción se ejecuta en un microsegundo. El cristal se conecta como se muestra en la Figura 1.1.3.

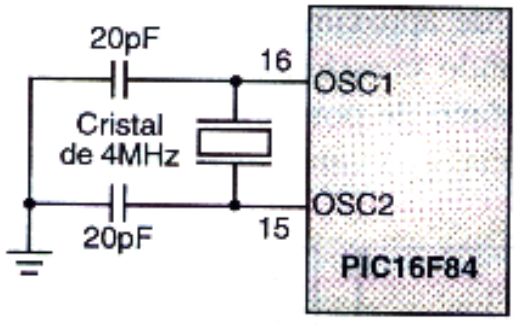


Figura 1.1.3 Conexión de un oscilador de cristal.

Dependiendo de la aplicación, se puede utilizar cristales de otras frecuencias; por ejemplo el de 3.579545 MHz por que 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 requiere economizar dinero, se puede utilizar una resistencia y un condensador, como se muestra en la Figura 1.1.4.

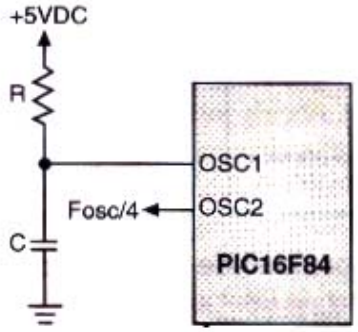


Figura 1.1.4 Conexión de un oscilador RC.

- 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 por que 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 para el encendido se consigue con dos temporizadores. El primero de es el OST (Oscillator Star-Up Timer: Temporizador de encendido del oscilador), orientado a mantener el microcontrolador en reset hasta que el oscilador del cristal

es estable. El segundo es el PWRT (Power-Up 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.

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 0FFh a 00h. Cuando se quiere tener control sobre el reset del sistema se puede conectar un botón como se muestra en la Figura 1.1.5.

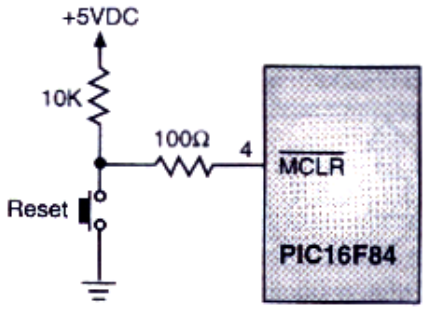


Figura 1.1.5 Conexión del botón de reset.

d) 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 1.1.6 muestra la arquitectura general del PIC16F84, en ella se pueden apreciar los diferentes bloques que lo componen y la forma en que se conectan.

Todos los elementos se conectan entre sí por medio de 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.

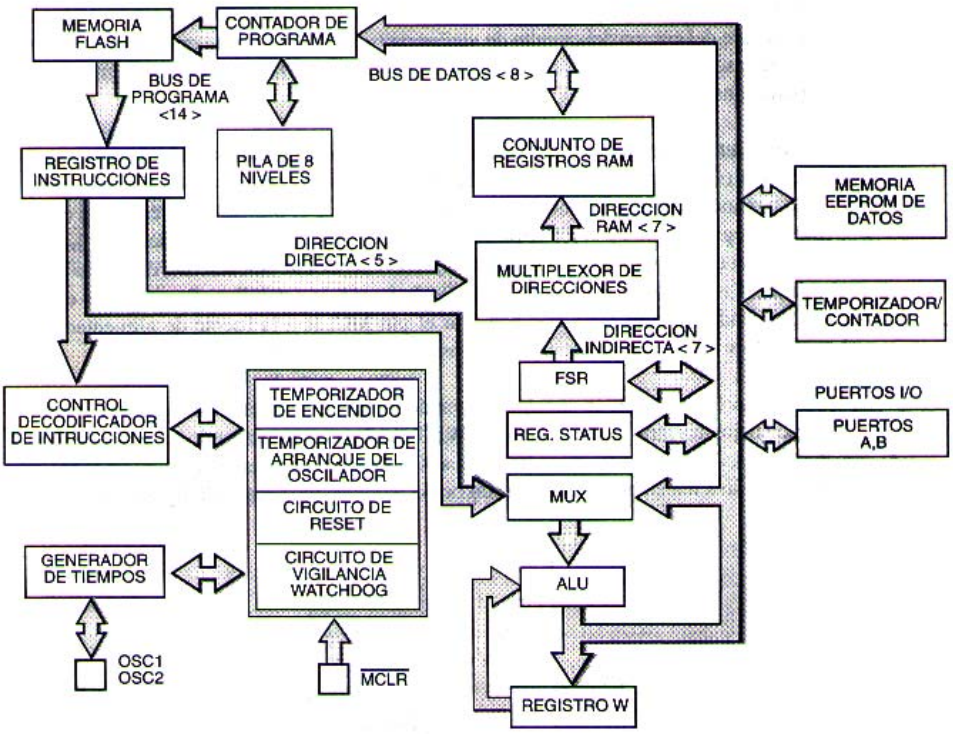


Figura 1.1.6 Arquitectura interna del PIC16F84.

- *Memoria de programa*

Es una memoria de 1 Kbyte de longitud con palabras de 14 bits. En ella se graba, el programa que el microcontrolador debe ejecutar. El PIC16F84 tiene un contador de programa de 13 bits, y 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 mayores a 3FF se causará un solapamiento con el espacio del primer 1K. En la Figura 1.1.7 se muestra el mapa de la memoria de programa.

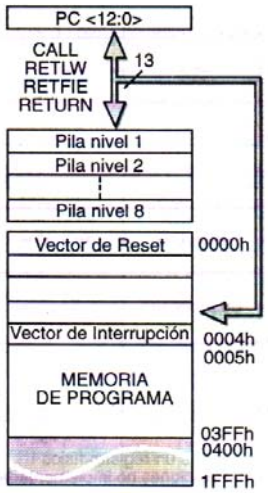


Figura 1.1.7 Mapa de la memoria de programa.

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.

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 (0-4F 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 1.1.8.

Los registros están organizados como dos arreglos (páginas) de 128 posiciones de 8 bits cada una (128 x 8); todas las posiciones se pueden acceder

directa o indirectamente (esta última a través del registro selector FSR). Para seleccionar que página de registro se trabaja en un momento determinado se utiliza el bit RP0 del registro STATUS.

- *Descripción de los registros.*

00h o INDO: Registro para direccionamiento indirecto de datos. Este no es un registro disponible físicamente; utiliza el contenido del FSR y el bit RP0 del registro STATUS para seleccionar indirectamente la memoria de datos o RAM del usuario; la instrucción determinara que se debe realizar con el registro señalado.

01h o TMR0. Temporizador/contador de 8 bits. Este se puede incrementar con una señal externa 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 de puede determinar por medio de un preescalador, localizado en el registro OPTION. Como una mejora, se le ha agregado la generación de interrupción cuando se rebasa la cuenta (el paso de 0FFh a 00h).

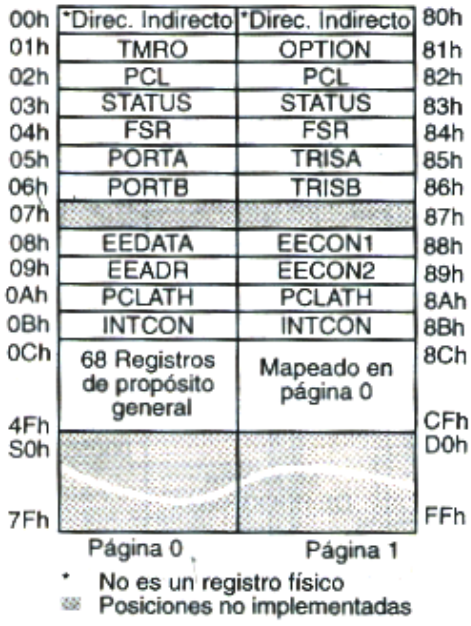


Figura 1.1.8 Registros del PIC16F84.

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 programa es de 13 bits de ancho, Figura 1.1.9. 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 (0Ah). 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.

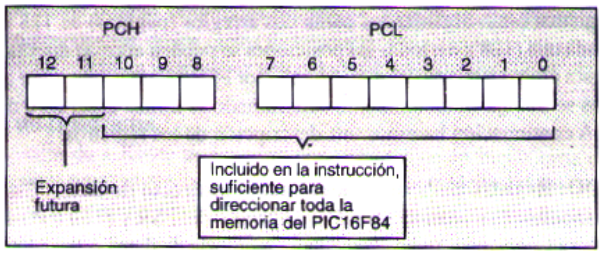


Figura 1.1.9 Contador de programa de (13 bits).

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. Figura 1.1.10. La selección de página para el direccionamiento directo de la memoria de datos; lo haremos con RP0 solo en el PIC16F84. RP1 se utilizara como bit de propósito general. Los bits T0 y PD no se pueden modificar por un proceso de escritura; ellos muestran la condición por la cual se ocasionó el último reset.

Registro: STATUS

IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C

bit 7 bit 0

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 puede utilizar como un *bit* de propósito general.
RP1,0: Selectores de página para direccionamiento directo. Solamente RP0 se utiliza en el PIC16F84. RP1 se puede utilizar como un *bit* de propósito general.
$\overline{TO}$: *Time Out* o *Bit* de finalización del temporizador. Se coloca en 0 cuando el circuito de vigilancia *Watchdog* finaliza la temporización.
$\overline{PD}$: *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 una operación lógica 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 acarrreo. En instrucciones aritméticas se activa cuando se presenta acarreo desde el *bit* más significativo del resultado.

Figura 1.1.10 Registro de Estados.

04h o FSR: Registro selector de registros. En conjunto con el registro INDO, se utiliza para seleccionar indirectamente los otros registros disponibles. Si en el programa no se utilizan llamadas indirectas, este registro se puede utilizar como un registro de propósito general.

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/Salida), se localiza 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 puerto puede leerse o escribirse como si se tratara de un registro cualquiera; algunos de sus pines 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.

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.

0Ah 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.

0Bh o INTCON: Registro para el control de interrupciones. Es el encargado del manejo de las interrupciones, los de este se muestran en la siguiente figura.

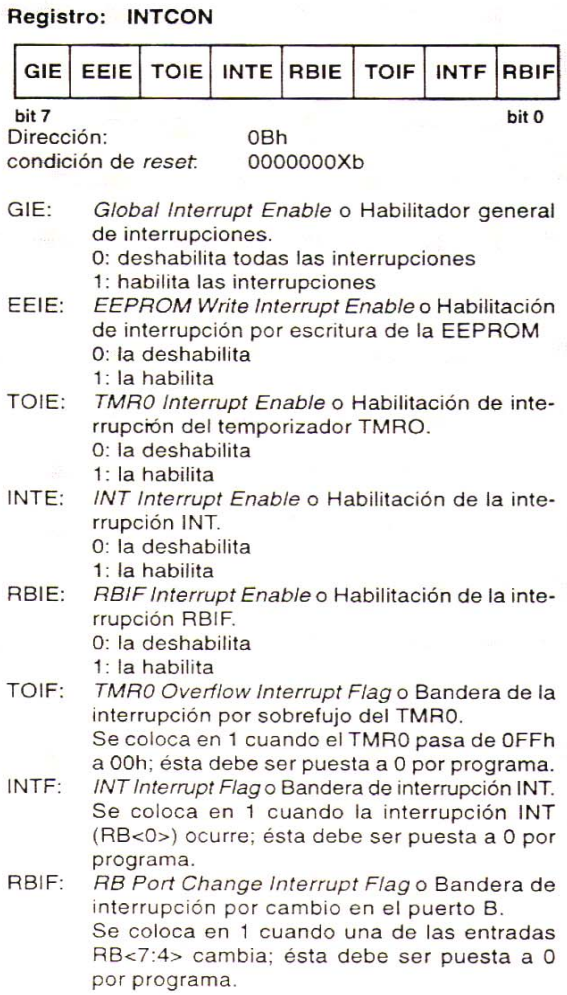


Figura 1.1.11 Registro INTCON.

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 sus funciones se muestran en la Figura 1.1.12.

85h o TRISA: Registro de configuración del puerto A. Es el registro de control del puerto 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.

Registro: OPTION

$\overline{\text{RBPU}}$	INTEDG	GRTS	RTE	PSA	PS2	PS1	PSO
bit 7							bit 0

Dirección: 81h
condición de *reset*: 11111111b

$\overline{\text{RBPU}}$: *PortB Pull-up Enable* o Habilitación 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: *TMR0 Signal Source* o Fuente de señal de TMR0.
0: ciclo de instrucciones interno (Temporizador)
1: transición en el pin RA4/TOCK (Contador)

RTE: *TMR0 Signal Edge* o Flanco de la señal TMR0
0: incremento en transición de bajo a alto
1: incremento en transición de alto a bajo

PSA: *Prescaler Assignment* o Asignación del preescalador
0: TRM0 (Contador/Temporizador)
1: WDT (Circuito de vigilancia)

PS2,1,0: *Prescaler Value* o Valores del preescalador.

Valor	TMR0	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

Figura 1.1.12 Registro OPTION.

88h o EECON1: Registro para el control de la memoria 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 siguiente figura se muestran las funciones de estos bits.

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 *bits* 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 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 sólo se limpia (se coloca en '0") por *hardware*, al finalizar la lectura de la posición de la EEPROM.

Figura 1.1.13 Registro EECON1.

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".

0ch 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.

***Registro de trabajo (W)*.** Este es el registro de trabajo principal, se comporta de manera similar al acumulador en los microprocesadores. Este registro participa en la mayoría de las instrucciones.

Pila (Snack). Estos registros no forman parte de ningún banco de memoria y no permiten el acceso por parte del usuario. Se usan para guardar el valor del contador de programa cuando se hace un llamado a una subrutina o cuando se atiende una interrupción. El PIC16F84 tiene una pila de 8 niveles, lo que significa que pueden anidar 8 llamados a subrutina sin tener problemas.

e) Características especiales

- *Circuito de vigilancia (Watchdog Timer)*

Su función es reestablecer el programa cuando éste se ha 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 un oscilador RC que se encuentra dentro del microcontrolador.

- *Temporizador de encendido (Power-up Timer)*

Proporciona un reset al micro en el momento de conectarlo a la fuente, lo que garantiza un arranque correcto del sistema. Al momento de grabar el micro se debe habilitar este fusible, Su tiempo de retardo es de 72 milisegundos.

- *Modo de bajo consumo (Sleep)*

Esta característica permite que el microcontrolador entre en un estado pasivo donde consume muy poca potencia. Aquí el oscilador principal se detiene, pero el temporizador del circuito de vigilancia se reinicia y empieza su conteo nuevamente.

- Interrupciones

Este microcontrolador incluye el manejo de interrupciones, lo cual representa grandes ventajas. El PIC16F84 posee 4 fuentes de interrupción a saber.

*Interrupción externa***.** Actúa sobre el pin RBO/INT y se puede configurar para activarse con el flanco de subida. La interrupción se puede deshabilitar colocando el bit de control INTE (INTE<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 (0FFh) en el TMRO se colocará el bit TOIF en uno (INTCON<2>). El bit de control respectivo es TOIE (INTCON<3>).

*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 (INTCON<6>).

Con el bit GIE (Global Interrupt Enable) en cero, se deshabilitan todas las interrupciones y con la instrucción RETFIE permite al usuario retornar de la interrupción, a la vez que las habilita de nuevo las interrupciones, al colocar GIE en uno.

- Memoria de datos EEPROM

El PIC16F84 tiene una memoria EEPROM de datos de 64 posiciones (0h a 3Fh), de 8 bits cada una. Este bloque de memoria no se encuentra mapeado 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.

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.

- Fusibles de configuración

El PIC16F84 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 de 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.

- Las pull-ups internas

Cada uno de los pines del puerto B tienen un débil elemento pull-up interno (250 μA 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 y uno o más operandos sobre los que se actúa. En la Tabla 2.1.1 del Anexo II, se encuentra la lista completa de instrucciones que incluye ejemplo y explicaciones, en total son 35 y tardan un ciclo de maquina, a excepción de los saltos que toman dos ciclos.

1.2 Características del PIC16C71

c) Registros (Memoria RAM)

5h 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 y, a demás de comportarse como entradas y salidas digitales, tienen funciones alternas, que se muestran en el Cuadro 1.2.1.

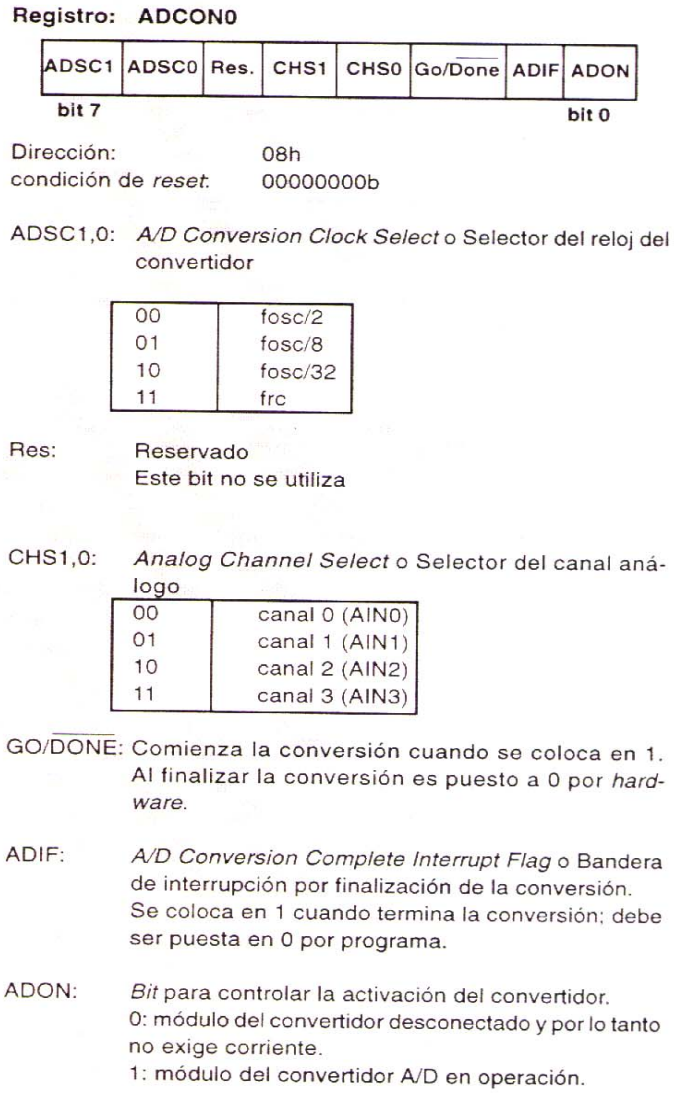
Cuadro 1.2.1 Funciones alternas de los pines del puerto A.

Pin	Función alterna
RA0 / AIN0	Entrada Análoga Canal 0
RA1 / AIN1	Entrada Análoga Canal 1
RA2 / AIN2	Entrada Análoga Canal 2
RA3 / AIN3 / Vref	Entrada Análoga Canal 3 o Voltaje de referencia (Vref)
RA4 / TOCK1	Entrada Externa de reloj para contador TMRO

Dos bits en el registro ADCON1 (registro 86h) configuran los cuatro pines menos significativos como E/S digitales o entradas análogas. Bajo la condición de

reset, estos pines se configuran como entradas análogas. El registro de control de este puerto (TRISA) está localizado en la página 1, en la posición 85h.

08h o ADCON0: Registro de control del convertidor. Este registro, junto con ADCON1 se dedica al control del convertidor analógico a digital. Los bits que éste contiene y la función que cumplen se muestran en la siguiente figura.



Registro: ADCON0

ADSC1	ADSC0	Res.	CHS1	CHS0	Go/Done	ADIF	ADON

bit 7 bit 0

Dirección: 08h
condición de *reset*: 00000000b

ADSC1,0: *A/D Conversion Clock Select* o Selector del reloj del convertidor

00	fosc/2
01	fosc/8
10	fosc/32
11	frc

Res: Reservado
Este bit no se utiliza

CHS1,0: *Analog Channel Select* o Selector del canal análogo

00	canal 0 (AIN0)
01	canal 1 (AIN1)
10	canal 2 (AIN2)
11	canal 3 (AIN3)

GO/DONE: Comienza la conversión cuando se coloca en 1. Al finalizar la conversión es puesto a 0 por *hardware*.

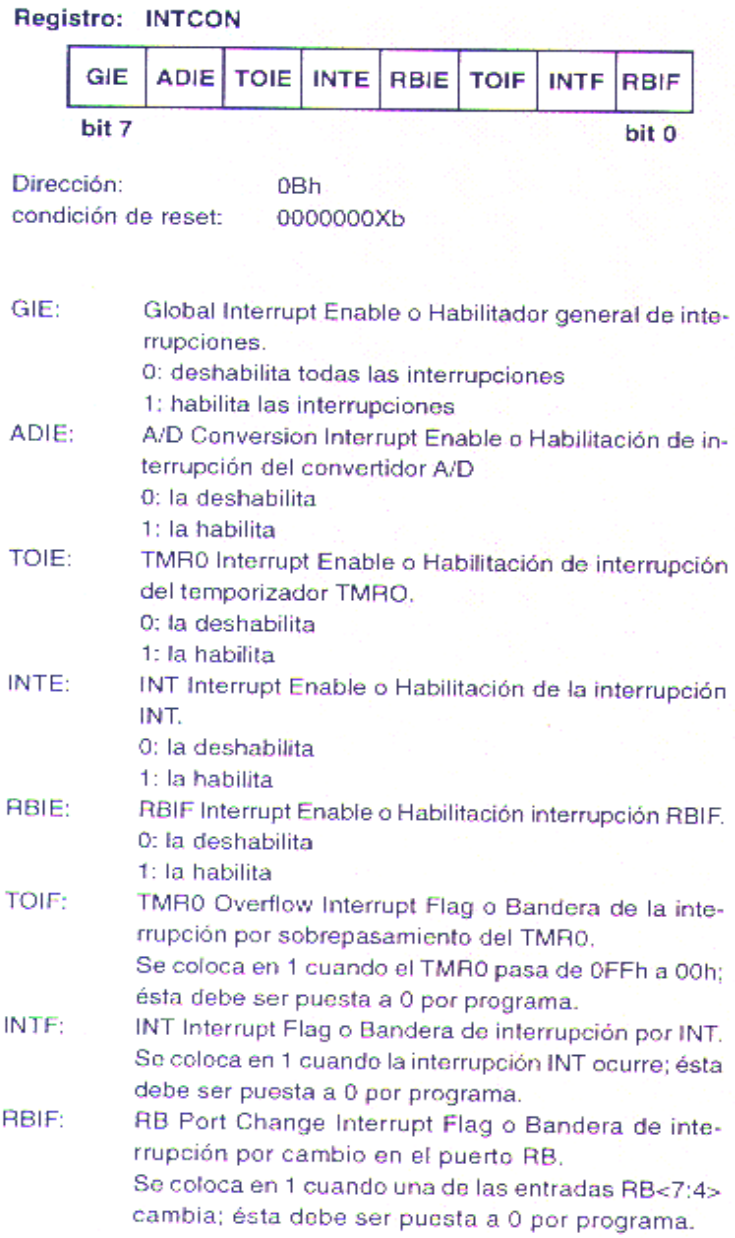
ADIF: *A/D Conversion Complete Interrupt Flag* o Bandera de interrupción por finalización de la conversión. Se coloca en 1 cuando termina la conversión; debe ser puesta en 0 por programa.

ADON: *Bit* para controlar la activación del convertidor.
0: módulo del convertidor desconectado y por lo tanto no exige corriente.
1: módulo del convertidor A/D en operación.

Figura 1.2.2 Registro ADCON0.

09h o ADRES: Registro para el resultado de la conversión. En esta posición se almacena el resultado digital de la conversión del valor análogo.

0Bh 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 Figura 1.2.3 . Contiene el bit ADIE para habilitar la interrupción del A/D.



Registro: INTCON

GIE	ADIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

bit 7 bit 0

Dirección: 0Bh
condición de reset: 0000000Xb

GIE: Global Interrupt Enable o Habilitador general de interrupciones.
0: deshabilita todas las interrupciones
1: habilita las interrupciones

ADIE: A/D Conversion Interrupt Enable o Habilitación de interrupción del convertidor A/D
0: la deshabilita
1: la habilita

TOIE: TMR0 Interrupt Enable o Habilitación de interrupción del temporizador TMR0.
0: la deshabilita
1: la habilita

INTE: INT Interrupt Enable o Habilitación de la interrupción INT.
0: la deshabilita
1: la habilita

RBIE: RBIF Interrupt Enable o Habilitación interrupción RBIF.
0: la deshabilita
1: la habilita

TOIF: TMR0 Overflow Interrupt Flag o Bandera de la interrupción por sobrepasamiento del TMR0.
Se coloca en 1 cuando el TMR0 pasa de 0FFh a 00h; ésta debe ser puesta a 0 por programa.

INTF: INT Interrupt Flag o Bandera de interrupción por INT.
Se coloca en 1 cuando la interrupción INT ocurre; ésta debe ser puesta a 0 por programa.

RBIF: RB Port Change Interrupt Flag o Bandera de interrupción por cambio en el puerto RB.
Se coloca en 1 cuando una de las entradas RB<7:4> cambia; ésta debe ser puesta a 0 por programa.

Figura 1.2.3 Registro INTCON.

88h o ADCON1: Registro auxiliar para el control del convertidor. Se dedica a la configuración de las entradas análogas y solo destina dos bits para ello; el resto

de bits permanece sin implementar. En la Figura 1.2.4, se muestran las funciones de estos bits.

Registro: ADCON1

U	U	U	U	U	U	PCFG1	PCFGO

bit 7 **bit 0**

Dirección: 88h
condición de *reset*: 00000000b

U: *Unimplemented* No implementadas.
Estos *bits* se leen como ceros.
PCFG1,0: Configuración de los pines RA0-RA3

PCFG1,0	RA0,RA1	RA2	RA3	Vref
00	E. análogas	E. análoga	E. análoga	Vdd
01	E. análogas	E. análoga	Referencia	RA3
10	E. análogas	E/S digital	E/S digital	Vdd
11	E/S digitales	E/S digital	E/S digital	Vdd

Figura 1.2.4 Registro ADCON1.

d) Características especiales

- Convertidor Análogo a Digital

El módulo del convertidor posee cuatro entradas análogas multiplexadas a un solo circuito de muestreo y sostenimiento y a un convertidor. El voltaje de referencia del convertidor se puede configurar por software para que sea el voltaje de alimentación del microcontrolador (VDD) o un voltaje externo aplicado al pin RA3, el valor mínimo que puede tomar el voltaje de referencia es de 3 voltios. El tiempo de conversión está en función del tipo de oscilador, considerándose un tiempo mínimo de 20 μs.

La conversión se inicia colocando el bit de control GO/DONE (ADCON0<2>) en “uno”, previa selección del canal deseado y de configurar el registro ADCON1, el cual define los modos en los cuales trabaja el convertidor. Al final de la conversión, el bit GO/DONE se coloca en “cero” y la bandera de interrupción ADIF del convertidor (ADCON0<1>) en “uno”, almacenándose el resultado de la

conversión en el registro ADRES, lo que permite que el resultado de la conversión se pueda leer por medio de interrupciones o sin ellas.

Esquema del reloj del convertidor El convertidor funciona con su propio reloj, derivado del reloj de entrada OSC1 o de su propio oscilador RC incorporado. El tiempo de conversión para cada bit es *tad* y el tiempo de conversión total es 10 x *tad*. La selección del reloj, a través de los bits ADC1, 0 del registro ADCON0, se debe hacer de tal manera que *tad* sea al menos de 2 μs. Al seleccionar el tipo de oscilador debe tenerse presente que la frecuencia varía con el voltaje, la temperatura y otros parámetros propios del proceso de fabricación del microcontrolador.

- Interrupciones

Interrupción por conversión A/D. el convertidor A/D coloca en 1 el bit ADIF (INTCON<1>) al finalizar una conversión. El bit de control respectivo es ADIE (INTCON<6>).

Nota: *los conceptos no descritos del PIC16C71, se suponen iguales al del PIC16F84.*

1.3 Características del PIC16F872

a) Características especiales.

- Actuación alta RISC CPU:

1. Único con 35 palabras en la memoria de instrucciones
2. Todas las instrucciones son de un ciclo salvo el programa ramas que son dos-ciclo
3. Velocidad de operación
   - DC - 20 MHz reloj entrada
   - DC 200 ciclo de instrucción de ns
4. 2K x 14 bits de Programa de Memoria tipo flash
5. 128 bytes en la Memoria de Datos (RAM)
6. 64 bytes de memoria de datos EEPROM
7. Pinout compatible con los PIC16C72A
8. Capacidad de interrupción (para 10 fuentes)
9. Ocho niveles de profundidad de la pila del hardware
10. Modos de direcciones, Directos, Indirectos y Relativos

- Rasgos periféricos

1. Corriente de Sumidero/fuente Alta: 25 mA
2. Tiempo0: 8-bit contador de tiempo con 8-bit preescalar
3. Un Tiempo de 16-bit: contador de tiempo con preescalar, puede incrementarse durante modo de bajo consumo vía externo oscilador/cristal.
4. Dos tiempos 8-bits: contador de tiempo con 8-bit periodo registro, preescalar y postescalar
5. Una Captura, Comparación, módulo de PWM
6. La captura máxima es 16-bit, la resolución es 12.5 ns
7. La comparación máxima es 16-bit, la resolución es 200 ns

8. El PWM máximo y la resolución es 10-bit
9. 5 canales de 10-bit, convertidor Analógico-a-Digital (A/D)
10. Sincronizador del Puerto serial (SSP) con SPI. (modo maestro) y I2C. (modo/Slave)
11. Circuitería para detección de Brown-out y Brown-out reset (BOR)

- Tecnología de CMOS:

1. El rango de voltaje de operación esta entre 2.0 V a 5.5 V
2. Modo de bajo consumo:
   - <2 mA típico @ 5V, 4 MHz,
   - 20 µA típico @ 3V, 32 kHz,
   - <1 µA de la corriente del estado de espera típica

- Características especiales del Microcontrolador

1. Power-on reset (POR), contador de tiempo Power-up (PWRT) y tiempo del Oscilador start-up (OST)
2. Contador de tiempo de Perro guardián (WDT) con su propio on-chip RC oscilador para el funcionamiento fiable
3. Protección del código Programable
4. Ahorro de energía en modo de bajo consumo
5. Oscilador con selección entre cuatro tipos distintos
6. Circuito de Programación en serie. (ICSP.) vía dos pines
7. 5V de capacidad en el circuito de programación serial
8. En-circuito pone a punto vía dos pines
9. Procesadores para leer/escribir se acceden para la programación de la memoria

b) Organización de Memoria

En la Figura 1.3.1, se encuentra la arquitectura general del PIC16F872, que muestra los diferentes bloques que lo componen y la forma en que están conectados entre sí por medio de líneas que transportan información entre dos o más módulos.

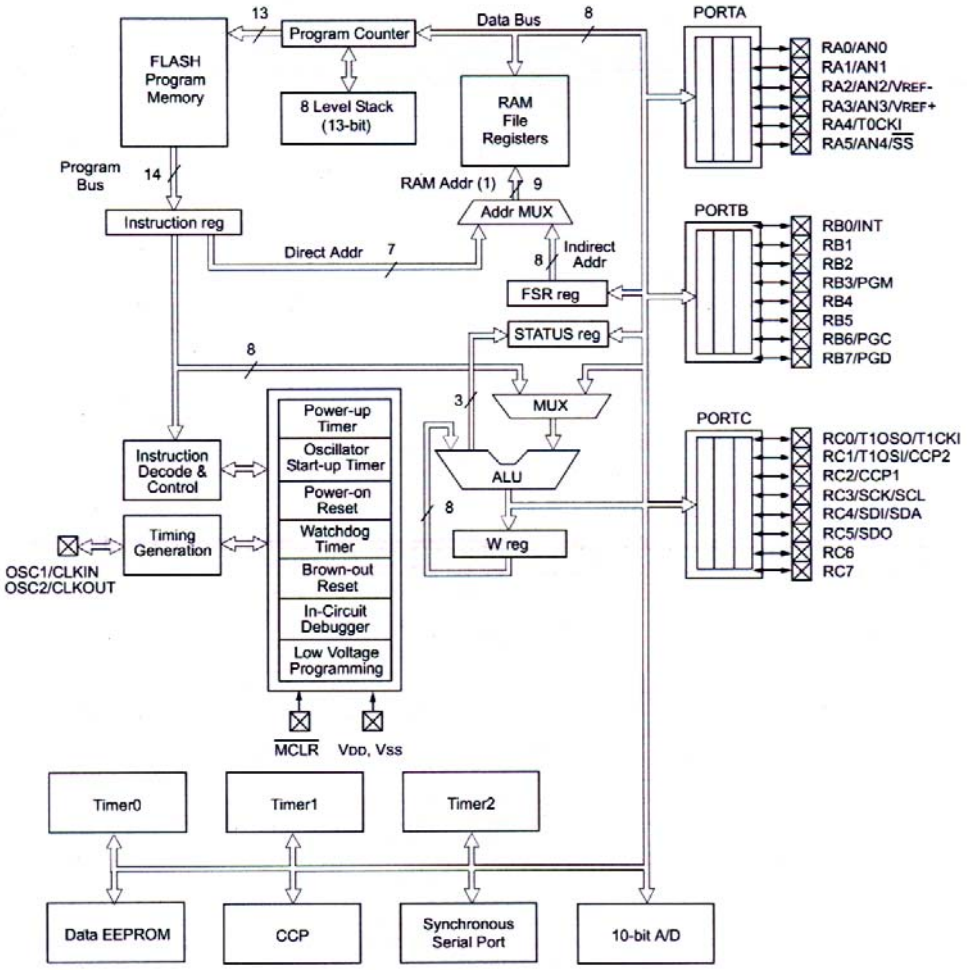


Figura 1.3.1 Arquitectura interna del PIC16F872.

- Organización de Memoria de programa

Los PIC16F872 tienen un contador de programa de 13-bit tiene una capacidad de direccionamiento de 8K x 14 bit de espacio en memoria de programa. El PIC16F872 es un dispositivo que tiene hasta 2K de palabras de Memoria de programa FLASH.

- Organización de Memoria de datos

La memoria de datos se divide en bancos múltiples qué contienen los Registros de Propósitos Generales y el Registros de la Función Especial. Cada banco se extiende a (128 bytes). Los bits más bajos de cada banco son reservados para Registros Especiales de la Función. La Figura 1.3.2 muestra el arreglo de los registros de la memoria RAM.

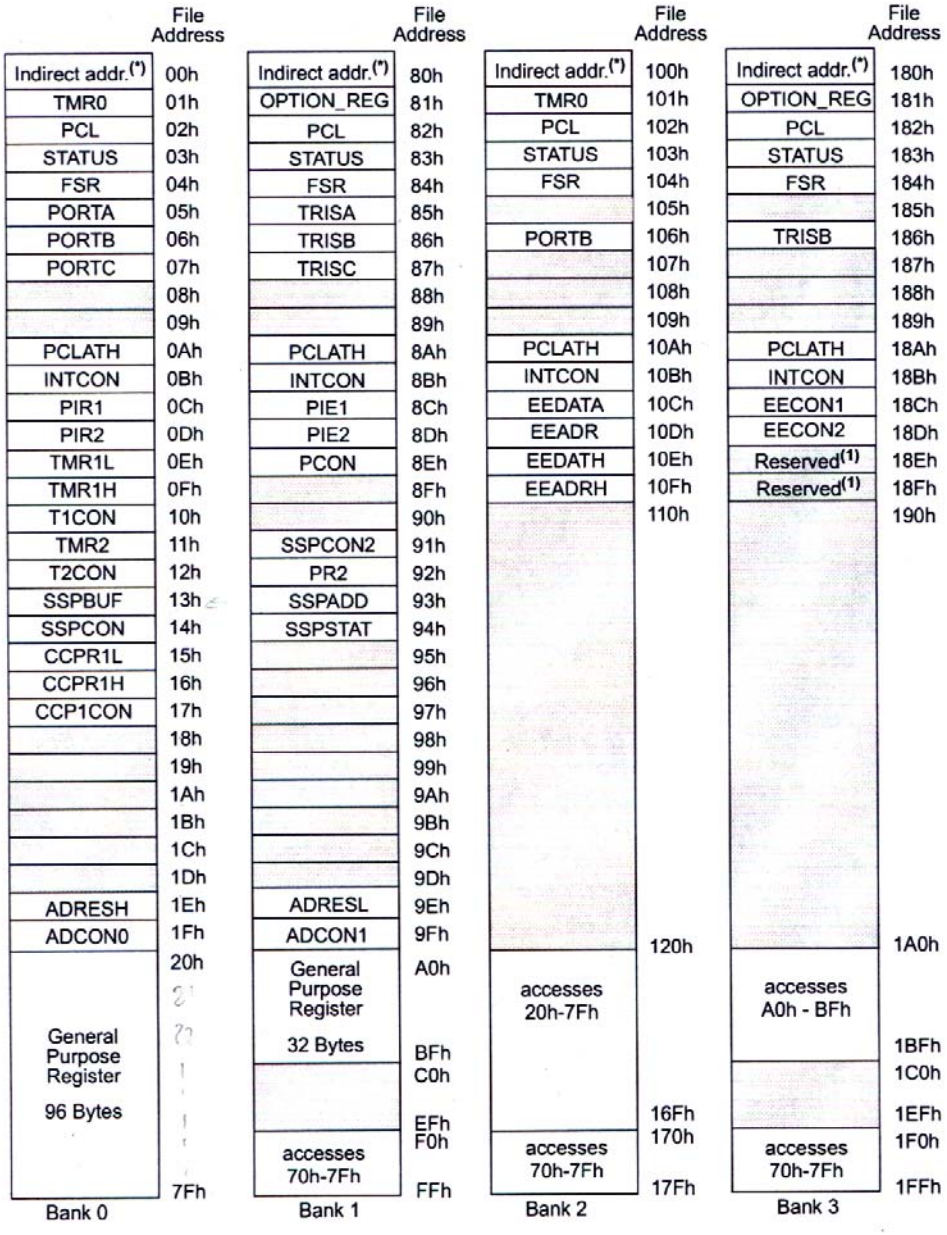


Figura 1.3.2 Registros del PIC16F872.

C) Función especial de registro del "PIC16F872"

En las Figuras 1.3.3, 1.3.4 y 1.3.5 se muestran los resgistros de direccionamiento de los bancos para el PIC16F872.

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
00h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	21, 93
01h	TMR0	Timer0 Module Register								xxxx xxxx	35, 93
02h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	20, 93
03h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	12, 93
04h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	21, 93
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	29, 93
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	31, 93
07h	PORTC	PORTC Data Latch when written: PORTC pins when read								xxxx xxxx	33, 93
08h	—	Unimplemented								—	—
09h	—	Unimplemented								—	—
0Ah(1,2)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	20, 93
0Bh(2)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	14, 93
0Ch	PIR1	(3)	ADIF	(3)	(3)	SSPIF	CCP1IF	TMR2IF	TMR1IF	r0rr 0000	16, 93
0Dh	PIR2	—	(3)	—	EEIF	BCLIF	—	—	(3)	-r-0 0--r	18, 93
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	40, 94
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	40, 94
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	--00 0000	39, 94
11h	TMR2	Timer2 Module Register								0000 0000	43, 94
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	43, 94
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	55, 94
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	53, 94
15h	CCPR1L	Capture/Compare/PWM Register1 (LSB)								xxxx xxxx	45, 94
16h	CCPR1H	Capture/Compare/PWM Register1 (MSB)								xxxx xxxx	45, 94
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	45, 94
18h	—	Unimplemented								—	—
19h	—	Unimplemented								—	—
1Ah	—	Unimplemented								—	—
1Bh	—	Unimplemented								—	—
1Ch	—	Unimplemented								—	—
1Dh	—	Unimplemented								—	—
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	84, 94
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	$\overline{GO/DONE}$	—	ADON	0000 00-0	79, 94

Figura 1.3.3 Banco0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 1											
80h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	21, 93
81h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	13, 94
82h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	20, 93
83h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	12, 93
84h(2)	FSR	Indirect data memory address pointer								xxxx xxxx	21, 93
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	29, 94
86h	TRISB	PORTB Data Direction Register								1111 1111	31, 94
87h	TRISC	PORTC Data Direction Register								1111 1111	33, 94
88h	—	Unimplemented								—	—
89h	—	Unimplemented								—	—
8Ah(1,2)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	20, 93
8Bh(2)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	14, 93
8Ch	PIE1	(3)	ADIE	(3)	(3)	SSPIE	CCP1IE	TMR2IE	TMR1IE	r0rr 0000	15, 94
8Dh	PIE2	—	(3)	—	EEIE	BCLIE	—	—	(3)	-r-0 0--r	17, 94
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	$\overline{BOR}$	---- --qq	19, 94
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	54, 94
92h	PR2	Timer2 Period Register								1111 1111	43, 94
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	58, 94
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	52, 94
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
95h	—	Unimplemented								—	—
95h	—	Unimplemented								—	—
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	—	Unimplemented								—	—
9Dh	—	Unimplemented								—	—
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	84, 94
9Fh	ADCON1	ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0	0--- 0000	80, 94

Figura 1.3.4 Banco1

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 2											
100h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	21, 93
101h	TMR0	Timer0 Module Register								xxxx xxxx	35, 93
102h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	20, 93
103h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	12, 93
104h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	21, 93
105h	—	Unimplemented								—	—
106h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	31, 93
107h	—	Unimplemented								—	—
108h	—	Unimplemented								—	—
109h	—	Unimplemented								—	—
10Ah(1,2)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	20, 93
10Bh(2)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	14, 93
10Ch	EEDATA	EEPROM Data Register Low Byte								xxxx xxxx	23, 94
10Dh	EEADR	EEPROM Address Register Low Byte								xxxx xxxx	23, 94
10Eh	EEDATH	—	—	EEPROM Data Register High Byte						xxxx xxxx	23, 94
10Fh	EEADRH	—	—	—	EEPROM Address Register High Byte					xxxx xxxx	23, 94
Bank 3											
180h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	21, 93
181h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	13, 94
182h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	20, 93
183h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	12, 93
184h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	21, 93
185h	—	Unimplemented								—	—
186h	TRISB	PORTB Data Direction Register								1111 1111	31, 94
187h	—	Unimplemented								—	—
188h	—	Unimplemented								—	—
189h	—	Unimplemented								—	—
18Ah(1,2)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	20, 93
18Bh(2)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	14, 93
18Ch	EECON1	EEPGD	—	—	—	WRERR	WREN	WR	RD	x--- x000	24, 94
18Dh	EECON2	EEPROM Control Register2 (not a physical register)								---- ----	23, 94
18Eh	—	Reserved; maintain clear								0000 0000	—
18Fh	—	Reserved; maintain clear								0000 0000	—

Figura 1.3.5 Banco2 y Banco3

I.4 Características de los Módulos LCD

c) Pines de conexión del módulo LCD

Cuadro 1.4.1 Función de los pines del módulo LCD

Pin (Terminal)	Nombre	Función
1	Vss	Conecte con tierra para LCD
2	Vdd	5V suministro de alimentación
3	Vo	Ajuste de voltaje de contraste (entre 0-5V)
4	RS	Selección Dato/Control
5	R/W	Lectura/Escritura en LCD
6	E	Habilitación
7	DB0	LSB datos de línea (D0 bit menos significativo)
8	DB1	D1
9	DB2	D2
10	DB3	D3
11	DB4	D4
12	DB5	D5
13	DB6	D6
14	DB7	MSB datos de línea (D7 bit más significativo)
15	BLA	LED backlight Anode
16	BLK	LED backlight Cathode

b) Controles y comandos del display

Display Clear: Esto limpia los contenidos en la memoria de datos y restablece la dirección del contador atrás a 0x00.

Return Home: (Retorno a casa). Esto restablece la dirección del contador a 0x00 y restablece el cambio del Display.

Entry Mode: Este comando pondrá la dirección del auto-incremento del cursor, y también el on/off del auto-cambio.

Display On/Off: Este comando enciende o apaga el display. También pone el cursor en encendido o apagado. Si el cursor esta encendido, puede aparecer como una raya parpadeando o un bloque sólido parpadeando.

Shift: Este comando cambiará los contenidos enteros de memoria a derecha o izquierda. Si el despliegue está menos de 80 caracteres, los caracteres no están en las direcciones del despliegue actual pero están en la memoria de datos y pueden cambiarse en la región del despliegue.

Set Functión: Éste es un comando de inicialización. Este pone al controlador para 8 bit o interfase de 4 bit, 1 línea o 2 líneas desplegadas, y 5x7 o 5x10 por el conjunto de punto del carácter.

Set CGRAM address: Éste es un comando especial para tener el punto de indicador de dirección en la línea RAM de Generador de Usuario.

Set DD RAM address: Este comando pondrá el indicador de dirección (o cursor) donde usted lo quiere en la pantalla.

Read Busy Flag: Esta instrucción hará a los LCD desplegar la dirección de indicador de dirección actual para el microcontrolador y mostrará la Bandera Ocupada.

Write Data: Este es el comando de Datos para escribir los byte que usted desea guardar en memoria.

Read Data: Alternadamente, tu puedes leer los contenidos de los datos guardados en la dirección actual en la que el indicador de dirección está.

- *Escritura del primer código para un módulo lcd*

Primero tendremos que inicializar nuestro PIC, podemos empezar usando la rutina de retraso de tiempo múltiple. De esta manera nosotros controlamos cuánto tiempo necesitamos tardar, sin necesidad de hacer subrutinas separadas durante tiempos diferentes. Después de esto, nosotros inicializamos el LCD. Puesto que el controlador tiene una inicialización interior, nosotros le daremos 20 ms para que se inicialice. Después de esto usamos el comando de la Función Fijo para cambiar a interfase de 8 bit /2 línea, display/5x7.

Usaremos el próximo comando de cambio para asegurarnos que tenemos el incremento del cursor al derecho. Luego, encendemos el display finalmente se proporciona nitidez al display, simplemente para asegurarse de que todo se alambro correctamente y enviamos un pequeño mensaje de prueba.

En el Anexo III Proyecto 1.2 mostramos nuestro código principal que registrará los pulsos de 3 botones y despliega un carácter por pulso. Si se aprieta el botón 1, nosotros le enviamos una orden al LCD recuperando un byte del Puerto B. Si se aprieta el botón 2, nosotros le enviamos un byte de datos al LCD recuperando un byte del Puerto B. Si se aprieta el botón 3, nosotros enviamos el cursor a casa.

Algunos comandos nos toman de 42 a 1.64 ms ejecutarlos, pero esos tiempos son requeridos para que inicialice el LCD a través de la instrucción en lugar de permitir que la inicialización interior haga todo el trabajo.

Verificamos la bandera ocupada después de cada comando enviado. Sólo entonces conseguimos los tiempos declarados en la lista de la Instrucción. El no verificar la bandera ocupada aumentará el tiempo exigido para ejecutar cada comando. Y todo lo que necesitamos hacer es llamar la subrutina Busyflag.

Cuando escribimos nuestro programa principal, lo que hacemos es llamar la rutina InitLCD. Durante la rutina de InitLCD, se usan dos llamadas diferentes: Wcommand y Command. Uno se encarga de verificar la bandera ocupada y el otro no hace nada. Durante la inicialización por instrucción, no verificamos la bandera ocupada hasta después de la función del tercer juego. Pero después de que la rutina se hace, ponemos un byte en el registro de trabajo y llamamos al subprograma correcto. Haciendo esto harán al LCD unir tiempos tan rápido como sea posible.

I.5 Módulo Analógico - Digital

En el caso de los PIC16F872, hay un módulo de 5 canales con pocas opciones, usando el Puerto A para las señales de entradas. Los 5 canales son multiplexados en el Puerto A, pero sólo uno puede ser usado una sola vez, pero no es necesario usar los 5 canales. El módulo puede tener una sola entrada analógica, o combinaciones de entradas de Digital/Analoga en los 5 canales.

El convertidor A/D trabaja básicamente con un voltaje analógico de entrada que puede ser cualquier voltaje de Vref - a Vref+, y el módulo da una salida en binario. Esta salida binario representa un rango de voltaje analógico, suponga que tiene un Vref+ = 5V, y Vref - = tierra en el bit 10 el convertidor de A/D dará una salida binario dependiendo de la entrada de la caída de voltaje (en un rango de +5 V y 0

V). De la resolución para el bit 10 hay 1024 niveles que pueden ser posibles. Así que si su entrada de voltaje es 0 V, entonces la salida en binario será 0x000. Como la entrada de voltaje aumenta a 5 mV, entonces la salida en binario cambia a 0x001. Por ejemplo, si usted tiene una entrada de 3.57 V, entonces cada número binario representa (5 V/1024) = 4.8828 mV. (3.57 V /4.8828 mV) = 731, qué se representaría en binario como 0x2DB.

El módulo de A/D en el PIC16F físicamente tiene seis series de trabajos. Después de hacer la configuración, el voltaje de la entrada analógico se aplica al pin RA0. El voltaje analógico carga un condensador interior que muestra el nivel de voltaje. Se requiere una cantidad adecuada de tiempo para realizar esta función. El módulo de A/D comienza convirtiendo la salida del voltaje internamente a un número binario, y cuando esto ha terminado, pondrá una bandera de interrupción y guardará el resultado binario en los registros de ADRESH/ADRESL.

La parte más difícil para usar el módulo de A/D es para calcular el tiempo de adquisición. Vea la siguiente ecuación:

$$Tacq = Tamp + Tc + Tcoff$$

Tamp tiene aproximadamente 2μ segundos. Tcoff depende de la temperatura del PIC en el tiempo de adquisición. Tcoff = [(Temperatura – 25 ºC)(0.05 μsegundos/ºC)], para una temperatura ambiental (25 ºC), Tcoff = 0. Aun cuando usemos el PIC en temperaturas altas (50 ºC), Tcoff = 1.25 μsegundos. Dependiendo de qué tipo de resistencia se usa en la entrada para el voltaje analógico. Si la resistencia de entrada es 5 kΩ, entonces Tc = 120 pF (13 kΩ) (7.625) = 11.9 μs, para hacer cosas simples la resistencia de la entrada mas alta recomendada es 10 kΩ, para que el tiempo requerido entre las conversiones sea alrededor de 20 μs.

Con la configuración, el próximo paso si se va a estar midiendo un voltaje, y se van a leer los resultados con el código escrito para el LCD se convierte el resultado binario a decimal. El valor binario será 8bit y significará una resolución de 19.5 mV por bit. Así mis resultados estarán expresados en hexadecimal en un rango de 0x7B a 0x00 (binario 123 a 0). Necesitamos convertir esto en el voltaje que estamos intentando medir (12 V variables). Esto me exigirá tomar el resultado y multiplicar por 5 para el divisor de voltaje y por 19.5 para la resolución del mV. Se usa la rutina matemática de 8bit por 8bit para desplegar números hexadecimales en el LCD, necesitamos también una rutina matemática para un BCD de 16bit.

Esta rutina la ponemos después de la interrupción del área de rutina, pero antes de la declaración Principal.

Existe un diseño de tiempo que causa un reset cada 500 ms, este evento ocurre en la rutina de interrupción. Constantemente con la ayuda del código Principal se realizan las conversiones A/D y se guardan los resultados en un GPR, cuando transcurren los 500 ms de la rutina de interrupción toma el resultado y lo despliega en el LCD.

Los resultados de la conversión se guardan en el GPR, se toman y se realiza la rutina de multiplicación de 8 x 8, incrementando después estos resultados a 5 dígitos que el BCD numera para ser desplegados en el LCD. Todo esto ocupa 2 líneas de código en mi rutina de servicio y toma más de 2 ciclos ejecutarse, el próximo paso es convertir los números de BCD en los caracteres proporcionados por el LCD.

Después se arregla el módulo A/D. En el programa principal, configuramos el registro TRIS para que RA0 sea una entrada, y el resto de RA salidas (para controles de líneas del LCD). El PORTC es la salida de los datos para el LCD, por

lo que TRISC contiene todas las salidas. El PORTB se acostumbra usar, para cualquier otra cosa.

Anexo II

2.1 Función especial de registros

Los microcontroladores responden a una serie de instrucciones ó códigos que se deben grabar en su memoria de programa, en total son 35 los mas empleados. La siguiente Tabla es una lista corta de las instrucciones.

Cuadro 2.1.1 Lista de instrucciones

Si d = 0 el resultado se almacena en W
Si d = 1 el resultado se almacena en el registro

Nemotécnico	Operación	Cód. de operación msb lsb	Estados afectados
Operaciones orientadas a registros			
ADDWF f,d	Sumar W y f	00 0111 dfff ffff	C,DC,Z
ANDWF f,d	AND entre W y f	00 0101 dfff ffff	Z
CLRF f	Limpiar f	00 0001 1fff ffff	Z
CLRW	Limpiar w	00 0001 0XXX XXXX	Z
COMF f,d	Complementar f	00 1001 dfff ffff	Z
DECF f,d	Decrementar f	00 0011 dfff ffff	Z
DECFSZ f,d	Decrementar f, saltar si cero	00 1011 dfff ffff	
INCF f,d	Incrementar f	00 1010 dfff ffff	Z
INCFSZ f,d	Incrementar f, saltar si cero	00 1111 dfff ffff	
IORWF f,d	OR entre W y f	00 0100 dfff ffff	Z
MOVF f,d	Mover f	00 1000 dfff ffff	Z
MOVWF f	Mover W a f	00 0000 1fff ffff	
NOP	No operación	00 0000 0XX0 0000	
RLF f,d	Rotar a la izquierda a través del carry	00 1101 dfff ffff	C
RRF f,d	Rotar a la derecha a través del carry	00 1100 dfff ffff	C
SUBWF f,d	Restar W de f	00 0010 dfff ffff	C,DC,Z
SWAPF f,d	Intercambiar nibbles de f	00 1110 dfff ffff	
XORWF f,d	OR exclusiva entre W y f	00 0110 dfff ffff	Z
Operaciones orientadas a bits			
BCF f,b	Limpiar bit b de f	01 00bb bfff ffff	
BSF f,b	Activar bit b de f	01 01bb bfff ffff	
BTFSC f,b	Probar bit b de f, saltar si es cero	01 10bb bfff ffff	
BTFSS f,b	Probar bit b de f, saltar si es uno	01 11bb bfff ffff	
Operaciones literales y de control			
ADDLW k	Sumar literal k a W	11 111X kkkk kkkk	C,DC,Z
ANDLW k	AND entre k y W	11 1001 kkkk kkkk	_Z_
CALL k	Llamar subrutina	10 0kkk kkkk kkkk	
CLRWDT	Limpiar WDT	00 0000 0110 0100	T0,PD
GOTO k	Salta a dirección k	10 1kkk kkkk kkkk	
IORLW k	OR entre k y W	11 1000 kkkk kkkk	Z
MOVLW k	Cargar a W con literal k	11 00XX kkkk kkkk	
RETFIE	Retornar de interrupción	00 0000 0000 1001	
RETLW k	Retornar y cargar a W con k	11 01XX kkkk kkkk	_ _
RETURN	Retornar de subrutina	00 0000 0000 1000	
SLEEP	Ir al modo de bajo consumo	00 0000 0110 0011	T0,PD
SUBLW k	Restarle k a W	11 110X kkkk kkkk	C,DC,Z
XORLW k	OR exclusiva entre k y W	11 1010 kkkk kkkk	Z

2.2 Lista descriptiva de lo que hace cada instrucción

ADDWF/SUBWF: Estos comandos toman el dato contenido en el registro del archivo y suma/resta el contenido del registro al contenido del registro W.

Sintaxis:	**ADDWF f,d**
Operación:	(W) + (f) k (destino)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C, DC, Z
Ejemplo:	ADDWF FSR,0
Antes de la instrucción:	W=17 FSR=C2
Después de la instrucción:	W=D9 FSR=C2

Sintaxis:	**SUBWF f,d**
Operación:	(f) - (W) k (destino)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C, DC, Z
Ejemplo1:	SUBWF regis,1
Antes de la instrucción:	regis=3, W=2, C=?
Después de la instrucción:	regis=1, W=2, C= 1 (positivo)
Ejemplo 2:	
Antes de la instrucción:	regis=2, W=2, C=?
Después de la instrucción:	regis=0, W=2, C= 1 (cero)
Ejemplo 3:	
Antes de la instrucción:	regis=1, W=2, C=?
Después de la instrucción:	regis=FF, W=2, C= 0 (negativo)

ANDWF: Este comando toma los contenidos del registro de archivo y realiza la operación lógica AND entre un registro y el registro W. El resultado se pone en el registro del archivo o el registro W.

Sintaxis:	**ANDWF f,d**
Operación:	(W) AND (f) k (destino)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	ANDWF FSR,1
Antes de la instrucción:	W=17 FSR=C2
Después de la instrucción:	W=17 FSR=02

CLRF (Borra el contenido del registro "f", lo carga con 00)**:** Está instrucción limpia todos los bits en el registro del archivo seleccionado. Pueden usarse al inicializar los puertos, o limpiar cualquier dato de los registros que tu necesites.

Sintaxis:	**CLRF f**
Operación:	00 k (f)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	CLRF regis
Antes de la instrucción:	regis = 5A
Después de la instrucción:	regis = 00

CLRW: Esto limpia todos los bits en el registro W, lo carga con 00.

Sintaxis:	**CLRW**
Operación:	00 k (W)

Ciclos de instrucción: 1
Bits del registro de
Estados que se afectan: Ninguno
Ejemplo: CLRW
Antes de la instrucción: W = 5A
Después de la instrucción: W = 00

COMF: Este comando complementa el contenido del registro “f”. Cualquier bit que fuera un 0, se convertirá en un 1, y viceversa. El resultado o se pone en el registro f o en el registro w

Sintaxis: **COMF f,d**
Operación: (f) k (destino)
Ciclos de instrucción: 1
Bits del registro de
Estados que se afectan: Z
Ejemplo: COMF regis,0
Antes de la instrucción: regis = 13, W =?
Después de la instrucción:

DECF/INCF: Este comando o substrae 1 (decrementa el contenido del registro “f”), o agrega 1 (incrementa el contenido del registro “f”). El resultado puede ponerse cualquier parte de atrás en el registro del archivo o el registro W.

Sintaxis: **DECF f,d**
Operación: (f) – 1 k (destino)
Ciclos de instrucción: 1
Bits del registro de
Estados que se afectan: Z
Ejemplo: DECF regis,1
Antes de la instrucción: regis = 13

Después de la instrucción: regis = 12

Sintaxis: **INCF f,d**

Operación: (f) + 1 k (destino)

Ciclos de instrucción: 1

Bits del registro de

Estados que se afectan: Z

Ejemplo: INCF regis,1

Antes de la instrucción: regis = 24

Después de la instrucción: regis = 25

DECFSZ/INCFSZ: Este comando hace la misma función que INCF/DECF, sólo que si el resultado es cero, entonces la próxima instrucción se salta. El resultado o puede ponerse en f (registro del archivo) o w (registro de trabajo).

Sintaxis: **DECFSZ f,d**

Operación: (f) – 1 k (destino) salta si es resultado = 0

Ciclos de instrucción: 1 (2)

Bits del registro de

Estados que se afectan: Ninguno

Ejemplo:

```
aquí        DECFSZ      regis,1
            GOTO        ciclo
            Continúa
```

Antes de la instrucción: contador de programa = aquí

Después de la instrucción: regis = regis – 1

si regis = 0, entonces

contador de programa = continua

si regis ≠ 0, entonces

contador de programa = aquí + 1

Sintaxis: **INCFSZ f,d**

Operación: (f) + 1 k (destino) salta si es resultado = 0

Ciclos de instrucción:	1 (2)
Bits del registro de	
Estados que se afectan:	Ninguno
Ejemplo:	aquí DECFSZ regis,1
	GOTO ciclo
	Continúa
Antes de la instrucción:	contador de programa = aquí
Después de la instrucción:	regis = regis + 1
	si regis = 0, entonces
	contador de programa = continua
	si regis ≠ 0, entonces
	contador de programa = aquí + 1

XORWF: Este comando exclusivamente realiza una operación lógica XOR, entre el registro W y el registro de F.

Sintaxis:	**XORWF f,d**
Operación:	(W) XOR (f) k (destino)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	XORWF regis,1
Antes de la instrucción:	regis = AF, W = B5
Después de la instrucción:	regis = 1ª, W = B5

MOVF: Este comando mueve el contenido del registro “f”. Nota: no hay ninguna manera de pasar los contenidos de un registro del archivo actualmente a otro registro del archivo sin pasar primero al registro de trabajo.

Sintaxis:	**MOVF**
Operación:	(f) k (destino)

Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	MOVF regis,0
Antes de la instrucción:	W =?
Después de la instrucción:	W = valor guardado en regis

MOVWF: Esto mueve el contenido del registro de trabajo (W), al registro del archivo (F).

Sintaxis:	**MOVWF**
Operación:	(W) k (f)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	MOVWF OPTION
Antes de la instrucción:	OPTION =?, W =48
Después de la instrucción:	OPTION =48 W =48

NOP: Funcionamiento nulo. Simplemente no hace nada durante un ciclo de la instrucción.

Sintaxis:	**NOP**
Operación:	k k (W)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	NOP

RLF: Rota el contenido del registro “f” a la izquierda, usando el carry

Sintaxis:	**RLF f,d**
Operación:	rota el contenido del registro “f” a la izquierda a través del carry. Si d=0, el resultado se guarda en W, si d=1 en “f”.
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C
Ejemplo:	RLF regis,0
Antes de la instrucción:	C = 0, W=?, regis = 11100110
Después de la instrucción:	C = 1, W= 11001100
	regis = 11100110

RRF: Rota el contenido del registro “f” a la derecha, usando el carry

Sintaxis:	**RRF f,d**
Operación:	rota el contenido del registro “f” a la derecha a través del carry. Si d=0, el resultado se guarda en W, si d=1 en “f”.
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C
Ejemplo:	RF regis,0
Antes de la instrucción:	C = 0, W=?, regis = 11100110
Después de la instrucción:	C = 0, W= 01110011
	regis = 11100110

SWAPF: Los contenidos de los 4 bits más altos se cambian con los 4 bits más bajos en el registro del archivo. El destino es f o w. Ejemplo: FR=01100000 Result=00000110

Sintaxis:	**SWAPF f,d**

Operación:	f<3:0> k d<7:4>
	f<7:4> k d<3:0>
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	WAPF regis,0
Antes de la instrucción:	regis = A5, W =?
Después de la instrucción:	regis = A5, W = 5A

IORWF: Operación lógica entre el registro W y el registro "f".

Sintaxis:	**IORWF f,d**
Operación:	(W) OR (f) k (destino)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	IORWF regis,0
Antes de la instrucción:	regis = 13, W =91
Después de la instrucción:	regis = 13 W =93

BCF: El bit correspondiente en el registro del archivo se pone en cero. Sólo un bit se pone en cero y el resto queda inalterado. Ej. Pone en cero el bit "b" del registro "f".

Sintaxis:	**BCF f,d**
Operación:	0 k (f<b>)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	BCF regis,7
Antes de la instrucción:	regis = C7

Después de la instrucción: regis = 47

BSF: Pone en uno el bit “b” del registro “f”.

Sintaxis:	**BCF f,d**
Operación:	0 k (f<b>)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	BCF regis,7
Antes de la instrucción:	regis = C7
Después de la instrucción:	regis = 47

BTFSC/BTFSS: El bit seleccionado en el registro del archivo se prueba para ver si está en cero (BTFSC), o en uno para (BTFSS). Si la prueba es TRUE, entonces la próxima instrucción se salta.

Sintaxis:	**BTFSC f,d**
Operación:	Salta si (f<b>) = 0
Ciclos de instrucción:	1 (2)
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	aquí BTFSC regis,0
	falso GOTO inicio
	verdad……..
Antes de la instrucción:	contador de programa = aquí
Después de la instrucción:	si el bit 0 del registro regis = 0
	contador de programa = verdad
	si el bit 0 del registro regis = 1
	contador de programa = falso

ADDLW: Suma un valor literal al contenido del registro W.

Sintaxis:	**ADDLW k**
Operación:	(W) + k k (W)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C, DC, Z
Ejemplo:	ADDLW 15
Antes de la instrucción:	W =10
Después de la instrucción:	W = 25

SUBLW: Resta el contenido del registro W de el literal “k” (usando el método de complemento a dos).

Sintaxis:	**SUBLW k**
Operación:	(k) - (W) k (W)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	C, DC, Z
Ejemplo:	SUBWF 02
Antes de la instrucción:	W=1, C=?
Después de la instrucción:	W=1, C= 1 (positivo)
Ejemplo 2:	
Antes de la instrucción:	W=2, C=?
Después de la instrucción:	W=0, C= 1 (cero)
Ejemplo 3:	
Antes de la instrucción:	W=3, C=?
Después de la instrucción:	W=FF, C= 0 (negativo)

ANDLW: Operación lógica AND entre el valor literal y el registro de W, y el resultado se guarda en el registro de W.

Sintaxis:	**ANDLW k**
Operación:	(W) AND (k) k (W)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	ANDLW 5F
Antes de la instrucción:	W =A3
Después de la instrucción:	W = 03

CALL: Llama una subrutina que está ubicada en la posición de memoria o etiqueta “k”.

Sintaxis:	**CALL k**
Operación:	(PC) + 1 k pila, k k (PC)
Ciclos de instrucción:	2
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	aquí CALL rutina
Antes de la instrucción:	contador de programa = aquí
Después de la instrucción:	contador de programa = rutina
	pila = dirección aquí

CLRWDT: Borra el conteo del Watchdog Timer.

Sintaxis:	**CLRWDT**
Operación:	00 k WDT
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	TO, PD
Ejemplo:	CLRWDT
Antes de la instrucción:	Contador WDT =?
Después de la instrucción:	Contador WDT = 00
	TO = 1, PD =1

GOTO: El contador del programa se salta a la dirección representada por el literal k. No hace cambios en la pila.

Sintaxis:	**GOTO k**
Operación:	k k PC
Ciclos de instrucción:	2
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	GOTO ciclo
Antes de la instrucción:	contador de programa = ?
Después de la instrucción:	contador de programa = ciclo

IORLW: Realiza una operación lógica OR entre el registro W y el literal “k”. El resultado se pone atrás en el registro de trabajo.

Sintaxis:	**IORLW k**
Operación:	(W) OR (k) k (w)

Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	IORLW 35
Antes de la instrucción:	W = 9A
Después de la instrucción:	W = BF

MOVLW : El valor literal se pone en el registro de trabajo.

Sintaxis:	**MOVLW k**
Operación:	k k (w)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	MOVLW 5A
Antes de la instrucción:	W =?
Después de la instrucción:	W = 5A

RETFIE: Después de que una interrupción se ha hecho, esta instrucción pondrá el lugar de la cima de la pila atrás en el contador del programa, retornando así al punto en el que la interrupción ocurrió.

Sintaxis:	**RETFIE**
Operación:	pila k contador de programa, 1 k intcon,gie
Ciclos de instrucción:	2
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	RETFIE
Antes de la instrucción:	contador de programa =?
Después de la instrucción:	contador de programa = pila

RETURN: Los contenidos de la cima de la pila están puestos en el contador del programa y lo devuelve así de una instrucción de la CALL a donde su código del programa se ha estado ejecutando.

Sintaxis:	**RETURN**
Operación:	pila k contador de programa
Ciclos de instrucción:	2
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	RETURN
Después de la interrupción	contador de programa = pila

RETLW: Retorno de interrupción y carga el registro W con la literal “k”.

Sintaxis:	**RETLW**
Operación:	k k W, pila k contador de programa
Ciclos de instrucción:	2
Bits del registro de	
Estados que se afectan:	ninguno
Ejemplo:	CALL tabla
	•
	•
	tabla ADDWF PC
	RETLW k1
	RETLW k2
	•
	•
	RETLW kn
Antes de la instrucción:	W =07
Después de la instrucción:	W = k7

SLEEP: Entra en modo dormido (Standby).

Sintaxis:	**SLEEP**
Operación:	00 k WDT, 1 k TO, 0 k PD
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	TO, PD
Ejemplo:	SLEEP

XORLW: Realiza una operación lógica XOR entre el registro W y la literal “k”. El resultado se pone en W.

Sintaxis:	**XORLW k**
Operación:	(w) XOR (k) k (W)
Ciclos de instrucción:	1
Bits del registro de	
Estados que se afectan:	Z
Ejemplo:	XORLW AF
Antes de la instrucción:	W = B5
Después de la instrucción:	W = 1A

ANEXO III

```
;**** Proyecto  No. 1  Conexión de Led y Dipswitch con el  PIC16F84****
;
;este programa lee el estado de 4 interruptores y de acuerdo a ello enciende o
;no 4 led
;en caso de que un número se escriba D'15': significa numero decimal
;en caso de que el numero se escriba B'00010101': significa numero binario
;en caso de que un número se escriba 15H: significa numero hexadecimal
;si no se especifica nada, se supone numeración hexadecimal
;----------------Definición de registros-----------------------
;
pc              equ             02h
status          equ             03h
ptoa            equ             05h             ;el puerto A esta en la dirección 05 de la RAM
ptobu           equ             06h             ;el puerto B esta en la dirección 06 de la RAM
trisa           equ             85h             ;registro de configuración del puerto A
trisb           equ             86h             ;registro de configuración del puerto B
w               equ             00h             ;indica que el resultado se guarda en w


reset           org             0               ;el vector de reset es la dirección 00
                goto            inicio          ;se salta al inicio del programa

                org             5               ;el programa empieza en la dirección de memoria 5

inicio          bsf             status,5        ;se ubica en el segundo banco de la RAM
                movlw           0f0h            ;se carga el registro W con 0f
                movwf           trisa           ;se programan los pines del puerto A como salidas
                movlw           0ffh            ;se carga el registro W con ff
                movwf           trisb           ;se programan los pines del puerto B como entraas
                bcf             status,5        ;se ubica en el primer banco de la memoria RAM
ciclo           movf            ptob,w          ;el valor del puerto B lo pasa al registro W
                xorwf           0ffh            ;con una operacion xor se invierte el valor
                                                ;del dato leído del puerto B
                movwf           ptoa
                goto            ciclo
                end
```

```
;***Proyecto  No. 2 Manejo de un módulo LCD con conexión a 4 bits***
;
; este programa hace que un mensaje se repita indefinidamente
; en un modulo lcd de 2 líneas con 16 caracteres
;---------- Definición de registros ----------
;
indf        equ     0h      ;para direccionamiento indirecto
tmro        equ     1h      ;contador de tiempo real
pc          equ     2h      ;contador de programa
status      equ     3h      ;registro de estados y bits de control
fsr         equ     4h      ;selecccion de bancos de memoria y registros
ptoa        equ     5h      ;puertos
ptob        equ     6h
r0c         equ     0ch     ;
r0d         equ     0dh     ;
r0e         equ     0eh     ;
r13         equ     13h     ;
z           equ     2h      ;bandera de cero
c           equ     0h      ;bandera de carry
w           equ     0h      ;para almacenar en w
r           equ     1h      ;para almacenar en el mismo registro
e           equ     1h      ;
rs          equ     0h      ;

            org     00      ;vector de reset
            goto    inicio  ;va a iniciar programa principal
            org     05h

retardo     movlw   0ffh
            movwf   r13
decre       decfsz  r13,r
            goto    decre
            retlw   0

limpia      clrf    r0c
limpi       movlw   " "
            call    dato
            incf    r0c,r
            movlw   50h
            xorwf   r0c,w
            btfss   status,z
            goto    limpi
            retlw   0

control     bcf     ptob,rs ;esta rutina genera las señales de control
            goto    dato2   ;para escribir en el modulo lcd y
dato        bsf     ptob,rs ;entrega el dato a ser mostrado en la pantalla
dato2       bsf     ptob,e  ;utiliza la interfase a 4 bits
            movwf   r0e
            movlw   0fh
            andwf   ptob,r
```

```
                movf    r0e,w
                andlw   0f0h
                iorwf   ptob,r
                call    retardo
                bcf     ptob,e
                call    retardo
                bsf     ptob,e
                movlw   0fh
                andwf   ptob,r
                swapf   r0e,w
                andlw   0f0h
                iorwf   ptob,r
                call    retardo
                bcf     ptob,e
                call    retardo
                retlw   0

tabla           addwf   pc,r            ;mensaje que se muestra
                retlw   " "
                retlw   " "
                retlw   "R"
                retlw   "E"
                retlw   "V"
                retlw   "I"
                retlw   "S"
                retlw   "T"
                retlw   "A"
                retlw   " "
                retlw   "E"
                retlw   "&"
                retlw   "C
                retlw   " "
                retlw   " "
                retlw   " "
                                        ;mensaje de la segunda línea
                retlw   " "
                retlw   "C"
                retlw   "E"
                retlw   "K"
                retlw   "I"
                retlw   "T"
                retlw   " "
                retlw   "-"
                retlw   " "
                retlw   "P"
                retlw   "E"
                retlw   "R"
                retlw   "E"
                retlw   "I"
                retlw   "R"
                retlw   "A"
```

```
                retlw       " "
                retlw       0

inicio          movlw       0ffh        ;programación de puertos
                tris        ptoa        ;según el circuito
                movlw       0ch         ;
                tris        ptob        ;
begin           movlw       02h         ;inicia display a 4 bits
                call        control
                movlw       28h         ;display a 4 bits y 2 lineas
                call        control
                movlw       0ch         ;activa display
                call        control
                movlw       06h         ;hace que el mensaje permanezca fijo
                call        control

blank           call        limpia      ;borra display
muestra         clrf        r0c         ;inicia contador de caracteres
ciclo           movf        r0c,w       ;hace barrido de la tabla
                call        tabla
                call        dato
                movlw       0ffh        ;retardo entre caracteres
                movwf       r0d
reta1           call        retardo
                decfsz      r0d,r
                goto        reta1
                incf        r0c,r       ;sigue con la tabla
                movlw       11h
                subwf       r0c,w       ;pregunta si esta mostrando el mensaje de la
                btfss       status,c    ;segunda línea
                goto        ciclo       ;
                movlw       11h         ;pregunta si es la primera vez que entra
                xorwf       r0c,w       ;a la segunda línea para ir a iniciar
                btfss       status,z    ;el puntero de la ram del modulo lcd
                goto        line2       ;
linea2          movlw       0c0h        ;ubica puntero de la ram del modulo lcd
                call        control     ;en la segunda línea
line2           movlw       21h         ;pregunta si termino la segunda línea
                xorwf       r0c,w       ;para ir a iniciar de nuevo el mensaje o
                btfss       status,z    ;para continuar en la segunda parte del mensaje
                goto        ciclo       ;
                movlw       080h        ;ubica puntero de ram en la primera fila
                call        control
                goto        blank       ;va a reiniciar el mensaje en blank
                end
```

```
;Proyecto  No. 3 Manejo de un LCD con los pines RA0 y RA1 para las señales de control.**
;
; este programa hace que un mensaje se repita indefinidamente
; en un modulo lcd de 2 líneas con 16 caracteres
; configurando los pines A1, A0, B4, B5, B6, B7 como salidas
; y A2, A3, A4, B0, B1, B2 como entradas
;---------- Definición de registros ----------
;
indf        equ     0h      ;para direccionamiento indirecto
tmro        equ     1h      ;contador de tiempo real
pc          equ     2h      ;contador de programa
status      equ     3h      ;registro de estados y bits de control
fsr         equ     4h      ;selecccion de bancos de memoria y registros
ptoa        equ     5h      ;puertos
ptob        equ     6h
r0c         equ     0ch     ;
r0d         equ     0dh     ;
r0e         equ     0eh     ;
r13         equ     13h     ;
z           equ     2h      ;bandera de cero
c           equ     0h      ;bandera de carry
w           equ     0h      ;para almacenar en w
r           equ     1h      ;para almacenar en el mismo registro
e           equ     1h      ;
rs          equ     0h      ;

            org     00      ;vector de reset
            goto    inicio  ;va a iniciar programa principal
            org     05h

retardo     movlw   0ffh
            movwf   r13
decre       decfsz  r13,r
            goto    decre
            retlw   0

limpia      clrf    r0c
limpi       movlw   " "
            call    dato
            incf    r0c,r
            movlw   50h
            xorwf   r0c,w
            btfss   status,z
            goto    limpi
            retlw   0

control     bcf     ptoa,rs ;esta rutina genera las señales de control
            goto    dato2   ;para escribir en el modulo lcd y
dato        bsf     ptoa,rs ;entrega el dato a ser mostrado en la pantalla
dato2       bsf     ptoa,e  ;utiliza la interface a 4 bits
```

```
                movwf   r0e
                movlw   0fh
                andwf   ptob,r
                movf    r0e,w
                andlw   0f0h
                iorwf   ptob,r
                call    retardo
                bcf     ptoa,e
                call    retardo
                bsf     ptoa,e
                movlw   0fh
                andwf   ptob,r
                swapf   r0e,w
                andlw   0f0h
                iorwf   ptob,r
                call    retardo
                bcf     ptoa,e
                call    retardo
                retlw   0

tabla           addwf   pc,r            ;mensaje que se muestra
                retlw   " "
                retlw   " "
                retlw   "R"
                retlw   "E"
                retlw   "V"
                retlw   "I"
                retlw   "S"
                retlw   "T"
                retlw   "A"
                retlw   " "
                retlw   "E"
                retlw   "&"
                retlw   "C"
                retlw   " "
                retlw   " "
                retlw   " "
                                        ;mensaje de la segunda linea

                retlw   " "
                retlw   "C"
                retlw   "E"
                retlw   "K"
                retlw   "I"
                retlw   "T"
                retlw   " "
                retlw   "-"
                retlw   " "
                retlw   "P"
                retlw   "E"
                retlw   "R"
```

```
                retlw       "E"
                retlw       "I"
                retlw       "R"
                retlw       "A"
                retlw       " "
                retlw       0

inicio          movlw       01ch        ;programacion de puertos
                tris        ptoa        ;segun el circuito
                movlw       0fh         ;
                tris        ptob        ;
begin           movlw       02h         ;inicia display a 4 bits
                call        control
                movlw       28h         ;display a 4 bits y 2 lineas
                call        control
                movlw       0ch         ;activa display
                call        control
                movlw       06h         ;hace que el mensaje permanezca fijo
                call        control

blank           call        limpia      ;borra display
muestra         clrf        r0c         ;inicia contador de caracteres
ciclo           movf        r0c,w       ;hace barrido de la tabla
                call        tabla
                call        dato
                movlw       0ffh        ;retardo entre caracteres
                movwf       r0d
reta1           call        retardo
                decfsz      r0d,r
                goto        reta1
                incf        r0c,r       ;sigue con la tabla
                movlw       11h
                subwf       r0c,w       ;pregunta si esta mostrando el mensaje de la
                btfss       status,c    ;segunda linea
                goto        ciclo       ;
                movlw       11h         ;pregunta si es la primera vez que entra
                xorwf       r0c,w       ;a la segunda linea para ir a iniciar
                btfss       status,z    ;el puntero de la ram del modulo lcd
                goto        line2       ;
linea2          movlw       0c0h        ;ubica puntero de la ram del modulo lcd
                call        control     ;en la segunda linea
line2           movlw       21h         ;pregunta si termino la segunda linea
                xorwf       r0c,w       ;para ir a iniciar de nuevo el mensaje o
                btfss       status,z    ;para continuar en la segunda parte del mensaje
                goto        ciclo       ;
                movlw       080h        ;ubica puntero de ram en la primera fila
                call        control
                goto        blank       ;va a reiniciar el mensaje en blank
                end
```

```
;********* Proyecto  No. 4 Manejo de un display de 7 segmentos ***
            ; Este programa hace un contador decimal en un display de 7 segmentos
status      equ     03h         ;registro de estados
ptoa        equ     05h         ;puerto A en dirección 05 de la RAM
ptob        equ     06h         ;Puerto B en dirección 06 de la RAM
conta       equ     0ch         ;lleva el conteo de los pulsos
loops       equ     0dh         ;utilizado en retardos (milisegundos)
loops2      equ     0eh         ;utilizado en retardos
trisa       equ     85h         ;registro de configuración del puerto A
trisb       equ     86h         ; registro de configuración del puerto B
z           equ     02h         ;bandera de cero del registro de estados
reset       org     00          ;reset en dirección 00
            goto    inicio      ;se salta al inicio del programa
            org     05          ;el programa empieza en la dirección 05
retardo     movlw   D'100'      ;subrutina de retardo de 100 milisegundos
            movwf   loops       ;contiene el numero de milisegundos de retardo
top2        movlw   D'110'      ;
            movwf   loops2      ;
top         nop
            nop
            nop
            nop
            nop
            nop
            decfsz  loops2      ;pregunta si termino 1 ms
            goto    top
            decfsz  loops       ;pregunta si termino el retardo
            goto    top2
            retlw   0
inicio      bsf     status,5    ;se ubica en el segundo banco de la RAM
            movlw   0f0h        ;se carga el registro W con 0f
            movwf   trisa       ;se programa el puerto A como salida
            movlw   0ffh        ;se carga el registro W con ff
            movwf   trisb       ;se programa el puerto B como entrada
            bcf     status,5    ;se ubica en el primer banco de la RAM
            clrf    conta       ;inicia contador en cero
ciclo       movf    conta,W     ;el valor del contador para al registro W
            movwf   ptoa        ;pasa el valor de W al puerto A (display)
            call    retardo     ;retardo esperando que suelten la tecla
pulsa       btfsc   ptob,0      ;pregunta si el pulsador esta oprimido
            goto    pulsa       ;si no lo esta continua revisándolo
            call    retardo     ;si esta oprimido retarda 100 milisegundos
            btfsc   ptob,0      ;para comprobar
            goto    pulsa       ;si no lo esta vuelve a revisar
            incf    conta       ;si lo confirma incrementa contador
            movf    conta,W     ;carga W con el valor del conteo
            xorlw   0ah         ;hace operación xor para comparar si = 0ah
            btfsc   status,z    ;prueba si el contador llego a 0ah (diez)
            goto    inicio      ;si es igual el conteo se pone en ceros
            goto    ciclo       ;si no llego a diez incrementa normalmente
            end                 ;y actualiza en display
```

```
;***Proyecto No. 5 Programa para probar la conexión de una interrupción externa**
;
; Programa que cuenta 10 pulsos y en ese momento despliega un numero
; almacenado en un registro manteniéndolo indefinidamente en
; la pantalla de un modulo lcd 2 lineas 16 caracteres
; los pulsos son introducidos a traves del pin RB0
;---------- Definición de registros ----------
;
indf        equ     0h      ;para direccionamiento indirecto
tmro        equ     1h      ;contador de tiempo real
pc          equ     2h      ;contador de programa
status      equ     3h      ;registro de estados y bits de control
fsr         equ     4h      ;selecccion de bancos de memoria y registros
ptoa        equ     5h      ;puertos
ptob        equ     6h
adres       equ     0fh
adres1      equ     15h
conta       equ     13h
trisa       equ     85h     ;programacion de los registros
trisb       equ     86h
intcon      equ     0bh
opcion      equ     81h
r0c         equ     0ch     ;
r0d         equ     0dh     ;
r0e         equ     0eh     ;
unidad      equ     10h     ;
decena      equ     11h     ;
centena     equ     12h     ;
r14         equ     14h     ;
r1b         equ     1bh     ;

                            ;bits especiales
                            ;bits del registro status
rp0         equ     5h      ;seleccion de pagina
z           equ     2h      ;bandera de cero
c           equ     0h      ;bandera de carry
w           equ     0h      ;para almacenar en w
r           equ     1h      ;para almacenar en el mismo registro
                            ;pines del puerto a
e           equ     1h      ;
rs          equ     0h      ;

;       ***********  programa principal  ************

            org     00      ;vector de reset
            clrf    adres
            goto    inicio  ;va a iniciar programa principal
            org     04h     ;el programa empieza en la direccion 04
```

```
                call        delay
                btfsc       ptob,0
                goto        sale
                btfss       intcon,1
                goto        sale
                incf        adres,r
                call        retardo
sale            bcf         intcon,1
                retfie

retardo         movlw       0ffh
                movwf       r1b
decre           decfsz      r1b,r
                goto        decre
                retlw       0

retar2          movlw       0ffh
                movwf       r14
decr2           call        retardo
                call        retardo
                decfsz      r14,r
                goto        decr2
                retlw       0

limpia          clrf        r0c
limpi           movlw       " "
                call        dato
                incf        r0c,r
                movlw       50h
                xorwf       r0c,w
                btfss       status,z
                goto        limpi
                retlw       0
                                        ;rutina que convierte binario en bcd
decimal         clrf        decena      ;borrar registros de trabajo
                clrf        centena
                movlw       d'100'
otra            subwf       unidad,r    ;restar 100 al valor inicial
                btfss       status,c    ;verifica el carry
                goto        sum         ;si es cero deja de restar 100
                incf        centena,r   ;si es 1 incrementa centena
                goto        otra        ;volver a restar
sum             addwf       unidad,r    ;sumarle 100
                movlw       d'10'
repite          subwf       unidad,r    ;restar 10 al valor
                btfss       status,c    ;verifica el carry
                goto        sum1        ;si es 0 deja de restar
                incf        decena,r    ;si es 1 incrementa decena
                goto        repite
sum1            addwf       unidad,r    ;sumarle 10 al valor
                retlw       0           ;el valor de la conversion binario a decimal
```

```
                                        ;se devuelve en los registros centena, decena y unidad
control     bcf     ptoa,rs             ;esta rutina genera las señales de control
            goto    dato2               ;para escribir en el modulo lcd y
dato        bsf     ptoa,rs             ;entrega el dato a ser mostrado en la pantalla
dato2       bsf     ptoa,e              ;utiliza la interface a 4 bits
            movwf   r0e
            movlw   0fh
            andwf   ptob,r
            movf    r0e,w
            andlw   0f0h
            iorwf   ptob,r
            call    retardo
            bcf     ptoa,e
            call    retardo
            bsf     ptoa,e
            movlw   0fh
            andwf   ptob,r
            swapf   r0e,w
            andlw   0f0h
            iorwf   ptob,r
            call    retardo
            bcf     ptoa,e
            call    retardo
            retlw   0

tabla       addwf   pc,r                ;mensaje que se muestra
            retlw   "C"
            retlw   "A"
            retlw   "N"
            retlw   "T"
            retlw   " "
            retlw   "D"
            retlw   "E"
            retlw   " "
            retlw   "P"
            retlw   "U"
            retlw   "L"
            retlw   "S"
            retlw   "O"
            retlw   "S"
            retlw   " "
            retlw   " "             ;mensaje de la segunda linea
            retlw   "I"
            retlw   "N"
            retlw   "T"
            retlw   "R"
            retlw   "O"
            retlw   "D"
            retlw   "U"
            retlw   "C"
            retlw   "I"
```

```
                retlw       "D"
                retlw       "O"
                retlw       "S"
                retlw       ":"
                retlw       " "
                retlw       " "
                retlw       " "
                retlw       " "
                retlw       0

inicio          bsf         status,rp0      ;seleccionar pagina 1
                movlw       01ch            ;configura ptoa segun el circuito
                movwf       trisa           ;
                movlw       0fh             ;configura ptob segun el circuito
                movwf       trisb           ;
                movlw       80h
                movwf       opcion
                bcf         status,rp0      ;vuelve a pagina 0
                movlw       90h
                movwf       intcon

begin           movlw       02h             ;inicia display a 4 bits
                call        control
                movlw       28h             ;display a 4 bits y 2 lineas
                call        control
                movlw       0ch             ;activa display y desactiva cursor
                call        control
                movlw       06h             ;selecciona el modo de desplazamiento
                call        control

blank           call        limpia          ;borra display
muestra         movlw       0               ;inicia contador de caracteres
                movwf       r0c
ciclo           movf        r0c,w           ;hace barrido de la tabla1
                call        tabla
                call        dato
                movlw       0ffh            ;retardo entre caracteres
                movwf       r0d

reta1           call        retardo
                decfsz      r0d,r
                goto        reta1
                incf        r0c,r           ;sigue con la tabla
                movlw       11h
                subwf       r0c,w           ;pregunta si esta mostrando el mensaje de la
                btfss       status,c        ;segunda linea
                goto        ciclo           ;
                movlw       11h             ;pregunta si es la primera vez que entra
                xorwf       r0c,w           ;a la segunda linea para ir a iniciar
                btfss       status,z        ;el puntero de la ram del modulo lcd
                goto        line2           ;
```

```
linea2      movlw       0c0h        ;ubica puntero de la ram del modulo lcd
            call        control     ;en la segunda linea
line2       movlw       21h         ;pregunta si termino la segunda linea
            xorwf       r0c,w       ;para ir a iniciar de nuevo el mensaje o
            btfss       status,z    ;para continuar en la segunda parte del mensaje
            goto        ciclo       ;

pulsa       movf        adres,w
            xorlw       d'10'
            btfsc       status,z
            goto        mide1
            goto        pulsa

delay       movlw       0fah
            movwf       conta
sigue       nop
            nop
            nop
            nop
            nop
            nop
            decfsz      conta,1
            goto        sigue
            return

mide1       movlw       d'17'
mide        movwf       unidad
            call        decimal     ;convertir el dato binario a decimal
            movlw       0cdh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        centena,w   ;primer digito = centenas
            addlw       30h
            call        dato
            movlw       0ceh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        decena,w    ;segundo digito = decenas
            addlw       30h
            call        dato
            movlw       0cfh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        unidad,w    ;segundo digito = unidades
            addlw       30h
            call        dato
            call        retar2
            goto        pulsa
            end
```

```
;***Proyecto  No. 6  Registro de 10 pulsos y despliegue en el LCD********

; Programa que cuenta 10 pulsos y en este punto despliega
; el numero 10 y ahi lo mantiene indefinidamente en la
; pantalla de un modulo lcd 2 lineas 16 caracteres
; los pulsos son introducidos a traves del pin RB0
;---------- Definición de registros ----------
;
indf        equ         0h          ;para direccionamiento indirecto
tmro        equ         1h          ;contador de tiempo real
pc          equ         2h          ;contador de programa
status      equ         3h          ;registro de estados y bits de control
fsr         equ         4h          ;selecccion de bancos de memoria y registros
ptoa        equ         5h          ;puertos
ptob        equ         6h
adres       equ         0fh
adres1      equ         15h
conta       equ         13h
trisa       equ         85h         ;programacion de los registros
trisb       equ         86h
intcon      equ         0bh
opcion      equ         81h
r0c         equ         0ch         ;
r0d         equ         0dh         ;
r0e         equ         0eh         ;
unidad      equ         10h         ;
decena      equ         11h         ;
centena     equ         12h         ;
r14         equ         14h         ;
r1b         equ         1bh         ;

                                    ;bits especiales
                                    ;bits del registro status
rp0         equ         5h          ;seleccion de pagina
z           equ         2h          ;bandera de cero
c           equ         0h          ;bandera de carry

w           equ         0h          ;para almacenar en w
r           equ         1h          ;para almacenar en el mismo registro
                                    ;pines del puerto a
e           equ         1h          ;
rs          equ         0h          ;

;       ***********  programa principal  ************

            org         00          ;vector de reset
            clrf        adres
            goto        inicio      ;va a iniciar programa principal
            org         04h         ;el programa inicia en la direccion de memoria 04
```

```
                call        delay
                btfsc       ptob,0
                goto        sale
                btfss       intcon,1
                goto        sale
                incf        adres,1
                call        retardo
sale            bcf         intcon,1
                retfie

retardo         movlw       0ffh
                movwf       r1b
decre           decfsz      r1b,r
                goto        decre
                retlw       0

retar2          movlw       0ffh
                movwf       r14
decr2           call        retardo
                call        retardo
                decfsz      r14,r
                goto        decr2
                retlw       0

limpia          clrf        r0c
limpi           movlw       " "
                call        dato
                incf        r0c,r
                movlw       50h
                xorwf       r0c,w
                btfss       status,z
                goto        limpi
                retlw       0
                                        ;rutina que convierte binario en bcd
decimal         clrf        decena      ;borrar registros de trabajo
                clrf        centena
                movlw       d'100'
otra            subwf       unidad,r    ;restar 100 al valor inicial
                btfss       status,c    ;verifica el carry
                goto        sum         ;si es cero deja de restar 100
                incf        centena,r   ;si es 1 incrementa centena
                goto        otra        ;volver a restar
sum             addwf       unidad,1    ;sumarle 100
                movlw       d'10'
repite          subwf       unidad,1    ;restar 10 al valor
                btfss       status,c    ;verifica el carry
                goto        sum1        ;si es 0 deja de restar
                incf        decena,1    ;si es 1 incrementa decena
                goto        repite
sum1            addwf       unidad,1    ;sumarle 10 al valor
                retlw       0           ;el valor de la conversion binario a decimal
```

```
                                        ;devuelve en los registros
                                        ;centena, decena y unidad
control     bcf         ptoa,rs         ;esta rutina genera las señales de control
            goto        dato2           ;para escribir en el modulo lcd y
dato        bsf         ptoa,rs         ;entrega el dato a ser mostrado en la pantalla
dato2       bsf         ptoa,e          ;utiliza la interface a 4 bits
            movwf       r0e
            movlw       0fh
            andwf       ptob,r
            movf        r0e,w
            andlw       0f0h
            iorwf       ptob,r
            call        retardo
            bcf         ptoa,e
            call        retardo
            bsf         ptoa,e
            movlw       0fh
            andwf       ptob,r
            swapf       r0e,w
            andlw       0f0h
            iorwf       ptob,r
            call        retardo
            bcf         ptoa,e
            call        retardo
            retlw       0

tabla       addwf       pc,r            ;mensaje que se muestra
            retlw       "C"
            retlw       "A"
            retlw       "N"
            retlw       "T"
            retlw       " "
            retlw       "D"
            retlw       "E"
            retlw       " "
            retlw       "P"
            retlw       "U"
            retlw       "L"
            retlw       "S"
            retlw       "O"
            retlw       "S"
            retlw       " "
            retlw       " "             ;mensaje de la segunda linea
            retlw       "I"
            retlw       "N"
            retlw       "T"
            retlw       "R"
            retlw       "O"
            retlw       "D"
            retlw       "U"
            retlw       "C"
```

```
                retlw   "I"
                retlw   "D"
                retlw   "O"
                retlw   "S"
                retlw   ":"
                retlw   " "
                retlw   " "
                retlw   " "
                retlw   " "
                retlw   0

inicio          bsf     status,rp0      ;seleccionar pagina 1
                movlw   01ch            ;configura ptoa como entradas
                movwf   trisa           ;
                movlw   0fh             ;configura ptob
                movwf   trisb           ;
                movlw   80h
                movwf   opcion
                bcf     status,rp0      ;vuelve a pagina 0
                movlw   90h
                movwf   intcon

begin           movlw   02h             ;inicia display a 4 bits
                call    control
                movlw   28h             ;display a 4 bits y 2 lineas
                call    control
                movlw   0ch             ;activa display y desactiva cursor
                call    control
                movlw   06h             ;selecciona el modo de desplazamiento
                call    control

blank           call    limpia          ;borra display
muestra         movlw   0               ;inicia contador de caracteres
                movwf   r0c
ciclo           movf    r0c,w           ;hace barrido de la tabla1
                call    tabla
                call    dato
                movlw   0ffh            ;retardo entre caracteres
                movwf   r0d
reta1           call    retardo
                decfsz  r0d,r
                goto    reta1
                incf    r0c,r           ;sigue con la tabla
                movlw   11h
                subwf   r0c,w           ;pregunta si esta mostrando el mensaje de la
                btfss   status,c        ;segunda linea
                goto    ciclo           ;
                movlw   11h             ;pregunta si es la primera vez que entra
                xorwf   r0c,w           ;a la segunda linea para ir a iniciar
                btfss   status,z        ;el puntero de la ram del modulo lcd
                goto    line2           ;
```

```
linea2      movlw       0c0h        ;ubica puntero de la ram del modulo lcd
            call        control     ;en la segunda linea
line2       movlw       21h         ;pregunta si termino la segunda linea
            xorwf       r0c,w       ;para ir a iniciar de nuevo el mensaje o
            btfss       status,z    ;para continuar en la segunda parte del mensaje
            goto        ciclo       ;

pulsa       movf        adres,w
            xorlw       d'10'
            btfsc       status,z
            goto        mide1
            goto        pulsa

delay       movlw       0fah
            movwf       conta
sigue       nop
            nop
            nop
            nop
            nop
            nop
            decfsz      conta,1
            goto        sigue
            return
mide1       movf        adres,w
            movwf       adres1
mide        movf        adres1,w    ;se carga w con el valor de adres
            movwf       unidad
            call        decimal     ;convertir el dato binario a decimal
            movlw       0cdh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        centena,w   ;primer digito = centenas
            addlw       30h
            call        dato
            movlw       0ceh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        decena,w    ;segundo digito = decenas
            addlw       30h
            call        dato
            movlw       0cfh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        unidad,w    ;segundo digito = unidades
            addlw       30h
            call        dato
            call        retar2
            goto        pulsa
            end
```

;Proyecto No. 7 Contador decimal de 255 pulsos desplegados en un módulo LCD****

*; **Programa final con el PIC16F84*** que cuenta 255 pulsos y los despliega en
; la pantalla de un modulo lcd 2x16 con forme
; se ban pulsando, se suministran a travez del RB0
;---------- *Definición de registros* ----------
;

```
indf        equ     0h      ;para direccionamiento indirecto
tmro        equ     1h      ;contador de tiempo real
pc          equ     2h      ;contador de programa
status      equ     3h      ;registro de estados y bits de control
fsr         equ     4h      ;selecccion de bancos de memoria y registros
ptoa        equ     5h      ;puertos
ptob        equ     6h
adres       equ     0fh
adres1      equ     15h
conta       equ     13h
trisa       equ     85h     ;programacion de los registros
trisb       equ     86h
intcon      equ     0bh
opcion      equ     81h
r0c         equ     0ch     ;
r0d         equ     0dh     ;
r0e         equ     0eh     ;
unidad      equ     10h     ;
decena      equ     11h     ;
centena     equ     12h     ;
r14         equ     14h     ;
r1b         equ     1bh     ;
                            ;bits especiales
                            ;bits del registro status
rp0         equ     5h      ;seleccion de pagina
z           equ     2h      ;bandera de cero
c           equ     0h      ;bandera de carry
w           equ     0h      ;para almacenar en w
r           equ     1h      ;para almacenar en el mismo registro
                            ;pines del puerto A
e           equ     1h      ;
rs          equ     0h      ;

;       ***********  programa principal  ************

            org     00      ;vector de reset
            clrf    adres
            goto    inicio  ;va a iniciar programa principal
            org     05h

            btfsc   ptob,0
            goto    sale
            btfss   intcon,1
            goto    sale
```

```
            incf        adres,r
            call        retardo
sale        bcf         intcon,1
            retfie

retardo     movlw       0ffh
            movwf       r1b
decre       decfsz      r1b,r
            goto        decre
            retlw       0

retar2      movlw       080h
            movwf       r14
decr2       call        retardo
            call        retardo
            decfsz      r14,r
            goto        decr2
            retlw       0

limpia      clrf        r0c
limpi       movlw       " "
            call        dato
            incf        r0c,r
            movlw       50h
            xorwf       r0c,w
            btfss       status,z
            goto        limpi
            retlw       0

                                    ;rutina que convierte binario en bcd
decimal     clrf        decena      ;borrar registros de trabajo
            clrf        centena
            movlw       d'100'
otra        subwf       unidad,r    ;restar 100 al valor inicial
            btfss       status,c    ;verifica el carry
            goto        sum         ;si es cero deja de restar 100
            incf        centena,r   ;si es 1 incrementa centena
            goto        otra        ;volver a restar
sum         addwf       unidad,r    ;sumarle 100
            movlw       d'10'
repite      subwf       unidad,r    ;restar 10 al valor
            btfss       status,c    ;verifica el carry
            goto        sum1        ;si es 0 deja de restar
            incf        decena,r    ;si es 1 incrementa decena
            goto        repite
sum1        addwf       unidad,r    ;sumarle 10 al valor
            retlw       0           ;el valor de la conversion binario a decimal
                                    ;se devuelve en los registros
                                    ;centena, decena y unidad
```

```
control     bcf     ptoa,rs     ;esta rutina genera las señales de control
            goto    dato2       ;para escribir en el modulo lcd y
dato        bsf     ptoa,rs     ;entrega el dato a ser mostrado en la pantalla
dato2       bsf     ptoa,e      ;utiliza la interface a 4 bits
            movwf   r0e
            movlw   0fh
            andwf   ptob,r
            movf    r0e,w
            andlw   0f0h
            iorwf   ptob,r
            call    retardo
            bcf     ptoa,e
            call    retardo
            bsf     ptoa,e
            movlw   0fh
            andwf   ptob,r
            swapf   r0e,w
            andlw   0f0h
            iorwf   ptob,r
            call    retardo
            bcf     ptoa,e
            call    retardo
            retlw   0

tabla       addwf   pc,r        ;mensaje que se muestra
            retlw   "C"
            retlw   "A"
            retlw   "N"
            retlw   "T"
            retlw   " "
            retlw   "D"
            retlw   "E"
            retlw   " "
            retlw   "P"
            retlw   "U"
            retlw   "L"
            retlw   "S"
            retlw   "O"
            retlw   "S"
            retlw   " "
            retlw   " "         ;mensaje de la segunda linea
            retlw   " "
            retlw   "I"
            retlw   "N"
            retlw   "T"
            retlw   "R"
            retlw   "O"
            retlw   "D"
            retlw   "U"
            retlw   "C"
            retlw   "I"
```

```
                retlw   "D"
                retlw   "O"
                retlw   "S"
                retlw   ":"
                retlw   " "
                retlw   " "
                retlw   " "
                retlw   0

inicio          bsf     status,rp0      ;seleccionar pagina 1
                movlw   01ch            ;configura ptoa como entradas
                movwf   trisa           ;
                movlw   0fh             ;configura ptob
                movwf   trisb           ;
                clrf    conta
                movlw   80h
                movwf   opcion
                bcf     status,rp0      ;vuelve a pagina 0
                movlw   90h
                movwf   intcon
begin           movlw   02h             ;inicia display a 4 bits
                call    control
                movlw   28h             ;display a 4 bits y 2 lineas
                call    control
                movlw   0ch             ;activa display y desactiva cursor
                call    control
                movlw   06h             ;selecciona el modo de desplazamiento
                call    control

blank           call    limpia          ;borra display
muestra         movlw   0               ;inicia contador de caracteres
                movwf   r0c
ciclo           movf    r0c,w           ;hace barrido de la tabla1
                call    tabla
                call    dato
                movlw   0ffh            ;retardo entre caracteres
                movwf   r0d
reta1           call    retardo
                decfsz  r0d,r
                goto    reta1
                incf    r0c,r           ;sigue con la tabla
                movlw   11h
                subwf   r0c,w           ;pregunta si esta mostrando el mensaje de la
                btfss   status,c        ;segunda linea
                goto    ciclo           ;
                movlw   11h             ;pregunta si es la primera vez que entra
                xorwf   r0c,w           ;a la segunda linea para ir a iniciar
                btfss   status,z        ;el puntero de la ram del modulo LCD
                goto    line2           ;
linea2          movlw   0c0h            ;ubica puntero de la ram del modulo LCD
                call    control         ;en la segunda linea
```

```
line2       movlw       21h         ;pregunta si termino la segunda linea
            xorwf       r0c,w       ;para ir a iniciar de nuevo el mensaje o
            btfss       status,z    ;para continuar en la segunda parte del mensaje
            goto        ciclo       ;


pulsa       btfsc       ptob,0
            goto        pulsa
            call        retardo
            btfsc       ptob,0
            goto        pulsa
            incf        conta
            movf        conta,w
            xorlw       0ffh
            btfsc       status,z
            goto        inicio
            movf        conta,w

mide        movwf       unidad
            call        decimal     ;convertir el dato binario a decimal
            movlw       0cdh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        centena,w   ;primer digito = centenas
            addlw       30h
            call        dato
            movlw       0ceh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        decena,w    ;segundo digito = decenas
            addlw       30h
            call        dato
            movlw       0cfh        ;ubico puntero de ram donde va el numero leido
            call        control
            movf        unidad,w    ;segundo digito = unidades
            addlw       30h
            call        dato
            call        retar2
            goto        pulsa
            end
```

```
;*****************    Proyecto 1.1 Conexión de Leds y un Dipswitch************
;
;*********************Primer ejercicio con el PIC16F872*****************
;Programa que enciende 8 Leds, a través de dos pulsadores conectados a RA0 y RA1*********

        list    p=16f872                ; para definir procesador
        #include <p16f872.inc>          ;procesador para definiciones de variables

        __CONFIG _WDT_OFF & _BODEN_OFF & _PWRTE_ON & _XT_OSC &
_LVP_OFF

;***** DEFINICIÓN DE VARIABLES

DATA1           equ             0x20
DATA2           equ             0x21
DATA3           equ             0x22
DATA4           equ             0x23
DATA5           equ             0x24
DATA6           equ             0x25
DATA7           equ             0x26
DATA8           equ             0x27
DATA9           equ             0x28
DATA10          equ             0x29
DATA11          equ             0x2A
DATA12          equ             0x2B
DATA13          equ             0x2C
DATA14          equ             0x2D
DATA15          equ             0x2E
DATA16          equ             0x2F
COUNT1          equ             0x30
COUNT2          equ             0x31
COUNT3          equ             0x32
COUNT4          equ             0x33
TEMP1           equ             0x34

                ORG     0x000                   ; vector de reset
                goto    Main                    ; va ha iniciar programa

Delay2          movlw           0x28            ;Función para tardar 20 msegundos.
                movwf           COUNT4
Loop4           movlw           0xA4
                movwf           COUNT3
Loop3           decfsz          COUNT3, f
                goto            Loop3
                nop
                nop
                decfsz          COUNT4, f
                goto            Loop4
                return

Delay1          movlw           0xF4;Función para tardar el despliegue del Led
```

```
                movwf   COUNT2
Loop2           movlw   0xF4
                movwf   COUNT1
Loop1           nop
                clrf    COUNT3          ;inicializa contador para delay2
                clrf    COUNT4
                btfsc   PORTA, 0        ;prueba si el boton  P esta presionado
                goto    Continu
                call    Delay2
                btfsc   PORTA, 0
                goto    Continu         ;si el boton P esta presionado
                call    Progmode        ;regresa a subrutina progmode
Continu         nop                     ;el resto continua adelante
                nop
                nop
                nop
                nop
                nop
                nop
                decfsz  COUNT1, f
                goto    Loop1
                decfsz  COUNT2, f
                goto    Loop2
                return
Progmodemovf    FSR, w                  ; Programando el modo para el LED
                movwf   TEMP1
                movlw   0x08
                movwf   PORTA           ;apaga el Led del pin2, enciende el
                                        ;Led del  pin3
                movlw   DATA1
                movwf   FSR             ;pone en dirección  FSR
Loop6           btfsc   PORTA, 1        ;esta el botón N presionado?
                goto    Loop6
                call    Delay2
                btfsc   PORTA, 1
                goto    Loop6
Loop8           btfss   PORTA, 1        ;prueba si el botón esta suelto
                goto    Loop8
                movf    PORTB, w        ;si lo esta, guarda el byte en la dirección
                movwf   INDF
                movf    FSR, w
                sublw   DATA16          ;
                btfsc   STATUS, Z       ;
                goto    Here
                incf    FSR, f          ;si no, suba un GPR
                goto    Loop6           ;y continue
Here            movf    TEMP1, w
                movwf   FSR             ;pone el valor original de FSR
                movlw   0x04            ;enciende Leds
                movwf   PORTA
                return                  ;y retorna
```

```
;**********************Inicia Programa*********************************

Main        bcf     STATUS, RP1     ;Selecciona Bank0
            bcf     STATUS, RP0
            clrf    PORTA           ;limpia puertos
            clrf    PORTB
            clrf    PORTC
            bsf     STATUS, RP0     ;Selecciona Bank1
            movlw   0xFF            ;configura portB como entradas
            movwf   TRISB
            clrf    TRISC           ;configura portC como salida
            movlw   0x03            ;pins0:1 entradas, pins2:3 salidas
            movwf   TRISA
            movlw   0x06            ; configura PortA como digital
            movwf   ADCON1
            bcf     OPTION_REG, 7   ;habilita pull ups
            bcf     STATUS, RP0     ;va a  Bank0
            clrf    DATA1           ;limpia registros
            clrf    DATA2
            clrf    DATA3
            clrf    DATA4
            clrf    DATA5
            clrf    DATA6
            clrf    DATA7
            clrf    DATA8
            clrf    DATA9
            clrf    DATA10
            clrf    DATA11
            clrf    DATA12
            clrf    DATA13
            clrf    DATA14
            clrf    DATA15
            clrf    DATA16
            bsf     PORTA, 2        ;Luz del LED indicando modo del
Start       movlw   DATA1           ; display
            movwf   FSR
Loop5       clrf    COUNT1          ;reset para retardos
            clrf    COUNT2
            movf    INDF, w
            movwf   PORTC           ;despliega GPR contenidos en PortC
            call    Delay1          ;retardo para display
            incf    FSR, f          ;mueve una dirección de GPR
            nop
            movf    FSR, w          ;pone  FSR en el registro W.
            sublw   DATA16          ;substrae la direccón para
                                    ;la dirección mas alta
            btfss   STATUS, Z       ;si el resultado es no = 0
            goto    Loop5           ;continue…..
            goto    Start           ;si el resultado = 0
                                    ; va a empezar
```

```
                END
;*******Proyecto 1.2  Programa  para la conexión a 8 bits***********
;
;***********Segundo ejercicio con el PIC16F872**********************
;Programa para la conexión a 8 bits entre el microcontrolador y el módulo LCD*********

        list    p=16f872                ; para definir procesador
        #include <p16f872.inc>          ;procesador para definiciones de variables

        __CONFIG _WDT_OFF & _BODEN_OFF & _PWRTE_ON & _XT_OSC &
_LVP_OFF

;***** DEFINICIÓN DE VARIABLES

COUNTX          equ             0x20
DELAYX          equ             0x21
TEMP            equ             0x22
RS              equ             0
RW              equ             1
E               equ             2

                ORG             0x000   ; vector de reset
                goto            Main    ; va a iniciar programa

;***********Rutina de retardo**************  0.5 ms tiempo de retardo para Xdelay
                                        ;requiere multiplicador en registro W
Delay500
                movlw           0xA5
                movwf           COUNTX
Delay5loop      decfsz          COUNTX, f
                goto            Delay5loop
                return
Xdelay          movwf           DELAYX
Xdelloop        call            Delay500
                decfsz          DELAYX, f
                goto            Xdelloop
                return

;**************Checa la badera ocupada***********
Busyflag
                bsf             STATUS, RP0     ;bank 1
                movlw           0xFF            ;cambia el PortC para entradas
                movwf           TRISC
                bcf             STATUS, RP0     ;bank 0
Busy            bcf             PORTA, E        ;habilita
                bcf             PORTA, RS       ;commando
                bsf             PORTA, RW       ;lectura
                bsf             PORTA, E        ;habilita
                btfsc           PORTC, 7        ;checa la bandera ocupada
                goto            Busy            ;todavia esta ocupada
                bcf             PORTA, E        ;habilita
```

```
                bsf             STATUS, RP0     ;bank 1
                clrf            TRISC           ;limpia salidas del Puerto C
                bcf             STATUS, RP0     ;bank 0
                return

;*****************escribir byte de comando*************
Wcommand
                movwf           TEMP
                call            Busyflag
                movf            TEMP, w
Command         bcf             PORTA, RS       ;comando
                bcf             PORTA, RW       ;escribir
                bsf             PORTA, E        ;habilita
                movwf           PORTC           ;envia comando
                bcf             PORTA, E        ;habilita
                return

;*****************escribe el byte de datos*****************
Wdata           movwf           TEMP
                call            Busyflag
                movf            TEMP, w
                bsf             PORTA, RS       ;dato
                bcf             PORTA, RW       ;escribir
                bsf             PORTA, E        ;habilita
                movwf           PORTC           ;envia byte de datos
                bcf             PORTA, E        ;habilita
                return

;**************lee el byte de datos******************
Rdata
                movwf           TEMP
                call            Busyflag
                movf            TEMP, w
                bsf             STATUS, RP0     ;bank 1
                movlw           0xFF            ;cambia PortC para las entradas
                movwf           TRISC
                bcf             STATUS, RP0     ;bank 0
                bsf             PORTA, RS       ;dato
                bsf             PORTA, RW       ;lee
                bsf             PORTA, E        ;habilita
                movf            PORTC, w        ; reciba byte de los datos
                bcf             PORTA, E        ;habilita
                bsf             STATUS, RP0     ;bank 1
                clrf            TRISC           ;limpia las salidas del PuertoC
                bcf             STATUS, RP0     ;bank 0
                return

;***************Inicializa LCD*******************
InitLCD
                movlw           0x28            ;40
                call            Xdelay          ;tiempo de retardo
```

```
                                        ; 500us x 40 = 20ms
                movlw       0x38        ;función de juego de comandos
                call        Command     ;envia comando
                movlw       0x09        ;9
                call        Xdelay      ;4.5 ms de reatardo
                movlw       0x38
                call        Command
                movlw       0x01        ;1
                call        Xdelay      ;500 us de retardo
                movlw       0x38
                call        Command
                call        Wcommand    ;empieza checando la bandera ocupada
                                        ;interface a 8 bit, 2 linias, 5x7 conjunto
                                        ;de caracteres
                movlw       0x08        ;apaga display
                call        Wcommand
                movlw       0x01        ;aclara display
                call        Wcommand
                movlw       0x06        ;Modo de entrada derecho de
                                        ;incremento el despliegue no cambió
                call        Wcommand
                movlw       0x0F        ;display encendido, cursor parpadeando
                call        Wcommand
                movlw       0x01        ;Retorno a Casa
                call        Wcommand
                return

;**************commandos aleatorios****************
Opt1
                movlw       0x28
                call        Xdelay      ;20ms
                btfsc       PORTA, 3
                return

Wait1           btfss       PORTA, 3    ;descarga del boton
                goto        Wait1
                movf        PORTB, w
                call        Wcommand
                return

Opt2            movlw       0x28
                call        Xdelay      ;20ms
                btfsc       PORTA, 4
                return

Wait2           btfss       PORTA, 4    ;descarga del boton
                goto        Wait2
                movf        PORTB, w
                call        Wdata
                return
Gohome          movlw       0x28
```

```
                call            Xdelay          ;20ms
                btfsc           PORTA, 5
                return

Wait3           btfss           PORTA, 5        ;descarga del boton
                goto            Wait3
                movlw           0x02
                call            Wcommand
                return

;**************Inicia  programa*******************
Main                                            ;Inicializa el Pic
                bcf             STATUS, RP1     ;Selecciona Bank0
                bcf             STATUS, RP0
                clrf            PORTA           ;limpia los puertos
                clrf            PORTB
                clrf            PORTC
                bsf             STATUS, RP0     ;Selecciona Bank1
                movlw           0xFF            ;configura portB entradas
                movwf           TRISB
                clrf            TRISC           ;configura portC como salida
                movlw           0x38            ;pines 1-3 salidas, pines 4-6 entradas
                movwf           TRISA
                movlw           0x06            ; configura PortA como digital
                movwf           ADCON1
                bcf             OPTION_REG, 7   ;No hay tiempo de interrupción
                                                ;habilita las pullups
                bcf             STATUS, RP0     ;va a  Bank0

;*****Inicializa  LCD ******************

                call            InitLCD         ;inicializa display
                movlw           0x52            ;R
                call            Wdata
                movlw           0x65            ;e
                call            Wdata
                movlw           0x61            ;a
                call            Wdata
                movlw           0x64            ;d
                call            Wdata
                movlw           0x79            ;y
                call            Wdata
                movlw           0x21            ;!
                call            Wdata
Input           btfss           PORTA, 3        ; Prueba el boton de orden
                call            Opt1
                btfss           PORTA, 4        ; prueba el boton de caracter
                call            Opt2
                btfss           PORTA, 5        ; prueba el boton de retorno a casa
                call            Gohome
                goto            Input
```

```
                END                     ;

;*****************    Proyecto 1.3 Conversión Analógico - Digital*************
;***********************************************
; ;************programa final con el PIC16F872**********************
;*****Toma el valor de voltaje y muestra una salida digital ******

        list    p=16f872                ; para definir procesador
        #include <p16f872.inc>          ;procesador para definiciones de variables

        __CONFIG _WDT_OFF & _BODEN_OFF & _PWRTE_ON & _XT_OSC &
_LVP_OFF

;***** VARIABLE DEFINITIONS

COUNTX          equ             0x20
DELAYX          equ             0x21
TEMP            equ             0x22
HIBYTE          equ             0x23
LOBYTE          equ             0x24
TEMPX           equ             0x25
MULCND          equ             0x26
R2              equ             0x29
R1              equ             0x2A
R0              equ             0x2B
WTEMP           equ             0x2C
STEMP           equ             0x2D
TMLO            equ             0x30
TMMID           equ             0x31
TMHI            equ             0x32
VOLT            equ             0x33
RS              equ             1
RW              equ             2
E               equ             3

                ORG             0x000           ;Vector de reset
                goto            Main            ;va a iniciar programa

                ORG             0x004           ;vector de interrupción
                movwf           WTEMP
                swapf           STATUS, w
                movwf           STEMP
                btfsc           INTCON, T0IF    ;prueba para T0
                goto            Timer           ;volver a la rutina del tiempo

Forever         nop
                goto            Forever
Timer           decfsz          TMMID, f                        ;substraer  256
                goto            EndISR                  ;Siga
                movf            TMHI, f                 ; prueba si TMHI = 0
                btfsc           STATUS, Z               ;si MID = HI = 0
```

```
                goto    Event                   ; haga Evento.
                decf    TMHI, f
                goto    EndISR
Event           call    Mpy8x8          ;convierte voltaje
                call    B2BCD           ;convierte a BCD
                movlw   0xC0
                call    Wcommand        ; indicador fijo
                movf    R0, w
                andlw   0X0F
                btfsc   STATUS, Z       ;prueba si el numero cero esta alto
                goto    Zip
                addlw   0x30            ;conversion ASCII
                call    Wdata
                goto    Rest
Zip             movlw   0x20
                call    Wdata           ;espacio
Rest            swapf   R1, w
                andlw   0x0F
                addlw   0x30
                call    Wdata
                movf    R1, w
                andlw   0x0F
                addlw   0x30
                call    Wdata
                swapf   R2, w
                andlw   0x0F
                addlw   0x30
                call    Wdata
                movf    R2, w
                andlw   0x0F
                addlw   0x30
                call    Wdata
                movlw   0x6D            ;m
                call    Wdata
                movlw   0x56            ;V
                call    Wdata
                movlw   0x20            ;agrega 500,000 para registros TM
                addwf   TMLO, f
                btfsc   STATUS, C       ;recuerda para carry
                incf    TMMID, f        ;agrega uno mas a mid
                movlw   0xA1
                addwf   TMMID, f
                movlw   0x07
                movwf   TMHI
EndISR          swapf   STEMP, w
                movwf   STATUS
                swapf   WTEMP, f
                swapf   WTEMP, w
                bcf     INTCON,T0IF
                retfie                  ; retorno para la interrupción
```

```
;************rutina de retardo*******        0.5 ms tiempo de retardo para Xdelay
                                            ;requiere multiplicador en registro W
Delay500        movlw       0xA5
                movwf       COUNTX
Delay5loop      decfsz      COUNTX, f
                goto        Delay5loop
                return
Xdelay          movwf       DELAYX
Xdelloop        call        Delay500
                decfsz      DELAYX, f
                goto        Xdelloop
                return

;**************Checks for busy flag************
Busyflag
                bsf         STATUS, RP0     ;bank 1
                movlw       0xFF            ;cambia PortC a entradas
                movwf       TRISC
                bcf         STATUS, RP0     ;bank 0

Busy            bcf         PORTA, E        ;habilita
                bcf         PORTA, RS       ;comando
                bsf         PORTA, RW       ;leer
                bsf         PORTA, E        ;habilita
                btfsc       PORTC, 7        ;checa la bandera ocupada
                goto        Busy            ;todavia esta ocupada?
                bcf         PORTA, E        ;habilita
                bsf         STATUS, RP0     ;bank 1
                clrf        TRISC           ;limpia las salidas del PuertoC
                bcf         STATUS, RP0     ;bank 0
                return

;**************escribe bytes de comandos*************
Wcommand

                movwf       TEMP
                call        Busyflag
                movf        TEMP, w
Command         bcf         PORTA, RS       ;comando
                bcf         PORTA, RW       ;escribir
                bsf         PORTA, E        ;habilita
                movwf       PORTC           ;envia comando
                bcf         PORTA, E        ;habilita
                return

;**************escribe bayte de datos*****************
Wdata
                movwf       TEMP
```

```
                call            Busyflag
                movf            TEMP, w
                bsf             PORTA, RS       ;datos
                bcf             PORTA, RW       ;escribe
                bsf             PORTA, E        ;habilita
                movwf           PORTC           ;envia byte de datos
                bcf             PORTA, E        ;habilita
                return

;***************Inicializa LCD*******************

InitLCD         movlw           0x28            ;40
                call            Xdelay          ;tiempo de retardo 500us x 40 =
20ms
                movlw           0x38            ;function de commando fijo
                call            Command         ;enviar comando
                movlw           0x09            ;9
                call            Xdelay          ;4.5 ms de retardo
                movlw           0x38
                call            Command
                movlw           0x01            ;1
                call            Xdelay          ;500 us de retardo
                movlw           0x38
                call            Command
                call            Wcommand        ;checa la bandera ocupada
                                                ;interface 8 bit, 2 linias, 5x7
caracteres
                movlw           0x08            ;apaga display
                call            Wcommand
                movlw           0x01            ;aclara display
                call            Wcommand
                movlw           0x06            ;
                                                ;incremento, el despliegue no cambio
                call            Wcommand
                movlw           0x0F            ;display encendido, cursor
parpadeando
                call            Wcommand
                movlw           0x01            ;retorno a casa
                call            Wcommand
                return

;*********multiplicación 8bit x 8bit ****  MULPLR x MPLCND --> HIBYTE;LOBYTE
                                                ;MULPLR = VOLT
Mpy8x8
                clrf            HIBYTE
                clrf            LOBYTE
                clrf            COUNTX
                bsf             COUNTX, 3
                movf            MULCND, w
                bcf             STATUS, C
LoopX           btfsc           VOLT, 0
```

```
                addwf       HIBYTE, f
                bcf         STATUS, C
                rrf         HIBYTE, f
                rrf         LOBYTE, f
                bcf         STATUS, C
                rrf         VOLT, f
                decfsz      COUNTX, f
                goto        LoopX
                return

;*********16bit binario a 5 digital BCD (16bit --> BCD 20bit)**
;           ACCa --> R2;R1;R0    R2
;           ACCa = HIBYTE/LOBYTE
;************************************************************
B2BCD
                bcf         STATUS, C
                clrf        COUNTX
                bsf         COUNTX, 4       ;pone contador a 16
                clrf        R0
                clrf        R1
                clrf        R2

Loop16a         rlf         LOBYTE, f
                rlf         HIBYTE, f
                rlf         R2, f
                rlf         R1, f
                rlf         R0, f
                decfsz      COUNTX, f
                goto        Adjdec
                return
Adjdec          movlw       R2      ; cargue como indicador de dirección indirecto
                movwf       FSR
                call        AdjBCD
                incf        FSR, f
                call        AdjBCD
                incf        FSR, f
                call        AdjBCD
                goto        Loop16a

AdjBCD          movf        INDF, w
                addlw       0x03
                movwf       TEMPX
                btfsc       TEMPX, 3        ; prueba si el resultado > 7
                movwf       INDF
                movf        INDF, w
                addlw       0x30
                movwf       TEMPX
                btfsc       TEMPX, 7        ;prueba si el resultado > 7
                movwf       INDF            ;
                return
```

```
Main                                        ;Initcializa el PIC
                bcf         STATUS, RP1     ;Selecciona Bank0
                bcf         STATUS, RP0
                clrf        PORTA           ;limpia puertos
                clrf        PORTB
                clrf        PORTC
                bsf         STATUS, RP0     ;Selecciona Bank1
                movlw       0xFF            ;configura portB como entradas
                movwf       TRISB
                clrf        TRISC           ;configura  portC como salidas
                movlw       0x01            ;pines 2-6 salidas, pin 1 entradas
                movwf       TRISA
                movlw       0x0E            ; RA0 analógico, Vdd de referencia
                movwf       ADCON1
                clrf        INTCON          ;limpia baderas
                movlw       0x01            ;asigne TMR0, y
                movwf       OPTION_REG      ; prescala de 1:4
                bcf         STATUS, RP0     ;Bank0
                clrf        TMR0            ;aclara el  tiempo
                bsf         STATUS, RP0     ;Bank1
                movlw       0x09            ;
                movwf       OPTION_REG      ;
                clrwdt                      ;aclara WDT y prescaler.
                movlw       0x08            ;escala de 1:1 y
                movwf       OPTION_REG      ; el WDT  que asignó
                bcf         STATUS, RP0     ;va a  Bank0
                movlw       0x41            ;8Tosc, canal 1, AD on
                movwf       ADCON0

;*****LCD Initialization******************

                call        InitLCD         ;inicializa display
                movlw       0x20            ;poner salida 500,000 en registro TM
                movwf       TMLO
                movlw       0xA1
                movwf       TMMID
                movlw       0x07
                movwf       TMHI
                movlw       0x56            ;V
                call        Wdata
                movlw       0x6F            ;o
                call        Wdata
                movlw       0x6C            ;l
                call        Wdata
                movlw       0x74            ;t
                call        Wdata
                movlw       0x61            ;a
                call        Wdata
                movlw       0x67            ;g
                call        Wdata
                movlw       0x65            ;e
```

```
                call        Wdata
                movlw       0x20            ;espacio
                call        Wdata
                movlw       0x6D            ;m
                call        Wdata
                movlw       0x65            ;e
                call        Wdata
                movlw       0x74            ;t
                call        Wdata
                movlw       0x65            ;e
                call        Wdata
                movlw       0x72            ;r
                call        Wdata
                movlw       0xC0
                call        Wcommand        ;ubica puntero en la segunda línea
                movlw       0x62            ;mueve 98 decimal dentro MULCND
                movwf       MULCND
                clrf        TMR0
                movlw       0xB0            ;interrupciones fijos GIE, TMR0IE
                movwf       INTCON
Start
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop                         ;21 usecond de retardo para el AD
                bsf         ADCON0, 2       ;va al bit fijo
Wait            btfss       PIR1, ADIF
                goto        Wait
                bcf         PIR1, ADIF
                movf        ADRESH, w
```

```
movwf       VOLT
goto        Start
END
```

ANEXOS