ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

CONSTRUCCIÓN DE UN CIRCUITO ELECTRÓNICO DIGITAL QUE PERMITA MEDIR LA CANTIDAD DE GASOLINA EXISTENTE EN EL TANQUE DE UN AUTOMOTOR.

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO EN ELÉCTRONICA Y TELECOMUNICACIONES

SANTIAGO IVÁN MONTESDEOCA ESPÍN
santi_montes28@hotmail.com

DIRECTOR: ING. PABLO LÓPEZ
pablo.lopez@epn.edu.ec

Quito, Febrero 2012

DECLARACIÓN

Yo, **Santiago Iván Montesdeoca Espín**, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad, intelectual correspondiente a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

_____

SANTIAGO IVÁN MONTESDEOCA ESPÍN

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por, Santiago Iván Montesdeoca Espín, bajo mi supervisión.

_____

Ing. Pablo López M.
DIRECTOR DE PROYECTO

DEDICATORIA

Primeramente a Dios fuente de inspiración en mis momentos de angustias, esmero, dedicación, aciertos y reveses, alegrías y tristezas que caracterizaron el terminar por este camino que hoy veo realizado, sin cuyo empuje no hubiese sido posible.

A mis padres *Nelson Montesdeoca y Concepción Espín* por ser ellos arboles principales que me acobijó bajo su sombra dándome así la fuerza para seguir caminando y lograr alcanzar un escalón de la vida con este trabajo que conjuntamente con ellos lo he logrado. Dios les bendiga, les de salud y mucha vida para poder retribuir un poco de lo que me han dado. Los amo para ustedes este logro y todos los que me faltan por alcanzar, este es solo el comienzo de una vida llena de éxitos para ustedes. Gracias por su paciencia y confiar en mí. El que obtener superación hoy es el resultado tener excelentes padres. Los quiero muchos.

A mis Hermanos *Cumanda, Martin, Sandro, Claudia* que siempre tengamos en cuenta que todo lo que nos propongamos en la vida lo podemos lograr si trabajamos fuerte y continuamente con rectitud, seguir adelante a ***Paul*** quien desde el cielo me está cuidando en cada paso que doy y me impulsa a seguir adelante en la vida.

A mi Tutor de Proyecto ***Ing. Pablo López*** que siempre estuvo presto en ayudarme en la elaboración de de este proyecto y saber guiarme.

Para **Alicia** quien está pendiente de mí y por el inmenso amor que me sabe dar, con su sonrisa y alegría me demuestra que vale esforzarse y ser persona útil en la sociedad.

AGRADECIMIENTO

Son numerosas las personas a las que debo agradecer por ayudarme en el logro de mi carrera, es demasiado poco, decir gracias, pero en el fondo de mi ser eternamente les estaré agradecidos y siempre presto a tenderles una mano cuando así lo requieran:

A mi Padre Nelson.

A mi Madre María Concepción.

A mis Hermanas Cumanda, Claudia.

A mis Hermanos Martin, Paul, Sandro.

A mi novia Alicia por tenerme a su lado

A mi Tío Gonzalo y Tía Rosa quienes me

acogieron en su hogar cuando llegue a

Quito para dar mis primeros pasos en la E.P.N.

A todos mis profesores de la ESFOT

A mi Tutor Ing. Pablo López

A todos mis compañeros y amigos de Aula

RESUMEN

El presente trabajo que se ha realizado, con el afán de desarrollar un circuito digital que permita medir y cuantificar la cantidad de combustible existente dentro del tanque de combustible de un automotor de forma de una manera

clara y sencilla, mediante la programación de un micro-controlador para ser visualizado los diferentes parámetros en un LCD . Este circuito podrá ser remplazado por sistema analógico que en algunos automotores que aun los poseen, este circuito se puede ser usado para verificar si la cantidad de combustible que se está ingresando así el tanque es el correcto y su valor que representa en dinero.

En el capítulo I, se hace una explicación introductora de la composición de los combustibles, los diferente tipo de sensores que existen para medir las magnitudes físicas que existen en la naturaleza, se realiza una descripción fundamental y básica del programa que se utilizo para la programación del micro-controlador (PROTON) y el programa para la elaboración del circuito impreso (PROTEUS) y finalmente se describe la estructura y funcionamiento del micro-controlador, de LCD y el teclado matricial respectivamente.

En el capítulo II, se enfoca la descripción y funcionamiento de cada una de las partes electrónica que constituye a este proyecto, elaboración del programa que permitirá ser programado el micro-controlador para sus funciones especificas, montaje y calibración en la placa electrónica de cada uno de los elementos electrónicos.

En el capítulo III se hace referencia a las conclusiones y recomendaciones que se obtuvo en la realización del circuito digital que permita medir la cantidad gasolina existente dentro del tanque de combustible del auto motor.

INDICE

CAPITULO 1

INFORMACIÓN TEÓRICA

1.1 OBTENCIÓN DEL PETRÓLEO Y LA GASOLINA

El **petróleo** es el combustible más importante en la historia de la humanidad, es un recurso natural no renovable que aporta el mayor porcentaje del total de la energía que se consume en el mundo.

El petróleo es un líquido negro, espeso y maloliente que se encuentra a 3 ó 4 Km de profundidad. Es una mezcla de diferentes sustancias denominadas hidrocarburos.

1.1.2 Procesos para obtener los derivados del petróleo entre ellos la gasolina

Se filtra el crudo para quitarle impurezas, y se lo somete a un proceso de decantación mientras se le agregan aglutinantes para secuestrar más impurezas. Luego se filtra de nuevo y se lo mete en lo que se llama "**Torre de crackeo**" donde se lo calienta lentamente a alta presión y en un ambiente controlado para que no se inflame. Al evaporarse, se condensa como un hidrocarburo más liviano, del que se obtienen grasas y aceites pesados. Estos vuelven a destilarse y vuelven a condensarse en aceites más livianos. Se repite y se obtiene Fuel Oil, que se utiliza como combustible de barcos, locomotoras. Se repite el proceso, y se obtiene Gas Oil, que se usa como combustible automotriz, y en maquinaria pesada. Vuelve a destilarse y se obtiene el Keroseno, luego la nafta, gasolina, y en el último grado de destilación, la aeronafta, que se utiliza en las turbinas de aviación. En cada paso, al producto de la destilación se le agregan aditivos, mejoradores y limpiadores de combustión, anilinas y aromatizantes, antidetonantes y estabilizadores químicos, según su utilización especifica. También se le agregan homogeneizadores para que mantengan su performance en diferentes condiciones. En cuanto a la nafta llamada "súper" y a la gasolina, se las somete

a un tratamiento de purificación especial y se le agrega un antidetonante extra, lo que hace es dificultar o impedir que la nafta se encienda fácilmente.

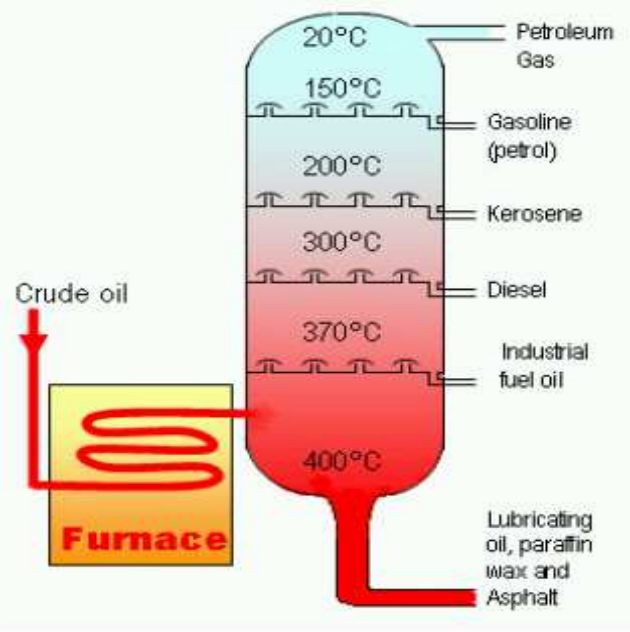


Figura 1.1 procesamiento de la gasolina

Así pues, la gasolina es la mezcla de hidrocarburos líquidos más ligeros que se usa como combustible en motores de combustión interna, como por ejemplo, en los motores de los automóviles. Las gasolinas obtenidas de estas maneras no se pueden emplear como combustible así como están, ya que se deben mezclar con otros compuestos que mejorarán el rendimiento

1.1.2.1 Química de la Gasolina

La gasolina es una mezcla de hidrocarburos líquidos ligeros. Los hidrocarburos son compuestos que sólo contienen dos elementos: el carbono y el hidrógeno. Hay una variedad de hidrocarburos y por eso se agrupan en familias. Una familia es la de los **alcanos**, que son hidrocarburos que tienen los átomos de carbono unidos por enlaces covalentes simples, la mayoría de los hidrocarburos del petróleo son de esta familia.

Como se puede observar en la *figura 1.1* 1) el carbono puede formar cuatro enlaces 2) y los átomos de carbono se unen entre sí 3) formando cadenas. Después, por ejemplo, forman hidrocarburos como la *figura 1.1*, en este caso

es el octano (C8H18). Algunas propiedades de los alcanos cambian en función de la longitud de la cadena de carbono. Al crecer la cadena, la temperatura de ebullición también crece y por tanto los alcanos son cada vez menos volátiles.

Así, los alcanos con cadenas cortas son gases a temperatura ambiente. Los alcanos con cadenas que contengan entre 5 y 19 átomos de carbono son líquidos y los que contengan más de 20 átomos de carbono son sólidos.

La gasolina contiene alcanos con cadenas entre 5 y 10 átomos de carbono en las moléculas o cadenas de hidrocarburos.

1.1.2.2 Tipos de Gasolina y sus Características

La característica distintiva de cada gasolina la constituye la resistencia a la detonación, que viene determinada por el octanaje, la gasolina Súper tiene un octanaje mayor a la de la gasolina Extra. Por lo tanto, el octanaje sirve para determinar el grado de resistencia a la detonación, es decir, si la gasolina tiene poco octanaje, en la cámara de combustión de un motor se producen detonaciones a destiempo en la cámara de combustión. Los índices de octanaje de cada gasolina son los siguientes: la gasolina Súper tiene 98 octanos, la gasolina Extra tiene 95. El índice de octanaje de una gasolina es una medida de su capacidad antidetonante**,** así pues, va en función del número de octanos de cada una de ellas.

1.1.3 Propiedades de los Combustibles

Como derivados del petróleo crudo, los combustibles están formados básicamente por hidrocarburos. Pueden contener además, O_2, S, N,.. etc. El combustible empleado debe poseer características muy importantes para obtener la combustión:

- Volatilidad
- Numero de octanos
- Contenido de azufre
- Presión de vapor Reid
- Densidad

1.1.3.1 Volatilidad*:*

Es la tendencia a pasar a fase vapor en unas condiciones determinadas. El combustible en este caso, cuando este en un motor. El combustible debe tener un punto de destilación bajo, para permitir un buen arranque en frío. La volatilidad se determina con la curva de destilación. El combustible líquido es una fracción de la destilación del crudo de petróleo. Se tiene una u otra cosa dependiendo de donde se corta en la destilación, es decir, de las temperaturas donde se recoja en el intervalo de destilación. No se tiene una temperatura única, sino que a medida que el volumen recogido va aumentando va variando la temperatura.

La temperatura va ascendiendo porque se tiene otros compuestos con más átomos de Carbones en la cadena que se van evaporando poco a poco. Después se condensan al ponerse en contacto con las paredes frías y se recogen. Así, cuanto mayor sea la temperatura, se evaporarán los más pesados, los de mayor número de átomos de carbono en la cadena.

Para asegurar la volatilidad hay que tener en cuenta las propiedades del combustible, diseño del motor y materiales con los que está fabricado. La eficaz utilización de un combustible en un motor depende del diseño del motor (para que haya un mayor rendimiento), de la preparación del combustible para que el motor tenga mayor potencia y rendimiento. Para que esto se cumpla el combustible que sale directamente de la destilación no tiene estos requisitos, por lo que necesita un tratamiento posterior para que se cumplan esos objetivos. Se deben añadir, aditivos y otros elementos

1.1.3.2 Octanaje

Es la medida de la tendencia de la gasolina a la *detonación* (sonido metálico que percibimos acompañado de recalentamiento, pérdida de potencia). Nos sirve el octanaje para *clasificar* las gasolinas. Para medirlo se usa un motor de dimensiones especificadas, monocilíndrico, en el que se puede variar su relación de compresión. La escala empleada para la medida del octanaje es totalmente arbitraria pero con dos puntos de referencia:

- Comportamiento del hepteno: índice 0
- Comportamiento del iso-octano: índice 100

El nº de octano es el porcentaje de *iso-octano* en una mezcla de heptano e iso-octano que presenta las mismas características detonantes que el combustible que estemos ensayando.

Existen dos procedimientos para medir el índice octano:

- Método Motor D-2700: Se mide el comportamiento de un motor a 'gran' velocidad
- Método Research D-2699: Se mide el comportamiento de un motor a baja velocidad

Para las gasolinas de automoción hay tres números de octano:

NOM: Número de octano MOTOR

NOR: Número de octano RESEARCH

RON (RDON): Número de octano en carretera

Como son todas escalas arbitrarias no coinciden los valores entre ellas. Sin embargo, existen relaciones entre las distintas escalas. Se han definido las siguientes magnitudes:

Sensibilidad: S=NOM-NOR

Depreciación en carretera: D=RDON-NOR.

1.1.3.3 Variación del número de octano:

Los hidrocarburos de cadena ramificada y corta van a tener NOR y NOM muy altos, tanto si son saturados como su presentan dobles enlaces en las moléculas.

Los hidrocaburos aromáticos (cíclicos) también presentan NOR y NOM altos

Los hidrocarburos lineales tienen NOR y NOM bajos

Las cicloparafina y naftnénicos $(CH_2)_N$, tienen el número de octano NOR y NOM en una escala intermedia.

Hay que decir que el número de octano no está en proporción con el funcionamiento del motor. El número de octano que va a presentar una gasolina dependerá de la naturaleza y del tipo de cadena que tengan los hidrocarburos. Conviene hidrocarburos con cadenas ramificadas, porque dan mejor número de octano.

Hay una serie de aditivos que nos permiten mejorar el índice de octano de una gasolina, ya que el octano inicial de la curva de destilación no es normalmente suficiente. Los primeros productos ensayados para adicionar a la gasolina fueron el tetraetilo de plomo, el problema está en los residuos que provoca. Se buscaron sustitutos como el plomo tetrametilo.

Tabla 1.1Valores de RON para mezclas de etanol con gasolina extra y súper

% Gasolina	% Etanol	RON
EXTRA	99	80,57
	95	82,62
	90	85,03
SUPER	99	91,26
	95	92,48
	90	92,61

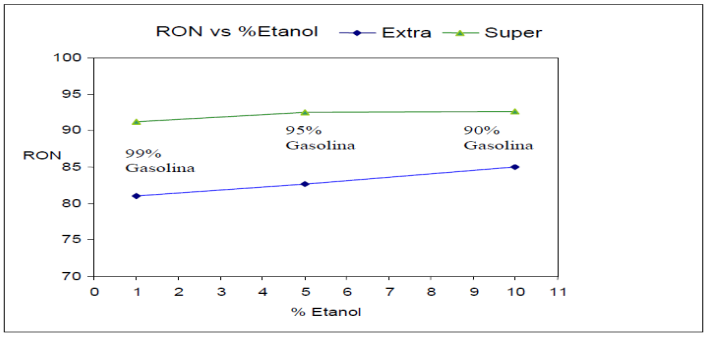


Figura 1.2 RON vs. % de etanol en mezcla con gasolina extra y súper

1.1.4 Estabilidad al Almacenamiento:

Se evalúa por la tendencia que presenta la gasolina a forma gomas. Las gomas son residuos que se forman durante el almacenamiento de los combustibles cuando parte de sus componentes se ha evaporado. Esta evaporación ha transcurrido en contacto con el aire y con metales. Estas gomas corresponden a compuestos originales por la oxidación y polimerización de los combustibles. Los problemas que pueden originar estos residuos pueden estar en el sistema de combustible o en el motor.

- Sistema de combustible: se deposita como residuo resinoso en la zona caliente de la toma de admisión. Si se va aumentando el residuo en capas, puede desprenderse y obturar el sistema de aspiración y filtros.
- Motor: obstruye las válvulas. Si se deposita en el colector puede llegar a dar humos en el tubo de escape (pérdida de potencia),

Todo esto se agrava si el combustible no está bien tratado. Un problema añadido es la propia degradación del combustible, lo que puede llevar a una disminución del nivel de octano, dando mal funcionamiento al motor.

1.1.4.1 Poder Calórico (Calor de Combustión).

Energía liberada cuando se somete el combustible a un proceso de oxidación rápido, de manera que el combustible se oxida totalmente y que una gran cantidad de calor que es aprovechable a nivel industrial.

1.1.4.1.1 Contenido en Azufre:

El azufre que se encuentra en un combustible líquido deriva del crudo de petróleo del que procede el combustible y a veces puede derivar de algún proceso al que ha sido sometido en el fraccionamiento, interesará que el contenido de azufre sea el menor posible, ya que la legislación marca unos límites. Los problemas que puede provocar el azufre contenido en un combustible líquido son:

- Corrosiones en los equipos en los que se quema el combustible.

- Contaminación ambiental, que se debe evitar.

- Influye sobre el poder calórico del combustible, pudiendo hacer que sea menor. Puede variarlo bastante.

1.1.4.1.2 Presión de Vapor de REID:

Aunque ésta no sea una medida exacta de la volatilidad, mide la tendencia que presenta el combustible a pasar a fase vapor. Para determinarla se mide la presión de vapor formado en el calentamiento de una muestra de un combustible líquido a 37.8ºC (ASTM-D323)

Esta prueba se emplea para saber qué ocurrirá en el almacenamiento de los productos en la refinería. Este ensayo no es una medida de la presión de vapor real, porque el aire que contiene la cámara va a estar en contacto con los vapores que se producen en el ensayo. Pero es una medida indirecta de elementos ligeros o muy volátiles que contiene el combustible a ensayar. De esto se deduce las conclusiones necesarias de cara al almacenamiento y transporte del combustible.

Tabla 1.2 Valores de PVR para mezclas de etanol con gasolina extra y súper

% Gasolina	% Etanol	PVR(kPa)
EXTRA	99	46,66
	95	51,48
	90	52,40
SUPER	99	50,79
	95	58,15
	90	60,22

En la actualidad se pueden hacer mezclas de gasolinas con índices de octano mayores que el del isooctano puro, o sea hasta de 110. Esto se logra agregando a la mezcla de gasolina compuestos llamados antidetonantes. El compuesto de este tipo más común es el tetraetilo de plomo (TEP)

1.1.4.1.3 Densidad

Los combustibles se comercializan en volumen, por ello es importante saber la densidad que tienen a temperatura ambiente.

Se define la densidad específica como:

$$\text{Densidad específica o relativa} = \frac{\text{Densidad absoluta de un producto (a una temperatura)}}{\text{Densidad del agua líquida (a 4C)}}$$

1.1.4.1.4 Viscosidad

Es la resistencia interna que presenta un fluido para el desplazamiento de sus moléculas. Esta resistencia viene del rozamiento de unas moléculas con otras, la unidad de la viscosidad es metro cuadrado por segundo (m^2/s), un combustible que tenga la viscosidad muy alta quiere decir que es poco fluido.

1.1.4.1.5 Punto de Inflamación

Se define como la mínima temperatura a la cual los vapores originados en el calentamiento de una muestra de combustible se inflaman cuando se ponen en contacto con una llama piloto de una forma determinada. Teniendo en cuenta el punto de inflamación se podrá estimar cuales van a ser las condiciones de almacenamiento de ese combustible.

1.1.4.1.6 Gasolina Comercial

La gasolina que compramos en las gasolineras se hace mezclando gasolina natural con diferentes porcentajes de gasolina proveniente de los procesos de polimerización, alquilación, isomerización, reformación y desintegración. A estas mezclas se les determina su octanaje, y se les agrega una serie de aditivos antes de venderlas al público.

1.1.4.1.7 Propiedades del Etanol

El etanol quizá sea el más importante de la serie de los alcoholes, debido a sus aplicaciones: como solvente inerte en laboratorios, preparación de bebidas alcohólicas, fabricación de perfumes, preparación de barnices, lacas y explosivos. En medicina como desinfectante y antiséptico. Y últimamente

mezclado con gasolina para uso como combustible en Motores de Combustión Interna. Todo esto debido a sus propiedades:

- Soluble al agua
- Mayor densidad que la gasolina
- Contiene el 72% de energía de la gasolina
- Mayor octanaje que la gasolina
- Su llama es invisible
- Conductor de electricidad
- Contiene oxígeno, etc.

1.1.5 Depósito de Combustible

El depósito de combustible en la mayoría de los vehículos se encuentra ubicado en la parte trasera y sujeto al bastidor. El depósito está construido por una chapa de metal o polietileno reticulado. Algunas veces contiene cierto número de deflectores o placas de metal sujetas en el interior y paralelas a sus caras. Estas placas están provistas de aberturas por las que pueda pasar el combustible. El objetivo principal de estas placas o deflectores es el de evitar durante los virajes del vehículo el paso súbito de combustible desde un extremo hacía el otro extremo del mismo.

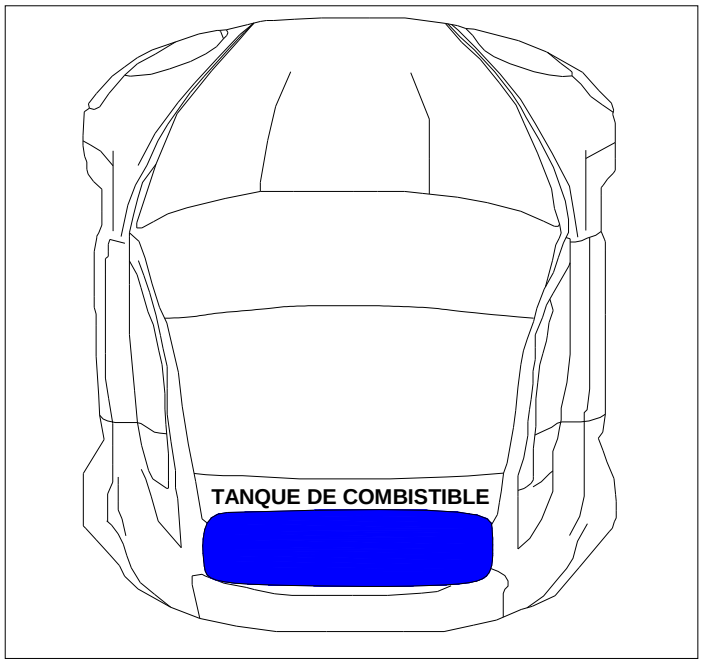


Figura 1.3 ubicación del tanque de combustible del auto motor.

El orificio de llenado del depósito de combustible está cerrado por medio de una tapa y el extremo inferior del tubo de llegada de combustible, en el depósito va sujeto al fondo de éste o muy cerca de él. Este extremo del tubo penetra en el depósito hasta un punto de altura algo superior al nivel del fondo con el fin la finalidad de evitar que la suciedad o el agua depositada en el fondo penetren en el mismo.

Figura 1.4 Tanque de combustible

El depósito de combustible trae acoplado en él un mecanismo que permite medir el nivel de combustible que del depósito, además contiene un elemento filtrante que permite impedir el paso de impurezas, el filtro se encuentra situado en el punto en el que el tubo de conducción se une al depósito; todo el combustible que sale del depósito pasa necesariamente a través del filtro.

La tapa del depósito o el mismo depósito tienen un tubo de respiradero para permitir que el aire penetre en el depósito a medida que este se vacía de combustible Si este respiradero se obstruye, se crearía un vacio en el depósito que impediría el paso normal de combustible hacia la bomba y la galería de inyectores.

Normalmente un depósito de combustible cuenta con las siguientes características:

- Almacenamiento seguro de combustible.

- El relleno debe ser sin riesgos (ej. chispas).
- Almacenamiento sin pérdidas por escape o evaporación.
- Proveer de un método para determinar el nivel de combustible en el depósito en todo momento. Para ello se usa un indicador del nivel de combustible.
- Venting (en caso de sobrepresión los vapores de combustible deben ser desviados por medio de válvulas).
- Alimentación del motor (por medio de una bomba).
- Anticipar posibles daños y riesgos para aumentar el potencial de sobrevivir.

1.1.5.1 Tecnologías para los Depósitos de Combustible en Automóviles.

- Depósitos de plástico, concretamente polietileno de alta densidad (HDPE) producidos por medio de moldeo por soplado. Esta tecnología está en auge por contar con emisiones de fuel muy reducidas. El plástico también posibilita geometrías complejas, por ejemplo, el depósito puede montarse directamente sobre el eje trasero, ahorrando espacio y mejorando la seguridad en caso de impacto. Inicialmente se tenían dudas sobre la dureza del material frente a grietas en comparación con materiales como el acero o el aluminio.

- Depósitos de metal (acero o aluminio) a partir de la soldadura de láminas estampadas. Aunque esta tecnología da buenos resultados en el control de emisiones del combustible, es cada vez menos competitiva en el mercado.

1.1.5.2 Células de Combustible para Competición

Una célula de combustible para competición es un contenedor de combustible que se diferencia del depósito ordinario en los siguientes aspectos:

- Tiene una alineación interior (inner liner) flexible para minimizar el riesgo de orificios en caso de colisión.

Está relleno con un núcleo de espuma de célula abierta para evitar la explosión del vapor en la parte vacía del depósito y para minimizar el sloshing del combustible durante la competición que pudiera desequilibrar el vehículo o provocar una alimentación de combustible inadecuada del motor

1.1.5.3 Automoción

El depósito de combustible se diseña de forma específica para cada vehículo una vez que el diseño es determinado, dado que han de optimizar el espacio libre disponible.

De hecho, a menudo se crean diferentes arquitecturas para el sistema del depósito de combustible dependiendo del tipo de vehículo, el tipo de combustible (gasolina o diesel), el tipo de dispensador de gasolina y la región donde se vende el vehículo.

1.1.6 Medición de Consumo de Combustible

La medición del consumo de combustible tiene por objeto controlar los costos por combustible ya sea de un vehículo particular o de una empresa que posee una flota de vehículos y mantenerlos dentro de valores razonables. Como sabemos el costo del combustible, en cualquier empresa de transportes, es el de mayor incidencia sobre los precios finales del servicio que brinda. Promediando, sin alejarnos mucho de la realidad. *“Este costo hoy rondan el 30 % (en franco incremento por los aumentos del petróleo en el orden mundial).”* Esto indica que las empresas no deben descuidar a este parámetro de tanta influencia en los precios finales y que, todo lo que se haga por reducirlo en algún punto decimal, redundará en números con cantidad de cifras sorprendentes al final de cada ejercicio contable. Como saber el consumo real de combustible de nuestro vehículo, es una interrogante bastante común entre los propietarios de los vehículos o dueños de flotas vehiculares; al obtener el dato real de cuanto combustible se consume, podremos saber si el vehículo está en buen estado, o está existiendo fugas no visibles de combustible, o también si la conducción es demasiado agresiva.

Cuando una persona compra un vehículo, junto a este viene un sinnúmero de especificaciones técnicas, entre ellas, el consumo especifico de combustible, pero como es bien sabido este valor no es real, este se encuentra muy por debajo, del consumo real; pues el valor que viene es las especificaciones técnicas del vehículo, se lo obtiene bajo condiciones optimas de funcionamiento, tanto del propio vehículo, como de las condiciones exteriores como el clima, tipo de carretera por la que circula.

1.1.6.1 Métodos de Medición de Consumo de Combustible

La duda que se presenta sobre si nuestros vehículos están consumiendo la cantidad adecuada de combustible o la posibilidad de existir una extracción indebida de combustible, es un tema que conduce a muchos propietarios de vehículos a cierto estado de desconfianza y de estrés, al saber que los rubros por consumo de combustible son extremadamente elevados en relación al rendimiento mismo del vehículo en carretera.
Esto, obviamente, incrementa los problemas; y en caso de empresas torna más difícil la relación con sus chóferes. Es por esto que se debe buscar una buena solución que permita llevar un control directo sobre las variables que afectan al consumo excesivo, o un monitoreo del mismo.

1.1.6.2 Método de Tanque Lleno

Este método es el más sencillo para determinar el consumo de combustible de un vehículo, este método implica que el tanque de combustible debe estar totalmente lleno, al principio y al final de la prueba, y este consumo será relacionado con la distancia recorrida por el vehículo o en caso de maquinaria pesada por las horas de trabajo; el procedimiento a seguir es el especificado a continuación.

1.1.6.2.1 Procedimiento:

1. Llenar completamente el tanque de combustible en la mañana al inicio de la jornada de trabajo.

2. Registrar el odómetro en cero o el horómetro en caso de que la prueba se realice en maquinaria pesada.
3. Seleccionar un conductor para que realice la actividad para relacionar el tipo de trabajo con el consumo de combustible, esto es para el caso en que la prueba se la realice en una flota vehicular.
4. Posteriormente al término de la jornada de trabajo se deberá volver a llenar el tanque de combustible, y registrar la cantidad de combustible consumido durante toda la jornada.
5. Además se deberá registrar la distancia recorrida del vehículo, o las horas de trabajo de la maquinaria.
6. Calcular el consumo específico de combustible, dividiendo la cantidad de combustible consumido entre la distancia recorrida o las horas de trabajo para el otro caso.

Esta prueba debe realizársela varias veces, por lo menos una semana que es un tiempo considerable para reunir los datos de todas las actividades normales que se realizan y en diferentes condiciones de trabajo del vehículo.

1.1.6.3 Método de la Varilla Calibrada

En este método como su nombre lo indica se utiliza una varilla calibrada, pero los datos que se obtienen en este método son limitados por dos factores muy visibles, como es la tolerancia en la calibración de la varilla y la posición del vehículo o de la máquina en el momento de realizar las mediciones respectivas.

Este método presenta una ventaja, que sus medidas pueden ser todas obtenidas en cualquier momento, el procedimiento es el siguiente.

1.1.6.3.1 Procedimiento:

1. Primeramente se debe colocar el vehículo o la maquina en un lugar nivelado verificándolo mediante un nivel.

2. A continuación se deben diseñar las varillas de medición cuando el vehículo se encuentre paralizado o cuando este se encuentre en reparación.
3. Luego se retira todo el combustible del depósito.
4. Retirar el tanque de combustible del vehículo, para posteriormente realizar la limpieza interior del mismo.
5. Después de haber realizado la limpieza del tanque de combustible lo colocamos en la posición de trabajo del vehículo.
6. Debemos llenar en un depósito cada cinco galones de combustible y depositarlo lentamente al tanque de una manera que el combustible esté lo más quieto posible en el recipiente para una medición
7. Con la regla graduada procedemos a medir la altura del agregado en el depósito, con la finalidad de obtener una relación en centímetros por galón *cm/gal*, se debe repetir este procedimiento hasta que el tanque se encuentre completamente lleno.
8. Tomamos todos los datos de las medidas de volumen, y longitud que marca la varilla de acero por cada cinco galones de combustible agregados al depósito.
9. Con la varilla calibrada procedemos a codificarla con el número de código de la unidad evaluada.
10. En el momento que se estén realizando las mediciones en el depósito de combustible del vehículo, este debe estar en una zona nivelada con la finalidad de obtener las mediciones lo más exactas posible.

1.2 DISPOSITIVOS ELECTRÓNICOS A UTILIZAR.

1.2.1 Sensor.

Un sensor es cualquier dispositivo que detecta una determinada acción externa. Los sensores existen desde siempre, y nunca mejor dicho, porque el hombre los tiene incluidos en su cuerpo y de diferentes tipos.

El hombre experimenta sensaciones como calor o frío, duro o blando, fuerte o flojo, agradable o desagradable, pesado o no. Y poco a poco le ha ido añadiendo adjetivos a estas sensaciones para cuantificarlas como frígido, fresco, tibio, templado, caliente, frio. Es decir, que día a día ha ido necesitando el empleo de magnitudes medibles.

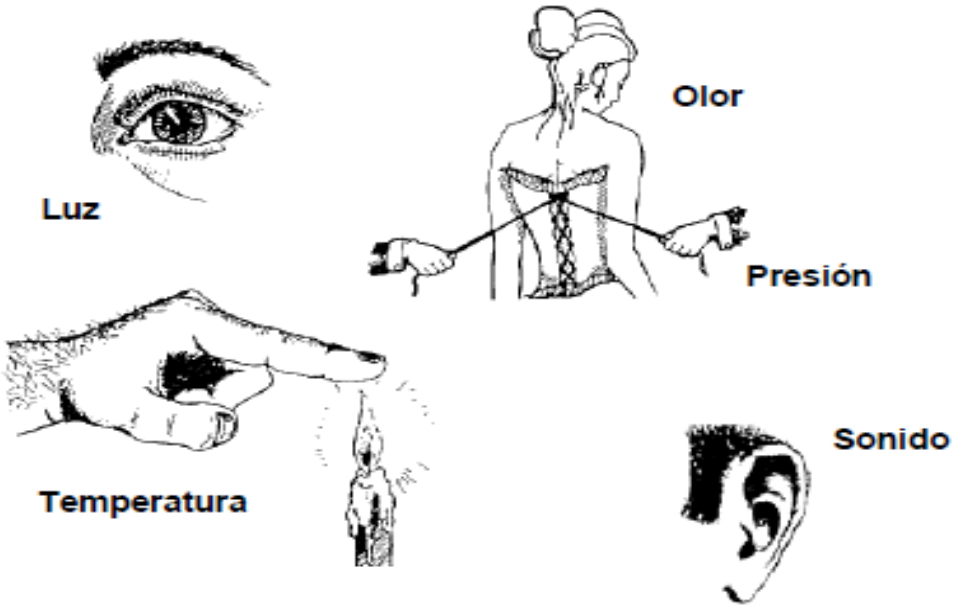


Figura 1.5

1.2.1.1 Sensores Electrónicos

Los sensores electrónicos han ayudado no solo a medir con mayor exactitud las magnitudes, sino a poder operar con dichas medidas. Pero no se puede hablar de los sensores sin sus acondicionadores de señal, ya normalmente los sensores ofrecen una variación de señal muy pequeña y es muy importante equilibrar las características del sensor con las del circuito que le permite medir, acondicionar, procesar y actuar con dichas medidas.

- Debido a la estructura electrónica de la materia, cualquier variación de un parámetro no eléctrico de un material viene acompañada por la variación del parámetro eléctrico. Elegido el material adecuado, esto permite realizar sensores de salida eléctrica para cualquier magnitud física no eléctrica.
- Dado que en el proceso de la medida no conviene extraer energía del sistema donde se mide, lo mejor es amplificar la señal de salida del

sensor, con amplificadores electrónicos que se puede obtener fácilmente ganancias de 10^{10} en una sola etapa, a baja frecuencia.

1.2.1.2 Tipo de sensores

1.2.1.2.1 Aporte de energía

Sensor Activo y Sensor Pasivo**.**

1.2.1.2.2 Sensor Activo.

La energía de la señal de salida procede, en su mayor parte, de una fuente de energía auxiliar la entrada solo controla la salida.

1.2.1.2.3 Sensor Pasivo.

La energía de salida es suministrada por la entrada.

1.2.1.2.4 Según la señal de salida

Sensor Analógico y Sensor Digital.

1.2.1.2.4.1 Sensores Analógicos

En los analógicos la salida varía a nivel macroscópico, de forma continua. La información está en la amplitud.

1.2.1.2.4.2 Sensores digitales

En los digitales varía en forma de saltos o pasos discretos con mayor exactitud no requieren de conversión A/D tiene mayor fiabilidad.

1.2.1.2.5 Modo de Operación

Sensores Deflexiones y Sensores de Comparación.

1.2.1.2.5.1 Sensores de Deflexiones

La magnitud medida produce un efecto físico, que genera algún efecto similar, pero opuesto en alguna parte del instrumento, que está relacionado con alguna variable útil.

1.2.1.2.5.2 Sensores de Comparación

Suelen ser más exactos por el efecto conocido opuesto se puede calibrar con un patrón o magnitud de referencia de calidad. El efecto de desequilibrio solo se mide alrededor de cero por lo tanto puede ser muy sensible y no necesita estar equilibrado.

Tiene en principio menor respuesta dinámica y si bien se puede automatizar mediante un servomecanismo, no se logra normalmente una respuesta tan rápida como en los de deflexión.

Tabla 1.3 clasificación de sensores

Criterio	Clase	Ejemplos
Aporte de Energía	Modulares Generadores	Termistores Termopar
Señal de salida	Analógicos Digitales	Potenciómetro Codificador de posición
Modo de operación	De deflexión De comparación	Acelerómetro de deflexión Servoacelerómetro

En la tabla 1.3 se recoge los sensores y métodos de detención ordinarios para las magnitudes más frecuentes.

1.2.1.2.6 Sensores Resistivos

Los sensores basados en la variación de la resistencia eléctrica de un dispositivo son los más usados y los más abundantes. Ello se debe a que son muchas las magnitudes físicas que afectan el valor de la resistencia eléctrica de una material. En consecuencia, ofrecen una solución válida para numerosos problemas de medida. En el caso de resistores variables de temperatura, ofrecen un método de compensación térmica en los sistemas de medida de otra magnitud.

Para la clasificación de los diversos sensores de esta clase se tomó como criterio el tipo de magnitud física a medir. El orden seguido es el de variables mecánicas, térmicas, magnéticas, ópticas y químicas.

1.2.1.2.6.1 Potenciómetros.

Un potenciómetro es un resistor con un contacto móvil deslizante o giratorio. La resistencia entre dicho móvil y uno de los terminales fijo es.

$$R = \frac{\rho}{A} l(1 - \alpha) = \frac{\rho}{A}(l - x)$$

Donde x es la distancia recorrida desde el otro terminal fijo, α la fracción de longitud correspondiente, ρ es la resistividad del material, l su longitud y A su sección transversal.

Se supone que la resistencia es uniforme a lo largo de todo el recorrido l, pero obviamente la resistencia no será perfectamente uniforme por lo que la linealidad del potenciómetro estará acotada. La diferencia entre el comportamiento esperado y el obtenido se denomina confortabilidad. Se supone que el contacto del cursor da una variación de resistencia continua no a saltos y que por lo tanto, la resolución es infinita, pero esto no es cierto para todos los tipos de elementos resistivos. A de más, el recorrido mecánico suele ser mayor que el recorrido mecánico, es decir no todo el soporte está recubierto de conductor.

Si se alimenta el potenciómetro con una tensión alterna, su inductancia debe ser despreciable. Para valores bajos, la inductancia no siempre es despreciable, en particular si se trata de modelos con resistencia bobinada, para valores altos la capacidad parásita puede tener importancia.

Los resistores cambian con la temperatura. Por lo tanto el modelo anterior es válido siempre y cuando no haya variaciones no uniformes de temperatura, estas pueden ser debidas no solo a fluctuaciones de la temperatura ambiente, sino también a un auto calentamiento motivado por la alimentación de potencia que se puede disipar en el potenciómetro. Si esta es P y se alimenta el potenciómetro con una tensión, el valores eficaz de esta en bornes de aquel, V, debe cumplir.

$$V \leq \sqrt{PR}$$

Si el circuito de medida no tiene una impedancia de entrada suficientemente alta y la carga al potenciómetro, puede haber calentamiento excesivo en determinada zonas.

1.2.1.2.6.1.1 Medidas de Potenciómetros

El comportamiento general de los sensores basado en una variación de resistencia x en respuesta a una variación se puede medir, se puede expresar como:

$$R = R_o f(x)$$

Para el caso en que la relación sea lineal se tiene:
Cuando: f(0) = 1

$$R = R_o = (1 + x)$$

Alimentar un potenciómetro con una fuente de tensión o corriente viene limitada por el hecho de que el valor máximo del cambio a medir puede ser incluso de solo 1%, y ello se supone tener que medir cambios de corriente o de tensión muy pequeños, superpuesto a valores estacionarios muy altos cuando x fuera grande.

1.2.1.2.6.1.2 Divisores de tensión

Para un potenciómetro de resistencia nominal Rn cuyo cursor se desliza o gira accionado por un dispositivo cuyo movimiento lineal o angular se tiene en la figura 1.6. El Circuito equivalente, considerado el dispositivo de medida tiene una resistencia de entrada finita Rm donde Vs y Rs son las tensiones del circuito abierto y la resistencia de salida de ambas figuras se deducen las siguientes relaciones.

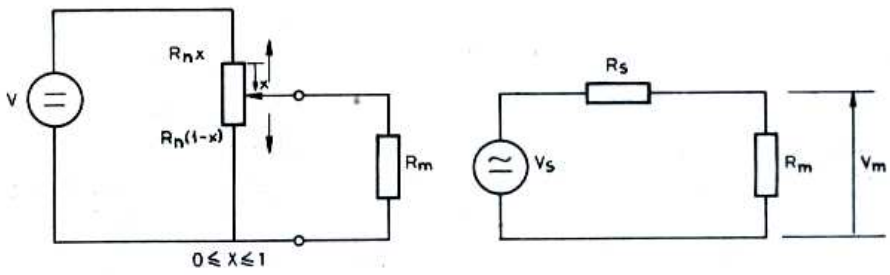


Figura 1.6 división de tensión

$$V_s = V(1-x)$$

$$R_s = R_n\, x\, (1-x)$$

$$V_m = \frac{V(1-x)}{R_n x\,(1-x) + R_m} \qquad R_m = \frac{V\,\alpha}{\dfrac{\alpha(1-\alpha)}{k} + 1}$$

Donde K = Rm/Rn y α =1 –x. se puede observar que la relacion entre la tension medida y el desplazamiento del cursor sólo es lineal cunado K es muy elevada. Es decir, la resistencia de entrada del voltimetro debe exceder con crecer a la resitencia del potenciometro. En la figura 1.7 se muestra como la tension medida se aparta de la tensiòn de salida en el circuito abierto tanto K es menor.

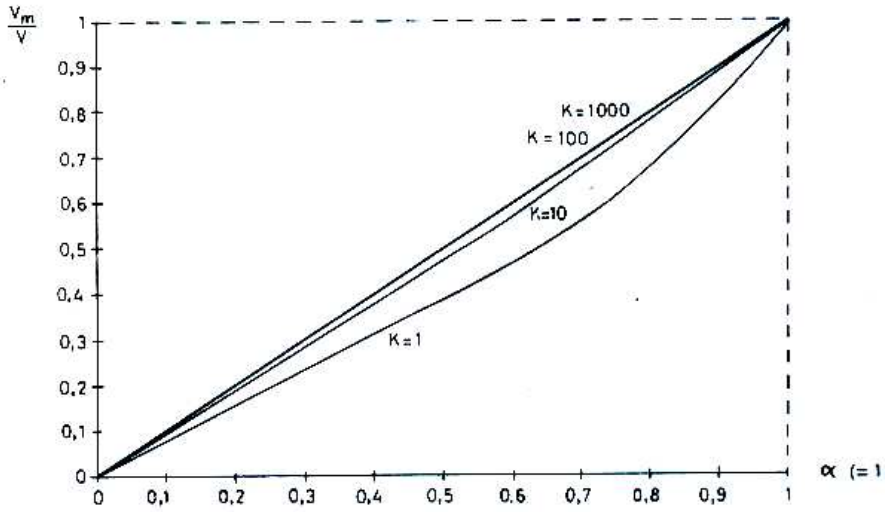


Figura 1.7 curba de tension movil y contacto fijo

Variación de la tensión entre el control móvil y el contacto fijo de un potenciómetro en función de la posición del cursor para distintos valores de K (relación entre resistencia del instrumento de medida y la nominal del potenciómetro).

1.2.1.2.7 Sensores nivel Tipo por Flotador

El principio de Arquímedes ¨un cuerpo sumergido totalmente o parcialmente en un fluido experimenta una fuerza de flotación proporcional a la masa de fluido que desaloja¨ se usa un sensor de nivel cuyo elemento sensor es un flotador hueco o de un material más ligero que el fluido a medir. El movimiento hacia

arriba y abajo del flotador se convierte en una señal de salida a través de un mecanismo y su transductor de desplazamientos.

Este desplazamiento que realiza el flotador está conectado al potenciómetro por medio de un brazo el cual permite variar la resistencia del mismo y de esta manera toma la medida requerida como se indica en la figura 1.8

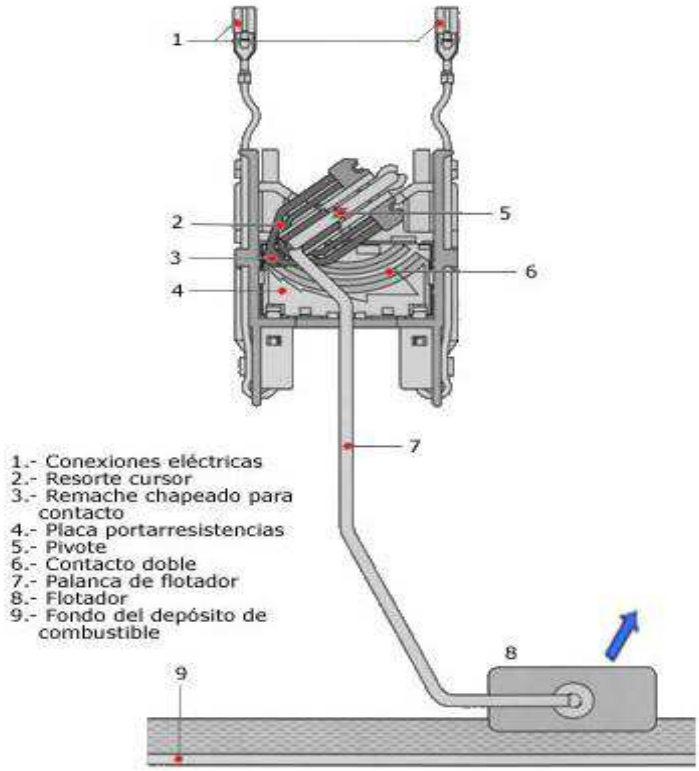


Figura 1.8 potenciómetro flotador

Factores a considerar en la elección de un sensor

Magnitud a medir.	**Características de salida.**
Margen de medida.	Sensibilidad.
Resolución	Tipo: tensión, corriente, frecuencia.
Exactitud deseada	Forma Señal unipolar, flotante, diferencial.
Estabilidad	Impedancia.
Ancho de banda	Destino: presentación analógica
Tiempo de respuesta	Conversión digital

Límites absolutos posibles	Telemedida: tipo de la magnitud a medir
Magnitudes interferentes.	

Características de alimentación	**Características ambientales**
Tensión	Margen de temperatura.
Corriente	Humedad.
Potencia disponible	Vibraciones.
Frecuencia (si alterna)	Agentes químicos.
Estabilidad	¿Atmosfera explosiva?
	Entorno electromagnético.

Otros factores

Peso	Longitud de cable necesario.
Dimensiones	Tipo de conector.
Vida media	Sustitución en el caso de fallo.
Coste de adquisición	Coste de verificación.
Disponibilidad	Coste de mantenimiento.
Tiempo de instalación	coste de sustitución.

*1.*3 LOS MICROCONTROLADORES[1]

1.3.1 INTRODUCCIÓN A LOS MICROCONTROLADORES

Los microcontroladores hicieron su aparición a principio de los ’80 y se trata de un circuito integrado programable que contiene toda la estructura de una microcomputadora. Es decir que, dentro de un microcontrolador podemos encontrar:

- Una CPU (Unidad Central de Proceso)
- Memoria RAM.

[1] José M Angulo Usategui, Susana Romero Yesa e Ignacio Angulo Martinez “Microcontroladores PIC – Diseño práctico de aplicaciones segunda parte- PIC16F87x”**,** *ed. Graw Hill* 1ª Edición

- Memoria ROM
- Memoria EEPROM
- Puertos de Entrada/Salida (Pines de E/S) e incluso muchos modelos de microcontroladores incorporan distintos módulos “periféricos”, como pueden ser; conversores analógico/digital, módulos PWM (control por ancho de pulso), módulos de comunicaciones seriales o en paralelo, ver figura 1.9

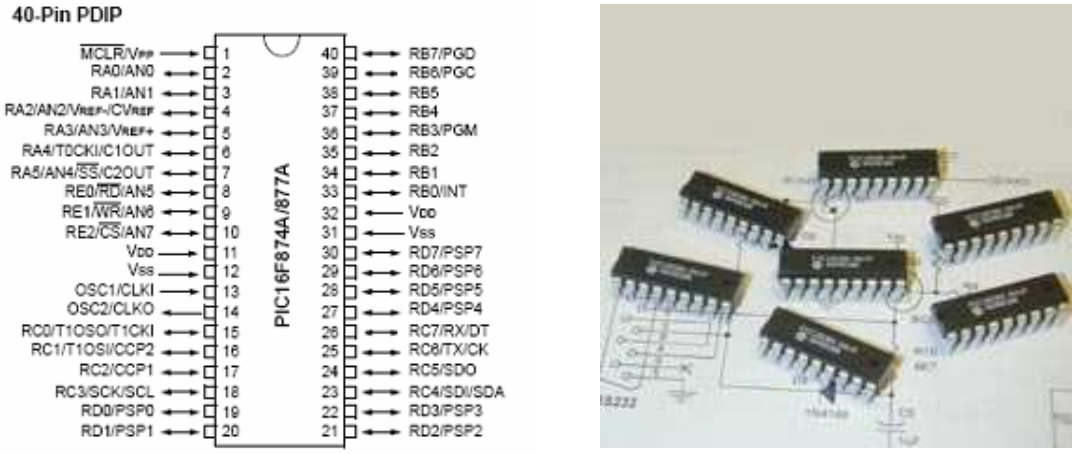


Figura 1.9 Circuito integrado del microcontrolador 16F877A

Cada vez existen más productos que incorporan microcontroladores con el fin de aumentar sustancialmente sus prestaciones, reducir su tamaño y costo, mejorar su confiabilidad y disminuir el consumo de energía.

1.3.1.1 Los Microcontroladores “PIC”

Los microcontroladores denominados “PIC”, corresponden exclusivamente a la marca “Microchip”. PIC significa "**Peripheral Interface Controller**" y fue desarrollado por Microchip a principio de los 80.

Existe una gran cantidad de modelos de microcontroladores cuyas características y prestaciones varían de un modelo a otro. De esta manera los fabricantes pueden seleccionar el modelo que mejor se ajuste a sus necesidades. Los distintos modelos de microcontroladores se agrupan por

"familia". Una familia puede estar formada por un conjunto de modelos cuyas características y prestaciones son bastante similares.

Cuando compramos un microcontrolador, la memoria del mismo se encuentra "vacía" y para que funcione es necesario que sea "programado".

1.3.2 ESTRUCTURA DE UN MICROCONTROLADOR[2]

Básicamente, un microcontrolador está compuesto por los siguientes componentes:

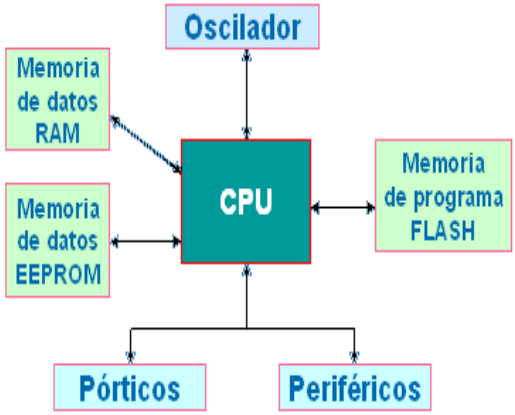


Figura 1.10 Partes de un microcontrolador

- Procesador o CPU (Unidad Central de proceso).
- Memoria para el programa tipo ROM.
- Memoria RAM para contener los datos.
- Líneas de E/S para comunicarse con el exterior.
- Diversos módulos para el control de periféricos (temporizadores, Puertas Serie y Paralelo, CAD: Conversores Analógico/Digital, CDA: Conversores Digital/Analógico, etc.).

En la figura 1.11 podemos ver la estructura de un microcontrolador PIC, en este caso de la familia 16F87X.

---

[2] Microchip, *http:// www.microchip.com*

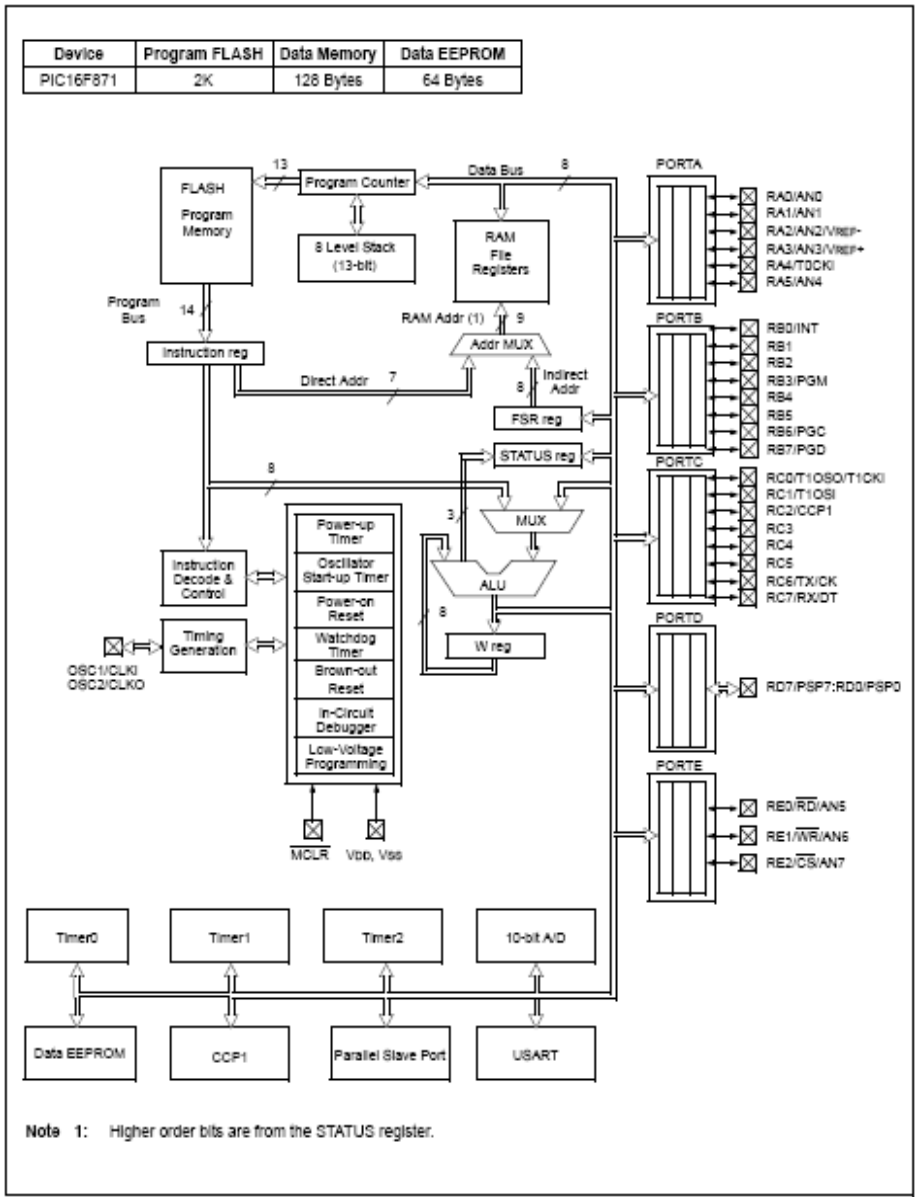


Figura 1.11 Estructura interna del PIC

1.3.2.1 El Procesador o CPU

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software. La CPU (Unidad Central de Proceso) se encarga la decodificación y ejecución del programa.

Actualmente, existen 3 tipos de arquitectura de procesadores:

- **CISC** (Juego de Instrucciones Complejo para Computadores): Disponen de más de 80 instrucciones en su repertorio, algunas de las cuales son muy sofisticadas y potentes, requiriendo muchos ciclos para su ejecución. Una ventaja de los procesadores CISC es que ofrecen instrucciones complejas que actúan como macros.

- **RISC** (Juego de Instrucciones Reducido para Computadores): En estos procesadores el repertorio de instrucciones es muy reducido y las instrucciones son simples y generalmente se ejecutan en un ciclo maquina. La ventaja de éstos es que la sencillez y rapidez de las instrucciones permiten optimizar el hardware y el software del procesador.

- **SISC** (Juego de Instrucciones Específico para Computadores): En los microcontroladores destinados a aplicaciones muy concretas, el juego de instrucciones, además de ser reducido, es "específico", o sea, las instrucciones se adaptan a las necesidades de la aplicación prevista.

1.3.2.2 Memoria ROM

La memoria ROM es una memoria no volátil, es decir, que no se pierden los datos al desconectar el equipo y se destina a contener el programa de instrucciones que gobierna la aplicación. Los microcontroladores disponen de capacidades de ROM comprendidas entre 512 bytes y 8 k bytes.

Existen distintos tipos de memorias ROM, la cual determinará la aplicación del microcontrolador.

- **ROM con máscara:** Es una memoria no volátil de sólo lectura cuyo contenido se graba durante la fabricación del chip.

- **OTP:** El microcontrolador contiene una memoria no volátil de sólo lectura "programable una sola vez" por el usuario. OTP (One Time Programmable).

- **EPROM:** Los microcontroladores que disponen de memoria EPROM (Erasable Programmable Read Only Memory) pueden borrarse y grabarse muchas veces. Para borrar el contenido, disponen de una ventana de cristal en su superficie por la que se somete a la EPROM a rayos ultravioleta durante varios minutos. Las cápsulas son de material cerámico.

- **EEPROM:** Es una memoria de sólo lectura, las cuales se puede escribir y borrar eléctricamente. EEPROM (Electrical Erasable Programmable Read Only Memory). Tanto la programación como el borrado, se realizan eléctricamente desde el propio grabador y bajo el control programado de un PC. El número de veces que puede grabarse y borrarse una memoria EEPROM es finito.

- **FLASH:** Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos energía y es más pequeña. A diferencia de la ROM, la memoria FLASH es programable en el circuito. Es más rápida y de mayor densidad que la EEPROM. Es más veloz y tolera más ciclos de escritura y borrado.

1.3.2.3 Memoria RAM

La memoria RAM es una memoria volátil, es decir, que se pierden los datos al desconectar la energía eléctrica y se destina a guardar las variables y los datos. Los microcontroladores disponen de capacidades de RAM comprendidas entre 20 y 512 bytes.

1.3.2.4 Registros y Bits

Un registro es una posición de memoria en la cual se puede almacenar un dato. Es decir que la memoria está dividida en pequeñas partes llamadas "**Registros**". Dentro de la memoria, cada registro se identifica mediante un

número, el cual se denomina "**Dirección de memoria**" y generalmente está expresado en formato Hexadecimal. El primer registro de una memoria corresponde a la dirección 00h.

Dado a que cada registro es identificado mediante un número hexadecimal puede resultar muy complejo a la hora de diseñar el programa, existe la posibilidad de asignar un "nombre" a una dirección de registro. En general, este nombre está directamente relacionado con la función que cada registro cumple dentro del sistema.

Entonces podemos decir que un Registro está formado por un conjunto de 8 bits.

Los sistemas digitales representan la información en forma de bits porque sus circuitos sólo pueden tener dos estados: encendido o apagado. En general podemos decir que:

- 1 = Encendido = Verdadero = "SI" = +5V
- 0 = Apagado = Falso = "NO" = 0V

Cada Bit se identifica por la posición que ocupa dentro del registro, siendo el primer Bit el número 0, que es el que se encuentra en el extremo derecho del registro, ver figura 1.4.

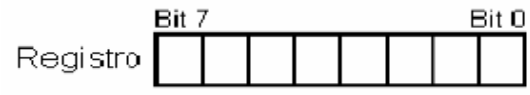


Figura 1.12 Estructura de un registro

En un registro se puede almacenar una combinación 8 ceros y/o unos. Esto nos da una cantidad de 2^8 combinaciones, es decir, 256 posibles combinaciones de ceros y unos. Esto significa que un registro puede procesar valores entre 0 y 255.

El siguiente ejemplo muestra el desarrollo de un cálculo de conversión de base de sistema binario (base 2) a sistema decimal (base 10):

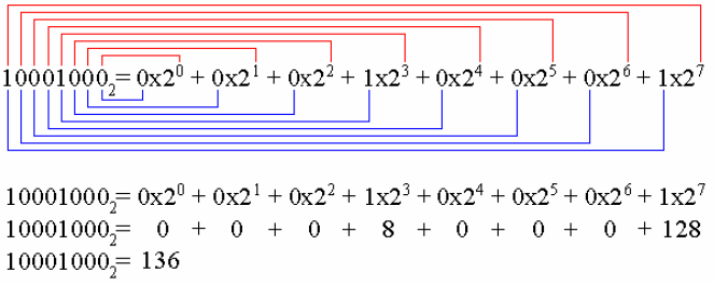


Figura 1.13 Cambio de Sistema binario a sistema decimal

Se llama "**Peso Binario**" al valor que representa un Bit según la posición que ocupa dentro del registro. El Bit que está ubicado más hacia la derecha del registro, es el Bit menos significativo (LSB) y tiene un peso de $2^0 = 1$. El Bit del extremo izquierdo del registro es el Bit más significativo (MSB) y tiene un peso de $2^7 = 128$.

Los pesos binarios crecen de derecha a izquierda en potencias de 2.

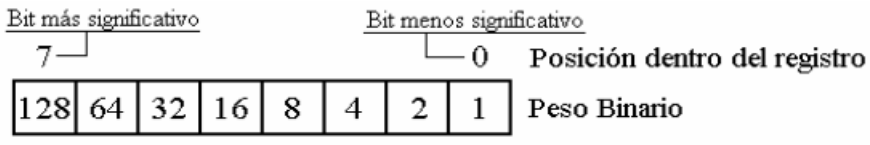


Figura 1.14 Sistema de pesos en el sistema binario.

La manera de simplificar la conversión de binario a decimal, es directamente sumar los valores de los pesos binarios de los bits cuyo valor sea 1.

Figura 1.15 Simplificación del cálculo de conversión.

El sistema hexadecimal es un sistema en base 16 y consta de 16 dígitos diferentes que son: del 0 al 9 y luego de la letra "**A**" a la "**F**", es decir, 10 dígitos

numéricos y seis caracteres alfabéticos. El sistema hexadecimal se usa como forma simplificada de representación de números binarios y debidos a que 16 es una potencia de 2 ($2^4 = 16$), resulta muy sencilla la conversión de los números del sistema binario al hexadecimal y viceversa.

Mediante el sistema hexadecimal podemos representar un valor numérico de 8 bits utilizando sólo 2 dígitos. De esta manera estamos dividiendo el registro de 8 bits en dos partes de 4 bits cada una llamada **Nibble**.

Al nibble correspondiente a los 4 bits menos significativos, se lo denomina "**Nibble Bajo**" y al nibble correspondiente a los 4 bits más significativos se lo denomina "**Nibble Alto**".

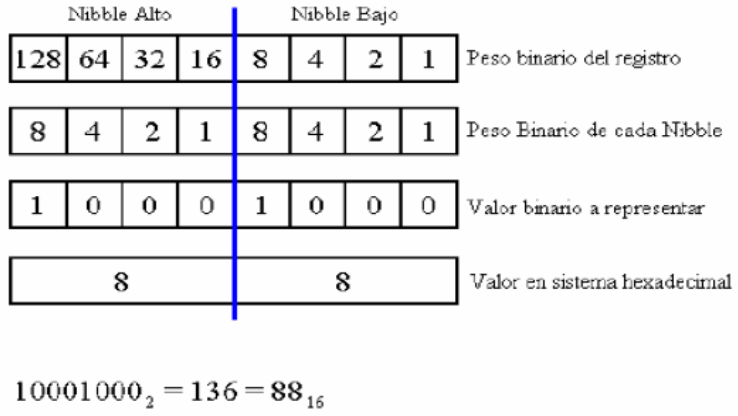


Figura 1.16 Separación entre Nibbles

El sistema hexadecimal es utilizado para identificar las direcciones de registros de las memorias, en sistemas digitales porque permite representar el valor de un **Nibble** con solo un dígito, ya que:

$$\mathbf{1111_2} \rightarrow (8+4+2+1) = 15 = \mathbf{F_{16}}$$

Figura 1.17 Sistema Hexadecimal

Esto permite representar números grandes utilizando unos pocos dígitos.
Por ejemplo:

$$\mathbf{FF\ FF_{16}} = 11111111\ 11111111_2 = 65535$$

Figura 1.18 Representación de 4 dígitos hexadecimales

En la programación de microcontroladores, es habitual utilizar los tres sistemas de numeración (Binario, Decimal y Hexadecimal), dependiendo del proceso que necesitemos realizar. Por eso es fundamental tener claros estos conceptos.

1.3.2.5 Líneas de Entrada/Salida (E/S), (Puertos)

Los microcontroladores cuentan con una serie de pines destinados a entrada y salida de datos o señales digitales. A estos pines se les denomina "**Puerto**".

Un puerto puede estar formado por menos de 8 pines. Un microcontrolador puede contener varios puertos dependiendo del modelo. A cada puerto se lo identifica con una letra. Por ejemplo; "Puerto A", "Puerto B", etc. Para poder utilizar un puerto, el mismo debe ser configurado. Cada pin de un puerto puede ser configurado como entrada o salida independientemente del resto de los pines del mismo puerto.

1.3.2.6 Módulos Temporizadores Internos (TMRS)

Un temporizador interno (TMR), es un módulo de hardware incluido en el mismo microcontrolador, el cual está especialmente diseñado para incrementar automáticamente el valor de un registro asociado al TMR cada vez que el módulo TMR recibe un pulso. A este pulso se lo llama "**señal de reloj**".

El módulo TMR siempre incrementa el valor del registro asociado, nunca decrementa dicho valor. Algunos microcontroladores pueden incluir más de un módulo TMR y la señal de reloj de cada uno de éstos puede ser de origen interno o externo. Si el origen de la señal de reloj está configurado como externo, el módulo temporizador puede ser utilizado como un contador de

eventos externos, incrementando el TMR con cada pulso recibido mediante el pin correspondiente.

Si el origen de la señal de reloj es interno, el TMR incrementa con cada ciclo del oscilador. Esto permite utilizar el temporizador como “contador de ciclos de programa”, donde, un ciclo corresponde al tiempo de ejecución de una instrucción, lo cual se puede calcular con la siguiente fórmula:

$$T \equiv te \equiv \frac{1}{Frec.Osc./4}$$

Donde “Frec. Osc.” es la frecuencia del oscilador utilizado, T es el periodo que es igual al tiempo de ejecución del microcontrolador en microsegundos (te).

Dado que la velocidad de ejecución del microcontrolador corresponde a ¼ de la velocidad del cristal utilizado, cada ciclo de programa se ejecuta en un tiempo determinado según el cristal que estemos utilizando. Por ejemplo; con un cristal de 4Mhz la velocidad real de procesamiento del microcontrolador es de 1 MHz. Aplicando la siguiente fórmula:

$$T = \frac{1}{4000000/4} = \frac{1}{1000000} = 1\mu s$$
$$\Rightarrow T = 1\mu s(microsegundos).$$

Esto significa que cada ciclo de programa se ejecuta a **[1/1000000] = 1*u*s** y dado que cada incremento del TMR corresponde a un ciclo de programa, si contamos los incrementos de un TMR, indirectamente podremos calcular el tiempo transcurrido.

1.3.3 UTILIZANDO UN MICROCONTROLADOR

- **Lenguaje de “Alto Nivel”:** permite que los algoritmos se expresen en un nivel y estilo de escritura fácilmente legible y comprensible por el hombre. En la actualidad se trata de lenguajes de tipo visual.

- **Lenguaje de "Bajo Nivel":** el usuario se acerca un poco más al lenguaje de máquina. Permiten un acceso más amplio al control físico de la máquina (hardware).

- **Lenguaje Ensamblador:** Podríamos considerarlo el lenguaje de más bajo nivel. El usuario escribe código en el mismo "idioma" del procesador. Se tiene control total del sistema. El lenguaje de programación es muy especifico para cada modelo de procesador, incluso puede variar de un modelo a otro de procesador dentro de un mismo fabricante.

Podemos decir que los lenguajes de alto nivel se asemejan más al lenguaje humano y que los lenguajes de bajo nivel se asemejan más al lenguaje de máquina y en el lenguaje ensamblador el usuario debe programar en el propio "idioma del procesador".

Figura 1.19 Niveles de lenguajes de programación

El microcontrolador sólo entiende de números, es decir que, el código Assembler (texto) no puede ser procesado directamente por el microcontrolador. Para poder grabar el programa en el microcontrolador, primero debemos convertir el texto del código Assembler a números, en general, en formato hexadecimal. A este proceso se le llama "**Compilación**".

Una vez desarrollado el código Assembler, el mismo debe ser "compilado". Esto se realiza con un "software compilador".

El archivo compilado tiene una extensión .hex. Por último, después de compilado, el programa está listo para ser grabado al microcontrolador. Esto realiza mediante una placa programadora. A esta placa programadora, comúnmente se la llama "**programador**".

Existen distintos tipos de programadores, los cuales pueden ser conectados a la PC mediante el puerto Serie (COM) o Paralelo (LPT).

A continuación se presenta un resumen del proceso de desarrollo del código y grabación de un microcontrolador:

- Escribir el código Assembler: Se genera un archivo con extensión ASM.
- Compilar el código Assembler: Se genera un archivo con extensión HEX.
- Grabar (transferir) el programa desde la PC al microcontrolador mediante un programador.

1.3.4 Características Técnicas de los PIC 16F87X

Inicialmente todos los microcontroladores incorporaron la arquitectura de Von Neumann, la cual se caracteriza por disponer de una sola memoria principal donde se almacenan los datos y las instrucciones. A esta memoria se accede a través de un sistema de buses único (direcciones, datos y control).

Los microcontroladores de la familia 16F87X, al igual que el resto de los microcontroladores de la actualidad, están diseñados con la arquitectura **Harvard**. La arquitectura Harvard dispone de dos memorias independientes; una que contiene sólo instrucciones, y otra donde se almacenan los datos, ver figura 1.20.

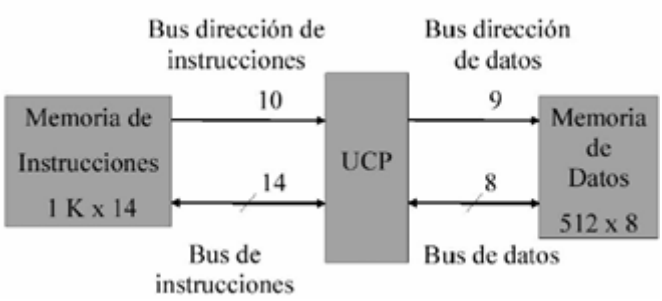


Figura 1.20 Arquitectura Harvard.

Ambas memorias cuentan con sus respectivos sistemas de buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias. Para que la CPU funcione, debe contar con un generador de impulsos de reloj externo que sincroniza el funcionamiento. Como origen de la señal de reloj externa utilizamos un cristal de cuarzo. En general, un ciclo de programa corresponde a una línea de código Assembler.

1.3.4.1 Los Puertos del PIC 16F87X

Los microcontroladores de la familia 16F87x disponen de 3 a 5 puertos según el modelo de microcontrolador: Estructura interna y especificaciones técnicas en el (ANEXO 1).

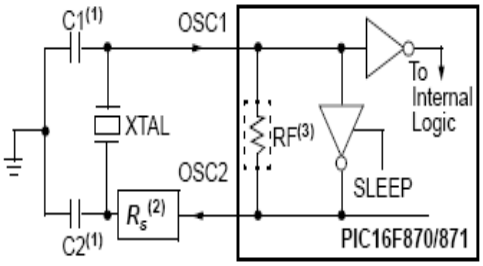
- Puerto A = 6 pines (5 pines A/D)
- Puerto B = 8 pines
- Puerto C = 8 pines
- Puerto D = 8 pines
- Puerto E = 3 pines (3 pines A/D)

1.3.4.2 Oscilador Externo 16F87X

Para la programación del microcontrolador PIC 16F870 se debe tener en cuenta toda su estructura externa y elementos que serán conectados a él, una buena operación en el tiempo de maquina se debe colocar los siguientes esquemas:

- Polarización: V_{DD} = 5V; V_{SS} = 0V
- OSC1/CLKIN: entrada del circuito oscilador externo
- OSC2/CLKOUT: Auxiliar del circuito oscilador

Configuración de cristal resonante (HS, XT o LP).



Osc Type	Crystal Freq.	Cap. Range C1	Cap. Range C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15-33 pF	15-33 pF
	20 MHz	15-33 pF	15-33 pF

These values are for design guidance only.
See notes following this table.

Figura 1.21 Configuración de cristal resonante (HS, XT o LP)

Oscilador de modo RC

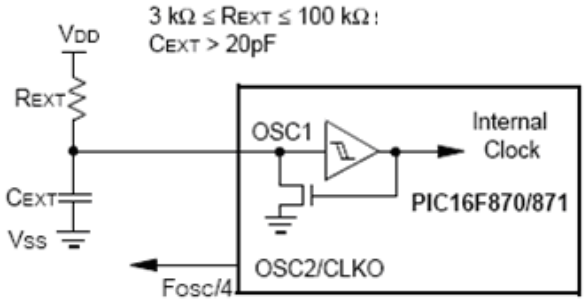


Figura 1.22 Oscilador de modo RC

1.4 PROTÓN

1.4.1 Características Generales

Protón es un programa editor de texto como un bloc de notas, pero con la diferencia que está hecho exclusivamente para facilitar la programación de los microcontroladores PIC.

Los procedimientos para programar son muy sencillos, los cuales deben estar acorde con el modelo de microcontrolador a usarse, con esto se debe elegir el microcontrolador en este caso es el PIC 16F877A, necesita de un compilador, ensamblador y programador para que esté completo**.**

1.4.2 Programación de Microcontroladores PIC en Proton IDE PLUS

Protón es un simple y fácil lenguaje de programación. Tiene sólo unas pocas normas y el control de estructuras que definen su gramática. En este tutorial vamos a aprender acerca de algunos principios básicos de este gran lenguaje de programación. Los códigos presentados aquí no pretenden ser programados en el microcontrolador, como tal, sino que se dan para explicar el tema. Una vez que haya pasado por esta introducción al lenguaje de BASIC, sólo entonces se puede ir a las áreas específicas de su interés. Lo que se presenta aquí se repite muchas veces a través de los siguientes capítulos del tutorial.

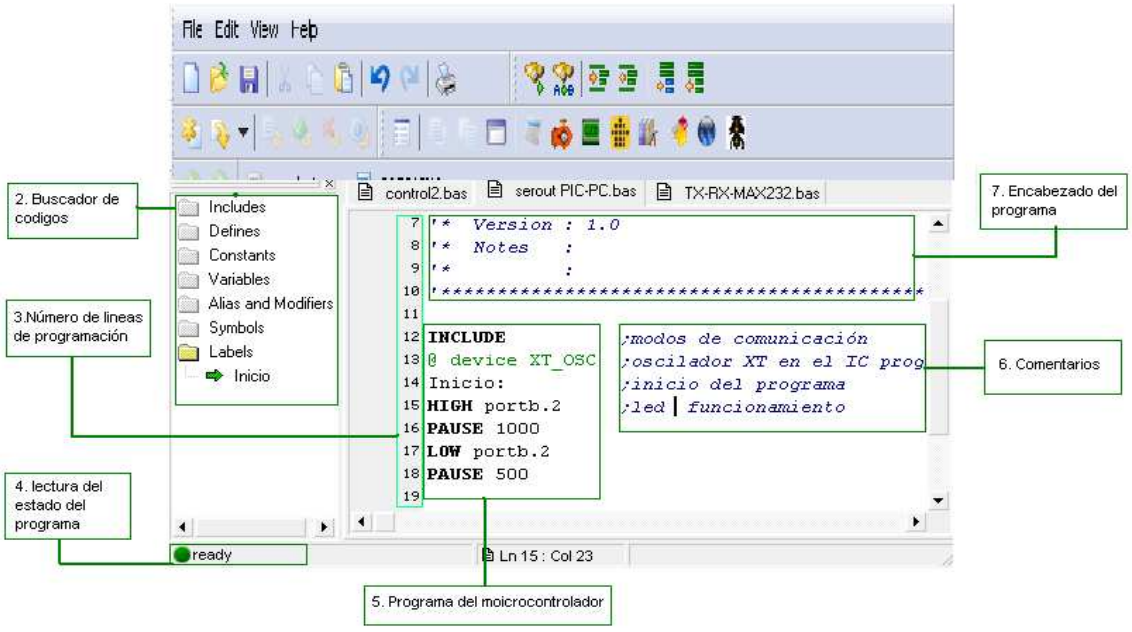


Figura 1.23 Partes de la pantalla del programa Protón.

1.4.2.1 Botones para Compilar

Figura 1.24 botones para compilar el programa

Estos dos botones están ubicados en la barra de herramientas superior de IDE. El botón de la izquierda es para compilar solamente. Cuando se activa sólo compila el programa y produce el .hex. El otro botón es para compilar y programar, a su lado hay una pequeña flecha, haga clic en la flecha y una lista de programadores aparece. Seleccione PIC KIT CLON, que es el que acabamos de configurar, si usted desea compilar y cargar el programa, solo se pulsa este botón.

1.4.2.2 Estructura del Programa en Protón

Un programa básico consta de:

- Programa de cabecera
- Declaraciones
- Explicación de los signos e identificadores
- Declaraciones y comandos.

Además de estas estructuras de base, algunos compiladores también permiten programación orientada a objetos, así como procedimientos y funciones. Sin embargo. Protón IDE no permite los procedimientos y funciones en el verdadero sentido, así como no es compatible con los objetos. Tiene enfoque orientado hacia el simple y sencillo llamado de instrucciones. La programación comienza en la parte superior, y la continúa hacia abajo. Sin embargo permite las repeticiones. Las primeras líneas del programa BASIC, difieren en la memoria de los microcontroladores, EEPROM, número de puertos y registros, etc., es necesario informar al compilador sobre el microcontrolador que se utilizará. En segundo lugar la velocidad de procesamiento depende de la frecuencia del cristal. Por lo tanto, a fin de calcular con precisión el calendario de funciones de retraso también es necesario informar al compilador de la frecuencia del cristal. Los programas de lenguaje BASIC usualmente comienzan así:

Device = **16F877A**
XTAL = 4

La primera línea indica el procesador y la segunda línea dice que el hardware utiliza 4MHz de cristal.

1.4.2.2.1 Declaraciones.

Son instrucciones especiales sobre los diversos dispositivos a utilizar, esto ayuda al compilador a generar instrucciones específicas. Por ejemplo, si estamos usando una pantalla LCD y se conecta en PORTD, entonces tenemos que informar las conexiones de nuestro LCD. Vamos a declarar este tipo de configuración generalmente después de la sección de encabezado utilizando los comandos:

```
Declare LCD_DTPIN PORTD.0
```

Hay un número de declaraciones, sin embargo, sólo las necesarias en el proyecto actual se fijan normalmente.

1.4.2.2.2 Identificadores.

Los identificadores son símbolos de texto especial que se utiliza para representar algo. Pueden ser utilizados como etiquetas para marcar ciertos lugares en el programa, de modo que el programa se puede hacer para ir a las etiquetas y luego continuar con el programa a partir de entonces. Del mismo modo los identificadores pueden usarse para nombrar algunas ubicaciones de memoria. Estos son por lo general variables llamadas, y son los identificadores más importantes en la programación. Los identificadores también se pueden utilizar como alias en cierto texto, de modo que en lugar de escribir el texto concerniente del identificador, el compilador inserta el texto pertinente en su lugar.

1.4.2.2.3 Las Declaraciones y Comandos

Hay tres tipos de declaraciones:

- Comparación y declaraciones condicionales
- Repetición y declaraciones de bucle
- Biblioteca de comandos

1.4.2.2.4 Comparación y Sentencias

Condicionales nos permiten comparar dos o más variables, los puertos, los pines del puerto o registros de funciones especiales y luego tomar la decisión de ejecutar un conjunto de instrucciones o de otro conjunto. Considerando la importancia de estas declaraciones, el lenguaje BASIC prevé diferentes construcciones de esto. Vamos a explorar estas a continuación.

1.4.2.2.5 La Repetición y de Bucle

Es una de las mayores ventajas de los microprocesadores. Podemos encargar al microcontrolador repetir continuamente ciertas instrucciones, ya sea indefinidamente, o hasta que una determinada condición existe. Por ejemplo, para mantener un LED, hasta que la temperatura es alta desde un punto de ajuste.

1.4.2.2.6 Biblioteca de Comandos

No son propiamente los comandos del lenguaje BASIC, pero son proporcionados por el fabricante del compilador para hacer las tareas comunes. Por ejemplo, un comando de biblioteca para mostrar algunos datos sobre la LCD o para leer los datos analógicos de un pin de entrada.

1.4.2.2.7 Etiquetas

Para marcar las declaraciones que el programa podría hacer referencia con los comandos Goto, Call o Gosub, PROTON utiliza las etiquetas de línea. A diferencia de los antiguos BASICS, PROTON no permite o exige a los números

de línea y no requiere que se etiquete cada línea. En cambio, cualquier línea puede comenzar con una etiqueta de línea, que es simplemente un identificador seguido de dos puntos (:).

```
PRINT "Hola Mundo"
GOTO LAB
```

Los nombres de etiqueta pueden ser de hasta 32 caracteres de longitud y puede contener cualquier caracter alfanumérico, pero no debe comenzar con un valor numérico.

1.4.2.2.8 Variables

Las variables se utilizan para almacenar datos temporalmente o para mantener números que se utilizarán en los cálculos. El número de variables que pueden ser utilizados en un programa depende de la memoria RAM del microcontrolador, la arquitectura del Microcontrolador, la parte de la memoria RAM, la memoria del programa.

Las variables son declaradas mediante una instrucción Dim, seguida del nombre de variable y su tipo de datos. La declaración Dim puede aparecer en cualquier lugar sin embargo es una buena práctica de programación ubicar todas las declaraciones Dim en la parte superior del programa....

```
Dim Dog As Byte ' Crea una variable de 8-bit (0 to 255)
Dim Cat As Bit ' Crea una variable de un bit (0 or 1)
```

1.4.2.2.9 Acceso a los Puertos y Registros

Son estructuras especiales internas del microcontrolador. Los puertos han sido nombrados como PORTA, PORTB, PORTC, etc. Todos los puertos son bidireccionales, es decir, que pueden ser utilizados para leer el estado del PIN o establecer el estado del PIN. La mayoría de los puertos son de 8 bits, pero

algunos son de menos salidas. Los puertos y sus bits se pueden acceder de la misma manera que las variables.

También se puede utilizar en las expresiones matemáticas. Además de los puertos hay una serie de registros internos con funciones específicas, que se asignan a los bits específicos de estos registros. A pesar que estos registros tienen una dirección especial, en la memoria y estas direcciones se utilizan para acceder a ellos. BASIC PROTON hace que sea más fácil acceder a estos registros por sus nombres para luego tratarlos como cualquier otra variable.

Estos nombres están predefinidos, y varían de acuerdo con el microcontrolador a utilizar. El compilador Basic de PROTON sabe que los puertos y los registros están disponibles en el microcontrolador seleccionado

```
PORTA = %01010101 ' escribir el valor binario en PORTA
```

1.4.2.2.10 Símbolos

Los símbolos son en realidad una manera de simplificar las cosas. Se asigna a un alias o un registro, una variable o un valor constante. El alias se utilizará en el programa, el compilador reemplazará el alias con los datos reales, antes de compilación

```
Symbol LED = PORTB.0
 High LED
```

1.4.2.2.11 La Repetición o Bucles

Como ya hemos visto que nuestro programa se ejecuta de arriba a abajo. Sin embargo, si un conjunto de instrucciones se tiene que repetir una y otra vez de alguna forma debe haber una declaración en la parte superior que controle el estado de la instrucción. Esto proceso se denomina un bucle.

El bucle más simple se puede construir mediante el uso de una etiqueta, y luego utilizar una instrucción Goto para saltar a la etiqueta.

```
Device=18F2550
Symbol LED = PORTC.0
Inicio:
High LED
DelayMS 500
Low LED
DelayMS 500
GoTo Inicio
```

Para poder utilizar de forma fácil y aprovechar las ventajas de este editor de texto programador se debe acoplar diferentes programas, como son el compilador y ensamblador que es el programa PicBasicPro y un programador como lo es IC-Prog 1.05E.

Los cuales son sumamente necesarios para una correcta generación de código hexadecimal, en cual va hacer utilizado por el microcontrolador PIC.

Corregido y encontrado el compilador y ensamblador con el PicBasicPro damos clic en la pestaña programador (programmer) pero no aparece.

El aspecto del programador lo podemos observar en la figura 1.40.

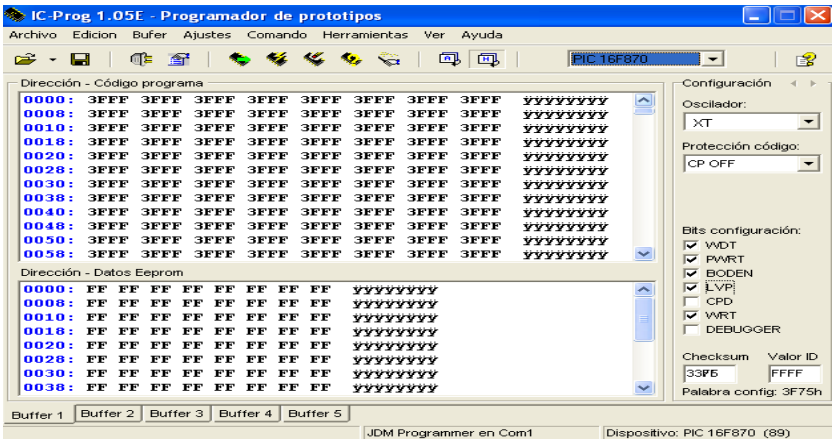


Figura 1.25 Ventana principal del programador IC-Prog 1.05E

Para que el programador **IC-Prog 1.05E,** funcione correctamente debemos tener instalado el driver. Caso contrario observará una serie de errores en la ejecución del programa.

Para instalarlo primero necesita encontrar el archivo **Icprog_drive.zip**, este archivo puede ser descargado por Internet completamente gratis, una vez descargado se procede a descomprimirle, y aparece la carpeta **icprog_driver,** en el cual contiene el archivo **icprog.sys**. Este archivo se debe mover junto al archivo ejecutable **icprog.exe** sólo de esta manera se lo podrá activar.

Una vez instalado el driver es importante habilitar al check box “Habilitar Driver NT/2000/XP” en caso de usar Windows NT, 2000 ó XP.

Para habilitar el **Driver**, vaya a la ventana principal del programador IC-Prog 1.05E se da clic en **Ajuste** y luego de un clic en **Opciones,** esta vez de un clic en **miscelánea** para habilitar driver señalar **Normal** y **Habilitar Driver NT/2000/XP** y luego clic en **OK**, ver figura 1.41.

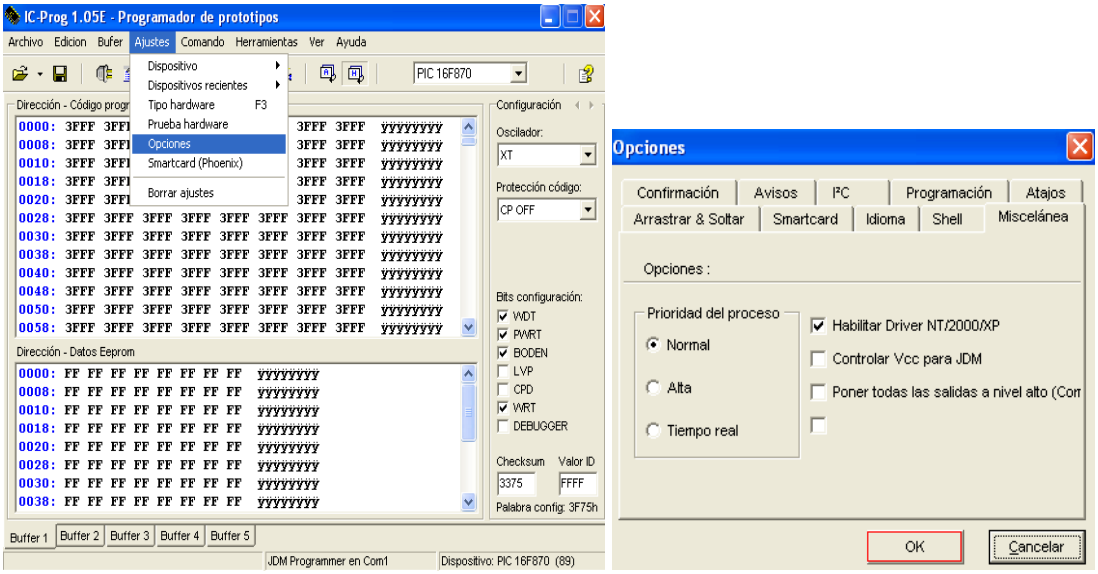


Figura 1.26 Ventana para habilitar el Driver NT/2000/XP

El IC-Prog1.05E, así que debe crearlo con la finalidad de que a futuro pueda acceder desde el Protón y programar directamente, para lo cual en la ventana

anterior figura 1.39 hacemos clic en la pestaña Programmer y colocamos Add New Programmer, inmediatamente aparece otra ventana figura 1.42 en la cual damos clic en create a custom programmer entry, y luego en next.

Figura 1.27 Ventanas de configuración del programador.

En la siguiente ventana colocamos el nombre del programador y siguiente, luego en Programmer Filename: Aquí colocamos el ejecutable del programa que es icprog.exe y presionamos encontrar automáticamente, una vez encontrado nos pide los parámetros pero no se debe colocar nada y dar un clic en finalizar**.**

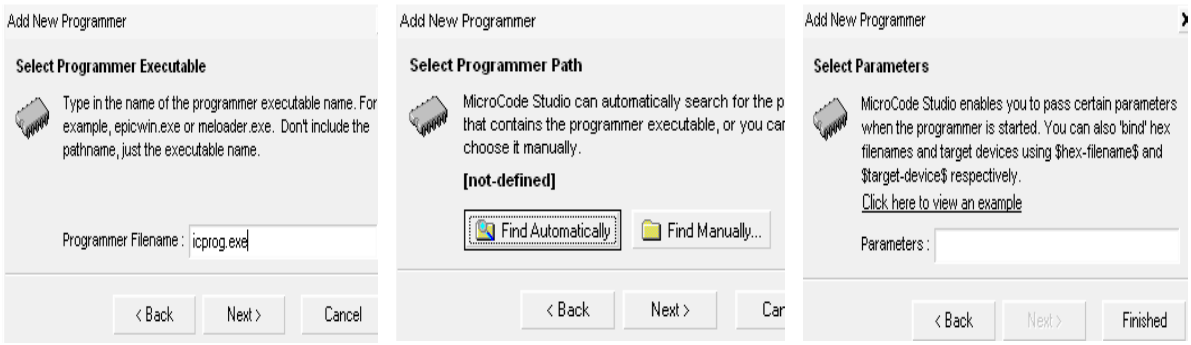


Figura 1.28 Ventanas de localización del programador utilizado para quemar el Micro.

Ya configurado todo como lo hemos hecho está listo para ser usado Protón, es muy fácil de utilizar si se conoce cada una de las herramientas que nos ofrece para la programación de microcontroladores PIC, en la siguiente imagen damos a conocer cada una de las partes de la pantalla que conforma Protón.

1.4.2.3 Utilización de Set de Instrucciones

Para la programación en Proton se utiliza el set de instrucciones proporcionado por el programa PicBasicPro, el cual es un lenguaje de alto nivel cuyo objeto es realizar las líneas de programación para el microcontrolador.
Cada instrucción tiene una tarea específica, dando, así a constituirse en las instrucciones que debe seguir el PIC en el cual va ha ser grabado, estas instrucciones no van directamente a colocarse en el PIC sino que se lo compila y ensambla para cambiarlo a datos en hexadecimal los cuales estos son grabados en los microcontroladores.

El set de instrucciones leído e interpretado por Protón no puede ser tomado como datos o variables por lo cual se coloca en negrilla y mayúsculas, en la tabla siguiente se encuentra las instrucciones con su respectiva acción en la programación, ver tabla 1.5.

Tabla 1.5 *Set de instrucciones Protón*

Set de Instrucciones	
@	Inserta una línea de código ensamblador.
ADCIN	Lee el conversor análogo.
ASM..ENDASM	Inserta una sección de código ensamblador.
BRANCH	GOTO computado (equiv. to ON…GOTO).
BRANCHL	BRANCH fuera de pagina (BRANCH largo).
BUTTON	Anti-rebote y auto_ repetición de entrada en el pin
CALL	Llamada a subrutina de ensamblador.
CLEAR	Hacer cero en todas las variables.
CLEARWDT	Hace cero el contador del Watchdog Timer.
COUNT	Cuenta el número de los pulsos en un pin.
DATA	Define en contenido inicial en un chip EEPROM.
DEBUG	Señal asincrónica de salida en un pin y baud.
DEBUGIN	Señal asincrónica de entrada en un pin fijo y baud.
DISABLE	Deshabilita el procesamiento de ON INTERRUP
DISABLE DEBUG	Deshabilita el procesamiento de ON DEBUG.

DISABLE INTERRUPT	Deshabilita el procesamiento de ON INTERRUPT.
DTMFOUT	Produce tonos telefónicos en un pin.
EEPROM	Define contenido inicial en un chip EEPROM.
ENABLE	Habilita en procesamiento de ON INTERRUPT.
ENABLE DEBUG	Habilita en procesamiento ON DEBUG.
ENABLE INTERRUPT	Habilita el procesamiento ON INTERRUPT.
END	Detiene la ejecución e ingresa en modo de baja potencia.
ERASECODE	Borrar bloque de códigos de la memoria
FOR...NEXT	Ejecuta declaraciones en forma repetitiva.
FREQOUT	Produce hasta 2 frecuencias en un pin.
GOSUB	Llama a una subrutina BASIC en la línea especifica.
GOTO	Continua la ejecución de la línea especifica.
HIGH	Saca un 1 lógico
HPWM	Salida de hardware con ancho de pulsos modulador.
HSERIN	Entrada serial asincrónica (hardware).
HSERIN2	Entrada serial asincrónica en segundo puerto.
HSEROUT	Salida serial asincrónica (hardware).
HSEROUT2	Entrada serial asincrónica en segundo puerto.
I2CREAD	Leer bytes de dispositivos I^2C
I2CWRITE	Graba bytes de dispositivos I^2C
IF..THEN..ELSE..ENDIF	Ejecute declaraciones condicionales.
INPUT	Convierte un pin en entrada.
LCDIN	Lee características desde una RAM de un LCD.
LCDOUT	Muestra características en un LCD
{LET}	Asigna el resultado de una expresión a una variable.
LOOKDOWN	Busca un valor constante en una tabla de constantes.
LOOKDOWN2	Busca un valor en una tabla de constantes o variables
LOOKUP	Obtiene un valor constante de una tabla.
LOOKUP2	Obtiene un valor constante o variable de una tabla.
LOW	Hace cero lógico a un pin especifico..

NAP	Apaga el procesador por un corto tiempo.
ON DEBUG	Ejecuta un DEBUG en Basic.
ON INTERRUPT	Ejecuta una subrutina BASIC en una interrupción.
OWIN	Entrada de dispositivos es un alambre.
OWOUT	Salida a dispositivos es unalambre.
OUTPUT	Convierte un pin en salida.
PAUSE	Demora una resolución 1 milisegundo.
PAUSEUS	Demora una resolución 1 microsegundo.
PEEK	Lee bytes desde un registro.
PEEKCODE	Lee bytes desde un espacio de código
POKE	Escribe bytes en un registro.
POKECOD	Escribe bytes en espacio de código programando en tiempo.
POT	Lee potenciómetros especificando el pin.
PULSIN	Mide el ancho de pulso en un pin.
PULSOUT	Genera un pulso hacia un pin.
PWM	Salida modulada en ancho de pulso por un pin especifico.
RANDOM	Genera un número seudo aleatorio.
RCTIME	Mide el ancho de pulso de un pin.
READ	Lee bytes del un chip de la EEPROM.
READCODE	Lee palabras desde un código de memoria.
REPEAT...UNTIL	Ejecuta declaraciones con condiciones de verdad.
RESUME	Continua ejecutando después de una interrupción.
RETURN	Continua en la declaración que sigue al último GOSUB.
REVERSE	Convierte un pin en salida en entrada.
SELECT CASE	Compara una variable con diferentes valores.
SERIN	Entrada señal asincrónica tipo (BS1).
SERIN2	Entrada señal asincrónica tipo (BS2).
SEROUT	Salida señal asincrónica (tipo BASIC stamp 1).
SEROUT2	Salida señal asincrónica (tipo BASIC stamp 2).
SHIFTIN	Entrada de señal sincrónica.

SHIFTOUT	Salida de señal sincrónica.
SLEEP	Apagar en procesador por un tiempo.
SOUND	Generar un tono o ruido blanco en un pin.
STOP	Detiene el programa de ejecución
SWAP	Intercambia los valores de dos variables.
TOGGLE	Hace salida a un pin y cambia su estado.
USBINIT	Inicializar USB
USBOUT	Salida USB.
WHILE..WEND	Ejecuta declaraciones mientras la condición sea cierta.
WRITE	Graba bytes en un chip EEPROM.
WRITECODE	Escribe palabras en código de memoria.
XIN	Entrada X -10.
XOUT	Salida X -10.

Especificación detallada de cada instrucción utilizada en Protón en el (ANEXO 4).

1.4.2.3.1 Funciones y Operaciones:

Todas las operaciones matemáticas son designadas y registradas con 16 bit de precisión. Las operaciones que soporta son:

Tabla 1.6 *Operaciones aritméticas, comparaciones y lógicas*

Operaciones Matemáticas	Descripción
+	Suma
-	Resta
*	Multiplicación
**	Multiplicación tope 16 bits
*/	Multiplicaron media 16 bits
/	División
//	Remanente (Modulo)
<<	Ir a la izquierda
>>	Ir a la derecha
ABS	Valor absoluto

COS	Coseno
DIG	Digito
MAX	Máximo*
MIN	Mínimo*
NCD	de código
REV	Invertir Bits
SIN	Sino
SQR	Cuadro or
&	AND
|	OR
^	Exclusive OR
~	NOT
&/	NOT AND
|/	NOT OR
^/	NOT Exclusive OR
=	Operador de asignación

Operadores de Comparación	Descripción
= o = =	Igual
<> o !=	No Igual
<	Menor
>	Mayor
<=	Menor o Igual
>=	Mayor o Igual

Operadores Lógicos	Descripción
AND o &&	AND Lógico
OR o ||	OR Lógico
XOR o ^ ^	OR Exclusivo
NOT AND	NAND lógico
NOT OR	NOR lógico
NOT XOR	NXOR lógico

1.5 PROTEUS.

El software de diseño y simulación Proteus VSM es una herramienta útil para estudiantes y profesionales que deseen mejorar habilidades para el desarrollo de aplicaciones analógicas y digitales.

El programa Proteus es una aplicación CAD (aplicación de diseño) que se compone de tres módulos:

1.5.1 ISIS (Intelligent Schematic Input System).

Es el modulo de captura de esquemas. Permite el diseño de circuitos empleando un entorno gráfico (figura 1.31), en el cual es posible colocar los símbolos de los componentes y realizar simulaciones de su funcionamiento, sin el riesgo de ocasionar daños a los circuitos.

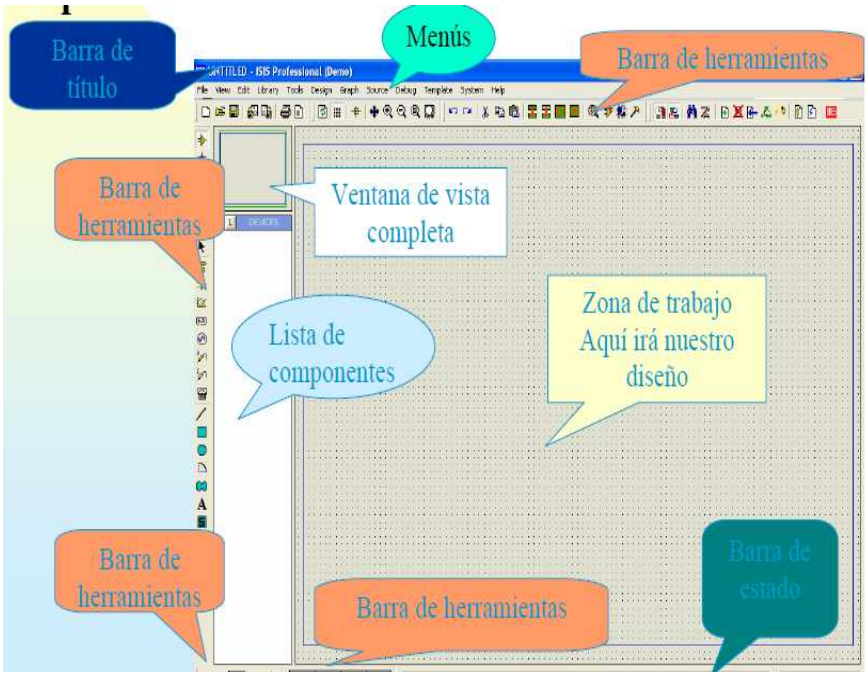


Figura 1.29 Pantalla de Proteus

1.5.2 Ares (Advanced Routing Modelling).

Es el modulo para realización de circuitos impresos. Proteus VSM tiene la capacidad de pasar el diseño a un programa integrado llamado ARES figura 1.29 en el cual se puede llevar a cabo el desarrollo de placas de circuitos impresos. Figura1.30 VSM (Virtual System Modelling): es el modulo de simulación.

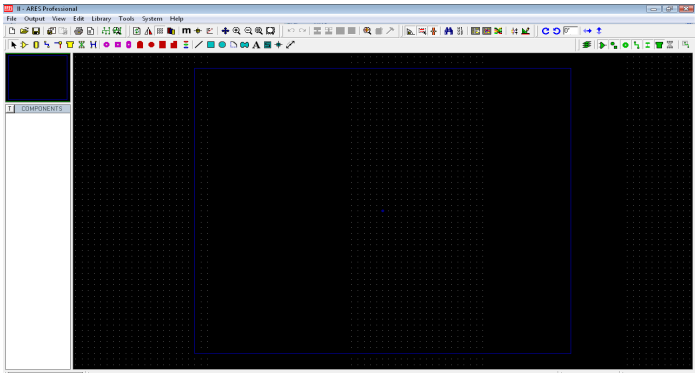


Figura 1.30 pantalla para realizar circuitos Impresos

1.5.2.1 VSM (Virtual System Modelling).

Es el modulo de simulación.

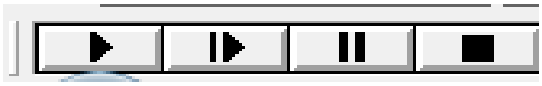


Figura 1.31 Barra de simulación.

1.5.2.1.1 Play 

Su pulsación hace que se inicie la simulación, cuando se está simulando cambia a color verde, mostrándose además, el tiempo que se lleva simulando, y la carga de CPU.

Figura 1.32 Barra en estado de simulación.

1.5.2.1.2 Pause 

Si nos encontramos en el modo *Play* su pulsación hace que la simulación se detenga, la tecla de pausa cambia de color,  y en la barra de simulación, se nos indica el tiempo transcurrido desde que se inició la simulación hasta que esta ha sido detenida. Una nueva pulsación de esta tecla hará que la simulación se reanude, en modo continuo.

Figura 1.33 Barra en estado de pausa.

1.5.2.1.3 Paso a paso 

Si nos encontramos en el modo *Play* su pulsación hace que la simulación se detenga, la tecla de pausa cambia de color, y en la barra de simulación, se nos indica el tiempo transcurrido desde que se inició la simulación hasta que ha sido detenida.

Figura 1.34 Barra de simulación paso a paso

Una nueva pulsación de esta tecla hará que la simulación se reanude, hasta que deje de presionarse o durante el tiempo especificado en las opciones de animación, es decir la simulación se hace paso a paso (a saltos)

1.5.2.1.4 Stop 

Si nos encontramos en el modo *Play* su pulsación hace que la simulación se detenga, saliendo el programa del modo simulación. CPU load : nos indica el % de utilización en el CPU, en aquellas simulaciones/animaciones en las que dicho % se acerque al 100%, la simulación no se estará realizando en tiempo real.

Figura 1.35 Barra de simulación boton de Stop

1.5.2.2 Barra de Título

Situada en la parte superior de la pantalla, en ella se muestra el icono del programa, el nombre del fichero abierto (Apuntes), la leyenda ISIS.

Professional (Demo), y en ocasiones mensajes de que el programa ha entrado en un modo particular de funcionamiento (por ejemplo *Animating* cuando se simula).

Figura 1.36 Barra de titulo

1.5.2.3 Barra de menús:

Permite el acceso a la mayor parte de las opciones del programa, sin embargo algunas opciones solo están disponibles en los iconos de las barras de herramientas.

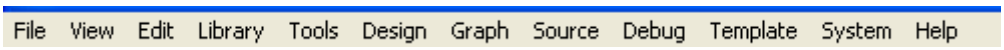


Figura 1.37 Barra de Menus

1.5.2.4 Barra de herramientas:

Son numerosas, el usuario las puede colocar en distintas posiciones de los bordes y son de suma importancia, ya que con ellos podemos seleccionar la opción indicada para realizar nuestro diseño. A continuación se mostraran algunos de los elementos más indispensables:

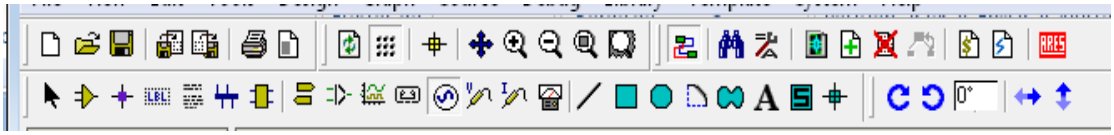


Figura 1.38 Barra de Herramientas.

1.5.2.5 Zoom All / View entire sheet. 

Muestra el esquema completo existente en la ventana de edición (hoja de trabajo)

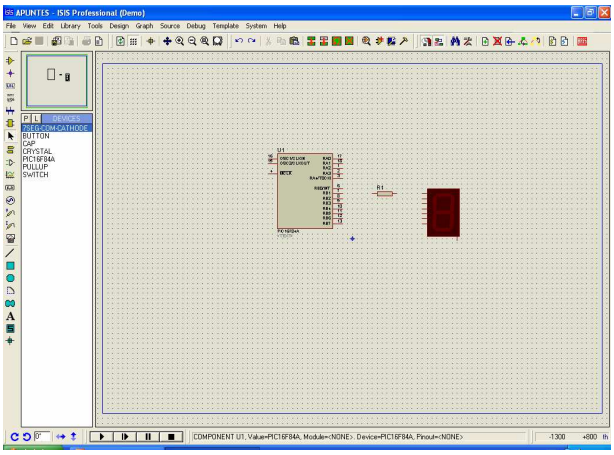


Figura 1.39 Boton Zoom

1.5.2.6 Zoom In / Increase Magnification.

A cada pulsación, aumenta la escala de la ventana de edición, que se mostrará con más detalle, manteniendo el centro de la pantalla.

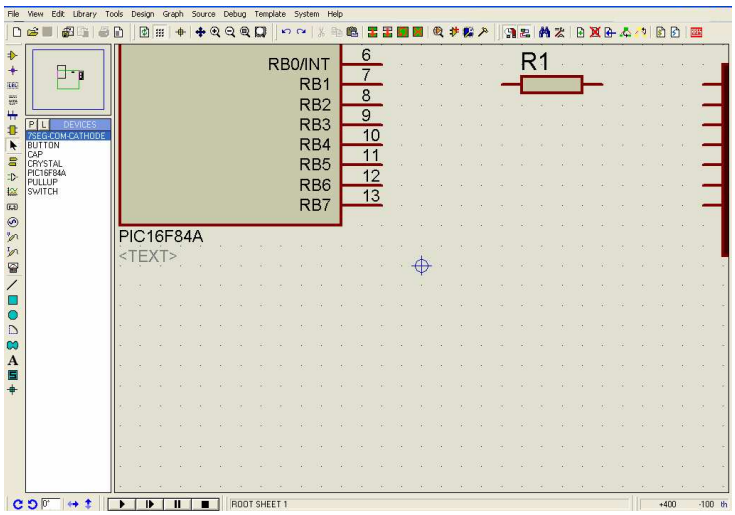


Figura 1.40 Boton Zoom de Incremento la pantalla.

1.5.2.7 Zoom Out / decrease magnification.

A cada pulsación, disminuye la escala de la ventana de edición, que se mostrará con menos detalle, manteniendo el centro de la hoja de trabajo.

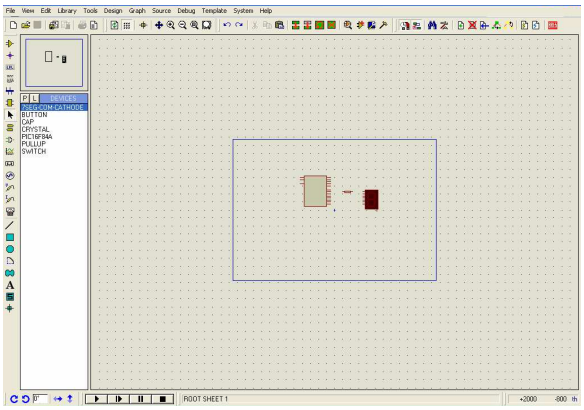


Figura 1.41 Boton Zoom decremento la pantalla.

1.5.2.8 Zoom to area / view select área : 

Permite seleccionar el área del esquema que se visualizará en la ventana de edición. Al activarlo, aparece un cursor con el que encerraremos la zona a visualizar.

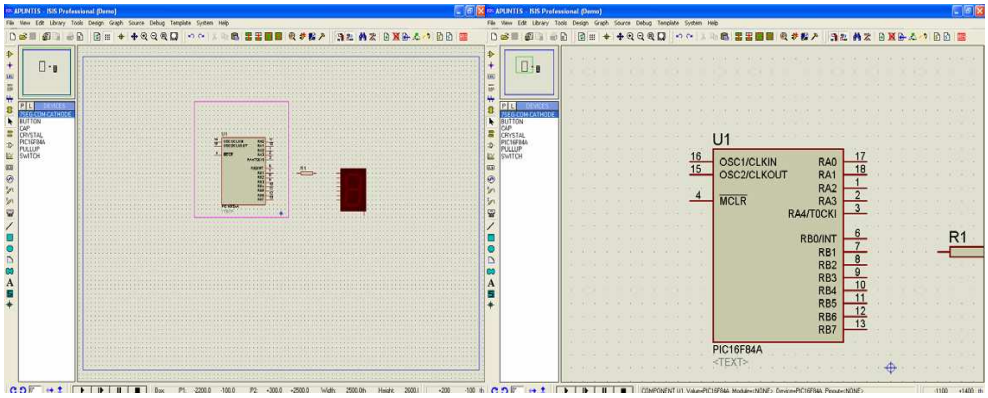


Figura 1.42 Boton Zoom lugar donde se ubique el cursor.

1.5.2.9 Modos de Trabajo en ISIS

La barra de herramientas de la izquierda nos permite seleccionar en cada instante el modo de trabajo en el que opera ISIS. Al hacer clic en el modo de trabajo este queda seleccionado y su icono muestra una visualización tipo “botón pulsado”. El modo de trabajo por defecto, y que más se utiliza es el modo componente, el resto de los modos excepto el de edición instantánea se utilizan ocasionalmente para poner puntos de unión, etiquetas o buses.

1.5.2.10 Component / Modo Componente

Permite la colocación y orientación de cualquier componente que se encuentre en la ventana de dispositivos. Además de permitir la edición de componentes que se encuentren dentro de la ventana de edición (diagrama esquemático).

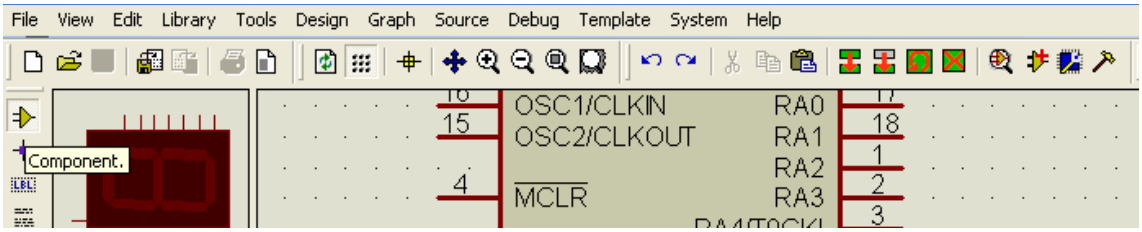


Figura 1.43 Boton de orientacion del los componentes.

1.5.2.11 Instant mode edit / edición en modo instantáneo:

Se utiliza normalmente cuando el esquema está totalmente realizado y solo resta editar los parámetros de los componentes**.**

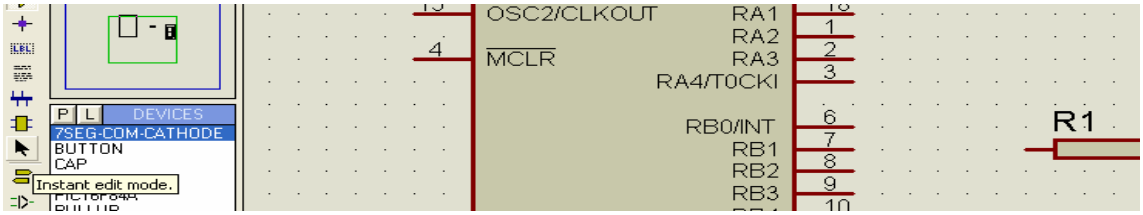


Figura 1.44 Boton para editar los elementos.

1.5.2.12 Bus :

Permite la colocación de buses y realizar el cableado de componentes en el diagrama esquemático

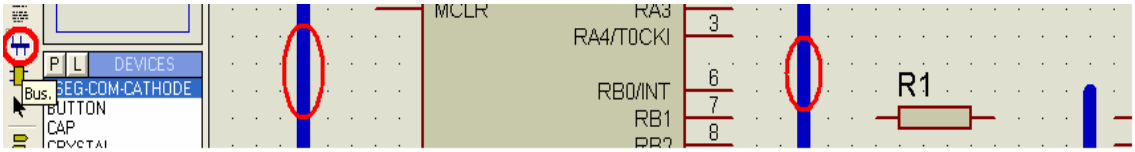


Figura 1.45 Boton para realizae cableado.

1.5.2.13 Wire Label :

El icono de etiquetado permite conectar terminales o cables entre sí mediante etiquetas (label). Para poner una etiqueta sobre un cable basta hacer clic sobre él siempre que estemos en le modo WIRE LABEL, abriéndose una ventana como la que sigue.

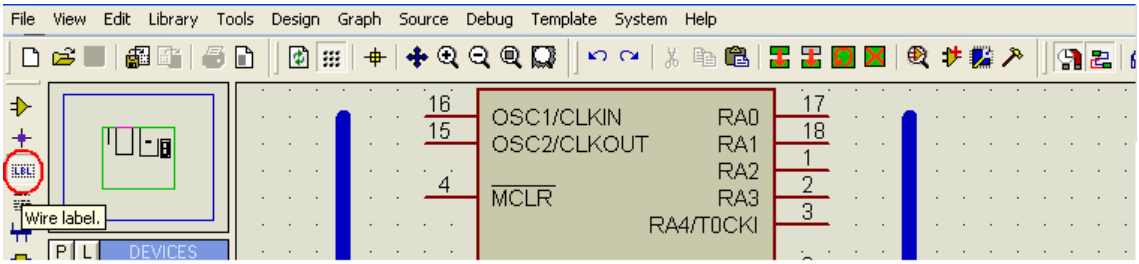


Figura 1.46 Boton conectar terminales mediante etiquetas.

Los nombre de las etiquetas pueden ser cualquiera, pero hay algunos reservados como son VCC VDD, GND VSS ya que identifican a los terminales de alimentación y tierra, utilizarlos provocará un cortocircuito.

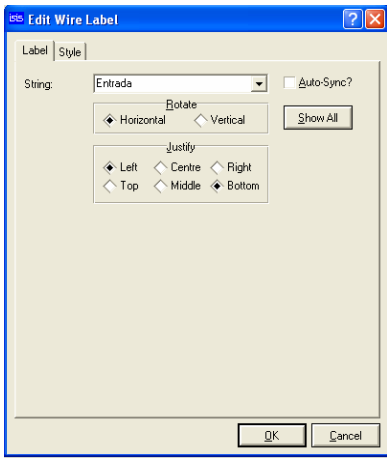


Figura 1.47 Pantalla para dar nombre alas etiquetas.

15.2.14 Junction dot : Puntos de unión.

En ocasiones es necesario poner puntos de unión entre los cables que interconexionan los componentes del circuito. Para ello es necesario poner el modo *Junction dot*. En este modo cada vez que se hace clic en la ventana de

edición se añade un punto de unión, para eliminarlo basta hacer dos veces clic con el botón derecho del ratón sobre él.

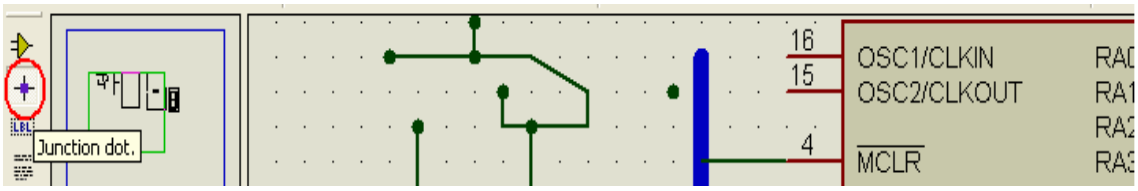


Figura 1.48 Boton Puntos de Union.

1.5.2.15 Text Script :

Permite situar y editar bloques de texto en el diseño. Si está activado y hacemos clic sobre la pantalla de edición se nos presentará una ventana como la que sigue, donde podemos escribir el bloque de texto que deseamos.

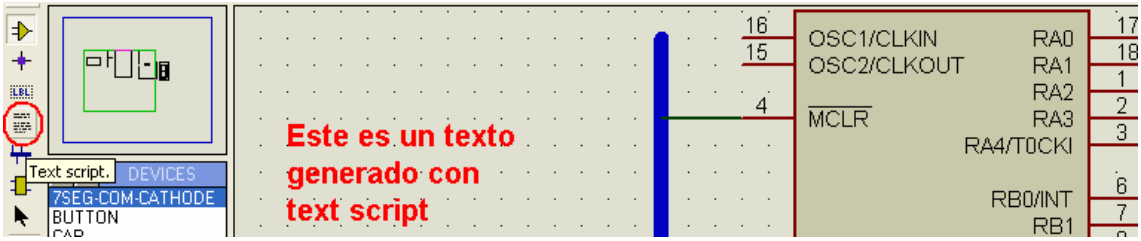


Figura 1.49 Boton para Escribir Texto.

Si pinchamos en la pestaña style, podemos definir parámetros como tipo de letra, tamaño, color, etc.

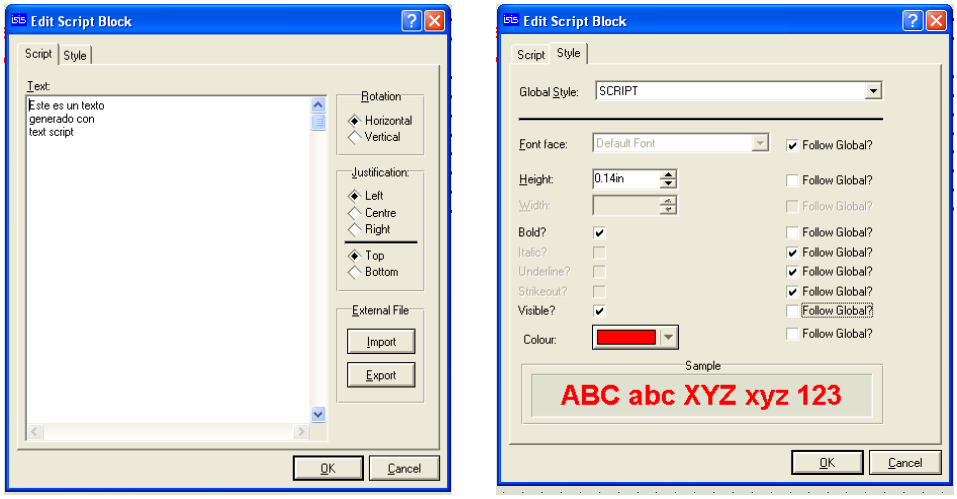


Figura 1.50 Pantalla para dar formato al Texto.

1.5.2.16 Terminales de Masa y Alimentación

Para acceder a los terminales de Masa y alimentación, pinchamos el icono y procedemos como con cualquier componente Default, Imput, Output, Bidir, Power, Ground, Bus.

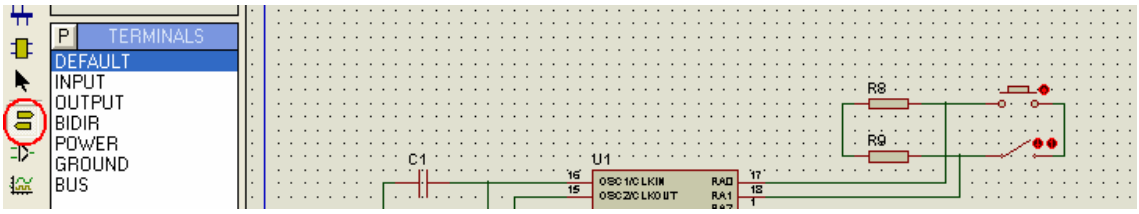


Figura 1.51 Boton para Alimentacion o Tierra.

1.5.2.17 Generadores de Señal para Análisis

Si activamos el icono, se nos presentará en la ventana de dispositivos los diferentes tipos de generadores que posee Proteus Las opciones que se nos presentan son:

DC SINE, PULSE, EXP, SFFM, PWLIN, FILE, AUDIO, DSTATE, DEDGE, DPULSE, DCLOCK, DPATTERN

La forma de insertar un generador en un punto es muy sencilla:

1. Se selecciona el generador deseado,
2. Se procede como si se tratara de cualquier Componente

Una vez conectado al circuito, ISIS le asigna automáticamente el nombre del nodo o la referencia a la patilla del componente que esté conectado al circuito integrado.

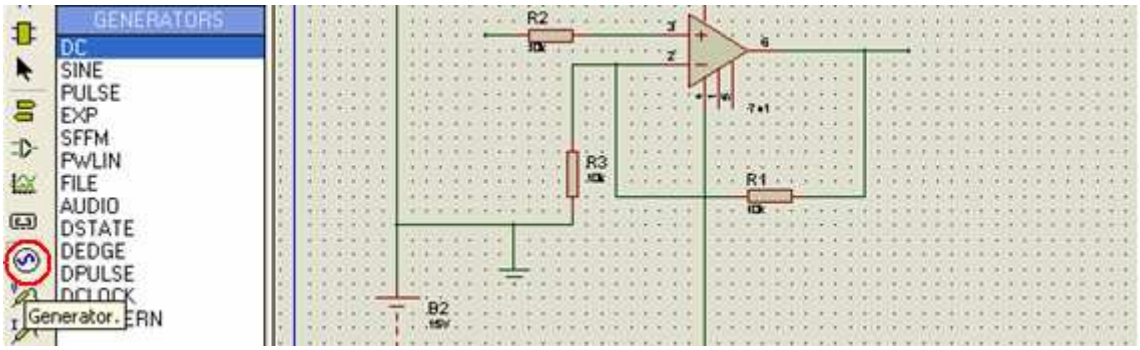


Figura 1.53 Boton Generador de señal.

1.5.2.18 Barra de Estado:

En ella se muestran mensajes informativos acerca de las opciones del menú, de los componentes de las simulaciones. Se indican las coordenadas del cursor, las unidades son en milésimas de segundo.

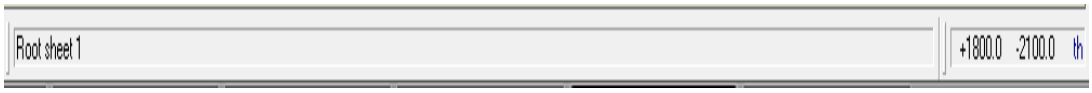


Figura 1.54 Barra de Estado.

1.5.2.19 Zona de Trabajo:

Es el lugar donde se creara el diseño.

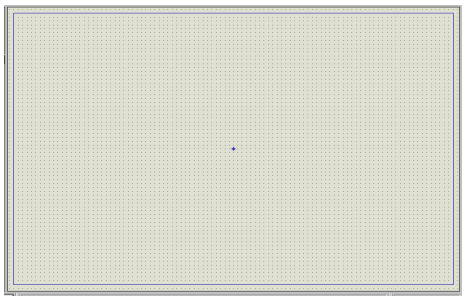


Figura 1.55 zona de Trabajo.

1.5.2.20 Ventana de Vista Completa/Zoom/Mapa del diseño:

Esta ventana nos muestra una visión global del diseño, y mediante el puntero podemos seleccionar que zona del diseño estará visible en la ventana de edición, si no fuese posible visualizar todo sobre dicha ventana. La zona visible

se encuentra encuadrada dentro de dicha ventana, mediante un recuadro verde.

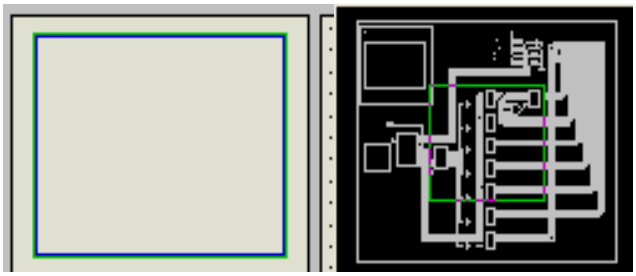


Figura 1.56 vista global del circuito.

1.5.2.21 Lista de Componentes:

En esta lista aparecen todos los componentes, terminales, pines, generadores, etc. Que se requieren introducir en el esquema, esta ventana dispone de 2 botones.

Figura 1.57 Botonpara lista de Componentes.

Que nos permite acceder a las librerías de componentes incluidas en ISIS.

Figura 1.58 pantalla de componentes.

El proceso de captura de captura de esquemas de circuitos electrónicos en ISIS consiste en realizar las siguientes tareas:

Elegir en las librearías de componentes todos aquellos elementos que se utilizan en el circuito a realizar. Situar los componentes que forman el circuito en la zona de trabajo.

Conectar las terminales de los componentes entre sí. Durante el comienzo del diseño de un circuito electrónico nos permitirá conocer su funcionamiento de los elementos que compone un circuito electrónico.

A continuación se les enseñara el mejor manejo del programa ISIS con algunos ejercicios de aplicación que irán de lo más fácil a lo complejo.

Capacitores.	TTL.
Resistencias.	CMOS.
Led.	Transistores.
Diodos.	Flip-flop (J,K)
Display de 7 segmentos.	Flip-Flop (D)
Microprocesadores.	Osciladores.
LCD.	Decodificadores.
Switches.	

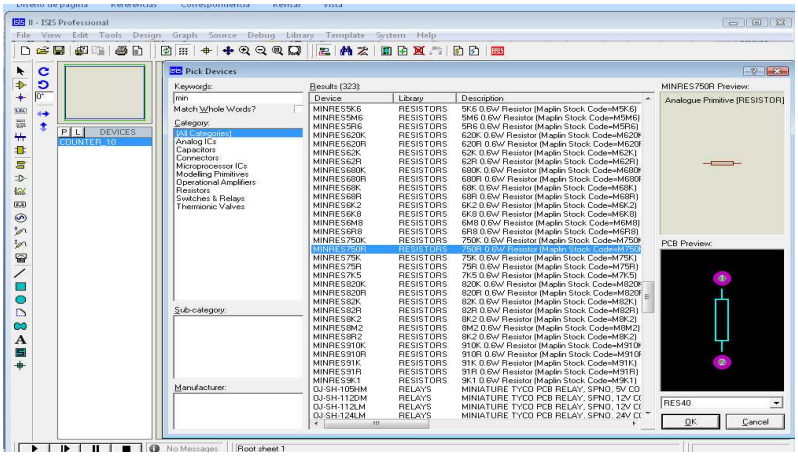


Figura 1.59 lista de componentes.

1.5.2.22 Conexión Entre Elementos.

La conexión mediante hilos se puede realizar en cualquier momento si colocamos el puntero encima de un terminal de cualquier componente comprobaremos que dicho puntero se transforma en una X indicándonos que se puede unir a otro terminal.

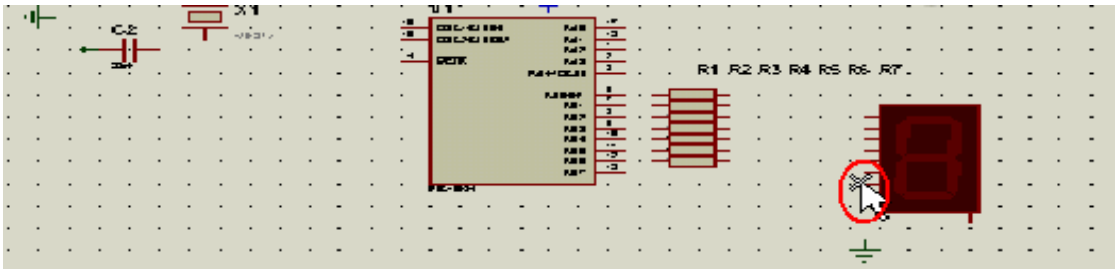


Figura 1.60 conexión entre elementos.

Cuando alcancemos un terminal de otro elemento, nos indicará el cursor en forma de X, como indica la *figura 1.60*, si en esta situación hacemos clic en el botón izquierdo del ratón la conexión eléctrica quedará realizada, pasando a color verde.

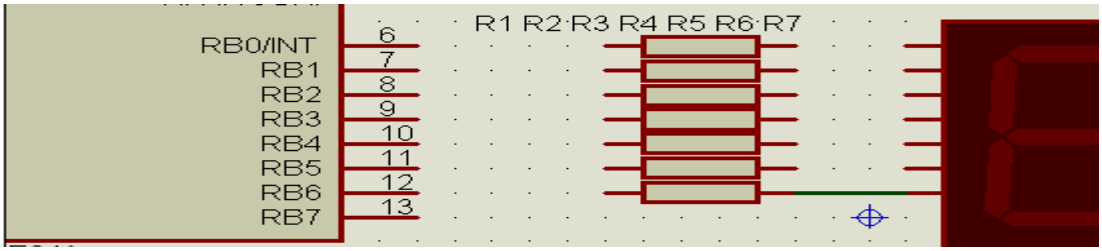


Figura 1.61 conexión entree elementos.

1.6 TECLADO MATRICIAL.

1.6.1 Descripción

Dispositivo de entrada de datos que consta de 16 teclas o pulsadores, dispuestos e interconectados en filas y columnas. Dispone de un conector SIL (Single In Line) macho de 8 pines que se corresponden con las 4 filas y las cuatro columnas de las que dispone.

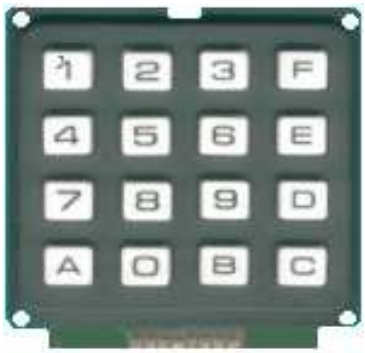


Figura 1.62 Teclado Matricial.

1.6.2 Funcionamiento

En la figura 1.63 se observa el esquema de conexión interno del teclado matricial y sus correspondientes pines de salida numerados de izquierda a derecha mirando el teclado tal y como se ve en la foto anterior. Cuando se presiona un pulsador se conecta una fila con una columna, teniendo en cuenta este hecho es muy fácil averiguar que tecla fue pulsada. También podemos ver el conexionado típico con el puerto B del µC PIC.

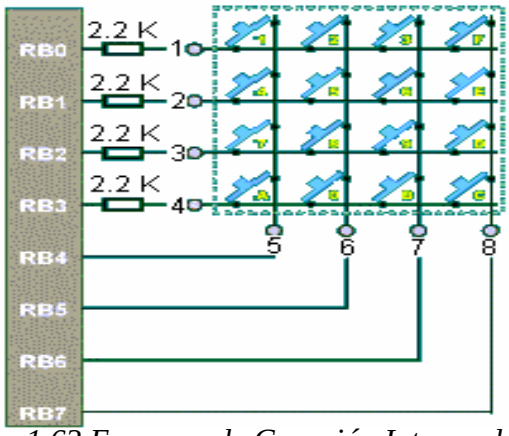


Figura 1.63 Esquema de Conexión Interna del Teclado.

Las resistencias de 2.2 K son necesarias para poder compartir el puerto del PIC independientemente del teclado y por ejemplo poder conectar un LCD o una barra de leds al mismo tiempo. Durante la fase de lectura del teclado la mitad de la puerta B es configurada como entrada y la otra mitad como lectura y durante la escritura en el lcd u otro sistema, la puerta B es configurada como salidas. Entonces se podrían cortocircuitar accidentalmente las salidas de los puertos provocando su destrucción, si pulsamos alguna tecla en ese momento, con lo que al poner estas resistencias evitamos este hecho y así si se produjera el cortocircuito tan solo circularía una pequeña corriente y el puerto del µC no correría ningún riesgo.

1.6.3 Ventajas del Teclado

Esta forma estratégica de colocar todas las teclas y conexiones permite manejar 16 pulsadores con solo llevar 8 cables. Si en lugar de esto se usa 16 pulsadores se tendría 32 cables, sin necesidad de pensar mucho se puede

ponerlos con un terminal común, con lo que se tendría 17 cables. La ventaja de usar la configuración matricial es clara.

El inconveniente que tiene es que para saber que tecla estás pulsando hace falta un microcontrolador (un PIC por ejemplo). Poder ir dando distintos valores a las columnas y saber, dependiendo de los valores que se reflejen en las filas, la tecla pulsada.

1.7 Módulo LCD[3]

El LCD (Liquid Crystal Display) es un dispositivo de visualización gráfico para la presentación de caracteres, símbolos o incluso dibujos (figura 1.64).

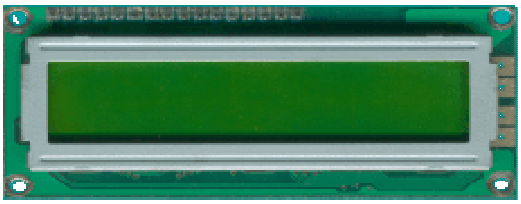


Figura 1.64 LCD 16X2

Habitualmente dispone de 2 filas de 16 caracteres cada una y cada caracter dispone de una matriz de 5x7 puntos (píxeles), aunque los hay de otros tamaños. Este dispositivo está gobernado internamente por un microcontrolador Hitachi 44780 y regula todos los parámetros de presentación, este modelo es el más usado y esta información se basará en el manejo de este u otro LCD compatible.[4]

1.7.1 Características Principales

- Pantalla de caracteres ASCII, además de los caracteres Kanji y Griegos.
- Desplazamiento de los caracteres hacia la izquierda o la derecha.
- Proporciona la dirección de la posición absoluta o relativa del caracter.
- Memoria de 40 caracteres por línea de pantalla.

[3] Ing. Costales, Alcívar - Curso Básico de Microcontroladores PICs- 2007
[4] Ing. Costales, Alcívar - Curso Básico de Microcontroladores PICs- 2007

- Permite que el usuario pueda programar 8 caracteres.
- Conexión a un procesador usando un interfaz de 4 u 8 bits.

1.7.2 Descripción de Pines

Normalmente un LCD tiene 14 pines. Si tiene 15 o 16 pines, entonces los pines restantes son usados para producir iluminación posterior, a continuación en la tabla 1.5, se detalla la descripción de cada uno de los pines que contiene un LCD, para nuestro proyecto utilizaremos un LCD de 16 pines ya que es indispensable la iluminación en la pantalla para la visualización de los mensajes en casos falta de visibilidad en la pantalla.

PIN	Nombre	Nivel	Descripción
1	Vss	0 V	Tierra
2	Vcc	5 V	Alimentación
3	V_{EE}	Pot	Potenciómetro (Contraste)
4	RS	Lógico	0L instrucción, 1L dato
5	R/W	Lógico	1L lee, 0L escribe en el LCD
6	E	Lógico	Pulso de habilitación
7 a 14	DB0-DB7	Lógico	BUS de datos

Tabla. 1.8 Descripción de pines del LCD.[5]

1.7.3 Conexión con bus de datos de 4 bits[5]

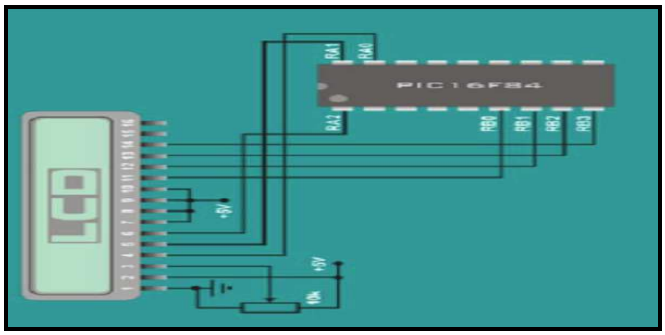


Figura 1.65 Conexión con bus de 4 Bits.

[5] Ing. Costales Alcívar - Curso Básico de Microcontroladores PICs- 2007

El LCD tiene 14 pines. Algunos tienen 1 o 2 pines más. Estos pines adicionales son usados para iluminación posterior.

- Pin1: GND.
- Pin2: Vcc. (5V).
- Pin3: para el contraste.
- Pines 4 a 6: para control.
- Pines 11 a 14: bus de datos.

1.7.4 Conexión con bus de datos de 8 bits.

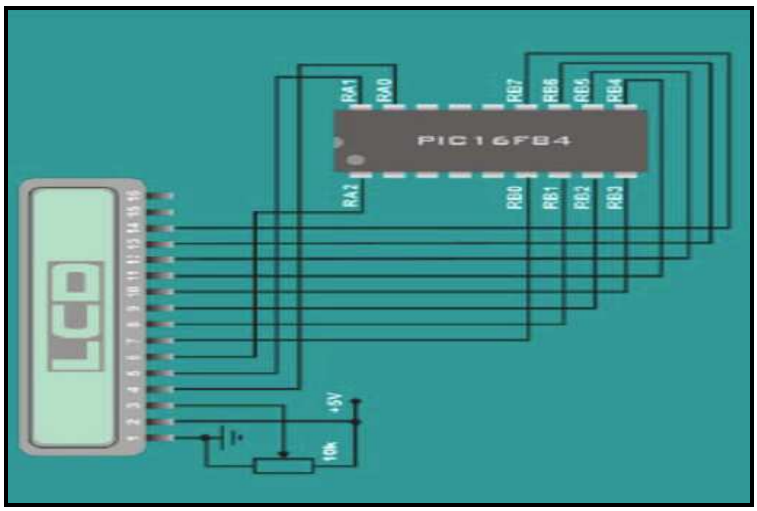


Figura 1.65 Conexión con bus de 8 Bits.

El bus de datos ocupa todo el pórtico B del PIC

- Pin1: GND
- Pin2: Vcc(5V)
- Pin3: para el contraste
- Pines 4 a 6: para control
- Pines 7 a 14: bus de datos

CAPITULO 2

2.1 DESCRIPCION GENERAL DEL CIRCUITO MEDIDOR DE GASOLINA DIGITAL.

Se desarrollo un circuito digital a partir del circuito análogo existente en los medidores que poseen los automotores que toma una variable natural siendo esta el nivel de gasolina. Para esto se dispuso de diferentes dispositivos mecánicos y electrónicos.

El mecanismo mecánico es la parte principal del sistema que es un potenciómetro encapsulado al tanque de combustible y conectado en forma de resistencia variable, un brazo cursor (resorte cursor), conductores impresos (contacto doble), una placa porta resistencias y conexiones eléctricas. La palanca en cuyo extremo se encuentra el flotador (orientable o fijo, en función de la aplicación) de nitrófilo resistente al combustible, está fijada en el eje giratorio (pivote) del potenciómetro y, por tanto, también en el resorte cursor. Y la forma del flotador y de su palanca están adaptados a la conformación respectiva del depósito de combustible el cual determina el nivel de gasolina existente de una manera análoga conforme este se nueva.

En la parte de dispositivos electrónicos consta de un PIC, una pantalla LCD 2x16 y un teclado matricial. El PIC es el que toma la señal análoga del sensor mecánico para que mediante un programa transforme la señal análoga a señal digital y muestre estos datos en la pantalla LCD 2x16 de manera numérica, mientras que con el teclado al oprimir una tecla respectiva nos permitirá acceder a las diferentes opciones que se desplegará en la pantalla del LCD 2x16 como cálculo de galones, cálculo de precio de galones cantidad de galones que ingresó al tanque etc.

A continuación se observa el diagrama de bloques del diseño del sensor de gasolina digital.

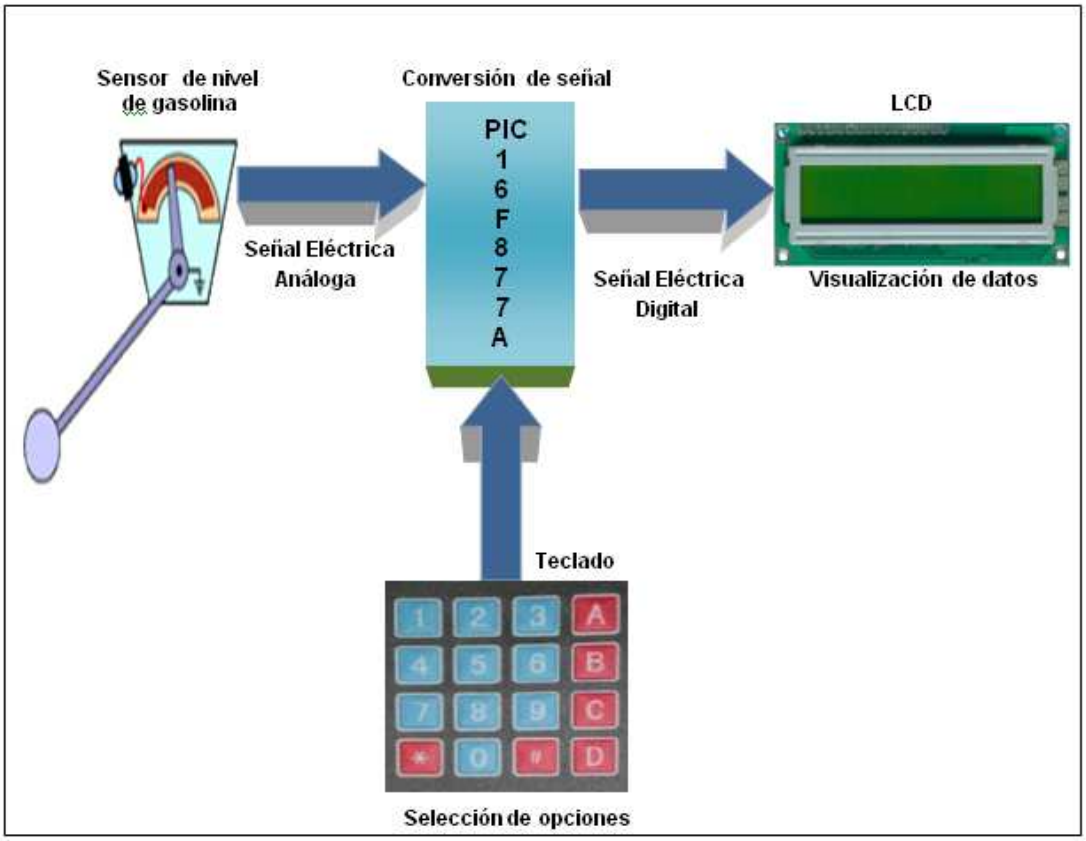


Figura 2.1 Diagrama de bloques del circuito digital medidor de gasolina.

2.2 FUNCIONAMIENTO GENERAL DEL CIRCUITO ELECTRÓNICO MEDIDOR DE GASOLINA DIGITAL.

Al variar el nivel de gasolina, el brazo detector, fijamente unido a través del pivote con la palanca del flotador, se desliza con sus cursores especiales a lo largo de las pistas resistivas del potenciómetro permitiendo así variar el voltaje de salida en uno de los extremos del potenciómetro (sensor de nivel). Unos topes de fin de carrera limitan el margen para los niveles mínimo y máximo. La tensión de funcionamiento es de 1.8……..4.98 V.

Estos voltajes son entregados a la entrada del PIC en el pin RA0 (2) el mismo que procesará los niveles de voltaje de forma análoga para luego entregar al LCD en forma digital y visualizar en forma numérica los datos obtenidos durante el proceso de la digitalización

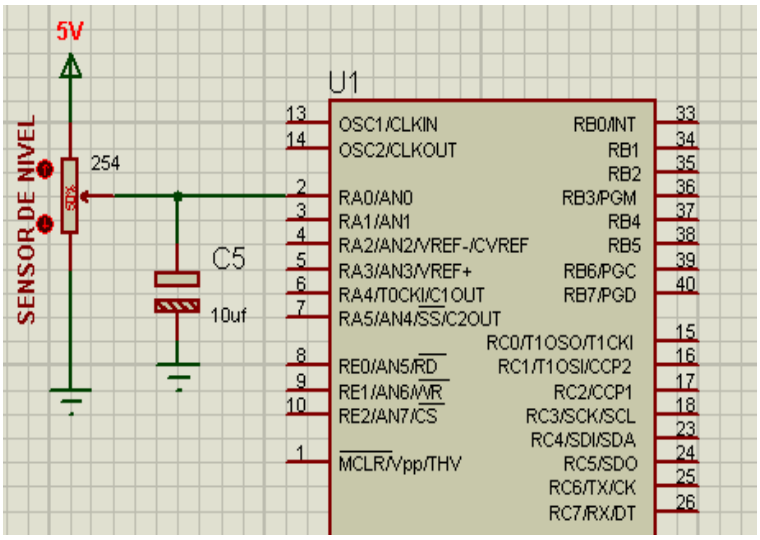


Figura 2.2 circuito de conexión del sensor de nivel de combustible hacia el PIC.

Cuando el tanque de combustible está a su capacidad máxima el sensor marcará un voltaje de entrada de 4.98V, este voltaje en el PIC transformará a su equivalente en galones y precio cuando en el tanque esté a su capacidad mínima el sensor marcará 1.5 V dándonos un mensaje que el combustible está por terminar, que se mantendrá con la reserva de combustible para que el conductor lleve el automotor hasta una estación de servicio para abastecer de gasolina. Mientras que con el teclado se podrá acceder a diferentes opciones que el circuito que nos permitirá hacer (cálculo de galones, cálculo del precio total de galones, cantidad de galones que ingreso al tanque de gasolina)

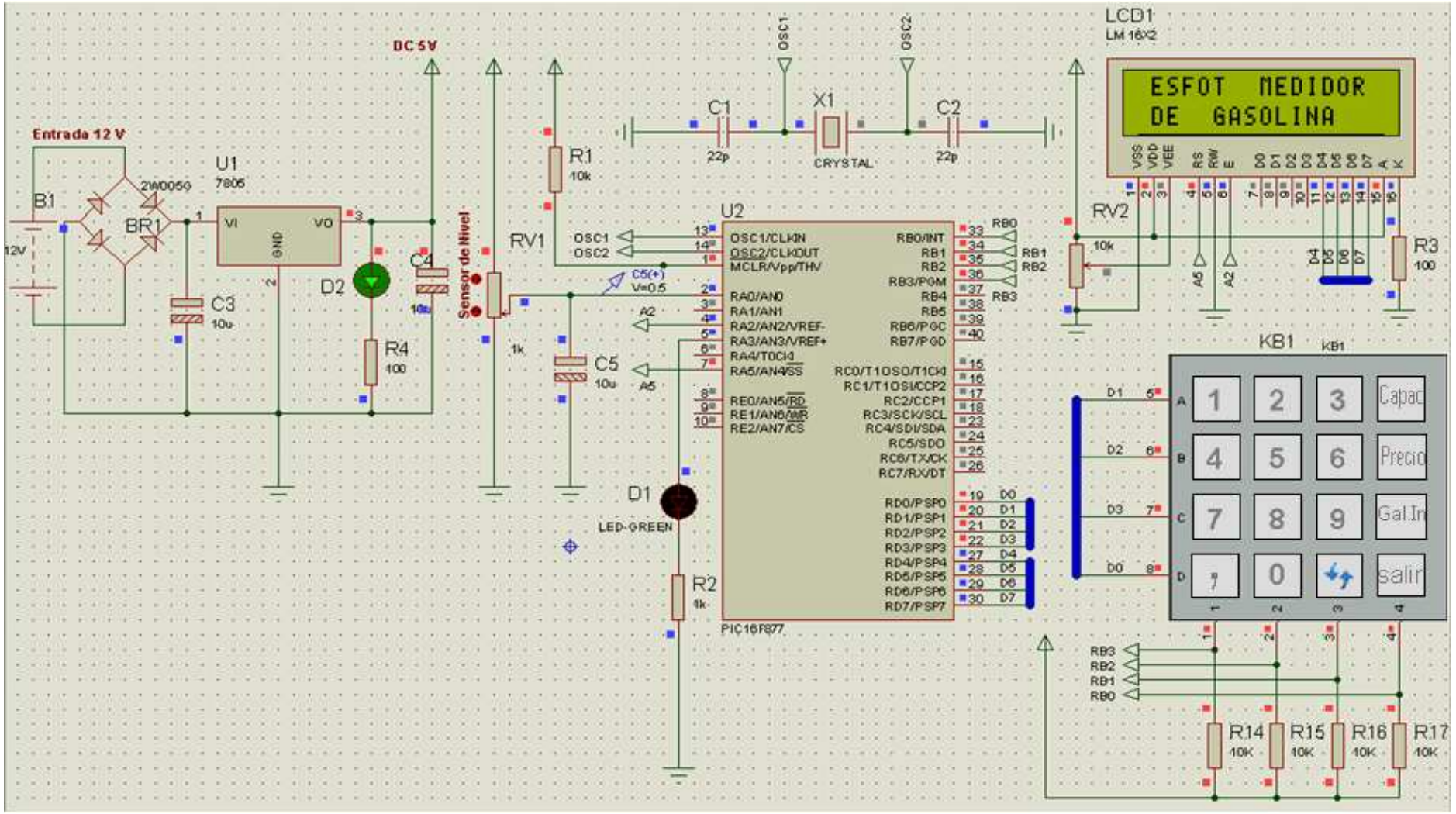


Figura 2.3 circuito lógico del medidor de combustible.

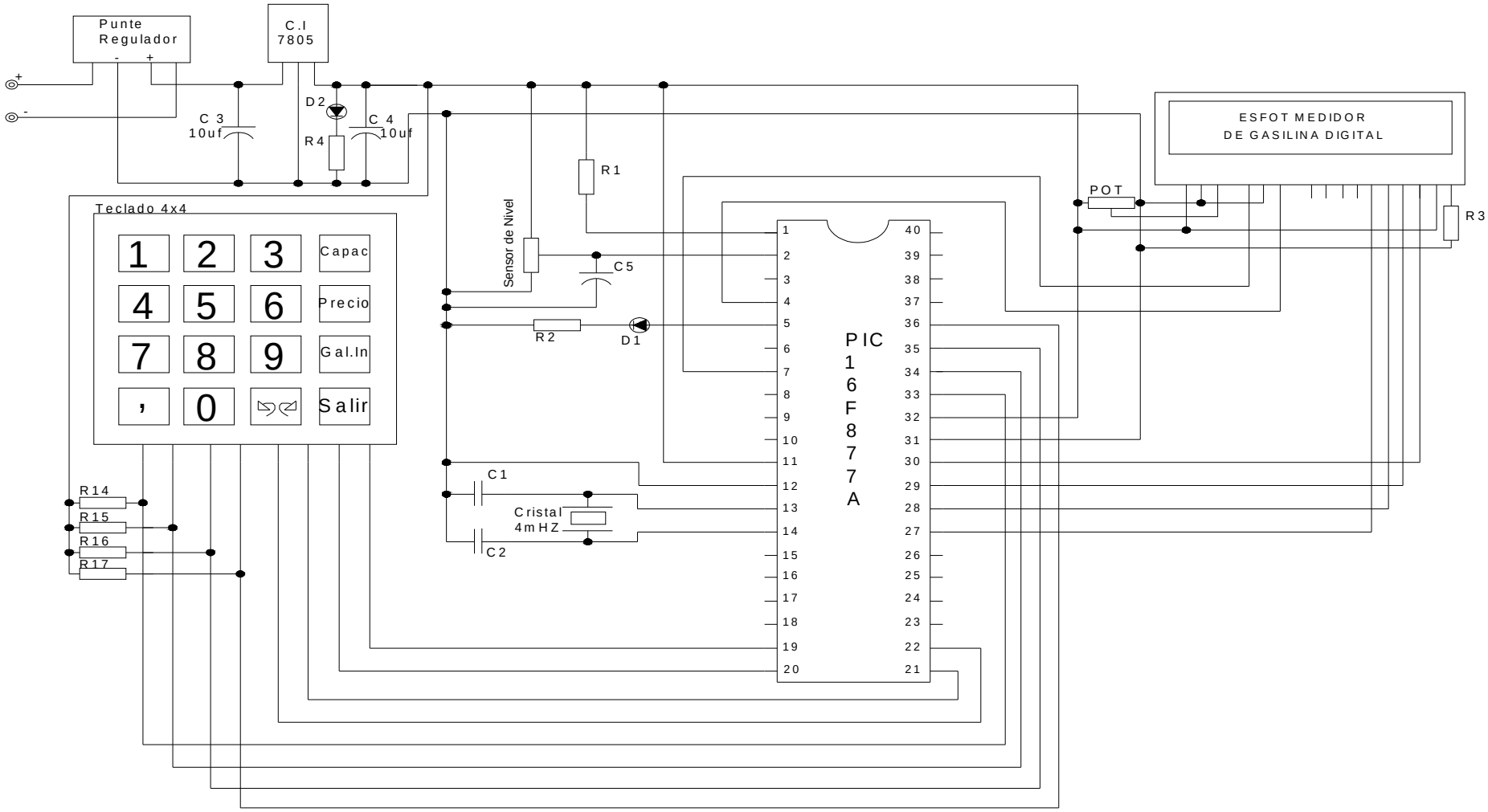


Figura 2.4 diagrama circuital del medidor de combustible.

2.3 DESCRIPCIÓN DEL FUNCIONAMIENTO POR ETAPAS DEL MEDIDOR DE GASOLINA DIGITAL

2.3.1 Etapa de Alimentación

La alimentación del circuito es el voltaje de la batería del automotor de $+12V_{DC}$. Dicho voltaje se regula a $+5V_{DC}$ mediante el circuito integrado 78L05, se utilizo los condensadores C3, C4 de 10µf a 25V por recomendaciones del fabricante. Los $+5V_{DC}$ se utilizan para la polarización de los sensores, y demás dispositivos digitales que funcionan con los niveles de voltaje TTL. Adicionalmente se conectó un diodo LED con una resistencia de 330 Ω como indicador de encendido y apagado del circuito digital.

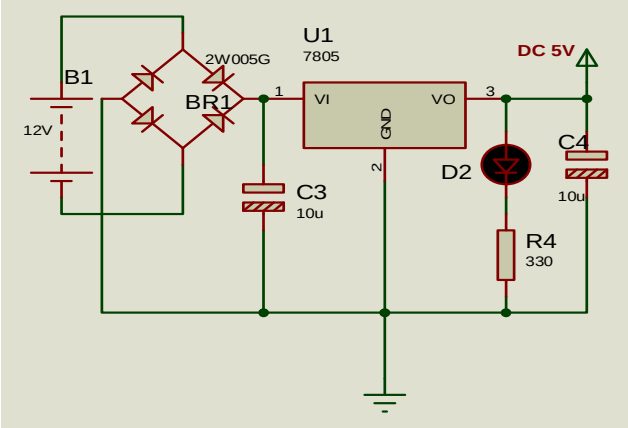


Figura 2.5 Circuito de la etapa de alimentación.

Figura 2.6 Circuito en etapa de alimentación.

2.3.2 Etapa de Polarización del Sensor.

Como se explicó un sensor de nivel está constituido por un potenciómetro mecánico que necesita ser alimentado para poder variar el voltaje de salida se, conectó directamente de los terminales del circuito CI 78L05 hacia los terminales del sensor de Nivel.

El sensor consta de tres terminales dos fijos y uno móvil la alimentación +5Vdc ingresa por uno de los terminales fijos, mientras que el otro terminal fijo se une con el terminal móvil, para de esta manera poder tener un divisor de tensión conjuntamente con uno de los terminales que se conecta al PIC.

Cuando la parte móvil de sensor alcance su valor mínimo, el voltaje circula por la resistencia interna del sensor y cuando la parte móvil del sensor alcance su valor máximo el voltaje circula hacia el microcontrolador.

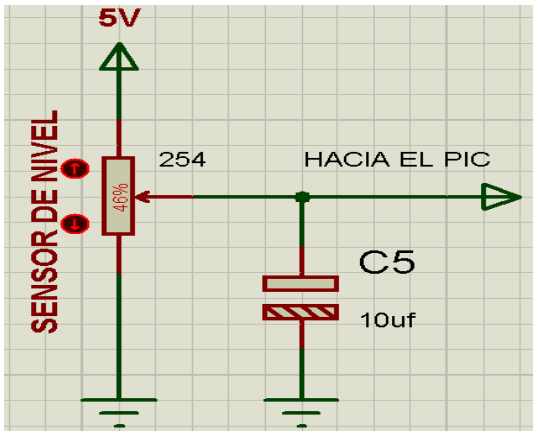


Figura 2.7 circuito de alintación hacia el sensor.

Figura 2.8 Circuito de alintación hacia el sensor Circuito en el Protoboard.

2.3.3 Etapa de Acondicionamiento del Sensor Hacia el PIC 16F877A

Para conectar hacia el PIC se toma desde el punto que está unido por el terminal de la resistencia y los terminales de sensor uno fijo y otro móvil que ingresa al pin dos del PIC 16F877A que corresponde al Pórtico A0 siendo este el encargado de transformar el voltaje análogo en muestras digitales para realizar la conversión respectiva, cuando el sensor está en su resistencia máxima en la entrada del Pin 2 del PIC se tendrá un valor de voltaje máximo de 4,94V y cuando el sensor este en su resistencia mínima en la entrada del Pin 2 del PIC se tendrá un valor de voltaje mínimo de 0,88V

Para la transformación de análoga a digital se toma 255 muestras del valor máximo de voltaje análogo para este caso 4,94V.

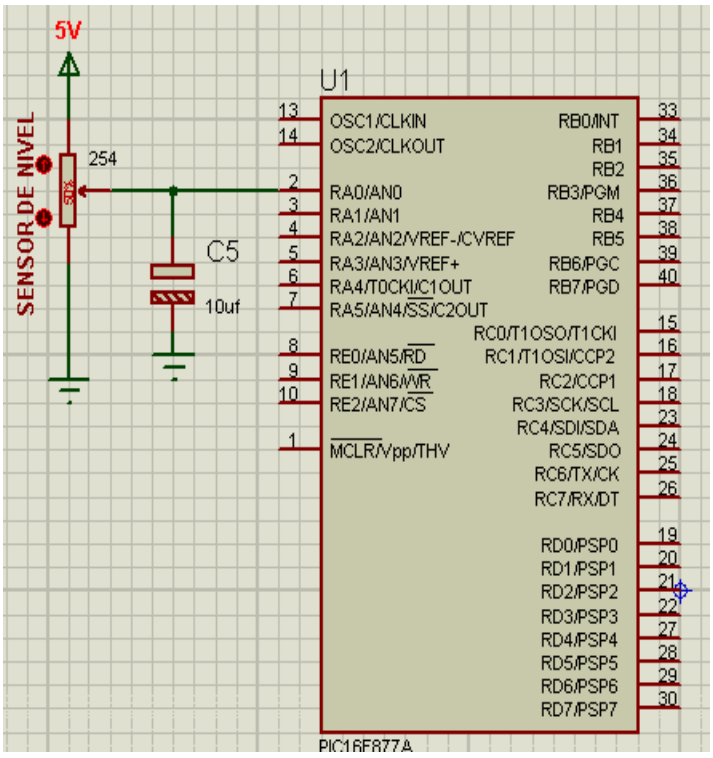


Figura 2.9 Circuito de conexión del sensor al PIC.

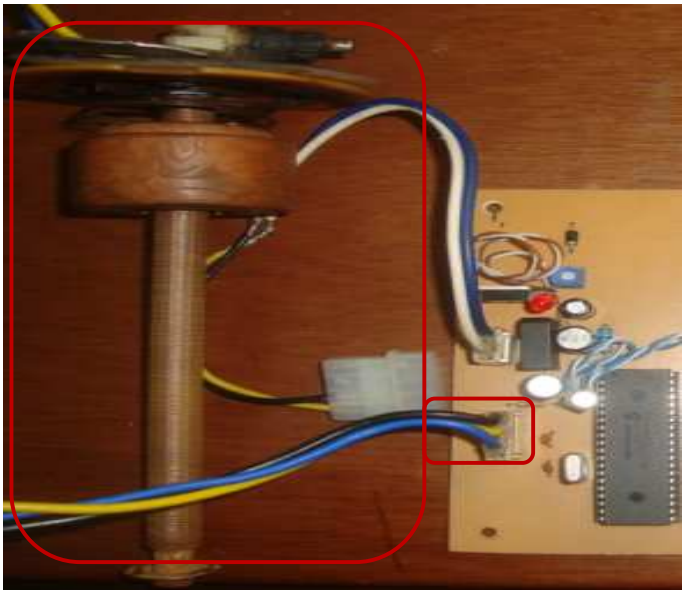


Figura 2.10 Circuito de conexión del sensor al PIC.

2.3.4 Conexión del PIC 16F877A al LCD.

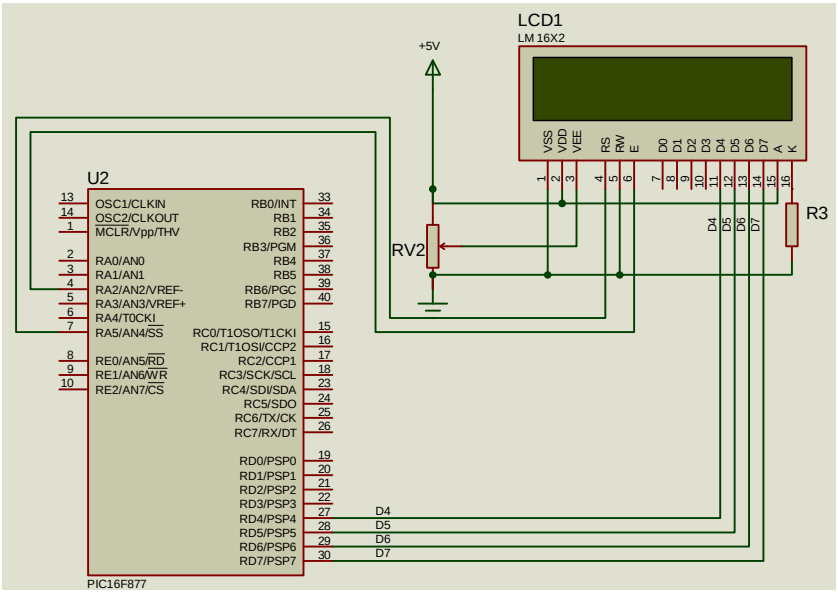


Figura 2.11 Conexiones del PIC16F877A con el LCD.

Figura 2.12 Conexiones del PIC16F877A con el LCD.

Una vez realizadas las operaciones del muestreo de la señal correspondiente será obtendrá los resultados. Se toma como salida de la información los 4 bits menos significativos del pórtico B, que se conectan con el bus de datos del LCD como se aprecia en la figura 2.8 además la conexión de un potenciómetro de 10KΩ para el contraste del LCD.

Las conexiones del LCD fueron dispuestas de la siguiente manera:

Pin 1: GND.

Pin 2: Vcc (5V).
Pin 3: para el contraste.
Pines 4 a 6: para control.
Pines 11 a 14: bus de datos.

2.3.5 Conexión de Teclado con el PIC 16F877A.

El teclado matricial de 16 teclas. Se trata del periférico de entrada por excelencia que va a permitir introducir todo tipo de datos se tendrá en cuenta es que, a pesar de tener 16 teclas, tan sólo son necesarias 8 líneas del microcontrolador para su total control. Ello es debido a su distribución matricial. En el caso que el teclado está conectado a las 8 líneas de la puerta (RD y RB) tal y como se muestra en el esquema eléctrico de la figura 2.11. Está organizado en 4 filas (A-B-C-D) que se conectan a RD0, RD1, RD2, RD3 y otras 4 columnas (1-2-3-4) que se conectan a RB0, RB1, RB2, RB3 la intersección fila-columna da lugar a una tecla. Es decir, si se pulsa por ejemplo la tecla 4, se unirá eléctricamente la fila FB con la columna C, que es tanto como decir que las líneas RD2 y RB3 del PIC se han unido.

Una subrutina del software encargada de explorar al teclado determinar qué tecla se ha pulsado. Por ejemplo, configura las líneas RB0-RB3 (las columnas) como salidas y RD0-RD3 (las filas) como entradas. Secuencialmente va activando cada una de las columnas al tiempo que lee el estado de las filas, cuando se detecta que una fila esté activada es porque se pulso una tecla. Basta conocer qué columna se activó en ese momento para sacar la relación fila-columna que define a cada tecla. Esta tarea conocida como "*barrido del teclado*" ha de repetirse de forma constante y periódica. De esta manera y, a la velocidad de trabajo del PIC.

Conexiones de las columnas	**Conexiones de las filas**
Columna 1 RB3	Fila A RD1
Columna 2 RB2	Fila B RD1

Columna 3 RB1 Fila C RD1

Columna 4 RB0 Fila D RD1

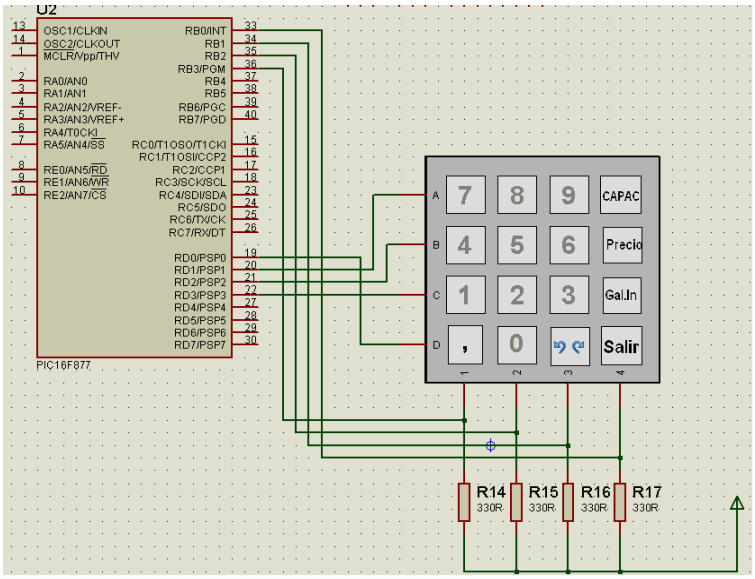


Figura 2.13 Conexiones del teclado al PIC 16F877A.

Figura 2.14 Conexiones del teclado al PIC 16F877A.

2.3.6 Conexión del Oscilador al PIC.

Aquí utilizaremos el cristal de 4 MHz, porque garantiza mayor precisión y un buen arranque del PIC. Internamente esta frecuencia es dividida por cuatro, lo que hace que la frecuencia efectiva de trabajo sea de 1 MHz, por lo que cada instrucción se

ejecuta en un microsegundo. El cristal debe ir acompañado de dos condensadores, para evitar interferencias y el modo de conexión es el siguiente.

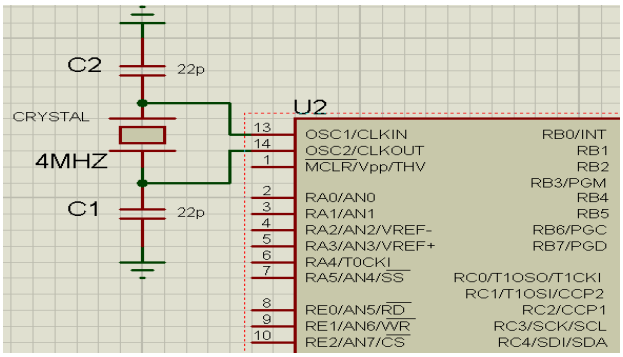
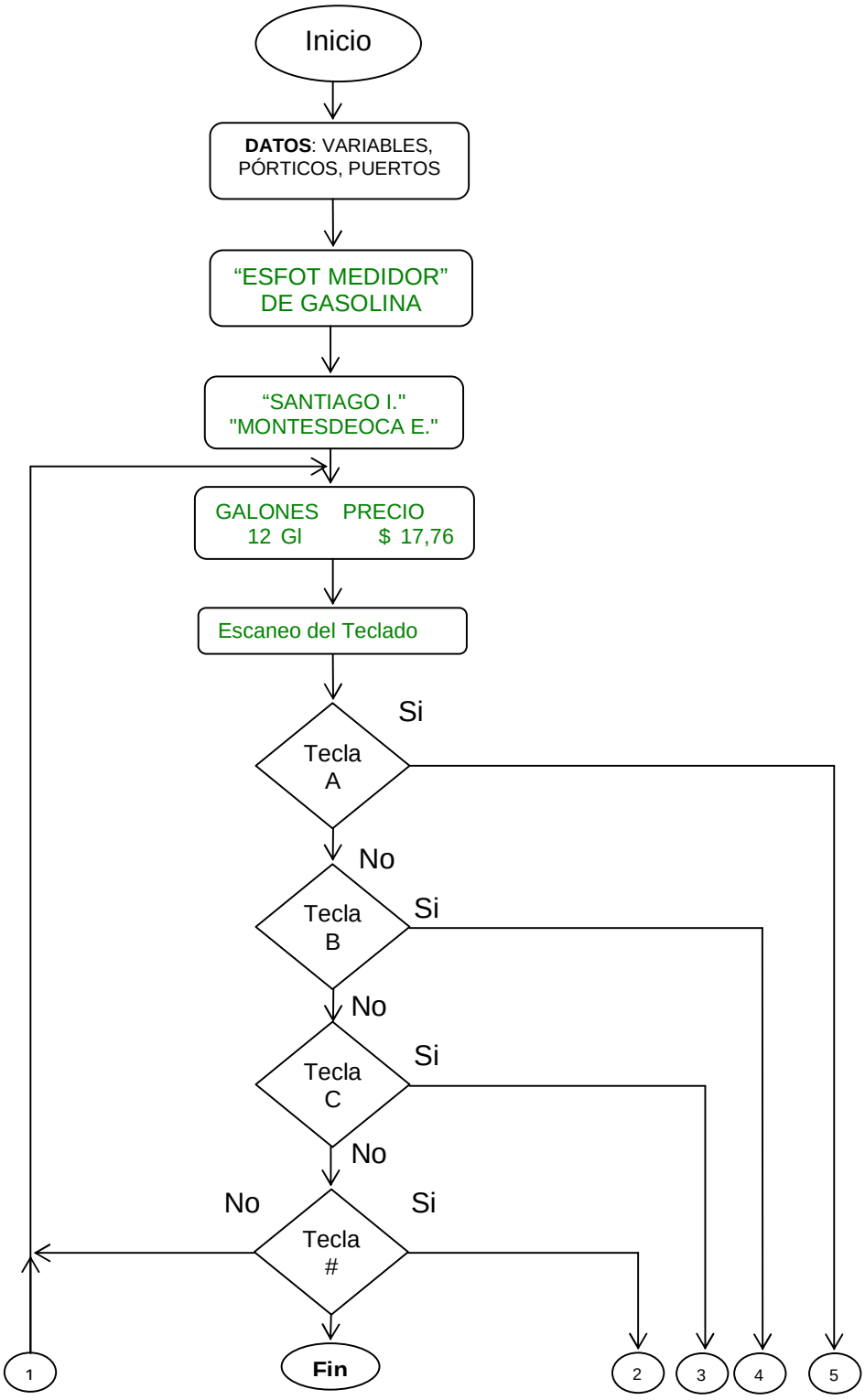


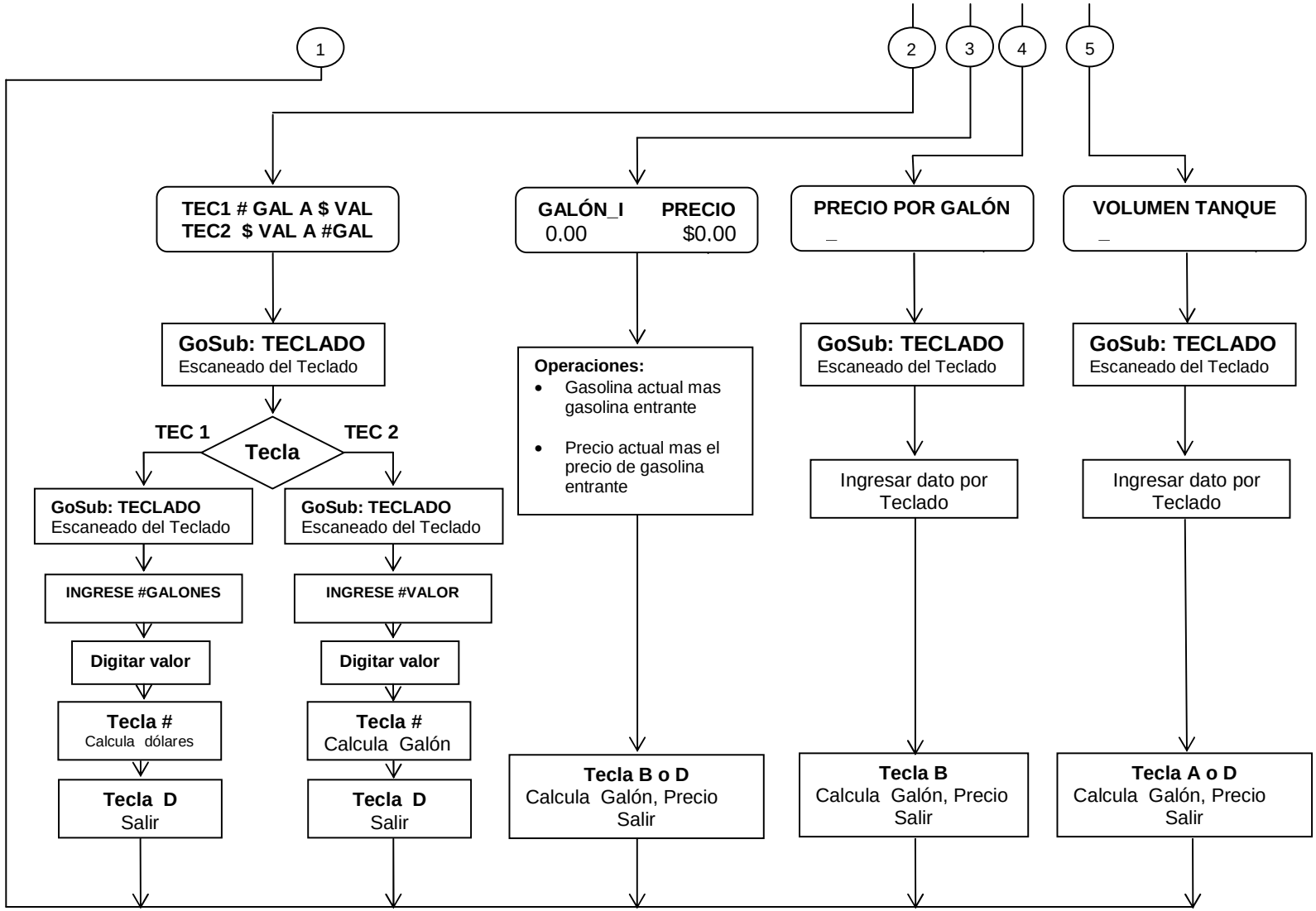
Figura 2.15 Conexiones del Oscilado Externo al PIC 16F877A.

Figura 2.16 Conexiones del Oscilado Externo al PIC 16F877A.

2.4 PROGRAMACION DEL PIC 16F877A.

2.4.1 DIAGRAMA DE FLUJO




1
2
3
4
5
TEC1 # GAL A $ VAL
TEC2 $ VAL A #GAL
GALÓN_I PRECIO
0.00 $0.00
PRECIO POR GALÓN
VOLUMEN TANQUE
GoSub: TECLADO
Escaneado del Teclado
Operaciones:
Gasolina actual mas gasolina entrante
Precio actual mas el precio de gasolina entrante
GoSub: TECLADO
Escaneado del Teclado
GoSub: TECLADO
Escaneado del Teclado
TEC 1
Tecla
TEC 2
Ingresar dato por Teclado
Ingresar dato por Teclado
GoSub: TECLADO
Escaneado del Teclado
GoSub: TECLADO
Escaneado del Teclado
INGRESE #GALONES
INGRESE #VALOR
Digitar valor
Digitar valor
Tecla #
Calcula dólares
Tecla #
Calcula Galón
Tecla D
Salir
Tecla D
Salir
Tecla B o D
Calcula Galón, Precio
Salir
Tecla B
Calcula Galón, Precio
Salir
Tecla A o D
Calcula Galón, Precio
Salir

2.4.2 EXPLICACIÓN DE LA PROGAMACIÓN DEL PIC16F877A.

El PIC16F877A constituye el dispositivo más importante en el circuito del medidor de gasolina digital. Debido a que este es el programado para las funciones específicas requeridas, en este caso la cantidad de gasolina, precio del la gasolina que entra al tanque del automóvil, convertidor de unidades. El programa realizado se detalla en el anexo 3.

2.4.2.1 Configuración de trabajo del PIC16F877A.

Primero se debe configurar las características internas del microcontrolador, esto se refiere a la definición de la frecuencia de trabajo, que viene dada por el empleo de un oscilador que puede ser interno ó externo. Se utilizó el oscilador interno de 4Mhz. que dispone el microcontrolador con el fin de aprovechar sus recursos, y optimizar el espacio físico en la tarjeta electrónica.

Configuración de pórticos de entrada o salida. El pórtico B se configuró como salida para el manejo de las conexiones del LCD exceptuando el terminal RB0 que se maneja como entrada del contador de pulsos. El pórtico A configurado como entrada digital en todos sus terminales para el manejo del código de detección de dirección específicamente en RA0, RA1, RA2, RA3, RA4, y para el control de visualización de las, unidades en los terminales RA7, RA6, RA5 respectivamente.

2.4.2.2 Declaración de variables

Las variables son palabras referidas necesarias para manejar información de entrada o salida de datos la mayoría se declaran de acuerdo a la función que desempeñan. Para la presentación de mensajes iníciales se empleo las variables de 8 bits *I*, En la obtención de datos digitales, para la medición de la cantidad de gasolina que ingresa al tanque de combustible de un automóvil intervienen las siguientes variables:

Para realizar el conteo global de pulsos en 1s se utiliza la variable *DATO* de 16 bits. Mediante las variables I*, DATO(I*) de 8 bits se realizó los conteos parciales de pulsos de medio segundo, a su vez en *DATO(I)* se almacena la suma total del número de pulsos por segundo, dato necesario para el proceso de cálculo de velocidad cuyo resultado es almacenado en las variables de 16 bits *entero, decimal* que permiten la visualización en el LCD.

2.4.2.3 Inicialización de variables

Este proceso empieza haciendo que los terminales del pórtico A sean entradas digitales, encera el contador global de pulsos, el contador de pulsos por segundo (variables *DATO, (I)*) y el pórtico B.

Se acondiciona una señal de reloj de 1KHz en el terminal RB1 del pórtico B que se encarga de indicar que arrancó el PIC. Y se visualizan los mensajes de presentación en el LCD "ESFOT MEDIDOR." en la primera línea, y " DE GASOLINA " en la segunda línea.

Figura 2.17 Nombre del proyecto.

Posteriormente se visualizan el nombre del autor del proyecto, "SANTIAGO I ", " MONTESDEOCA E " esto se puede apreciar en la figura 2.17

Figura 2.18 Autor de proyecto.

Seguidamente se visualiza en el LCD las mediciones con que se encuentra el tanque de combustible mediante las variables *GASOLINA* Y *PRECIO* en unidades, gasolina en Galones *(GL)* Precio en Dólares *($)* que se encuentran almacenados en las variables expresados en entero, decimal, respectivamente.

2.4.2.4 Desarrollo del programa.

En esta parte del proyecto se detalla muy detenidamente como está constituido el programa que se elaboro para el presente proyecto.

En esta parte se define los parámetros del que se va a utilizar, que vienen incluido en el micro controlador. Device indica el tipo de micro-controlador que se va a utilizar, XTAL define la frecuencia a la que el cristal trabajara, para este caso la frecuencia se eligió de 4 MHZ, TRISB activación del puerto RB como salida de datos por medio de 1 lógico, TRISD activación del puerto RD como entrada de datos por medio de 0 lógico, ADCON1 se utilizo para la transformación de la señal análoga a digital, TRISA.3 activación de pin 3 como entrada de datos por donde ingresa la señal a digitalizarse.

Device 16F877A
XTAL=4
TRISB=%11111111
TRISD=0
ADCON1=%01000100
TRISA.3=0

Ajustar algunos aspectos algunos parámetros que tiene el PIC código, es decir Cristal, frecuencia, LCD, puerto y pins. Por recomendaciones del fabricante

Declare **LCD_DTPIN PORTD.**4
Declare **LCD_ENPIN PORTA.2**
Declare **LCD_RSPIN PORTA.5**
Declare **LCD_LINES 2**
Declare **ADIN_RES10**
Declare **ADIN_TAD2**

Declare **ADIN_STIME 100**

Declaraciones de todas las variables que se utilizan en la programación del presente proyecto. Variables para números enteros

Dim **TANQUE** As Byte
Dim **DATO** As Byte
Dim **I** As Byte
Dim **TECLA** As Byte
Dim **N** As Byte
Dim **PUNTO** As Byte
Dim **PUNTOS** As Byte
Dim **PRECIO_G4[8]** As Byte

Variables para números con punto decimal

Dim **ANALOGICO** As Word
Dim **GASOLINA** As Float
Dim **PRECIO_G** As Float
Dim **PRECIO** As Float
Dim **PRECIO_G1** As Float
Dim **GALON_CALCULADO** As Float
Dim **GALON_ACTUAL** As Float
Dim **GALON_INGRESADO** As Float

2.4.2.4.1 Variables que se utiliza para el escaneo del teclado

Symbol X1 = PORTD.3
Symbol X2 = PORTD.2
Symbol X3 = PORTD.1
Symbol X4 = PORTD.0
Symbol Y1 = PORTB.3
Symbol Y2 = PORTB.2
Symbol Y3 = PORTB.1
Symbol Y4 = PORTB.0

En esta parte del programa es la presentación inicial del proyecto mediante el lazo For da un conteo de 16 caracteres que se muestra en la pantalla con un tiempo de100 mili segundos que aparase de izquierda a derecha del LCD

```
Low led
DelayMS 1000
For I=0 To 15
DATO=LookUpL I , [ "  ESFOT  MEDIDOR  " ]
Print At 1 , I ,  DATO
DelayMS 100
Next I
For I=0 To 15
LookUpL I,[ "  DE  GASOLINA  " ],DATO
Print At 2 , I , DATO
DelayMS 100
Next I
DelayMS 1000
Cls
For I=0 To 15
LookUpL I,["  SANTIAGO  I.  "],DATO
Print At 1 , I , DATO
DelayMS 100
Next I
For I=0 To 15
LookupL I,[" MONTESDEOCA E. "],DATO
Print At 2 , I , DATO
DelayMS 100
Next
DelayMS 100
Cls
```

Presenta el cálculo que realiza el programa para determinar la cantidad de gasolina que está dentro del tanque de combustible, mediante la variable

ANALOGICO= ADIn 0 que efectúa la conversión de análogo a digital de la señal que emite el sensor de nivel, la cantidad de gasolina calcula mediante el valor analógico multiplicado por la capacidad del tanque ingresado dividido todo esto para un valor 65455, el resultado esta expresado en galones.

Si la variable gasolina es menor a 1,5 galones en el LCD muestra un mensaje de "COMBUSTIBLE POR TERMINAR" lo cual encenderá un led de color Azul. Ver figura 2,19 Si la variable gasolina son mayores a 1,5 galones en el LCD muestra la cantidad de gasolina que está dentro del tanque de combustible con su respectivo precio expresados en dólares, lo que estará en repetición constante hasta que el nivel de gasolina baje o se seleccione otra opción mediante el teclado.

Figura 2.19 Cuando el tanque esta vacio y quedarce sin gasolina.

```
inicio:
GoSub TECLADO
ANALOGICO= ADIn 0
GASOLINA=ANALOGICO*TANQUE/65455
If GASOLINA<1.5 Then
  Print At 1,1 , "COMBUSTIBLE POR "
  Print At 2,1 , "   TERMINAR     "
  High led
  normal:
  ANALOGICO= ADIn 0
  GASOLINA=ANALOGICO*TANQUE/65455
  If GASOLINA< 2 Then GoTo normal
  Print At 2,1 , "                "
End If
```

Realiza el cálculo del precio de la cantidad de gasolina mediante la variable GASOLINA y es visualizado en el LCD en la primera fila se ubica los galones y precio en forma de texto, mientras que en la segunda fila estará los valores calculados de galones y precio.

```
PRECIO=GASOLINA*PRECIO_G
Print At 1,1, "GALONES "
Print At 1,10, "PRECIO  "
Print At 2,1, Dec GASOLINA,"Gl  "
Print At 2,10, "$",Dec PRECIO ,"  "
```

En esta parte del programa realiza un escaneo de la tecla **A** si esta es pulsada para seleccionar e ingresar el volumen del tanque mediante el teclado, la misma que llama a la variable VOLUMEN_TANQUE este cálculo se expilara más adelante y tendrá un tiempo de espera de 1000 milisegundos, que será visualizado en el LCD.

```
If TECLA=10 Then
  Print At 1,1 ," VOLUMEN TANQUE "
  Print At 2,1 ,"              "
  DelayMS 1000
  GoSub VOLUMEN_TANQUE
End If
```

Realiza un escaneo de la tecla **B** si esta es pulsada para seleccionar e ingresar el PRECIO POR GALÓN mediante el teclado, la misma que llama a la variable PRECIO_GALON este cálculo se expilara más adelante y tendrá un tiempo de espera de 1000 milisegundos, que será visualizado en el LCD

```
If TECLA=11 Then
  Print At 1,1 ,"PRECIO POR GALON"
  Print At 2,1 ,"              "
  DelayMS 1000
  GoSub PRECIO_GALON
End If
```

Programa que permite el escaneo del teclado las filas como columnas por un tiempo de 200 milisegundos para poner un funcionamiento de las teclas para ser activado en cualquier momento en que se desee.

TECLADO:

TECLA=16

Low X1

If Y1=0 Then TECLA= 1: High X1: DelayMS 200: Return

If Y2=0 Then TECLA= 2: High X1: DelayMS 200: Return

If Y3=0 Then TECLA= 3: High X1: DelayMS 200: Return

If Y4=0 Then TECLA=10: High X1: DelayMS 200: Return

High X1

Low X2

If Y1=0 Then TECLA= 4: High X2: DelayMS 200: Return

If Y2=0 Then TECLA= 5: High X2: DelayMS 200: Return

If Y3=0 Then TECLA= 6: High X2: DelayMS 200: Return

If Y4=0 Then TECLA=11: High X2: DelayMS 200: Return

High X2

Low X3

If Y1=0 Then TECLA= 7: High X3: DelayMS 200: Return

If Y2=0 Then TECLA= 8: High X3: DelayMS 200: Return

If Y3=0 Then TECLA= 9: High X3: DelayMS 200: Return

If Y4=0 Then TECLA=12: High X3: DelayMS 200: Return

High X3

Low X4

If Y1=0 Then TECLA=15: High X4: DelayMS 200: Return

If Y2=0 Then TECLA= 0: High X4: DelayMS 200: Return

If Y3=0 Then TECLA=14: High X4: DelayMS 200: Return

If Y4=0 Then TECLA=13: High X4: DelayMS 200: Return

High X4

Return

2.4.2.4.2 Volumen_Tanque:

En esta parte del programa permite seleccionar el volumen del tanque que se requiere almacenar en el tanque de combustible, al pulsar la tecla **A**, en el LCD mostrara en la primera fila volumen del tanque, en la segunda fila se visualizar la capacidad que se elige mediante la selección del teclado en cantidades enteras que puede almacenar el tanque de combustible.

Figura 2.20 Capaciada del Tanque.

Figura 2.21 Imgreso de valor para seleccioner la Capaciada del Tanque.

```
TECLA=16
While TECLA<> 10
 GoSub TECLADO
  If TECLA=13 Then
  TECLA=10
  End If
  If TECLA<10 Then
  If N=0 Then
  TANQUE=TECLA
   N=1
   Else
  TANQUE=TANQUE*10+TECLA
  EWrite 1 , [ TANQUE ]
```

```
  If N=1 Then N=0
  End If
  Print At 2,1,Dec TANQUE," Gl  "
  End If
  Wend
  N=0
TECLA=16
DelayMS 200
Return
```

2.4.2.4.3 Calculadora:

Parte del programa que permite realizar la seleccionar mediante la tecla # las opciones cantidad de galones a cantidad de dinero y cantidad de dinero a cantidad de galones.

Figura 2.22 Para seleccionar calculadora.

```
Print At 1,1,"TEC1 #GAL A $VAL"
Print At 2,1,"TEC2 $VAL A #GAL"
TECLA=16
DelayMS 200
While TECLA<>14
GoSub TECLADO
If TECLA=13 Then TECLA=14
If TECLA=1 Then GoSub GALVAL
If TECLA=2 Then GoSub VALGAL
If y=1 Then Break
```

Wend
y=0
Return

2.4.2.4.3.1 Galval

Programa que permite realizar la conversión de la cantidad de galones a cantidad de dinero, mediante la selección de una opción, mediante la tecla 1 del teclado donde permite realizar el cálculo.

Al seleccionar la tecla **1** opción 1 en el LCD muestra ingrese la cantidad de galones que se digitara mediante el teclado, para ver el resultado de se presiona la tecla # donde mostrará el resultado de la cantidad de dinero que corresponderá a la cantidad de galones que se ingresa, si no se quiere ver el resultado se puede presionar a la tecla **D** que saldrá del este programa y mostrara nivel de combustible que estará censando.

Figura 2.23 para ingreso de galones para el calculo de dinero.

Figura 2.24 Ingreso de la cantidad galones a transformar a dinero.

Figura 2.25 Calculo de dinero.

```
Print At 1,1,"INGRESE #GALONES"
Print At 2,1,"               "
TECLA=16
N=1
While TECLA<> 14
 If TECLA=13 Then TECLA=14
 GoSub TECLADO
 If TECLA=13 Then
 TECLA=11
 End If
 If TECLA=15 Then  'DETECTO PUNTO DECIMAL
 PUNTOS=1
 N=N+1
 PRECIO_G4[N]="."         'GUARDO LA VARIBLE DECIMAL
 Print At 2,N,PRECIO_G4[N] 'VISUALIZO EL PUNTO DECIMAL EN EL LCD
 End If
 If TECLA<10 Then
 If PUNTOS=1 Then PUNTO=PUNTO+1
 If N=1 Then
 N=2
 GALON_CALCULADO=TECLA
 PRECIO_G4[1]=TECLA
 Print At 2,1," ",Dec PRECIO_G4[1]
 Else
 N=N+1
```

```
PRECIO_G4[N]=TECLA
Print At 2,N,Dec PRECIO_G4[N]
GALON_CALCULADO=GALON_CALCULADO*10+TECLA
If N=6 Then
N=1
PUNTO=0
PUNTOS=0
PRECIO_G=0
End If
End If
End If
GoSub TECLADO
If TECLA=13 Then
y=1
Break
End If
Wend
If y=0 Then
PRECIO_G1=GALON_CALCULADO
If PUNTO=1 Then PRECIO_G1=GALON_CALCULADO/10'GALONES
If PUNTO=2 Then PRECIO_G1=GALON_CALCULADO/100
GALON_CALCULADO=PRECIO_G1
PRECIO=GALON_CALCULADO*PRECIO_G
Print At 2,9," $",Dec PRECIO,"  "
DelayMS 1000
PUNTO=0
PUNTOS=0
N=0
TECLA=16
End If
DelayMS 200
```

Return

2.4.2.4.3.2 Valgal.

Programa que permite realizar la conversión de la cantidad de dinero a cantidad de gasolina, mediante la selección de una opción, mediante la tecla 2 del teclado donde permite realizar el cálculo.

Al seleccionar la tecla **2** opción 2 en el LCD muestra ingrese la cantidad de dinero, mediante el teclado, para ver el resultado se presiona la tecla # donde mostrará el resultado de la cantidad de galones este valor corresponderá a la cantidad de dinero que se ingresado, si no se quiere ver el resultado puede ser presionara la tecla **D** que saldrá del este programa y mostrara nivel de combustible que estará censando.

Figura 2.26 Ingreso de dinero para el calculo de galones

Figura 2.27 Ingreso de la cantidad dinero a transformar a galones

Figura 2.28 Resultado de Transformacion

Print At 1,1," INGRESE #VALOR "

```
Print At 2,1,"               "
TECLA=16
N=1
While TECLA<> 14
If TECLA=13 Then TECLA=14
GoSub TECLADO
If TECLA=13 Then
TECLA=11
End If
If TECLA=15 Then
PUNTOS=1
N=N+1
PRECIO_G4[N]="."
Print At 2,N,PRECIO_G4[N]
End If
If TECLA<10 Then
If PUNTOS=1 Then PUNTO=PUNTO+1
If N=1 Then
N=2
GALON_CALCULADO=TECLA
PRECIO_G4[1]=TECLA
Print At 2,1," ",Dec PRECIO_G4[1]
Else
N=N+1
PRECIO_G4[N]=TECLA
Print At 2,N,Dec PRECIO_G4[N]
GALON_CALCULADO=GALON_CALCULADO*10+TECLA
If N=6 Then
N=1
PUNTO=0
PUNTOS=0
```

```
PRECIO_G=0
End If
End If
End If
GoSub TECLADO
If TECLA=13 Then
y=1
Break
End If
Wend
If y=0 Then
PRECIO_G1=GALON_CALCULADO
If PUNTO=1 Then PRECIO_G1=GALON_CALCULADO/10 'GALONES PRECIO
If PUNTO=2 Then PRECIO_G1=GALON_CALCULADO/100
GALON_CALCULADO=PRECIO_G1
PRECIO=GALON_CALCULADO/PRECIO_G
Print At 2,8,Dec PRECIO,"Gl  "
DelayMS 1000
PUNTO=0
PUNTOS=0
N=0
TECLA=16
End If
DelayMS 200
Return
```

2.4.2.4.3.3 Precio_Galon:

Programa donde mediante el teclado se puede ingresar el precio del combustible deseable, al seleccionar la tecla **B** en el teclado matricial en el LCD muestra ingrese el precio del combustible, una vez seleccionado el precio se presiona la

tecla para **B** para salir del programa y el valor ser guardado el memoria EEPRON de microcontrolador para que cuando el automotor se apague este valor no se borre.

Figura 2.28 Para el Ingreso del Precio.

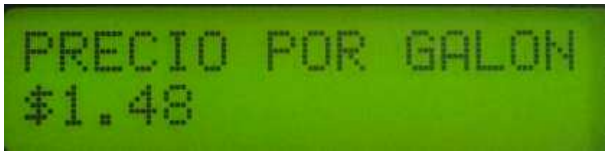


Figura 2.29 Valor del Precio a Guardar en el Micro.

```
TECLA=16
 N=1
 While TECLA<> 11
 GoSub TECLADO
 If TECLA=13 Then
 TECLA=11
  End If
 If TECLA=15 Then
 PUNTOS=1
 N=N+1
 PRECIO_G4[N]="."
 Print At 2,N,PRECIO_G4[N]
 End If
 If TECLA<10 Then
 If PUNTOS=1 Then PUNTO=PUNTO+1
 If N=1 Then
 N=2
 PRECIO_G=TECLA
 PRECIO_G4[1]=TECLA
```

```
Print At 2,1,"$",Dec PRECIO_G4[1]
Else
N=N+1
PRECIO_G4[N]=TECLA
PRECIO_G=PRECIO_G*10+TECLA
If N=6 Then
N=1
PUNTO=0
PUNTOS=0
PRECIO_G=0
End If
End If
End If
Wend
If PUNTO=1 Then PRECIO_G1=PRECIO_G/10
If PUNTO=2 Then PRECIO_G1=PRECIO_G/100
PRECIO_G=PRECIO_G1
EWrite 3 , [ PRECIO_G ]
DelayMS 1000
PUNTO=0
PUNTOS=0
N=0
TECLA=16
DelayMS 200
Return
End
```

2.4.2.4.4 Galon_Actual.

Realiza un escaneo de la tecla **C** para visualizar la cantidad de gasolina en galones que está ingresando mediante una resta entre los valores GASOLINA menos GALON_ACTUAL y el precio respectivamente.

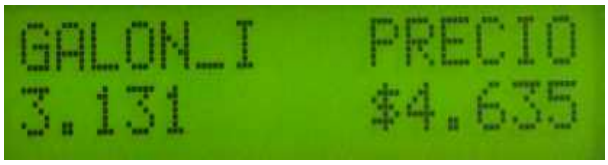


Figura 2.30 de la cantidad de galones que se está llenado en el tanque y el precio.

GALON_INGRESADO=GASOLINA-GALON_ACTUAL
Print At 2,1,Dec GALON_INGRESADO," "
Print At 2,11,"$",Dec PRECIO," "
Wend
End If.

2.5 MONTAJE DEL CIRCUITO ELECTRONICO EN PROTOBOARD.

Como se observa en la figura 2.31 se realiza el montaje previo del circuito electrónico del medidor de gasolina digital con todos y cada uno de sus componentes y dispositivos electrónicos. Con el fin de revisar el correcto funcionamiento y también su tiempo de respuesta.

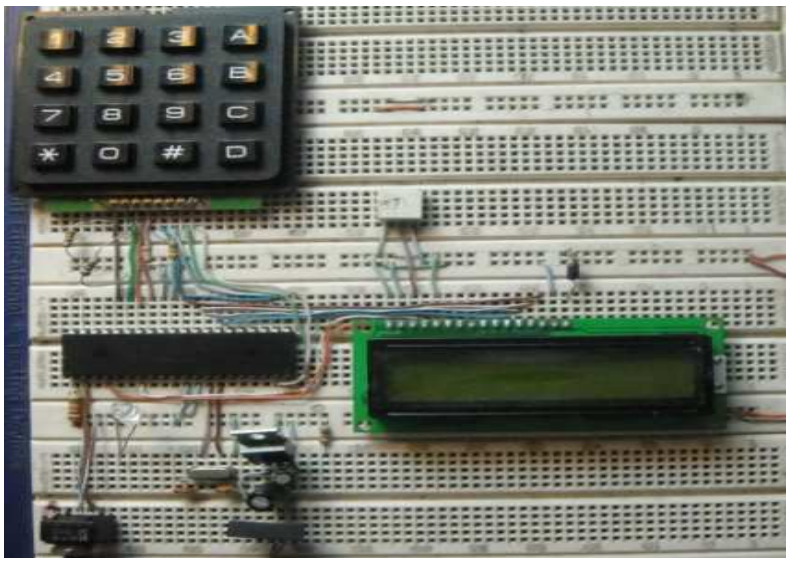


Figura 2.31 Montaje de prueba en Protoboard.

Se comprobó lo siguiente:

- ✓ Regulación de voltaje de +12V a +5V en el terminal de salida del CI 78L05
- ✓ Señales de arranque y trabajo del PIC mediante los diodos LEDS 1-2 respectivamente
- ✓ Respuesta de voltaje al variar la resistencia del sensor que envía la señal hacia el microcontrolador.
- ✓ Ingreso de datos al microcontrolador 16F877A
- ✓ Respuesta de teclado para en el momento de seleccionar cada una de sus teclas y las opciones dadas en el programa, estén de acuerdo a los requerimientos planteado.
- ✓ Visualización de los datos respectivos en el LCD.

2.5.1 CONSTRUCCIÓN Y ENSAMBLAJE DEL EQUIIPO.

En esta etapa se describe los pasos que se debió seguir para desarrollar la construcción del equipo entre los cuales están los siguientes:

- o Diseño y fabricación de las tarjetas electrónicas en circuito impreso
- o Montaje de componentes electrónicos
- o Ensamblaje del medidor de gasolina digital en la carcasa.

2.5.2 DISEÑO Y FABRICACIÓN DE LAS TARJETAS ELECTRÓNICAS EN CIRCUITO IMPRESO.

2.5.2.1 Circuito impreso.

Es un circuito eléctrico que se fabrica depositando material conductor sobre la superficie de una base aislante, denominada placa de circuito impreso (PCB). En este tipo de circuitos, el cableado usado en circuitos tradicionales es sustituido por una red de finas líneas conductoras, impresas y unidas sobre el PCB. Pueden introducirse dentro del circuito otros componentes electrónicos, como resistencias, condensadores, zócalos, interruptores, etc., previamente en orificios taladrados.

Para la elaboración de circuitos impresos a menudo se utilizan programas de diseño electrónico automatizado, para distribuir e interconectar los componentes. Estos programas almacenan información relacionada con el diseño circuital, facilita la edición de sus pistas del circuito, y puede también automatizar tareas repetitivas.

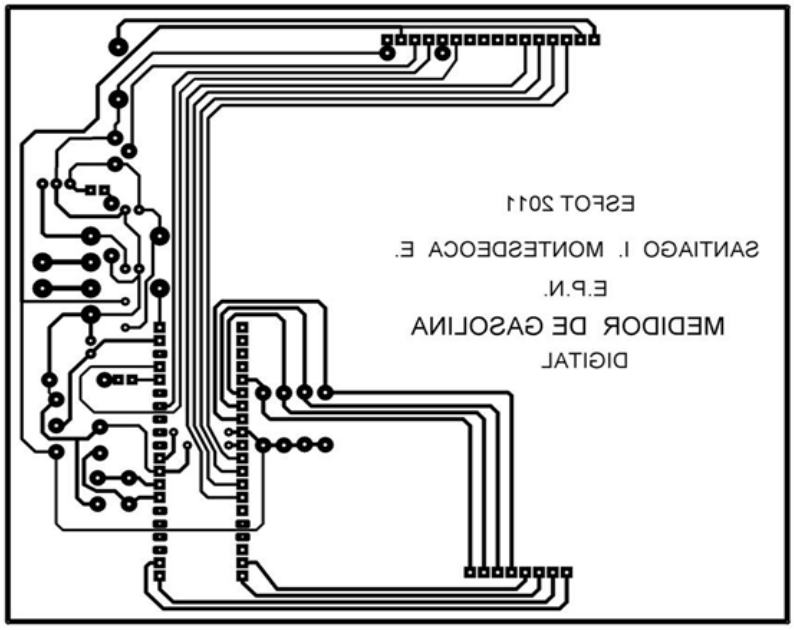


Figura 2.32 Pistas del Circuto

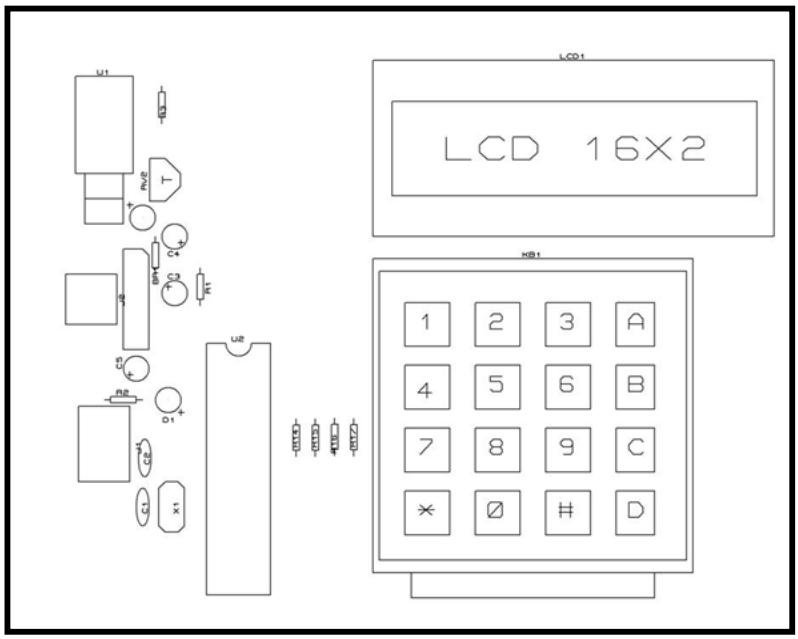


Figura 2.33 Colocación de cada uno del los elementos electrónicos.

En la figura 2.34 se observa el circuito impreso del medidor de combustible digital con las perforaciones, pistas correspondientes para la ubicación de los diversos elementos y dispositivos electrónicos, que posteriormente deben ser soldados.

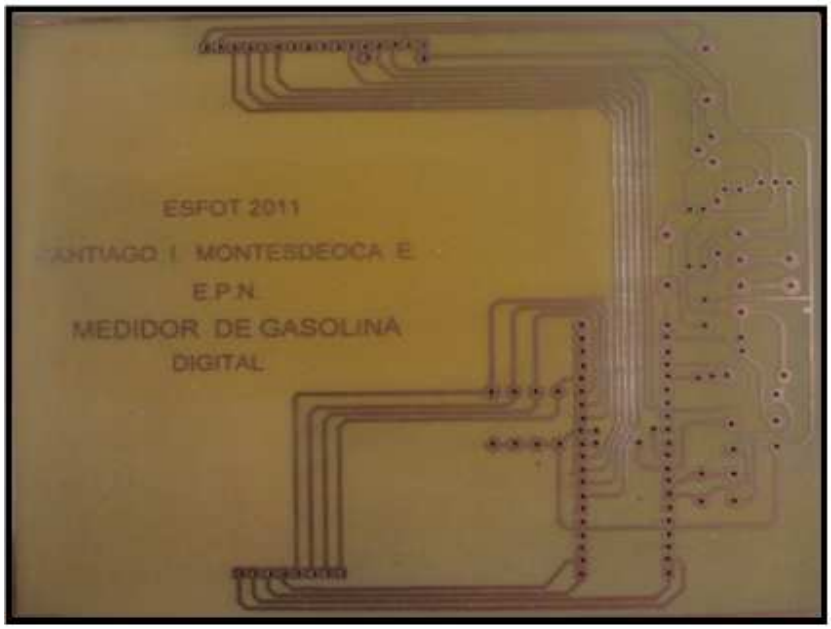


Figura 2.34 Tarjeta electrónica

2.5.2.2 Montaje de Elementos.

Una vez que se dispone del circuito impreso, el siguiente paso es realizar el ensamblaje de la tarjeta electrónica. Esto se refiere a la ubicación, fijación, soldadura de los componentes electrónicos, y de más ajustes en la placa. Para lo cual es indispensable disponer de un orden de los componentes electrónicos a instalar en la tarjeta. A continuación u orden a seguir para facilitar su instalación.

Resistencias:

- ✓ R3, R4, R14, R15, R16, R17 330Ω
- ✓ R2 de 1KΩ

Condensadores:

- ✓ C1, C2 de 22 *pf* a 10V
- ✓ C3, C4, C5, 10 uf de 10.

Diodos Leds:

- ✓ LED1,LED2

Potenciómetro:

- ✓ P1 de 10KΩ

Cristal:

- ✓ Mx 4Mhz

Circuitos Integrados:

- ✓ Regulador de voltaje 78L05
- ✓ PIC16F877A
- ✓ Modulo LCD Luz verde 2x16
- ✓ Teclado matricial 4x4

Conectores:

- ✓ GP16 beige x 4
- ✓ GP16 beige x 5
- ✓ GP16 beige x 8
- ✓ GP16 beige x 16

Puente de Diodos:

- ✓ RS207

Por lo general se empieza instalando componentes básicos como: resistores, condensadores, cristal (R2, R3, R4, R14, R15, R16, R17, C1, C2, C3, C4, Mx) además de la conexión de puentes de ser necesarios. Después se instala el zócalo correspondiente para el circuito integrado (PIC16F877A), de igual manera proceda con los conectores (GP16 beige x4, x5, x8, x16). Destinados a diversas conexiones mediante cables a elementos que no pueden ser fijados directamente en la tarjeta, potenciómetro de contraste P1, bus de datos para LCD 2x16. Finalmente se instala el regulador de voltaje y el conector para alimentación por adaptador. En la figura 2.34 finalmente se aprecia en detalle la ubicación de los componentes electrónicos, así como también las conexiones de los mismos mediante las pistas.

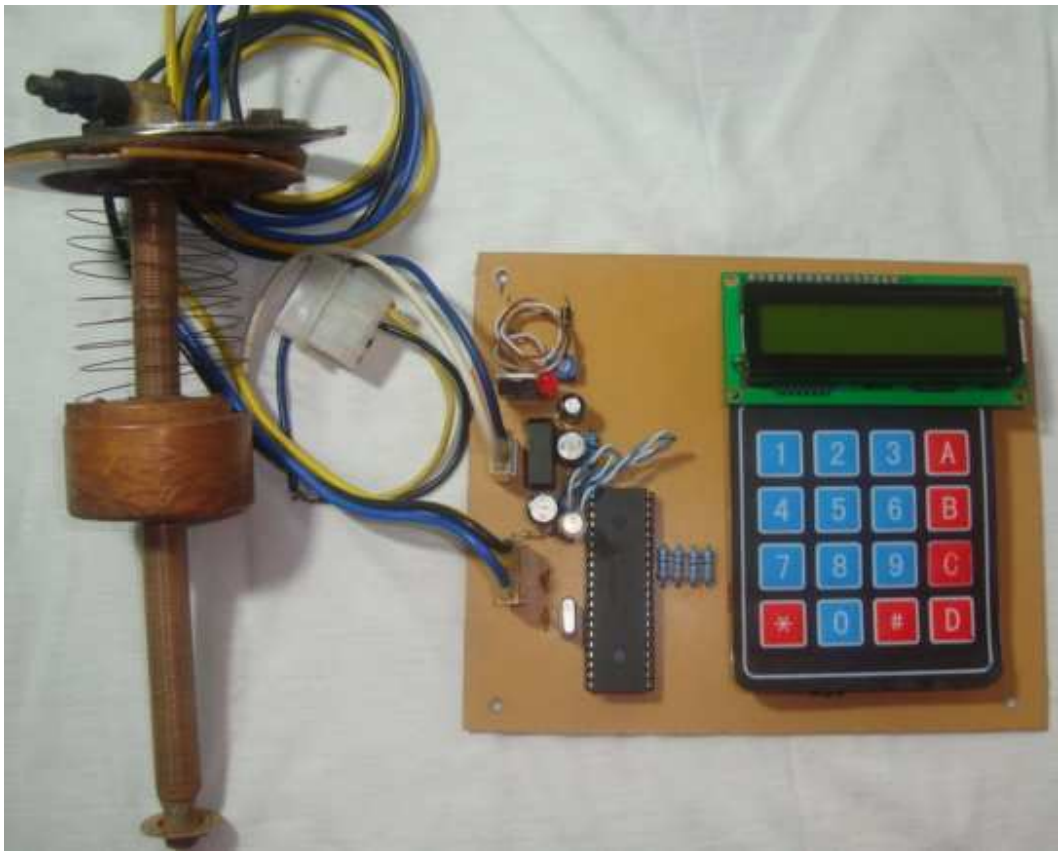


Figura 2.35 Placa electrónica con todos sus elementos.

2.6 GUÍA DE USUARIO.

El medidor de gasolina digital es un equipo muy fácil de utilizar, este puede ser operado por personas con conocimiento de electrónica básica.

A continuación se describe un breve resumen de guía de usuario:

- ON/OFF: botón de encendido ó apagado del equipo medidor.
- Unidades – en galones.
- Unidades – en Dólares
- Potenciómetro: sirve para regular el contraste de la pantalla del LCD de acuerdo al criterio del operador

- Posee 2 leds de visualización los mismos que indican.
    - Encendido del equipo Led 1 (rojo).
    - Cuando el combustible está por terminar Led 2 (azul).
- El equipo de medición se conecta al flotado del tanque de combustible que posee un automóvil mediante tres cables, el mismo que conduce el voltaje al micro-controlador para ser convertido de análogo a digital y ser visualizado mediante un LCD la cantidad de galones que está almacenado y el precio que corresponde los galones que está dentro del tanque de combustible, mediante el teclado se puede seleccionar la cantidad de galones que se puede almacenar en el tanque de combustible, grabar el precio actual de la gasolina, visualizar en el LCD el momento en que la gasolina está por terminarse, realizar el cálculo de la cantidad de galones que se puede llenar con una determinada suma de dinero, el cálculo de dinero si se ingresa un determinado número de galones, observar la cantidad de galones que ingreso al tanque de combustible cuando se está llenando y finalmente visualizar el led azul su encendido y apagado cuando este por terminarse el combustible del auto motor.

2.7 ANÁLISIS TÉCNICO-ECONÓMICO.

2.7.1 ANÁLISIS TÉCNICO.

Según las características del equipo, y de acuerdo a los requerimiento de la capacidad de almacenamiento del tanque de combustible del automóvil lo que lleva a determinar la cantidad exacta de dinero con la que se puede llenar el tanque de combustible. El equipo constituye una herramienta muy eficaz que permita medir y cuantificar los diversos parámetros con exactitud siendo esta cantidad de gasolina que se encuentra dentro del tanque de combustible, el valor en dinero con lo que se llena el tanque de combustible del auto motor.

De acuerdo a las variaciones señaladas, con el equipo terminado se puede realizar mediciones exactas para las distintas aplicaciones como son las siguientes:

- Una mejora en la productividad y rendimiento del automotor al saber con exactitud la cantidad de combustible que se encuentra almacenado en el tanque del automotor independientemente de la actividad a desarrollar.
- Prevenir de posibles estafas que se comente al momento del llenado del combustible en el automotor en la estación de servicios.
- Notificar a las entidades de control si los equipos que suministran combustibles están fallando o si se encuentran adulterados.
- Evaluación del servicio que dan las estaciones despachadoras de combustible hacia los consumidores de este producto.
- Determinar si lo que le está cobrando al usuario de una estación de servicios es correcta y comparar lo marcado en el surtidor con lo que está dentro del tanque de combustible del automotor.
- Evitar que el automotor se quede sin combustible cuando se está en marcha en las carreteras.

2.7.2 ANÁLISIS ECONÓMICO.

En la construcción del equipo medidor de gasolina digital, se empleó dispositivos conocidos y accesibles en el medio, relativamente económicos para el usuario que para quien lo desea adquirir.

En la tabla 2.1 se puede observar, los correspondientes costos de los dispositivos y elementos empleados en la construcción del Medidor de Gasolina Digital, así como también el valor de la mano de obra y costo total.

Tabla 2.1 *Costos parciales y totales del Medidor de Gasolina Digital.*

Item	**Cant**	**Descripción**	**Total USD**
1	1	Diodo	0,30
2	1	Potenciómetro 10K	0,50
3	5	Resistencias (1/4 watio)	2,70
4	2	Condensadores de 10µf electrolíticos	0,30
5	2	Condensadores de 22 pf cerámicos	0,30
6	1	Condensadores de 1µf electrolíticos	0,40
7	2	Leds	0,60
8	1	Sócalo de CI de 40 pines	2,00
9	1	Cristal	0,60
10	1	Regulador de voltaje LM7805CT 5v Positivo	1,60
11	1	Microcontrolador PIC 16F877A	10,00
12	1	Modulo LCD Luz amarilla 2x16	8,00
13	1	Teclado matricial 4x4	8,00
14	1	Placa	6,00
15	3	Cables 50 cm	1,50
16	2	Acido cúprico	1,00
17	1	Papel fotográfico	1,00
18	1	Accesorios e implementos varios	30,00
19		Subtotal Materiales:	74,50
20	1	Mano de obra	100,00
		TOTAL:	174,50

De acuerdo a los costos totales y los beneficios que proporciona el Medidor de Gasolina Digital se concluye que es accesible para el personal técnico y el público en general que lo necesite.

CAPITULO 3

3.1 Conclusiones.

- Mediante este proyecto se logró construir un sistema que permitirá medir la cantidad de gasolina que ingresa al tanque de un auto motor de una manera digital y de forma económica y sencilla con ayuda de un microcontrolador.

- Los objetivos específicos que se cumplieron en este proyecto son los siguientes. Determinar cantidad de gasolina que está dentro del tanque de combustible del automóvil, el precio en dólares de cada galón, con la ayuda de un elemento mecánico (boya de combustible). Para posteriormente conseguir su representación numérica

- Toda medida realizada por un aparato mecánico se la puede digitalizar mediante un dispositivo electrónico adecuado, adaptado a este y valiéndose siempre de un circuito electrónico.

- Es necesario basarse en las características técnicas de los elementos usados, pues existen recomendaciones que se tienen que tomar en cuenta al momento de implementar el circuito electrónico para la conexión del mismo.

- Antes de escoger un microcontrolador, es mejor saber cuál es el uso que lo vamos a dar, y su alcance de memoria interna para la programación, de lo

contrario estaríamos usando un microcontrolador inadecuado, para este caso especifico el PIC16F877A, sirvió para realizar las operaciones de suma y multiplicación y guardar datos medidos.

- Se consiguió que el microcontrolador envíe la señal digital correspondiente de la cantidad de gasolina que está dentro del tanque de combustible, hacia el modulo LCD, para ser visualizar dicho valor de forma numérica de dichos valores y otras aplicaciones.

- Luego de haber realizado las diferentes pruebas al proyecto, se concluye que éste cumple con el objetivo planteado que es informar al usuario de manera numérica de la cantidad de gasolina y precio que corresponde a cada galón ingresado se comprobó que las diferentes cálculos realizados en la programación funcionan de forman compresiva y eficaz.

- Al finalizar este proyecto se pudo concluir que en el parque automotor existen algunos modelos de vehículos que no posen un sistema digital para la verificación de algunos parámetros del automóvil de manera digital. Este proyecto eliminaría esta falencia en este tipo de vehículos. Ya que es fácil de instalar y no es difícil de entender su funcionamiento.

3.2 Recomendaciones.

- El sistema implementado si bien es cierto cumple con los alcances planteados, podría ser modificando la programación del microcontrolador para que permita realizar tareas como:

  - Ayuda en el parqueo del automóvil.
  - Velocidad promedio.
  - Alerta de velocidad máxima para evitar multas.
  - Tiempo de autonomía en el tanque de combustible.
  - Distancia recorrida por cada galón que se consume
  - Estadísticas de consumo de combustible.
  - Estadísticas de manejo del automóvil.
  - Información de posición utilizando un sistema de posicionamiento Global.
  - Alarma del vehículo.
  - Control de elevadores de vidrios, etc.

- Para la programación de microcontrolador se debe seleccionar un programa que sea de fácil programación y que se pueda conseguir en el mercado libremente.

- Por lo menos cada año inspeccione el funcionamiento de los cables de enlace entre los instrumentos de medición y el equipo de visualización, verificar el funcionamiento correcto del LCD.

- Debido a que cuando arranca el vehículo éste consume gran cantidad de corriente es recomendable que cualquier sistema que se le añada al vehículo tome la fuente después que el carro ha arrancado, por lo que se debe implementar una fuente con retardo en el encendido.

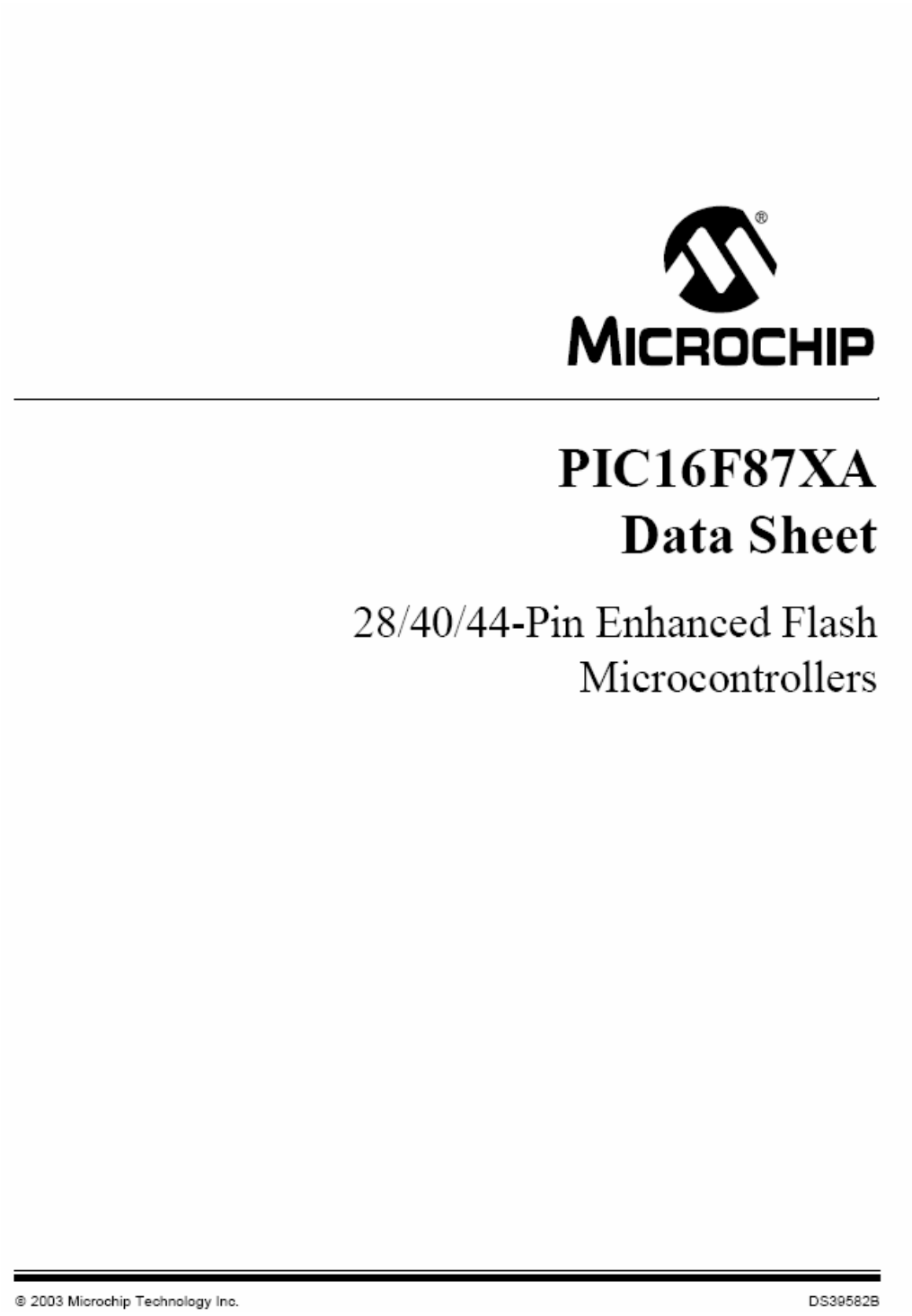
- Finalmente no queda más que decir que el campo de desarrollo en sistemas de información para vehículos es muy amplio y conforme avanza la tecnología mayor cantidad de aditamentos podrían ser implementados para que el dueño de vehículos tenga el mayor confort posible, por lo que no se debería dejar de lado la investigación en este campo.

3.3 Bibliografía

- Costales, A. (2007) Curso de Microcontroladores, PICs. Quito, Escuela Politécnica Nacional.
- Reyes, C. (2008) Aprenda rápidamente a programar microcontroladores PIC.
- Corrales, V. (2006) Electrónica Práctica con Microcontroladores PIC.
- Peña C. (2009) Profesor de planta-Ingeniería mecatrónica, Universidad de Pamplona.
- http://www.datasheetCatalog.com
- http://www.microchip.com
- http://www.unicrom.com
- http://www.areaelectronica.com
- http://www.windpower.dk.com
- http://www.mitecnologico.com/ibq/Main/ObtencionDeGasolinaComercial


ANEXOS

Anexo 1. Microcontrolador PIC 16F877A



MICROCHIP

PIC16F87XA
Data Sheet

28/40/44-Pin Enhanced Flash
Microcontrollers

MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

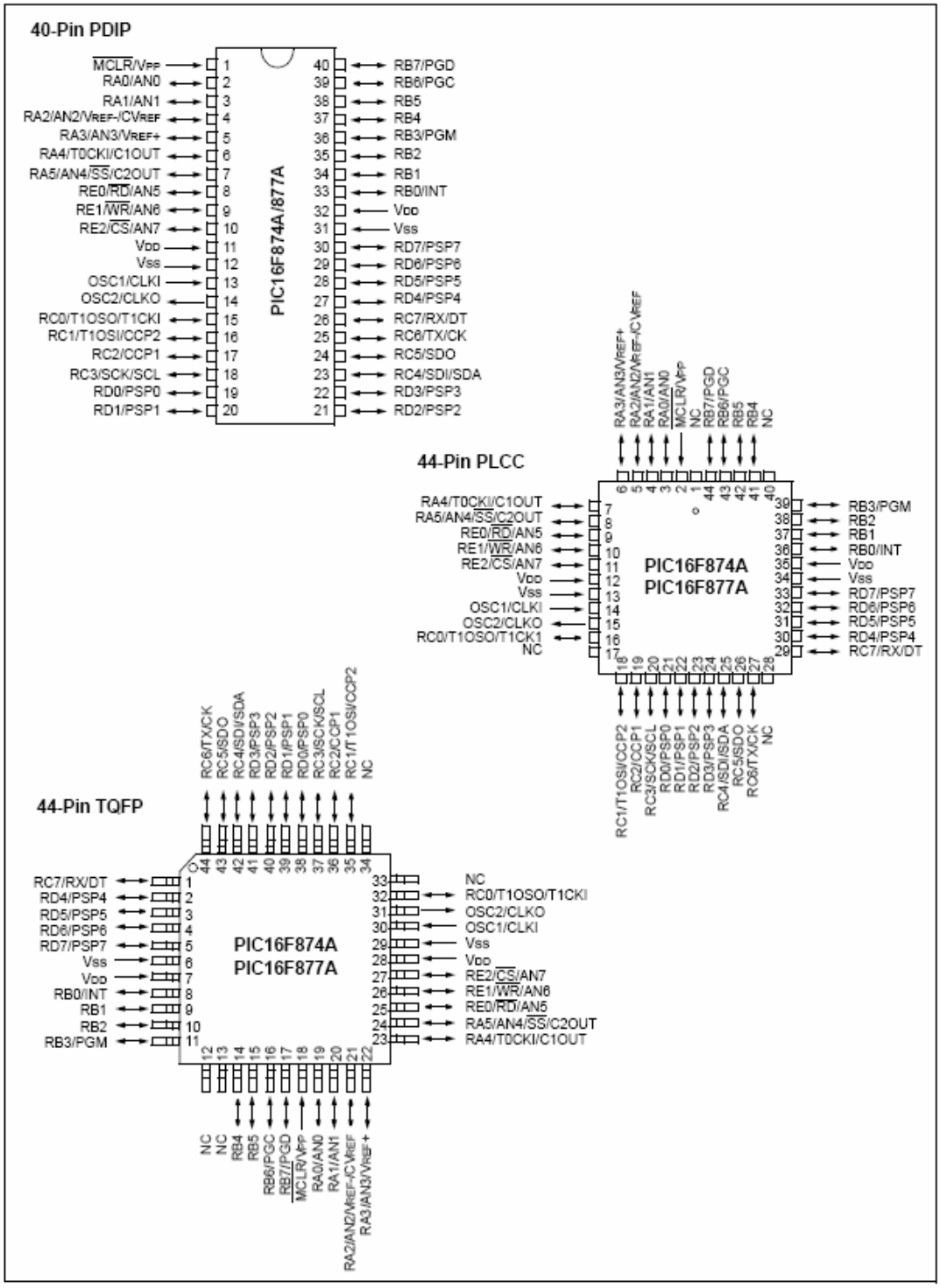
- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

Pin Diagrams (Continued)



1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

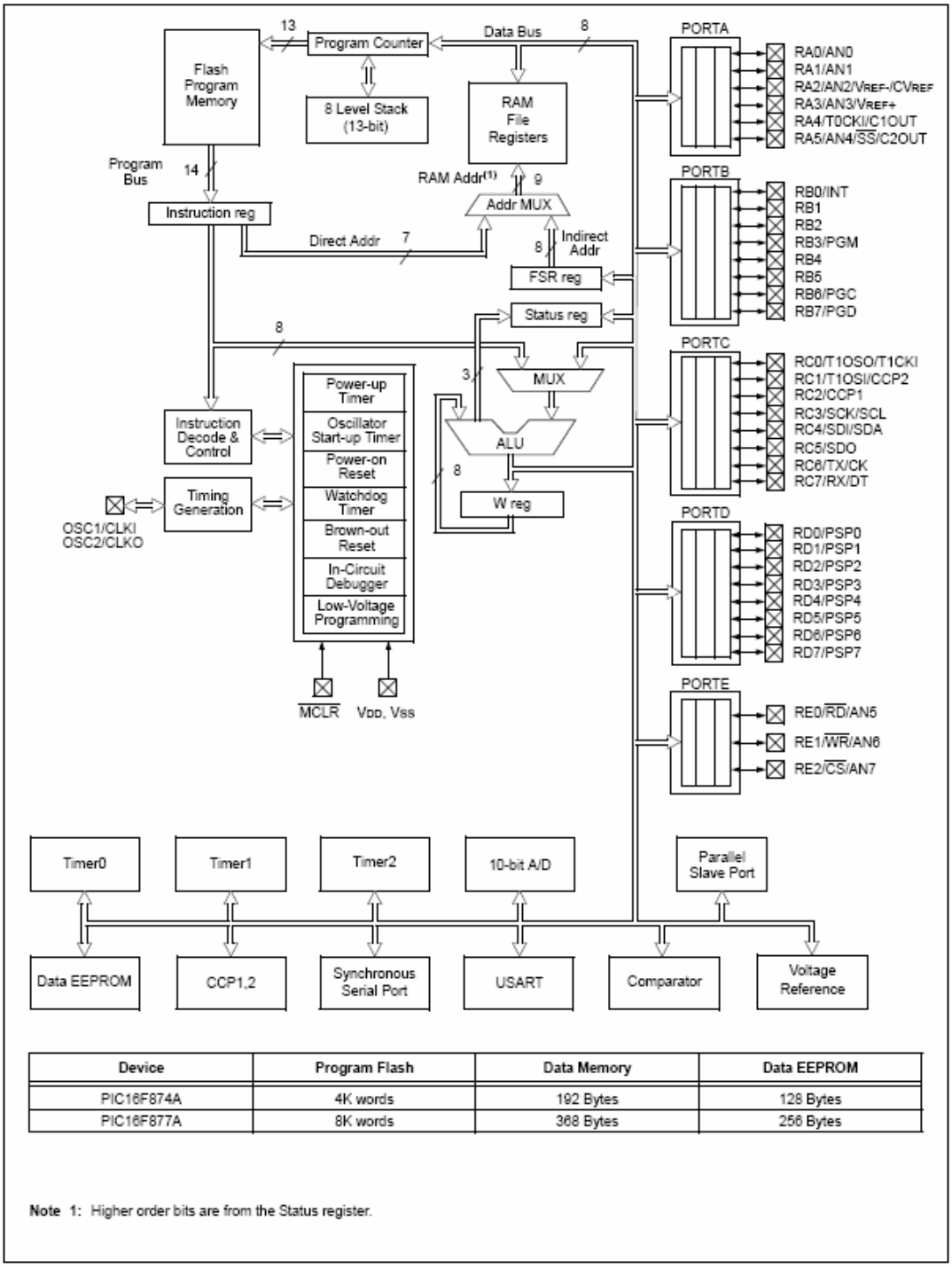
The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

TABLE 1-1: PIC16F87XA DEVICE FEATURES

Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN

FIGURE 1-2: PIC16F874A/877A BLOCK DIAGRAM



Device	Program Flash	Data Memory	Data EEPROM
PIC16F874A	4K words	192 Bytes	128 Bytes
PIC16F877A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKI	13	14	30	32		ST/CMOS(4)	Oscillator crystal or external clock input.
OSC1					I		Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS.
CLKI					I		External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO	14	15	31	33		—	Oscillator crystal or clock output.
OSC2					O		Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO					O		In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	1	2	18	18		ST	Master Clear (input) or programming voltage (output).
$\overline{MCLR}$					I		Master Clear (Reset) input. This pin is an active low Reset to the device.
VPP					P		Programming voltage input.
							PORTA is a bidirectional I/O port.
RA0/AN0	2	3	19	19		TTL	
RA0					I/O		Digital I/O.
AN0					I		Analog input 0.
RA1/AN1	3	4	20	20		TTL	
RA1					I/O		Digital I/O.
AN1					I		Analog input 1.
RA2/AN2/VREF-/CVREF	4	5	21	21		TTL	
RA2					I/O		Digital I/O.
AN2					I		Analog input 2.
VREF-					I		A/D reference voltage (Low) input.
CVREF					O		Comparator VREF output.
RA3/AN3/VREF+	5	6	22	22		TTL	
RA3					I/O		Digital I/O.
AN3					I		Analog input 3.
VREF+					I		A/D reference voltage (High) input.
RA4/T0CKI/C1OUT	6	7	23	23		ST	
RA4					I/O		Digital I/O – Open-drain when configured as output.
T0CKI					I		Timer0 external clock input.
C1OUT					O		Comparator 1 output.
RA5/AN4/$\overline{SS}$/C2OUT	7	8	24	24		TTL	
RA5					I/O		Digital I/O.
AN4					I		Analog input 4.
$\overline{SS}$					I		SPI slave select input.
C2OUT					O		Comparator 2 output.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
							PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	33	36	8	9		TTL/ST(1)	
RB0					I/O		Digital I/O.
INT					I		External interrupt.
RB1	34	37	9	10	I/O	TTL	Digital I/O.
RB2	35	38	10	11	I/O	TTL	Digital I/O.
RB3/PGM	36	39	11	12		TTL	
RB3					I/O		Digital I/O.
PGM					I		Low-voltage ICSP programming enable pin.
RB4	37	41	14	14	I/O	TTL	Digital I/O.
RB5	38	42	15	15	I/O	TTL	Digital I/O.
RB6/PGC	39	43	16	16		TTL/ST(2)	
RB6					I/O		Digital I/O.
PGC					I		In-circuit debugger and ICSP programming clock.
RB7/PGD	40	44	17	17		TTL/ST(2)	
RB7					I/O		Digital I/O.
PGD					I/O		In-circuit debugger and ICSP programming data.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
							PORTC is a bidirectional I/O port.
RC0/T1OSO/T1CKI	15	16	32	34		ST	
RC0					I/O		Digital I/O.
T1OSO					O		Timer1 oscillator output.
T1CKI					I		Timer1 external clock input.
RC1/T1OSI/CCP2	16	18	35	35		ST	
RC1					I/O		Digital I/O.
T1OSI					I		Timer1 oscillator input.
CCP2					I/O		Capture2 input, Compare2 output, PWM2 output.
RC2/CCP1	17	19	36	36		ST	
RC2					I/O		Digital I/O.
CCP1					I/O		Capture1 input, Compare1 output, PWM1 output.
RC3/SCK/SCL	18	20	37	37		ST	
RC3					I/O		Digital I/O.
SCK					I/O		Synchronous serial clock input/output for SPI mode.
SCL					I/O		Synchronous serial clock input/output for I^2C mode.
RC4/SDI/SDA	23	25	42	42		ST	
RC4					I/O		Digital I/O.
SDI					I		SPI data in.
SDA					I/O		I^2C data I/O.
RC5/SDO	24	26	43	43		ST	
RC5					I/O		Digital I/O.
SDO					O		SPI data out.
RC6/TX/CK	25	27	44	44		ST	
RC6					I/O		Digital I/O.
TX					O		USART asynchronous transmit.
CK					I/O		USART1 synchronous clock.
RC7/RX/DT	26	29	1	1		ST	
RC7					I/O		Digital I/O.
RX					I		USART asynchronous receive.
DT					I/O		USART synchronous data.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
							PORTD is a bidirectional I/O port or Parallel Slave Port when interfacing to a microprocessor bus.
RD0/PSP0	19	21	38	38		ST/TTL(3)	
RD0					I/O		Digital I/O.
PSP0					I/O		Parallel Slave Port data.
RD1/PSP1	20	22	39	39		ST/TTL(3)	
RD1					I/O		Digital I/O.
PSP1					I/O		Parallel Slave Port data.
RD2/PSP2	21	23	40	40		ST/TTL(3)	
RD2					I/O		Digital I/O.
PSP2					I/O		Parallel Slave Port data.
RD3/PSP3	22	24	41	41		ST/TTL(3)	
RD3					I/O		Digital I/O.
PSP3					I/O		Parallel Slave Port data.
RD4/PSP4	27	30	2	2		ST/TTL(3)	
RD4					I/O		Digital I/O.
PSP4					I/O		Parallel Slave Port data.
RD5/PSP5	28	31	3	3		ST/TTL(3)	
RD5					I/O		Digital I/O.
PSP5					I/O		Parallel Slave Port data.
RD6/PSP6	29	32	4	4		ST/TTL(3)	
RD6					I/O		Digital I/O.
PSP6					I/O		Parallel Slave Port data.
RD7/PSP7	30	33	5	5		ST/TTL(3)	
RD7					I/O		Digital I/O.
PSP7					I/O		Parallel Slave Port data.
							PORTE is a bidirectional I/O port.
RE0/$\overline{RD}$/AN5	8	9	25	25		ST/TTL(3)	
RE0					I/O		Digital I/O.
$\overline{RD}$					I		Read control for Parallel Slave Port.
AN5					I		Analog input 5.
RE1/$\overline{WR}$/AN6	9	10	26	26		ST/TTL(3)	
RE1					I/O		Digital I/O.
$\overline{WR}$					I		Write control for Parallel Slave Port.
AN6					I		Analog input 6.
RE2/$\overline{CS}$/AN7	10	11	27	27		ST/TTL(3)	
RE2					I/O		Digital I/O.
$\overline{CS}$					I		Chip select control for Parallel Slave Port.
AN7					I		Analog input 7.
VSS	12, 31	13, 34	6, 29	6, 30, 31	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	12, 35	7, 28	7, 8, 28, 29	P	—	Positive supply for logic and I/O pins.
NC	—	1, 17, 28, 40	12,13, 33, 34	13	—	—	These pins are not internally connected. These pins should be left unconnected.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

2.0 MEMORY ORGANIZATION

There are three memory blocks in each of the PIC16F87XA devices. The program memory and data memory have separate buses so that concurrent access can occur and is detailed in this section. The EEPROM data memory block is detailed in **Section 3.0 "Data EEPROM and Flash Program Memory"**.

Additional information on device memory may be found in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

2.1 Program Memory Organization

The PIC16F87XA devices have a 13-bit program counter capable of addressing an 8K word x 14 bit program memory space. The PIC16F876A/877A devices have 8K words x 14 bits of Flash program memory, while PIC16F873A/874A devices have 4K words x 14 bits. Accessing a location above the physically implemented address will cause a wraparound.

The Reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PIC16F876A/877A PROGRAM MEMORY MAP AND STACK

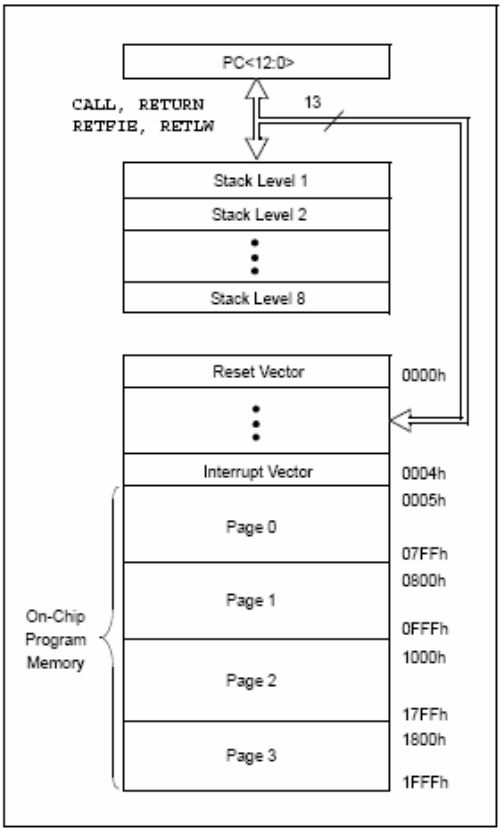


FIGURE 2-2: PIC16F873A/874A PROGRAM MEMORY MAP AND STACK

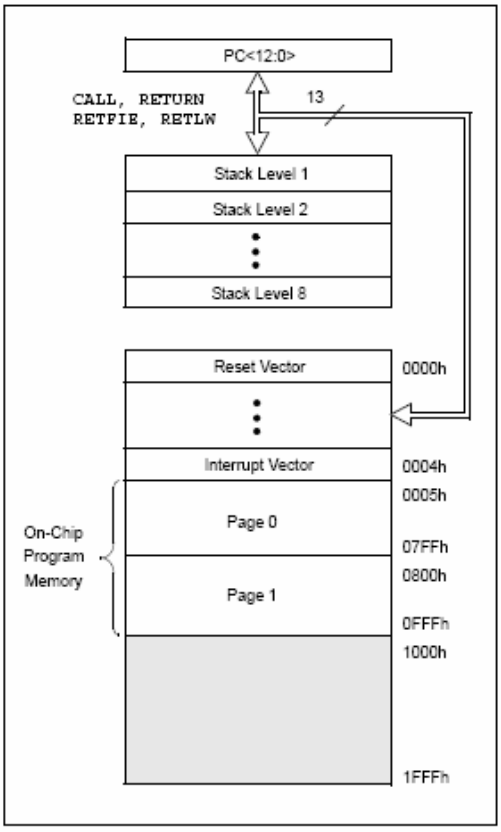


FIGURE 2-3: PIC16F876A/877A REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	CMCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	CVRCON	9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
			EFh		16Fh		1EFh
accesses 70h-7Fh	7Fh	accesses 70h-7Fh	F0h FFh	accesses 70h-7Fh	170h 17Fh	accesses 70h - 7Fh	1F0h 1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F876A.
2: These registers are reserved; maintain these registers clear.

14.2 Oscillator Configurations

14.2.1 OSCILLATOR TYPES

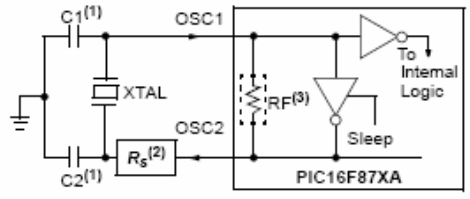
The PIC16F87XA can be operated in four different oscillator modes. The user can program two configuration bits (Fosc1 and Fosc0) to select one of these four modes:

- LP Low-Power Crystal
- XT Crystal/Resonator
- HS High-Speed Crystal/Resonator
- RC Resistor/Capacitor

14.2.2 CRYSTAL OSCILLATOR/CERAMIC RESONATORS

In XT, LP or HS modes, a crystal or ceramic resonator is connected to the OSC1/CLKI and OSC2/CLKO pins to establish oscillation (Figure 14-1). The PIC16F87XA oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturer's specifications. When in XT, LP or HS modes, the device can have an external clock source to drive the OSC1/CLKI pin (Figure 14-2).

FIGURE 14-1: CRYSTAL/CERAMIC RESONATOR OPERATION (HS, XT OR LP OSC CONFIGURATION)



Note 1: See Table 14-1 and Table 14-2 for recommended values of C1 and C2.

2: A series resistor (R_S) may be required for AT strip cut crystals.

3: RF varies with the crystal chosen.

FIGURE 14-2: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

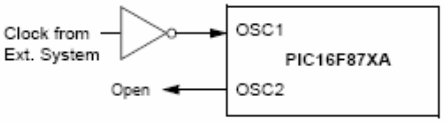


TABLE 14-1: CERAMIC RESONATORS

Ranges Tested:			
Mode	**Freq.**	**OSC1**	**OSC2**
XT	455 kHz 2.0 MHz 4.0 MHz	68-100 pF 15-68 pF 15-68 pF	68-100 pF 15-68 pF 15-68 pF
HS	8.0 MHz 16.0 MHz	10-68 pF 10-22 pF	10-68 pF 10-22 pF
These values are for design guidance only. See notes following Table 14-2.			
Resonators Used:			
2.0 MHz	Murata Erie CSA2.00MG	± 0.5%	
4.0 MHz	Murata Erie CSA4.00MG	± 0.5%	
8.0 MHz	Murata Erie CSA8.00MT	± 0.5%	
16.0 MHz	Murata Erie CSA16.00MX	± 0.5%	
All resonators used did not have built-in capacitors.			

TABLE 14-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq.	Cap. Range C1	Cap. Range C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15-33 pF	15-33 pF
	20 MHz	15-33 pF	15-33 pF
These values are for design guidance only. See notes following this table.			
Crystals Used			
32 kHz	Epson C-001R32.768K-A	± 20 PPM	
200 kHz	STD XTL 200.000KHz	± 20 PPM	
1 MHz	ECS ECS-10-13-1	± 50 PPM	
4 MHz	ECS ECS-40-20-1	± 50 PPM	
8 MHz	EPSON CA-301 8.000M-C	± 30 PPM	
20 MHz	EPSON CA-301 20.000M-C	± 30 PPM	

Note 1: Higher capacitance increases the stability of oscillator but also increases the start-up time.

2: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

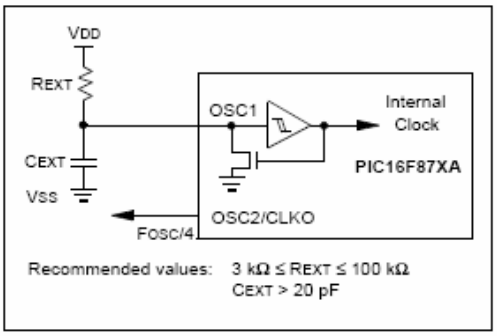
3: R_s may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification.

4: When migrating from other PICmicro® devices, oscillator performance should be verified.

14.2.3 RC OSCILLATOR

For timing insensitive applications, the "RC" device option offers additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal process parameter variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 14-3 shows how the R/C combination is connected to the PIC16F87XA.

FIGURE 14-3: RC OSCILLATOR MODE



Anexo 2. Regulador LM7805

FAIRCHILD SEMICONDUCTOR®

www.fairchildsemi.com

MC78XX/LM78XX/MC78XXA

3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Description

The MC78XX/LM78XX/MC78XXA series of three terminal positive regulators are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

TO-220

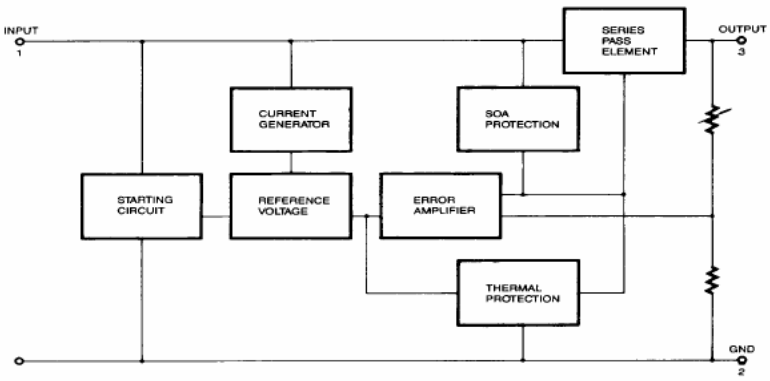
1

D-PAK

1

1. Input 2. GND 3. Output

Internal Block Digram



Rev. 1.0.1

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	°C/W
Operating Temperature Range	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	T_{STG}	-65 ~ +150	°C

Electrical Characteristics (MC7805/LM7805)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I = 10V, C_I= 0.33µF, C_O= 0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		MC7805/LM7805			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		4.8	5.0	5.2	V
		5.0mA ≤ Io ≤ 1.0A, P_O ≤ 15W V_I = 7V to 20V		4.75	5.0	5.25	
Line Regulation (Note1)	Regline	T_J=+25 °C	V_O = 7V to 25V	-	4.0	100	mV
			V_I = 8V to 12V	-	1.6	50	
Load Regulation (Note1)	Regload	T_J=+25 °C	I_O = 5.0mA to1.5A	-	9	100	mV
			I_O =250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.03	0.5	mA
		V_I= 7V to 25V		-	0.3	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O= 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A=+25 °C		-	42	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_O = 8V to 18V		62	73	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	15	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_O due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Anexo 3. Programa

```
Device 16F877A
XTAL=4

TRISB=%11111111     DEFINIMOS COMO ENTRADAS
TRISD=0
ADCON1=%11000100    CONVERSION  HACIA LA DERECHA A 10BITS
TRISA.3=0

Declare LCD_DTPIN       PORTD.4
Declare LCD_ENPIN       PORTA.2
Declare LCD_RSPIN       PORTA.5
Declare LCD_LINES         2
Declare ADIN_RES          10          DEFINE 10 BITS
Declare ADIN_TAD          2
Declare ADIN_STIME        100         DELAY ADC

Dim GALON_ACTUAL      As Float      VARIABLE
Dim GALON_INGRESADO       As Float      VARIABLE
Dim TANQUE                As Byte       VARIABLE
Dim DATO              As Byte       VARIABLE
Dim I                 As Byte       VARIABLE
Dim TECLA              As Byte      VARIABLE
Dim ANALOGICO         As Word       VARIABLE
Dim GASOLINA              As Float      VARIABLE
Dim voltaje           As Float      VARIABLE
Dim PRECIO_G              As Float      VARIABLE
Dim PRECIO            As Float      VARIABLE
Dim PRECIO_G4[8]      As Byte       VARIABLE PUNTO FLOTANTE LCD
Dim PRECIO_G1         As Float      VARIABLE
Dim GALON_CALCULADO As Float    VARIABLE
```

```
Dim N                    As Byte     VARIABLE
Dim PUNTO                 As Byte    VARIABLE
Dim PUNTOS               As Byte     VARIABLE
Dim y                    As Byte     VARIABLE
'*************************************
'*************************************
Symbol X1 = PORTD.     3
Symbol X2 = PORTD.     2
Symbol X3 = PORTD.     1
Symbol X4 = PORTD.     0
Symbol Y1 = PORTB.     3
Symbol Y2 = PORTB.     2
Symbol Y3 = PORTB.     1
Symbol Y4 = PORTB.     0
Symbol led = PORTA.    3

Cls
'*************************************
'*************************************

Low led
DelayMS 1000
For I=0 To 15                          , numero de caracteres
DATO=LookUpL I , [ "  ESFOT  MEDIDOR  " ] ,muestra caracteres en el LCD
Print At 1 , I ,  DATO                 , contador de caracteres
DelayMS 100                            , tiempo que dura los caracteres LCD
Next I                                 , salto

For I=0 To 15                          , numero de caracteres
LookUpL I,[ "  DE  GASOLINA  " ],DATO ,muestra caracteres en el LCD
Print At 2 , I , DATO                  , contador de caracteres
DelayMS 100                            , tiempo que dura los caracteres LCD
```

```
Next I                              , salto

DelayMS 1000                        , tiempo de la frase  en el LCD
Cls
For I=0 To 15                       , numero de caracteres
LookUpL I,["  SANTIAGO  I.  "],DATO  ,muestra caracteres en el LCD
Print At 1 , I , DATO               , contador de caracteres
DelayMS 100                         , tiempo que dura los caracteres LCD

Next I                              , salto

For I=0 To 15                       , numero de caracteres
LookUpL I,[" MONTESDEOCA E. "],DATO ,muestra caracteres en el LCD
Print At 2 , I , DATO               , contador de caracteres
DelayMS 100                         , tiempo que dura los caracteres LCD
Next                                , salto

DelayMS 1000                        , tiempo de la frase  en el LCD
Cls

TANQUE = ERead 1                    , valor que se guarda en la eeprom
If TANQUE>100 Then TANQUE=11        , valor de la capacidad máxima

PRECIO_G=ERead 3                    , valor que se guarda en la eeprom
If PRECIO_G>10 Then PRECIO_G=1.23 , valor de la capacidad máxima

'************************************
'************************************

inicio:                             , rutina
GoSub TECLADO                       , llamado de la rutina
```

```
ANALOGICO= ADIn                          , convertidor de AC/DC
GASOLINA=ANALOGICO*TANQUE/1023 , calculo capacidad del tanque
If GASOLINA<1.5 Then  , valor de voltaje menor a 1,5v  visualiza mensaje LCD
  Print At 1,1 , "COMBUSTIBLE POR " , mensaje menor 1,5v
  Print At 2,1 , "   TERMINAR     "        , mensaje menor 1,5v
   High led                              , led 1L que dura mensaje
  DelayMS 100                            , tiempo que esta el led 1L
  Low led                                , led 0L que dura  mensaje
  DelayMS 100                            , tiempo que esta el led 0L
'************************************************************
 normal:                                 , rutina
  GoSub TECLADO                          , llamado de la rutina
  If TECLA=12 Then                       , tecla de que señala la opción
  Print At 1,1,"GALON_I   PRECIO"        , mensaje luego de tecla seleccionada
  Print At 2,1,"              "          , escribe valor del precio por galón de gasolina
  GALON_ACTUAL=GASOLINA                  , compara entre  valores
  TECLA=16                               , escaneo de las 16 teclas del teclado
  DelayMS 200                            , tiempo que se ve el mensaje
  While  TECLA<>12                       , condición si se aplasta la tecla 12
  ANALOGICO= ADIn 0                      , convertidor de AC/DC
  GASOLINA=ANALOGICO*TANQUE/1023 , cálculo capacidad del tanque
  PRECIO=GALON_INGRESADO*PRECIO_G, cálculo del precio de la gasolina
*************************************************************************
   GoSub TECLADO                         , llamado de la rutina
  If TECLA=13 Then                       , tecla de que señala la opción
     TECLA=12                            , tecla para salir de la opción anterior
   End If
  GALON_INGRESADO=GASOLINA-GALON_ACTUAL,cálculo de gl. ingresado
  Print At 2,1,Dec GALON_INGRESADO,"  " msje luego de tecla seleccionada

  Print At 2,11,"$",Dec PRECIO,"  "      , msje luego de tecla seleccionada
```

```
  GoSub TECLADO                          , llamado de la rutina
  If TECLA=13 Then Break, tecla para cortar la función que está realizando
  Wend
End If
  ANALOGICO= ADIn 0                      , convertidor de AC/DC
  GASOLINA=ANALOGICO*TANQUE/1023, cálculo capacidad del tanque
  If GASOLINA< 1.5 Then GoTo normal, reactiva el programa mayor de 1.5v
  Print At 2,1 , "              "
End If

 '*************************************************************
PRECIO=GASOLINA*PRECIO_G         , cálculo del precio
Print At 1,1, "GALONES "         , mesj 1ra línea del LCD posición 1
Print At 1,10, "PRECIO  "        , mesj 1ra línea del LCD posición 10
Print At 2,1, Dec GASOLINA,"Gl  "  , mesj 2da línea del LCD posición 1
Print At 2,10, "$", Dec PRECIO ,"  "  , mesj 2da línea del LCD posición 10

If TECLA=10 Then                         , tecla de que señala la opción
  Print At 1,1 ," VOLUMEN TANQUE "       , mesj 1ra línea del LCD posición 1
  Print At 2,1 ,"              "         , mesj 2da línea del LCD posición 1
  DelayMS 1000                           , tiempo para visualizar el mensaje
  GoSub VOLUMEN_TANQUE                   , llama a la rutina
End If

If TECLA=11 Then                         , tecla de que señala la opción
  Print At 1,1 ,"PRECIO POR GALON"       , mesj 1ra línea del LCD posición 1
  Print At 2,1 ,"              "         , mesj 2da línea del LCD posición 1
  DelayMS 1000                           , tiempo para visualizar el mensaje
  GoSub PRECIO_GALON                     , llama a la rutina
End If
```

```
If TECLA=12 Then                          , tecla de que señala la opción
  Print At 1,1,"GALON_I   PRECIO"         , mesj 1ra línea del LCD posición 1
  Print At 2,1,"              "           , mesj 2da línea del LCD posición 1
  GALON_ACTUAL=GASOLINA                   , compara entre  valores
  TECLA=16                                , escaneo de las 16 teclas del teclado
  DelayMS 200                             , tiempo para visualizar el mensaje
  While  TECLA<>12                        , condición si se aplasta la tecla 12
  ANALOGICO= ADIn 0                       , convertidor de AC/DC
  GASOLINA=ANALOGICO*TANQUE/1023 ,cálculo capacidad del tanque
  PRECIO=GALON_INGRESADO*PRECIO_G , cálculo del precio
  GoSub TECLADO                           , llamado de la rutina
  If TECLA=13 Then                        , tecla de que señala la opción
    TECLA=12                              , tecla para salir de la opción anterior
  End If
  GALON_INGRESADO=GASOLINA-GALON_ACTUAL , calculo galones ingr.
  Print At 2,1,Dec GALON_INGRESADO,"  " , mesj 2da línea del LCD posición 1
  Print At 2,11,"$",Dec PRECIO,"  "       , mesj 2da línea del LCD posición 11
  Wend
End If

If TECLA=14  Then                         , tecla de que señala la opción
GoSub CALCULADORA                         , llamado de la rutina

'*********************************************************
GoTo inicio                               , ir de la rutina
TECLADO:                                  , rutina
TECLA=16                                  , define teclado de 16 teclas
Low X1                                    , teclas 0L
        , escaneo de las teclas 1, 2, 3, y 10 a 2 segundos
If Y1=0 Then TECLA=  1:High  X1:DelayMS 200:Return
If Y2=0 Then TECLA=  2:High  X1:DelayMS 200:Return
```

```
If Y3=0 Then TECLA=  3:High  X1:DelayMS 200:Return
If Y4=0 Then TECLA=10:High  X1:DelayMS 200:Return
High X1                                  , teclas 1L
Low X2                                   , teclas 0L
        , escaneo de las teclas 4, 5, 6, y 11 a 2 segundos
If Y1=0 Then TECLA=  4:High X2:DelayMS 200:Return
If Y2=0 Then TECLA=  5:High X2:DelayMS 200:Return
If Y3=0 Then TECLA=  6:High X2:DelayMS 200:Return
If Y4=0 Then TECLA=11:High X2:DelayMS 200:Return
High X2                                  , teclas 1L
Low X3                                   , teclas 0L
        , escaneo de las teclas 7, 8, 9, y 12 a 2 segundos
If Y1=0 Then TECLA=  7:High X3:DelayMS 200:Return
If Y2=0 Then TECLA=  8:High X3:DelayMS 200:Return
If Y3=0 Then TECLA=  9:High X3:DelayMS 200:Return
If Y4=0 Then TECLA=12:High X3:DelayMS 200:Return
High X3                                  , teclas 1L
Low X4                                   , teclas 0L
        , escaneo de las teclas 15, 0, 14, y 13 a 2 segundos
If Y1=0 Then TECLA=15:High X4:DelayMS 200:Return
If Y2=0 Then TECLA=  0:High X4:DelayMS 200:Return
If Y3=0 Then TECLA=14:High X4:DelayMS 200:Return
If Y4=0 Then TECLA=13:High X4:DelayMS 200:Return
High X4                                  , teclas 1L
Return
'*********************************************************
VOLUMEN_TANQUE:                          , rutina
TECLA=16                                 , define teclado de 16 teclas
While TECLA<> 10                         , espera para ser activada la tecla 10
  GoSub TECLADO                          , llamado de la rutina
  If TECLA=13 Then                       , tecla de que señala la opción
```

```
    TECLA=10                          , tecla para salir de la opción anterior
   End If
  End If
  If TECLA<10 Then                    , para salir luego de digital el valor
    If N=0 Then
      TANQUE=TECLA                    , escribir el valor deseado
      N=1                             , variable
    Else
      TANQUE=TANQUE*10+TECLA          , calculo para digitar el valor requerido
      EWrite 1 , [ TANQUE ]           , guarda valor en la eeprom
      If N=1 Then N=0
    End If
    Print At 2,1,Dec TANQUE," Gl  "   , mensaje luego del valor digitado
  End If
Wend
N=0
TECLA=16                              , define teclado de 16 teclas
DelayMS 200                           , tiempo del mensaje en el LCD
Return
  '*************************************************************
CALCULADORA:                          , rutina
Print At 1,1,"TEC1 #GAL A $VAL", mensaje luego de seleccionar la tecla 1
Print At 2,1,"TEC2 $VAL A #GAL", mensaje luego de seleccionar la tecla 2

TECLA=16                              , define teclado de 16 teclas
DelayMS 200                           , tiempo del mensaje en el LCD

While TECLA<>14                       , espera para ser activada la tecla 14
GoSub TECLADO                         , llamado de la rutina
If TECLA=13 Then TECLA=14             , espera para ser activada la tecla 14
If TECLA=1 Then GoSub GALVAL          , llamado de la rutina
```

```
If TECLA=2 Then GoSub VALGAL       , llamado de la rutina
If y=1 Then Break                  , para salir de la acción en cual quier momento
Wend
y=0                                , variable
Return
 '*************************************************************
GALVAL:                            , rutina
Print At 1,1,"INGRESE #GALONES"    , mensaje en el LCD
Print At 2,1,"                "    , espacio para escribir el valor de galones

TECLA=16                           , define teclado de 16 teclas
N=1                                , variable
While TECLA<> 14                   , espera para ser activada la tecla 14
 If TECLA=13 Then TECLA=14         ,
 GoSub TECLADO                     , llamado de la rutina
 If TECLA=13 Then                  , espera para ser activada la tecla 13
   TECLA=11                        , espera para ser activada la tecla 11
 End If
 If TECLA=15 Then                  , detecta punto decimal
  PUNTOS=1                         , ubica del punto en el LCD
  N=N+1                            , posición del la variable
  PRECIO_G4[N]="."                 , guardo la variable decimal
  Print At 2,N,PRECIO_G4[N]        ,visualizo el punto decimal en el LCD
 End If
 If TECLA<10 Then
  If PUNTOS=1 Then PUNTO=PUNTO+1, incrementa el punto en el LCD
  If N=1 Then
    N=2                            , variable para el punto decimal
    GALON_CALCULADO=TECLA          , calculo para # de galones
    PRECIO_G4 [1]=TECLA            , calculo para # de galones
    Print At 2,1," ",Dec PRECIO_G4[1] , visualizo el precio en el LCD
```

```
    Else
      N=N+1                              , incremento del punto decimal
      PRECIO_G4 [N]=TECLA                , ubicación del punto decimal en  valor
      Print At 2,N,Dec PRECIO_G4[N]      , visualizo punto decimal en LCD
      GALON_CALCULADO=GALON_CALCULADO*10+TECLA  , calculo para #
                                                          de galones
      If N=6 Then
        N=1                              , variable
        PUNTO=0                          , ubica del punto
        PUNTOS=0                         , ubica del punto
        PRECIO_G=0                       , precio diferente de cero
      End If
    End If
  End If
  GoSub TECLADO                          , llamado a la rutina
  If TECLA=13 Then                       , tecla de que señala la opción
    y=1                                  , variable
    Break                    , corta la operación que está realizando el programa
  End If
Wend
If y=0 Then
   If PUNTO=1 Then PRECIO_G1=GALON_CALCULADO/10  , galones
                                                  precio y punto decimal
  If PUNTO=2 Then PRECIO_G1=GALON_CALCULADO/100 , punto decimal
  PRECIO=GALON_CALCULADO*PRECIO_G  , calculo de precio
  Print At 2,9," $",Dec PRECIO,"  "      ,visualice mensaje en LCD
  DelayMS 1000                           , tiempo del mensaje
  PUNTO=0                                ,  escribir punto para  en el LCD
  PUNTOS=0                               , escribir punto parar en el LCD
  N=0
  TECLA=16                               , define teclado de 16 teclas
```

```
End If
DelayMS 200                              , tiempo del mensaje
Return

 '*************************************************************
VALGAL:                                  , rutina
Print At 1,1," INGRESE #VALOR "          , mensaje en el LCD
Print At 2,1,"                "          , espacio para escribir el valor dólares
TECLA=16                                 , define teclado de 16 teclas
N=1                                      , variable
While TECLA<> 14                         , espera para ser activada la tecla 14
 If TECLA=13 Then TECLA=14
 GoSub TECLADO                           , llamado a la rutina
 If TECLA=13 Then                        , espera para ser activada la tecla 13
   TECLA=11                              , espera para ser activada la tecla 11
 End If
 If TECLA=15 Then                        , detecto punto decimal
  PUNTOS=1                               , ubicación del punto en el LCD
  N=N+1                                  , posición del la variable
  PRECIO_G4[N]="."                       , guardo la variable decimal
  Print At 2,N,PRECIO_G4[N]              , visualizo el punto decimal en el LCD
 End If
 If TECLA<10 Then
  If PUNTOS=1 Then PUNTO=PUNTO+1, incrementa el punto en el LCD
  If N=1 Then
   N=2                                   , variable para el punto decimal
   GALON_CALCULADO=TECLA                 , calculo para # de galones
   PRECIO_G4[1]=TECLA                    , calculo para el precio
   Print At 2,1," ",Dec PRECIO_G4[1] , visualizo el precio en el LCD
  Else
   N=N+1                                 , incremento del punto decimal
```

```
    PRECIO_G4[N]=TECLA              , ubica el punto decimal en la variable
    Print At 2,N,Dec PRECIO_G4[N]   , ubicación del punto decimal en  valor
    GALON_CALCULADO=GALON_CALCULADO*10+TECLA, calculo para #
                                                          de galones
    If N=6 Then
      N=1                           , variable
      PUNTO=0                       , ubica del punto
      PUNTOS=0                      , ubica del punto
      PRECIO_G=0                    , precio diferente de Cero
    End If
   End If
  End If
  GoSub TECLADO                     , escanea teclas del teclado 4x4
  If TECLA=13 Then                  , espera para ser activada la tecla 13
   y=1                              , variable
   Break                 , corta la operación que está realizando el programa
  End If
Wend
If y=0 Then
  If PUNTO=1 Then PRECIO_G1=GALON_CALCULADO/10 ,  cálculo galones
                                                  precio y  punto decimal
  If PUNTO=2 Then PRECIO_G1=GALON_CALCULADO/100 , cálculo galones
                                                  precio y punto decimal
  PRECIO=GALON_CALCULADO/PRECIO_G , calculo de precio
  Print At 2,8,Dec PRECIO,"Gl  "    , mensaje en el LCD línea 2 posición 8
  DelayMS 1000                      , tiempo del mensaje Gl
  PUNTO=0                           , ubica del punto
  PUNTOS =0                         , ubica del punto
  N=0                               , variable del punto
  TECLA=16                          , escanea teclas del teclado 4x4
End If
```

```
DelayMS 200                          , tiempo del mensaje
Return
 '*************************************************************
PRECIO_GALON:                        , rutina
TECLA=16                             , escanea teclas del teclado 4x4
N=1                                  , variable
While TECLA<> 11                     , espera para ser activada la tecla 11
  GoSub TECLADO                      , llamado le la rutina
  If TECLA=13 Then  TECLA=11
   End If
  If TECLA=15 Then                   , detecto punto decimal
    PUNTOS=1                         , ubica del punto en el LCD
    N=N+1                            , incremento del punto decimal
    PRECIO_G4[N]="."                 , guardo la variable decimal
    Print At 2,N,PRECIO_G4[N]        ,  visualizo el punto decimal en el LCD
  End If
  If TECLA<10 Then
    If PUNTOS=1 Then PUNTO=PUNTO+1, incrementa el punto en el LCD
    If N=1 Then
      N=2                            , variable para el punto decimal
      PRECIO_G=TECLA                 , calculo para el precio
      PRECIO_G4[1]=TECLA             , calculo para el precio
      Print At 2,1,"$",Dec PRECIO_G4[1] , ubicación del punto decimal en  valor
    Else
      N=N+1                          , incremento del punto decimal
      PRECIO_G4[N]=TECLA             , ubica el punto decimal en la variable
      Print At 2,N,Dec PRECIO_G4[N]  , ubicación del punto decimal en  valor
      PRECIO_G=PRECIO_G*10+TECLA  , calculo  precio y ubica el decimal
      If N=6 Then
        N=1                          , variable
        PUNTO=0                      , ubica del punto
```

```
      PUNTOS=0                    , ubica del punto
      PRECIO_G=0                  , precio diferente de Cero
    End If
  End If
End If
Wend
If PUNTO=1 Then PRECIO_G1=PRECIO_G/10  , cálculo  precio y ubicación
                                          del punto decimal
If PUNTO=2 Then PRECIO_G1=PRECIO_G/100  , cálculo  precio y ubicación
                                           del punto decimal
PRECIO_G=PRECIO_G1            , cálculo del precio
EWrite 3 , [ PRECIO_G ]       , guarda el precio calculado en la eepron
DelayMS 1000                  , tiempo de espera

PUNTO=0                       , ubica del punto
PUNTOS =0                     , ubica del punto
N=0                           , variable
TECLA=16                      , escanea las teclas del teclado 4x4
DelayMS 200                   ,  tiempo del mensaje
Return
End                           , FIN DEL PROGAMA
```

Anexo 4. Set de Instrucciones

ADIN	Read the on-board analogue to digital converter.
ASM-ENDASM	Insert assembly language code section.
BOX	Draw a square on a graphic LCD.
BRANCH	Computed **GOTO** (equiv. to ON..GOTO).
BRANCHL	**BRANCH** out of page (long **BRANCH**).
BREAK	Exit a loop prematurely.
BSTART	Send a **START** condition to the I^2C bus.
BSTOP	Send a **STOP** condition to the I^2C bus.
BRESTART	Send a **RESTART** condition to the I^2C bus.
BUSACK	Send an **ACKNOWLEDGE** condition to the I^2C bus.
BUSIN	Read bytes from an I^2C device.
BUSOUT	Write bytes to an I^2C device.
BUTTON	Detect and debounce a key press.
CALL	Call an assembly language subroutine.
CDATA	Define initial contents in memory.
CF_INIT	Initialise the interface to a Compact Flash card.
CF_SECTOR	Point to the sector of interest in a Compact Flash card.
CF_READ	Read data from a Compact Flash card.
CF_WRITE	Write data to a Compact Flash card.
CIRCLE	Draw a circle on a graphic LCD.
CLEAR	Place a variable or bit in a low state, or clear all RAM area.
CLEARBIT	Clear a bit of a port or variable, using a variable index.
CLS	Clear the LCD.
CONFIG	Set or Reset programming fuse configurations.
COUNTER	Count the number of pulses occurring on a pin.
CREAD	Read data from code memory.
CURSOR	Position the cursor on the LCD.
CWRITE	Write data to code memory.
DATA	Define initial contents in memory.
DEC	Decrement a variable.
DECLARE	Adjust library routine parameters.
DELAYMS	Delay (1mSec resolution).
DELAYUS	Delay (1uSec resolution).
DEVICE	Choose the type of PICmicro™ to compile with.
DIG	Return the value of a decimal digit.
DIM	Create a variable.
DISABLE	DISABLE software interrupts previously ENABLED.
DTMFOUT	Produce a DTMF Touch Tone note.
EDATA	Define initial contents of on-board EEPROM.
ENABLE	ENABLE software interrupts previously DISABLED.
END	Stop execution of the BASIC program.
EREAD	Read a value from on-board EEPROM.
EWRITE	Write a value to on-board EEPROM.
FOR...TO...NEXT...STEP	Repeatedly execute statements.
FREQOUT	Generate one or two tones, of differing or the same frequencies.
GETBIT	Examine a bit of a port or variable, using a variable index.
GOSUB	Call a BASIC subroutine at a specified label.
GOTO	Continue execution at a specified label.
HBSTART	Send a **START** condition to the I^2C bus using the MSSP module.
HBSTOP	Send a **STOP** condition to the I^2C bus using the MSSP module.
HBRESTART	Send a **RESTART** condition to the I^2C bus using the MSSP module.
HBUSACK	Send an **ACK** condition to the I^2C bus using the MSSP module.
HBUSIN	Read from an I^2C device using the MSSP module.

HBUSOUT Write to an I^2C device using the MSSP module.
HIGH Make a pin or port high.
HPWM Generate a PWM signal using the CCP module.
HRSIN Receive data from the serial port on devices that contain a USART.
HRSOUT Transmit data from the serial port on devices that contain a USART.
HSERIN Receive data from the serial port on devices that contain a USART.
HSEROUT Transmit data from the serial port on devices that contain a USART.
HRSIN2 Same as **HRSIN** but using a 2nd USART if available.
HRSOUT2 Same as **HRSOUT** but using a 2nd USART if available.
HSERIN2 Same as **HSERIN** but using a 2nd USART if available.
HSEROUT2 Same as **HSEROUT** but using a 2nd USART if available.
IF..THEN..ELSEIF..ELSE..ENDIF Conditionally execute statements.
INC Increment a variable.
INCLUDE Load a BASIC file into the source code.
INKEY Scan a keypad.
INPUT Make pin an input.
[LET] Assign the result of an expression to a variable. (Optional command).
LCDREAD Read a single byte from a Graphic LCD.
LCDWRITE Write bytes to a Graphic LCD.
LEFT$ Extract *n* amount of characters from the left of a String. For 18F devices only.
LDATA Place information into code memory. For access by **LREAD**.
LINE Draw a line in any direction on a graphic LCD.
LINETO Draw a straight line in any direction on a graphic LCD, starting from the previous LINE command's end position.
LOADBIT Set or Clear a bit of a port or variable, using a variable index.
LOOKDOWN Search a constant lookdown table for a value.
LOOKDOWNL Search constant or variable lookdown table for a value.
LOOKUP Fetch a constant value from a lookup table.
LOOKUPL Fetch a constant or variable value from lookup table.
LOW Make a pin or port low.
LREAD Read a value from an **LDATA** table and place into Variable.
LREAD8, LREAD16, LREAD32 Read a single or multi-byte value from an **LDATA** table with more efficiency than **LREAD**.
MID$ Extract *n* amount of characters from a String beginning at *n* characters from the left. For 18F devices only.
ON INTERRUPT Execute a subroutine using a SOFTWARE interrupt.
ON_INTERRUPT Execute an ASSEMBLER subroutine on a HARWARE interrupt.
ON_LOW_INTERRUPT Execute an ASSEMBLER subroutine when a LOW PRIORITY HARDWARE interrupt occurs on a 16-bit core device.
ON GOSUB Call a Subroutine based on an Index value. For 18F devices only.
ON GOTO Jump to an address in code memory based on an Index value. (Primarily for smaller PICmicros)
ON GOTOL Jump to an address in code memory based on an Index value. (Primarily for larger PICmicros)
OUTPUT Make a pin an output.
OREAD Receive data from a device using the Dallas 1-wire protocol.
OWRITE Send data to a device using the Dallas 1-wire protocol.
ORG Set Program Origin.

POT	Read a potentiometer on specified pin.
PRINT	Display characters on an LCD.
PULSIN	Measure the pulse width on a pin.
PULSOUT	Generate a pulse to a pin.
PWM	Output a pulse width modulated pulse train to pin.
RANDOM	Generate a pseudo-random number.
RCIN	Measure a pulse width on a pin.
READ	Read a value from memory.
REM	Add a remark to the source code.
REPEAT...UNTIL	Execute a block of instructions until a condition is true.
RESTORE	Adjust the position of data to **READ**.
RESUME	Re-enable software interrupts and return.
RETURN	Continue at the statement following the last **GOSUB**.
RIGHT$	Extract *n* amount of characters from the right of a String. For 18F devices only.
RSIN	Asynchronous serial input from a fixed pin and baud rate.
RSOUT	Asynchronous serial output to a fixed pin and baud rate.
SEED	Seed the random number generator, to obtain a more random result.
SELECT..CASE..ENDSELECT	Conditionally run blocks of code.
SERIN	Receive asynchronous serial data (i.e. RS232 data).
SEROUT	Transmit asynchronous serial data (i.e. RS232 data).
SERVO	Control a servo motor.
SET	Place a variable or bit in a high state.
SET_OSCCAL	Calibrate the internal oscillator found on some PICmicro™ devices.
SETBIT	Set a bit of a port or variable, using a variable index.
SHIN	Synchronous serial input.
SHOUT	Synchronous serial output.
SLEEP	Power down the processor for a period of time.
SNOOZE	Power down the processor for short period of time.
SOUND	Generate a tone or white-noise on a specified pin.
SOUND2	Generate 2 tones from 2 separate pins.
STOP	Stop program execution.
STR	Load a Byte array with values.
STRN	Create a NULL terminated Byte array.
STR$	Convert the contents of a variable to a NULL terminated String.
SWAP	Exchange the values of two variables.
SYMBOL	Create an alias to a constant, port, pin, or register.
TOGGLE	Reverse the state of a port's bit.
TOLOWER	Convert the characters in a String to lower case. For 18F devices only.
TOUPPER	Convert the characters in a String to UPPER case. For 18F devices only.
TOSHIBA_COMAMND	Send a command to a Toshiba T6963 graphic LCD.
TOSHIBA_UDG	Create User Defined Graphics for Toshiba T6963 graphic LCD.
UNPLOT	Clear a single pixel on a Graphic LCD.
USBINIT	Initialise the USB interrupt on devices that contain a USB module.
USBIN	Receive data via a USB endpoint on devices that contain a USB module.
USBOUT	Transmit data via a USB endpoint on devices that contain a USB module.
VAL	Convert a NULL terminated String to an integer value.
VARPTR	Locate the address of a variable.
WHILE...WEND	Execute statements while condition is true.
XIN	Receive data using the X10 protocol.
XOUT	Transmit data using the X10 protocol.