Universidad Nacional Autónoma de México

Facultad de Ingeniería

“Desarrollo de una interfaz serial para una pantalla de cristal líquido y teclado que sean adaptables y compatibles a un módulo Field Point”

Tesis para obtener el titulo de Ingeniero Eléctrico y Electrónico

P r e s e n t a:

Gustavo Moreno Lozada

Asesor: Ing. Enrique Ramón Gómez Rosas

Ciudad Universitaria México, D.F. 2005

Agradecimientos

A mis padres, Carlos Moreno Vázquez y Yolanda Lozada Prado, por ser pilares fundamentales en todas las decisiones de mi vida.

A mi hermano, Roberto Moreno, por su confianza y apoyo hacia mi.

A Paloma Larios, por ser mi compañera en el difícil camino.

A mi hijo, Diego Aldebarán, por ser mi motivación e inspiración.

A mi Abuelo, mis tíos, tías, primos y primas, que más que solo familiares son mis amigos.

A los ingenieros Enrique Gómez y Rodolfo Peters, por su apoyo, orientación y por compartir conmigo sus conocimientos.

A la profesora Antonieta Guevara, por instruirme en el inicio de mi educación.

A mis amigos del grupo de teatro Toputshi y mis amigos de la Pandilla, por escucharme, entenderme y aceptarme; en especial a Rafael M., Germán B., Hugo H., Esteban R., Noe N., Enrique A., Jesús E., Antonio R., Iván T., Daniel L., Miguel R., Fernando P. y Sergio M.

ÍNDICE

INTRODUCCIÓN

El objetivo de la presente la tesis es lograr el diseño y elaboración de un módulo que permita usar un teclado y una pantalla LCD (Módulo Teclado Pantalla ó MTP), de 4X40 caracteres y adaptarlos a un módulo Field Point (FP) de National Instruments, así como el desarrollo del programa necesario para la comunicación entre el módulo Field Point y el módulo a construir.

Los módulos FP son muy versátiles, pero su principal desventaja es que no poseen la capacidad de despliegue en sitio, por lo que es necesario el uso de una computadora ajena al módulo, lo que implica un equipo más complejo y costoso, por lo que la elaboración del MTP propuesto facilitará en gran medida la labor del usuario. Es importante destacar que los equipos FP existentes no cuentan con ningún módulo similar al propuesto.

A lo largo del desarrollo de está tesis se tocarán aspectos como, la utilidad del MTP, así como su justificación y ventajas sobre otros módulos, se hará una enumeración de causas que propiciaron la elección de los componentes usados, así como una descripción de la elaboración del hardware, incluyendo la implementación de los dispositivos usados, de igual manera una detallada descripción de la elaboración del software utilizado.

CAPÍTULO I ANTECEDENTES

1.1 Adquisición de datos

El proceso de adquisición de datos es usado en diversos aspectos de la vida cotidiana, sencillas acciones que efectuamos diariamente representan adquisición de datos, el simple hecho de ver la hora en un reloj es la adquisición de un dato, que en este caso es la hora.

La mayoría de los proyectos de investigación necesitan datos para contestar a un problema propuesto. Los datos que necesitan ser adquiridos, y las fuentes de tales datos, se deben identificar como cuestión de importancia extrema. Ninguna cantidad o profundidad del análisis de datos subsecuente puede compensar una carencia original, la cantidad, los datos o calidad.

Los objetivos (o las hipótesis) necesitan ser construidos muy cuidadosamente y ser definidos claramente, pues dictan los datos que necesitan ser obtenidos y ser analizados. Además, la cantidad de datos, su calidad, para que se haga un muestreo y se midan, tienen implicaciones en la opción y eficacia de las técnicas del análisis de datos a usar en el análisis subsecuente.

Preguntas fundamentales.-Las preguntas fundamentales que se piden con respecto a la investigación y a los datos propuestos incluyen:

¿Qué datos son necesarios?

¿Qué datos necesitan ser medidos o ser obtenidos?

¿Los datos existen ya y pueden ser obtenidos?

¿Si es así cuáles son las fuentes de los datos?

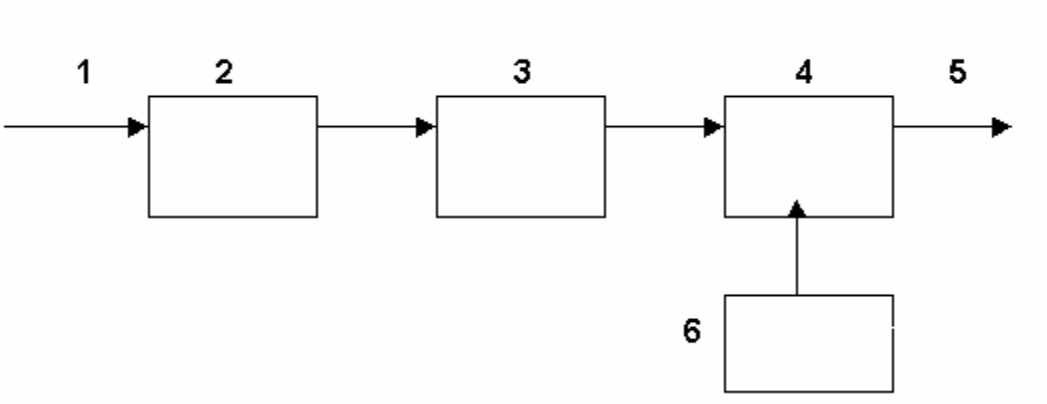
¿Cómo fueron medidos los datos?

¿Cuáles son las características de los datos en términos de su tipo, calidad, resolución, precisión, exactitud, y cobertura?

¿Es la cantidad de datos suficiente?

Introducción a la adquisición de datos.- Un sistema de adquisición de datos es un equipo que permite tomar señales físicas del entorno y convertirlas en datos que posteriormente se podrán procesar y presentar. A veces el sistema de adquisición es parte de un sistema de control, y por tanto la información recibida se procesa

para obtener una serie de señales de control. En este diagrama se pueden ver los bloques que componen un sistema de adquisición de datos:



Como se ve, los componentes principales son estos:

1.- Señal de entrada

2.- El transductor

3.- El acondicionamiento de señal

4.- El convertidor analógico-digital

5.- La etapa de salida (interfaz con la lógica)

6.- Referencia de voltaje

El transductor es un elemento que convierte la magnitud física que se va a medir en una señal de salida que puede ser procesada por el sistema. Salvo que la señal de entrada sea eléctrica, se puede decir que el transductor es un elemento que convierte energía de un tipo en otro. Por tanto, el transductor debe tomar poca energía del sistema bajo observación, para no alterar la medida.

El acondicionamiento de señal es la etapa encargada de filtrar y adaptar la señal proveniente del transductor a la entrada del convertidor analógico / digital. Esta adaptación suele ser doble y se encarga de:

Adaptar la salida del transductor con la entrada del convertidor. La adaptación entre los rangos de salida del convertidor y el de entrada del convertidor tiene como objetivo el aprovechar el margen dinámico del convertidor,

de modo que la máxima señal de entrada debe coincidir con la máxima del convertidor.

Por otro lado, la adaptación de impedancias es imprescindible ya que los transductores presentan una salida de alta impedancia, que normalmente no puede excitar la entrada de un convertidor, cuya impedancia típica suele estar entre 1 y 10 k.

El convertidor Analógico / Digital es un sistema que presenta en su salida una señal digital a partir de una señal analógica de entrada, (normalmente de tensión) realizando las funciones de cuantificación y codificación.

La cuantificación implica la división del rango continuo de entrada en una serie de pasos, de modo que para infinitos valores de la entrada la salida sólo puede presentar una serie determinada de valores. Por tanto la cuantificación implica una pérdida de información que no podemos olvidar.

La codificación es el paso por el cual la señal digital se ofrece según un determinado código binario, de modo que las etapas posteriores al convertidor puedan leer estos datos adecuadamente. Este paso hay que tenerlo siempre en cuenta, ya que puede hacer que se obtengan datos erróneos, sobre todo cuando el sistema admite señales positivas y negativas, es el momento en el cual la salida binaria del convertidor da tanto la magnitud como el signo de la tensión que ha sido medida.

La etapa de salida es el conjunto de elementos que permiten conectar el sistema de adquisición de datos con el resto del equipo, y puede ser desde una serie de buffers digitales incluidos en el circuito convertidor, hasta una interfaz RS 232, RS 485 o Ethernet para conectar a un ordenador o estación de trabajo, en el caso de sistemas de adquisición de datos comerciales.

A continuación se describen las características esenciales que se deben tener en cuenta para realizar las medidas de un modo eficiente. No se mencionaran todas, sino las más básicas. Las características que no debemos olvidar son:

Impedancia de entrada

Rango de entrada

Número de bits

Resolución
Tensión de fondo de escala
Tiempo de conversión
Error de conversión

Errores. Un convertidor no es un circuito perfecto, sino que presenta una serie de errores que debemos tener en cuenta. Algunos de los que más importancia tiene son los siguientes:

-Error de offset.-El error de offset es la diferencia entre el punto nominal de offset (cero) y el punto real de offset. Concretamente, para un convertidor A/D este punto es el punto central de todos aquellos valores de la entrada que nos proporcionan un cero en la salida digital del convertidor. Este error afecta a todos los códigos de salida por igual, y puede ser compensado por un proceso de ajuste.

-Error de cuantificación.- Es el error debido a la división en escalones de la señal de entrada, de modo que para una serie de valores de entrada, la salida digital será siempre la misma. Este valor se corresponde con el escalonado de la función de transferencia real, frente a la ideal. Cada valor digital tiene un error de cuantificación de +- ½ LSB (Bit menos significativo). Por tanto, cada código digital representa un valor que puede estar dentro del ½ LSB a partir del punto medio entre valores digitales continuos.

-Error de linealidad (linealidad integral).- Este error es la manifestación de la desviación entre la curva de salida teórica y la real, de modo que para iguales incrementos en la entrada, la salida indica distintos incrementos.

-Error de apertura.- Es el error debido a la variación de la señal de entrada mientras se está realizando la conversión. Este error es uno de los más importantes cuando se están muestreando señales alternas de una frecuencia algo elevada, (como por ejemplo el muestreo de voz) pero tiene poca importancia cuando medimos señales cuasi-continuas, como temperatura, presión, o nivel de líquidos. Para minimizar este tipo de error se usan los circuitos de muestreo y retención.

Acondicionamiento de la señal.- Con más detalle, en una etapa de acondicionamiento se pueden encontrar estas etapas, aunque no todas están siempre presentes:

Amplificación

Excitación

Filtrado

Multiplexado

Aislamiento

Linealización

Amplificación - Es el tipo más común de acondicionamiento. Para conseguir la mayor precisión posible la señal de entrada deber ser amplificada de modo que su máximo nivel coincida con la máxima tensión que el convertidor pueda leer.

Excitación - La etapa de acondicionamiento de señal a veces genera excitación para algunos transductores, como por ejemplo termistores o RTD, que necesitan de la misma, bien por su constitución interna, (como el termistor, que es una resistencia variable con la temperatura) o bien por la configuración en que se conectan.

Filtrado - El fin del filtro es eliminar las señales no deseadas de la señal que estamos observando. Por ejemplo, en las señales cuasi-continuas, (como la temperatura) se usa un filtro de ruido de unos 4 Hz, que eliminará interferencias, incluidos los 50/60 Hz de la red eléctrica. Las señales alternas, tales como la vibración, necesitan un tipo distinto de filtro, conocido como filtro antialiasing, que es un filtro pasabajo pero con un corte muy brusco, que elimina totalmente las señales de mayor frecuencia que la máxima a medir, ya que si no se eliminasen aparecerían superpuestas a la señal medida, con el consiguiente error.

Multiplexado - El multiplexado es la conmutación de las entradas del convertidor, de modo que con un sólo convertidor podemos medir los datos de diferentes canales de entrada. Puesto que el mismo convertidor está midiendo diferentes canales, su frecuencia máxima de conversión será la original dividida por el número de canales muestreados.

Aislamiento - Otra aplicación habitual en acondicionamiento de la señal es el aislamiento eléctrico entre el transductor y el ordenador, para proteger al mismo de transitorios de alta tensión que puedan dañarlo. Un motivo adicional para usar aislamiento es el garantizar que las lecturas del convertidor no son afectadas por diferencias en el potencial o por tensiones en modo común.

Linealización - Muchos transductores, como los termopares, presentan una respuesta no lineal ante cambios lineales en los parámetros que están siendo medidos. Aunque la linealización puede realizarse mediante métodos numéricos en el sistema de adquisición de datos, suele ser una buena idea el hacer esta corrección mediante circuitería externa.

1.2 Utilidad de los módulos Field Point.

El equipo FieldPoint (FP) es una tecnología recientemente desarrollada e impulsada por National Instruments (NI) la cual consiste de un módulo de red, ya sea RS-232, RS485, Ethernet o FieldBus, el cual se comunica con otros módulos los cuales incluyen acondicionadores específicos para diferentes transductores. Así pues se cuenta con módulos acondicionadores para voltaje, corriente, termopares, celdas de deformación, contadores, etc. Los módulos de acondicionamiento permiten, en general, la conexión directa con el sensor.

Adicionalmente la arquitectura del equipo permite utilizar conectores baratos los cuales permiten alambrar pruebas en prototipo y posteriormente, cuando se deba realizar la prueba, conectar toda la instrumentación en forma muy rápida.

La velocidad de muestreo de los equipos FP aunque es relativamente baja, 30 muestras por segundo, es adecuada para la mayor parte de las aplicaciones que realizan en la industria ya que estas son, desde el punto de vista de los sistemas de adquisición de datos, estáticas. Se debe recordar que éste equipo no substituye a las tarjetas de adquisición de datos ya que la velocidad de muestreo de estas normalmente llega a las 200,000 muestras por segundo, sin embargo estos equipo no tienen acondicionamiento, lo que implica que es necesario construir un acondicionador para cada uno de los canales usados, en el caso de la tecnología FP cada canal puede estar acondicionado de acuerdo al módulo utilizado.

Si se utilizan módulos de red Ethernet, del tipo inteligente, que son una computadora de estado sólido con disco duro (sólido o hay partes móviles con capacidad de hasta 32 MB para datos), lo que permite la creación de adquisidores de datos remotos, es decir pueden operar aislados guardando la información en ellos. Adicionalmente, dado que cuentan con un puerto Ethernet, se puede formar una pequeña red de adquisición de datos con lo cual el cableado de prototipos disminuye mucho, pues basta con llevar el cable Ethernet a puntos estratégicos donde se conectarán los cables de los transductores. Es importante hacer notar que se pueden interconectar más de un módulo, recordando que se debe de utilizar un concentrador con n nodos como grupos de módulos se tengan. Por otra

parte es necesario indicar que debido a que este sistema solo cuenta con un puerto Ethernet se debe de contar con una computadora con tarjeta Ethernet a fin de que sea viable la visualización de la información y la programación del mismo,

El crecimiento del equipo, al ser modular, es económico ya que si en un momento se adquiere un transductor para el cual no se cuenta con su acondicionador basta con comprar el módulo de acondicionamiento específico siendo reutilizable todo el equipo anterior.

Es importante hacer notar que el funcionamiento del equipo esta regido por un programa el cual puede ser modificado cuantas veces se requiera, por lo que para poder obtener un buen desempeño del equipo es indispensable que el instrumentista, no el usuario final, sea capacitado en el manejo de las herramientas de programación.

Entre las principales aplicaciones (De la infinidad que tiene) que se le dan al Fieldpoint, destacan:

-Control de motores

-Controladores PID

-Sistemas de adquisición de datos

-Desarrollo de aplicaciones Data Logger

1.3 Instrumentación Virtual.

La instrumentación virtual es un concepto introducido por la compañía National Instruments (2001). En el año de 1983, respondiendo al problema de crear un software que permitiera utilizar la computadora personal (PC) como un instrumento para realizar mediciones. Tres años fueron necesarios para crear la primera versión del software que permitió, de una manera gráfica y sencilla, diseñar un instrumento en la PC. De esta manera surge el concepto de instrumento virtual (VI), definido como, "un instrumento que no es real, se ejecuta en una computadora y tiene sus funciones definidas por software."

(NationalInstruments, 2001). A este software le dieron el nombre de *Laboratory Virtual Instru-ment Engineering Workbenc*h, más comúnmente conocido por las siglas LabVIEW. Apartir del concepto de instrumento virtual, se define la instrumentación virtual como un sistema de medición, análisis y control de señales físicas con un PC por medio de instrumentos virtuales. Lab-VIEW, el primer software empleado para diseñar instrumentos en la PC, es un software que emplea una metodología de programación gráfica, a diferencia de los lenguajes de programación tradicionales. Su código no se realiza mediante secuencias de texto, sino en forma gráfica, similar a un diagrama de flujo.

Lab-VIEW es un lenguaje de programación gráfica, que se ejecuta a velocidades comparables con programas compilados en C, un instrumento virtual es un módulo de software, realizado gráficamente para que parezca un instrumento físico; tiene un panel frontal que sirve como interface interactiva para entradas y salidas, un diagrama de bloque que determina la funcionalidad del VI. Una característica muy importante del LabVIEW es que, por ser conceptualmente simple, los usuarios se pueden concentrar en el contenido básico del experimento, no perdiendo tiempo en actividades menos importantes, como la recolección de datos.

Un instrumento tradicional, se caracteriza por realizar una o varias funciones específicas que no pueden ser modificadas. Un VI es una combinación de elementos de hardware y software usados en una PC, que cumple las mismas funciones que un instrumento tradicional. A diferencia de un instrumento convencional, un VI es altamente flexible y puede ser diseñado por el usuario de acuerdo con sus necesidades y sus funciones pueden ser cambiadas a voluntad modificando el programa. Estas características de los instrumentos virtuales los convierten en una herramienta didáctica muy
importante para aplicarse en el aprendizaje de los estudiantes de las ciencias naturales y de ingeniería.

Definitivamente, las mejores posibilidades para aprovechar las ventajas que ofrece la instrumentación virtual se encuentran en la implementación de sus aplicaciones. La instrumentación virtual es también una solución a los problemas

de costos y obsolescencia de los equipos de los laboratorios. Reemplazar los instrumentos tradicionales por instrumentos virtuales que se ejecutan en computadoras, permite que las funciones de los mismos vayan a la par del desarrollo de las nuevas tecnologías de las computadoras, cuyos costos siguen una tendencia decreciente.

1.4 Necesidad de despliegue en sitio.

Se entiende por despliegue en sitio al hecho de tener la posibilidad de saber que datos que se están adquiriendo en un sistema, esto puede ser en una pantalla, impreso en papel o de alguna otra forma.

Como se mencionó anteriormente los módulos FieldPoint tienen el inconveniente de no tener la capacidad de despliegue en sitio por lo cual será muy útil contar con el módulo MTP propuesto.

El despliegue en sitio es fundamental ya que sin el no se tendría la posibilidad de saber lo que se está censando, obteniendo o simplemente procesando. Está característica la presentan muchísimos de los sistemas que usamos cotidianamente, desde una calculadora hasta una computadora, la primera presenta el despliegue en sitio por medio de la pantalla con la que cuenta y la segunda despliega sus resultados en el monitor, cabe mencionar que en estos ejemplos también se cuenta con un dispositivo para ingreso de datos que es un teclado.

CAPÍTULO II ASPECTOS DE LA INTERFAZ

2.1 Aspecto técnico

El módulo MTP a realizar tendrá como característica fundamental el ser fácil de manejar además de presentar un tamaño cómodo y manejable para el usuario. Se contará con un Microcontrolador, un teclado y una pantalla de cristal líquido.

El Microcontrolador deberá estar dentro de un circuito el cual será diseñado y elaborado de acuerdo a las especificaciones del Microcontrolador, así mismo contara con las terminales para la conexión del teclado y la pantalla, mismos que también serán seleccionados de acuerdo a sus características. Se pretende que el Microcontrolador seleccionado sea económico, además de que cuente con una unidad de recepción y transmisión serial, la cual será usada para la comunicación con el módulo FieldPoint.

La pantalla deberá tener la característica de poseer resistencia a condiciones climáticas extremas, pues es muy posible que el módulo a realizar sea usado en exteriores y sometido a elevada temperatura y humedad

De igual manera el teclado a usar tendrá características de uso rudo además de ser de tamaño discreto, por lo cual se prevé que sea de 16 teclas y de conexión serial.

2.2 Aspecto económico

Una de las principales ventajas que debe presentar la realización del MTP debe ser su economía, por lo cual los componentes de este módulo deben tener una fácil adquisición y un bajo costo

El adquirir un módulo con estas características es definitivamente más caro que la realización del mismo, para lo cual se deberán seleccionar cuidadosamente los componentes, es pertinente mencionar que la variación de existencia y precio en el mercado es enorme, lo cual presenta una ventaja ya que de está forma se tendrán más opciones.

En cuanto al circuito impreso en el cual se localizará el Microcontrolador y las conexiones de teclado y de pantalla, será por economía más conveniente fabricarlo, de igual forma en el caso del teclado es una opción viable su realización, ya que esta no implica gran trabajo, pero eso se contemplará en el momento de la selección.

En este aspecto económico hay que tener en cuenta el mercado de adquisición de los componentes, ya que si se adquieren en el mercado extranjero se presenta el inconveniente del tiempo de entrega, además de un posible incremento en el costo por efecto de importación, por otro lado si se obtienen dentro del mercado nacional se cuenta con la ventaja de obtener el componente de manera rápida y es posible que sea más barato

2.3 Justificación

La necesidad de realización de un módulo MTP de estas características responde a que los módulos FieldPoint no poseen un dispositivo de estas características y como se vio anteriormente es necesario contar con un dispositivo de despliegue en sitio, que en éste caso es la pantalla y que además contara con un práctico teclado

2.4 Ventajas sobre otros módulos

El MTP a elaborar en este proyecto presenta diversas ventajas sobre módulos ya existentes, entre las cuales destaca tal vez la más importante que es el bajo costo de realización de este, pues se espera contar con componentes de bajo costo y que al hacer una evaluación general de costo, el de este módulo sea inferior al de otros ya existentes.

También como cuestión innovadora se contará con una pantalla de 4X40 caracteres y un teclado de 16 posiciones, lo cual no es para nada una combinación común.

Habiendo repasado las ventajas económicas y técnicas, vale la pena mencionar que el software del Microcontrolador que controlará el teclado y la pantalla se diseñara especialmente para esta tesis y se procurará sea sencillo, eficiente y fácil de entender.

CAPÍTULO III SELECCIÓN DE LOS COMPONENTES A UTILIZAR

3.1 Análisis de existencia en el mercado internacional

Los diversos componentes a utilizar en el desarrollo del MTP pueden ser fácilmente encontrados en el mercado internacional, pues además de que son componentes sencillos, el mercado internacional ofrece una gran variedad de ellos.

Cabe destacar que el mercado internacional, como ya se mencionó posee la ventaja de ofrecer muchas opciones para la adquisición de los componentes a usar, en esta sección se ofrecerá un panorama de dichas opciones.

Al mencionar componentes, nos referimos el Controlador, las pantallas, al teclado y el Microcontrolador, a continuación se hará el análisis de existencia por componente.

Controlador:

Desde el inicio del proyecto se definió que el procesador con el que se va a trabajar es el FieldPoint de National Instruments, por lo que a continuación se presenta una descripción; cabe mencionar que se coloco esta información en la sección de existencia en el mercado internacional.

Primero seleccionar el modo de comunicación entre los módulos FieldPoint y el computador entre Ethernet (ítem 1), Serial RS-232 (ítem 2) y RS-485 (ítem 3).

MODULOS DE RED:

Item	Modelo	Descripción
1	777792-20	FP-2000, LabVIEW RT Módulo Red
2	777517-00	FP-1000 RS-232/RS-485 Módulo Red
3	777517-01	FP-1001 RS-485 Módulo Red

El FP-2000 (ítem 1) tiene la ventaja adicional de trabajar con tecnología LabVIEW RT, es decir puede funcionar independientemente de un computador y en Tiempo Real (RT).

Se pueden escoger hasta 9 módulos de entrada o salida para cada módulo de red. Dependiendo de las señales que maneje los módulos son los siguientes:

Item	Modelo	Descripción
4	777518-110	FP-AI-110 8-Ch. Módulo FieldPoint de entrada análoga de 16 bits; requiere terminales base FP-TB-1 o FP-TB-2
5	777518-120	FP-TC-120 8-Ch. Entrada termopar Módulo FieldPoint de entrada análoga para termopares; requiere FP-TB-1, FP-TB-2
6	777518-122	FP-RTD-122 8-Ch. Módulo FieldPoint de entrada análoga para RTD de 16 bits; requiere terminales base FP-TB-1 o FP-TB-2
7	777518-124	FP-RTD-124 Módulo Field Point con entrada análoga para RTD de 4 cables y 8 canales
8	777518-140	FP-SG-140 Módulo de entrada de 8 canales Strain Gauge
9	777518-210	FP-AO-210, Módulo de salida de 8 canales de voltaje
10	777518-200	FP-AO-200 Módulo de salida 8-Ch. 4-20 mA
11	777518-330	FP-DI-330 Módulo entrada universal discreta para 5-240 VDC
12	777518-400	FP-DO-400 Módulo de salida discreta
13	777518-420	FP-RLY-420 8-Ch. Módulo de salida SPST Relay
14	777518-510	FP-QUAD-510, Módulo de salida codificada de 4 ejes

Por cada módulo de entrada o salida se debe colocar una base terminal con los conectores respectivos. Además para el módulo de red se requiere de una fuente de

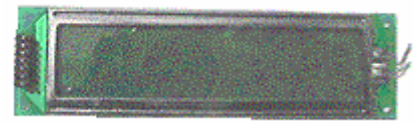
alimentación regulada de 24 V. Los ítems 17 y 18 son los drivers respectivos para FieldPoint utilizando LabVIEW junto con la documentación respectiva.

Item	*Modelo*	*Descripción*
15	***77751901***	***FP-TB-1 Terminales base universal***
16	***77758401***	***PS-2 fuente de poder, 24 VDC, 0.8 A, US 120 VAC***
17	***77752001***	***FieldPoint RS-232/485 Documentación y software que incluyeespecificaciones del FP***
18	***77752003***	***FieldPoint Documentación y software sobre LabVIEW RT***

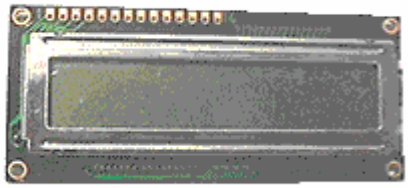
Pantallas:

A continuación se presenta un panorama de la existencia de algunas pantallas de cristal líquido.

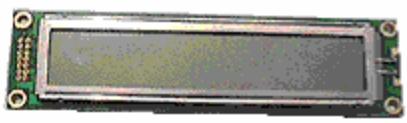
Marca Data Vision
Modelo 16278-S1MTLY



Marca Seiko
Modelo L1642el



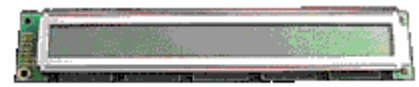
Marca Seiko
Modelo L1651



Marca Seiko
Modelo L1671 BIP

Marca Seiko

Modelo L4042BIL



Marca Seiko

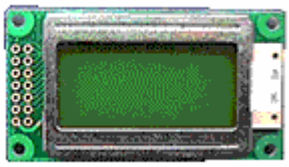
Modelo L4052

Marca Seiko

Modelo M1632

Marca Sanyo

Modelo NDM082-NANOX-TW-2294



Marca TIANMA

Modelo TM404ABC

Teclados:

La existencia de teclados de 16 posiciones que cumplan con los requerimientos que busca el proyecto, no es tan abundante como se quisiera, pues solo se encontraron los siguientes.

Marca Jameco
Modelo 196171

Marca Jameco
Modelo 152063

Microcontrolador:

La familia Microchip tiene la ventaja de ser internacional, ya que la mayoría de sus productos (En este caso los PIC) están presentes en muchos países del mundo, por lo cual se puede decir que la existencia de éstos componentes en el mercado nacional e internacional es prácticamente la misma, salvo una variación en precio, la cual es muy bien compensada adquiriendo el producto en el mercado nacional, pues hay un considerable ahorro en el gasto de importación, de esta manera consideraremos la misma existencia en ambos mercados y dejaremos su análisis y descripción para la sección del mercado nacional.

3.2 Análisis de existencia en el mercado nacional

Los componentes a utilizar, aunque en teoría son fáciles de obtener, pueden presentar dificultades para conseguirlos en el mercado nacional por lo que se tendría que recurrir al mercado internacional, cuestión que no va de acuerdo con la intención del proyecto, pues se decidió dar mas peso al mercado nacional, pero tomando en cuenta la falta de existencia existe alta probabilidad de que se tenga que recurrir al recurso de importación o bien fabricación de los componentes necesarios.

Al igual que en la sección anterior se hará un análisis de existencia por componente en el mercado nacional.

Controlador:

Se puede considerar la opción de recurrir a la importación de este dispositivo. O bien contactar a un distribuidor en México.

Pantallas:

De manera similar al módulo FieldPoint, la existencia de pantallas de cristal líquido que cumplan con las necesidades de tamaño en el mercado nacional es muy escasa, pero si existen y su adquisición no implica gran problema.

Teclado:

En lo que respecta al teclado de 16 posiciones, como se observo en el análisis de existencia de mercado internacional, la abundancia es escasa y en el caso de el mercado nacional es nula, por lo que se recurrirá a la importación de esté componente o bien a su fabricación, pues hay que tener en cuenta que a diferencia de una pantalla, la elaboración de un teclado es algo relativamente sencillo y tal vez sea más económico.

Microcontrolador:

La existencia de los diferentes tipos de PIC es abundante en México y se pueden hallar estos componentes en muchas tiendas de electrónica. Esta familia, desarrollada por la casa Microchip, se divide en cuatro gamas, gama enana, baja, media y alta. Las principales diferencias entre estas gamas radica en el número de instrucciones y su longitud, el número de puertos y funciones, lo cual se refleja en el encapsulado, la complejidad interna y de programación, y en el número de aplicaciones.

- Gama baja o gama enana **PIC12CXXX**, de 8 patas.

Su principal característica es su reducido tamaño, al disponer todos sus componentes de 8 patas. Se alimentan con un voltaje de corriente continua comprendido entre 2,5 V y 5,5 V, y consumen menos de 2 mA cuando trabajan a 5 V y 4 MHz. El formato de sus instrucciones puede ser de 12 o de 14 bits y su repertorio es de 33 o 35 instrucciones, respectivamente.

Aunque los PIC enanos sólo tienen 8 patas, pueden destinar hasta 6 como líneas de E/S para los periféricos porque disponen de un oscilador interno R-C, lo cual es una de sus principales características.

- Gama baja o básica: **PIC16C5X** con instrucciones de 12 bits.

Se trata de una serie de PIC de recursos limitados, pero con una de la mejores relaciones costo/prestaciones. Sus versiones están encapsuladas con 18 y 28 patas y pueden alimentarse a partir de una tensión de 2,5 V, lo que les hace ideales en las aplicaciones que funcionan con pilas teniendo en cuenta su bajo consumo (menos de 2 mA a 5 V y 4 MHz). Tienen un repertorio de 33 instrucciones cuyo formato consta de 12 bits. No admiten ningún tipo de interrupción y la Pila sólo dispone de dos niveles.

Al igual que todos los miembros de la familia PIC16/17, los componentes de la gama baja se caracterizan por poseer los siguientes recursos: Sistema “Power On Reset”, Perro guardián (Watchdog o WDT), Código de protección, Sep,etc.

-Gama media. **PIC16CXXX** con instrucciones de 14 bits

Es la gama más variada y completa de los PIC. Abarca modelos con encapsulado desde 18 patas hasta 68, cubriendo varias opciones que integran abundantes periféricos.

En esta gama sus componentes añaden nuevas prestaciones a las que poseían los de la gama baja, haciéndoles más adecuados en las aplicaciones complejas. Admiten interrupciones, poseen comparadores de magnitudes analógicas, convertidores A/D, puertos serie y diversos temporizadores.

El repertorio de instrucciones es de 35, de 14 bits cada una y compatible con el de la gama baja. Sus distintos modelos contienen todos los recursos que se precisan en las aplicaciones de los microcontroladores de 8 bits. También dispone de interrupciones y una Pila de 8 niveles que permite el anidamiento de subrutinas

- Gama alta: **PIC17CXXX** con instrucciones de 16 bits.

Se alcanzan las 58 instrucciones de 16 bits en el repertorio y sus modelos disponen de un sistema de gestión de interrupciones vectorizadas muy potente. También incluyen variados controladores de periféricos, puertos de comunicación serie y paralelo con elementos externos, un multiplicador por hardware de gran velocidad y mayores capacidades de memoria, que alcanza los 8 k palabras en la memoria de instrucciones y 454 bytes en la memoria de datos.

Quizás la característica más destacable de los componentes de esta gama es su arquitectura abierta, que consiste en la posibilidad de ampliación del microcontrolador con elementos externos. Para este fin, las patas sacan al exterior las líneas de los buses de datos, direcciones y control, a las que se conectan memorias o controladores de periféricos. Esta facultad obliga a estos componentes a tener un elevado número de patas comprendido entre 40 y 44.

Esta filosofía de construcción del sistema es la que se empleaba en los microprocesadores y no suele ser una práctica habitual cuando se emplean microcontroladores.

3.3 Análisis económico comparativo

Analizar los diferentes precios dentro del mercado nacional e internacional de los componentes a utilizar, es fundamental, pues dentro del objetivo general del proyecto esta contemplado que el MTP sea lo más económico y eficiente que sea posible, de está manera, teniendo en cuenta la variedad existente en ambos mercados, se hará un análisis de los diversos precios que tienen dichos componentes.

Controlador:

La existencia de los módulos FieldPoint y sus aditamentos apareció anteriormente, ahora se especifican sus precios, cabe destacar que la adquisición de este componente no será adquirido, pues el Instituto de Ingeniería lo proporcionara, los precios son solo para dar u panorama.

Módulos de red:

Item	*Modelo*	*Descripción*	*p.u. U.S.D.*
1	*777792-20*	*FP-2000, LabVIEW RT Módulo Red*	*1,314.00*
2	*777517-00*	*FP-1000 RS-232/RS-485 Módulo Red*	*522.00*
3	*777517-01*	*FP-1001 RS-485 Módulo Red*	*522.00*

Accesorios:

Ítem	*P/n*	*Descripción*	*U.S.D.*
15	*777519-01*	*FP-TB-1 Terminales base universal*	*126.00*
16	*777584-01*	*PS-2 fuente de poder, 24 VDC, 0.8 A, 120 VAC*	*90.00*

Pantallas:

A continuación se presenta un panorama de los precios de las pantallas de cristal líquido ilustradas anteriormente.

Item	*Marca*	*Modelo*	*Precio*
1	*Data Vision*	*16278-S1MTLY*	*$35.00 U.S.D.*
2	*Seiko*	*L1642el*	*$36.00 U.S.D.*
3	*Seiko*	*L1651*	*$30.00 U.S.D*
4	*Seiko*	*L1671 BIP*	*$26.00 U.S.D.*
5	*Seiko*	*L4042BIL*	*$65.00 U.S.D.*
6	*Seiko*	*L4052*	*$60.00 U.S.D.*
7	*Seiko*	*M1632*	*$26.00 U.S.D.*
8	*Sanyo*	*NDM082-NANOX-TW-2294*	*$38.00 U.S.D.*
9	*Tianma*	*TM404ABC*	*$87.00 U.S.D*

Teclados:

Los precios de teclados de 16 posiciones que se encontraron son los siguientes

Item	*Marca*	*Modelo*	*Precio*
1	*Jameco*	*196171*	*$14.95 U.S.D.*
2	*Jameco*	*152063*	*$16.95 U.S.D.*

Microcontrolador:

Los precios de los distintos modelos del PIC tienen variación en cada gama, a continuación se muestra la variación aproximada en cada gama.

Item	*Gama*	*Modelo*	*Precio Entre*
1	*Enana*	*12CXXX*	*$.95 y $2.05 U.S.D*
2	*Baja*	*16C5X*	*$1.05 y $4.75 U.S.D.*
3	*Media*	*16CXXX*	*$1.2 y $8.7 U.S.D.*
4	*Alta*	*17CXXX*	*$6.31 y $12.2 U.S.D.*

3.4 Análisis funcional comparativo

Es fundamental que los componentes a usar posean además de un precio razonable, un satisfactorio cumplimiento de las características necesarias y que su funcionalidad sea compatible con los fines buscados en éste proyecto, a continuación se presenta un panorama de las características de los componentes analizados.

Controlador:

Las características de cada uno de los FieldPoint analizados aparecen en la sección de existencia.

Pantallas:

A continuación se presenta un panorama de las características de las pantallas de cristal líquido analizadas anteriormente.

Item	*Marca*	*Modelo*	*Características*
1	*Data Vision*	*16278-S1MTLY*	*2X16 Caracteres con backlight*
2	*Seiko*	*L1642el*	*2X16 Caracteres con backlight*

3	***Seiko***	***L1651***	***1X16 Caracteres con backlight Controlador HD44780***
4	***Seiko***	***L1671 BIP***	***1X6 Caracteres con backlight Controlador KSD066F0D***
5	***Seiko***	***L4042BIL***	***2X40 Caracteres con backlight Controlador HD44780***
6	***Seiko***	***L4052***	***2X6 Caracteres con backlight Controlador HD44780***
7	***Seiko***	***M1632***	***2X6 Caracteres con backlight Controlador HD44780***
8	***Sanyo***	***NDM082NANOX-TW-2294***	***2X8 Caracteres con backlight Controlador LC7985NA***
9	***Tianma***	***TM404ABC***	***4X40 Caracteres con backlight Controlador S6A0069X01-Q0RJ***

Teclado:

Los dos teclados encontrados poseen las mismas características que son:

Item	***Marca***	***Modelo***	***Características***
1	***Jameco***	***196171 y 152063***	***16 posiciones, Matriz de 4X4 y 2X8 respectivamente y Salida serial***

Microcontrolador:

Las características generales las diferentes gamas del PIC se mostraron en el análisis de existencia, pero ahora se mostraran tablas con las características delos diferentes componentes de cada gama

-Gama enana

Modelo	Memoria Programa	Memoria Datos	Frecuencia Maxima	Lineas E / S	Temporizadores	Pines
PIC12C509	1024x12	41x8	4 MHz	6	TMR0 + WDT	8
PIC12C670	512x14	80x8	4 MHz	6	TMR0 + WDT	8
PIC12C671	1024x14	128x8	4 MHz	6	TMR0 + WDT	8
PIC12C672	2048x14	128x8	4 MHz	6	TMR0 + WDT	8

-Gama baja

Modelo	Memoria Programa	Memoria Datos	Frecuencia Maxima	Lineas E / S	Temporizadores	Pines
PIC16C52	384	25	4MHz	4	TMR0 + WDT	18
PIC16C54	512	25	20MHz	12	TMR0 + WDT	18
PIC16C54A	512	25	20MHz	12	TMR0 + WDT	18
PIC16CR54	512	25	20MHz	12	TMR0 + WDT	18
PIC16C55	512	24	20MHz	20	TMR0 + WDT	28
PIC16C56	1K	25	20MHz	12	TMR0 + WDT	18
PIC16C57	2K	72	20MHz	20	TMR0 + WDT	28
PIC16C57B	2K	72	20MHz	20	TMR0 + WDT	28
PIC16C58A	2K	73	20MHz	12	TMR0 + WDT	18

-Gama media

Modelo	Memoria Programa	Memoria Datos	Registros Especificos	Lineas E / S	Temporizador	Rango Voltaje	Pines
PIC16C84	1K	36	11	13	TMR0 + WDT	2 - 6	18
PIC16F84	1K	68	11	13	TMR0 + WDT	2 - 6	18
PIC16F83	512	36	11	13	TMR0 + WDT	2 - 6	18
PIC16R84	1K	68	11	13	TMR0 + WDT	2 - 6	18
PIC16R83	512	36	11	13	TMR0 + WDT	2 - 6	18

-Gama alta

Modelo	Memoria Programa	Memoria Datos	Registros Especificos	Lineas E / S	Temporizador	PWM	Pines
PIC17C42	2K	232	48	33	4 + WDT	2	40 a 44
PIC17C43	4K	454	48	33	4 + WDT	2	40 a 44
PIC17C44	8K	454	48	33	4 + WDT	2	40 a 44
PIC17C752	8K	454	76	50	4 + WDT	3	64 a 65
PIC17C756	16K	902	76	50	4 + WDT	3	64 a 65

3.5 Análisis de resistencia al ambiente

Para efectos del presente proyecto, resulta de fundamental importancia que los componentes del MTP posean una alta resistencia a los efectos ambientales, puesto que como se mencionó el módulo podrá ser para uso rudo y lugares donde seguramente estará expuesto a condiciones inestables de temperatura, humedad, etc. Por lo que para hacer la selección de los dispositivos a usar se tendrá que tener en cuenta que las características de ellos trabajen óptimamente, aun en condiciones no ideales (No necesariamente).

A continuación se presentan las características de resistencia ambiental de los componentes seleccionados

Controlador:

Las especificaciones de resistencia al ambiente de los módulos FieldPoint en general son:

Parámetro	Rango
Temperatura de operación	-40° a 70°C
Temperatura extrema	-55° a 90°C
Humedad	5% a 90%
Alimentación	11VDC a 30VDC

Pantallas:

La única pantalla de la que pudo obtener especificaciones de resistencia al ambiente fue de la pantalla de TIANMA modelo TM404ABC y estas son:

Parametro	***Rango***
Temperatura de operación	***0° a 50°C***
Temperatura extrema	***-20° a 60°C***
Humedad	***10% a 80%***
Alimentación	***4.5VDC a 5.5VDC***
Presión atmosférica Máxima	***40Kpa***

Teclado:

Las características de resistencia al ambiente de los teclados de la marca Jameco son:

Parametro	***Rango***
Temperatura de operación	***-20° a 60°C***
Temperatura extrema	***-30° a 70°C***
Alimentación máxima	***25VDC***
Barrido	***3MHz X Tecla***

Microcontrolador:

Las especificaciones de resistencia ambiental para las 4 gamas del PIC no varían mucho entre si y en general son:

Parametro	***Rango***
Temperatura de operación	***-55° a 125°C***
Temperatura extrema	***-60° a 150°C***
Alimentación	***4VDC a 6VDC***

3.6 Selección de un Controlador

El Controlador con el que se trabajara será el Compact FieldPoint cFP- 2020, LabVIEW RT Módulo Red de National Instruments y el módulo de entrada FP-AI-100, que cumple bastante bien con los requerimientos técnicos y presenta una buna resistencia ambiental , de está manera se trabajará con el Compact FieldPoint 2020, cuyo análisis aparece en el capítulo 5 en el cual se indica el desarrollo de las librerías que se le implementaron y las herramientas con las cuales se logro esto.

3.7 Selección de la pantalla de cristal líquido

Una vez que se ha tomado en cuenta la existencia de las pantallas en el mercado nacional e internacional, así como sus diferentes precios, funcionalidad y condiciones de resistencia ambiental, se tomó la decisión de trabajar con la pantalla de la marca TIANMA modelo TM404ABC, que como se vio anteriormente cuenta con un display de 4X40 caracteres, un precio que no es el más económico, pero es accesible y condiciones de resistencia ambiental optimas.

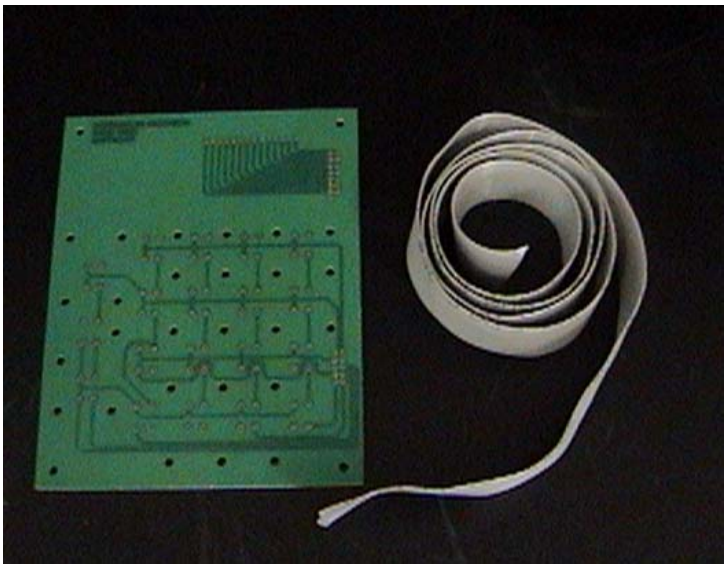
3.8 Selección del teclado

De igual manera que en el caso anterior tomando en cuenta las diferentes características de los teclados en principio se decidió trabajar con el teclado de la marca Jameco modelo 152063 que cuenta con una matriz de teclas de 2X8, pero debido a que el tiempo de entrega era demasiado prolongado, se tomó la decisión de elaborar uno mediante la realización de un circuito impreso, la implementación de los botones y la colocación del cable, lo cual no presenta mayor complicación

De está manera se empezó por diseñar el circuito impreso sobre el que se habría de montar el teclado, esto se realizó con un programa sencillo de computadora llamado Tango, que como ventaja principal tiene un sencillo uso, cabe mencionar que existen otros programas para desarrollar circuito impresos tales como Orcad, Circuit Maker, etc. Pero se

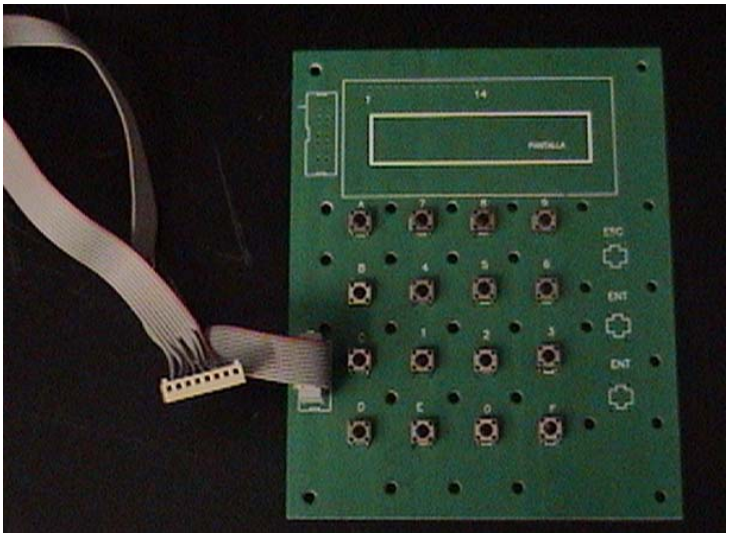
consideró que éste presentaba mayor facilidad en su manejo y para la sencillez del circuito a diseñar, cumplía muy bien las necesidades.

Así pues, después de diseñar el circuito y someterlo a un proceso en el cual se consiguió la tablilla de cobre, se le monto el diseño de pistas y se le sometió al proceso de implantación de pistas mediante su introducción a cloruro férrico, quedo el siguiente circuito, en la imagen se ilustra también el cable que será usado para su conexión



Después de elaborar el circuito impreso se procedió a una sencilla prueba de continuidad con el multímetro y luego a colocar los botones y la base para la conexión del cable, también se adecuo el cable para los fines de éste trabajo, éste fue un cable plano de 8 hilos con un conector doble en un extremo para el teclado y un molex serial en el otro extremo para el circuito de control.

De esta manera el teclado quedo:



3.9 Selección del Microcontrolador

Los diferentes PICs que existen en el mercado tienen enormes diferencias entre sí en cuanto a funcionalidad y precio, pues en características de resistencia al ambiente son muy similares.

Se decidió trabajar con el PIC 16C65B debido a que su precio es relativamente bajo y cumple bastante con los fines para los que se requiere, entre los cuales destaca el contar con un puerto serial (USART), cuenta además con puertos de entrada y salida y un conjunto de instrucciones corto, pero que tiene las necesarias para la elaboración del programa que habrá de controlar el teclado y la pantalla.

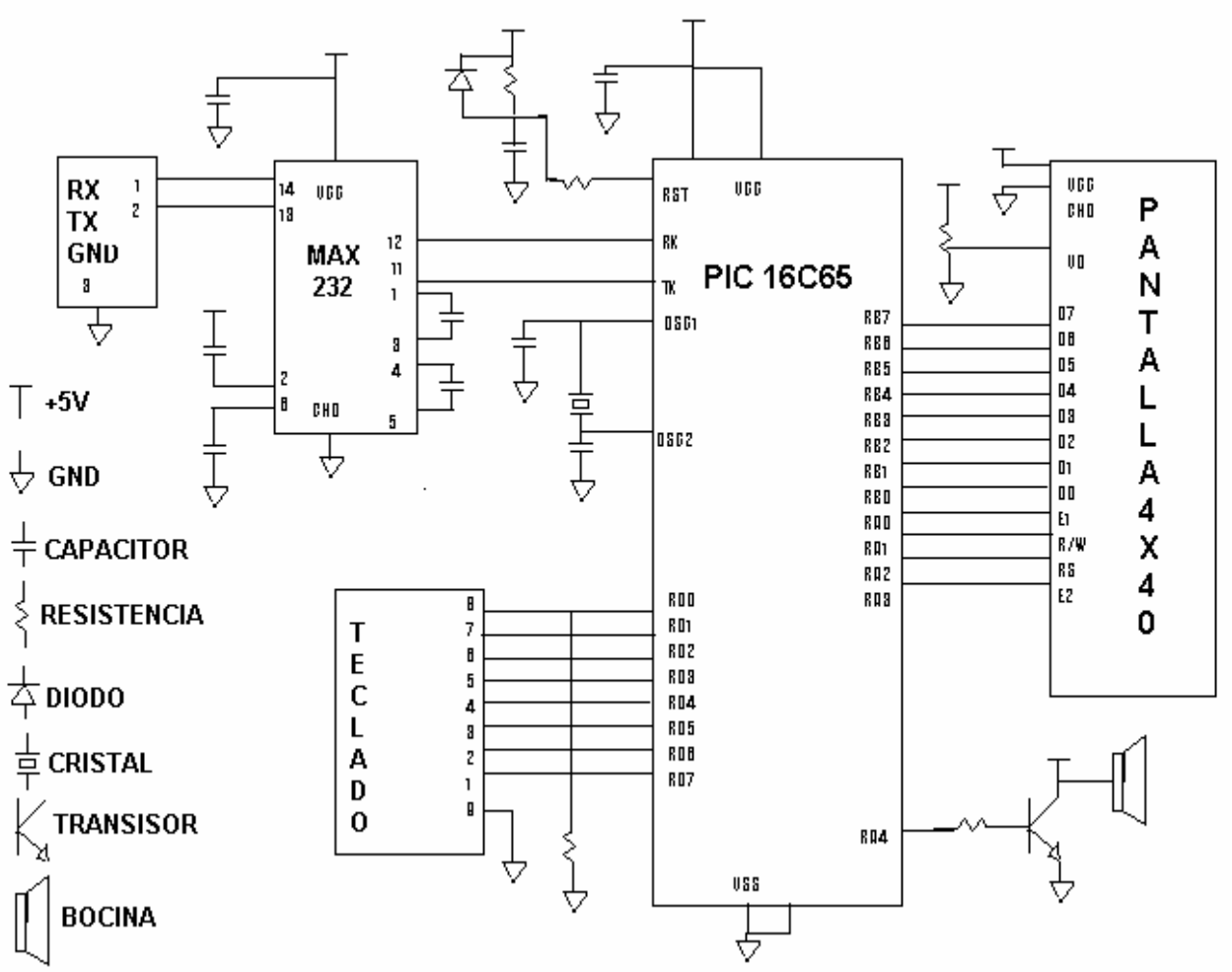
Cabe mencionar que se pudo elegir otro PIC de la misma gama u otra superior pero para los fines buscados con el elegido es suficiente, pues otro con características más avanzadas seria un desperdició.

CAPÍTULO IV ESPECIFICACIÓN DE LOS PROCESOS A REALIZAR

4.1 Proceso de Hardware

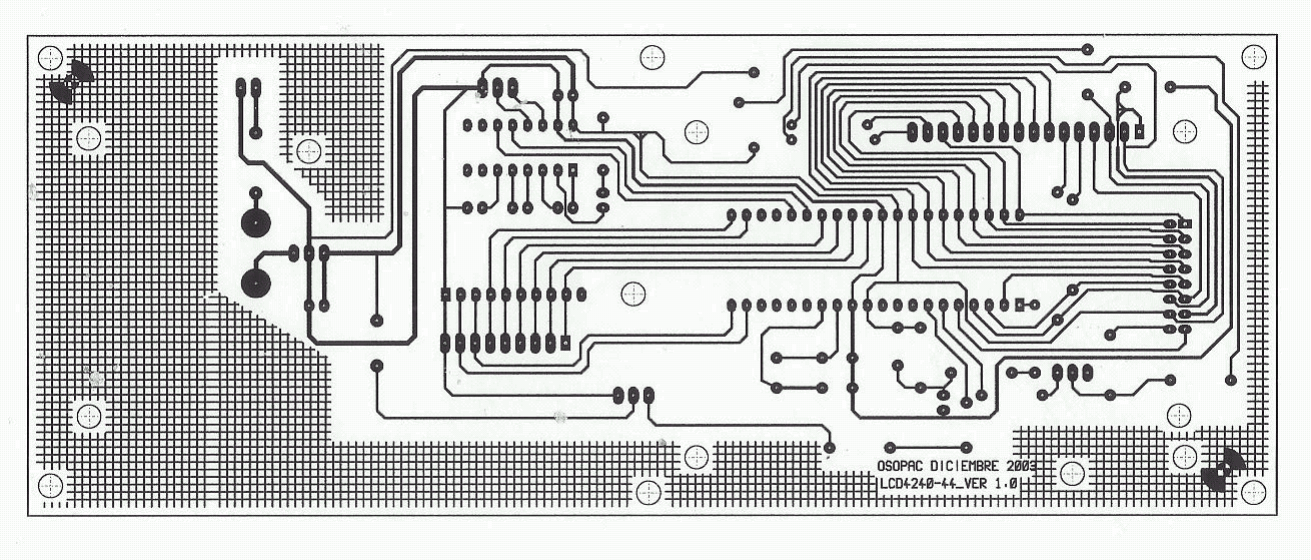
Una vez seleccionados los componentes a utilizar para la elaboración del MTP, en el caso del teclado se menciona que fue elaborado, la pantalla elegida TM404ABC fue adquirida, al igual que el PIC16C65B y el Compact FieldPoint, se procedió a la elaboración del circuito impreso sobre el cual se colocaría el PIC y las terminales tanto del teclado, como de la pantalla.

Este circuito cuenta además con un regulador de voltaje de forma que se polariza con 10VDC aproximadamente y reduce los voltajes a 5 VDC, para la alimentación del PIC, la pantalla y el teclado, resistencias de polarización y capacitores de acoplamiento, después de ser diseñado el diagrama para el circuito quedo así:

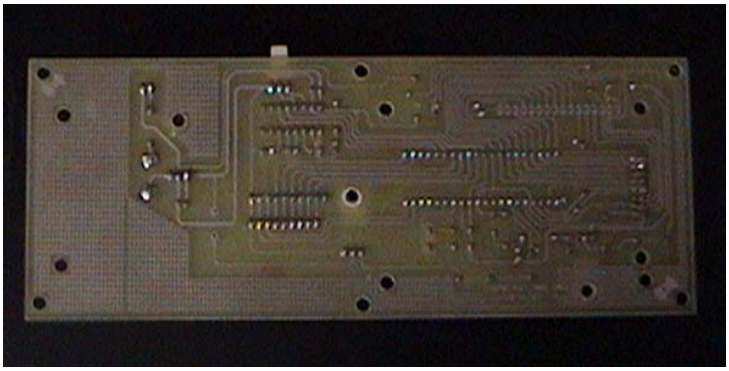


4.1.1 Diseño y elaboración del circuito impreso

El circuito impreso sobre el cual se montaría el PIC y las terminales fue diseñado con base en el diagrama anterior, tomando en cuenta la distribución de los dispositivos, de igual manera que en el caso del teclado, esté circuito impreso fue diseñado en Tango que cómo ya se mencionó es un programa sencillo de usar cuya imagen es:



Después de su diseño se procedió a su realización la que consistió en la adquisición de la tabilla de cobre, montaje sobre ella del diseño en tango y finalmente se le sometió al cloruro férrico para quedar finalmente así:



4.1.2 Implementación de los componentes en el circuito

Los componentes que formarán parte del circuito son:

-PIC16C65B

-MAX232N

-2 Transistores TIP120

-4 Capacitores de 10uf a 16V

-Capacitor de 1uf a 16V

-Capacitor de 47uf a 16V

-Cristal X-TAL de 16MHz

-Diodo IN4001

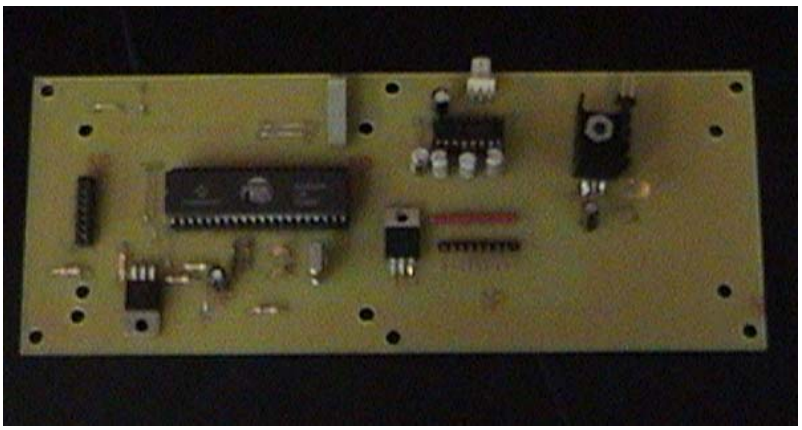
-Diodo zener a 4.5V

-Potenciómetro de 10K

-Resistencias de 1.5K, 57K, 790, 33, 47 y una de línea de 1K

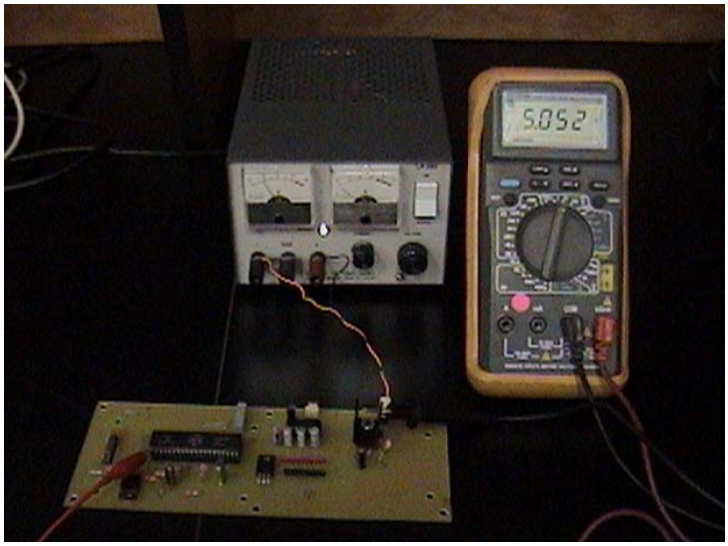
-Postes para la conexión del teclado, pantalla, cable serial y cable de alimentación

Todos los dispositivos anteriores son de fácil adquisición y de un costo relativamente bajo, de acuerdo a la disposición del circuito impreso se procedió a la implementación de éstos dispositivos, cosa que fue sencilla y después de terminar de soldar, la imagen final del circuito es:



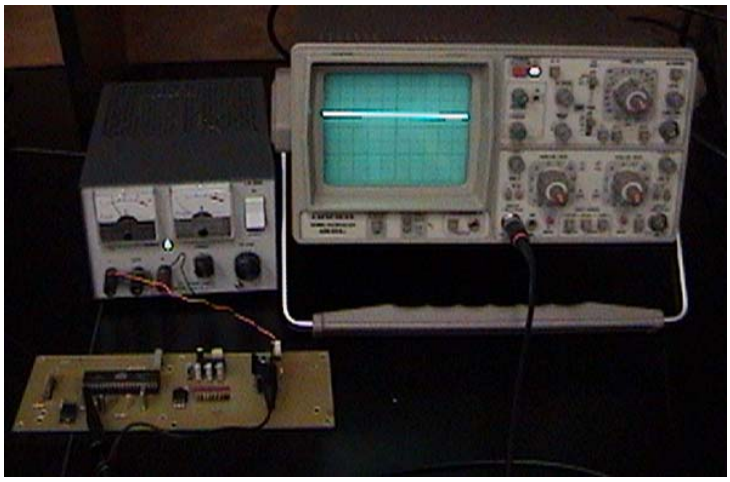
4.1.3 Revisión del circuito

Una vez armado el circuito impreso se realizaron sencillas pruebas a éste entre las cuales destacan, la prueba de voltaje en los diferentes puntos del circuito, misma prueba que es fácilmente realizable con el multimetro



Como se observa en la imagen el nivel de voltaje de la pata 1 del PIC (Que corresponde al voltaje de alimentación) registra los 5 VDC antes mencionados, por lo que se puede dar la prueba de voltaje por satisfactoria.

También se realizó una prueba de oscilación del PIC, usando el Osciloscopio, como se vio en el primer diagrama, el circuito cuenta con un cristal oscilador a 16 MHz, por lo cual es de esperarse que la oscilación del PIC sea también a esta frecuencia



Después de colocar el osciloscopio en la escala adecuada de voltaje y frecuencia y de colocar la punta de prueba en la pata indicada, se verificó que la oscilación era casi exacta a 16 MHz, por lo que también se considera la prueba de oscilación cómo satisfactoria.

CAPÍTULO V DESARROLLO DEL SOFTWARE NECESARIO PARA EL MICROCONTROLADOR

5.1 Breve repaso del lenguaje ensamblador

Todo procesador, grande o pequeño, desde el de una calculadora hasta el de una supercomputadora, ya sea de propósito general o específico, posee un lenguaje único que es capaz de reconocer y ejecutar. Por razones que resultan obvias, este lenguaje ha sido denominado Lenguaje de Máquina y más que ser propio de un computador pertenece a su microprocesador. El lenguaje de máquina está compuesto por una serie de instrucciones, que son las únicas que pueden ser reconocidas y ejecutadas por el microprocesador. Este lenguaje es un conjunto de números que representan las operaciones que realiza el microprocesador a través de su circuitería interna. Estas instrucciones, por decirlo así, están grabadas o "alambradas" en el hardware y no pueden ser cambiadas. El nivel más bajo al que se puede aspirar a llegar en el control de un microprocesador es precisamente el del lenguaje de máquina.

La importancia del lenguaje ensamblador radica principalmente que se trabaja directamente con el microprocesador; por lo cual se debe de conocer el funcionamiento interno de este, tiene la ventaja de que en el se puede realizar cualquier tipo de programas que en los lenguajes de alto nivel no lo pueden realizar. Otro punto sería que los programas en ensamblador ocupan menos espacio en memoria.

Ventajas del Lenguaje Ensamblador

1. Velocidad.- Como trabaja directamente con el microprocesador al ejecutar un programa, pues como este lenguaje es el más cercano a la máquina la computadora lo procesa más rápido que con el uso de otro lenguaje.
2. Eficiencia de tamaño.- Un programa en ensamblador no ocupa mucho espacio en memoria porque no tiene que cargan librerías, como en los lenguajes de alto nivel en los cuales si son necesarias otras cuestiones.
3. Flexibilidad.- Es flexible porque todo lo que puede hacerse con una máquina, puede hacerse en el lenguaje ensamblador de esta máquina; los lenguajes de alto nivel tienen en una u otra forma limitantes para explotar al máximo los recursos de la máquina. O sea que en lenguaje ensamblador se pueden hacer tareas especificas que en un lenguaje de alto nivel no se pueden llevar acabo porque tienen ciertas limitantes que no se lo permiten.

Desventajas del Lenguaje Ensamblador

1. Tiempo de programación.- Como es un lenguaje de bajo nivel requiere más instrucciones para realizar el mismo proceso, en comparación con un lenguaje de alto nivel. Por otro lado, requiere de más cuidado por parte del programador, pues es propenso a que los errores de lógica se reflejen más fuertemente en la ejecución.

2. Programas fuente grandes.- Por las mismas razones que aumenta el tiempo, crecen los programas fuentes; Simplemente se requieren más instrucciones primitivas para describir procesos equivalentes. Esto es una desventaja porque dificulta el mantenimiento de los programas, y nuevamente reduce la productividad.

3. Peligro de afectar recursos inesperadamente.- Que todo error que se pueda cometer, o con todo riesgo que se pueda tener, se pueden afectar los recursos de la maquina, programar en este lenguaje lo más común que pueda pasar es que la máquina se bloquee o se reinicie. Porque con este lenguaje es perfectamente posible (y sencillo) realizar secuencias de instrucciones inválidas, que normalmente no aparecen al usar un lenguaje de alto nivel.

4. Falta de portabilidad.- Porque para cada máquina existe un lenguaje ensamblador; por ello, evidentemente no es una selección apropiada de lenguaje cuando se desea codificar en una máquina y luego llevar los programas a otros sistemas operativos o modelos de computadoras.

Registros.-Hay nueve indicadores de un bit en este registro de 16 bits. Los cuatro bits más significativos están indefinidos, mientras que hay tres bits con valores determinados: los bits 5 y 3 siempre valen cero y el bit 1 siempre vale uno.

CF (Carry Flag, bit 0): Si vale 1, indica que hubo "arrastre" (en caso de suma) o "préstamo" (en caso de resta). Este indicador es usado por instrucciones que suman o restan números que ocupan varios bytes. Las instrucciones de rotación pueden aislar un bit de la memoria o de un registro.

PF (Parity Flag, bit 2): Si vale uno, el resultado tiene paridad par, es decir, un número par de bits a 1. Este indicador se puede utilizar para detectar errores.

AF (Auxiliar carry Flag, bit 4): Si vale 1, indica que hubo "arrastre" o "préstamo" del nibble (cuatro bits) menos significativo al nibble más significativo. Este indicador se usa con las instrucciones de ajuste decimal.

ZF (Zero Flag, bit 6): Si este indicador vale 1, el resultado de la operación es cero.

SF (Sign Flag, bit 7): Refleja el bit más significativo del resultado. Como los números negativos se representan en la notación de complemento a dos, este bit representa el signo: 0 si es positivo, 1 si es negativo.

TF (Trap Flag, bit 8): Si vale 1, el procesador está en modo paso a paso. En este modo, la CPU automáticamente genera una interrupción interna después de cada instrucción, permitiendo inspeccionar los resultados del programa a medida que se ejecuta instrucción por instrucción.

IF (Interrupt Flag, bit 9): Si vale 1, la CPU reconoce pedidos de interrupción externas.

DF (Direction Flag, bit 10): Si vale 1, las instrucciones con cadenas sufrirán "auto-decremento", esto es, se procesarán las cadenas desde las direcciones más altas de memoria hacia las más bajas. Si vale 0, habrá "auto-incremento", lo que quiere decir que las cadenas se procesarán de "izquierda a derecha".

OF (Overflow flag, bit 11): Si vale 1, hubo un desborde en una operación aritmética con signo, esto es, un dígito significativo se perdió debido a que tamaño del resultado es mayor que el tamaño del destino.

Relación entre el código binario y el lenguaje ensamblador.-En el código binario se utilizan ceros y unos, mientras que el lenguaje ensamblador es una colección de símbolos mnemónicos que representan: operaciones, nombres simbólicos, operadores y símbolos especiales. La relación entre estos dos lenguajes sería que el binario es el lenguaje que la máquina entiende y el ensamblador se acerca mas lenguaje de esta.

Cada programa en lenguaje ensamblador es creado a partir de un archivo fuente de código ensamblador. Estos son archivos de texto que contienen todas las declaraciones de datos e instrucciones que componen al programa y que se agrupan en áreas o secciones, cada una con un propósito especial. Las sentencias en ensamblador tienen la siguiente sintaxis:

[*nombre*] *mnemónico* [*operandos*] [*;comentarios*]

En cuanto a la estructura, todos los archivos fuente tienen la misma forma: uno o más segmentos de programa seguidos por una directiva END. No hay una regla sobre la estructura u orden que deben seguir las diversas secciones o áreas en la creación del código fuente de un programa en ensamblador. Sin embargo la mayoría de los programas

tiene un segmento de datos, un segmento de código y un segmento de stack, los cuales pueden ser puestos en cualquier lugar.

Las cadenas de carácter y constantes alfanuméricas son formadas como ´caracteres´ o "caracteres". Para referencias simbólicas se utilizan cadenas especiales denominadas *nombres*. Los *nombres* son cadenas de caracteres que no se entrecomillan, los caracteres restantes pueden ser cualquiera de los permitidos, y solamente los 31 primeros caracteres son reconocidos.

Declaración de segmentos.- En lo que respecta a la estructura del programa se tienen las directivas SEGMENT y ENDS que marcan el inicio y final de un segmento del programa. Un segmento de programa es una colección de instrucciones y/o datos cuyas direcciones son todas relativas para el mismo registro de segmento. Su sintaxis es:

nombre SEGMENT [*alineación*] [*combinación*] [*´clase´*]
nombre ENDS

El nombre del segmento es dado por *nombre*, y debe ser único. Segmentos con el mismo nombre se tratan como un mismo segmento. Las opciones alineación, combinación, y clase proporcionan información al LINK sobre cómo ajustar los segmentos. Para alineación se tienen los siguientes valores: byte (usa cualquier byte de dirección), Word (usa cualquier palabra de dirección, 2 bytes/word), para (usa direcciones de párrafos, 16 bytes/párrafo, deafult), y page (usa direcciones de página, 256 bytes/page). Combinación define cómo se combinarán los segmentos con el mismo nombre. Puede asumir valores de: public (concatena todos los segmentos en uno solo), stack (igual al anterior, pero con direcciones relativas al registro SS), common (crea segmentos sobrepuestos colocando el inicio de todos en una misma dirección), memory (indica al LINK tratar los segmentos igual que MASM con public, at) address (direccionamiento relativo a address). clase indica el tipo de segmento, señalados con cualquier nombre. Cabe señalar que en la definición está permitido el anidar segmentos, pero no se permite de ninguna manera el sobreponerlos.

Fin de código fuente.- Otra directiva importante es la que indica el final de un módulo. Al alcanzarla el ensamblador ignorará cualquier otra declaración que siga a ésta. Su sintaxis es:

END [*expresión*]

La opción expresión permite definir la dirección en la cual el programa iniciará.

Etiquetas.- Las etiquetas son declaradas

nombre:

Donde nombre constituye una cadena de caracteres.

Declaración de datos.- Estos se declaran según el tipo, mediante la regla

[*nombre*] *directiva valor*,,,

Donde directiva puede ser DB (bytes), DW (palabras), DD (palabra doble), DQ (palabra cuádruple), DT (diez bytes). También pueden usarse las directivas LABEL (crea etiquetas de instrucciones o datos), EQU (crea símbolos de igualdad) y el símbolo = (asigna absolutos) para declarar símbolos. Estos tienen la siguiente sintaxis:

nombre = *expresión*

nombre EQU *expresión*

nombre LABEL *tipo*

Donde tipo puede ser BYTE, WORD, DWORD, QWORD, TBYTE, NEAR, FAR.

Declaración de estructuras.- Para la declaración de estructuras de datos se emplea la directiva STRUC. Su sintaxis es:

nombre STRUC

campos

nombre ENDS

Instrucciones: Los operandos permitidos se enlistan a continuación:

Constantes.- Pueden ser números, cadenas o expresiones que representan un valor fijo. Por ejemplo, para cargar un registro con valor constante usaríamos la instrucción MOV indicando el registro y el valor que cargaríamos dicho registro.

```
mov ax,9
mov al,´c´
mov bx,65535/3
mov cx,count
```

Directos.- Aquí se debe especificar la dirección de memoria a acceder en la forma segmento:offset.

```
mov ax,ss:0031h
mov al,data:0
mov bx,DGROUP:block
```

Relocalizables.- Por medio de un símbolo asociado a una dirección de memoria y que puede ser usado también para llamados.

```
mov ax, value
call main
mov al,OFFSET dgroup:tabla
```

Registros.- Cuando se hace referencia a cualquiera de los registros de propósito general, apuntadores, índices, o de segmento.

Basados.- Un operador basado representa una dirección de memoria relativa a uno de los registros de base (BP o BX). Su sintaxis es:

desplazamiento[BP]
desplazamiento[BX]
[*desplazamiento*][BP]
[BP+*desplazamiento*]
[BP].*desplazamiento*
[BP]+*desplazamiento*

En cada caso la dirección efectiva es la suma del desplazamiento y el contenido del registro.

```
mov ax,[BP]
mov al,[bx]
mov bx,12[bx]
mov bx,fred[bp]
```

Indexado.- Un operador indexado representa una dirección de memoria relativa a uno de los registros índice (SI o DI). Su sintaxis es:

desplazamiento[DI]
desplazamiento[SI]
[*desplazamiento*][DI]
[DI+*desplazamiento*]
[DI].*desplazamiento*
[DI]+*desplazamiento*

En cada caso la dirección efectiva es la suma del desplazamiento y el contenido del registro.

```
mov ax,[si]
mov al,[di]
mov bx,12[di]
mov bx,fred[si]
```

Base-indexados.- Un operador base-indexado representa una dirección de memoria relativa a la combinación de los registros de base e índice. Su sintaxis es:

desplazamiento[BP][SI]
desplazamiento[BX][DI]
desplazamiento[BX][SI]
desplazamiento[BP][DI]
[*desplazamiento*][BP][DI]
[BP+DI+*desplazamiento*]
[BP+DI].*desplazamiento*
[DI]+*desplazamiento*+[BP]

En cada caso la dirección efectiva es la suma del desplazamiento y el contenido del registro.

```
mov ax,[BP][si]
mov al,[bx+di]
mov bx,12[bp+di]
mov bx,fred[bx][si]
```

Operadores y expresiones.- Se cuenta con los siguientes operadores:

-Aritméticos

expresión1 * *expresión2*
expresión1 / *expresión2*
expresión1 MOD *expresión2*
expresión1 + *expresión2*
expresión1 - *expresión2*
+ *expresión*
- *expresión*

-De corrimiento

expresión1 SHR *contador*

expresión1 SHL *contador*

-Relacionales

expresión1 EQ *expresión2*

expresión1 NE *expresión2*

expresión1 LT *expresión2*

expresión1 LE *expresión2*

expresión1 GT *expresión2*

expresión1 GE *expresión2*

- De bit

NOT *expresión*

expresión1 AND *expresión2*

expresión1 OR *expresión2*

expresión1 XOR *expresiòn2*

-De índice

[*expresión1*] [*expresión2*]

5.2 Desarrollo del PIC

En 1965, la empresa GI creó una división de microelectrónica, GI Microlelectronics, esta división comenzó fabricando memorias EPROM y EEPROM, que conformaban las familias AY3-XXXX y AY5-XXXX. A principios de los años 70 diseño el microprocesador de 16 bits CP1600, razonablemente bueno, pero no manejaba eficazmente entradas y salidas. Para solventar este problema, en 1975 diseño un chip destinado a controlar E/S: **El PIC (Peripheral Interface Controller).** Se trataba de un controlador rápido pero con muy pocas instrucciones pues iba a trabajar en conjunto con el CP1600.

La arquitectura del PIC se comercializó en 1975, era sustancialmente la misma que la que posee en la actualidad la serie PIC16C5X. En aquel momento se fabricaba con tecnología NMOS y el producto sólo se ofrecía con memoria ROM y con un pequeño pero robusto microcódigo.

La década de los 80 no fue buena para GI, que tuvo que reestructurar sus negocios concentrando sus actividades en los semiconductores de potencia. La GI Microelectronics

División se convirtió en una empresa subsidiaria , llamada GI Microelectronics Inc. Finalmente en 1985 la empresa fue vendida y rebautizada como Arizona Microchip Technology (Microchip) y orientaron su negocio a los PIC, las memorias EPROM paralelo y las EEPROM serie. Se comenzó rediseñando los PIC, que pasaron a fabricarse con tecnología CMOS, surgiendo la familia de gama baja PIC16CSX, considerada como la "clásica".

Una de las razones de éxito de los PIC se basa en su utilización. Cuando se aprende a manejar uno de ellos, conociendo su arquitectura y su repertorio de instrucciones, es fácil emplear otro modelo.

Los microcontroladores PIC, basados en la arquitectura RISC (Código de instrucciones reducido), mantienen la mayoría de las características de esta arquitectura. Entre ellas destacan, un conjunto homogéneo de instrucciones, alta velocidad y un número reducido de instrucciones.

Por su reducido costo, su amplia gama y la cantidad de información disponible se ha cubierto un espacio bastante importante en el mercado de los microcontroladores, siendo Microchip una empresa líder junto a Motorola e Intel.

Los microprocesadores de la familia PIC, se pueden agrupar en 3 o 4 categorías diferentes, aunque la gama enana de los PIC esta casi descontinuada y para efectos de esta tesis no es útil, por lo cual omitimos comentarios de ella. Estas categorías están dadas por el tamaño de la palabra de la instrucción. Las categorías son:

-Gama baja; cuenta con una palabra de 12 bits
-Gama media; cuenta con una palabra de 14 bits
-Gama alta; cuenta con una palabra de 16 bits

PIC de gama baja

La gama baja de los PIC encuadra nueve modelos fundamentales actualmente. La memoria de programa puede contener de 512 K a 2 K palabras y pueden ser de tipo ROM, EPROM y también existen modelos con memoria OTP, la cual sólo puede ser grabada una vez.

La memoria de datos puede tener una capacidad comprendida entre 23 y 73 bytes, solo cuentan con un temporizador (TMR0), un set de 33 instrucciones, entre 12 y 20 pines para E/S, el voltaje de alimentación puede variar entre 2 y 6.5 V.

La pila o “stack” sólo dispone de dos niveles, por lo que no se pueden encadenar mas de dos subrutinas, además los PIC de gama baja no admiten interrupciones.

En general se caracterizan por contar con los siguientes recursos:

- Sistema POR (Power On reset): Es la facultad de autoreinicializarse al conectarse la alimentación de voltaje.

- Perro de guardia (Watchdog): Es un temporizador que produce un reset automático si no es recargado antes que pase un tiempo previamente fijado, de esta manera se evita que el sistema quede bloqueado de esta manera el programa no recarga dicho temporizador y se genera un reset.

-Código de protección: Cuando se procede a realizar la grabación del programa puede protegerse para evitar su lectura.

-Líneas de E/S de alta corriente: Las líneas de E/S de los PIC pueden absorber una corriente de salida comprendida entre 20 y 25 mA, que es capaz de excitar ciertos periféricos.

- Modo de bajo consumo (SLEEP): De esta forma el oscilador principal y el CPU se detienen, reduciendo notablemente el consumo de energía.

PIC de gama media

En esta gama se agregan nuevas prestaciones a las ya existentes en los de gama baja, haciéndolos más adecuados en las aplicaciones complejas, pues admiten interrupciones, poseen comparadores de magnitudes analógicas, convertidores A/D, puertos serie y diversos temporizadores.

Comercialmente se ofrecen cuatro variantes de microcontroladores en prácticamente todas las gamas.

- Tipo EPROM: Este tipo se puede grabar y borrar, posteriormente exponiendo la ventana de cristal con la que cuenta la cápsula durante unos minutos a rayos ultravioleta procedentes de lámparas fluorescentes especiales.

- Tipo OTP: En este caso se trata de dispositivos grabables solo una vez, físicamente son similares a los anteriores, pero sin ventana y sin posibilidad de ser regrabados.

- Tipo QTP: En este caso es el propio fabricante quien se encarga de grabar el código en todos los chips que fueron pedidos previamente.

- Tipo SQTP: El fabricante sólo graba algunas posiciones especiales para fines de identificación, como número de serie, palabra clave, checksum, etc.

PIC de gama alta

Corresponden a microcontroladores de arquitectura abierta pudiéndose expandir al exterior al poder sacar los buses de datos, direcciones y control. De está manera se configuran sistemas similares a los de los microcontroladores convencionales, siendo capaces de ampliar la configuración interna del PIC añadiendo nuevos dispositivos de memoria y de E/S externas, su capacidad de memoria alcanza 8 K en memoria de instrucciones y 454 bytes en la memoria de datos.

Estructura de los PIC

En general se pueden clasificar todas las partes de un PIC dentro de alguno de estos grupos.

- Corazón
- Periféricos
- Funciones especiales

Corazón: El corazón del dispositivo contiene las funciones básicas requeridas para que el dispositivo opere, esto incluye:

- Oscilador
- Lógica de reinicialización
- CPU (Central Processing Unit)
- ALU (Arithmetic Logical Unit)
- Organización de mapa de memoria
- Operación de las interrupciones
- Set de instrucciones

Periféricos: Son funciones que permiten el contacto con el mundo exterior (como son las entradas / salidas, entradas A/D y salidas de PWM) y tareas internas como las de bases de tiempo. En general los de categoría mediana contienen los siguientes periféricos:

- Entradas y salidas de uso general
- Timer 0, 1 y 2
- Captura, comparación y PWM (modulación de ancho de pulso)
- Puerto básico de comunicación serial síncrona
- Puerto maestro de comunicación serial síncrona
- USART (unidad síncrona, asíncrona de recepción y transmisión)
- Referencia de voltajes
- Convertidor de 8 bits analógico/digital
- Puerto paralelo esclavo
- Convertidor de 10 bits analógico/digital

Funciones especiales: estas permiten realizar una o más de las siguientes tareas:

- Configuración de los bits del dispositivo
- On-chip Power-on reset
- Brown-out
- Watchdog timer
- Modo de bajo consumo de energía
- Oscilador interno RC
- Programación in-circuit serial (ICSP)

Arquitecturas.- En la actualidad existen dos tipos de arquitecturas diferentes, la tradicional (Von Newman) y el modelo Harvard. La arquitectura de los PIC sigue el modelo Harvard. A continuación se da una explicación de ambos modelos.

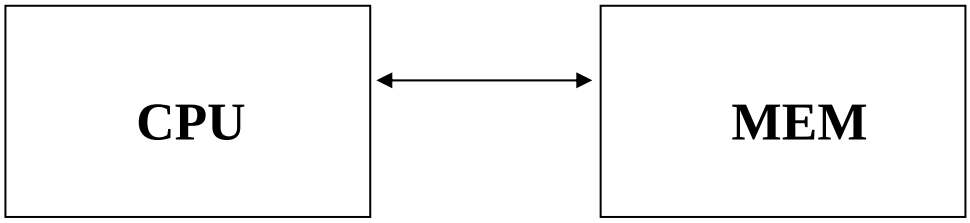
Arquitectura Von Newman: Es un esquema tradicional y fue propuesto por John Von Newman, en el cual la unidad central de proceso, o CPU, esta conectada a una memoria única que contiene las instrucciones del programa y los datos. El tamaño de la unidad de datos o instrucciones está fijado por el ancho del bus de la memoria. Es decir que un microprocesador de 8 bits, que tiene además un bus de 8 bits que lo conecta con la memoria, deberá manejar datos e instrucciones de una o más unidades de 8 bits de longitud. Cuando deba acceder a una instrucción o dato de más de un byte de longitud,

deberá realizar más de un acceso a la memoria. Por otro lado este bus único limita la velocidad de operación del microprocesador, ya que no se puede buscar en la memoria una nueva instrucción. Las dos principales limitantes de esta arquitectura son:

- Que la longitud de la instrucción esta limitada por la unidad de longitud de los datos, por lo tanto el microprocesador deberá hacer varios accesos para buscar instrucciones complejas

- Que la velocidad de operación está limitada por el efecto de cuello de botella que significa un bus único para datos e instrucciones, esto impide superponer ambos tiempos de acceso.

El siguiente es un diagrama sencillo de la manera en la que se realiza la conexión entre el CPU y la memoria.



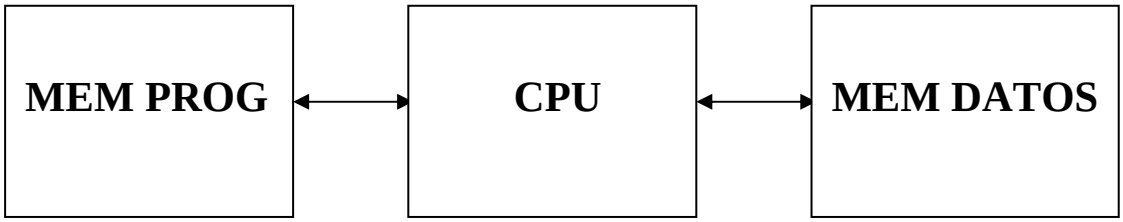
Arquitectura Harvard: Consiste en un esquema en el que el CPU está conectado a dos memorias por medio de dos buses separados. Una de las memorias contiene solamente las instrucciones del programa y es conocida como memoria de programa, la otra memoria sólo almacena los datos y como es de esperarse se llama memoria de datos, ambos buses son totalmente independientes y pueden ser de distintos anchos.

Para un procesador de conjunto de instrucciones reducido, o RISC (Reduced Instruction Set Computer) el conjunto de instrucciones y el bus de la memoria de programa pueden diseñarse de manera tal que todas las instrucciones tengan una sola posición de memoria de programa. Las principales ventajas de esta arquitectura son:

- El tamaño de las instrucciones no esta relacionado con el de los datos y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa.

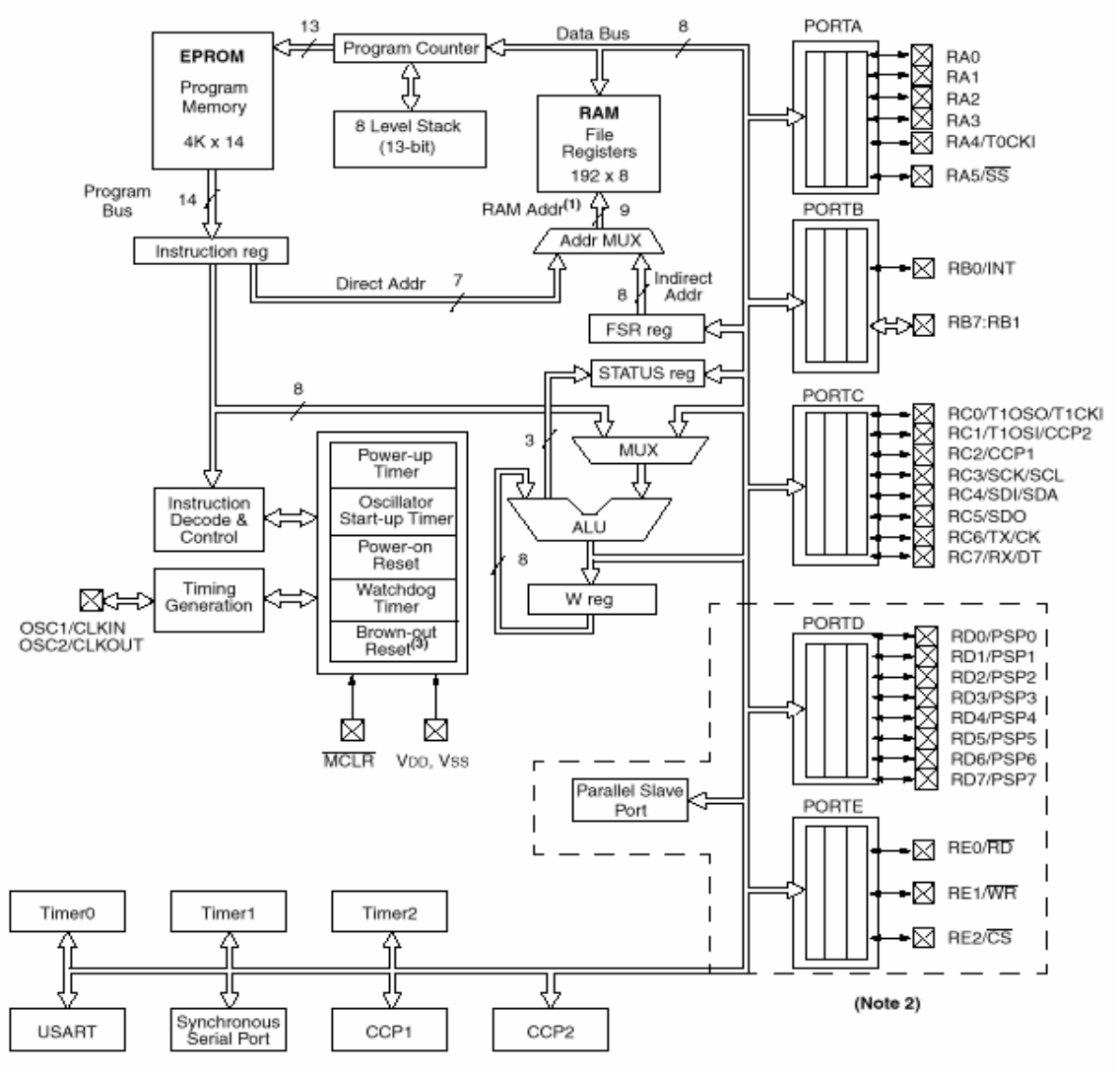
- El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad de operación.

A continuación se ilustra la conexión del CPU y las memorias.



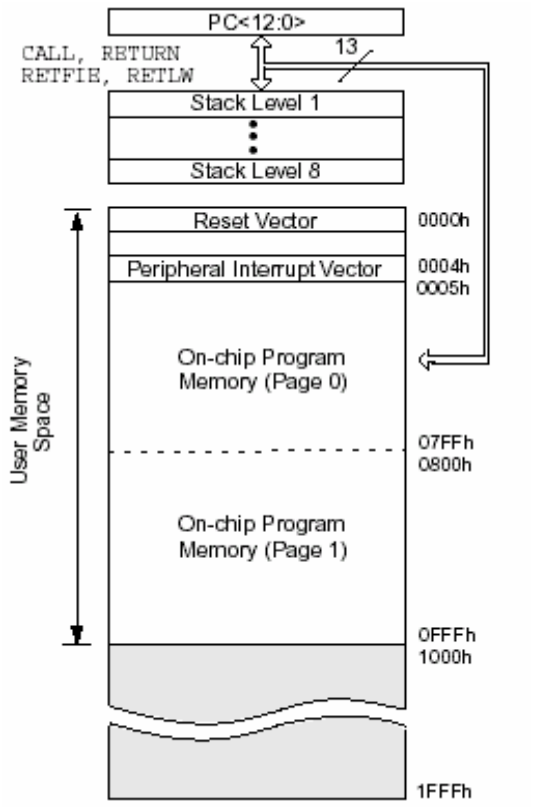
5.3 Características del PIC 16C65B

Partiendo desde lo elemental, a continuación se muestra el diagrama de bloques del PIC16C65B



Memoria Interna (RAM) Organización.- La memoria interna de datos, también llamada archivo de registros (register file), esta dividida en dos grupos: los registros especiales, y los registros de propósito generales. Los primeros ocupan las 32 posiciones primeras que van desde la 00 a la 1F, y los segundos las posiciones que siguen, o sea de la 20 a la 7F. Los registros especiales contienen la palabra de estado (STATUS), los registros de datos de los puertos de entrada salida (Puerto A, Puerto B, Puerto C, Puerto D, Puerto E), los 8 bits menos significativos del program counter (PC), el contador del Real Time Clock/Counter (RTCC) y un registro puntero llamado File Select Register (FSR). La posición 00 no contiene ningún registro en especial y es utilizada en el mecanismo de direccionamiento indirecto.

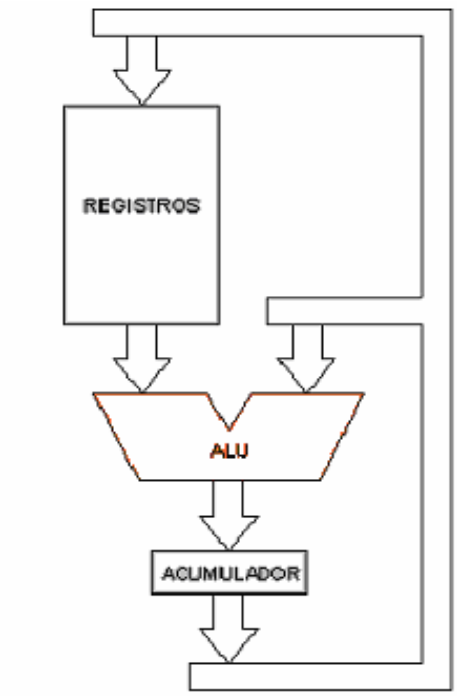
El siguiente es el mapa de memoria de programa y pila del PIC16C65B

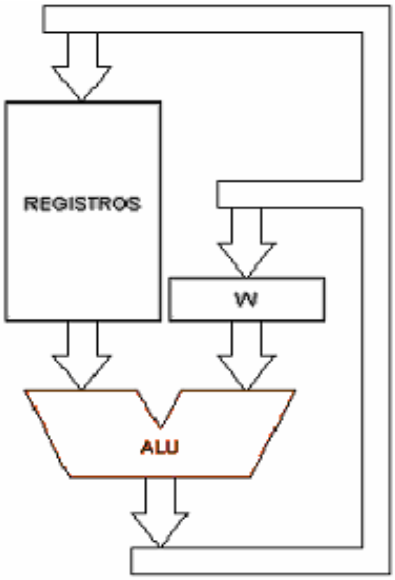


Memoria de Programa.- La memoria de programa, que en los PIC16C6X puede ser de 512 x 14 a 1K x 14 instrucciones, debe ser considerada a los efectos de la programación,

como compuesta por secciones o páginas de 512 posiciones. A su vez cada página debe considerarse dividida en dos mitades de 128 posiciones cada una. Esto se debe, a las limitaciones de direccionamiento de las instrucciones de salto

La figura derecha representa un diagrama simplificado de la arquitectura interna del camino de los datos en el CPU de los microcontroladores PIC. Este diagrama puede no representar con exactitud el circuito interno de estos microcontroladores, pero es exacto y claro desde la óptica del programador. La figura izquierda representa el mismo diagrama para un microprocesador ficticio de arquitectura tradicional. Se puede observar que la principal diferencia entre ambos radica en la ubicación del registro de trabajo, que para los PIC's se denomina W (Working Register), y para los tradicionales es el Acumulador (A).





. **MICROPROCESADOR TRADICIONAL**

MICROPROCESADOR PIC

En los microcontroladores tradicionales todas las operaciones se realizan sobre el acumulador. La salida del acumulador esta conectada a una de las entradas de la Unidad Aritmética y Lógica (ALU), y por lo tanto éste es siempre uno de los dos operandos de cualquier instrucción. Por convención, las instrucciones de simple operando (borrar,

incrementar, decrementar, complementar), actúan sobre el acumulador. La salida de la ALU va solamente a la entrada del acumulador, por lo tanto el resultado de cualquier operación siempre quedara en este registro. Para operar sobre un dato de memoria, luego realizar la operación siempre hay que mover el acumulador a la memoria con una instrucción adicional.

En los microcontroladores PIC, la salida de la ALU va al registro W y también a la memoria de datos, por lo tanto el resultado puede guardarse en cualquiera de los dos destinos. En las instrucciones de doble operando, uno de los dos datos siempre debe estar en el registro W, como ocurría en el modelo tradicional con el acumulador. En las instrucciones de simple operando el dato en este caso se toma de la memoria (también por convención). La gran ventaja de esta arquitectura es que permite un gran ahorro de instrucciones ya que el resultado de cualquier instrucción que opere con la memoria, ya sea de simple o doble operando, puede dejarse en la misma posición de memoria o en el registro W, según se seleccione con un bit de la misma instrucción. Las operaciones con constantes provenientes de la memoria de programa (literales) se realizan solo sobre el registro W.

En la memoria de datos de los PIC's se encuentran ubicados casi todos los registros de control del microprocesador y sus periféricos autocontenidos, y también las posiciones de memoria de usos generales.

Contador de Programa.- Este registro, normalmente denominado PC, es totalmente equivalente al de todos los microprocesadores y contiene la dirección de la próxima instrucción a ejecutar. Se incrementa automáticamente al ejecutar cada instrucción, de manera que la secuencia natural de ejecución del programa es lineal, una instrucción después de la otra. Algunas instrucciones que llamaremos de control, cambian el contenido del PC alterando la secuencia lineal de ejecución. Dentro de estas instrucciones se encuentran el GOTO y el CALL que permiten cargar en forma directa un valor constante en el PC haciendo que el programa salte a cualquier posición de la memoria. Otras instrucciones de control son los SKIP o "saltos" condicionales, que producen un incremento adicional del PC si se cumple una condición especifica, haciendo que el programa salte, sin ejecutar, la instrucción siguiente.

Al resetearse el microprocesador, todos los bits del PC toman valor 1, de manera que la dirección de arranque del programa es siempre la ultima posición de memoria de programa. En esta posición se deberá poner una instrucción de salto al punto donde verdaderamente se inicia el programa.

A diferencia de la mayoría de los microprocesadores convencionales, el PC es también accesible al programador como registro de memoria interna de datos, en la posición de 02. Es decir que cualquier instrucción común que opere sobre registros puede ser utilizada para alterar el PC y desviar la ejecución del programa. El uso indiscriminado de este tipo de instrucciones complica el programa y puede ser muy peligroso, ya que puede producir comportamientos difíciles de predecir. Sin embargo, algunas de estas instrucciones utilizadas con cierto método, pueden ser muy útiles para implementar poderosas estructuras de control tales como el goto computado. Como el microprocesador opera con datos de 8 bits, y la memoria de datos es también de 8 bits, estas instrucciones solo pueden leer o modificar los bits 0 a 7 del PC.

Stack.- En los microcontroladores PIC el stack es una memoria interna dedicada, de tamaño limitado, separada de las memorias de datos y de programa, inaccesible al programador, y organizada en forma de pila, que es utilizada solamente, y en forma automática, para guardar las direcciones de retorno de subrutinas e interrupciones. Cada posición es de 11 bits y permite guardar una copia completa del PC. Como en toda memoria tipo pila, los datos son accedidos de manera tal que el primero que entra es el último que sale.

En los 16C6X el stack es de 8 posiciones, mientras que en los 17CXX es de 16 posiciones. Esto representa, en cierta medida, una limitación de estos microcontroladores, ya que no permite hacer uso intensivo del anidamiento de subrutinas. En los 16C6X, solo se pueden anidar dos niveles de subrutinas, es decir que una subrutina que es llamada desde el programa principal, puede a su vez llamar a otra subrutina, pero esta última no puede llamar a una tercera, porque se desborda la capacidad del stack, que solo puede almacenar dos direcciones de retorno. Esto de hecho representa una traba para el programador y además parece impedir o dificultar la programación estructurada, sin embargo es una buena solución de compromiso ya que estos microcontroladores están diseñados para aplicaciones de alta velocidad en tiempo real, en las que el overhead (demoras adicionales)

que ocasiona un excesivo anidamiento de subrutinas es inaceptable. Por otra parte existen técnicas de organización del programa que permiten mantener la claridad de la programación estructurada, sin necesidad de utilizar tantas subrutinas anidadas.

Como ya se menciono anteriormente, el stack y el puntero interno que lo direcciona, son invisibles para el programador, solo se los accede automáticamente para guardar o rescatar las direcciones de programa cuando se ejecutan las instrucciones de llamada o retorno de subrutinas, o cuando se produce una interrupción o se ejecuta una instrucción de retorno de ella.

La palabra de estado del procesador contiene los tres bits de estado de la ALU (C, DC y Z), y otros bits que por comodidad se incluyeron en éste registro. 7 6 5 4 3 2 1 0 Registro STATUS El bit Z indica que el resultado de la ultima operación fue CERO. El bit C indica acarreo del bit más significativo (bit 7) del resultado de la ultima operación de suma. En el caso de la resta se comporta a la inversa, C resulta 1 si no hubo pedido de préstamo. El bit DC (digit carry) indica acarreo del cuarto bit (bit 3) del resultado de la última operación de suma o resta, con un comportamiento análogo al del bit C, y es útil para operar en BCD (para sumar o restar números en código BCD empaquetado). El bit C es usado además en las operaciones de rotación derecha o izquierda como un paso intermedio entre el bit 0 y el bit 7.

El bit PD (POWER DOWN) sirve para detectar si la alimentación fue apagada y encendida nuevamente, tiene que ver con la secuencia de inicialización. El bit TO (TIME-OUT) sirve para detectar si una condición de reset fue producida por el watch dog timer, esta relacionado con los mismos elementos que el bit anterior. Los bits de selección de pagina PA0/PA1/PA2 se utilizan en las instrucciones de salto GOTO y CALL.

Otros registros especiales.- Las ocho primeras posiciones del área de datos están reservadas para alojar registros de propósito especial, quedando las restantes libres para contener los datos u operandos que se desee (registros de propósito general).

El registro INDF que ocupa la posición 0 no está implementando físicamente y, como se ha explicado, se le referencia en el direccionamiento indirecto de datos aunque se utiliza el contenido de FSR. En la dirección esta el registro TAR0 (Temporizador) que puede ser leído y escrito como cualquier otro registro. Puede incrementar su valor con una señal externa aplicada al pin T0CKI o mediante el oscilador interno.

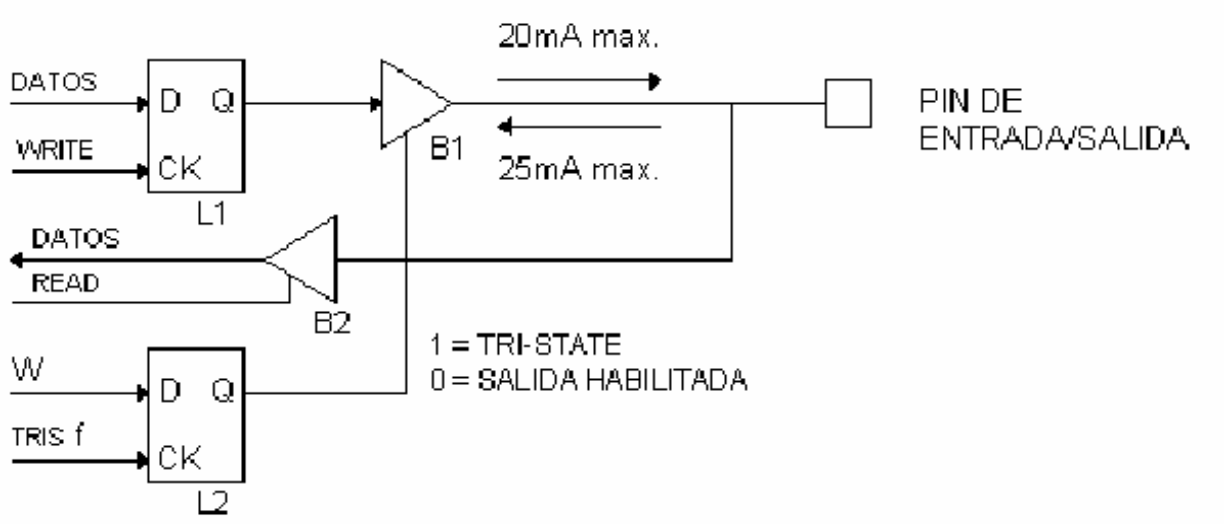
El PC ocupa la posición 2 del área de datos en donde se halla el registro PCL al que se añaden 3 bits auxiliares y se conectan con los dos niveles de la Pila en las instrucciones CALL y RETLW.

El registro de Estado ocupa la posición 3 y entre sus bits se encuentran los señalizadores C, DC y Z y los bits PA1 y PA0 que seleccionan la página en la memoria de programa. El bit 7 (PA2) no está implementando en los PIC de la gama baja.

FRS se ubica en la dirección 4 y puede usarse para contener la dirección del dato en las instrucciones con direccionamiento indirecto y también para guardar operandos en sus 5 bits de menos peso.

Puertos de entrada / salida Los microprocesadores.- PIC16C6X tienen varios puertos de entrada/salida paralelo de usos generales llamados Puerto A, Puerto B, Puerto C, Puerto D y Puerto E. El Puerto A es de cuatro bits y los demás son de 8 bits cada uno.

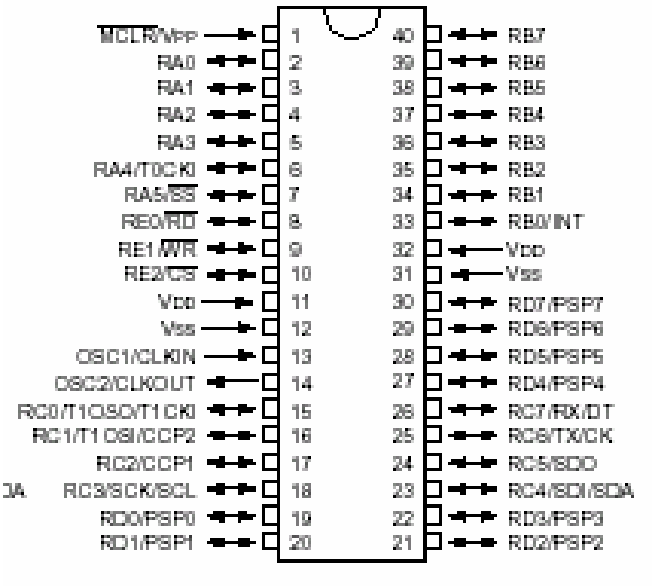
El circuito equivalente de un bit cualquiera de un puerto de entrada salida es el siguiente



El latch L1 corresponde a un bit del registro de datos del puerto, mientras que L2 es un bit del registro de control de tristate del mismo. B1 es el buffer tristate de salida que tiene capacidad de entregar 20 mA y drenar 25 mA. B1 es controlado por L2. Si L2 tiene cargado un “1”, B1 se encuentra en tri-state, es decir con la salida desconectada (en alta impedancia), y el puerto puede ser usado como entrada. Si L2 tiene cargado un “0”, la salida

de B1 esta conectada (baja impedancia) y el puerto esta en modo de salida. B2 es el buffer de entrada, es decir el que pone los datos en el bus interno del microcontrolador cuando se lee el registro de datos del puerto. Puede verse que el dato leído es directamente el estado del pin de entrada.

Diagrama lógico para los microcontroladores PIC16C65

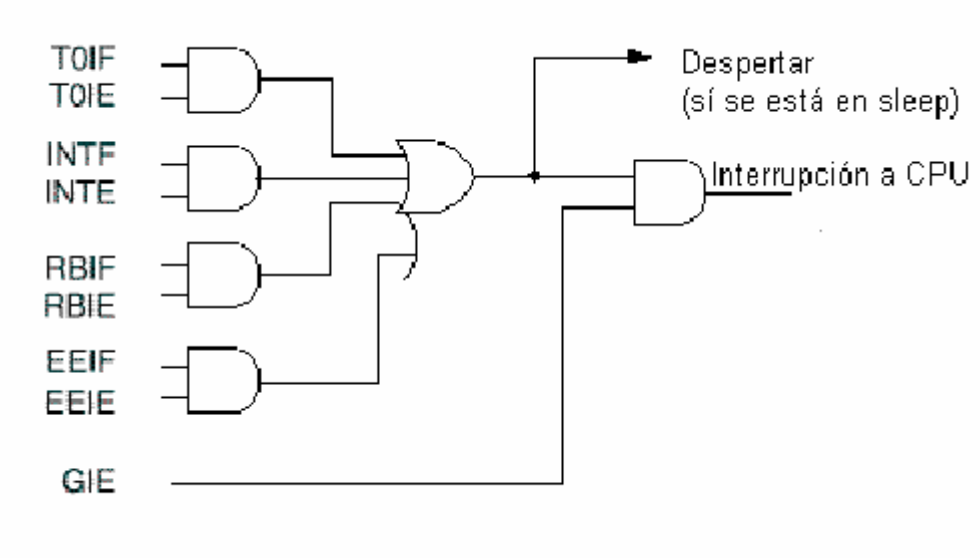


Interrupciones.- Los 16C6X agregan la posibilidad de contar con sistema de interrupciones. Este sistema consiste en un mecanismo por el cual un evento interno o externo, asincrónico respecto del programa, puede interrumpir la ejecución de éste produciendo automáticamente un salto a una subrutina de atención, de manera que pueda atender inmediatamente el evento, y retomar luego la ejecución del programa exactamente en donde estaba al momento de ser interrumpido. Este mecanismo es muy útil por ejemplo para el manejo de timers o rutinas que deben repetirse periódicamente (refresh de display, antirebote de teclado, etc.), detección de pulsos externos, recepción de datos, etc.

Existen de tres a doce eventos que pueden generar interrupciones en los PIC16C6X existentes hasta el momento, pero nada impide que puedan agregarse más en versiones futuras.

Las interrupciones se comportan casi exactamente igual que las subrutinas. Desde el punto de vista del control del programa, al producirse una interrupción se produce el mismo

efecto que ocurriría si el programa tuviese un CALL 0004h en el punto en que se produjo la interrupción. En uno de los registros de control del sistema de interrupciones existe un bit de habilitación general de interrupciones GIE, que debe ser programado en 1 para que las interrupciones puedan actuar. Al producirse una interrupción, este bit se borra automáticamente para evitar nuevas interrupciones. La instrucción RETFIE que se utiliza al final de la rutina de interrupción, es idéntica a un retorno de subrutina, salvo que además coloca en uno automáticamente el bit GIE volviendo a habilitar las interrupciones. Dentro de la rutina de interrupción, el programa deberá probar el estado de los flags de interrupción de cada una de las fuentes habilitadas, para detectar cual fue la que causo la interrupción y así decidir que acción tomar.



La señal que produce la interrupción es en realidad una sola, que resulta de la combinación de todas las fuentes posibles y de los bits de habilitación. Existen dos grupos de fuentes, unas que se habilitan con solo colocar en uno el bit GIE, y otras que además necesitan que este puesto a uno el bit PEIE. En algunas versiones de los 16CXX solo existe el primer grupo. Además, cada fuente de interrupciones tiene su respectivo bit de habilitación individual.

Conjunto de instrucciones.- El conjunto de instrucciones de los microprocesadores PIC 16C6X consiste en un pequeño repertorio de solo 35 instrucciones, que pueden ser agrupadas para su estudio en tres a cinco grupos. Desde el punto de vista del programador,

en cinco categorías bien definidas de acuerdo con la función y el tipo de operandos involucrados. En primer lugar se agrupan las instrucciones que operan con bytes y que involucran algún registro de la memoria interna. En segundo lugar están las instrucciones que operan solo sobre el registro W y que permiten cargarle una constante implícita o incluida literalmente en la instrucción (literales). En tercer lugar se agrupan las instrucciones que operan sobre bits individuales de los registros de la memoria interna. En cuarto lugar se clasifican las instrucciones de control de flujo del programa, es decir las que permiten alterar la secuencia lineal de ejecución de las instrucciones. Por último se agrupan unas pocas instrucciones que llamaremos especiales, cuyas funciones o tipos de operandos son muy específicos y no encajan en ninguna de las clasificaciones anteriores.

Instrucciones de Byte que operan con Registros.- Estas instrucciones pueden ser de simple o doble operando de origen. El primer operando de origen será siempre el registro seleccionado en la instrucción, el segundo, en caso de existir, será el registro W. El destino, es decir donde se guardara el resultado, será el registro seleccionado o el W, según se seleccione con un bit de la instrucción.

Las instrucciones siguientes son las tres operaciones lógicas de doble operando :

ANDWF f,d ;operación AND lógica, destino = W _ f

IORWF f,d ;operación OR lógica, destino = W _ f

XORWF f,d ;operación XOR lógica, destino = W _ f

Los nombres mnemónicos de estas instrucciones provienen de : AND W con F, Inclusive OR W con F y XOR W con F.

Las que siguen son las cuatro operaciones aritméticas y lógicas sencillas de simple operando :

MOVF f,d ;movimiento de datos, destino = f

COMF f,d ;complemento lógico, destino = NOT f

INCF f,d ;incremento aritmético, destino = f + 1

DECF f,d ;decremento aritmético, destino = f - 1

Los mnemónicos de estas instrucciones provienen de : MOVe File, COMplement File, INCrement File y DECrement File.

En las siete instrucciones anteriores el único bit afectado de la palabra de estado del procesador es el Z, que se pone en 1 si el resultado de la operación es 00000000, y se pone en 0 si el resultado tiene cualquier otro valor.

A continuación siguen las dos instrucciones de rotación de bits a través del CARRY :

RLF f,d ;rotación a la izquierda, destino = f ROT _

RRF f,d ;rotación a la derecha, destino = f ROT _

En estas operaciones (Rotate Left File y Rotate Right File) los bits son desplazados de cada posición a la siguiente, en sentido derecho o izquierdo. El desplazamiento es cerrado, formando un anillo, con el bit C (CARRY) de la palabra de estado.

En estas dos instrucciones, el único bit afectado de la palabra de estado del procesador es el bit C, que tomará el valor que tenia el bit 7 o el bit 0, según sea el sentido del desplazamiento.

Estas instrucciones son muy útiles para la manipulación de bits, y además para realizar operaciones aritméticas, ya que en numeración binaria, desplazar un número a la izquierda es equivalente a multiplicarlo por 2, y hacia la derecha, a dividirlo por 2.

La instrucción siguiente realiza el intercambio de posiciones entre los cuatro bits menos significativos y los cuatro más significativos (nibble bajo y nibble alto).

SWAPF f,d ;intercambia nibbles, destino = SWAP f

Esta instrucción (SWAP File) no afecta ninguno de los bits de la palabra de estado del procesador.

Esta instrucción es muy útil para el manipuleo de números BCD empaquetados, en los que en un solo byte se guardan dos dígitos BCD (uno en cada nibble). Las dos operaciones que siguen son la suma y la resta aritméticas.

ADDWF f,d ;suma aritmética, destino = f + W

SUBWF f,d ;resta aritmética, destino = f - W

Estas operaciones (ADD W a F y SUBstract W de F) afectan a los tres bits de estado C, DC y Z. El bit Z se pone en 1 si el resultado de la operación es 00000000, y se pone en 0 si el resultado tiene cualquier otro valor.

La suma se realiza en aritmética binaria pura sin signo. Si hay un acarreo del bit 7, es decir que el resultado es mayor que 255, el bit C (carry) resulta 1, en caso contrario resulta 0. Si hay un acarreo del bit 3, es decir que la suma de las dos mitades (nibbles) menos

significativas (bits 0 a 3) resulta mayor que 15, se pone en 1 el bit DC (digit carry), en caso contrario se pone en 0.

Los bits de estado C y DC toman el valor normal correspondiente a la suma de f con el complemento a 2 de W. De esta manera el significado para la operación de resta resulta invertido, es decir que C (carry) es 1 si no hubo desborde en la resta, o dicho de otra manera, si el contenido de W es menor que el de f. El bit DC se comporta de manera similar, es decir que DC es 1 si no hubo desborde en la mitad menos significativa, lo que equivale a decir que el nibble bajo del contenido de W es menor que el del registro f.

Las instrucciones que siguen son de simple operando, pero son casos especiales ya que el destino es siempre el registro seleccionado :

CLRF f ;borrado de contenido, f = 0

MOVWF f ;copia contenido W _ f, f = W

La instrucción CLRF (CLeaR File) afecta solo al bit Z que resulta siempre 0. La instrucción MOVWF (MOVe W a F) no afecta ningún bit de la palabra de estado.

Instrucciones de Byte que operan sobre W y Literales, Estas instrucciones se refieren todas al registro W, es decir que uno de los operandos de origen y el operando de destino son siempre el registro W. En las instrucciones de este grupo que tienen un segundo operando de origen, este es siempre una constante de programa literalmente incluida en la instrucción, llamada constante literal o simplemente literal.

Las tres instrucciones que siguen son las operaciones lógicas tradicionales, similares a las que ya vimos anteriormente, pero realizadas entre una constante de programa y el registro W :

IORLW k ; operación OR lógica, W = W _ k

ANDLW k ; operación AND lógica, W = W _ k

XORLW k ; operación XOR lógica, W = W _ k

En estas tres instrucciones (Inclusive OR Literal W, AND Literal W y XOR Literal W) el único bit afectado de la palabra de estado del procesador es el Z, que se pone en 1 si el resultado de la operación es 00000000, y se pone en 0 si el resultado tiene cualquier otro valor.

La instrucción que sigue sirve para cargar una constante de programa en el registro W :

MOVLW k ;carga constante en W, W = K

Esta (MOV Literal W) instrucción no afecta ninguno de los bits de estado del procesador.

La instrucción que sigue (CLeaR W) no correspondería incluirla en este grupo, y pertenece en realidad al primero, el de las instrucciones que operan sobre registros, ya que se trata de un caso especial de la instrucción CLRF, con destino W, y f = 0. Se incluye aquí porque como se le ha asignado un mnemónico particular referido específicamente al registro W, se entiende que, desde el punto de vista del programador, es más útil verla dentro del grupo de instrucciones referidas a W.

CLRW ;borra el contenido de W, W = 0

Al igual que en la instrucción CLRF, el único bit de estado afectado es el Z que resulta 1.

Estas instrucciones no tienen especificación de destino, ya que el mismo es siempre el registro seleccionado.

BCF f,b ;borra el bit b de f ;bit f(b) = 0

BSF f,b ;coloca en uno el bit b de f ;bit f(b) = 1

Estas instrucciones (Bit Clear File y Bit Set File) no afectan ningún bit de la palabra de estado del procesador.

Instrucciones de Control

GOTO k ;salto a la posición k (9 bits) del programa

Esta es la típica instrucción de salto incondicional a cualquier posición de la memoria de programa (que en la mayoría de los microprocesadores convencionales se llama JUMP). La constante literal k es la dirección de destino del salto, es decir la nueva dirección de memoria de programa a partir de la cual comenzarán a leerse las instrucciones después de ejecutar la instrucción GOTO. Esta instrucción simplemente carga la constante k en el registro PC (contador de programa).

La que sigue es la instrucción de llamado a subrutina:

CALL k ;salto a la subrutina en la posición k (8 bits)

Su comportamiento es muy similar al de la instrucción GOTO, salvo que además de saltar guarda en el stack la dirección de retorno de la subrutina (para la instrucción RETLW). Esto lo hace simplemente guardando en el stack una copia del PC incrementado, antes de

que el mismo sea cargado con la nueva dirección k. La única diferencia con la instrucción GOTO respecto de la forma en la que se realiza el salto, es que en la instrucción CALL la constante k tiene solo 8 bits en vez de 9. En este caso también se utilizan PA0 y PA1 para cargar los bits 9 y 10 del PC, pero además el bit 8 del PC es cargado siempre con 0. Esto hace que los saltos a subrutina solo puedan realizarse a posiciones que estén en las primeras mitades de las paginas mencionadas. El programador debe tener en cuenta este comportamiento y asegurarse de ubicar las posiciones de inicio de las subrutinas en las primeras mitades de las paginas.

La instrucción que aparece a continuación es la de retorno de subrutina:

RETURN ;retorno de subrutin

La operación que realiza consiste simplemente en regresar de la ultima instrucción CALL ejecutada, por lo tanto es la dirección de la instrucción siguiente a dicho CALL, dado que el stack es de 11 bits, el valor cargado en el PC es una dirección completa, y por lo tanto se puede retornar a cualquier posición de la memoria de programa, sin importar como estén los bits de selección de pagina. Esta instrucción además carga siempre una constante literal en el registro W.

A continuación se presentan las dos únicas instrucciones de “salto” (skip) condicional. Estas instrucciones son los únicos medios para implementar bifurcaciones condicionales en un programa. Son muy generales y muy poderosas ya que permiten al programa tomar decisiones en función de cualquier bit de cualquier posición de la memoria interna de datos, y eso incluye a los registros de periféricos, los puertos de entrada/salida e incluso la palabra de estado del procesador. Estas dos instrucciones reemplazan y superan a todo el conjunto de instrucciones de salto condicional que poseen los microprocesadores sencillos convencionales (salto por cero, por no cero, por carry, etc.).

BTFSC f,b ;salto si bit = 0, bit = f(0) _ salta

BTFSS f,b ;salto si bit = 1, bit = f(1) _ salta

BTFSC (Bit Test File and Skip if Clear) salta la próxima instrucción si el bit b del registro f es cero. La instrucción BTFSS (Bit Test File and Skip if Set) salta si el bit es 1. Estas instrucciones pueden usarse para realizar o no una acción según sea el estado de un bit, o, en combinación con GOTO, para realizar una bifurcación condicional.

Las instrucciones que siguen son casos especiales de las de incremento y decremento vistas anteriormente. Estas instrucciones podrían categorizarse dentro del grupo de instrucciones orientadas a byte sobre registros (primer grupo), ya que efectivamente operan sobre los mismos, y el formato del código de la instrucción responde al de ese grupo, pero, a diferencia de las otras, pueden además alterar el flujo lineal del programa y por eso se les incluyó en este grupo.

DECFSZ f,d ;decrementa y salta sí 0, destino= f - 1, = 0 _ salta

INCFSZ f,d ;incrementa y salta sí 0, destino= f + 1, = 0 _ salta

Estas dos instrucciones (DECrement File and Skip if Zero, e INCrement File and Skip if Zero) se comportan de manera similar a DECF e INCF, salvo que no afectan a ningún bit de la palabra de estado. Una vez realizado el incremento o decremento, si el resultado es 00000000, el microprocesador saltara la próxima instrucción del programa. Estas instrucciones se utilizan generalmente en combinación con una instrucción de salto (GOTO), para el diseño de ciclos o lazos (loops) de instrucciones que deben repetirse una cantidad determinada de veces.

Instrucciones Especiales En este grupo se reunieron las instrucciones que controlan funciones específicas del microprocesador o que actúan sobre registros especiales no direccionados como memoria interna normal.

La instrucción que sigue es la típica NO OPERATION, existente en casi todos los microprocesadores.

NOP ;no hace nada, consume tiempo

Esta instrucción solo sirve para introducir una demora en el programa, equivalente al tiempo de ejecución de una instrucción. No afecta ningún bit de la palabra de estado.

Resumen de instrucciones:

Mnemonico, Operando	***Descripción***	***Status Afectado***

ADDWF f,d	Suma W y F	C,DC,Z
ANDWF f,d	AND entre W y F	Z
CLRF f	Limpia F	Z
CLRW -	Limpia W	Z
COMF f,d	Complemento a F	Z
DECF f,d	Decrementa F	Z
DECFSZ f,d	Decrementa F y salta si es cero	
INCF f,d	Incrementa F	Z
INCFSZ f,d	Incrementa F y salta si es cero	
IORWF f,d	OR entre W y F	Z
MOVF f,d	Mueve F a d	Z
MOVWF f	Mueve W a F	
NOP -	No operar	
RLF f,d	Rota a la izquierda con carry	C
RRF f,d	Rota a la derecha con carry	C
SUBWF f,d	Resta W de F	C,DC,Z
SWAPF f,d	Intercambia nibbles de F	
XORWF f,d	XOR entre W y F	Z
BCF f,b	Pone 0 en el bit de F	
BSF f,b	Pone 1 en el bit de F	
BTFSC f,b	Verifica si el bit de F es cero y salta	
BTFSS f,b	Verifica si el bit de F es uno y salta	
ADDLW k	Suma W y K	C,DC,Z
ANDLW k	AND entre W y K	Z
CALL k	Llama a subrutina K	
CLRWDT -	Limpia perro de guardia	
GOTO k	Salta a K	
IORLW k	OR entre W Yk	Z
MOVLW k	Mueve K a W	
RETFIE -	Regresa de interrupción	

RETLW k	***Regresa con K***	
RETURN -	***Regreso de subrutina***	
SLEEP -	***Va a modo de espera***	
SUBLW k	***Resta W de K***	
XORLW k	***XOR entre W y K***	***Z***

Donde f es el archivo o registro sobre el que se esta operando, W es el registro de trabajo o acumulador, d es el destino donde se almacena el resultado de la operación (Ya sea w ó f), b es el bit que se opera en un registro y k es una constante o literal.

En esta tabla de resumen del conjunto de instrucciones se pueden observar los mnemónicos, la explicación y los bits afectados del registro STATUS para cada una de las instrucciones.

Modos de direccionamiento .- Direccionamiento de la memoria de datos (RAM) La memoria interna se direcciona en forma directa por medio de los 5 bits “f” contenidos en las instrucciones que operan sobre registros. De esta manera se puede direccionar cualquier posición desde la 00 a la 1F , las primeras direcciones corresponden a los bancos de registros, por lo tanto, en los microcontroladores que tengan más de un banco, antes de acceder a alguna variable que se encuentre en esta zona, el programador deberá asegurarse de haber programado los bits de selección de banco en el registro FSR.

Direccionamiento de la memoria de programa (EPROM, OTP) La instrucción GOTO dispone solo de 9 bits en el código de operación para especificar la dirección de destino del salto. Al ejecutar una instrucción GOTO el microprocesador toma los dos bits que restan para completar la dirección de 11 bits, de los bits 5 y 6 de la palabra de estado. Estos últimos son llamados bits de selección de página (PA0 y PA1). El programador deberá asegurarse de que estos dos bits tengan el valor correcto antes de toda instrucción GOTO.

Deberá tenerse en cuenta además que es posible avanzar de una página a otra en forma automática cuando el PC se incrementa. Esto ocurre si el programa empieza en una página y continúa en la siguiente. Sin embargo, al incrementarse el PC desde la última posición de una página a la primera de la siguiente, los bits PA0 y PA1 no se modifican, y por lo tanto sí se ejecuta una instrucción GOTO, CALL o alguna que actúe sobre el PC, esta producirá un salto a la página anterior, a menos que el programador tenga la precaución de

actualizar el valor de dichos bits. Por este motivo es conveniente dividir el programa en módulos o rutinas que estén confinados a una página.

En el caso de la instrucción CALL, el direccionamiento se complica un poco más, ya que la misma solo dispone de 8 bits para especificar la dirección de destino salto. En este caso también se utilizan los mismos bits de selección de página para completar los bits décimo y decimoprimero de la dirección, pero falta el noveno bit. En estas instrucciones este bit se carga siempre con 0, lo que implica que solo se pueden realizar saltos a subrutina a las mitades inferiores de cada página. En este caso también el programador deberá asegurarse que el estado de los bits PA0 y PA1 sea el correcto al momento de ejecutarse la instrucción.

Las instrucciones que operan sobre el PC como registro y alteran su contenido provocando un salto, responden a un mecanismo muy similar al de las instrucciones CALL para la formación de la dirección de destino. En este caso los bits 0 a 7 son el resultado de la instrucción, el bit 8 es 0 y los bits restantes se toman de PA0 y PA1.

Este mecanismo se llama paginado, y a pesar de que representa una complicación bastante molesta para el programador, resulta muy útil ya que permite ampliar la capacidad de direccionamiento de memoria de programa para las instrucciones de salto

Subrutinas y llamados

La mayoría de los microcontroladores incluyen en su repertorio de instrucciones algunas que permiten saltar a una rutina y, cuando se complementa su ejecución, retornar al programa principal.

El empleo de subrutinas aporta muchas ventajas entre las que se destacan las siguientes:

1. Se pueden escribir como subrutinas secciones de código y ser empleadas en muchos programas.
2. Dan a los programas un carácter modular, es decir, se pueden codificar diferentes módulos para usarlos en cualquier programa.
3. Se reduce notablemente el tiempo de programación, la detección de errores, usando repetidamente una subrutina.
4. El código es más fácil de interpretar, dado que las instrucciones de las subrutinas no aparecen en el programa principal. Solo figuran las llamadas CALLs.

La instrucción CALL (llamada la subrutina) consigue que la ejecución del programa continúe en la dirección donde se encuentra la subrutina a la que hace referencia. Es similar a GOTO pero coloca en la pila la dirección de la siguiente instrucción que se debe ejecutar después de la CALL.

La subrutina finaliza con la instrucción RETURN (Retorno de la subrutina) que retoma la dirección guardada en le pila y la coloca en el contador del programa PC continuando el flujo de control con la instrucción que sigue a la CALL.

En la familia PIC de gama media la pila tiene ocho niveles de memoria del tipo FIFO (primero en entrar, último en salir). Si se produce la llamada a una subrutina durante la ejecución de otra subrutina, la dirección de retorno de esta segunda es colocada en la cima de la pila sobre la dirección anterior. Esta segunda dirección es la primera en salir de la pila mediante la instrucción RETURN.

Con la pila de ocho niveles, una subrutina puede llamar a otra y ésta, a su vez, llamar a otra hasta un máximo de ocho. La gama baja sólo puede realizar dos llamadas de este tipo al poseer una pila de sólo dos niveles.

Las subrutinas deben colocarse al comienzo de las páginas debido a que el bit 8 del contador del programa es puesto a 0 por la instrucción CALL (o por cualquier instrucción que modifica el PC). Las subrutinas deben colocarse en la mitad inicial de las páginas (las 256 palabras).

Conversión a ASCII.- El conjunto de carácteres ASCII (American Standard Code for Information Interchange) es el código de representación en hexadecimal del alfabeto, los números del 0 al 9 y los principales símbolos de puntuación y algunos caracteres de control.

De los problemas usuales en la programación está el convertir un número hexadecimal representado en 8 bits a dos caracteres ASCII los cuales sean la representación de dicho número para permitir la visualización en terminales de datos tales como Monitores de video o Impresoras entre otras.

Temporización .- Existen momentos dentro de la programación en los que se necesita realizar un retardo de tiempo. Los retardos de tiempo se pueden obtener mediante hardware o por medio de ciclos repetitivos basados en software. La precisión de los retardos generados por software depende en esencia del tipo de oscilador que se utilice como base de tiempo en el microcontrolador, la mayor precisión se obtiene de los cristales de cuarzo.

La velocidad a la que se ejecuta el código (instrucciones) depende de la velocidad del oscilador y del número de ciclos de máquina ejecutados. Las instrucciones necesitan 1 ó 2 ciclos de máquina para ser ejecutadas. Un ciclo de máquina es un tiempo utilizado por el microcontrolador para realizar sus operaciones internas y equivale a cuatro ciclos del oscilador. Por tanto:

$T_{ciclo\ máq.} = 4 * T_{osc}$ ___ $T_{ciclo\ máq} = 4 / f_{osc}$ El número de ciclos de máquina utilizados por una instrucción para ser ejecutada depende de la misma. Las instrucciones que modifican el contador de programa necesitan dos (2) ciclos de máquina, mientras que todas las demás necesitan tan solo uno (1).

El hecho de generar ciclos repetitivos por medio del programa y calcular el tiempo total de ejecución nos puede ayudar a generar tiempos precisos.

Operaciones Entrada / Salida Objetivos:

.- Verificar el modo en el que se debe programar el sentido de los puertos

.- Realizar la entradas por puerto mediante la lectura de interruptores "dip-switch"

.- Escribir sobre un puerto de salida visualizando sobre LEDs

En el proceso de utilización de un puerto debe tenerse en cuenta como primera instancia la programación del sentido en que dicho puerto va a utilizarse. Una vez energizado el microcontrolador todos y cada uno de los puertos quedan programados como entrada, entonces, tan solo deben programarse los que se quieren utilizar como salida.

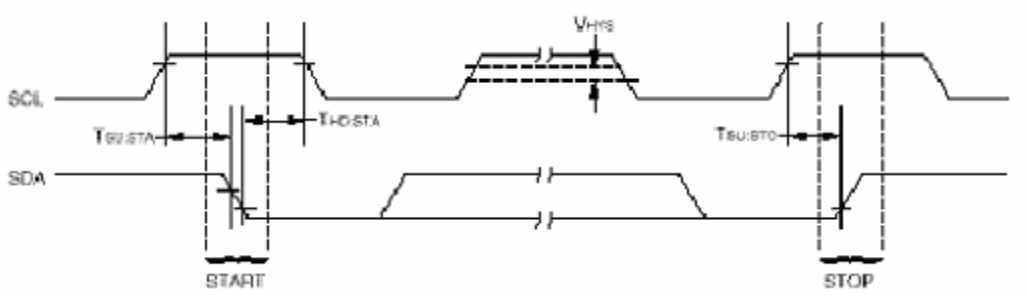
Objetivos de la comunicación serial

.- Verificar la comunicación serial síncrona y asíncrona

- Comprobar los algoritmos de comunicación serial

Comunicación serial síncrona: La comunicación síncrona se caracteriza porque los pulsos de sincronización deben ser transmitidos a lo largo de la línea de comunicación entre el transmisor y el receptor. Dentro de los varios tipos de comunicación serial síncrona destacan el protocolo I_2C ó de dos hilos y el protocolo SPI ó de tres hilos.

I_2C El bus I_2C es un bus diseñado para que sobre éste puedan colocarse varios dispositivos dentro de la misma tarjeta electrónica (comunicación multipunto), cada dispositivo tendrá una dirección lógica asignada físicamente mediante los pines A0, A1 y A2 de acuerdo al nivel lógico al que estos sean alambrados.



En la figura se observa la forma en que las señales SCL y SDA deben ser manejadas. Para iniciar la comunicación sobre un dispositivo I_2C debe realizarse la secuencia denominada bit de START que consiste en pasar la línea de datos SDA de nivel alto a bajo mientras que la línea SCL permanece en alto. Para la culminar la comunicación con el dispositivo I_2C debe ejecutarse la secuencia denominada bit de STOP la cual consiste en pasar la línea de datos SDA de nivel bajo a alto mientras que la línea de reloj SCL permanece en alto. Un bit de datos es aceptado por el dispositivo mientras que sobre la línea de datos SDA permanece el nivel adecuado al bit en cuestión, y sobre la línea de reloj SCL se lleva a cabo un pulso, es decir, el paso de nivel de bajo a alto y luego de alto a bajo. Los tiempos implicados en esta secuencia dependen básicamente del fabricante del dispositivo.

El bus SPI es un bus diseñado para que sobre éste se coloque un dispositivo maestro y un dispositivo esclavo (comunicación punto a punto) ver figura Con relación al bus I2C se puede notar que éste soporta mayor velocidad de comunicación.

Comunicación serial asíncrona: En este tipo de comunicación tanto el transmisor como el receptor tienen incluido el reloj de sincronización de tal forma que no se transmite a lo largo de la línea de comunicación.

5.4 Planteamiento de rutinas necesarias

Las rutinas que compondrán el programa deberán conjuntamente controlar la pantalla, el teclado y verificar la recepción en el puerto serie.

La primera rutina a realizar es la inicialización de pantalla, en al cual se envían comandos y retrasos a la pantalla.

En la parte del programa principal lo primero que se realiza es la verificación de la bandera de recepción del USART, si se recibió algún dato se salta a la sección en la cuál se verifica lo que se recibió, ahí se verá si se recibió un comando o un caracter y también a que sección de la pantalla se realizo dicho envío, una vez verificado esto se envía pro el puerto serial USART un comando de recepción parcial, con el cuál el procesador procederá a enviar el dato (comando o carácter antes declarado), después de que se recibió el dato se envía a la pantalla, vuelve a verificar la bandera de ocupado en el puerto serial.

Al mismo tiempo, si no se recibe nada por el puerto serial, se va a la sección de barrido de teclado, la cuál se realiza continuamente y si detecta que se oprimió alguna tecla, se envía el comando de dicha tecla a la sección de transmisión del USART, el cual lo manda al procesador.

5.5 Elaboración de las rutinas

En la sección anterior se plantearon de manera muy general las rutinas a realizar, en esta sección se verá más a detalle su composición.

La primera acción a realizar es la declaración de librerías a través del comando include y el nombre de la librería, dentro de la cuál se encuentran los registros de funciones especiales (PORTA, PORTB, TRISA, TRISB, STATUS, SPBRG, TXSTA, RCSTA, etc.)

A continuación aparece la declaración de las localidades auxiliares, a las que se les asigna un nombre y una sección dentro del banco de propósito general, dentro de estás localidades se guardarán datos durante la ejecución del programa, lo anterior se hace con el fin de contar con una especie de acumuladores alternos, dentro de los cuales se guardarán y extraerán datos para apoyo de rutinas como retrasos, decrementos, etc.

Posteriormente se entra al programa principal en el cuál de inicio se declara la sección de inició de programa mediante el mnemónico ORG que como ya se menciono dicta a partir de que sección se realizará el ensamblado del programa, inmediatamente después se definen los puertos a usar como entradas o salidas.

Ahora se realiza, como se menciono en la sección anterior, la rutina de inicialización de pantalla, sin la cuál no podríamos realizar ningún desplegado en ella, posteriormente se configura el puerto serial USART (unidad síncrona asíncrona de recepción y transmisión), lo anterior se hace enviando el valor de la tasa de transmisión, así como el tipo de recepción y transmisión, ya sea esta síncrona o asíncrona.

Habiendo inicializado todo, se puede empezar con la parte central del programa, en la cual como ya se menciono ser realiza un barrido continuó de teclado y una verificación del puerto seria, para ver si no se recibe nada, de ser así se la da un adecuado tratamiento al dato.

La rutina de barrido de teclado es muy sencilla, pues consiste en introducir un 10 al acumulador y compararlo con el puerto que está conectado al teclado, si la comparación dicta que son iguales, esto indica que se presionó una tecla de la primera columna, de no ser así entonces se rota un bit lo que está en el acumulador, de tal suerte que ahora se tendrá un 20 y se realiza nuevamente la comparación y así sucesivamente hasta que se encuentre una tecla presionada, una vez que se detecta, se envía al USART, que lo mandará al procesador.

La rutina de exploración del puerto serial es un poco más complicada, pues consta de la verificación de lo que se recibe, esto se hace verificando un determinado bit del comando recibido, mediante la instrucción BTFSS ó BTFSC, de tal suerte que así se distinguirá entre comando o caractrer o la sección de pantalla en la que se trabajara, para que así se pueda proseguir en el desarrollo de la rutina, pues si no se da la segunda recepción en la cual va inmerso el dato, después de un tiempo se regresará al inicio y se dará el dato por perdido

A lo largo del programa se incluyen subrutinas necesarias para el correcto funcionamiento de este, estas son:

ENABLE.- En esta subrutina se habilitan y deshabilitan los enables del apantalla de tal manera que no es necesario escribir las habilitaciones en el programa, ahorrando espacio con esto

BUSY_FLAG.- Dentro de la cuál se verifica que la pantalla esté desocupada y así poder seguir trabajando, pues si estuviese ocupada y se envía un carácter simplemente se perdería

ESCRIBE.- En esta parte se recibe el caracter y se envía a la pantalla

RETRASOS.- Estas subrutinas son simplemente para ser introducidas en secciones del programa en la que se necesite una espera, estas subrutinas se realizan simplemente con decrementos repetitivos, que al ser sumados dan el tiempo esperado

RECIBE y TRANSMITE.- Son simplemente parte de la verificación de banderas del USART y se colocan como subrutinas para ser llamadas dentro del programa y no escribir esta verificación en él, ahorrando de está forma espacio que es innecesario ocupar.

A continuación se muestra la estructura final del programa, con las subrutinas antes mencionadas y los comentarios en el desarrollo de todo.

```
;********************** SE INCLUYE LIBRERIAS Y SE DECLARAN VARIABLES AUXILIARES **************

  INCLUDE P16C65B.INC

TEMP1           EQU 0X20
TEMP2           EQU 0X22
TEMP3           EQU 0X24
TEMP4           EQU 0X26
TIEMPO          EQU 0X28
DECREM          EQU 0X30
TECLA           EQU 0X32
DECREM2         EQU 0X34
SECCION         EQU 0X36
ELECCION        EQU 0X38
DAPANTA         EQU 0X40
ESPERA          EQU 0X42
ELEGIDO         EQU 0X44
BKL             EQU 0X46
ALMACEN_TEMP    EQU 0X48
AUX_TEMP        EQU 0X50

;************************************** INICIA PROGRAMA PRINCIPAL ***************************************
                        ORG             0X05            ;DIRECCIÓN DE INICIO
;************************************************ CONFIGURA PUERTOS ***********************************************
```

```
                BSF     STATUS,RP0      ;SE ELIGE BANCO 1
                MOVLW   0X00            ;SE ELIGE
                MOVWF   TRISA           ;PORTA COMO SALIDA
                MOVLW   0X00            ;SE ELIGE
                MOVWF   TRISB           ;PORTB COMO SALIDA
                MOVLW   0X0F            ;SE CARGA EL PUERTO D
                MOVWF   TRISD           ;COMO ENTRADAS Y SALIDAS
                BCF     STATUS,RP0      ;SE ELIGE BANCO 0
;******************************************** CONFIGURA PANTALLA ***********************************************
                MOVLW   0X0F            ;SE CARGA UN15
                MOVWF   TIEMPO          ;EN VARIABLE AUXILIAR
INI1            CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
                DECFSZ  TIEMPO,1        ;SE DECREMENTO HASTA QUE
                GOTO    INI1            ;SE REPITA 15 VECES RETRASO
                MOVLW   0X38            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                MOVLW   0X04            ;SE CARGA UN 5
                MOVWF   TIEMPO          ;EN UNA VARIABLE AUXILIAR
INI2            CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
                DECFSZ  TIEMPO,1        ;SE DECREMENTE HASTA QUE
                GOTO    INI2            ;SE REPITA 5 VECES RETRASO
                MOVLW   0X38            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                CALL    RETRASO         ;LLAMA ESPERA DE 1 MS
                MOVLW   0X38            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
                MOVLW   0X38            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
                MOVLW   0X0C            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
                MOVLW   0X01            ;NUMERO DE COMANDO A ENVIAR
                MOVWF   PORTB           ;AL PUERTO B
                CALL    ENABLE          ;SE LLAMA SUBRUTINA DE ENABLE
                CALL    RETRASO         ;SE LLAMA RUTINA DE ESPERA
```

```
                    CALL    RETRASO     ;SE LLAMA RUTINA DE ESPERA
                    MOVLW   0X06        ;NUMERO DE COMANDO A ENVIAR
                    MOVWF   PORTB       ;AL PUERTO B
                    CALL    ENABLE      ;SE LLAMA SUBRUTINA DE ENABLE
                    CALL    RETRASO     ;SE LLAMA RUTINA DE ESPERA
;************************************ CONFIGURA USART E INTERRUPCIONES*******************************
                    MOVLW   0X19        ;SE CARGA UN 25 EN HEXADECIMAL
                    BSF     STATUS,RP0  ;SE SELECCIONA BANCO 1
                    MOVWF   SPBRG       ;SE INICIA SPBRG CON 51
                    MOVLW   0X20        ;SE INICIALIZA LA TRANSMISIÓN
                    MOVWF   TXSTA       ;EN TXSTA
                    BCF     STATUS,RP0  ;SE SELECCIONA BANCO 0
                    MOVLW   0X90        ;SE INICIALIZA RECEPCIÓN
                    MOVWF   RCSTA       ;EN RCSTA
                    MOVLW   0X40        ;HABIL ITAINTER. GLOBALES Y PERIF.
                    MOVWF   INTCON      ;EN EL INTCON
                    MOVLW   0X30        ;HABILITA INTERRUPCIONES TX Y RX
                    BSF     STATUS,RP0  ;SE SELECCIONA BANCO 1
                    MOVWF   PIE1        ;EN EL PIE1
;********************************* VERIFICA RECEPCION EN USART ***************************************
RECSER              CALL    RECIBE      ;LLAMA SUBRUT. QUE VERIFICA RECEP.
                    BTFSC   PIE1,RCIE   ;VERIFICA RCIE SOLO SI RCIF SE ACT.
                    GOTO    RECIBIO     ;SI ESTA EN UNO VA A RECIBIO
;********************************************** BARRE TECLADO ****************************************
TECLADO             BCF     STATUS,RP0  ;SI ES CERO SE CAMBIA A BANCO 0
                    MOVLW    0X10       ;SE PONE UN 10 EN ACUMULADOR
                    MOVWF   TECLA       ;SE CARGA EN REGISTRO TEMPORAL
                    MOVLW   0X05        ;CARGA UN 5 EN ACUMULADOR
                    MOVWF   DECREM      ;LO ENVÍA A VARIABLE TEMPORAL
ROTADO              MOVF    TECLA,0     ;SE MUEVE EL CONTENIDO DE TECLA
                    MOVWF   PORTD       ;SE MANDA AL PUERTO D
                    MOVF    PORTD,0     ;ESTE SE REGRESA AL ACUMULADOR
                    ANDLW   0X0F        ;SE LE APLICA UNA AND CON 0F
                    BTFSC   STATUS,2    ;SI EL RESULTADO ES 0 NO SE PRESION
                    GOTO    ROTAR       ;TECLA Y VA A ETIQUETA ROTAR
                    GOTO    SUMA        ;SI ES DIFERENTE DE 0 VA A SUMA
ROTAR               RLF     TECLA,1     ;ROTA UN BIT A LA IZQUIERDA
                    DECFSZ  DECREM,1    ;DISMINUYE EN UNO EL REG. DECREM
                    GOTO    ROTADO      ;VA A ROTADO, SI NO HA LLEGADO A 0
                    GOTO    RECSER      ;SI LLEGO A CERO, REGRESA AL INICIO
SUMA                IORWF   TECLA,1    ;SE APLICA UNA OR PARA ENCONTRAR LA
TECSER              CALL    TRANSMITE   ;TECLA PRESIONADA Y LLAMA SUBRUT.
```

```
                BTFSS   PIE1,TXIE      ;SE VERIFICA BANDERAS DE TRANSM.
                GOTO    TECSER        ;HASTA QUE SE PUEDA DAR TRANSM.
                BCF     STATUS,RP0    ;CAMBIA DE BANCO
                MOVF    TECLA,0       ;MUEVE EL VALOR  TECLA PRESIONADA
                MOVWF   TXREG         ;AL BUFFER DE TRANSMISIÓN
                MOVLW   0X05          ;SE CARGA UN 5 EN UNA VARIABLE
                MOVWF   DECREM2       ;AUXILIAR PARA LOGRAR UN RETRASO
OTRA            CALL    RETRASO2      ;DE MEDIO SEGUNDO, ANTES DE
                DECFSZ  DECREM2,1     ;REGRESAR AL INICIO Y VERIFICAR SI
                GOTO    OTRA          ;RECIBE ALGO AL PUERTO SERIE O SI
                GOTO    RECSER        ;HAY OTRA TECLA PRESIONADA
;********************************* VERIFICA LO QUE SE RECIBIO EN USART **********************************
RECIBIO         BCF     STATUS,RP0    ;CAMBIA AL BANCO 0
                MOVF    RCREG,0       ;MUEVE EL CONTENIDO DE RCREG A W
                MOVWF   ELECCION      ;LO COLOCA EN UN REGISTRO AUXILIAR
                BTFSC   ELECCION,3    ;VERIFICA EL BIT 3 PARA VER SI ES UNA
                GOTO    FUNESP        ;FUNCIÓN ESPECIAL O TIPO CARÁCTER
                BTFSS   ELECCION,1    ;VERIFICA EL BIT 1 PARA VER A QUE
                GOTO    PANT1         ;SECCIÓN DE PANTALLA SE DIRIGE
                GOTO    PANT2         ;SI ES 0 ES PANTALLA 1, SI ES 1 ES LA 2
PANT1           MOVLW   0X01          ;CARGA VALOR QUE HABILITA PANTA 1
                MOVWF   SECCION       ;EN UNA VARIABLE AUXILIAR Y VA A LA
                GOTO    CARCOM        ;SECCIÓN QUE DEFINE QUE SE RECIBE
PANT2           MOVLW   0X08          ;CARGA  VALOR QUE HABILITA PANTA 2
                MOVWF   SECCION       ;EN VARIABLE AUXILIAR
CARCOM          BTFSS   ELECCION,2    ;VERIFICA  BIT 2 PARA VER QUE RECIBE
                GOTO    COMANDO       ;SI ES 0 ES UN COMANDO Y VA A ETIQ.
                GOTO    CARACTER      ;SI ES 1 ES UN CARÁCTER Y VA A ETIQ.
COMANDO         MOVLW   0X00          ;CARGA EL VALOR DE RS EN MODO
                MOVWF   ELEGIDO       ;COMANDO EN UNA VARIABLE AUXILIAR
                GOTO    LLEGDAT       ;VA A ETIQUETA LLEGDAT
CARACTER        MOVLW   0X04          ;CARGA EL VALOR DE RS EN MODO
                MOVWF   ELEGIDO       ;CARACTER EN UNA VARIABLE AUXILIAR
LLEGDAT         CALL    RECIBE        ;LLAMA SUBRUT. QUE VERIF. BANDERA
                BTFSC   PIE1,RCIE     ;VERIFICA RCIE SOLO SI RCIF SE ACTIVA
                GOTO    DATO          ;VA A  TRATAMIENTO DE DATO
                BCF     STATUS,RP0    ;SELECCIONA BANCO 0
                MOVLW   0X02          ;SI NO SE HA RECIBIDO DATO ESPERA
                MOVWF   DECREM        ;UN TIEMPO A QUE LO HAGA
LLEGA           CALL    RETRASO2      ;SE LLAMA SUBRUT. QUE DURA UN SEG.
                CALL    RECIBE        ;VUELVE A VER SI SE RECIBIÓ EL DATO
                BTFSC   PIE1,RCIE     ;VERIFICA RCIE SOLO SI RCIF SE ACTIVA
```

```
                GOTO    DATO        ;SI YA, VA A ETIQUETA, SI NO
                BCF     STATUS,RP0  ;DECREMENTA LA VARIABLE AUXILIAR
                DECFSZ  DECREM,1    ;PARA QUE SE VEA DE NUEVO SI LLEGO
                GOTO    LLEGA       ;O PARA REGRESAR A INICIO Y ESPERAR
                BCF     STATUS,RP0  ;NUEVO O TECLA, PUES LO
                GOTO    RECSER      ;RECIBIDO SE PERDIÓ
DATO            BCF     STATUS,RP0  ;SELECCIONA BANCO 0
                MOVF    RCREG,0     ;MUEVE EL CONTENIDO DE RCREG A W
                MOVWF   DAPANTA     ;LO COLOCA EN UN REGISTRO AUXILIAR
                CALL    BUSY_FLAG   ;VERIFICA BANDERA DE OCUPADO
                CALL    ESCRIBE     ;LLAMA SUBRUT. QUE LO MANDA A LCD
                GOTO    RECSER      ;VA A CONFIRMACIÓN TOTAL
FUNESP          BTFSC   ELECCION,1  ;VERIFICA QUE TIPO DE FUN. ESPEC. ES
                GOTO    BUZZER      ;SI ES 0 ES TIMBRE, SI ES 1 ENCENDIDO
BACKL           BTFSS   ELECCION,0  ;VERIF. SI DEBE ENCENDER O APAGAR
                GOTO    BKLON       ;EL LCD Y VA ETIQUETA QUE LE
                GOTO    BKLOFF      ;CORRESPONDA
BKLON           MOVLW   0X08        ;CARGA VALOR QUE PRENDE PANTALLA
                MOVWF   BKL         ;EN REGISTRO AUXILIAR
                CALL    BACKLIGHT   ;LLAMA SUBRUT. QUE LO ENVÍA A LCD
                GOTO    RECSER      ;VA A CONFIRMACIÓN TOTAL
BKLOFF          MOVLW   0X00        ;CARGA VALOR QUE APAGA PANTALLA
                MOVWF   BKL         ;EN REGISTRO AUXILIAR
                CALL    BACKLIGHT   ;LAMA SUBRUTINA QUE LO ENVÍA A LCD
                GOTO    RECSER      ;VA A CONFIRMACIÓN TOTAL
BUZZER          BTFSS   ELECCION,0  ;VERIFICA SI SE DEBE ENCENDER O
                GOTO    BZON        ;APAGAR EL TIMBRE Y VA A LA SECCIÓN
                GOTO    BZOFF       ;QUE LO REALIZA
BZON            BSF     PORTA,5     ;ENCIENDE TIMBRE PONIENDO 1 EN PA5
                GOTO    RECSER      ;VA A CONFIRMACIÓN TOTAL
BZOFF           BCF     PORTA,5     ;APAGA TIMBRE
                GOTO    RECSER      ;VA A  CARACTER, COMANDO O TECLA

;*********************************************  SUBRUTINAS **********************************************
ENABLE          MOVLW   0X09        ;HABILITA ENABLE 1 Y 2
                MOVWF   PORTA       ;EN EL PUERTO A
                CLRW                ;Y LUEGO LOS DESHABILITA
                MOVWF   PORTA       ;LIMPIANDO EL PUERTO
                RETURN              ;REGRESO DE SUBRUTINA

BUSY_FLAG       BCF     STATUS,RP0  ;CAMBIA A BANCO 0
                BCF     PORTA,2     ;PASA RS A COMANDO
```

```
                BSF     PORTA,1     ;SE PONE LCD EN LEER
                BSF     STATUS,RP0  ;SE ELIGE BANCO 1
                MOVLW   0XFF        ;SE DEFINE
                MOVWF   TRISB       ;PORTB COMO ENTRADA
                BCF     STATUS,RP0  ;SE ELIGE EL BANCO 0
CONTINUA        MOVF    SECCION,0   ;MUEVE EL CONTENIDO DE SECCION A
                IORWF   PORTA,1     ;ACUMULAD. Y LO MANDA A PUERTO A
                MOVF    PORTB,0     ;MUEVE PORTB A W
                BCF     PORTA,0     ;DESACTIVA ENABLE 1
                BCF     PORTA,3     ;Y ENABLE 2
                ANDLW   0X80        ;VERIFICA QUE EL BUSY
                BTFSS   STATUS,2    ;FLAG ESTE DESOCUPADO, SI NO
                GOTO    CONTINUA    ;ESPERA A QUE SE DESOCUPE
                BCF     PORTA,1     ;SE PONE LCD EN ESCRIBIR
                RETURN              ;REGRESO DE SUBRUTINA

ESCRIBE         BCF     PORTA,1     ;PONE R/W EN ESCRIBE
                MOVF    ELEGIDO,0   ;SE PONE RS EN COMANDO O
                IORWF   PORTA       ;CARACTER SEGÚN SE DEFINIÓ
                BSF     STATUS,RP0  ;SE ELIGE BANCO 1
                MOVLW   0X00        ;SE DEFINE
                MOVWF   TRISB       ;PORTB COMO SALIDA
                BCF     STATUS,RP0  ;SE ELIGE BANCO 0
                MOVF    DAPANTA,0   ;MUEVE REGISTRO A W
                MOVWF   PORTB       ;MUEVE SU CONTENIDO A PORTB
                MOVF    SECCION,0   ;SE HABILITA LA SECCIÓN DE
                IORWF   PORTA,1     ;PANTALLA ELEGIDA
                NOP                 ;TIEMPO DE ESTABILIZACIÓN
                BCF     PORTA,0     ;SE DESHABILITA EL ENABLE 1
                BCF     PORTA,3     ;Y ENABLE 2
                BSF     STATUS,RP0  ;SE ELIGE BANCO 1
                MOVLW   0XFF        ;SE DEFINE POR
                MOVWF   TRISB       ;SEGURIDAD PORTB COMO ENTRADA
                BCF     STATUS,RP0  ;CAMBIO DE BANCO
                RETURN              ;REGRESO DE SUBRUTINA

RETRASO         MOVLW   0XFA        ;SE MANDA UN 250 A LA LOCALIDAD
                MOVWF   TEMP1       ;TEMP1
OTRA_VEZ        NOP                 ;UN CICLO SIN OPERACIÓN, PARA
                DECFSZ  TEMP1,1     ;QUE EL LOOP TENGA 4 CICLOS, QUE
                GOTO    OTRA_VEZ    ;REPETIDOS 250 VECES DARÁ UN
                RETURN              ;RETRASO DE UN MILISEGUNDO
```

```
RETRASO2        MOVLW       0XFF        ;SE CARGA CON 255 UNA LOCALIDAD
                MOVWF       TEMP2       ;EN EL BANCO DE PROPÓSITO GENERAL
                MOVLW       0XFF        ;SE CARGA CON 255 UNA LOCALIDAD
                MOVWF       TEMP3       ;EN EL BANCO DE PROPÓSITO GENERAL
DE_NUEVO        DECFSZ      TEMP2,1     ;EL LOOP DE 3 CICLOS DA UN RETRASO
                GOTO        DE_NUEVO    ;DE 700 MICROSEGUNDOS APROXIM.
                DECFSZ      TEMP3,1     ;LIGADO AL ANTERIOR, ESTE LOOP DURA
                GOTO        DE_NUEVO    ;100MILISEGUNDOS APROXIM.
                RETURN                  ;REGRESO DE SUBRUTINA

RECIBE          BCF         STATUS,RP0  ;SELECCIONA BANCO 0, VERIFICA
                BTFSC       PIR1,RCIF   ; SI RCIF RECIBE INTERRUPCIÓN
                BSF         STATUS,RP0  ;VA A BANK1 SI RCIF ESTA ACTIVADA
                RETURN
TRANSMITE       BCF         STATUS,RP0  ;SELECCIONA BANK 0
                BTFSC       PIR1,TXIF   ;VERIFICA PARA INTERRUPCIÓN TXIF
                BSF         STATUS,RP0  ;VA A BANK1 SI TXIF ESTA ACTIVADA
                RETURN

BACKLIGHT       BSF         STATUS,RP0  ;SE ELIGE BANCO 1
                MOVLW        0X00       ;SE DEFINE
                MOVWF       TRISB       ;PORTB COMO SALIDA
                BCF         STATUS,RP0  ;SE ELIGE BANCO 0
                BCF         PORTA,1     ;SE DEFINE ESCRIBIR
                BCF         PORTA,2     ;SE DEFINE COMANDO
                MOVF        BKL,0       ;SE ENVÍA EL CONTENIDO
                MOVWF       PORTB       ;DE REGISTRO AUXILIAR A PUERTO B
                MOVLW       0X09        ;SE APLICA A AMBAS SECCIONES
                MOVWF       PORTA       ;DE LA PANTALLA
                CLRW                    ;SE DESHABILITAN LO ENABLES
                MOVWF       PORTA       ;PARA RECIBIR INSTRUCCIÓN
                RETURN                  ;REGRESO DE SUBRUTINA
                END                     ;FIN DE PROGRAMA
```

5.6 Simulación en MPLAB

EL MPLAB es un Entorno de Desarrollo Integrado (Integrated Development Environment, IDE) que corre en Windows , mediante el cual se pueden desarrollar aplicaciones para los microcontroladores de las familias PIC 16/17. EL MPLAB que permite

escribir, depurar y optimizar los programas (firmware) de sus diseños con PIC 16/17. EL MPLAB incluye un editor de texto, un simulador y un organizador de proyectos. Además, el MPLAB soporta el emulador PICMASTER y a otras herramientas de desarrollo de Microchip como el PICSTART Plus. Con el MPLAB se puede:

.-Depurar sus programas fuente.

.-Detectar errores automáticamente en sus programas fuente para editarlos.

.-Depurar los programas utilizando puntos de corte (breakpoints) mediante valores de los registros internos.

.-Observar el flujo del programa con el simulador MPLAB -SIM, ó seguirlo en tiempo real utilizando el emulador PICMASTER.

-Realizar medidas de tiempo utilizando un cronómetro.

-Mirar variables en las ventanas de observación.

-Encontrar respuestas rápidas a sus preguntas, utilizando la Ayuda en línea del MPLAB.

Herramientas:

El Organizador de Proyectos (Proyect Manager) Es parte fundamental de MPLAB. Sin crear un proyecto no se puede realizar depuración simbólica. Con el Organizador de Proyectos (Proyect manager) se pueden realizar las siguientes operaciones:

.-Crear un proyecto.

.-Agregar un archivo de programa fuente de proyecto.

.-Ensamblar o compilar programas fuente.

.-Editar programas fuente.

.-Reconstruir todos los archivos fuente, o compilar un solo archivo.

.-Depurar su programa fuente.

Software ensamblador.- El software ensamblador que presenta Microchip viene en dos presentaciones, una para entorno DOS llamado MPASM.EXE y la otra, para entorno Windows llamado MPASMWIN.EXE Las dos presentaciones soportan a TODOS los microcontroladores de la familia PIC de Microchip.

El conjunto de instrucciones de los microcontroladores PIC es en esencia la base del lenguaje ensamblador soportado por este software.

Una vez instalado adecuadamente el MPLAB, para realizar la simulación de un programa deben seguirse los siguientes pasos:

1. Hacer doble click en el ícono correspondiente a MPLAB.
2. Crear el archivo fuente correspondiente (menú File, New Source).
3. Salvar el archivo (con extensión .ASM) una vez terminada la edición (menú FILE...Save).
4. Debe a continuación crearse un nuevo proyecto (menú Project...New Project).
5. Cuando aparezca la ventana de New Project editar las cajas de texto: Project path and Name y Development Mode, hacer click en OK.
6. En la siguiente ventana Edit Project, hacer click en la sección Non-project files sobre el nombre del archivo fuente realizado en los pasos 2 y 3.
7. Hacer click en el botón add y luego de que éste aparezca en la sección Project Files hacer click sobre el botón OK.
8. Salvar el proyecto (en el menú Project, Save project).
9. Realizar la construcción de todo el proyecto (menú Project, Build All).
10. En esta etapa se realiza en forma automática el ensamble del programa fuente y el vaciado de éste en memoria de simulación. El proceso de ensamble generará un archivo de errores en caso de que estos existan, sí es así deben corregirse directamente sobre el archivo fuente, salvar las correcciones y reconstruir el proyecto (menú Project, Build All). En esta etapa del proceso ya se tiene el entorno listo para la simulación.

La pantalla se divide en varias secciones:

1. Barra de título: Se observa el nombre del proyecto
2. Barra de menus: Acceso a las diferentes opciones del entorno
3. Barra de herramientas: Cada ícono ejecuta las acciones correspondientes
4. Barra de estados: Indica el estado del entorno y sus ventanas

Simulación:

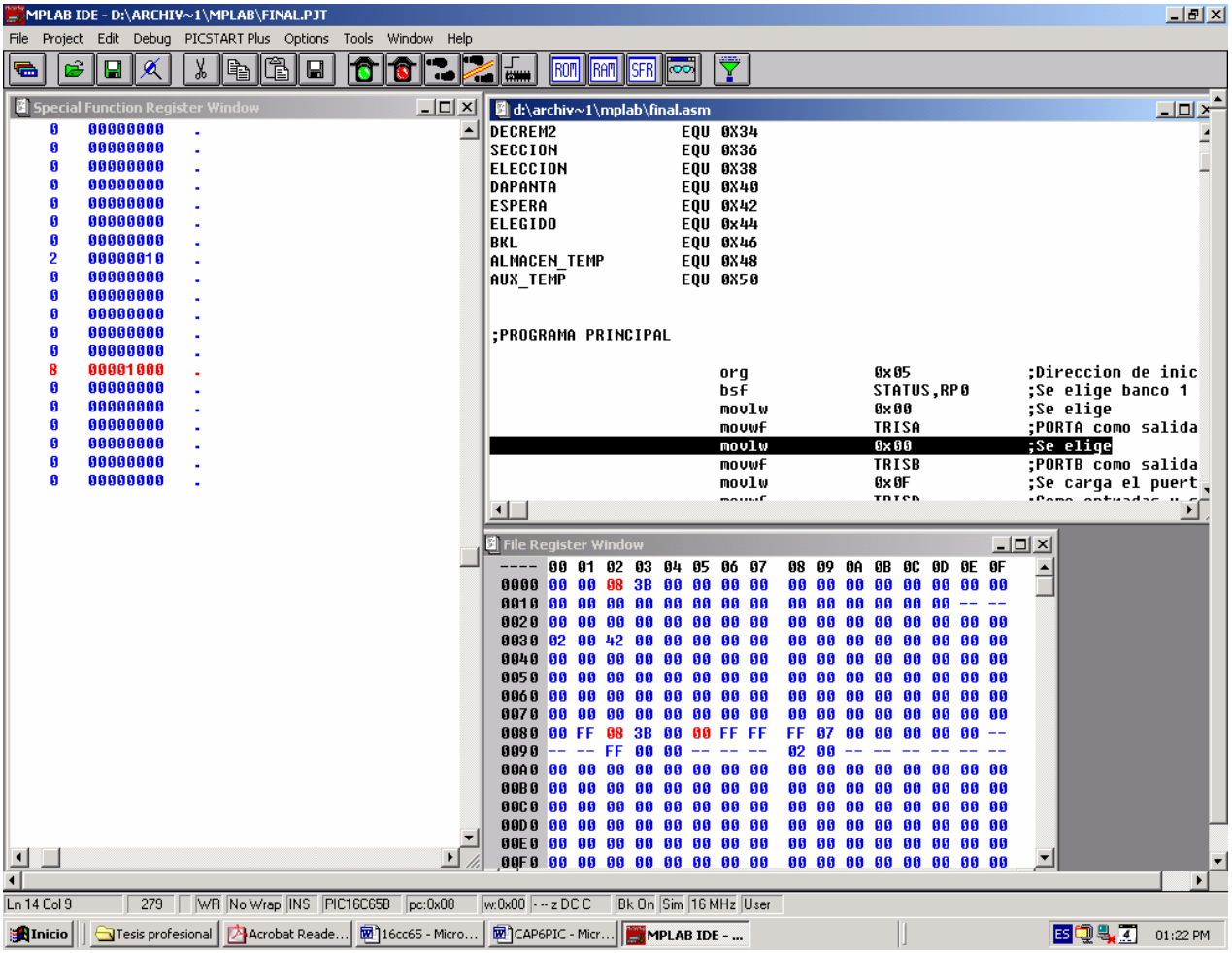
1. Resetear el procesador (menú Debug...Run...Reset) ó con F6 ó con el ícono correspondiente en la barra de herramientas.

2. Crear una nueva ventana donde se incluyan las variables que queremos tener en cuenta (Window...New Watch Window)

3. Empezar a correr paso a paso el programa haciendo el seguimiento detallado de todos y cada uno de los pasos (menú Debug...Run...Step) ó con la tecla F7 ó con el ícono correspondiente en la barra de herramientas.

El proceso de simulación permite detectar y corregir problemas de lógica, problemas de situaciones que no se hayan tenido en cuenta que son errores que no pueden ser detectados en el momento del ensamble del programa.

A continuación se muestra la pantalla en la que se realizó la simulación del programa desarrollado.



5.7 Implementación del programa al PIC

La implementación del programa, después de haber sido verificado, ensamblado y simulado en MPLAB, al PIC se realiza con ayuda del PicStart Plus que es un sistema de desarrollo de la tecnología de microchip que provee al ingeniero del desarrollo una

herramienta de diseño de costo altamente flexible, bajo MCU para todos los dispositivos 8-bit de PICMicro (la INMERSIÓN empaqueta hasta 40 pines).

Se necesita una PC conectada para funcionar. El software libre MPLAB de Windows permite que el usuario desarrolle y envíe fácilmente software abajo al Picstart Plus. Programa los dispositivos insertados con su software.

Es un programador de desarrollo y no se recomienda para el uso en un ambiente de la producción. Para los funcionamientos de producción se necesita un programador de producción tal como el Promate II .

El Picstart Plus requiere MPLAB, un cable RS232, una fuente de alimentación y un dispositivo de muestra.

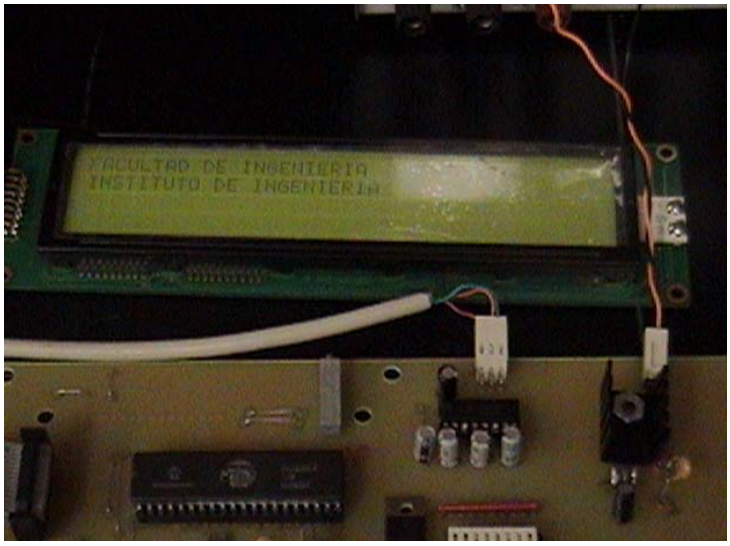
Se conecta simplemente el PicStart a la PC mediante el cable RS232 o DB9, se alimenta con una fuente, se coloca el PIC en la sección indicada y se da la orden de grabación en el MPLAB.

La siguiente es una imagen del PicStart Plus, en la cuál se observa parte de los cables de alimentación y de conexión con la PC y la sección donde se inserta el PIC para su grabación.

Después de ser implementado el programa al PIC se verifica su funcionamiento y si no es el correcto se puede modificar el programa hasta que funcione óptimamente y volver a grabarlo en el PIC, antes de esta acción el PIC tuvo que ser borrado, como se mencionó en la sección de características del PIC, una de ellas es que puede ser borrado mediante su exposición a luz

fluorescente, entonces antes de ser grabado de nuevo se insertaba un dispositivo como el siguiente.

Una vez que se implemento el programa corregido, se verifico su optima función y como dicen que una imagen vale más que mil palabras, se muestra el desplegado en pantalla.



CAPÍTULO VI DESARROLLO DEL SOFTWARE NECESARIO PARA EL CONTROLADOR

6.1 Introducción a Labview

El Laboratorio de instrumentación virtual se establece con la intención de crear un espacio para desarrollar equipos y generar investigación dentro de la rama de instrumentación electrónica.

Así dentro del ámbito de la academia, el laboratorio (Labview) busca desarrollar proyectos en instrumentación que beneficien y sean útiles a las diferentes áreas de la electrónica en que se necesiten mediciones de diversas magnitudes físicas. De este modo se involucran las nuevas generaciones de estudiantes en el manejo de computadoras para la instrumentación y control de eventos, teniendo la posibilidad de tener arquitecturas abiertas.

Labview es un lenguaje de programación gráfico para el diseño de sistemas de adquisición de datos, instrumentación y control. Labview permite diseñar interfaces de usuario mediante una consola interactiva basada en software.

Se puede diseñar especificando su sistema funcional, el diagrama de bloques o una notación de diseño de ingeniería. Labview es a la vez compatible con herramientas de desarrollo similares y puede trabajar con programas de otra área de aplicación, como por ejemplo Matlab. Tiene la ventaja de que permite una fácil integración con hardware, específicamente con tarjetas de medición, adquisición y procesamiento de datos (incluyendo adquisición de imágenes).

Programación rápida: esté lenguaje de programación permite desarrollar de una forma más rápida cualquier aplicación, especialmente de instrumentación, en comparación con lenguajes de programación tradicionales basados en texto, sin embargo si se desea una aplicación sencilla como un programa que sume dos números, definitivamente construirlo bajo Labview es más demorado y tedioso, sería más sencillo mediante un programador de texto donde simplemente se incluirá una línea. Pero para un programa más complejo se puede diseñar un prototipo y modificarlo de una manera más rápida con Labview gracias a que es un lenguaje programación gráfico.

Características de Labview.- Una de las principales características de Labview es su modularidad, es decir, la capacidad de utilizar bloques funcionales para la definición de la especificación.

Labview permite conectarse a otras aplicaciones mediante un intercambio de datos como Active X, librerías dinámicas, bases de datos, Excel y/o a protocolos de comunicación como DataSocket, TCP/IP, UDP, RS-232, entre otras.

Una característica de cada aplicación o función consiste en que se puede utilizar en cualquier parte de otro programa, dándole a Labview una estructura Jerárquica.

Otra característica se encuentra en el flujo de datos, que muestra la ejecución secuencial del programa, es decir, una tarea no se inicia hasta no tener en todas sus variables de entrada información o que las tareas predecesoras hayan terminado de ejecutarse. Debido al lenguaje gráfico el compilador con que cuenta Labview es más versátil ya que sobre el mismo código de programación se puede ver fácilmente el flujo de datos, así como su contenido.

Labview también puede ser un programa en tiempo real donde la aplicación trabaja sin la necesidad de otro sistema operativo, este programa denominado Labview RT viene con su propio Kernel que se encarga de la administración de las tareas. Mediante un constructor de aplicaciones también es posible generar un archivo que puede ejecutarse fuera de Labview.

Aplicaciones de Labview.- Labview tiene su mayor aplicación en sistema de medición, como monitoreo de procesos y aplicaciones de control, un ejemplo de esto pueden ser sistemas de monitoreo en transportación, Laboratorios para clases en universidades, procesos de control industrial.

Labview es muy utilizado en procesamiento digital de señales (wavelets, FFT, Distorsión Armónica Total TDH), procesamiento en tiempo real de aplicaciones biomédicas, manipulación de imágenes y audio, automatización, diseño de filtros digitales, generación de señales, entre otras, etc.

Programación gráfica con Labview.- Cuando se diseñan programas con Labview se está trabajando siempre bajo algo denominado VI, es decir, un instrumento virtual, se pueden crear VI a partir de especificaciones funcionales que se diseñen. Este VI puede utilizarse en cualquier otra aplicación como una subfunción dentro de un programa general. Los VI's se caracterizan por: ser un cuadrado con su respectivo símbolo relacionado con su funcionalidad, tener una interfaz con el usuario, tener entradas con su color de identificación de dato, tener una o varias salidas y por su puesto ser reutilizables.

En el ambiente de trabajo de Labview existen dos paneles, el panel frontal y el panel de programación; en el panel frontal se diseña la interfaz con el usuario y en el panel de programación se relacionan los elementos utilizados en la interfaz mediante operaciones que determinan en sí como funciona el programa o el sistema, exactamente es la parte donde se realizan las especificaciones funcionales.

En el panel de programación se puede diseñar de manera gráfica y como si fuera un diagrama de bloques el funcionamiento de su sistema. La programación gráfica se basa en la realización de operaciones mediante la asignación de iconos que representen los datos numéricos e iconos que representan los procedimientos que se deben realizar (VI's), con estos iconos y mediante una conexión simple como lo es una línea recta se enlazan para determinar una operación y/o una función.

Al diseñar el programa de forma gráfica, se hace visible una programación orientada al flujo de datos, donde se tiene una interpretación de los datos también de forma gráfica, por ejemplo un dato booleano se caracteriza por ser una conexión verde, cada tipo de dato se identifica con un color diferente dentro de Labview, también es necesario tener en cuenta que cuando se realiza una conexión a un VI esta conexión se identifica por un tipo de dato especifico, que debe coincidir con el tipo de dato de la entrada del VI (aunque esto no necesariamente es cierto ya que puede haber varios tipos de datos conectados de VI a VI, además de que un arreglo de datos ``cluster`` puede albergar varios tipo de variables) permitiendo una concordancia en el flujo de datos; no siempre el tipo de dato de la entrada del VI es el mismo que el de la salida, pero sin embargo para la mayoría de los casos si se cumple. El flujo de datos va de izquierda a derecha en el panel de programación y esta determinado por las operaciones o funciones que procesan los datos. Es fácil observar en el panel de programación como se computan los datos en cada parte del programa cuando se realiza una ejecución del programa paso a paso.

En Labview las variables se representan mediante una figura tanto en el panel frontal como en el panel de programación, de esta forma se puede observar su respuesta en la interfaz del usuario y en el flujo de datos de código del programa. Otros objetos como gráficas y accesos directos a páginas web cumplen estas misma condiciones.

Modularidad.- Cuando se ha diseñado la aplicación se puede definir como un VI, de esta forma se puede reutilizar en un nuevo programa, esto se hace mediante la selección

del diagrama funcional y la opción crear VI, también se puede diseñar el símbolo que represente su aplicación y definir las entradas y las salidas, esto consiste en la asignación de cada variable de entrada a un sub-bloque dentro del bloque general que representa el símbolo.

Compilador Gráfico.- En Labview se realiza la compilación bajo el principio básico de programación de forma gráfica. Debido a que se trabaja con flujo de datos es fácil ver el comportamiento de estos a través del programa, teniendo la posibilidad de ver el depurador ejecutarse paso a paso, se puede observar como cambian los datos en cualquier parte del programa, y como van pasando de una función a otra dentro del diagrama de bloques. El compilador optimiza internamente el código manteniendo una buena velocidad en la ejecución del programa.

Controles.- La programación G (gráfica) de Labview consta de un panel frontal y un panel de código como se menciono antes. En el panel frontal es donde se diseña la interfaz de usuario y se ubican los controles e indicadores. En el panel de código se encuentran las funciones. Cada control que se utiliza en la interfaz tiene una representación en el panel de código, igualmente los indicadores necesarios para entregar la información procesada al usuario tienen un icono que los identifica en el panel de código o de programación.

Enseguida se ilustra la Paleta de controles.

Los controles pueden ser booleanos, numéricos, strings, un arreglo matricial de estos o una combinación de los anteriores; y los indicadores pueden ser como para el caso de controles pero pudiéndolos visualizar como tablas, gráficos en 2D o 3D, browser, entre otros.

Funciones.- Las funciones pueden ser VIs prediseñados y que pueden ser reutilizados en cualquier aplicación, estos bloques funcionales constan de entradas y salidas, igual que en un lenguaje de programación estándar las funciones procesan las entradas y entregan una o varias salidas, estos VI pueden también estar conformados de otros subVIs y así sucesivamente, de está forma se pueden representar como un árbol genealógico donde un VI se relaciona o depende de varios SubVIs.

Labview tiene VIs de adquisición de datos e imágenes, de comunicaciones, de procesamiento digital de señales, de funciones matemáticas simples, hasta funciones que utilizan otros programas como Matlab o HiQ para resolver problemas, otras más complejas como "nodos de formula" que se utilizan para la resolución de ecuaciones editando directamente estas como en lenguajes de programación tradicionales y definiendo las entradas y las salidas.

Enseguida se ilustra la Paleta de funciones

Labview también se puede utilizar para graficar en tres dimensiones, en coordenadas polares y cartesianas, tiene disponibles herramientas para análisis de circuitos RF como la Carta de Smith, tiene aplicaciones en manejo de audio y se puede comunicar con la tarjeta de sonido del computador para trabajar conjuntamente.

Entre sus muchas funciones especiales se encuentran las de procesamiento de imágenes, como capturar una imagen a través de una tarjeta de adquisición como la PCI-1408 (monocromática), analizarla y entregar respuestas que difícilmente otros sistemas realizarían.

Adquisición de datos.- Labview como su nombre lo indica es un lenguaje que se enfoca hacia el laboratorio, la realización de mediciones y por lo tanto la adquisición y análisis de datos. Mediante el uso de tarjetas es posible obtener señales análogas o digitales a partir de una conexión al bus PCI en una computadora, estas tarjetas se diferencian por el número de muestras por segundo que pueden realizar, por el número de bits (resolución) y por el número de canales que manejan. Por ejemplo una tarjeta de bajo costo como la PCI-1200 realiza 100KS/s, con una resolución de 8 bits para 3 puertos

digitales de entrada salida y dos de salida análoga, junto con 12 bits de resolución para 8 canales de entrada análogos, y puertos de temporización mediante una PIT (8253); la tarjeta esta conformada por PPIs, una PIT que usa la tarjeta para su programación y otra para disponibilidad del usuario y conversores digitales a análogo (DAC) y demás componentes básicos. La tarjeta se debe programar para definir los niveles de voltaje que debe manejar y si son bipolares o unipolares. Las características de la tarjeta se pueden configurar con la herramienta Measurement & Automation que se encuentra en el explorador de Windows como un icono principal. Esta herramienta permite verificar que tarjetas hay instaladas en el computador, esto solo para el caso de tarjetas de adquisición de datos o de imágenes de National Instruments. Labview también permite comunicarse con otras tarjetas mediante el puerto paralelo o serial.

Análisis de datos.- Labview es una herramienta que tiene bastante aplicación en el análisis de datos, se puede decir que dentro de los tópicos básicos: adquisición, análisis y presentación, el análisis de datos es la más importante y la más robusta. Desde luego siempre lo que se desea es analizar variables reales, como la temperatura, la velocidad, entre otras; por lo tanto es necesario convertir esas variaciones a algo que el computador pueda procesar, las tarjetas de adquisición realizan esa labor y la interfaz tiene la función de presentar esos datos ya analizados y procesados. El análisis de datos se convierte por lo tanto en una de las potencialidades de Labview ya que se puede manejar mas fácilmente la señal que se quiere analizar, ya sea análoga o digital (o también podría ser una imagen) y por ejemplo aplicarle un algoritmo que opere con la transformada de Fourier y obtener la respuesta de manera inmediata.

Como bastantes de las funciones utilizadas para procesamiento digital de señales ya están prediseñadas el problema de construir algoritmos se reduce solo a saber aplicarlos.

Presentación de datos.- Cuando se diseño por primera vez Labview este lenguaje no permitía diseñar de forma flexible la interfaz con el usuario, ahora se puede partir de la interfaz con el usuario para generar el código, esta nueva forma de programación que se basa en Labview viene integrada dentro de un nuevo paquete llamado Bridgeview, esto permite que el programador diseñe su instrumento y el programa se encargue de generar y optimizar el código. La interfaz de todas formas esta limitada a los controles e indicadores

que proporciona Labview, de cualquier manera se puede usar otro tipo de controles (controles Active X), o diseñar uno propio a partir de otro lenguaje de programación.

Iniciando a programar en Labview.- Al iniciar se debe tener en cuenta que siempre se trabaja en paralelo en el panel de programación y el panel frontal, además siempre se debe tener presente la paleta de herramientas que tiene lo necesario para realizar una programación gráfica, por el ejemplo la herramienta de conexión esta representada por un carrete. Para ejecutar y depurar el programa se tienen varios iconos en la parte superior de los dos paneles y que permiten iniciar la ejecución del programa, detenerlo, depurarlo paso a paso y efectuar las demás funciones de un depurador clásico. Una forma de conocer mas a Labview y comprender su potencialidad es conocer cada uno de los elementos que se presentan en la paleta de controles y la paleta de funciones, ya que da una versatilidad del programa, ayudando a conseguir mas fácilmente los elementos que se necesitan, además de ahorrar el tener que diseñar VIs que cumplan una tarea para la cual ya existe otro VI.

6.2 Análisis del módulo Compact Field Point de National Instruments

Si se necesita control embebido o realizar medidas en un espacio reducido, se deben considerar los productos E/S Compact FieldPoint de National Instruments. Desde la producción de los módulos controladores FieldPoint FP-2000, los desarrolladores de LabVIEW han sido capaces de crear aplicaciones y desplegarlas en controladores FieldPoint pequeños. Con Compact FieldPoint, LabVIEW se cuenta con un objetivo de despliegue más pequeño que es más favorable para aplicaciones embebidas que antes. Compact FieldPoint es un formato más pequeño y robusto de FieldPoint. El nuevo sistema aproximadamente la mitad del tamaño de FieldPoint, cuenta con un tablero sólido con terminación de señales de masa.

Todos los controladores pequeños de Compact FieldPoint ejecutan LabVIEW Real Time. Con LabVIEW Real Time, se puede usar la paleta de funciones estándar de LabVIEW para realizar análisis de datos, registro y almacenamiento de datos y comunicación por red. Simplemente se desarrolla y depura su sistema en Windows y luego lo descarga y ejecuta su programa en el procesador dedicado en el controlador Compact FieldPoint.

Compact FieldPoint es capaz de desempeñar ciclos de control a tasas de hasta 200 veces/seg. Al distribuir LabVIEW justo al lado de sus fuentes de señal, se elimina la PC, lo cual incrementa la comodidad de su sistema, reduce la complejidad del cableado y disminuye el riesgo de captar ruido en cableados largos.

Aunque la nueva plataforma Compact FieldPoint es sólo de la mitad del tamaño del FieldPoint tradicional, contiene la misma funcionalidad y selección de opciones E/S. Compact FieldPoint cuenta con tres controladores, el cFP-2000, cFP-2010 y cFP-2020. Cada módulo contiene memoria Flash interna no volátil para almacenar aplicaciones embebidas y almacenar datos. También contienen memoria DRAM utilizada para ejecutar las aplicaciones de LabVIEW embebidas. El cFP-2010 y cFP-2020, con mayor capacidad de memoria DRAM, son capaces de ejecutar aplicaciones más grandes y complejas. Todos los controladores ejecutan un sistema operativo a tiempo real (RTOS), basado en Windows, con bajos requisitos de procesamiento en un procesador x86 Intel. En la tabla siguiente se expone la diferencia entre módulos controladores:

Controlador Compact FieldPoint LabVIEW Real-Time	DRAM	Memoria Flash a bordo No-volátil	Puertos RS-232	Puertos RS-485	Memoria CompactFlash Removible
cFP-2000	16 MB	16 MB	1	0	--
cFP-2010	32 MB	32 MB	2	0	--
cFP-2020	32 MB	32 MB	3	1	Hasta 512 MB

El sistema operativo tiempo real que se ejecuta en cada módulo proporciona dos ventajas sobre un sistema operativo típico.

1. Este sistema operativo no tiene la complejidad de un sistema operativo estándar. Esto significa que puede operar con menos recursos. El sistema operativo y la máquina de LabVIEW Real Time utiliza 8 MB de DRAM y 6 MB de Flash. El código de LabVIEW se ejecuta rápida y eficazmente en una arquitectura con un procesador x86 de bajo consumo, de manera que la unidad cFP-20xx disipa solo 4.5 W. Debido a que el sistema operativo esta racionalizado, no sufre de las inestabilidades inherentes a un sistema operativo

complejo. Se puede utilizar sistemas cFP-20xx con confianza en aplicaciones de control donde una PC podría ser muy incómodo.

2. El sistema operativo a tiempo real tiene la capacidad de asignar prioridades a cada ciclo de control. Una porción de control de una aplicación puede ejecutarse a alta prioridad mientras una menos importante de comunicación o almacenamiento de datos puede ejecutarse a una prioridad menor. Esto significa que múltiples ciclos PID se ejecutan a tiempo real para asegurar la estabilidad del proceso mientras una interfaz con el usuario (HIM) se ejecuta a una prioridad menor. El sistema operativo a tiempo Real garantiza que los cálculos PID obtengan recursos del procesador cuando se necesiten mientras el HMI utiliza los ciclos libres.

Compact FieldPoint incluye capacidades Ethernet, de manera que se puede distribuir fácilmente inteligencia y control dentro de una fábrica o planta de producción. Sin embargo, más allá de la habilidad de distribuir medida y control, Compact FieldPoint es también una de las plataformas más sencillas de usar para Adquisición de Datos, y ofrece un corto tiempo hasta la primera medida. Tanto la configuración del equipo como la interfaz de programación de la aplicación (API) son simples y fáciles de usar, además el equipo cuenta con acondicionamiento de señal incorporado para conexión rápida y directa de los sensores. Por ejemplo, el módulo cFP-TC-120 filtra una señal de termopar y la mide con un convertidor A/D delta-sigma con 16-bits de resolución. Éste lee el valor de compensación de junta fría del bloque conector, realiza los cálculos de linearización del termopar y retorna un valor de temperatura en unidades de ingeniería.

Se configura el sistema Compact FieldPoint utilizando menús en el Explorador de FieldPoint (FieldPoint Explorer) Con el explorador de FieldPoint, se configura el sistema completo, incluyendo parámetros de red, parámetros de módulos E/S y objetos de canal con nombre. Cuando se configura un sistema Ethernet de FieldPoint, el Explorador de FieldPoint lo lleva de la mano a través de la asignación de parámetros de red, como dirección IP, y busca la subred local para nodos Ethernet de FieldPoint.

Se pueden configurar fácilmente parámetros E/S como rangos de entrada, estados de salida iniciales de arranque, estados del temporizador de seguridad o watchdog al utilizar diálogos de ventana intuitivos.

Adicionalmente, se pueden probar interactivamente los módulos E/S y sus canales (usando la ventana de pruebas), observando valores de los datos de entrada y fijando los valores de salida.

Finalmente el API o interfaz de LabVIEW es una arquitectura simple de lectura / escritura de un solo punto. En LabVIEW se abre una sesión hacia la E/S de Compact FieldPoint utilizando el nombre que se configuró en el explorador de FieldPoint y cablea la referencia de dicha sesión a un VI FP-Read o FP-Write. El VI retornará el valor leído del módulo. No hay necesidad de configurar tamaños de buffer o tasas de adquisición, simplemente lee y escribe puntos únicos.

Con esta arquitectura es muy fácil construir aplicaciones distribuidas. Por ejemplo, se pueden configurar estaciones de registro de datos para monitorizar señales como temperatura, presión y flujo; rastrear la condición de válvulas y almacenar los datos localmente o transmitirlos a través de la red Ethernet, todo esto es muy fácil de hacer utilizando VI´s de LabVIEW incorporados.

Tanto el diseño mecánico, arquitectura de software como las capacidades de control de LabVIEW se combinan para hacer de Compact FieldPoint una plataforma ideal para control embebido en tiempo-real

Mecánica .– Compact FieldPoint, desde la mesa de dibujo, fue diseñado para tener el tamaño, características y especificaciones correctas que funcionen bien en aplicaciones de control embebidas. La primera decisión fue minimizar el tamaño de Compact FieldPoint pero mantener las capacidades de medida de FieldPoint. Al rotar los módulos de E/S y rediseñando el bus E/S, Compact FieldPoint es menos de la mitad del tamaño de FieldPoint pero tiene las mismas capacidades de medida. En muchas aplicaciones de control embebido, el sistema de control es embebido en una máquina que puede experimentar alto impacto y vibración. Para soportar las vibraciones, Compact FieldPoint fue diseñado con un tablero de metal sólido con tornillos de acero para fijar el controlador y los módulos E/S. Por esta razón, Compact FieldPoint puede para soportar impacto de hasta 50 g y 5 g de vibración continua. Finalmente, para aplicaciones donde Compact FieldPoint estará desempeñando control embebido en una máquina, la línea de productos fue diseñada para utilizar conectores estándares de 37 pines.

Gracias a estos conectores comunes y económicos, el usuario puede construir cables personalizados para instalaciones, reduciendo costos de cableado y eliminando errores de cableado.

Arquitectura de Software .- La teoría estándar de control está basada en toma de decisiones punto a punto. Debido a que Compact FieldPoint utiliza una arquitectura simple de lectura / escritura punto a punto tiene una disposición natural para control embebido. Utilizando el procesamiento punto a punto disponible en LabVIEW Real Time se puede realizar procesamiento y control avanzado de señales manteniendo un modelo de adquisición de un solo punto sencillo de entender. En esta aplicación el programa lee un punto a la vez y pasa cada punto dentro de funciones especiales diseñadas para realizar funciones continuas como promedios, cálculos alto / bajo, desviación estándar y cálculos de frecuencia. Con estos programas se pueden configurar fácilmente ciclos de control a tiempo real sin la complicación adicional de recopilar búferes de datos.

Capacidades de Control de LabVIEW .- Cuando su aplicación requiere potente funcionalidad y desarrollo rápido, la clave es un software flexible que se integre perfectamente con el equipo. LabVIEW es un ambiente de desarrollo gráfico estándar en la industria que proporciona todas las herramientas necesarias para crear las aplicaciones avanzadas con características completas de medida y control. LabVIEW facilita la construcción simple de aplicaciones complejas utilizando una paleta extensiva de funciones y herramientas, desde ciclos simples control PID analógico para procesos hasta sistemas de control híbridos con muchos canales que combinan componentes analógicos y discretos. El Juego de Herramientas de Control PID para LabVIEW contiene bloques para PID básico y avanzado, prealimentación (feedforward) y control difuso; se utiliza también para pruebas, modelación y simulación lineal y no lineal.

Cuando se desarrolla un sistema de control, se pueden aprovechar las capacidades de LabVIEW para implementar control básico o incorporar fácilmente técnicas de control avanzado como desacoplamiento de perturbaciones, planificación de ganancias y control de lógica difusa. Para usuarios sin experiencia, hay herramientas potentes como la auto-sintonización

PID que facilitan la iniciación.

Registro y almacenamiento de Datos.- Todos los controladores Compact FieldPoint cuentan con memoria Flash no volátil ya incorporada para almacenamiento de datos. El cFP-2020, además de la memoria Flash a bordo, también cuenta con una ranura para memoria CompactFlash extraíble, de manera que se pueden almacenar hasta 512 MB en datos. Dichos datos pueden ser registrados en cualquier formato compatible con DOS. Una vez almacenados, los datos pueden ser transferidos fácilmente a un PC utilizando el servidor FTP embebido en los controladores Compact FieldPoint.

Más allá del típico registrador de datos, el cual simplemente almacena datos en un disco a bordo, con LabVIEW Real Time se puede crear un registrador de datos inteligente embebido en los módulos Compact FieldPoint. Un registrador de datos inteligente puede hacer cálculos adicionales y tomar decisiones para reducir los datos innecesarios. En sistemas más avanzados, los registradores de datos embebidos incorporarán la funcionalidad de un ordenador y tendrán la capacidad de realizar procesamiento y control a tiempo real a bordo. En un sistema Compact FieldPoint, National Instruments combina la reducción del volumen de datos, algoritmos de control, HMI (Interfaz Hombre-Máquina), y en algunos sistemas, la, capacidad de comunicarse con otros nodos de la red.

Tecnologías habilitadas para Web con LabVIEW .- Una de las características más importantes de Compact FieldPoint, es la capacidad de embeber el panel frontal de LabVIEW y compartirlo a través de la red. Ahora, sin ninguna programación, se puede crear un panel de usuario interactivo, habilitado en la Web y completamente gráfico. Los controladores a Tiempo Real de Compact FieldPoint incluyen un buscador de Web embebido que maneja hasta 20 conexiones de paneles remotos simultáneas. Con los paneles remotos, se puede utilizar un buscador de Internet para conectarse a la interfaz de usuario del panel frontal de su aplicación de LabVIEW. Varios buscadores de Web clientes pueden ver simultáneamente el panel frontal de LabVIEW, y un buscador de Web a la vez puede tanto ver como controlar la aplicación.

Además de la capacidad de publicar conexiones de paneles remotos interactivos, también puede publicar páginas HTML estándar desde el servidor Web incorporado en el cFP-20xx. Por lo tanto, puede compartir datos almacenados, crear y compartir reportes Web y publicar páginas web en general.

Para compartir datos, los sistemas Compact FieldPoint también vienen configurados con un servidor FTP incorporado, de manera que se pueden compartir datos almacenados y actualizar código de control y archivos HTML desde locaciones remotas. En múltiples procesos se necesita comunicarse y controlar dispositivos externos desde el controlador principal del proceso. Estos pueden ser dispositivos serie estándares RS-232 o RS-485 o pueden ser dispositivos Ethernet. Todos los controladores Compact FieldPoint vienen con uno o más puertos serie RS-232 y un puerto Ethernet 10/100BaseTX. Para las aplicaciones que requieren más control y almacenamiento de datos, las cuales implican control de dispositivos externos como lectores de códigos de barras, unidades GPS, válvulas de control y unidades teclado / pantalla LCD, el controlador cFP-2020 ofrece tres puertos RS-232 y un puerto serie RS-485 con aislamiento. El RS-485 provee una red multipuntos de bajo coste y de larga distancia (1.2 Km) para conexión de dispositivos externos. Estos puertos son accesibles a través de código de LabVIEW, de manera que usted puede leer, escribir y controlar dispositivos serie externos. Hay una librería completa de drivers o controladores serie existentes (como Modbus) que facilitan la conexión con dispositivos industriales de terceros. Si un software controlador no existe para un dispositivo, las funciones de desglose de texto de LabVIEW facilitan el diseño de un driver o controlador personalizado de comunicación para dispositivos serie o Ethernet.

Si se necesita ejecutar LabVIEW en un ambiente hostil, FieldPoint cuenta con un amplio rango de temperaturas de operación (- 25 hasta 60°C) significa que Compact FieldPoint puede ejecutar LabVIEW en aplicaciones que pueden causar que muchas PCs industriales fallen. Debido al amplio rango de temperaturas, Compact FieldPoint esta siendo investigado para su uso en los oleoductos de Alaska para monitorización de corrosión y registro histórico. Para aplicaciones que requieren alta tolerancia a las vibraciones, los módulos de Compact FieldPoint utilizan un montaje seguro por tornillos entre los módulos E/S, módulo controlador y el tablero principal. También cuentan con conectores de cable con tornillos para los conectores de 37 pines hacia el tablero. Estos mecanismos de aseguramiento positivos eliminan fallas de desconexión causados por impacto mecánico y vibración. Los módulos están calificados para impacto de 50 g y vibración de 5 g para 10-500 hz. Compact FieldPoint también está diseñado para funcionar en áreas con ruido electromagnético. Cada tablero metálico cuenta con tornillos de tierra para

proveer una ruta para descarga de corriente electrostática. Los sistemas están también diseñados y calificados para total cumplimiento sin recinto metálico blindado. Finalmente, los módulos Compact FieldPoint son cambiables en plena marcha para instalación y mantenimiento rápido; también cuentan con encendido y estados de temporizador de seguridad (watchdog) programables para operación predictiva.

Aplicaciones de LabVIEW en Todo Lugar.- FieldPoint ha resuelto un amplio rango de aplicaciones desde monitorizar los manuscritos de Leonardo da Vinci, máquinas que automatizan el enlatado de hongos y conservas, controlar torres de perforación petroleras hasta pruebas móviles de máquinas de camiones. Con todos estos fundamentos, Compact FieldPoint combina acondicionamiento y adquisición de señales y código de LabVIEW embebido en un módulo más pequeño y más robusto, facilitando la construcción de sistemas de control y medida distribuibles.

6.3 Planteamiento de las utilerías necesarias

El desarrollo de las utilerías que se necesitaran para la comunicación del FieldPoint con el MTP se realizará en LABVIEW, en donde como ye se mencionó se cuenta con un panel frontal que aparece en gris y un panel de control que es en el que se realiza la conexión de los elementos.

En esta ocasión se utilizó LabView versión 6.1 y se definió que lo que realizaría el programa sería la detección en el cambio de potencial en ocho potenciometros previamente colocados en la caja de demostración, cada potenciometro estará conectado a un canal del cFP 2020, ya que como se vio en las especificaciones de este, cuenta con ocho canales, de esta manera al recibir los datos de cambio de potencial provenientes de los potenciometros el FieldPoint los procesará y sacará un promedio, estos datos serán enviados al puerto serie que esta conectado al PIC, que se programará de tal forma que recibirá los datos y los enviará a la pantalla de cristal liquido tomando en cuenta la sección de la pantalla a la que se hará dicho envío, esto es definido previamente en la utilería de LabView correspondiente, también se definirán caracteres que quedarán fijos en la parte superior de la pantalla, estos formarán las palabras “INSTITUTO DE INGENIERIA INSTRUMENTACIÓN” en un el

renglón superior y en el siguiente, no al inicio sino en la parte en la que debajo de ellos aparecerán datos numéricos, se formaran las palabras “V. INSTANTÁNEO V. PROMEDIO”

En la parte baja de la pantalla se desplegarán en pares el número de canales y sus correspondientes valores numéricos que variarán al mover los potenciometros, el cambio de canales se hará presionando una tecla del teclado previamente elaborado, para lo cual se desarrollará también una sección de lectura de teclado con una VISA READ que registra la tecla presionada y a la cual se le da una función especifica, además del cambio de canal se asignará otra tecla par reinicializar pantalla y otras dos para aparecer y desaparecer el cursor parpadeante o blinking.

Cabe mencionar que para el correcto funcionamiento de este programa se tienen que usar retrasos que serán mencionados posteriormente, además de la configuración del puerto que debe coincidir con la del PIC

Vale la pena destacar que la aplicación que se la dará al Fieldpoint es meramente demostrativa y por lo tanto muy sencilla, dejando en claro que la cantidad de aplicaciones que se le pueden dar son muchísimas más y más complejas, de tal suerte que basta con cambiar un poco la utilerías en labview y asegurarse que el MTP pueda procesar y desplegar los datos recibidos

Dentro la rutina principal y sus diferentes casos o situaciones se incluyen subrutinas que aparecerán dentro de cuadros con sus iniciales, estas son EPA (Escribe pantalla), ECA (Escribe carácter) y ECO (Escribe comando), en las cuales se realiza el envío a sección pantalla y si es comando o carácter. Esto se hace con fines comodidad y modularidad, pues además de ahorrar espacio y hacer menos complejo el programa principal se tienen rutinas estándares de envío a pantalla que pueden usarse en otras aplicaciones.

6.4 Desarrollo y simulación de rutinas elaboradas

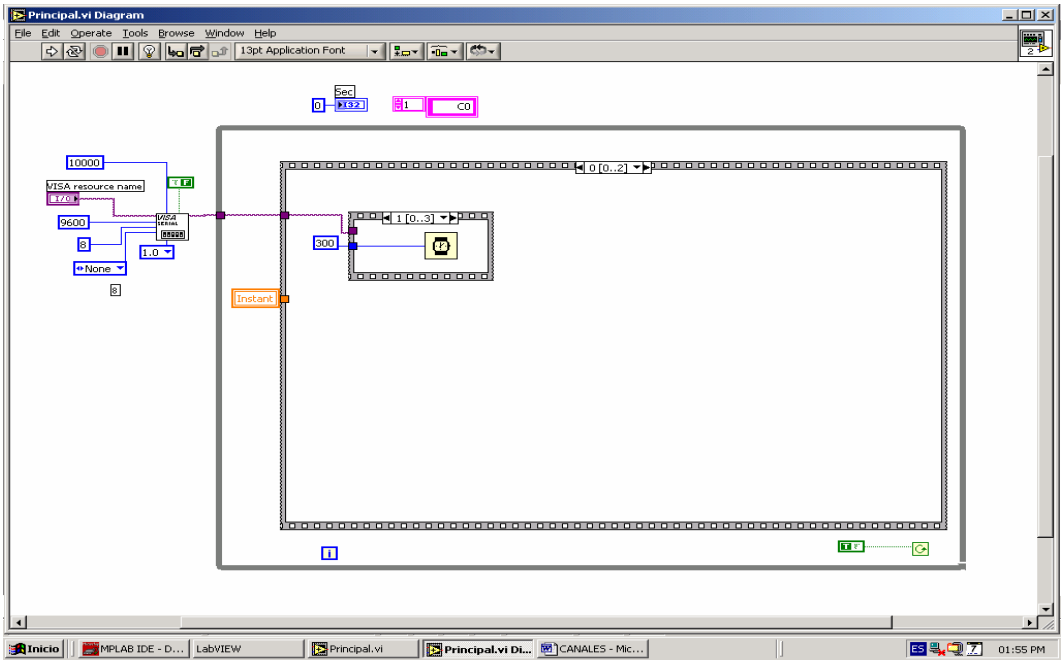
El desarrollo y simulación del programa se realiza dentro del panel de control de LabView, seleccionando dentro de la paleta de controles la aplicación adecuada, colocándola en el diagrama y uniéndola a otro elemento que le corresponda usando el cursor en modalidad de carrete, esto en cuanto a la construcción y para la simulación basta con correr el programa en forma animada (usando el foco que aparece en la barra de

tareas) y de esta forma se indica si el programa funciona de manera eficiente y de no ser así marca el lugar en donde esta el error.

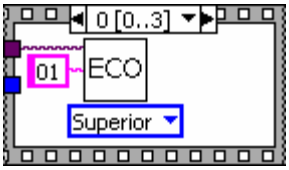
El primer paso después de abrir la ventana del panel de control es abrir una VISA y declarar sus parámetros, como son el puerto de comunicación, el número de bits, el baudaje, los bits de paridad y los bits de paro.

El primer caso que se considera es el reset de pantalla cada vez que se inicie el programa, para lo cual se realiza una rutina en la que se manda un retraso, se limpia la pantalla superior con el comando “01”, se da otro retraso y se manda el comando para limpiar la pantalla baja.

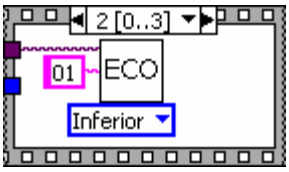
Esta ventana ilustra la VISA de comunicación y el “frame” de la rutina de reset de pantalla, en este caso se ilustra el primer retraso



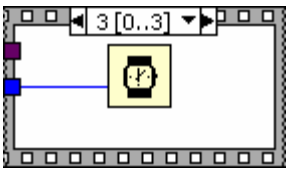
En esta ventana se muestra la parte en la que se manda el comando de limpiar pantalla superior, esto se manda a la rutina ECO que es el envío de caracteres, dicha ruina se verá a fondo después.



Al igual que la ventana anterior, está ilustra el envío de limpia pantalla a la parte inferior por medio de la rutina ECO

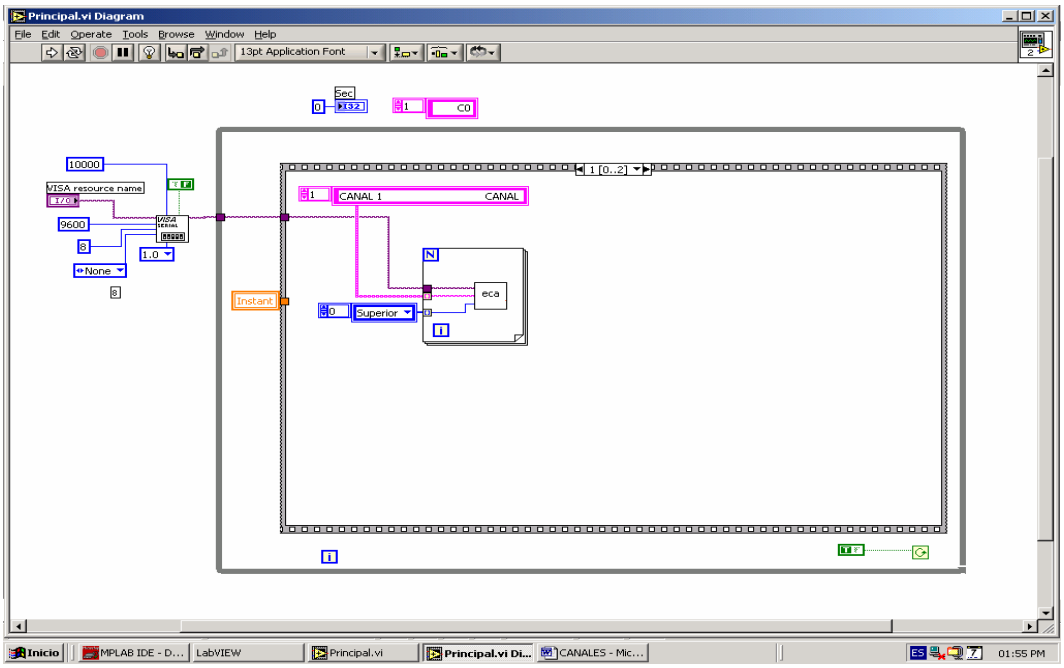


Está es la ventana que ilustra el retraso que se le da a la rutina

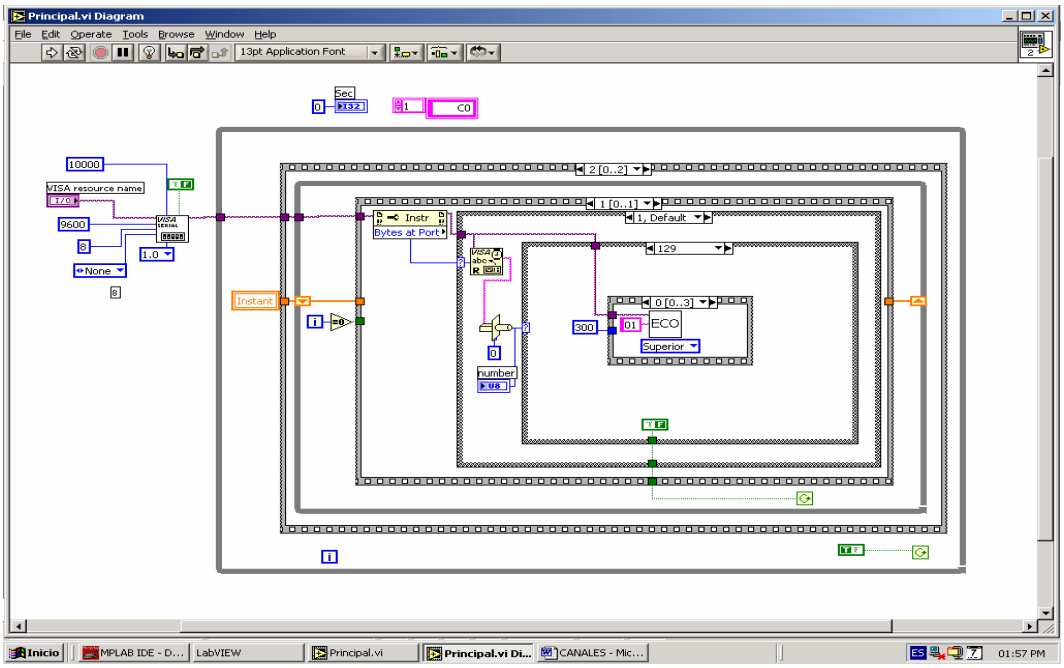


Cabe destacar que las ventanas anteriores son parte de la sección de reset a pantalla.

La siguiente ventana muestra los elementos que se interconectaron para mandar el mensaje fijo “INSTITUTO DE INGENIERIA INSTRUMENTACIÓN V. INSTANTÁNEO V. PROMEDIO” a la pantalla alta, por medio de un “string “ en el cual se escribe lo que se va a enviar y se manda a la subrutina “eca” que es la encargada de enviar caracteres, dicha subrutina se analizara a fondo posteriormente.

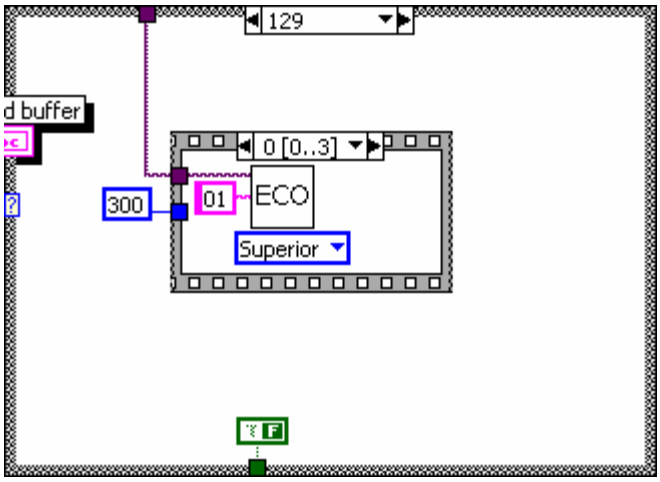


Está ventana muestra el tratamiento que se le dará al comando enviado a una VISA READ una vez que se oprimió una tecla

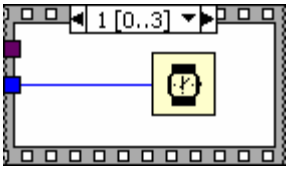


El teclado posee 16 teclas, para efectos de esta aplicación solo se usarán las 4 teclas superiores que de acuerdo al arreglo de comandos corresponden a los números en hexadecimal 129, 65, 33 y 17 que resetean pantalla, cambian canales, desaparece y aparece cursor respectivamente.

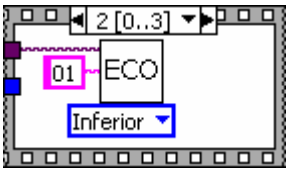
El primero que aparece es el reset de pantalla, cabe mencionar que en la ventana del “case” respectivo se introducirá su correspondiente código en decimal, lo primero es introducir el código que resetea la pantalla, el cual de cuerdo a la configuración de esta es 01 y la parte a la que se envía, como se muestra a continuación .



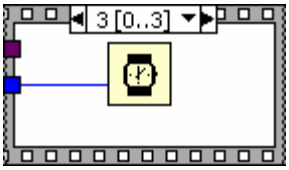
En la siguiente ventana de esta sección se introduce un retraso de 300ms mientras se espera para la siguiente instrucción.



Ahora se manda la orden de limpiar pantalla inferior como se indica a continuación.

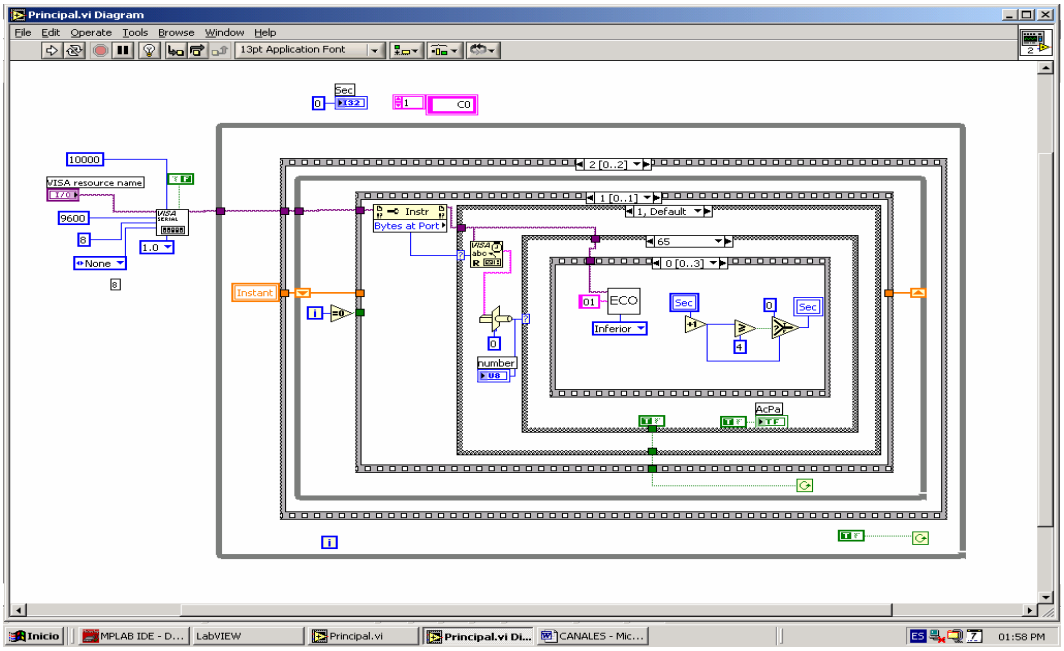


Por último se da oro retraso de tiempo para dejar lista la pantalla.

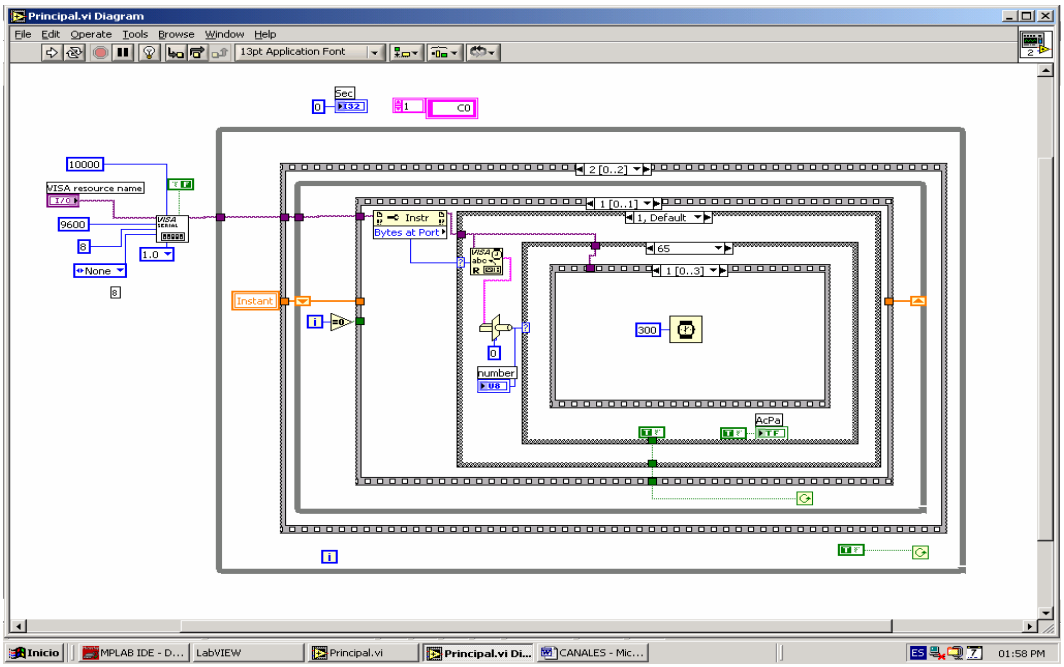


A continuación se analiza la segunda tecla que realiza el cambio de canal que consta de 4 pasos es conveniente mencionar que el tercer paso se introducen 4 pasos más.

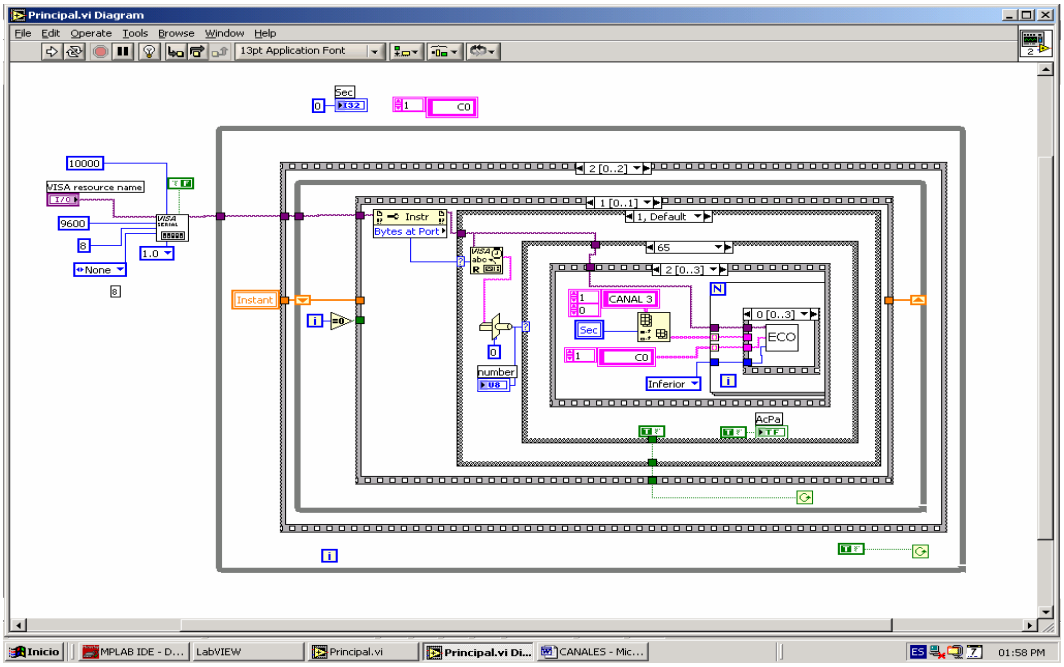
El que aparece a continuación es el primero en el cual se limpia la pantalla inferior y se preparan los caracteres a enviar.



Ahora se introduce un retraso de 300ms antes de enviar los caracteres.



Esta sección como ya se comento tiene cuatro “frames” adicionales en el primero se define en donde se escribirá, que en este caso es en el inicio de la parte alta de la pantalla baja.

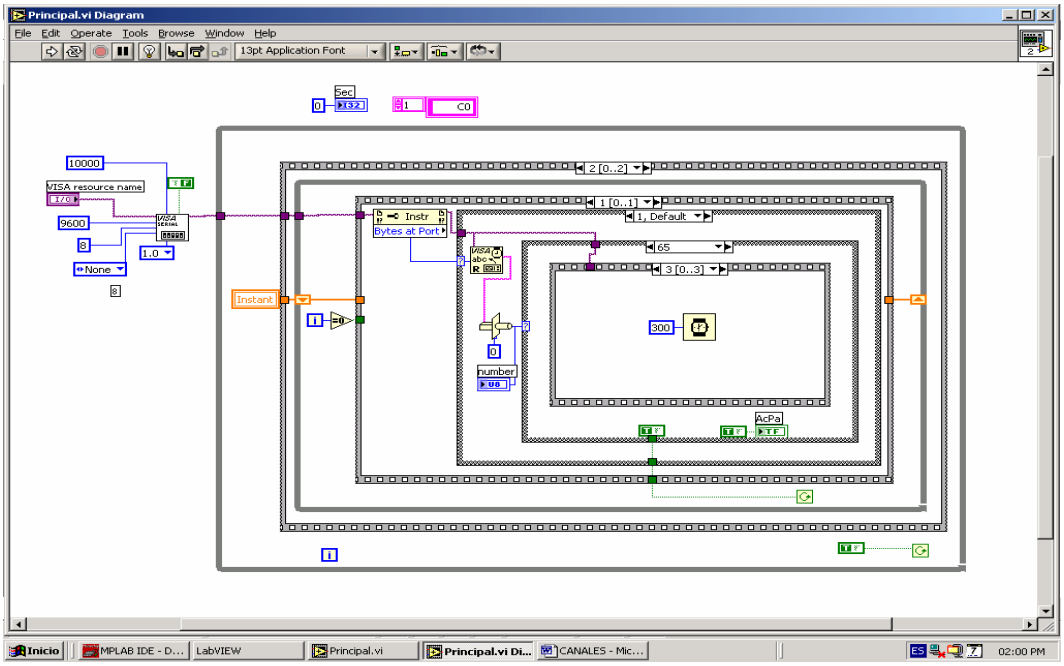


Ahora se envía el mensaje CANAL N y sus respectivos valores.

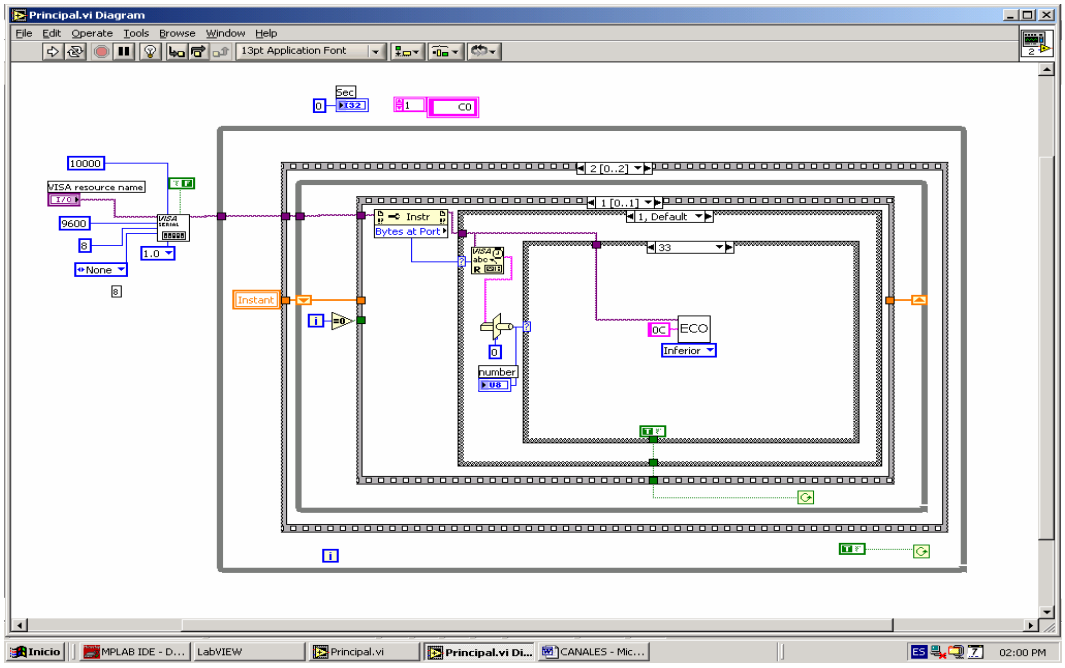
Posteriormente se define la sección donde enviará el siguiente mensaje, que en este caso es en el inicio de la parte baja de la pantalla baja.

Después se envía el mensaje CANAL N+1 y sus valores.

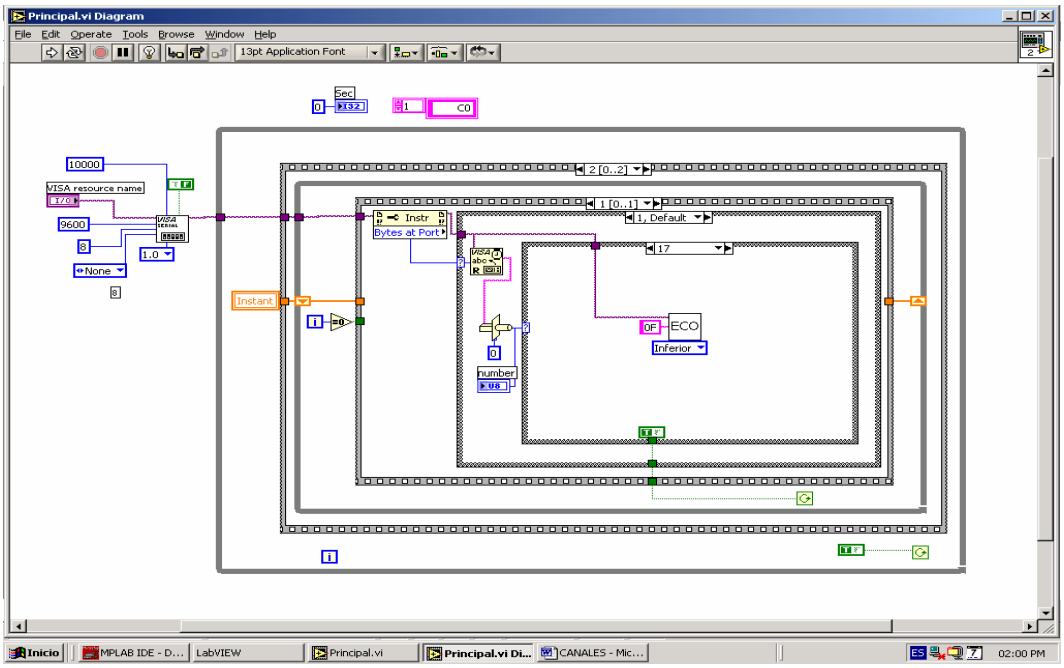
Lo último del tratamiento de esta tecla es la introducción de un retraso de 300 ms para la estabilización.



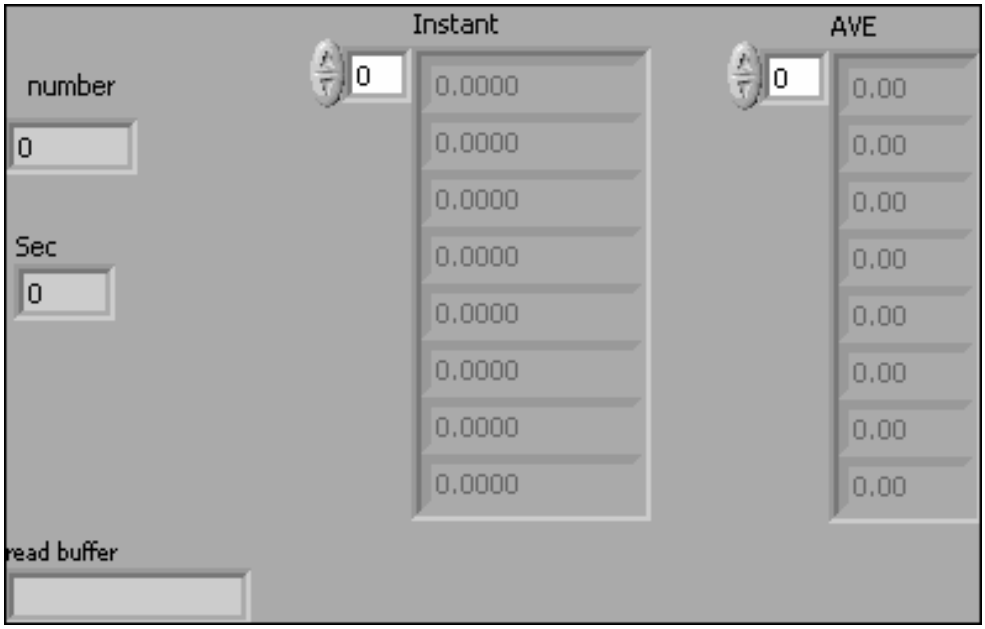
Ahora se analiza lo que se realiza cuando se presiona la tercera tecla, que es simplemente enviar 0C a la pantalla inferior que corresponde a desaparecer cursor parpadeante.



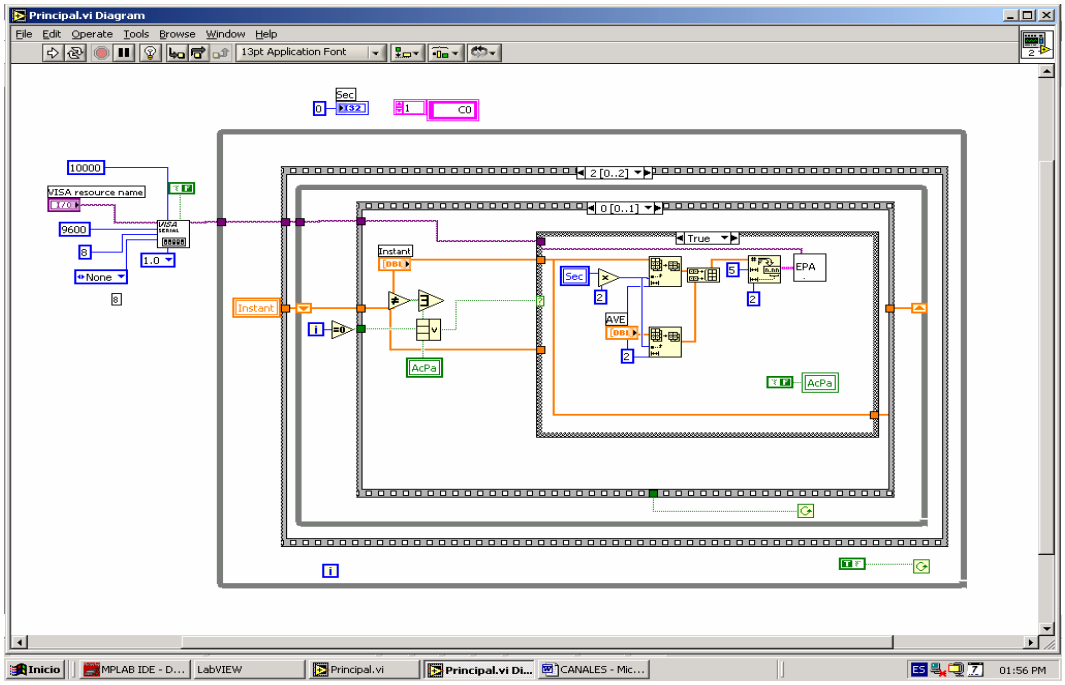
En el caso de la cuarta tecla se realiza un proceso muy similar al anterior, con la diferencia que el comando enviado es 0F que corresponde a aparecer el cursor parpadeante.



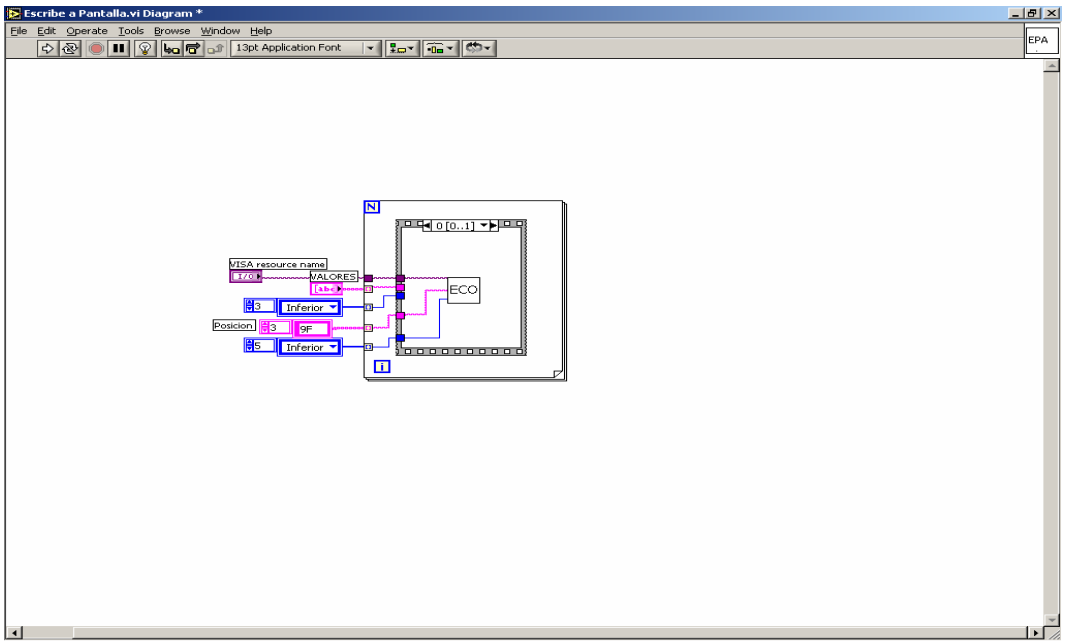
El panel frontal que aparece a continuación es el correspondiente al programa principal y en el se introducen los valores de los diferentes canales que serán los que aparecen en la pantalla, cabe destacar que esto se hace solo para la simulación, pues para efectos prácticos el cambio de canales se dará por la manipulación de los potenciometros.



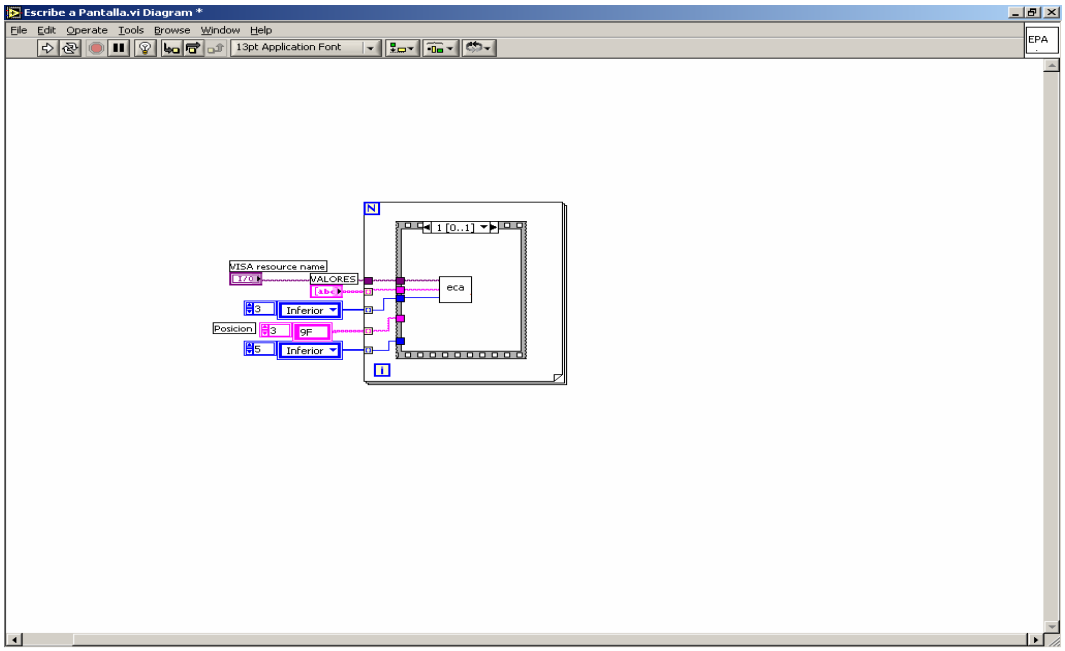
Esta es la ventana que ilustra el envío de los datos numéricos a la pantalla en la que se define el número de dígitos y la precisión del número, como se observa también se incluye la subrutina EPA que se encargará del trato de dichos números.



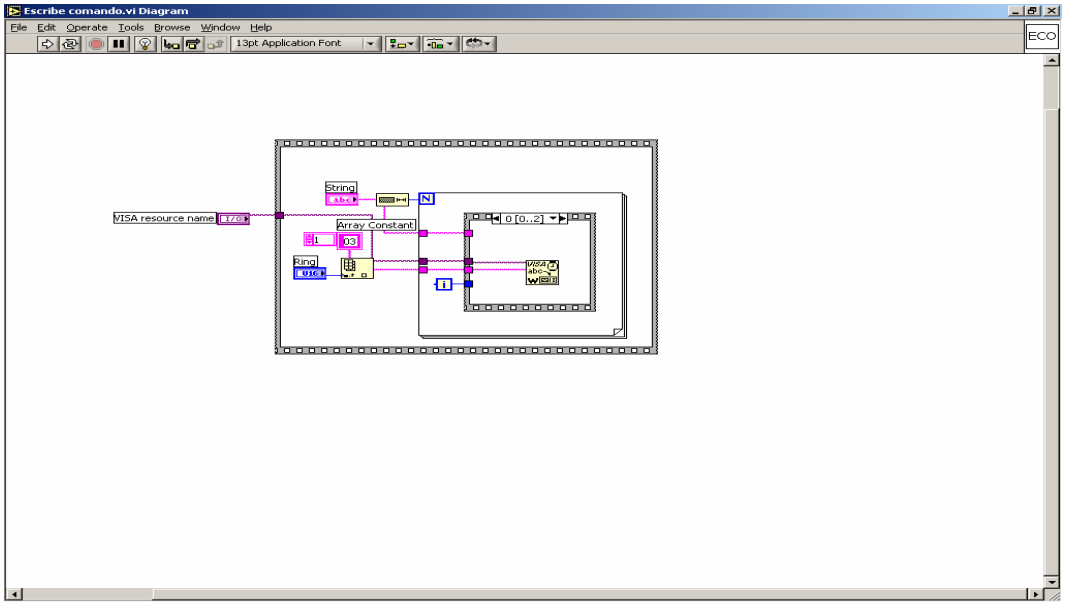
La subrutina EPA (escribe a pantalla) consta de dos ventanas, la primera se encarga de definir el lugar en el que se escribirán los datos numéricos en la pantalla inferior, haciendo uso para ello de otra subrutina ECO (envía comando), que se analizará posteriormente.



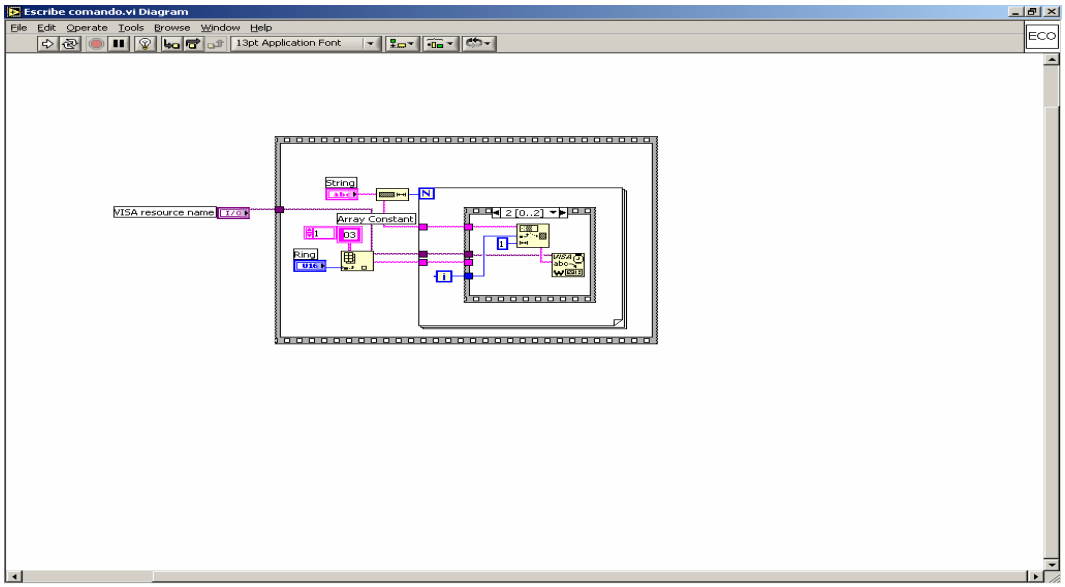
La otra parte de la subrutina EPA es la que aparece a continuación en la cual se simplemente se envían los números a la sección de pantalla antes definida, usando otra subrutina eca (envía caracter).



Esta ventana corresponde a la subrutina envía comando, que consta de dos partes, en la primera se lee el dato.



En esta ventana que corresponde a la segunda parte de ECO se mandan los datos a pantalla, es importante mencionar que en esta sección se define que lo que se enviara es un comando a la pantalla baja mediante el numero 03 pues así lo leerá el PIC.

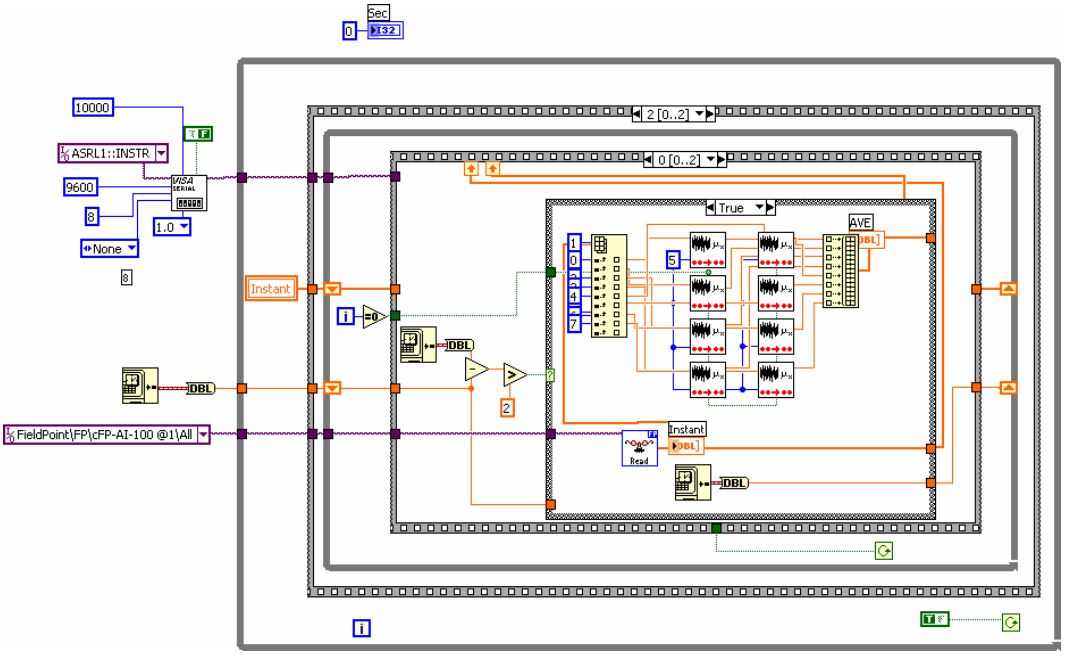


En cuanto a la subrutina de “eca” se mencionara que es idéntica a la anterior ECO, con la única diferencia de que lo que se define es un número 05 que es envío de carácter a pantalla baja.

6.5 Implementación de utilerías al módulo

La implementación se realiza directamente de la PC usando un cable Ethernet , habiendo definido previamente el tipo de FP que se usara que en esta ocasión es el cFP 2020, esto se realiza en el programa Mesurement & Automation Explorer de National instruments.

Es importante destacar que se deben realizar cambios al programa principal que se ilustran en la ventana que aparece a continuación.



Estos cambios son fundamentalmente el cambio de la VISA que se definió al inicio del capitulo anterior por el icono que representa el módulo FP a usar, además de que en el “case” que aparece al centro de la ventana se sacan los promedios que se envían a la pantalla

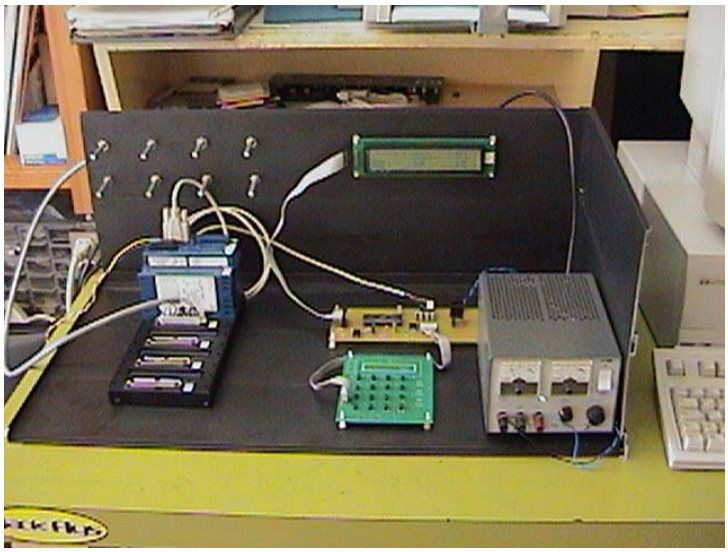
CAPÍTULO VII VERIFICACIÓN

7.1 Verificación de funcionalidad

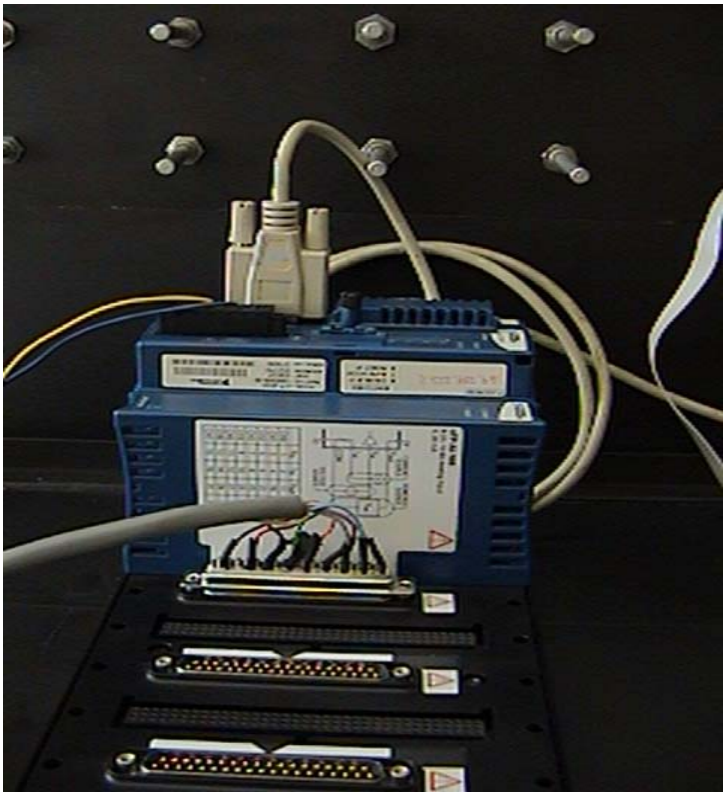
En este breve capítulo se hace una descripción de las pruebas que se hicieron al módulo una vez terminado.

La siguiente imagen muestra el aspecto final del módulo con el PIC, el teclado, la pantalla, el FieldPoint, la fuente de alimentación, los potenciometros

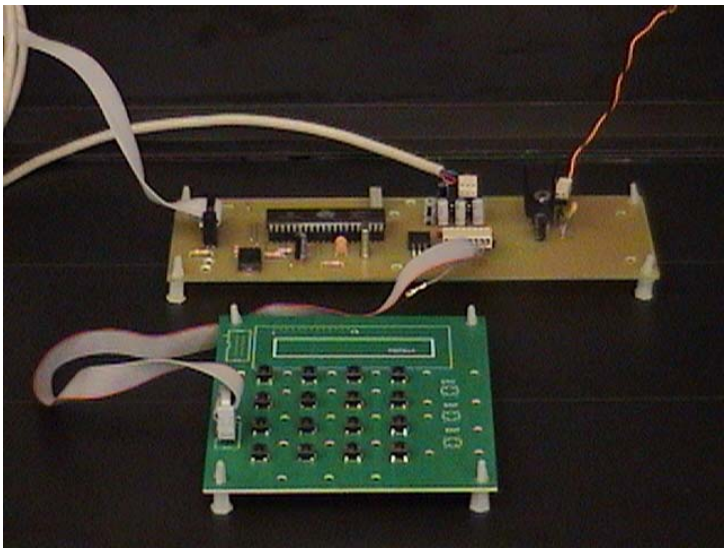
Y los cables necesarios. Los componentes antes mencionados están montados sobre una base de aluminio negro que fue perforada par ala colocación de los componentes mismos que están sujetos con postes de plástico que fueron previamente machueleados a la base de aluminio. En el caso del FP se fijo con tornillos, mientras el cable que conecta a los potenciometros y el DB37 fue un cable blindado de 10 hilos.



Esta imagen muestra el cFP2020 con sus cables de alimentación, el cable DB9 que se conecta al circuito del PIC y el cable DB37 que se conecta a los potenciometros que también se lustran.



En la siguiente imagen se ilustra el teclado y el circuito que controla al PIC, como ya se menciono antes en esta aplicación se utilizan los cuatro botones dela parte superior, mismos que realizan las funciones de reset, cambio de canales, desaparecer y aparecer cursor parpadeante, como se menciono en el capítulo del FieldPoint lo que se realiza es el despliegue en pantalla de los voltajes presentes en los ocho potenciometros colocados en la pared de la caja, dichos datos son recibidos en cada uno de los ocho canales del FieldPoint y promediados también por el después de esto con ayuda del las librerías instaladas en el se envía al puerto serie del PIC, mismo que despliega en pantalla los valores de voltaje de cada uno de los canales.



Aquí se muestra el desplegado en pantalla antes mencionado, por motivos de espacio solo se muestran los dos primeros canales, pero como ya se menciono con el segundo botón del teclado se hace el cambio para ver otros canales o sea el 3 y 4, 5 y 6, 7 y 8, cabe destacar que basta con mover los potenciometros para cambiar el valor instantáneo de voltaje y que se realice el promedio con el valor anterior.

Como seria de esperarse se tuvieron que realizar varias pruebas y modificaciones de software antes de poder dar por concluidas y satisfactorias las pruebas de funcionalidad, como se menciono ya en un capítulo anterior esta aplicación es meramente demostrativa y es muy sencilla, pero dar una aplicación más compleja requeriría de muy pocos cambios, pues el diseño y el software permite mucha flexibilidad, de esta manera se concluye la prueba de funcionalidad con la imagen del módulo (MTP) realizando la función deseada.

7.2 Verificación de optimización económica

La comparación del módulo con otros similares no se puede realizar puesto que no existe en el mercado alguno igual ya que este posee varias ventajas sobre algunos similares que solo poseen pantalla y esta no es de las características de la que tiene el módulo realizado.

La caja en la que se colocaron los diferentes componentes es meramente demostrativa, pues en una aplicación industrial se colocaría dentro de una caja donde solo se observaría la pantalla y las teclas que se utilizarán, dicha caja sería hermética y de menores dimensiones, pero en esta ocasión la caja demostrativa de aluminio tuvo un costo bajo pues se compro la vigueta de aluminio que costo alrededor de $150 pesos y su realización se realizo de manera sencilla y sin inversión de dinero.

La pantalla costo alrededor de $900 pesos, el teclado que se realizo tuvo un costo bajo de $100 pesos aproximadamente, en cuanto al PIC su precio oscila entre los $100 pesos y su circuito de control con dispositivos adiciónales costo $150 pesos aproximadamente y el gasto de cables y potenciometros no sobrepaso los $100 pesos.

Por lo mencionado anteriormente se define el MTP con un costo aproximado de $1500 pesos, precio que no es elevado considerando lo innovador y útil del módulo.

CONCLUSIONES

La realización de esta tesis tuvo como objetivo principal ofrecer una opción de despliegue en sitio para los módulos FieldPoint pues el hecho de contar con un teclado y una pantalla para observar los datos adquiridos es una innovación y en ello radica el merito principal del trabajo pues además la pantalla no es de características comunes ya que es un pantalla de 4X40 caracteres, de esta forma la cantidad de datos que se pueden desplegar es mayor. Como se observo en el capítulo de verificación el funcionamiento del módulo realizado es optimo por lo que en general se puede concluir que el módulo es útil, no solo para esta aplicación, pues como se ha mencionado a lo largo del trabajo la implementación de otra aplicación solo requeriría de algunos sencillos cambios de tal suerte que la eficaz conclusión de este módulo deja listo el campo para sus aplicaciones en el uso de diversos proyectos en los que se vea inmerso el uso del FieldPoint de National Instruments.

BIBLIOGRAFÍA

LIBROS

Alatorre González Antonio
"Libro de Programación en LabVIEW"
Versión LabVIEW 7.0, 2003.

Angulo Usategui J. Ma. , Martín Cuenca E. y Angulo Martínez J.
"Microcontroladores PIC, la solución en un chip"
Ed. Paraninfo, 1997.

Angulo Usategui J. Ma. , Martín Cuenca E. y Angulo Martínez J.
"Aplicaciones de los microcontroladores PIC de Microchip"
Ed. McGraw Hill, 1998.

Beltrán de Heredia Jhon
"Lenguaje ensamblador de los 80X86"
Ed. Anaya Multimedia, 200 p. 1ª edición, Junio 1994.

Charte Ojeda Francisco
"Ensamblador"
Ed. Anaya Multimedia, 688 p + 1 CD-ROM 1ª edición, Enero 2003.

González Vásquez José Adolfo
"Introducción a los microcontroladores"
Ed. McGraw Hill, 1999.

Joyanes Aguilar Luis
“Problemas de metodología de la programación”
Ed. McGraw Hill, 1998

Lázaro Antonio Manuel
“Adquisición de datos (DAQ)”
Ed. Paraninfo, 2000.

López Navarro, J.M. y otros
“Sistemas de adquisición de datos basados en ordenador personal”
Dpto. Publicaciones EUITT, 1996.

Martín Fernández Alberto
“Transductores y acondicionadores de señal”
Dpto. Publicaciones EUITT, 1998.

Tavernier J.
“Microcontroladores PIC”
Ed. Paraninfo, 1998.

Tekcien Ltda.
“Cursos sobre Microcontroladores PIC, Niveles Básico y Avanzado”.
1999

Woods Tony
“Programación en lenguaje ensamblador”
Ed. McGraw Hill, 176p, 1985

MANUALES

Embedded Control Handbook, Microchip PIC 16/17, PDF

LabVIEW User Manual, PDF

Microchip, AN587 Application Note, PDF

Microchip, AN774 Application Note, PDF

Microchip, MPLAB IDE User's Guide, PDF

Microchip, PIC16CX, PDF

Microchip, PIC Microcontrollers Data Book, Microchip Technology Inc. PDF

Microchip, PICSTART PLUS User's Guide, PDF

Microchip, The embedded control solutions company, 1997

Microcontroller data Book, Microchip MPASM assembler

National Instruments, cFP_latam, PDF

National Instruments, Measurement and Automation Catalog. 2000

PAGINAS EN INTERNET

www.digikey.com

www.eio.com/lcdprodt.htm

www.jameco.com

www.microchip.com

www.ni.com

www.ni.com/fieldpoint/cfp-20xx/quickstarguide

www.ni.com/labview/
www.ni.com/labview/real-time/moduleusermanual

www.seetron.com/bpp440_1.htm

MANUAL DE USUARIO

Descripción del Equipo

El equipo está formado por una microcomputadora basada en un microcontrolador PIC16C65 el cual consta de 4 puertos paralelos un puerto serie, al cual se le han integrado los siguientes dispositivos:

-Teclado de 16 botones

-Pantalla de cristal líquido de 4X40 iluminada

Objetivo del Equipo

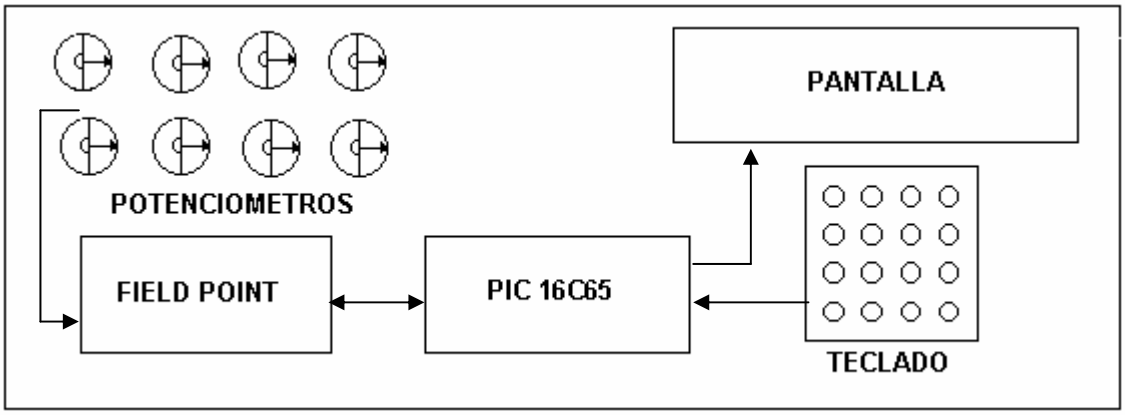
El objetivo es lograr la comunicación entre la microcomputadora antes mencionada y un módulo Field Point de National Instruments.

Para la operación del se equipo cuenta con un juego de 16 botones así como una pantalla de cristal líquido (LCD) de 4X40 caracteres con iluminación en donde son desplegados los valores adquiridos de manera instantánea.

Para este caso en particular se cuenta con ocho potenciometros que dan variación a un voltaje previamente alimentado y será desplegado en la pantalla, a este volatje desplegado se le calcula el promedio que tambien es desplegado.

Operación del Equipo

En esta aplicación se utilizan los cuatro botones dela parte superior, que realizan las funciones de reset, cambio de canales, desaparecer y aparecer cursor parpadeante, lo que se realiza es el despliegue en pantalla de los voltajes presentes en los ocho potenciometros colocados en la pared de la caja, dichos datos son recibidos en cada uno de los ocho canales del FieldPoint y promediados también por el después de esto con ayuda del las librerías instaladas en el se envía al puerto serie del PIC, mismo que despliega en pantalla los valores de voltaje de cada canal.



Al encender el equipo se presentará la siguiente pantalla:

INSTITUTO DE INGENIERIA INSTRUMENTACION		
	V. INSTANTANEO	V. PROMEDIO
CANAL 1	3.50	3.00
CANAL 2	7.87	7.50

Al presionar el botón se resetea el equipo, si se presiona el segundo botón cambia en la pantalla el valor de los canles actuales por los siguientes dos, es decir 1 y 2 por 3 y 4 y asi sucesivamente

Para cambiar los valores que aparecen desplegados se manipulan los potenciometros, el voltaje instantáneo ingresa de manera directa al Field Point y es transmitido al PIC que lo envía a la pantalla, en tanto que el voltaje promedio es obtenido por la operación del Fiel Point y se transmite de la misma forma.

Figura 1 Circuito impreso

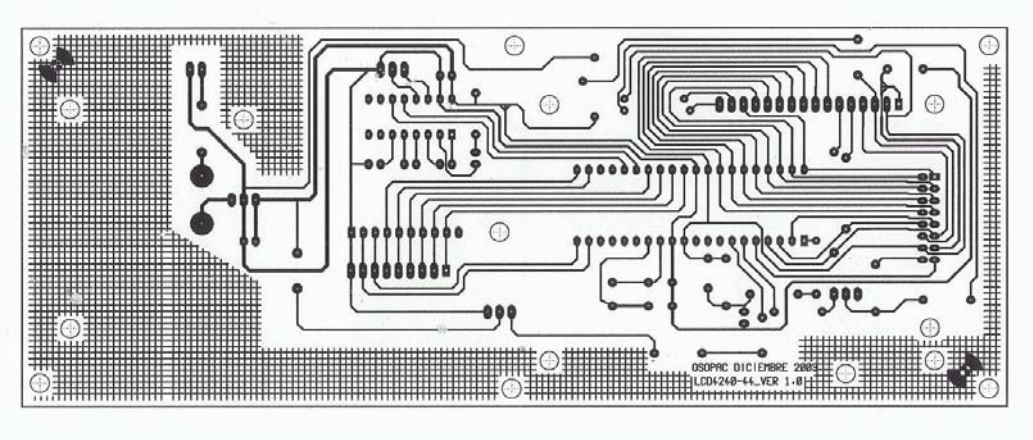


Figura 2 Fuente

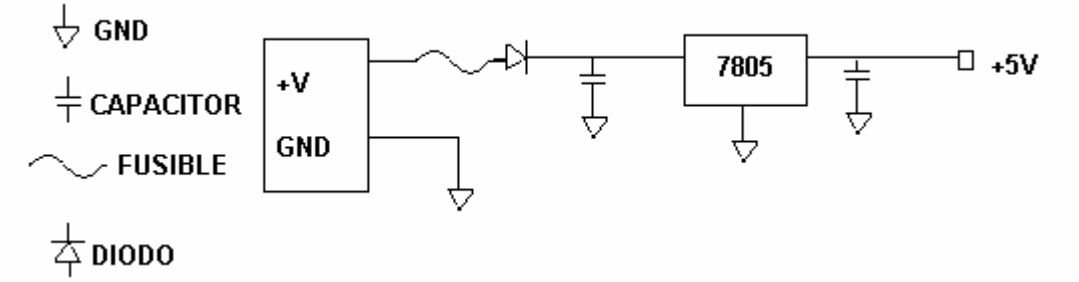


Figura 3 Diagrama del Microcontrolador