



**ESCUELA SUPERIOR POLITÉCNICA DEL  
EJÉRCITO  
ESPE – LATACUNGA**

**CARRERA DE INGENIERÍA ELECTRÓNICA**

**PROYECTO DE GRADO PARA LA OBTENCIÓN  
DEL TÍTULO EN INGENIERÍA ELECTRÓNICA  
EN INSTRUMENTACIÓN**

**“Diseño y Construcción de un Control: Local y Remoto  
de un Variador de Frecuencia para el Manejo de un  
Motor Ac Utilizando el MICROCONTROLADOR PIC  
16F877 y la Plataforma de Desarrollo Labview.”.**

Recalde Tapia Cristian Paúl  
Zurita Armijos Patricio Sebastián

Latacunga-Ecuador

2005

## **CERTIFICACIÓN**

Certificamos que el presente documento fue elaborado por los señores:

Cristian Paúl Recalde Tapia  
Patricio Sebastián Zurita Armijos

Bajo nuestra dirección, como un requisito para la obtención del título en Ingeniero Electrónica en Instrumentación

---

Ing. Franklin Silva

DIRECTOR DEL PROYECTO

---

Ing. Marco Singaña

CODIRECTOR DEL PROYECTO

## **AGRADECIMIENTO**

Nuestro más profundo agradecimiento a Dios, nuestros padres, hermanos y familiares que nos han apoyado a lo largo de nuestra carrera.

A nuestro director Ing. Franklin Silva, codirector Ing. Marco Singaña, Ing. Nancy Guerrón, por brindarnos su amistad, confianza y valiosos consejos, en el desarrollo del presente proyecto de grado.

Cristian Paúl Recalde Tapia  
Patricio Sebastián Zurita Armijos

## **DEDICATORIA**

Este trabajo esta dedicado a mis queridos Padres Victor y Margoth por creer siempre en mí y siempre estar a mi lado en los momentos más difíciles, por brindarme su apoyo incondicional, y por enseñarme que nada es imposible de alcanzar, a mis hermanos por brindarme su confianza, cariño y amistad; a mis familiares y a todas las personas que con sus palabras y consejos me ayudaron a lograr uno de mis objetivos en mi vida.

Reitero mis agradecimientos a toda mi familia, porque constituyen la parte fundamental de mi vida, muchas gracias por todo el cariño y confianza que me han demostrado para cumplir con este propósito.

Cristian Paul Recalde Tapia.

## DEDICATORIA

Un hombre vive de lo que sueña y sobrevive de lo que siembra, hay que tener en cuenta que solo fracasamos cuando dejamos de intentarlo y ahí debemos saber levantarnos.

El presente trabajo esta dedicado a todas las personas que tienen la seguridad en mí y no solo creyeron. A mis padres Pablo y Francisca que me enseñaron que hay diferencia entre conocer el camino y recorrerlo, con un ejemplo de sencillez, humildad y abnegado trabajo, a mis hermanos que me dan la confianza y me enseñan que las limitaciones que se me presentan solo son las que pongo yo, llegándolas a superar y estar en este punto de mi vida, a todos aquellos que aportando con sus consejos y alientos me han llevado a cumplir un gran propósito de esta mi vida.

Reitero mis más sinceros agradecimientos a mi familia que con su apoyo incondicional me han ayudado a cumplir éste propósito que un día me planteé y ahora se concreto.

Patricio Sebastián Zurita Armijos.

# INDICE

## CAPITULO I

### FUNDAMENTOS

1.1 Descripción del Problema	1
1.2 Variadores de Frecuencia	4
1.2.1 Generalidades	4
1.2.2 Métodos de Control de Velocidad	7
1.2.2.1 Modulación por Ancho de Pulso PWM	7
1.2.2.2 Ajuste de la Frecuencia de Alimentación	7
1.2.2.3 Generación De V/Hz Constante	8
1.2.2.4 Modulación Del Espacio Vectorial	11
1.2.3 Control electrónico de Motores AC.	12
1.2.3.1 Diagrama en Bloques de un Drive de Ca	13
1.2.3.2 Controladores Vectoriales a Lazo Abierto	14
1.3 Características Generales de los Microcontroladores PIC	14
1.3.1 Introducción	14
1.3.2 Microcontrolador 16F877	16
1.3.2.1 Detalle de pines de un PIC de 40 pines	17
1.3.2.2 Descripción de Pines para la programación del PIC 16F877	18
1.3.2.3 Características de Microcontroladores PIC de 40 pines	19
1.3.3 Lenguaje Ensamblador	19
1.3.4 Sistema de Adquisición de Datos	21
1.3.5 Transmisión de Información	22
1.3.5.1 Señalización	22
1.3.6 Otras Particularidades	24
1.4 Dispositivos Para la Visualización de Información	26

---

1.4.1	Display	26
1.4.2	LCD's	27
1.4.2.1	Aplicaciones	28
1.4.4.2	Puesta en servicio	29
1.5	Dispositivos para el Ingreso de Información	29
1.5.1	Teclado numérico serie RS232-TTL	29
1.5.1.1	Aplicaciones	30
1.5.1.2	Características técnicas principales	30
1.6	Transductores para Cantidades Eléctricas	30
1.7	Procesamiento de Datos por medio de LabVIEW	32
1.7.1	Introducción	32
1.7.1.1	Panel Frontal	34
1.7.1.2	Diagrama de bloques	35
1.7.1.3	Paletas	36
1.7.1.4	Paleta de herramientas (Tools palette)	36
1.7.1.5	Paleta de controles (Controls palette)	36
1.7.1.6	Paleta de funciones (functions palette)	37
1.7.2	LabVIEW como Herramienta de Adquisición de Datos	38
1.7.3	Adquisición de Datos por los Puertos Serie y Paralelo	39
1.7.3.1	Manejo del Puerto Serie	39
1.7.3.1.1	Direccionamiento del puerto Serie	40
1.7.3.2	Manejo del puerto paralelo	41
1.7.4	Software de Visualización y Monitoreo	43

## **CAPITULO II**

### **ANÁLISIS Y DISEÑO**

2.1	Diagrama de Bloques del Proyecto	45
2.2	Descripción de Cada Etapa del Diagrama	47
2.2.1	Placa Base para PIC 16F877	47
2.2.1.1	Descripción	47

---

2.2.1.2	Aplicaciones	47
2.2.2	Teclado numérico serie RS232-TTL	48
2.2.2.1	Descripción	48
2.2.2.2	Aplicaciones	49
2.2.2.3	Características técnicas principales	49
2.2.2.4	Conexión de la Placa Base al Teclado numérico serie RS232-TTL	49
2.2.2.5	Definición del Puerto Serie en el Software	50
2.2.2.6	Configuración de los puertos del Microcontrolador	50
2.2.2.7	Captura de datos del teclado	50
2.2.3	Interfaz de display LCD sincrónico	51
2.2.3.1	Descripción	51
2.2.3.2	Configuración de la librería lcdser.h	51
2.2.3.3	Arranque del display y ejemplo de uso.	51
2.2.3.3.1	Conector para control del interfaz	53
2.2.3.3.2	Conector para alimentación	53
2.2.4	Variador de Frecuencia ABB ACS 300	53
2.2.4.1	Conexión a la red	53
2.2.4.2	Conexión a motor	54
2.2.4.3	Conexiones de control programables	54
2.3	Especificación de Requisitos del Sistema	56
2.3.1	Computador Personal	56
2.3.1.1	Características Técnicas del Computador Personal	57
2.4	Diseño del hardware	57
2.4.1	Programador: PIC Programmer II	57
2.4.2	Placa Base	59
2.4.3	Convertidor AD-DA	61
2.4.4	Teclado numérico serie RS232-TTL	62
2.4.5	Interfaz de display LCD sincrónico	63

2.5 Diseño del software	64
2.5.1 IC-Prog	64
2.5.1.1 Área de Código (Programa)	65
2.5.1.2 Área de datos (eeprom)	66

## **CAPITULO III**

### **RESULTADOS Y PRUEBAS EXPERIMENTALES**

3.1 Descripción del sistema	67
3.1.1 Control Local	67
3.1.2 Control Remoto	70
3.2 Pruebas experimentales	71
3.2.1 Control Local	71
3.2.2 Control Remoto	76
3.3 Análisis de Resultados	80
3.3.1 Análisis del Control Local	80
Análisis del literal 3.2.1.a) Incremento de frecuencia	80
Análisis del literal 3.2.1.b) Decremento de frecuencia	80
Análisis del literal 3.2.1.c) Realizar Rampas	80
Análisis del literal 3.2.1.d) Realizar Procesos	81
3.3.2 Análisis del Control Remoto	81
Análisis del literal 3.2.2.a) Visualización	81
Análisis del literal 3.2.1.b) Visualización y Control	82
Análisis del literal 3.2.2.b.2) Incremento de frecuencia	82
Análisis del literal 3.2.2.b.3) Decremento de frecuencia	82
Análisis del literal 3.2.2.b.4) Realizar Rampas	82

---

Análisis del literal 3.2.2.b.5) Realizar Frecuencias Constantes.	83
3.4 Alcances y Limitaciones	84
3.4.1 Alcances	84
3.4.2 Limitaciones	86
3.5 Análisis Técnico – Económico	87
3.5.1 Introducción	87
3.5.2 Conclusión Técnico – Económico	88
<b>CONCLUSIONES</b>	<b>89</b>
<b>RECOMENDACIONES</b>	<b>91</b>
<b>GLOSARIO</b>	<b>94</b>
<b>ANEXOS</b>	<b>96</b>
<b>REFERENCIAS BIBLIOGRÁFICAS</b>	

## PRÓLOGO

La automatización de los procesos industriales a través de los años ha dado lugar a un avance espectacular de la industria. Todo ello ha sido posible gracias a una serie de factores entre los que se encuentran las nuevas tecnologías en el campo mecánico o industrial, la introducción de los computadores y con éstos software desarrollados para tareas específicas (instrumentación, control, comunicaciones, entre otros) y sobre todo el control y la regulación de sistemas y procesos, esto a llevado a encontrar nuevas formas de combinarlas para solucionar problemas industriales, de una misma forma pero con diferentes equipos o elementos.

La aparición de la microelectrónica y de los microprocesadores ha facilitado el desarrollo de técnicas de control complejas, la robotización, la implementación de sistemas de gobierno y la planificación. Todos estos elementos llevan consigo la reducción de costos, el aumento de la productividad y la mejora de calidad del producto.

Sabemos que la automatización industrial es la ejecución de procesos controlados mecánica o electrónicamente, liberando al ser humano de operaciones rutinarias, disminuyendo así errores y a su vez aumentando la producción, todo esto nos ha llevado a realizar este proyecto, conjugando todo lo expuesto, plantearnos un problema y solucionarlo.

El problema era controlar un variador de frecuencia, sus variables, y llevarlo hacer tareas específicas de una manera local a través de un teclado y remota a través de un computador personal, de aquí nació la idea **del “DISEÑO Y CONSTRUCCION DE UN CONTROL: LOCAL Y REMOTO DE UN VARIADOR DE FRECUENCIA PARA EL MANEJO DE UN MOTOR AC UTILIZANDO EL MICROCONTROLADOR PIC 16F877 Y LA PLATAFORMA DE DESARROLLO LABVIEW”**. Este módulo tiene la finalidad de controlar a un variador de frecuencia, y éste a su vez motores de corriente alterna.

Los elementos utilizados para el diseño están en el mercado, un LCD, un Variador de Frecuencia ACS 300, un Teclado numérico serie RS232-TTL, un Interfaz de display LCD sincrónico, un Conversor D/A, una Placa base para PIC 16F877, la licencia del software utilizado tiene la Escuela Politécnica del Ejército – Latacunga, los costos serán detallados en la realización del proyecto.

Para cumplir con los objetivos planteados, el proyecto esta dividido en tres capítulos, a continuación se detalla cada uno de estos.

Capítulo I: Se aborda la parte teórica, los fundamentos de los elementos que intervienen en el desarrollo del proyecto, así como, conceptos generales sobre la parte de Hardware, como las placas, el computador personal, las interfases, variadores de frecuencia, motores AC, entre otros, y Software utilizado LabVIEW, además de una introducción referida al presente proyecto.

Capítulo II: Se trata sobre el análisis y diseño del proyecto de la parte de hardware y software, la forma en que se comunican las diferentes interfases, con el variador de frecuencia y la PC; detalla características para la programación de éstas, instrucciones para habilitarlas, comandos que se utiliza en el programa general del proyecto, realizado en Lenguaje “C”, configuración de puertos de la PC y de los software’s utilizados para el desarrollo del proyecto.

Capítulo III: Este capítulo trata de resultados y pruebas experimentales, una descripción general del sistema, pruebas experimentales en forma real del proyecto, realizando un respectivo análisis de los datos obtenidos, sus alcances y limitaciones, al final del capítulo se realizó un análisis técnico – económico del proyecto, con sus respectivas conclusiones.

## CAPITULO I

### FUNDAMENTOS

#### 1.1 DESCRIPCIÓN DEL PROBLEMA.

<sup>1</sup>La gran mayoría de industrias incorporan en sus respectivos procesos una infinidad de elementos sean estos mecánicos, eléctricos, electrónicos o neumáticos, los mismos que cumplen determinadas funciones para llegar al cumplimiento de su cometido final, esto es el de realizar una acción concreta de acuerdo al tipo de proceso a tratarse.

El motor de corriente alterna, a pesar de ser un motor robusto, de poco mantenimiento, liviano e ideal para la mayoría de las aplicaciones industriales, tiene el inconveniente de ser un motor rígido en cuanto a su velocidad. La velocidad del motor asincrónico depende de la forma constructiva del motor y de la frecuencia de alimentación.

Como la frecuencia de alimentación que entregan las Compañías de electricidad es constante, la velocidad de los motores asincrónicos es constante, salvo que se varíe el número de polos, el deslizamiento o la frecuencia.

El método más eficiente de controlar la velocidad de un motor eléctrico es por medio de un variador electrónico de frecuencia. No se requieren motores especiales, son mucho más eficientes y tienen precios cada vez más competitivos.

---

<sup>1</sup> La información suministrada en este capítulo, ha sido obtenida del material detallado en la bibliografía.

El variador de frecuencia regula la frecuencia del voltaje aplicado al motor, logrando modificar su velocidad. Sin embargo, con el cambio de frecuencia, debe variarse el voltaje aplicado al motor para evitar la saturación del flujo magnético con una elevación de la corriente que dañaría el motor, esto se realiza automáticamente por el cambio de frecuencia.

Los Microcontroladores son una herramienta muy poderosa en el diseño de aplicaciones que llama la atención por su reducido tamaño, bajo costo y características técnicas que los hacen altamente aplicables. En el presente proyecto la inclusión del microcontrolador PIC 16F877, resulta adecuado por sus magníficas prestaciones, que constituyen un gran atractivo para el desarrollo del proyecto en cuestión.

Es por esta razón, que los autores pretenden realizar un control electrónico de un variador de frecuencia para un motor AC, para a través de éste, controlar parámetros como velocidad, sentido de giro y frenado, localmente desde un módulo electrónico o en forma remota a través de una PC del motor AC, con lo que se asemejaría a tener un sistema de control SCADA<sup>2</sup>, el cuál puede utilizarse para fines de investigación en el laboratorio.

De igual manera, existen procesos en los cuales el ambiente en el que se desarrolla puede ser inaccesible ya sea por la presencia de agentes como el ruido, gases, ácidos, etc, el motor de todas maneras no debe ser sustituido, entonces es imprescindible establecer un control por mando remoto dada la situación física incómoda especialmente para el operario. Por lo que estableciendo una comunicación vía RS-232 entre un PIC y la PC y utilizando para ello la plataforma de desarrollo LabVIEW<sup>3</sup>, que es un software que brinda muchas ventajas y facilidades de programación, resulta ser el indicado para poder cumplir con el fin de esta etapa del

---

<sup>2</sup> Supervisión Control y Adquisición de Datos

<sup>3</sup> Software de programación gráfica

proyecto, permitiendo la visualización, monitoreo y el control del variador de frecuencia y de éste al motor AC.

En la actualidad se cuenta con una tecnología en lo que respecta a elementos electrónicos tales como microcontroladores, circuitos integrados, LCD's<sup>4</sup>, conjuntamente con un computador facilitan el desarrollo de una aplicación. Se pretende aprovechar al máximo estas herramientas de hardware disponibles en el mercado a fin de construir un módulo que sea de fácil interpretación para cualquier individuo y de esta manera no constituya un inconveniente el hecho de adquirir conocimientos muy profundos de la Electrónica para operar la presente aplicación.

Al disponer las variables físicas y eléctricas que se relacionan con un motor, se pretende también representarlas en forma gráfica, esto ayudará a una mejor comprensión de las curvas características que el fabricante del motor las puede suministrar, para establecer un análisis de resultados y formulación de las respectivas conclusiones, haciendo las comparaciones y contrastaciones pertinentes.

En resumen, el proyecto demandará el análisis e investigaciones del caso en los siguientes aspectos:

- Control electrónico de un variador de frecuencia.
- Utilización de los recursos disponibles en un microcontrolador PIC 16F877 tales como interrupciones, puertos I/O, comunicación serial, generación de PWM, etc.
- Manejo de un Visualizador de Cristal Líquido LCD.
- Manejo de un teclado como dispositivo de entrada de datos al PIC.
- Comunicación serial empleando LabVIEW.
- Software de desarrollo de las Interfaces Gráficas empleando LabVIEW.

---

<sup>4</sup> Visualizadores Cristal Líquido, ver [www.datex-ohmeda.es/aula-bioingenieria/numero8/Tecnologia\\_LCD.htm](http://www.datex-ohmeda.es/aula-bioingenieria/numero8/Tecnologia_LCD.htm)

## 1.8 Variadores de Frecuencia

### 1.8.1 Generalidades.

Está demostrado que el método más eficiente para el control de la velocidad de un motor eléctrico asíncrono es por medio de la variación de frecuencia y por lo tanto utilizando un Variador de Frecuencia. Además de su eficiencia, se puede nombrar otras ventajas de los Variadores de Frecuencia: economizan, puesto que no requieren de motores especiales; son más eficientes y más precisos; y además prolongan la vida útil del motor al evitar vibraciones.

Los variadores de frecuencia están compuestos por :

- **Etapa Rectificadora.** Convierte la tensión alterna en continua mediante rectificadores de diodos, tiristores, etc.
- **Etapa intermedia.** Filtro para suavizar la tensión rectificada y reducir la emisión de armónicos.
- **Inversor o "Inverter".** Convierte la tensión continua en otra de tensión y frecuencia variable mediante la generación de pulsos. Actualmente se emplean IGBT's (Isolated Gate Bipolar Transistors) para generar los pulsos controlados de tensión. Los equipos más modernos utilizan IGBT's inteligentes que incorporan un microprocesador con todas las protecciones por sobrecorriente, sobretensión, baja tensión, cortocircuitos, puesta a masa del motor, sobretemperaturas, etc.
- **Etapa de control.** Esta etapa controla los IGBT para generar los pulsos variables de tensión y frecuencia. Y además controla los parámetros externos en general, etc.

Los variadores más utilizados, utilizan modulación PWM (Modulación de Ancho de Pulsos) y usan en la etapa rectificadora puente de diodos rectificadores. En la etapa intermedia se usan condensadores y bobinas para disminuir las armónicas y mejorar el factor de potencia. Los fabricantes que utilizan bobinas en

la línea, en lugar del circuito intermedio tienen la desventaja de ocupar más espacio y disminuir la eficiencia del variador.

El Inversor o Inverter convierte la tensión continua de la etapa intermedia en una tensión de frecuencia y tensión variables. Los IGBT envían pulsos de duración variable y se obtiene una corriente casi senoidal en el motor.

La frecuencia portadora de los IGBT se encuentra entre 2 a 16kHz. Una portadora con alta frecuencia reduce el ruido acústico del motor pero disminuye el rendimiento del motor y la longitud permisible del cable hacia el motor. Por otra parte, los IGBT's generan mayor calor.

Las señales de control para arranque, parada y variación de velocidad (potenciómetro o señales externas de referencia) deben estar aisladas galvánicamente para evitar daños en sensores o controles y evitar ruidos en la etapa de control.

Los variadores de frecuencia tienen sus principales aplicaciones en los siguientes tipos de máquinas:

- Transportadoras.
- Bombas y ventiladores centrífugos.
- Bombas de desplazamiento positivo.
- Ascensores y elevadores.
- Extrusoras.
- Centrífugas.
- Prensas mecánicas y balancines.
- Máquinas textiles.
- Compresores de aire.
- Pozos petroleros.
- Otras aplicaciones: Elevadores de cangilones, transportadores helicoidales, continuas de papel, máquinas herramientas, máquinas para soldadura, pantógrafos, máquinas para vidrios, fulones de

curtiembres, secaderos de tabaco, clasificadoras de frutas, conformadoras de cables, trefiladoras de caños, laminadoras, mezcladoras, trefiladoras de perfiles de aluminio, cable, etc, trituradoras de minerales, trapiches de caña de azúcar, balanceadoras, molinos harineros, hornos giratorios de cemento, hornos de industrias alimenticias, puentes grúa, bancos de prueba, secadores industriales, tapadoras de envases, agitadores, cardeadoras, dosificadoras, dispersores, reactores, pailas, lavadoras industriales, lustradoras, molinos rotativos, pulidoras, fresas, bobinadoras y desbobinadoras, arenadoras, separadores, vibradores, cribas, locomotoras, vehículos eléctricos, escaleras mecánicas, aire acondicionado, portones automáticos, plataformas móviles, tornillos sin fin, válvulas rotativas, calandras, tejedoras, chipeadoras, extractores, posicionadores, etc.

Entre las Industrias que utilizan variadores de frecuencia se tiene:

- Metalúrgicas
- Alimenticias
- Construcción
- Automovilísticas
- Plásticos
- Papeleras
- Cueros
- Químicas
- Petroleras
- Textiles
- Madereras
- Caucho
- Otras: Aeronáuticas, tabacaleras, vidrio, aguas sanitarias, cerealeras, universidades, empresas de ingeniería, minería, acerías, agropecuarias, preparadores de vehículos de competición, etc.

De ello se deduce que la electrónica digital al presentar una facilidad en el acople e interconexión con el Variador de Frecuencia, permiten desarrollar un sistema de control electrónico. Los controles de motores eléctricos fueron desarrollados para controlar la operación de los mismos. Un controlador de un motor está en la capacidad de monitorear y controlar las acciones del motor.

### **1.8.2 Métodos de Control de Velocidad.**

Los métodos de control de velocidad son variados, dependiendo de los parámetros a variar, a continuación se presenta algunos métodos.

#### **1.2.2.1 Modulación por Ancho de Pulso PWM.**

Un Método de Control de velocidad es la Modulación por Ancho de Pulso PWM, técnica utilizada en áreas como las Telecomunicaciones y Sistemas de Control en donde permiten el desarrollo de aplicaciones como control de temperatura, control de servos. La técnica consiste en un tren de pulsos a frecuencia fija a los cuales se les varía sus anchos. Las características que definen una señal PWM son dos: Ciclo de Trabajo y la Frecuencia.

El ciclo de trabajo es el tiempo, representado en porcentaje, que la señal esta "alta" (nivel lógico "1"), o lo que es lo mismo, el tiempo que la salida permanece encendida. La Frecuencia del tren de pulsos se define en función de la aplicación condicionada por tiempos de respuesta, rango útil de los pulsos, etc.

Lo que hace un controlador PWM, es conmutar la alimentación del motor a alta velocidad entregando una serie de pulsos. El control de la velocidad del motor se obtiene variando (modulando) el ancho de los pulsos, de esta manera se logra tener un control al voltaje aplicado en el motor AC.

#### **1.2.2.2 Ajuste de la Frecuencia de Alimentación.**

Mediante el ajuste de la frecuencia de alimentación y el voltaje de alimentación que es suministrada al motor a inducción, se ajusta la velocidad sincrónica del motor. Esto brinda un aumento en la familia de curvas torque vs velocidad del motor (figura 1.1).

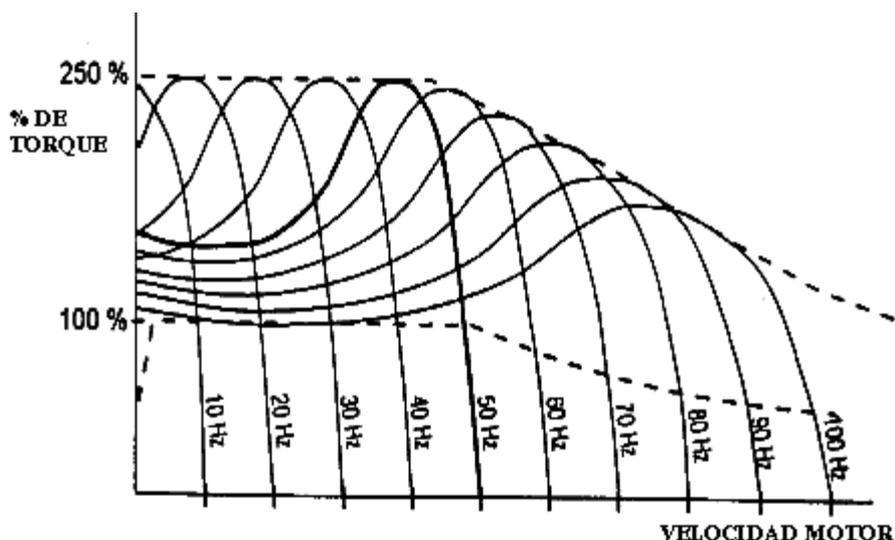


Figura 1.1. Curvas de Torque vs Velocidad Variando la Frecuencia

1.2.2.3 Generación de Voltaje/Frecuencia (V/Hz) Constante.

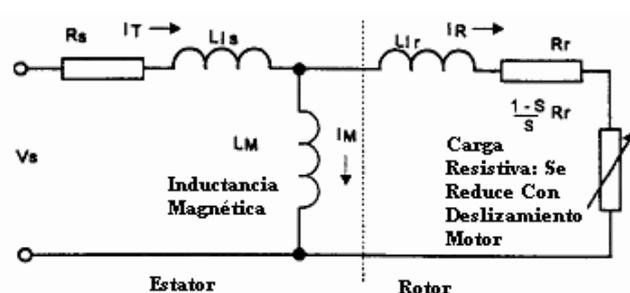


Figura 1.2.1. Circuito Equivalente Completo

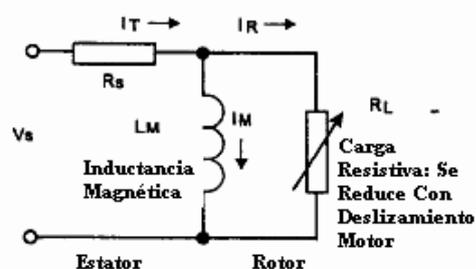


Figura 1.2.2. Circuito Equivalente Simplificado

Figura 1.2. Circuito Equivalente de un Motor a Inducción de Fase

<sup>5</sup>Con respecto a la figura 1.2.1. es el circuito equivalente completo de una fase de un motor a inducción jaula de ardilla. Estudiando este circuito, se pueden entender las características de salida requeridas de un controlador de frecuencia variable. En la figura mencionada:

$L_{is}$  = Inductancia de fuga del estator – debida a un imperfecto acoplamiento magnético entre los bobinados adyacentes del estator

$R_s$  = Resistencia del bobinado del estator – debida a la resistencia del hilo de cobre – causa de pérdidas de estator

<sup>5</sup> Principios de los Controladores de Motores a Inducción. Lisjak Leandro

$L_m$  = Inductancia de magnetismo del estator – aumenta al campo rotatorio del estator

$L_{lr}$  = Inductancia de fuga del rotor – debida a un imperfecto acoplamiento magnético entre barras adyacentes del rotor

$R_r$  = Resistencia del rotor – debida a la resistencia de las barras del rotor

$1 - S$

$R_L = \frac{R_T}{S} =$  Resistencia de carga equivalente, cambia con el deslizamiento del motor.

El circuito puede ser simplificado ignorando las inductancias de fuga y combinando  $R_r$  con  $R_L$ . Esto se indica en la figura 1.2.2. Cabe destacar que existen dos caminos paralelos para la corriente del motor en este circuito equivalente.

Un camino es la corriente de carga ( $I_R$ ). Esta corriente es “real” y fluye hacia el rotor. Esta corriente y el deslizamiento del motor aumentan con el incremento de carga del motor.

El otro camino es la corriente de magnetismo ( $I_M$ ), y fluye hacia el estator. Esta corriente es “imaginaria” (está retrasada  $90^\circ$  de la corriente real). La corriente de magnetismo determina el flujo en el motor. Es el resultado del entrehierro entre el rotor y el estator. Esta corriente debería mantenerse esencialmente constante a medida que la carga es variada.

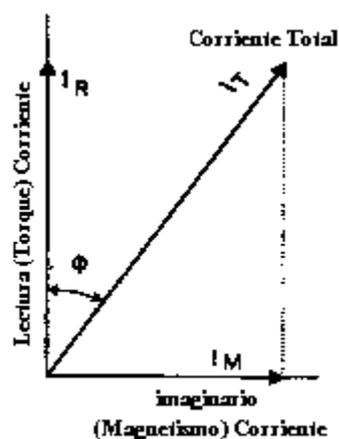


Figura 1.3.1. Carga Completa

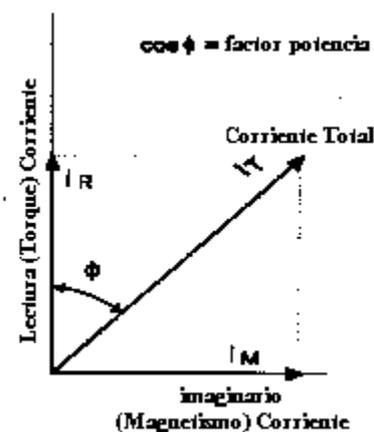


Figura 1.3.2. Carga Media

**Figura 1.3. Diagrama Vectorial de la Corriente del Motor**

La Figura 1.3 ilustra estos componentes de corriente como vectores. Note como con carga en incremento, el **ángulo de factor de potencia** entre la corriente total  $I_T$  y la corriente de carga  $I_R$  se reduce, mejorando así el factor de potencia del motor. También la corriente de magnetismo  $I_M$  se mantiene constante, con respecto de la carga. Esta corriente de magnetismo es la que genera el campo magnético del estator, afectando así la capacidad de generación de torque del motor.

La magnitud de esta corriente de magnetismo puede ser calculada, mediante la siguiente ecuación:

$$I_M = \frac{V}{2\pi fL}$$

Donde:

$V$  = Tensión de alimentación

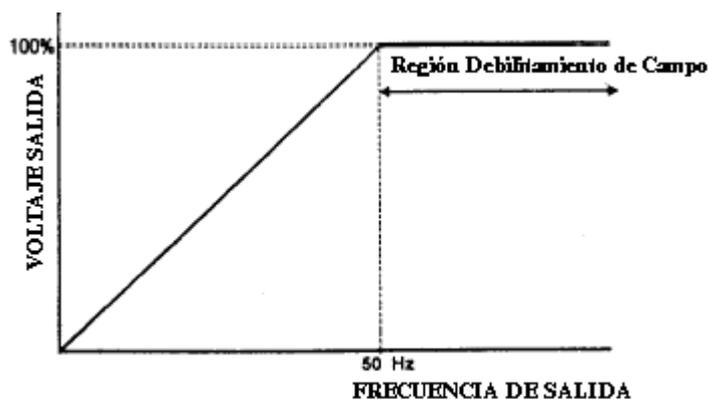
$f$  = Frecuencia de alimentación

$L$  = Inductancia de magnetismo del estator

De esto, se puede observar que si la frecuencia de alimentación es reducida, se reduce la tensión, entonces la corriente de magnetismo se incrementará inversamente con la frecuencia.

Este efecto causará que el motor se sature magnéticamente, a menos que la tensión suministrada al motor sea reducida con la frecuencia.

Entonces, en un controlador de velocidad electrónico, la tensión de alimentación al motor debe ser ajustada en proporción con la frecuencia, para mantener la corriente en el camino de magnetismo constante.



**Figura 1.4** Tensión de Salida vs Frecuencia para Controladores de Motor de Frecuencia Variable.

Con respecto a la figura 1.4. En el controlador de velocidad, a medida que la frecuencia es ajustada de 0 a 60 Hz, la salida de tensión es ajustada de 0 a 100% (p. ej. 0 a 400 Vca); esto asegura condiciones de flujo para el motor aparentemente constantes. Esto se conoce como mantenimiento de un valor de tensión-frecuencia constante.

Se observa que la frecuencia puede ser incrementada más allá de los 60 Hz. Así la velocidad del motor puede ser aumentada. Sin embargo, la tensión al motor no puede ser incrementada más allá del 100%, debido a la tensión de alimentación limitada. De esta manera, encima de 60 Hz, el valor tensión-frecuencia se reduce, reduciendo así el flujo en el motor. Esto reduce el torque total del motor disponible a velocidades por encima de 60 Hz. Esto es conocido como **debilitamiento de campo**.

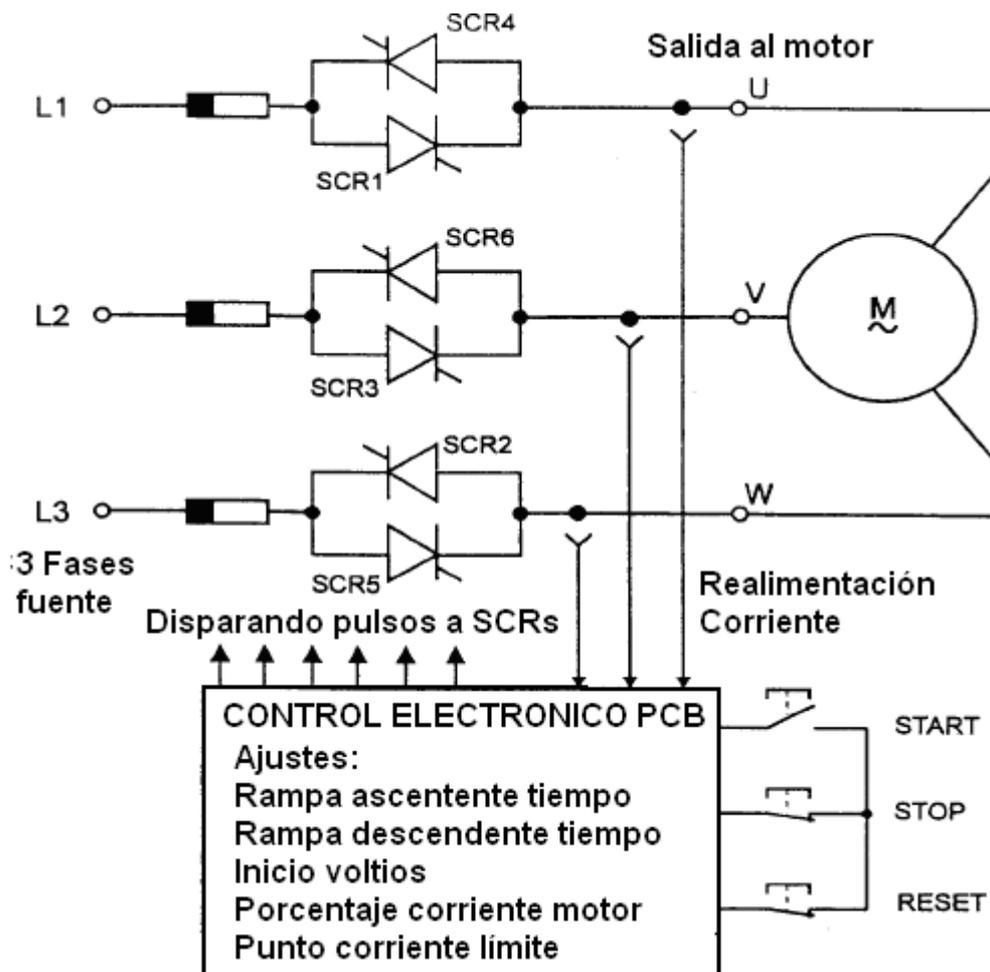
#### 1.2.2.4 Modulación Del Espacio Vectorial.

Una onda senoidal de tres fases puede ser representada como un conjunto de tres vectores rotatorios (fasores). La velocidad de rotación (revs/seg) se relaciona directamente con la frecuencia, las posiciones vectoriales instantáneas se relacionan con el punto en el ciclo y el largo del vector representa la magnitud de tensión o corriente que está siendo representada.

En la figura 1.5 se presenta la tensión de fase (línea a neutro), es representada por las longitudes  $V_{A-N}$ ,  $V_{B-N}$ ,  $V_{C-N}$ , mientras que la tensión de línea es representada por las distancias entre las puntas de flecha. ( $V_{A-B}$ ,  $V_{B-C}$ ,  $V_{C-A}$ ).



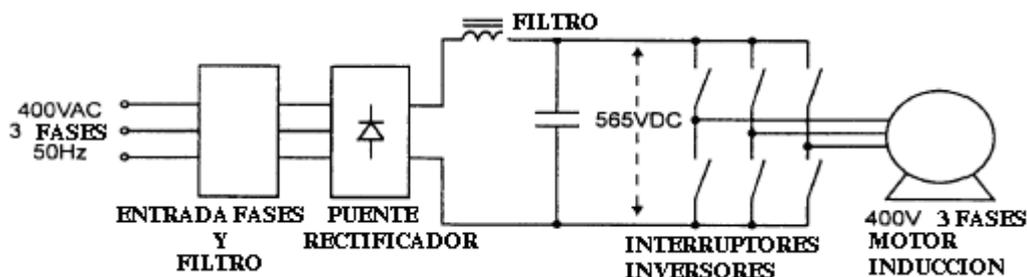
Estos SCRs tienen la fase controlada por los pulsos de disparo de la Electrónica de Control de la PCB. Estos pulsos de control están dirigidos a cada uno de los seis SCRs por un transformador de pulsos u opto-acoplador, para dar aislación galvánica entre la Electrónica de Control y los circuitos de potencia. De esta manera el motor puede ser arrancado y parado, y la tensión en el motor puede ser controlada automáticamente.



*Figura 1.6 Configuración del Circuito de un Arrancador Electrónico de Tensión Reducida*

### 1.2.3.1 Diagrama en Bloques de un Driver de Ca

En la figura 1.7 se muestra un diagrama en bloques simplificado de un Drive de CA. Una red de tres fases alimenta la entrada. Aquí es rectificadada a CC y filtrada por un filtro L-C. Este "Bus de CC" alimenta un puente inversor de tres fases, donde es convertida nuevamente a una forma de onda de CA de tres fases, pero a la tensión y frecuencia necesarias para funcionar el motor a la velocidad deseada.



**Figura 1.7** Diagrama en Bloques del Drive de CA

### 1.2.3.2 Controladores Vectoriales a Lazo Abierto

Un controlador vectorial de flujo a lazo abierto (también llamado controlador vectorial sin encoder) controla la velocidad de un motor a inducción usando técnicas similares a las del Microvector, para sintetizar las corrientes del estator para entregar el torque requerido del motor. Sin embargo no se requiere un encoder de eje en el motor. El resultado es un controlador de motor a inducción que tiene un cercano desempeño vectorial (tanto en respuesta de velocidad como en capacidad de torque a baja velocidad) pero sin el problema de tener que conectar un encoder de eje en el motor, es decir sin realimentar la señal de velocidad.

La posición del motor es estimada midiendo la tensión en los terminales del motor y aplicando una transformación matemática. Tal controlador requiere un microprocesador muy poderoso para manejar rutinas matemáticas extras en el corto ciclo de tiempo requerido.

El control vectorial a lazo abierto está en sus inicios, por lo que no se puede mostrar algún grafico, se puede decir que será probablemente el modo de control del futuro.

## 1.3 Características Generales de los Microcontroladores PIC

### 1.3.1 Introducción

Desde la invención del circuito integrado, el desarrollo constante de la electrónica digital ha dado lugar a dispositivos cada vez más complejos. Entre

ellos los microprocesadores y los microcontroladores, los cuales son básicos en las carreras de ingeniería electrónica.

Se pretende explicar conceptos que ya han sido probados y aceptados, considerando que su vigencia se mantendrá en un periodo mas o menos largo. Esta limitación no se considera importante, ya que si se comprende a fondo un microcontrolador, los demás pueden aprenderse con facilidad partiendo de la estructura del primero. Además, el estudio de un microcontrolador particular elimina la posibilidad de una presentación superficial o confusa y permite al usuario enfrentarse a problemas reales en la práctica.

Para estar al tanto de las innovaciones se recomienda que el usuario se mantenga en contacto con la pagina electrónica de Microchip Technology, <http://www.microchip.com>, solo así podrá obtener un conocimiento completo y actualizado, tanto de los aspectos que aquí se tratan, como de los nuevos que vayan surgiendo.

La forma de estructurar los temas responde a la experiencia de que, para el usuario que por primera ocasión estudia microcontroladores, resulta más sencillo aprender primero lo referente a la construcción interna del dispositivo y la arquitectura general de los microcontroladores y una vez dominada pasar entonces a los detalles electrónicos de la circuiteria y la programación.

A diferencia del procesamiento de aplicaciones, tal es el caso de las PC's y Estaciones de Trabajo, el control de los elementos mediante computo requiere que existan dispositivos de control dentro de las aplicaciones. Al usuario del producto solo le interesa qué es lo que necesita saber para utilizar la interfaz que se le presente (keypad, teclados, comandos), siempre y cuando funcione y/o cumpla con sus expectativas y objetivos. En muy pocas ocasiones el usuario final conoce (o le interesa conocer) que sistema de control es el que se encuentra incrustado en su aplicación (a diferencia de las personas que adquieren una PC, que se preocupa del tipo de procesador, memoria, velocidad de reloj, etc.).

Así es que, de cualquier forma, es vital para la mayoría de los diseñadores de aplicaciones con elementos de control dentro de ellas, seleccionar los

dispositivos y las compañías más indicadas de controladores. Los productos de control incrustado se encuentran en la mayoría de los sectores del mercado: sector comercial, consumibles, periféricos de computadoras, telecomunicaciones (incluyendo los productos de telecomunicación personal de emergencia), automotriz e industrial. La mayoría de los productos de control incrustado deben satisfacer requerimientos especiales: eficiencia, bajo costo, baja potencia y un alto nivel de integración en los sistemas.

### **1.3.2 Microcontrolador 16F877.**

El microcontrolador 16F877 pertenece a la gama alta, estos dispositivos responden a microcontroladores de arquitectura abierta pudiéndose expansionar en el exterior al poder sacar los buses de datos, direcciones y control. Así se pueden configurar sistemas similares a los que utilizan los microprocesadores convencionales, siendo capaces de ampliar la configuración interna del PIC añadiendo nuevos dispositivos de memoria y de E/S externas. Esta facultad obliga a estos componentes a tener un elevado número de pines comprendido entre 40 y 44. Admiten interrupciones, poseen puerto serie, varios temporizadores y mayores capacidades de memoria, que alcanza los 8 k palabras en la memoria de instrucciones y 454 bytes en la memoria de datos.

Los microcontroladores de la gama alta reúnen las prestaciones más completas a un precio muy competitivo. Disponen de comparadores analógicos, convertidores Analógicos/Digitales (CAD) de 8 bits, circuito de Captura/Comparación, Interfaz serie sincrónico SP/I<sup>2</sup> de 2, 3 o 4 hilos, Interfaz serie asíncrónico SCI de 8 o 9 bits, PWM para manipular impulsos de 8 o 10 bits con modulación de anchura y PSP, que consiste en un Puerto paralelo esclavo que proporciona la conexión con otros microprocesadores. Además, estos recursos se distribuyen eficazmente entre los diversos modelos para que el usuario disponga de un gran abanico de posibilidades para optimizar su elección.

Todos los dispositivos de esta gama disponen de reset automático al conectar la alimentación, reset ante el fallo de la alimentación, Perro guardián, Código de Protección opcional y líneas de E/S digitales de alta corriente.

En la figura 1.8 se presenta la distribución de los pines del Microcontrolador PIC 16F877.

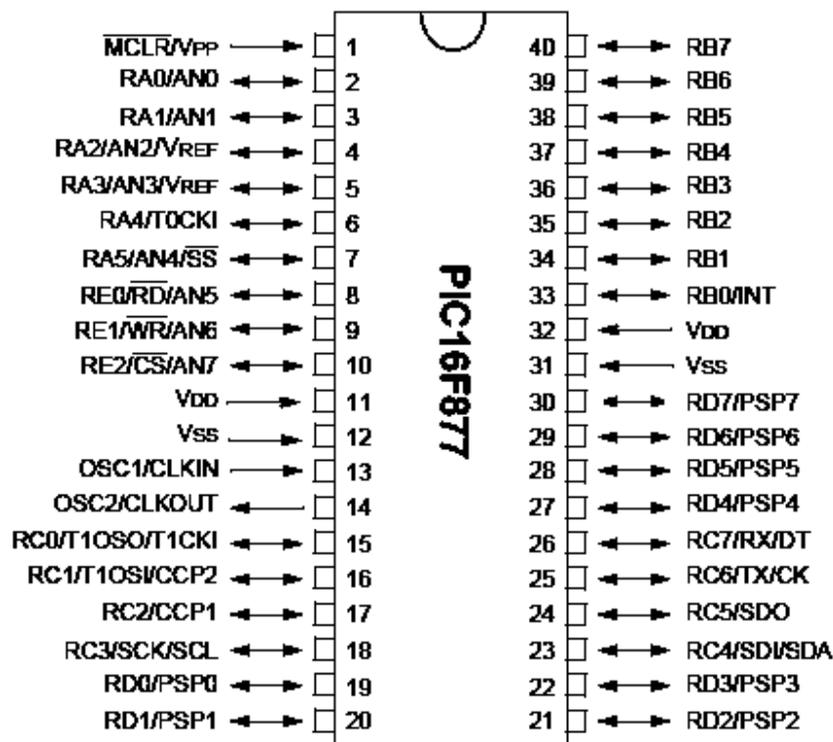


Figura 1.8 Descripción pines Microcontrolador PIC 16F877

### 1.3.2.1 Detalle de pines de un PIC de 40 pines

A continuación se detalla los pines de un PIC de 40 pines:

- RA0 → An0 Entrada analógica 0
- RA1 → An1 Entrada analógica 1
- RA2 → Entrada Digital 0
- RA3 → An1 Entrada analógica 3
- RA4 → Entrada Digital 1
- RA5 → Entrada Digital 2
- RE0 → Ck LCD
- RE1 → Data LCD
- RE2 → Strobe LCD
- RB0 → Keyboard Input (RS232 TTL, Int0)
- RB1 → Entrada Digital 7 (Int1 18FXXX)
- RB2 → Salida Digital 8 / (Entrada) RX Rs232 (Int2 18FXXX)

- RB3 → Salida Digital 9 / (Out) TX Rs232
- RB4 → Salida Digital t 10 / (Salida) RTS Rs232 JP6=ON
- RB5 → Salida Digital t 11 / (Entrada) CTS Rs232 JP7=ON
- RB6 → Salida Digital 12 / ICP
- RB7 → Salida Digital 13 / ICP
- RC0 → Entrada Digital 3
- RC1 → Entrada Digital 4
- RC2 → Entrada Digital 5
- RC3 → SCL I2C (Serial Clock)
- RC4 → SDA I2C (Serial Data)
- RC5 → CTS COM1 (Enable RX RS485) / Digital Output 7 (JP5=ON sin chip 485)
- RC6 → TX COM1 (RS485)
- RC7 → RX COM1 (RS485)
- RD0 → Digital Output 0 / Buzzer
- RD1 → Salida Digital 1
- RD2 → Salida Digital 2
- RD3 → Salida Digital 3
- RD4 → Salida Digital 4
- RD5 → Salida Digital 5
- RD6 → Salida Digital 6
- RD7 → RTS COM1 RS485(JP4=OFF) / Entrada Digital 6 (JP4=ON, sin chip 485)

### 1.3.2.2 Descripción de Pines para la programación del PIC 16F877

En la tabla 1.1 se indica la descripción de pines para la programación del Microcontrolador PIC 16F877.

Nombre Pin	Durante la Programacion		
	Funcion	Tipo Pin	Descripcion Pin
RB6	CLOCK	I	Entrada Reloj
RB7	DATA	I/O	Entrada/Salida Datos
MCLR	VTEST MODE	P*	Seleccion Modo Programa
VDD	VDD	P	Fuente Poder
Vss	Vss	P	Tierra

**Tabla. 1.1.** Descripción de Pines para la programación del Microcontrolador PIC 16F877

### 1.3.2.3 Características de Microcontroladores PIC de 40 pines

A continuación en la tabla 1.2 se indica las características de Microcontroladores PIC de 40 pines.

<b>Key Features PICmicro™ Mid-Range Reference Manual (DS33023)</b>	<b>PIC16F873</b>	<b>PIC16F874</b>	<b>PIC16F876</b>	<b>PIC16F877</b>
<b>Operating Frequency</b>	DC - 20 MHz			
<b>Resets (and Delays)</b>	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
<b>FLASH Program Memory (14-bit words)</b>	4K	4K	8K	8K
<b>Data Memory (bytes)</b>	192	192	368	368
<b>EEPROM Data Memory</b>	128	128	256	256
<b>Interrupts</b>	13	14	13	14
<b>I/O Ports</b>	Ports A,B,C	Ports A,B,C,D,E	Ports A,B,C	Ports A,B,C,D,E
<b>Timers</b>	3	3	3	3
<b>Capture/Compare/PWM modules</b>	2	2	2	2
<b>Serial Communications</b>	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
<b>Parallel Communications</b>	—	PSP	—	PSP
<b>10-bit Analog-to-Digital Module</b>	5 input channels	8 input channels	5 input channels	8 input channels
<b>Instruction Set</b>	35 Instructions	35 Instructions	35 Instructions	35 Instructions

**Tabla. 1.2. Características de Microcontroladores PIC de 40 pines**

### 1.3.3 Lenguaje Ensamblador.

Si se desea realizar la programación de los microcontroladores PIC en un lenguaje como el C, es preciso utilizar un *compilador de C*.

Dicho compilador genera ficheros en formato Intel-hexadecimal, que es el necesario para programar (utilizando un programador de PIC) un microcontrolador de 6, 8, 18 ó 40 pines.

El compilador de C que se va a utilizar es el PCW de la casa CCS Inc. A su vez, el compilador se integrara en un *entorno de desarrollo integrado* (IDE) que va a permitir desarrollar todas y cada una de las fases que se compone un proyecto, desde la edición hasta la compilación pasando por la depuración de errores. La última fase, a excepción de la depuración y retoques hardware finales, será programar el PIC.

Al igual que el compilador de Turbo C, éste "traduce" el código C del archivo fuente (.C) a lenguaje máquina para los microcontroladores PIC, generando así un archivo en formato hexadecimal (.HEX).

A continuación se presenta el conjunto de las instrucciones en Ensamblador para el microcontrolador (tabla 1.3).

CONJUNTO DE INSTRUCCIONES							
OPERACIONES ORIENTADAS A REGISTROS			11-6		(5)		(4-0)
			CÓDIGO DE OPERACIÓN		d	f (REGISTRO N)	
			d = 0 si el destino es W		d = 1 si el destino es el registro		
binario	Hex	Nombre	Name	Nemónica	Operandos	Operación	Estados afectados
0001 11df ffff	1Cf	Suma W y f	Add W and f	ADDWF	f,d	W+f -> d	C,DC,Z
0001 01df ffff	14f	AND entre W y f	AND W and f	ANDWF	f,d	W & f -> d	Z
0000 011f ffff	06f	Limpia f	Clear f	CLRF	f	0 -> f	Z
0000 0100 0000	040	Limpia W	Clear W	CLRWF	-	0 -> W	Z
0010 01df ffff	24f	Complementa f	Complement f	CONF	f,d	f -> d	Z
0000 11df ffff	0Cf	Decrementa f	Decrement f	DECWF	f,d	f - 1 -> d	Z
0010 11df ffff	2Cf	Dec f, salta si cero	Decrement f, skip if zero	DECFSZ	f,d	f - 1 -> d, salta si cero	None
0010 10df ffff	28f	Incrementa f	Increment f	INCF	f,d	f + 1 -> d	Z
0011 11df ffff	3Cf	Inc f, salta si cero	Increment f, skip if zero	INCFSZ	f,d	f + 1 -> d, salta si cero	None
0001 00df ffff	10f	Or inclusiva entre Wyf	Inclusive OR W and f	IORWF	f,d	W v f -> d	Z
0010 00df ffff	20f	Mueve f	Move f	MOVF	f,d	f -> d	Z
0000 001f ffff	02f	Mueve W a f	Move W to f	MOVWF	f	W -> f	None
0000 0000 0000	000	No operación	No operation	NOP	-	-	None
0011 01df ffff	34f	Rota a la izquierda f	Rotate left f	RLF	f,d	f(n) -> d(n+1), C -> d(0), f(7) -> C	C
0011 00df ffff	30f	Rota a la derecha f	Rotate right f	RRF	f,d	f(n) -> d(n-1), C -> d(7), f(0) -> C	C
0000 10df ffff	08f	Resta W de f	Subtract W from f	SUBWF	f,d	f - W -> d (f+W+1 -> d)	C,DC,Z
0011 10df ffff	38f	Intercambia mitades de	Swap halves f	SWAPF	f,d	f(0-3) -> f(4-7) -> d	None
0001 10df ffff	18f	Or exclusiva entre W y f	Exclusive OR W and f	XORWF	f,d	W + f -> d	Z
OPERACIONES ORIENTADAS A BITS			11-8		(7-5)		(4-0)
			CÓDIGO DE OPERACIÓN		d	f (REGISTRO N)	
0100 bbbf ffff	4bf	Limpia bit f	Bit Clear f	BCF	f,b	0 -> f (b)	None
0101 bbbf ffff	5bf	Activa bit f	Bit Set f	BSF	f,b	1 -> f (b)	None
0110 bbbf ffff	6bf	Probar bit f, salta si cero	Bit Test f, skip if clear	BTFSC	f,b	prba bit (b) en reg. (0: salta si cero prba bit (b) en reg. (f): salta si uno	None
0111 bbbf ffff	7bf	Probar bit f, salta si uno	Bit Test f, skip if set	BTFSS	f,b	prba bit (b) en reg. (f): salta si uno	None
OPERACIONES ORIENTADAS A BITS			(11-8)		(7-0)		
			CÓDIGO DE OPERACIÓN		d	f (REGISTRO N)	
1110 kkkk kkkk	Ekk	And entre W y un literal	AND literal and W	ANDLW	k	k&W -> W	
1001 kkkk kkkk	9kk	Llama subrutina	Call routine	CALL	k	PC+1 -> Stach.k -> PC	
0000 0000 0100	004	Limpia Watchdog	Clear Watchdog timer	CLRWD	-	0 -> WDT (y prescalador)	
101k kkkk kkkk	Akk	Salta a dirección	Go To address (k is 9 b)	GOTO	k	k -> PC (9bits)	
1101 kkkk kkkk	Dkk	Or Inc. Entre W y literal	Incl. OR literal and W	IORLW	k	k v W -> W	
1100 kkkk kkkk	Ckk	Mueve literal a W	Move literal to W	MOVLW	k	k -> W	
0000 0000 0010	002	Carga registro OPTION	Load OPTION register	OPTION	-	W -> Registro OPTION	

1000 kkkk kkkk	8kk	Retorna, carga literal en	Return, place literal in V	RETLW	k	Ir + W, pila -> PC	
0000 0000 0011	003	Va modo de espera	Go into standby mode	SLEEP	-	0 -> WDT, detiene oscilador	
0000 0000 0fff	OOf	Configura puerto	Tristate port f	TRIS	f	W -> control I/O del registro f	
1111 kkkk kkkk	Fkk	OR excl. Entre literal y W	Excl.OR literal and W	XORLW	k	k W -> W	

**Tabla. 1.3. Conjunto de instrucciones**

### 1.3.4 Sistema de Adquisición de Datos.

La placa base para PIC 16F877 contiene Sistemas de adquisición de datos, almacenamiento, procesamiento y comunicaciones con aplicaciones de PC. Al disponer de reloj de tiempo real y memoria EEPROM, la placa base para PIC 16F877 puede ser usado como un sistema de adquisición de datos autónomo.

El reloj de tiempo real es el HT1380. Este RTC opera en base a un cristal de 32768 Hz y permite llevar la hora, minutos, segundos, día, mes y año actual. Para su uso se requiere de la librería ht1380.h

La memoria serie EEPROM del tipo 24LC16 de 16Kbits de capacidad (2Kbytes) que permite almacenar información de forma no volátil. Para mayor capacidad se puede reemplazar esta memoria por una de mayor capacidad como las 24LC256 de 256Kbits (32Kbytes). Para utilizar dicha memoria se requiere la inclusión de la librería correspondiente denominada 2416.h

La adquisición de datos será por las entradas analógicas del PIC y por el puerto de comunicaciones RS232.

Para el caso de entradas analógicas del PIC, se lee un canal analógico y el valor se almacena en la dirección **x0**, mediante el conversor A/D del PIC.

El conversor A/D del PIC es de 10 bits. Esto significa que entrega valores entre 0 y 1023. (Nota: 2 elevado a la 10 es 1024).

Cuando recibe 0V ..... **x0** vale 0

Cuando recibe 5V..... **x0** vale 1023

Como se puede ver no es posible utilizar una variable tipo **INT** (0 a 255) ya que no soporta valores de 0 a 1023. Es por ello que se debe usar una tipo **LONG** (0 a 65535).

Las Entradas Analógicas del PIC 16F877 son las siguientes:

Entrada Analógica 0. Conectado al RA0 y habilitado como AN0. Canal analógico 0.

Entrada Analógica 1. Conectado al RA1 y habilitado como AN1. Canal analógico 1.

Entrada Analógica 3. Conectado al RA3 y habilitado como AN3. Canal analógico 3.

Pin4: GND

En los microcontroladores PIC la configuración de los canales analógicos están configurados de tal forma que para habilitar solamente 3 canales analógicos se deben utilizar las entradas 0,1 y 3. Es por ello que el AN2 no se utiliza.

En el caso de adquisición de datos por el puerto RS232 se utiliza para capturar datos provenientes desde el puerto serie de la PC.

### **1.3.5 Transmisión de Información.**

La Placa Base para PIC 16F877 posee un puerto de comunicaciones RS232 accesible mediante un conector DB9 hembra. Se conecta al puerto serie de la PC mediante un cable DB9-DB9 macho-hembra pin a pin.

#### **1.3.5.1 Señalización.**

Detrás del conector DB9 se encuentran dos leds:

**TX** led rojo: Se activa cuando el PIC transmite datos al puerto RS232.

**RX** led verde: Se activa cuando el PIC recibe datos desde el puerto RS232.

CHIP MAX232: Debido a que el PIC opera con niveles de tensión TTL (0 Y 5v) y el puerto de comunicaciones RS232 con niveles +-12V se requiere de un chip que convierta dichos niveles. El MAX232 se encarga de esta función.

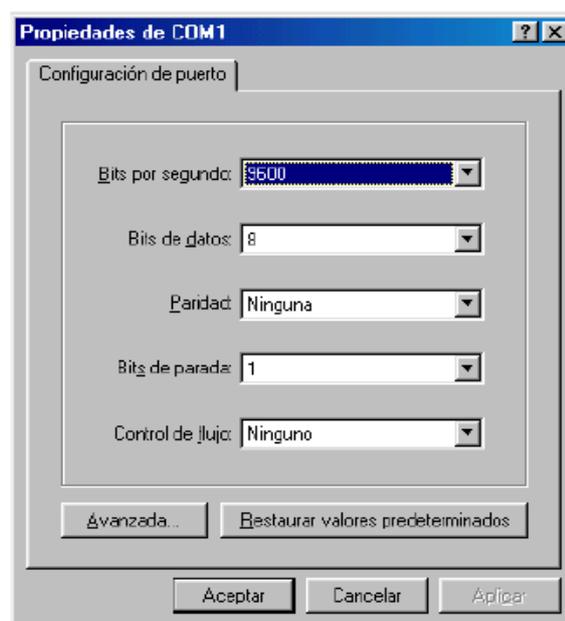
Para transmitir datos por este puerto, se debe primero definir en el encabezamiento del programa (incluido en la plantilla).

### **//Puerto Rs232**

```
#use rs232(baud=9600, xmit=PIN_C6,rcv=PIN_C7,errors,rs232)
```

Esta definición indica que el puerto se llamará **rs232**, que operará a una velocidad de transmisión de 9600 bps (bits por segundo), que los datos salen del pin RC6 (Tx) y entran por el pin RC7 (Rx) del PIC. Ver en el plano del PIC 16F877 que los pines RC6 y RC7 corresponden a dichas funciones.

Cuando se desea transmitir datos por el puerto serie se debe cargar en la PC un programa de terminal (como por ejemplo el Hyperterminal) y configurarlo para operar a la misma velocidad con la que se definió en el puerto RS232 en la PC, en la figura 1.9 se indica la ventana de propiedades configuración del puerto serie del PC.



**Figura 1.9** Propiedades Configuración Puerto Serie del PC

Transmisión de una cadena de caracteres (String): Indicado entre comillas dobles:

```
//Transmitir "Pepe" por el puerto rs232  
fputs("Pepe",rs232);
```

Transmisión de un carácter: Indicado entre comillas simples

```
//Transmitir 'A'  
fputc('A',rs232);
```

Para recibir datos por el puerto serie provenientes del programa de terminal se debe utilizar:

```
a=fgetc(rs232); //para recibir de a un carácter por vez
```

donde **a** es una variable tipo **char** previamente definida (Ej **char a;**)  
**fgets(trama,rs232); //para recibir un string (terminado con Enter)** donde **trama** es una variable tipo string previamente definida (Ej **char trama[20];**)

### 1.3.6 Otras Particularidades

Todos los componentes de la **gama alta** responden a la misma arquitectura básica del procesador interno y de la organización de memoria y disponen del mismo repertorio de instrucciones. Sus diferencias estriban en la capacidad de la memoria y la disponibilidad de diferentes periféricos. En general, las aportaciones realizadas en esta gama con respecto a lo anterior son las siguientes:

1. La longitud del código OP de las instrucciones pasa a ser de 14 bits frente a los 12 de los PIC16C5X. Este incremento permite manejar memorias de programa de 4k palabras de 14 bits y memorias de datos de hasta 192 bytes frente a los 32 que permitían los microcontroladores de la gama baja.
2. En la gama alta existe la posibilidad de atender interrupciones, tan necesarias en la aplicaciones en tiempo real. El vector de interrupción se halla ubicado en la dirección 0004h de la memoria de programa.
3. La Pila aumenta su profundidad hasta ocho niveles.

4. Se añaden cuatro nuevas instrucciones al repertorio de la gama baja: RETURN, RETFIE, ADDLW y SUBLW. Dos instrucciones, OPTION y TRIS, se modifican para mantener la compatibilidad.
5. Se ha redefinido la organización de la memoria de datos, que queda estructurada en bancos de 128 bytes cada uno.
6. Se modifica el registro de estado.
7. El vector de reset se cambia a la posición 0000h.
8. Varía la actuación del reset sobre todos los registros, admitiéndose hasta cinco tipos diferentes de reset.
9. Es posible salir del modo de Reposo (SLEEP) mediante interrupción.
10. Se añade el registro PCLATH para contener los bits de más peso del PC, permitiendo manejar la paginación de la memoria. Este cambio obliga a redefinir la misión de los bits PA2, PA1 y PA0 del Registro de estado.
11. Los registros OPTION y TRIS pueden ser direccionados.
12. Se incluyen dos temporizadores independientes, el que controla el “start up” y el de “power up”.
13. Las líneas del Puerto B pueden programarse para disponer de carga “pull up” interna si actúan como entradas y también pueden provocar interrupciones.
14. El terminal RTCC puede actuar en algunos modelos como una línea más del Puerto A (RA4/T0CKI).
15. El registro de FSR actúa completo.

16. Se hace posible la programación del chip dentro del sistema puesto que para esta operación solo se necesitan cinco pines:  $V_{DD}$ ,  $V_{SS}$ ,  $V_{PP}$ , RB6 (reloj) y RB7(Entrada o Salida del dato).

17. Surge un nuevo registro denominado PCON, que tiene dos bits para diferenciar si el reset se ha producido por un fallo en la alimentación (Brown-Out) o por conexión de la alimentación (Power-On-Reset).

18. Se ha mejorado el sistema de protección de código para actuar solo sobre las zonas concretas del programa que se seleccionen.

19. Para convertir el código utilizado en los microcontroladores de la gama baja en código ejecutable para la gama media, se deben tener en cuenta las siguientes comprobaciones:

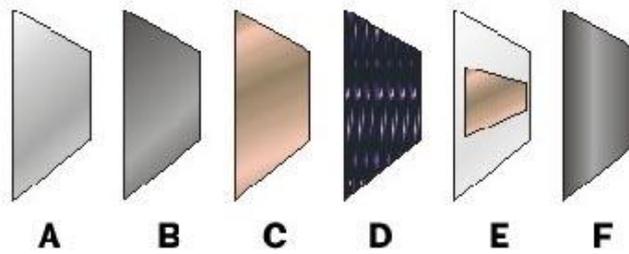
- Se deben modificar las operaciones que seleccionaban una página de la memoria en las instrucciones CALL y GOTO, revisando los valores de los bits PA2, PA1 y PA0.
- Revisar las operaciones de salto que escribían sobre el PC o le añadían un valor para asegurarse que se utilizan las páginas correctas.
- Eliminar los saltos entre los bancos de la memoria de datos.
- Verificar todas las escrituras sobre los Registros de estado, OPTION y FSR dada su nueva estructura.
- Cambiar el vector de reset a la posición 0000h.

## **1.4 Dispositivos Para la Visualización de Información**

### **1.4.1 Display's.**

Son dispositivo de ciertos aparatos electrónicos, como los teléfonos y las calculadoras, destinado a la representación visual de información.

En la figura 1.10 se observa un ejemplo de la estructura de un display muy básico; al activarse se visualizaría un rectángulo horizontal.



**Figura 1.10** Estructura básica de un display

Donde cada una es:

**A:** Espejo

**B:** Capa de vidrio con filtro polarizador vertical

**C:** Electrodo transparente (común)

**D:** Cristal líquido

**E:** Capa de vidrio con electrodo transparente ( en forma de rectángulo )

**F:** Filtro polarizador horizontal

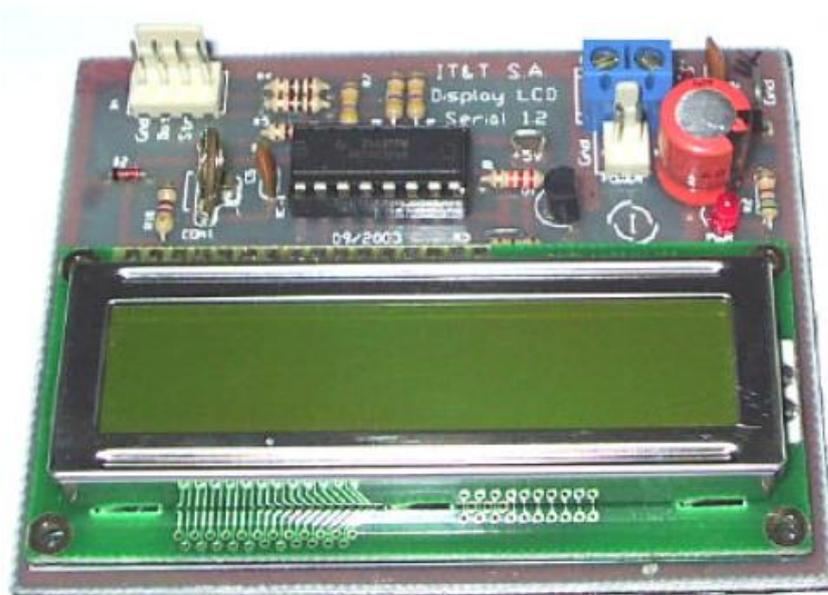
El mecanismo básico de funcionamiento es la luminiscencia de inyección, que produce radiación visible por un proceso de dos etapas; la inyección de portadores minoritarios y la recombinación radiactiva.

#### **1.4.2 LCD's.**

La figura 1.11 hace referencia a la interfaz LCD-Síncrono. Esta especialmente desarrollado para aplicaciones con microcontroladores PIC.

Esta basado en display's LCD estándares de 2x16 caracteres pero con la particularidad de permitir una rápida implementación y un muy bajo requerimiento de líneas de puerto del microcontrolador.

Normalmente para utilizar un LCD se requieren de 4 u 8 líneas de datos, 2 o 3 líneas de control y 1 línea más si se requiere activar la luz del display.



**Figura 1.11** Interfaz de display LCD sincrónico

Esto da un total de entre 6 y 12 líneas requeridas para operar un LCD con un microcontrolador. Además, es necesaria la implementación de rutinas especiales para su operación.

El LCD-Sinc soluciona todos estos aspectos de una forma muy eficiente. Por un lado, el LCD-Sinc puede ser operado con solo 3 líneas, de las cuales 2 de ellas no son necesariamente exclusivas del Interfaz. Además, las librerías en 'C' provistas con la interfase, lo hacen un dispositivo de uso inmediato sin requerir de programación alguna.

#### **1.4.2.1 Aplicaciones.**

Este interfaz permite al usuario diseñador de sistemas microprocesador disponer de forma inmediata de un dispositivo alfanumérico de visualización de información con un muy bajo requerimiento de hardware de parte del microcontrolador.

Normalmente en las aplicaciones de control industrial se requiere la implementación de sistemas operativos que permitan el interfaz hombre-máquina.

En estos casos el módulo de display LCD-Sinc cumple perfectamente bien esta función.

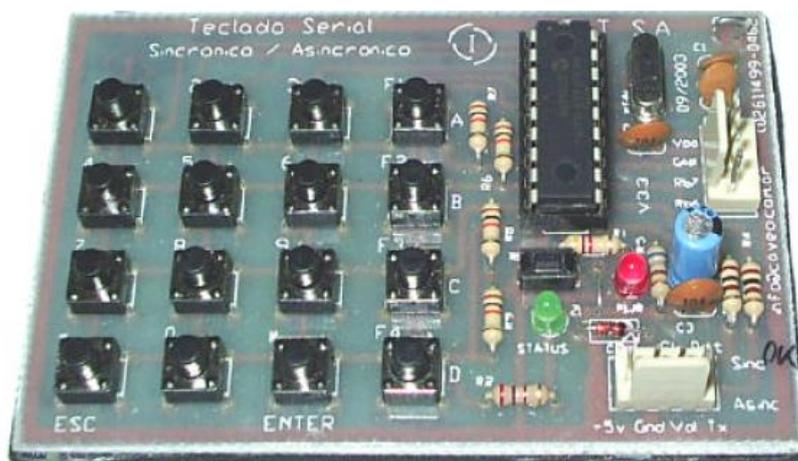
### 1.4.2.2 Puesta en servicio.

Para hacer funcionar el interfaz se debe disponer de la librería <sup>6</sup>lcdser.h la cual debe incluirse en la compilación del programa del usuario.

La configuración de la librería se realiza de forma muy sencilla. En su encabezamiento se definen los puertos que comandarán las líneas de CK, DATA y STROBE. De acuerdo con el hardware, deberá asignar a cada una de estas líneas de control el puerto del PIC en donde desee conectarla.

## 1.5 Dispositivos para el Ingreso de Información

### 1.5.1 Teclado numérico serie RS232-TTL



**Figura 1.12** Teclado numérico serie RS232-TTL

Como se observa en la figura 1.12, un teclado especialmente desarrollado para aplicaciones con microcontroladores PIC.

Esta constituido por una matriz de 16 pulsadores indicados con los números 0 a 9 mas 4 pulsadores de función F1, F2, F3 y F4 las cuales también tienen indicaciones de 4 flechas: arriba, abajo, izquierda y derecha, y además cuenta con una tecla de ESC o \* y otra de ENTER o # La lógica del interfaz esta comandada por un PIC16F84, el cual se encarga de todo el procesamiento del interfaz.

<sup>6</sup> Librerías de programación, Interfaz de display LCD sincrónico en el Anexo II

A diferencia de otros teclados que para conectarlos a un microcontrolador requieren del uso de muchas de sus líneas de puerto, el teclado numérico serie se conecta a un sistema microprocesado con solo una línea. A lo sumo puede llegar a requerir de dos.

La información del pulsador presionado es transmitida por el interfaz en formato RS232 con niveles TTL (0 y 5V) a 9600 bps.

#### **1.5.1.1 Aplicaciones.**

Este interfaz permite al usuario diseñador de sistemas microprocesador disponer de forma inmediata de un dispositivo de entrada de datos numéricos con un muy bajo requerimiento de hardware de parte del microcontrolador.

Normalmente en el diseño de aplicaciones de control industrial se requiere la implementación de sistemas operativos con un eficiente interfaz hombre-máquina. Para estos casos es necesario de por lo menos un display y un teclado de comando.

Este interfaz esta diseñado para ser utilizado en prototipos. Dependiendo de la aplicación en particular se puede solicitar un teclado más robusto, adecuado a ser instalado en ambiente industrial. De todas formas, a nivel de diseño del sistema, el uso de cualquiera de los modelos es totalmente transparente al programador.

#### **1.5.1.2 Características técnicas principales.**

- Alimentación 5V DC
- Protecciones contra inversión de polaridad
- Salida de datos a 9600 bps, 8 bits, sin paridad

### **1.6 Transductores para Cantidades Eléctricas.**

Un transductor es, simplemente, un dispositivo que proporciona una salida utilizable en respuesta a una magnitud física, propiedad o condición específica que se desea medir. Generalmente se trata de un dispositivo utilizado para convertir un fenómeno físico en una señal eléctrica.

Los transductores se conocen con nombres diferentes en las distintas disciplinas técnicas: sensores (de presión, de fuerza, de temperatura), detectores, galgas, captadores, sondas, etc.

En el caso de los transductores que proporcionan como salida una señal eléctrica, se pueden distinguir genéricamente las siguientes partes:

- sensor (o sonda), en contacto con la magnitud física.
- mecanismos auxiliares.
- captador, que proporciona una cierta señal eléctrica.
- preamplificador o acondicionador de la señal.

La descripción de un transductor se basa generalmente en la mayoría o todas de las consideraciones siguientes:

- 1) ¿Qué se intenta medir? (magnitud).
- 2) ¿Cuál es el principio operativo de la porción eléctrica del transductor con en el que se origina la salida? (principio de transducción).
- 3) ¿Qué elemento del transductor responde directamente a la magnitud que se mide? (elemento sensor).
- 4) ¿Qué prestaciones o ventajas especiales se incorporan en el transductor?.
- 5) ¿Cuáles son sus principales características?
  - rango de operación.
  - sensibilidad.
  - compatibilidad ambiental: temperatura, presiones, tamaño, restricciones de montaje.
  - robustez.
  - características eléctricas de la señal de salida.

A partir de la descripción anterior surge una primera clasificación de los transductores: según el método de transducción o según la variable que son capaces de medir. A nivel introductorio, parece más apropiada una clasificación atendiendo al primero de los criterios.

El grupo más importante de transductores es quizás el de parámetro variable, que se caracterizan por proporcionar una salida relacionada con la variación de un determinado parámetro eléctrico pasivo, originada a su vez por la variación del fenómeno físico que se desea medir. Son de gran robustez y simplicidad de construcción.

Por otra parte se puede definir los transductores llamados activos o autogeneradores, que son aquéllos que producen una señal eléctrica (sin necesidad de alimentación externa) cuando son estimulados por alguna forma física de energía: electromagnéticos, piezoeléctricos, fotovoltaicos, termoeléctricos. Aquéllos que necesitan de algún elemento adicional en la transducción, son los transductores pasivos.

## **1.7 Procesamiento de Datos por medio de LabVIEW**

### **1.7.1 Introducción**

LabVIEW es un revolucionario ambiente de desarrollo gráfico con funciones integradas para realizar adquisición de datos, control de instrumentos, análisis de mediciones y presentaciones de datos. LabVIEW permite flexibilidad de un poderoso ambiente de programación sin la complejidad de los ambientes tradicionales.

Es una herramienta ampliamente empleada por científicos e ingenieros de todo el mundo para el desarrollo de instrumentos virtuales (VI) ya que engloba todas las fases del desarrollo de VI bajo un único entorno, permitiendo crear soluciones multiplataforma de gran rendimiento y vistosidad.

Las ventajas que proporciona el empleo de LabVIEW se resumen en las siguientes:

- Se reduce el tiempo de desarrollo de las aplicaciones al menos de 4 a 10 veces, ya que es muy intuitivo y fácil de aprender.
- Dota de gran flexibilidad al sistema, permitiendo cambios y actualizaciones tanto del hardware como del software.

- Da la posibilidad a los usuarios de crear soluciones completas y complejas.
- Con un único sistema de desarrollo se integran las funciones de adquisición, análisis y presentación de datos.
- El sistema está dotado de un compilador gráfico para lograr la máxima velocidad de ejecución posible.
- Tiene la posibilidad de incorporar aplicaciones escritas en otros lenguajes.

LabVIEW es un entorno de programación destinado al desarrollo de aplicaciones, similar a los sistemas de desarrollo comerciales que utilizan el *lenguaje C* o *BASIC*. Sin embargo, LabVIEW se diferencia de dichos programas en un importante aspecto: los citados lenguajes de programación se basan en líneas de texto para crear el código fuente del programa, mientras que LabVIEW emplea la programación gráfica o *lenguaje G* para crear programas basados en diagramas de bloques.

Para el empleo de LabVIEW no se requiere gran experiencia en programación, ya que se emplean iconos, términos e ideas familiares a científicos e ingenieros, y se apoya sobre símbolos gráficos en lugar de lenguaje escrito para construir las aplicaciones. Por ello resulta mucho más intuitivo que el resto de lenguajes de programación convencionales.

LabVIEW posee extensas librerías de funciones y subrutinas. Además de las funciones básicas de todo lenguaje de programación, LabVIEW incluye librerías específicas para la adquisición de datos, control de instrumentación VXI, GPIB y comunicación serie, análisis presentación y guardado de datos.

LabVIEW también proporciona potentes herramientas que facilitan la depuración de los programas.

Los programas desarrollados mediante LabVIEW se denominan *Instrumentos Virtuales (VIs)*, porque su apariencia y funcionamiento imitan los de un instrumento real. Sin embargo son análogos a las funciones creadas con los

lenguajes de programación convencionales. Los VIs tienen una parte interactiva con el usuario y otra parte de código fuente, y aceptan parámetros procedentes de otros VIs.

Todos los VIs tienen un *panel frontal* y un *diagrama de bloques*. Las *paletas* contienen las opciones que se emplean para crear y modificar los VIs. A continuación se procederá a realizar una somera descripción de estos conceptos.

### 1.7.1.1 Panel Frontal.

Se trata de la interfaz gráfica del VI con el usuario. Esta interfaz recoge las entradas procedentes del usuario y representa las salidas proporcionadas por el programa. Un *panel frontal* está formado por una serie de botones, pulsadores, potenciómetros, gráficos, etc.

Cada uno de ellos puede estar definido como un *control* (a) o un *indicador* (b). Los primeros sirven para introducir parámetros al VI, mientras que los indicadores se emplean para mostrar los resultados producidos, ya sean datos adquiridos o resultados de alguna operación, en la figura 1.13 se observa un ejemplo de panel frontal.

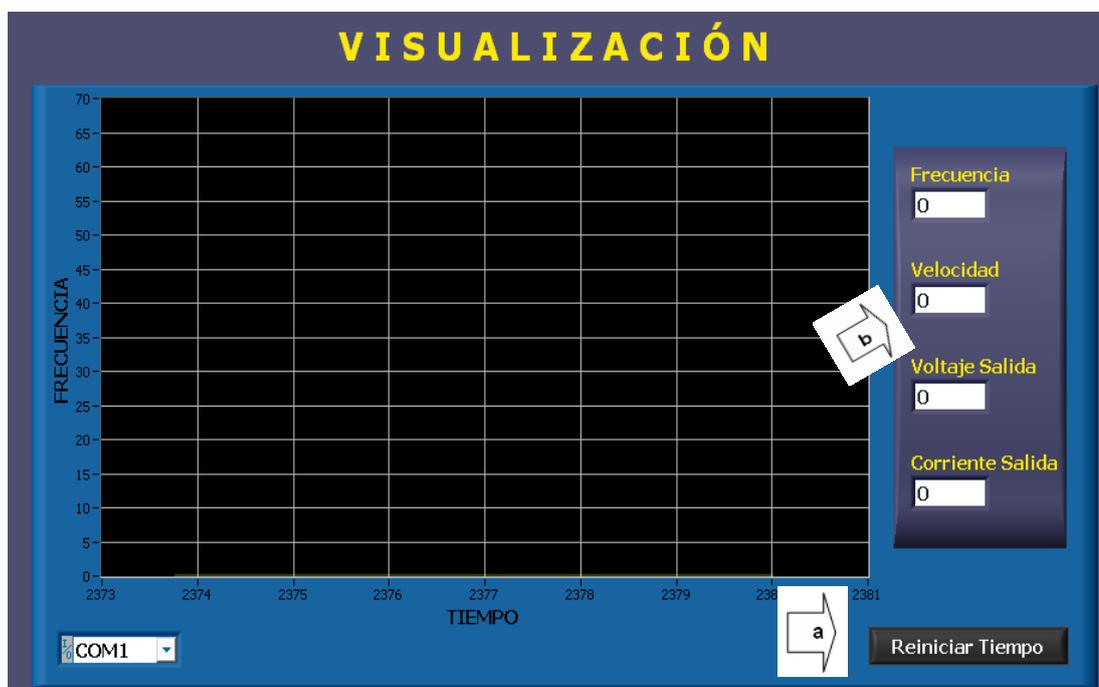


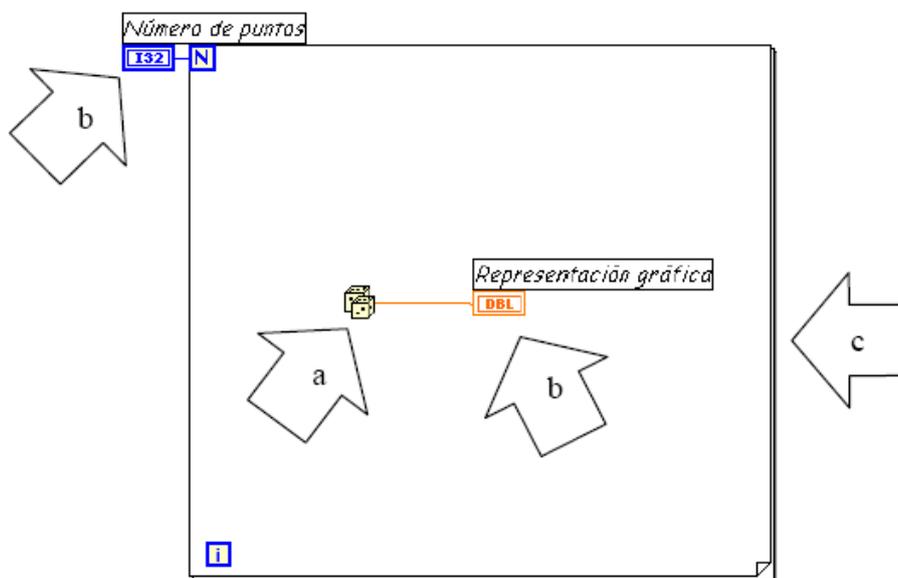
Figura 1.13 Ejemplo de Panel Frontal

### 1.7.1.2 Diagrama de bloques

El *diagrama de bloques* constituye el código fuente del VI. En el *diagrama de bloques* es donde se realiza la implementación del programa del VI para controlar o realizar cualquier procesado de las entradas y salidas que se crearon en el *panel frontal*.

El *diagrama de bloques* incluye *funciones* y *estructuras* integradas en las librerías que incorpora LabVIEW. En el *lenguaje G* las *funciones* y las *estructuras* son nodos elementales. Son análogas a los operadores o librerías de funciones de los lenguajes convencionales, en la figura 1.13 se observa un ejemplo de un diagrama de bloques.

Los *controles* e *indicadores* que se colocaron previamente en el Panel Frontal, se materializan en el diagrama de bloques mediante los terminales. En la figura 1.14 se presenta un ejemplo de lo recién citado:



**Figura 1.14** Ejemplo Diagrama de Bloques

- (a) Función.
- (b) Terminales (control e indicador).
- (c) Estructura.

El *diagrama de bloques* se construye conectando los distintos objetos entre sí, como si de un circuito se tratara. Los cables unen terminales de entrada y salida con los objetos correspondientes, y por ellos fluyen los datos.

LabVIEW posee una extensa biblioteca de *funciones*, entre ellas, aritméticas, comparaciones, conversiones, funciones de entrada/salida, de análisis, etc.

Las *estructuras*, similares a las declaraciones causales y a los bucles en lenguajes convencionales, ejecutan el código que contienen de forma condicional o repetitiva (bucle *for*, *while*, *case*,...).

Los cables son las trayectorias que siguen los datos desde su origen hasta su destino, ya sea una función, una estructura, un terminal, etc. Cada cable tiene un color o un estilo diferente, lo que diferencia unos tipos de datos de otros.

### 1.7.1.3 Paletas.

Las *paletas* de LabVIEW proporcionan las herramientas que se requieren para crear y modificar tanto el *panel frontal* como el *diagrama de bloques*. Existen las siguientes paletas:

#### 1.7.1.4 Paleta de herramientas (Tools palette)

En la figura 1.15 se observa la paleta de herramientas, esta se emplea tanto en el *panel frontal* como en el *diagrama de bloques*. Contiene las herramientas necesarias para editar y depurar los objetos tanto del *panel frontal* como del *diagrama de bloques*.



Figura 1.15 Paleta de Herramientas

#### 1.7.1.5 Paleta de controles (Controls palette)

En la figura 1.16 se observa la paleta de controles, esta se utiliza únicamente en el *panel frontal*. Contiene todos los *controles* e *indicadores* que se emplearán para crear la interfaz del *VI* con el usuario.



**Figura 1.16** Paleta de Controles

#### 1.7.1.6 Paleta de funciones (functions palette)

En la figura 1.17 se observa la paleta de de funciones, esta se emplea en el diseño del *diagrama de bloques*. La *paleta de funciones* contiene todos los objetos que se emplean en la implementación del programa del *VI*, ya sean *funciones* aritméticas, de entrada/salida de señales, entrada/salida de datos a fichero, adquisición de señales, temporizar la ejecución del programa, entre otras.



**Figura 1.17** Paleta de Funciones

### 1.7.2 LabVIEW como Herramienta de Adquisición de Datos.

El objetivo es tener una herramienta para la adquisición de datos, que permita realizar un muestreo de un voltaje variable en el tiempo a intervalos de tiempo iguales y con la mayor precisión posible.

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.

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, una FPGA 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, utilizando la herramienta LabWindowsCVI de National Instruments. Con este software se pueden diseñar en C las DLL para el funcionamiento de las tarjetas, estas DLL se pueden integrar a LabVIEW y a todas su funcionalidad o creadas desde otros programas clásicos como Visual Basic o Borland C.

### **1.7.3 Adquisición de Datos por los Puertos Serie y Paralelo.**

#### **1.7.3.1 Manejo del Puerto Serie.**

Las comunicaciones serie se utilizan para enviar datos a través de largas distancias, ya que las comunicaciones en paralelo exigen demasiado cableado para ser operativas. Los datos serie recibidos desde un módem u otros dispositivos son convertidos a paralelo gracias a lo cual pueden ser manejados por el bus del PC.

Los equipos de comunicaciones serie se pueden dividir entre simplex, half-duplex y full-duplex. Una comunicación serie simplex envía información en una sola dirección (p.e. una emisora de radio comercial). Half-duplex significa que los datos pueden ser enviados en ambas direcciones entre dos sistemas, pero en una sola dirección al mismo tiempo.

En una transmisión full-duplex cada sistema puede enviar y recibir datos al mismo tiempo.

El puerto serie del PC es compatible con el estándar RS-232C, los PC modernos utilizan los DB-9 macho. Se puede encontrar uno o más de estos conectores en el panel posterior del PC. Los voltajes para un nivel lógico alto están entre -3V y -15V. Un nivel lógico bajo tendrá un voltaje entre +3V y +15V. Los voltajes más usados son +12V y -12V.

Los parámetros que caracterizan estas comunicaciones son: Velocidad; paridad; bits de datos y bits de parada. En la literatura sobre el tema es frecuente expresar estos datos en forma resumida. Por ejemplo: 1200 8 N 1, para

indicar una transmisión de 1200 baudios con 8 bits de datos sin paridad y un bit de Stop.

Los parámetros anteriores están relacionados con la forma en que se transmite la información serie. En esta comunicación cada carácter va incluido en un cuadro ("Frame"); generalmente el comienzo es un bit de inicio (siempre un 1); después le sigue el dato, que puede ser de 5 a 8 bits de longitud; después puede haber un bit de control de paridad, y por último un bit final (siempre es un 1) de longitud variable (el equivalente a 1, 1.5 o 2 bits).

Las señales más utilizadas se listan a continuación:

/DTR (Data-Terminal-Ready): El PC indica al módem que esta encendido y listo para enviar datos.

/DSR (Data-Set-Ready): El módem indica al PC que esta encendido y listo para transmitir o recibir datos.

/RTS (Request-To-Send): El PC pone esta señal a 1 cuando tiene un carácter listo para ser enviado.

/CD (Carrier-Detect): El módem pone esta señal a 1 cuando ha detectado el ordenador.

/CTS (Clear-To-Send): El módem esta preparado para transmitir datos. El ordenador empezara a enviar datos al módem.

TxD: El módem recibe datos desde el PC.

RxD: El módem transmite datos al PC.

#### **1.7.3.1.1 Direccionamiento del Puerto Serie.**

Hay dos maneras de direccionar el puerto serie, a través de la interrupción 14H de la BIOS y a través de la interrupción 21H del DOS.

La interrupción 14H de la BIOS utiliza cuatro funciones para programar el puerto serie. Cada función es seleccionada asignando un valor al registro AH del microprocesador. Las cuatro funciones son listadas a continuación:

**Función 00H:** Inicializa el puerto serie y selecciona la velocidad, el número de bits de datos de inicio y de parada y los parámetros de paridad.

**Función 01H:** Envía un carácter al puerto serie especificado.

**Función 02H:** Lee un carácter desde el puerto serie especificado.

**Función 003:** Devuelve el estado del puerto serie especificado.

Hay varias funciones de la interrupción 21H del DOS relacionadas a la operación del puerto serie:

**Función 03H:** Lee un carácter desde el puerto COM1.

**Función 04H:** Escribe un carácter desde el puerto COM1.

**Función 40H:** Esta función envía un número de bytes desde un buffer a un dispositivo especificado.

### 1.7.3.2 Manejo del puerto paralelo.

En las aplicaciones que involucran el computador y la adquisición de datos es necesario emplear alguno de los puertos del computador, o bien realizar la aplicación enfocada a trabajar con una tarjeta de expansión, ya sea de estándar ISA (casi desaparecido) o bien con tecnología PCI.

Las dos últimas opciones, son completamente posibles pero no se consideran adecuadas puesto que son tecnologías cambiantes, que obligan a una renovación del hardware de la aplicación, según sea el caso.

En general el puerto paralelo de cualquier computador personal puede trabajar en varios modos dependiendo de aspectos como si se trata de entrada y salida de datos, si se requieren el uso de la memoria del puerto, etc. El modo más empleado, que además lo poseen todos los computadores sin importar que tan viejo sea, es el modo estándar o SPP (Standard Parallel Port).

El modo SPP está diseñado empleando un enfoque en el que el computador envía datos a través del puerto hacia una impresora y recibe únicamente algunas señales de control, por lo que el puerto se maneja empleando tres registros: el de datos, que funciona como salida de datos; el

registro de control, que posee función mixta de entrada y salida; y el de estado que funciona como entrada.

En la tabla 1.4 se aprecia la distribución de registros con sus correspondientes pines en el puerto paralelo.

Pin No (Tipo D-25)	Pin No (Centronics)	Señal SPP	Dirección Ent/Sal	Registro	Hardware Invertido
1	1	nStrobe	Ent/Sal	Control	Si
2	2	Dato 0	Salida	Dato	
3	3	Dato 1	Salida	Dato	
4	4	Dato 2	Salida	Dato	
5	5	Dato 3	Salida	Dato	
6	6	Dato 4	Salida	Dato	
7	7	Dato 5	Salida	Dato	
8	8	Dato 6	Salida	Dato	
9	9	Dato 7	Salida	Dato	
10	10	nAck	Entrada	Estado	
11	11	Busy	Entrada	Estado	Si
12	12	Salida Papel Fin Papel	Entrada	Estado	
13	13	Selección	Entrada	Estado	
14	14	nAuto-Pie linea	Ent/Sal	Control	Si
15	32	nError/nFalta	In	Estado	
16	31	nInicializar	Ent/Sal	Control	
17	36	nSelec-Impres nSelec-In	Ent/Sal	Control	Si
18 - 25	19 - 30	Tierra	Gnd		

**Tabla. 1.4.** Distribución de pines del puerto paralelo, se indica a que registro pertenece cada bit.

Como ya se mencionó, estos pines están agrupados en tres registros y se encuentran mapeados como tal en el computador. La ubicación de estos registros varía de acuerdo a tres direcciones base usadas para direccionarlos como se indica en la tabla 1.5.

Dirección Reg. Datos	Dirección Reg. Status	Dirección Reg. Control
Base	Base + 1	Base + 2
3BCh	3BDh	3BEh
378h	379h	37Ah
278h	279h	27Ah

**Tabla. 1.5. Direcciones Base para la ubicación de los Registros**

#### 1.7.4 Software de Visualización y Monitoreo

Como se indicó anteriormente, National Instruments LabVIEW es la plataforma sobre la cual se desarrolló el software de esta aplicación. LabVIEW es un software de programación gráfica, obviamente enfocada al manejo de objetos, y desarrollado pensando en sistemas de instrumentación y control, lo que lo hace de gran utilidad para la adquisición y análisis de los datos.

Un desarrollo en LabVIEW se basa en el empleo de los “VI” o instrumentos virtuales, que en un sentido práctico consiste en un objeto con entradas y salidas, y una función especificada dentro de este. La estructura de este lenguaje de programación denominado G permite tener o llamar un VI dentro de otro, es decir permite anidar VI's.

La creación de un VI dentro de LabVIEW supone dos partes: el diseño del panel frontal, en donde se encuentran las entradas y salidas del VI, y el diagrama de flujo de datos, en donde se configura el funcionamiento del VI; en un sentido práctico es el algoritmo que define el comportamiento de la aplicación.

Para acceder a la paleta de controles dentro del panel de control basta con oprimir el botón derecho del ratón teniendo el puntero sobre la ventana correspondiente a éste, de forma similar se logra el acceso a la paleta de funciones dentro del diagrama de flujo de datos (llamado simplemente diagrama).

LabVIEW permite manipular estructuras de datos, como datos boléanos (binarios), datos numéricos, cadenas, arreglos y clusters. Los clusters son una poderosa estructura que permite realizar arreglos compuestos por distintos tipos

de datos. Dentro del diagrama la estructura del dato se distingue por el color que toma el lazo correspondiente, por ejemplo, los boléanos se distinguen por el color verde.

En el ambiente de programación G (LabVIEW) el puntero del ratón cumple diversas funciones, distinguiéndose cada una por la figura que toma el puntero, por ejemplo si el cursor tiene forma de carrete de hilo funciona como herramienta de conexión. Para cambiar la función del puntero se emplea la tecla TAB.

## CAPITULO II

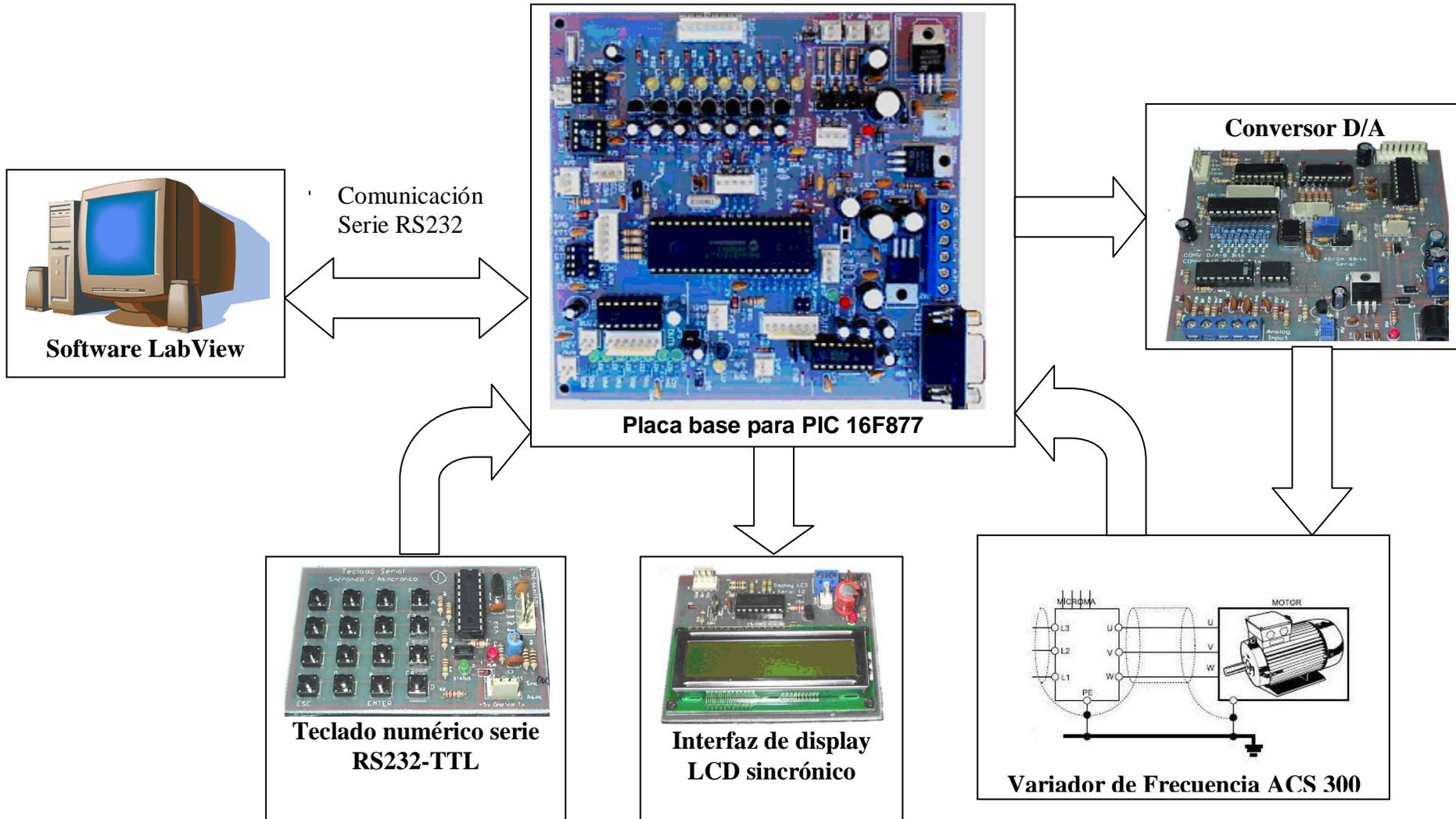
### ANÁLISIS Y DISEÑO

#### 2.1 Diagrama de Bloques del Proyecto

En la figura 2.1 se observa un diagrama de bloques del proyecto, en el cuál hay varias placas electrónicas y la que más sobresale corresponde a la que se le denomina PLACA BASE para el microcontrolador PIC 16F877, la cuál posee un Sistema de adquisición de datos, almacenamiento, procesamiento y comunicaciones con aplicaciones de PC. Al disponer de reloj de tiempo real y memoria EEPROM, este puede ser usado como un sistema de adquisición de datos autónomo. Además posee un teclado numérico serie RS232-TTL permitiendo el ingreso de información a la placa base y por lo tanto generar un mando local que corresponde a una fase del proyecto; se monitorea el estado del variador de frecuencia y el ingreso de datos a la placa base a través de un LCD – Serial estándar 2x16 caracteres, que esta desarrollado para aplicaciones con microcontroladores PIC al igual que el teclado numérico.

Para el acondicionamiento de señales se utilizó un conversor digital–análogo que se encarga de envíar una señal de voltaje de 0 a 5 voltios la cuál es amplificada por un factor de 2, correspondiente a la frecuencia ingresada por el operador, que es la frecuencia a la cual se desea que trabaje el variador de frecuencia, y una placa para acoplar las salidas digitales del microcontrolador PIC con un respectivo acondicionamiento para habilitar el estado del variador de frecuencia como: inicio, frenado, inversión de giro o que trabaje a frecuencias constantes ya establecidas.

Lo ya expuesto en su mayoría corresponde a la parte del mando local del variador de frecuencia; la comunicación vía RS-232 entre la placa base del microcontrolador PIC 16F877 y la PC, utilizando la Plataforma de Desarrollo LabVIEW permite un mando remoto del variador de frecuencia en procesos en los



**Figura 2.1 Diagrama Bloques del Proyecto**

cuales el ambiente en el que se desarrolla puede ser inaccesible ya sea por la presencia de agentes como el ruido, gases, ácidos, etc, permitiendo al operador acceder de una forma sencilla el mando.

Para la programación el microcontrolador PIC se utilizó LENGUAJE C dando mayor facilidad para cumplir con el objetivo del proyecto. estableciendo que para la parte de mando remota se realiza a través de la plataforma de desarrollo LabVIEW mediante comunicación , cabe aclarar que las dos formas de mando utilizan la placa base para cumplir con su fin.

## **2.3 Descripción de Cada Etapa del Diagrama**

### **2.3.1 <sup>7</sup>Placa Base para PIC 16F877**

#### **2.3.1.1 Descripción.**

La Placa Base es un controlador programable basado en un PIC de 40 pines. Su arquitectura permite operar con el PIC 16F877.

Se lo utiliza normalmente en aplicaciones de control industrial de alcance limitado donde se requiere utilizar tecnología de bajo costo.

La programación del equipo se realiza íntegramente en lenguaje 'C', para lo cual se dispone de una conjunto de librerías para su uso inmediato.

Para el caso de aplicaciones en donde se requiera un filtrado especial de la alimentación, se dispone de una fuente de alimentación adicional con entrada de AC/DC con filtrado especial para eliminación de ruidos eléctricos industriales.

#### **2.2.1.2 Aplicaciones.**

- Para lectura de señales analógicas. Sensores de temperatura, humedad, presión, nivel, infrarrojos etc. Las entradas de

---

<sup>7</sup> Plantillas de Programación, Placa Base para PIC 16F877 en el Anexo III

tensión/corriente permiten conectarle al PLC todo tipo de sensores comerciales estándares.

- Lectura de sensores con salida digital como: caudalímetros, encoders etc.
- Dataloggers. Sistemas de adquisición de datos, almacenamiento, procesamiento y
- Comunicaciones con aplicaciones de PC. Al disponer de reloj de tiempo real y memoria EEPROM, la placa base utilizada puede ser usado como un sistema de adquisición de datos autónomo.
- Control automático de procesos. Utilizando un display LCD o de 7 segmentos y un teclado de comando se pueden realizar diferentes interfaces Hombre-Máquina.
- Operación de la Placa Base mediante software de SCADA comerciales utilizando protocolo de comunicación Modbus.
- Conexión de interfaz de comunicaciones RS422 para vinculación en ambientes industriales punto a punto con otros equipos a 4 hilos
- Las comunicaciones RS232 y RS485 incluidas en el interfaz permiten vincular la Placa Base con una PC o en red con otros controladores utilizando el protocolo físico RS485 estándar, especialmente diseñado para comunicaciones en ambientes industriales.

## **2.2.2 <sup>8</sup>Teclado numérico serie RS232-TTL.**

### **2.2.2.1 Descripción.**

Esta especialmente desarrollado para aplicaciones con microcontroladores PIC.

Esta constituido por una matriz de 16 pulsadores indicados con los números 0 a 9 mas 4 pulsadores de función F1, F2, F3 y F4 las cuales también tienen indicaciones de 4 flechas: arriba, abajo, izquierda y derecha, y además cuenta con una tecla de ESC o \* y otra de ENTER o # La lógica del interfaz

---

<sup>8</sup> Librerías para la Programación de Interfases, Teclado numérico serie RS232-TTL en el Anexo II

esta comandada por un PIC16F84, el cual se encarga de todo el procesamiento del interfaz.

A diferencia de otros teclados que para conectarlos a un microcontrolador requieren del uso de muchas de sus líneas de puerto, el teclado numérico serie se conecta a un sistema microprocesado con solo una línea. A lo sumo puede llegar a requerir de dos.

La información del pulsador presionado es transmitida por el interfaz en formato RS232 con niveles TTL (0 y 5V) a 9600 bps, que se define en el software:

`rs232(baud=9600, xmit=PIN_xx ,rcv=PIN_a4);` que se explica más adelante.

#### **2.2.2.2 Aplicaciones.**

Este interfaz permite al usuario diseñador de sistemas microprocesador disponer de forma inmediata de un dispositivo de entrada de datos numéricos con un muy bajo requerimiento de hardware de parte del micro.

Normalmente en el diseño de aplicaciones de control industrial se requiere la implementación de sistemas operativos con un eficiente interfaz hombre-máquina. Para estos casos es necesario de por lo menos un display y un teclado de comando.

Este interfaz esta diseñado para ser utilizado en prototipos. Dependiendo de la aplicación en particular se puede solicitar un teclado más robusto, adecuado a ser instalado en ambiente industrial. De todas formas, a nivel de diseño del sistema, el uso de cualquiera de los modelos es totalmente transparente al programador.

#### **2.2.2.3 Características técnicas principales.**

- Alimentación 5V DC
- Protecciones contra inversión de polaridad
- Salida de datos a 9600 bps, 8 bits, sin paridad

#### 2.2.2.4 Conexión de la Placa Base al Teclado numérico serie RS232-TTL.

La Placa Base alimenta al teclado con 5V y gnd. El teclado le envía al CPU los datos serie en ASCII del switch presionado por el pin de TX.

Opcionalmente puede hacer falta un cable más para Vd (valid data)

#### 2.2.2.5 Definición del Puerto Serie en el Software.

Los siguientes comandos permiten definir el puerto serial:

**#use rs232(baud=9600, xmit=PIN\_xx ,rcv=PIN\_a4)**

**xmit=PIN\_xx** : A definir. No interesa para leer por el teclado.

**rcv=PIN\_a4**: Con esta definición, le indico al compilador que los pulsos que ingresen por el pin RA4 sean tomados como datos serie a 9600 bps.

#### 2.2.2.6 Configuración de los puertos del microcontrolador.

Para que el microcontrolador pueda recibir datos por la línea RA4, es necesario que esta línea este definida como entrada. Se debe recordar que con un 1 se indica entrada y con un 0 salida.

**set\_tris\_a (0b00010011)**; Esta instrucción define el sentido de los puertos desde el A7 al A0 (izquierda a derecha). El bit=1 indicado en negrita corresponde al RA4.

*Nota: 0b indica que lo que viene después es un número binario.*

#### 2.2.2.7 Captura de datos del teclado.

Para recibir datos por el teclado se utilizan las siguientes instrucciones:

**kbhit()** Vale=1 si esta entrando un dato por el puerto definido como entrada RS232

Lectura de un carácter proveniente del teclado:

**a=fgetc(teclado)**; // a variable definida como char

Lectura de un string proveniente del teclado (finalizado con enter)

```
fgets(trama,teclado); // trama, string definido como char trama[20]  
(Ej.20  
Caracteres)
```

## 2.2.3 <sup>9</sup>Interfaz de display LCD sincrónico.

### 2.2.3.1 Descripción.

El interfaz LCD-Sincrono. Esta especialmente desarrollado para aplicaciones con microcontroladores PIC. Permitiendo visualizar de una manera sencilla las variables y constantes para en control del variador de frecuencia, o a su vez los valores de frecuencia y tiempos que el usuario desee que trabaje éste, esto se realiza a través de la Placa Base para PIC 16F877.

Se debe mencionar que para el uso de esta interfaz LCD-Sincrono debe estar configurado de acuerdo como se especifica en su respectiva librería y plantilla de programación.

### 2.2.3.2 Configuración de la librería lcdser.h.

El interfaz de display LCD sincrónico opera con 3 líneas de control: CK, DATA y STROBE. Cada una de estas líneas se debe conectar a un puerto del PIC.

Si por ejemplo se coloca la línea de control CLOCK a RB2, la línea DATA a RB4 y la STROBE a RB3, la librería lcdser.h se deberá configurar de la siguiente forma:

```
#define ck0 output_low(PIN_b2)  
#define ck1 output_high(PIN_b2)  
#define data0 output_low(PIN_b4)  
#define data1 output_high(PIN_b4)  
#define strobe1_off output_low(PIN_b3)  
#define strobe1_on output_high(PIN_b3)
```

---

<sup>9</sup>Librerías para la Programación de Interfases, Interfaz de display LCD sincrónico en el anexo II

### 2.2.3.3 Arranque del display y ejemplo de uso.

Para utilizar el display LCD se debe cargar la librería correspondiente en el encabezamiento del código:

```
#include <lcdser.h>
```

Antes de utilizar el LCD se debe primero inicializar.

Esto se realiza en el main() antes del ciclo de SCAN.

```
Lcd_ini();
```

Luego se pueden utilizar las siguientes funciones (descriptas en lcdser.h)

```
//Encender luz de display
```

```
lcd_luz_on();
```

```
//Apagar luz de display
```

```
lcd_luz_off();
```

```
//Enviar un string al LCD (a partir de la última posición impresa)
```

```
lcd_putc("Hola");
```

```
//Borrar display LCD
```

```
lcd_clear();
```

```
//Ubicar el cursor en una posición determinada Ej.
```

```
lcd_linea1(5); //posiciona cursor en linea1 columna5
```

```
lcd_linea2(10); //posiciona cursor en linea2 columna10
```

```
//Encender u apagar cursor
```

```
lcd_cursor_on();
```

```
lcd_cursor_off();
```

```
//Enviar variables al display
```

```
lcd_clear();
```

```
printf(lcd_putc,"Lect= %4lu",x);
```

Por ejemplo, envía al LCD el texto "Lect= " y luego coloca el valor de la variable x que es un entero largo sin signo dejando 4 lugares en total.

Si por ejemplo x=23 muestra por el display

```
Lect= 23
```

El LCD se puede conectar y desconectar en cualquier momento no siendo necesario desconectar de la placa previamente.

#### **2.2.3.3.1 Conector para control del interfaz**

Se define los terminales de los puertos que comandarán las líneas de **CK, DATA y STROBE** que permite funcionar el interfaz con la Placa Base para el PIC 16F877.

De acuerdo con el hardware de cada usuario, deberá asignar a cada una de estas líneas de control el puerto del PIC en donde desee conectarla.

#### **2.2.3.3.2 Conector para alimentación:**

Para alimentar el LCD, utiliza un conector con voltaje constante de 5Vcc y GND.

### **2.2.4 Variador de Frecuencia ABB ACS 300**

Los variadores de frecuencia de corriente alterna (AC) se utilizan para controlar la velocidad y el torque de un motor eléctrico de inducción estándar. La tecnología de accionamientos en AC amplía el rango de velocidades del motor de cero a muy por encima de la velocidad nominal, lo que incrementa la productividad del proceso accionado.

Cuando se requiere una capacidad reducida del accionamiento, el variador de frecuencia disminuye la velocidad del motor consiguiendo un ahorro importante de energía.

ABB es líder en el mercado mundial de variadores de frecuencia, es el creador de la tecnología DTC (Control de Torque directo) la cual incorporan sus últimas series de variadores de frecuencia, permitiendo una mayor precisión de la velocidad y un control más rápido del torque en los motores de inducción AC.

#### **2.2.4.1 Conexión a la red.**

Rango de tensión y potencia:

Trifásica, 380 a 480 V, +10/-15%, 0,75 - 355 kW

Trifásica, 200 a 240 V, +10/-15%, 0,75 - 75 kW

Autoidentificación de la red de alimentación

Frecuencia: 48 a 63 Hz

Factor de potencia: 0,98

#### **2.2.4.2 Conexión a motor.**

Tensión Trifásica, de 0 a V ALIMENTACIÓN

Frecuencia 0 a 500 Hz

Frecuencia de conmutación:

Estándar de fábrica 4 kHz

Seleccionable 0,75 - 90 kW 1 kHz, 4 kHz, 8 kHz hasta 355  
kW 1 kHz, 4 kHz

Tiempo de aceleración: 0,1 a 1800 s

Tiempo de deceleración: 0,1 a 1800 s

Límites ambientales:

Temperatura ambiente: -15 a 40 o C

No se permite escarcha: 40 a 50 o C

f conmutación. 4 kHz, para derrateo consulte al proveedor

Humedad relativa Inferior al 95% (sin condensación)

Clase de protección IP 21 o IP 54

Niveles de contaminación IEC 721-3-3

No se permite polvo conductor.

#### **2.2.4.3 Conexiones de control programables.**

Dos entradas analógicas:

Señal de tensión 0 (2) a 10 V,  $R_{in} > 312\Omega$  terminación única

Señal de intensidad: 0 (4) a 20 mA,  $R_{in} = 100\Omega$  terminación única

Valor de ref. de potenciómetro 10 V  $\pm 2\%$  máx. 10 mA,  $R < 10\text{ k}\Omega$

Demora máxima 12...32 ms

Resolución 0,1%

Precisión  $\pm 1\%$

Dos salidas analógicas 0 (4) a 20 mA, carga  $< 500 \Omega$

Tensión auxiliar 24 V CC  $\pm 10\%$ , máx. 250 mA

Seis entradas digitales 12 V... 24 V CC con alimentación interna o externa, PNP y NPN

Impedancia de entrada 2,4 k  $\Omega$

Demora máxima 5 ms  $\pm 1$  ms

Tres salidas de relé

Tensión de conmutación máx. 250 V CA/30 V CC

Intens. de conmutación máx. 6 A/30 V CC; 1500 V A/230 V CA

Intensidad continua máxima 2 A eficaces

Límites de protección

Límites de disparo por sobretensión

V CC de marcha 842 (corr. a entrada de 595 V)

V CC de inhibición de marcha 661 (corr. a entrada de 380 - 415 V),  
765 (corr. a entrada de 440 a 480 V)

Límites de disparo por subtensión

V CC de marcha 333 (corr. a entrada de 247 V)

V CC de inhibición de marcha 436 (corr. a entrada de 380 - 415 V),  
505 (corr. a entrada de 440 - 480 V)

Cumplimiento del producto

Sistema de garantía de calidad ISO 9001 y

Sistema medioambiental ISO 14001

Aprobaciones CE, UL y cUL

EMC (según EN61800-3)

Distribución restringida en 1<sup>er</sup> entorno con cable de 30 m como estándar

Distribución no restringida en 2<sup>do</sup> entorno con cable de 100 m como estándar.

En la figura 2.2 se observa el diagrama de bloques del variador de frecuencia ABB ACS 300, el cual es utilizado para el desarrollo del presente proyecto.

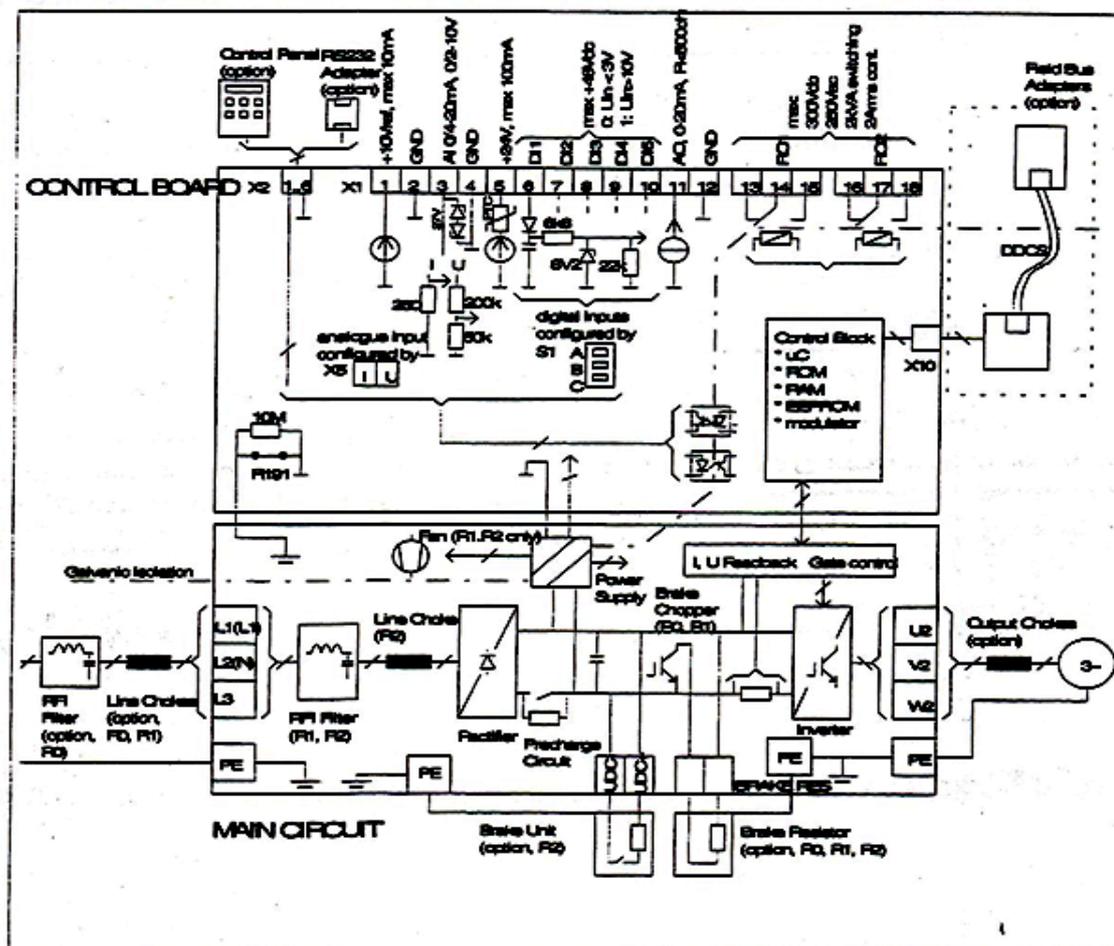


Figura 2.2 Diagrama Bloques del ACS 300

## 2.3 Especificación de Requisitos del Sistema

### 2.3.1 Computador Personal.

El computador a ser utilizado debe contar con las prestaciones tanto de hardware como de software para garantizar la eficiencia de la administración del sistema. Actualmente la tecnología del computador personal cuenta con

características de procesador, memoria RAM, capacidad de disco duro, tarjeta madre y velocidad del bus de fácil adquisición en el mercado.

La computadora a utilizarse, deberá tener adicionalmente un puerto serial RS-232 para comunicación con la Placa Base.

### **2.3.1.1 Características Técnicas del Computador Personal.**

Descripción: El PC constituye el elemento principal para realizar el control remoto del variador de frecuencia.

Función: El PC tiene las siguientes funciones:

Recopilación de los datos y almacenamiento de estos si desea el operador.

Comunicación con la Placa Base

Monitoreo del estado del variador de frecuencia.

Confrontación de los niveles frente a parámetros preestablecidos.

Control del variador de frecuencia.

Sistema Operativo: Con capacidad de administración TCP/IP

Tarjeta Madre: Intel 845 Mín.

Procesador: Intel P3, 2 GHz. Mín.

Memoria: 256 Mb.

Disco Duro: 30 Gb.

Puertos: RS-232.

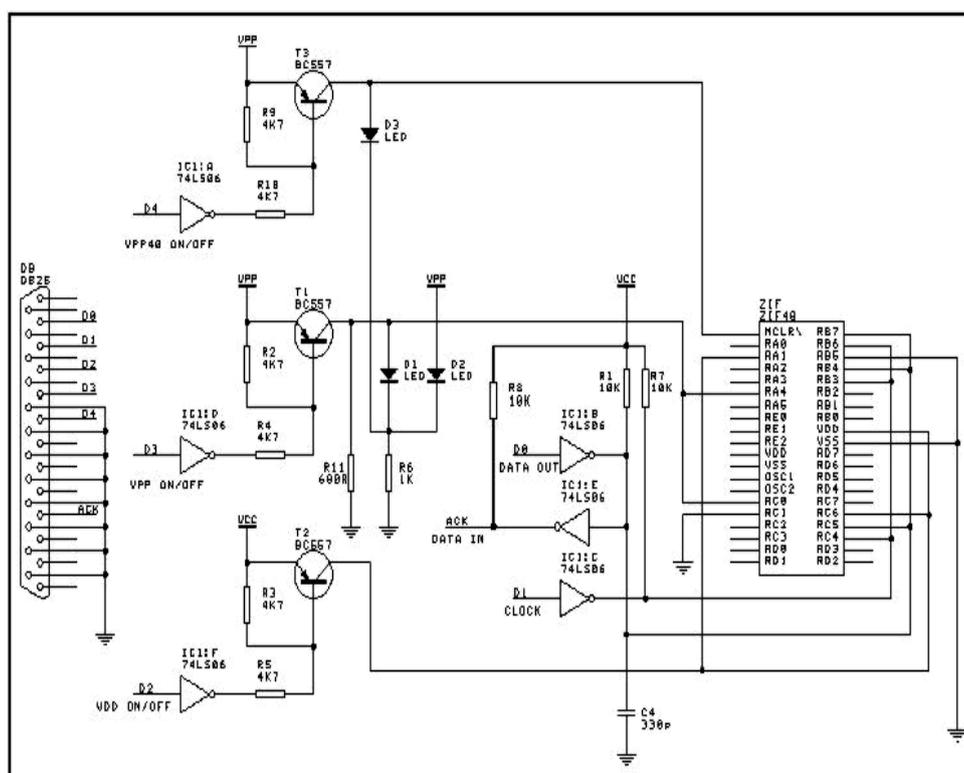
Accesorios: Monitor: Mayor 17", mouse, teclado, conductores y conectores.

## **2.4 Diseño del hardware.**

### **2.4.1 Programador: PIC Programmer II.**

En la figura 2.3, se observa el diagrama esquemático del Programador de PIC "Pic Programmer II", la cuál fue adquirida para el desarrollo del presente

módulo, ésta interfaz se utiliza para la programación microcontroladores PIC tipo flash de 18 pines . Además, utilizando una tarjeta de expansión o vía ICP puede programar también los microcontroladores PIC de 28 y 40 pines además de memorias EEPROM seriales 24LCXXX y algunas 93LCXX. Posee un puerto de conexión ICP (para Programación en Circuito) que permite programar los dispositivos directamente en la placa base donde se encuentren sin necesidad de ser retirados de su zócalo.



**Figura 2.3 Programador Pic Programmer II**

La alimentación del programador se realiza mediante una fuente de alimentación de 12V de corriente continua x 500 mA no regulada.

Este interfaz se conecta al PC mediante un cable extensor de puerto paralelo DB25 . Se utiliza normalmente con dos programas: el PICALL y el ICPROG (ambos de dominio público y provistos en el CDROM) que operan bajo Windows.



- 7 Salidas digitales colector abierto para operación de actuadores externos, todas señalizadas con leds. (**OUT1**).
- 7 Entradas digitales con protección para operación por contacto a masa, todas estas señalizadas con leds y con filtro anti rebote. (**DIG-INP**).
- Conector para display sincrónico LCD o asincrónico de 1 a 6 dígitos de 7 segmentos (**DISPLAY**).

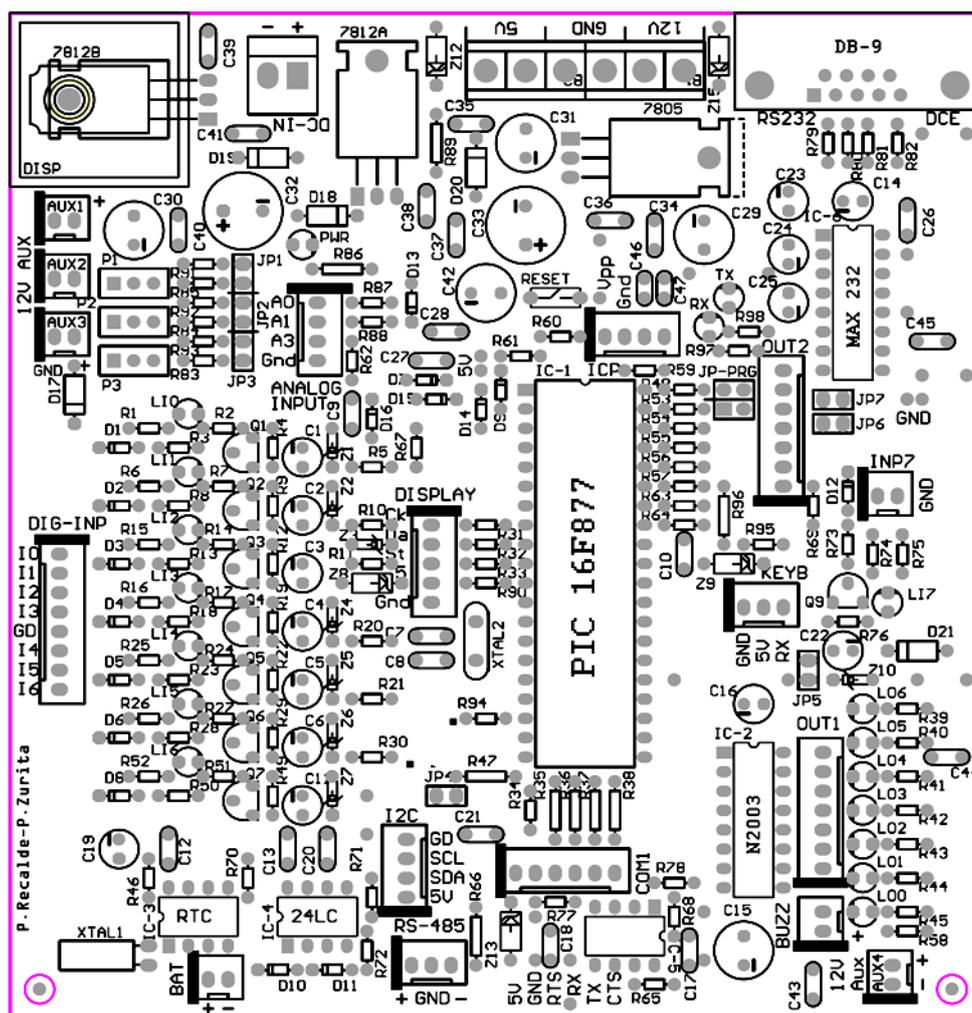


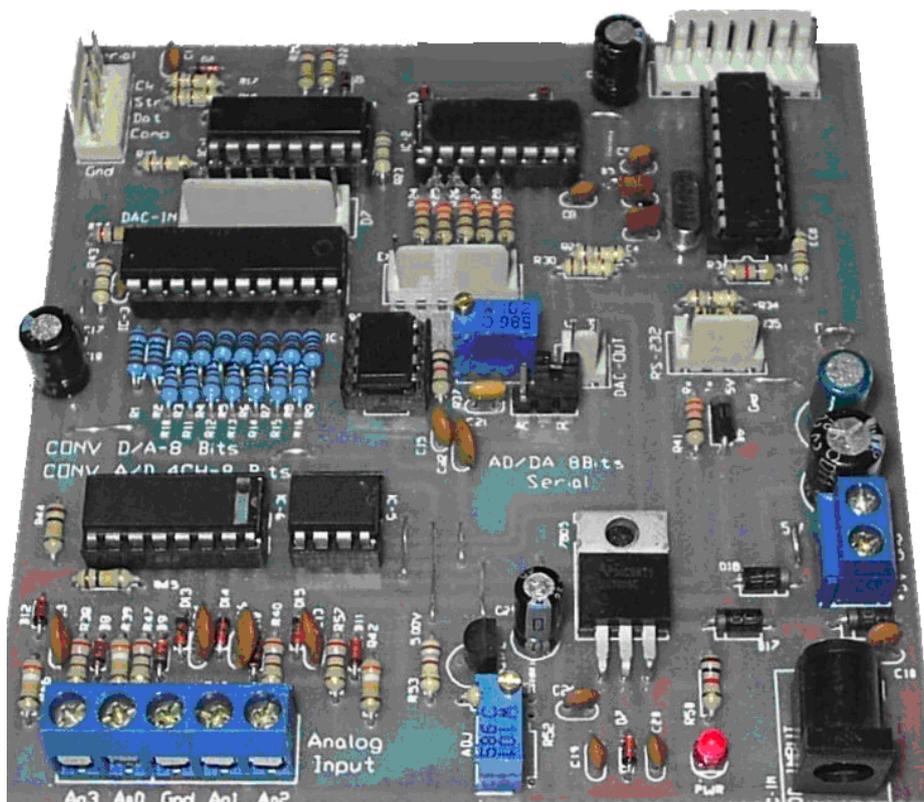
Figura 2.5 Placa Base

- Conector para teclado asincrónico rs232-TTL (**KEYB**).
- puertos de comunicaciones. Un puerto RS232(**COM2**) y un puerto RS485.

- Conector de expansión de comunicaciones para operación de puertos RS232, RS422 o RS485 adicionales (**COM1**).
- Conector DB9 para acceso al puerto RS232.
- Banco de memoria I2c para EEPROM 24LCxxx. (**IC-4**).
- Reloj de tiempo real DS1307, con conexión de batería de backup. (**IC-3**).
- Bus de expansión I2c. (**I2C**).
- Conector para buzzer externo. (**BUZZ**).
- Programación en circuito ICP (In Circuit Programmer).(**ICP**).
- Alimentación 15 a 24V DC (**DC-IN**).
- Salidas de alimentación auxiliar de 5V y 12V para alimentación de dispositivos externos (Borneras y housing).
- Operación a 20 Mhz con cristal de cuarzo.

### 2.4.3 ConversorAD-DA.

En la figura 2.6 se observa el Conversor A/D – D/A de 4 canales con salida serial, interfaz usada ingresar una señal analógica al variador de frecuencia, para su mando.



**Figura 2.6** Conversor AD-DA

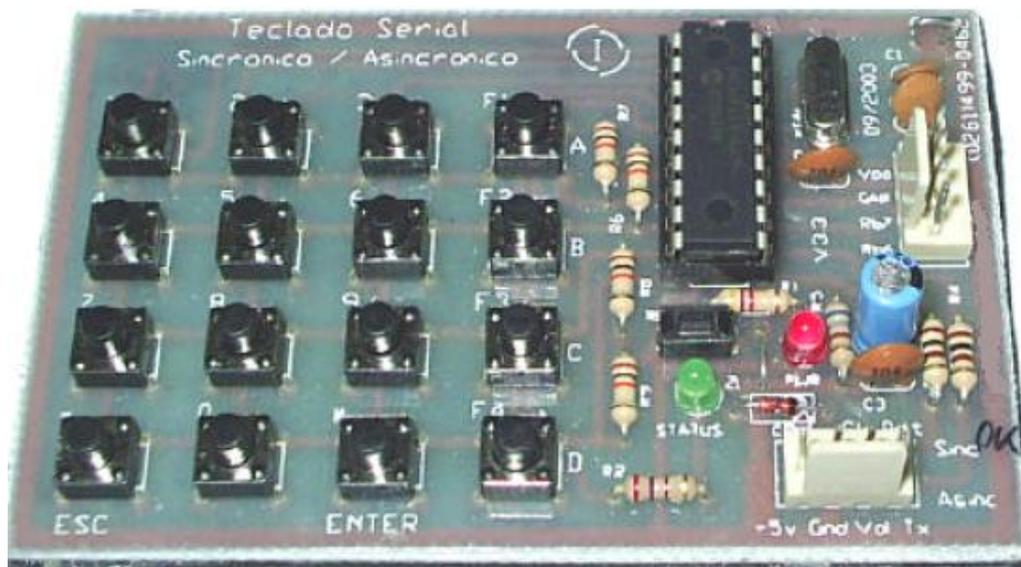
Las características técnicas principales son:

- Incluye un convertor D/A o un convertor A/D con salida acoplada en DC/AC. Este puede ser operado de forma paralela o serie, según sea el caso que se necesite.
- Esta tarjeta tiene múltiples usos, permite ser usada como un convertor Digital/Analógico con salida de tensión o bien se utiliza para lectura de 4 canales analógicos y su correspondiente conversión a digital.
- El comando de la tarjeta se puede hacer de varias formas, en paralelo, en bus sincrónico o a 1 hilo (trama serie RS232 con los 4 valores del A/D multiplexados).

- Se utiliza este interfaz para poder ingresar valores analógicos, como por ejemplo la salida de un sensor de temperatura, a un microcontrolador del tipo PIC16F84, el cual no posee internamente un conversor A/D.
- De acuerdo al firmware instalado se puede operar de diversas formas. Por ejemplo, puede ser utilizada para conversión A/D de 1CH con salida de 8 bits para ser usada en conjunto con la interfaz digital de I/O para puerto paralelo.
- Entradas y salidas señalizadas con leds.
- Alimentación: fuente de 12VCC de 500mA no regulada.
- Salida alimentación auxiliar de 5V regulada para alimentar otros dispositivos adicionales.

#### **2.4.4 Teclado numérico serie RS232-TTL.**

En la figura 2.7 se observa el Interfaz teclado numérico serie, para entrada de datos numéricos, la cuál fue adquirida para el desarrollo del presente módulo; la cual está especialmente desarrollado para aplicaciones con microcontroladores PIC, consta por una matriz de 16 pulsadores indicados con los números 0 a 9 más 4 pulsadores de función F1, F2, F3 y F4 las cuales también tienen indicaciones de 4 flechas: arriba, abajo, izquierda y derecha, y además cuenta con una tecla de ESC o \* y otra de ENTER o #.



**Figura 2.7** Teclado numérico serie RS232-TTL

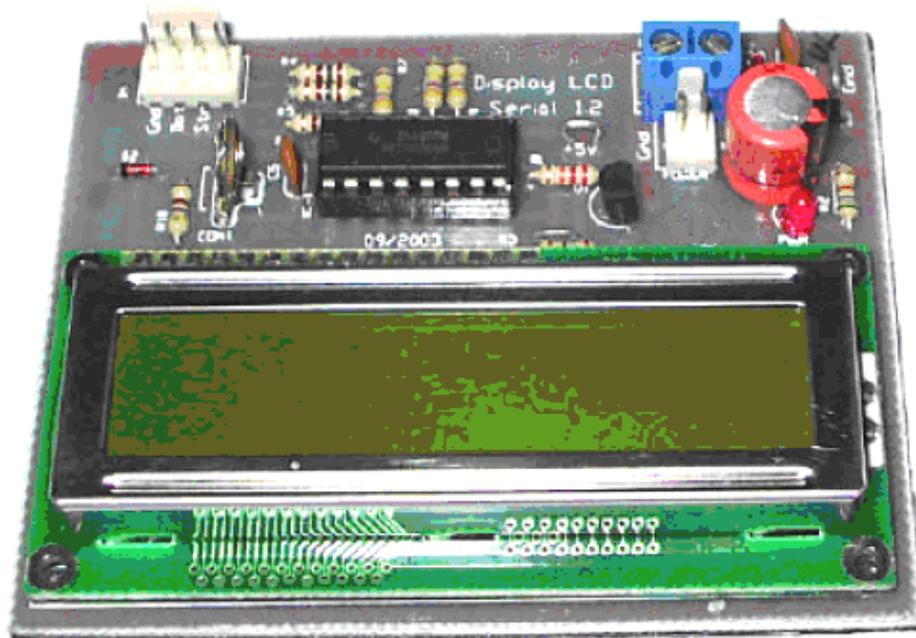
Tiene un microcontrolador PIC 16F84, el cuál controla la lógica del interfaz el cual se encarga de todo el procesamiento del interfaz. La información del pulsador presionado es transmitida por el interfaz en formato RS232 con niveles TTL (0 y 5V) a 9600 bps hacia la Placa Base para PIC 16F877.

Las características técnicas principales son:

- Alimentación 5V DC
- Protecciones contra inversión de polaridad
- Salida de datos a 9600 bps, 8 bits, sin paridad

#### **2.4.5 Interfaz de display LCD sincrónico.**

En la figura 2.8 se observa el Interfaz de Display LCD sincrónico, la cuál fue adquirida para el desarrollo del presente módulo, consta de un LCD estándares de 2x16 caracteres, posee un conector especial para la conexión de este interfaz, que permite tener una comunicación RS-232 con la Placa Base por lo que puede ser operado con solo 3 líneas, de las cuales 2 de ellas no son necesariamente exclusivas del Interfaz.



**Figura 2.8** Interfaz de display LCD sincrónico

Las características técnicas principales son:

- Alimentación 5V DC
- Protecciones contra inversión de polaridad
- Comando por bus sincrónico a 3 hilos: Ck, Data, Strobe
- Rutinas en 'C' para uso inmediato.
- Comando para backlight.

## **2.5 Diseño del software.**

### **2.5.1 IC-Prog.**

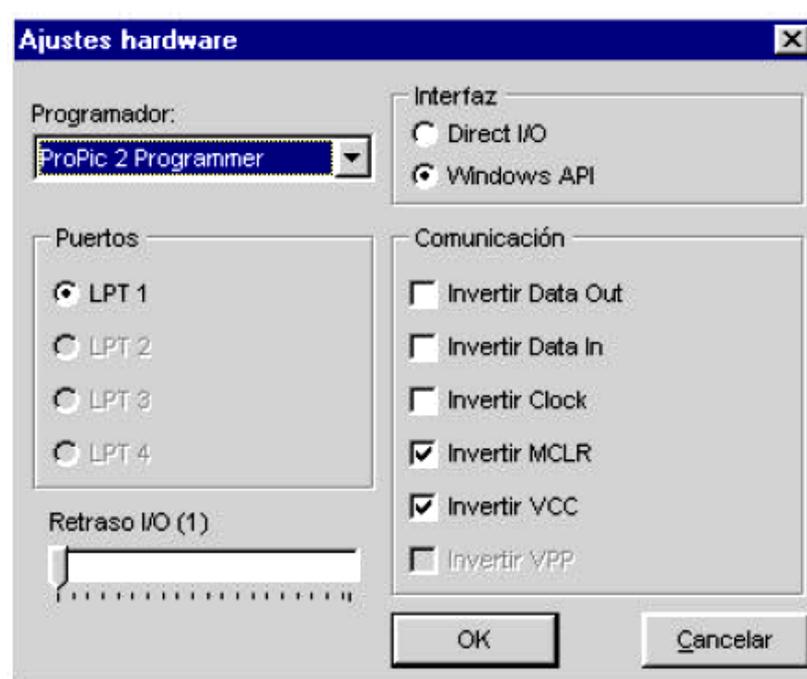
El IC-Prog requiere Windows 95, 98, ME, NT, o 2000 y un coprocesador interno o externo para funcionar. Todos los procesadores compatibles y superiores a un 386 con 8Mb de Memoria RAM deberían funcionar correctamente.

El IC-Prog es un software de programación basado en ventanas que permite la programación de diversos dispositivos y soporta numerosos tipos de programadores.

El IC-Prog es un programa que funciona bajo Windows para controlar un programador de microcontroladores PIC. Para operar este programa se necesitan conocimientos básicos de Windows y de electrónica.

Para que el programa funcione se deberá conectar a la computadora un programador, y configurar correctamente tanto a éste como al programa.

En la figura 2.9 se observa la pantalla de configuración y ajuste para PIC Programmer II, el programador a utilizar, el puerto paralelo de la PC, modo de comunicación, tipo de interfaz. Para ingresar a esta pantalla, en IC-Prog se selecciona en la barra de menú la opción Ajustes, después la opción Tipo hardware o F3, en la cuál se define los parámetros ya mencionados.



**Figura 2.9** Configuración y ajustes para Pic Programmer II

### 2.5.1.1 Área de Código (Programa).

En la figura 2.10 se observa Área de Código, la cuál siempre se presenta al ejecutar el software IC-Prog, que en valor hexadecimal utilizando 4 dígitos es

la representación del programa elaborado en lenguaje C el cuál va ser cargado al PIC 16F877, donde cada fila mostrará 8 palabras, por lo que de una fila a la otra la dirección se incrementará en 8. Una palabra posee normalmente una longitud de 16 bits por lo que el IC-Prog mostrará un valor hexadecimal entre 0000 y FFFF.

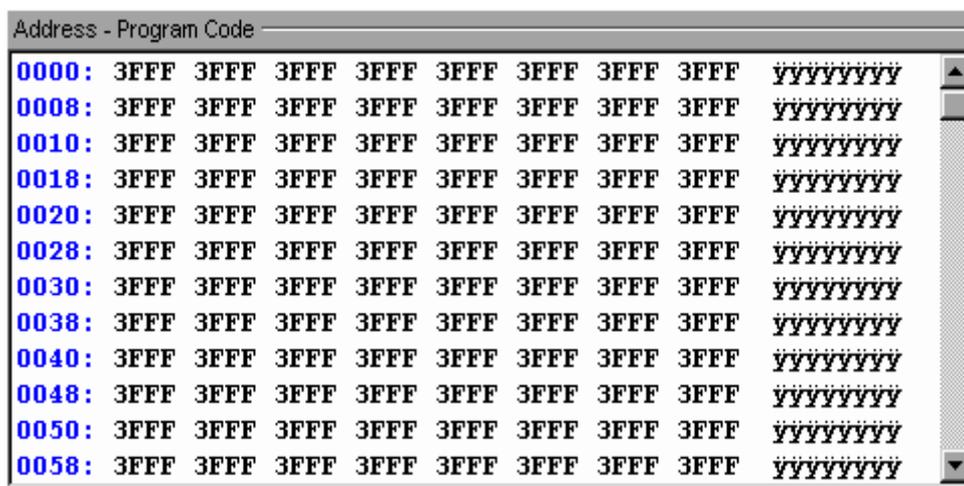


Figura 2.10 Área de Código

### 2.5.1.2 Área de datos (EEPROM).

En la figura 2.11 se observa Área de Datos, al igual que el área de código, esta se presenta al ejecutar el software IC-Prog, en cambio presenta la parte del programa a cargar el una memoria EEPROM, donde cada fila mostrará 8 palabras, pero en este caso las palabras son siempre de 8 bits por definición Siempre se mostrarán las palabras con 2 dígitos hexadecimales con un valor entre 00 y FF.

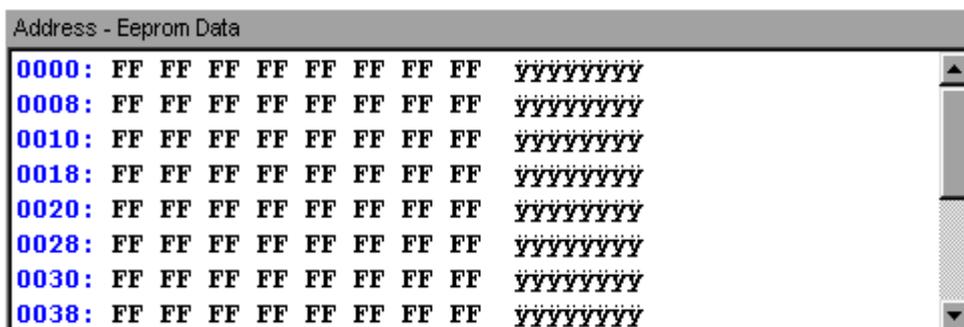


Figura 2.11 Área de Datos

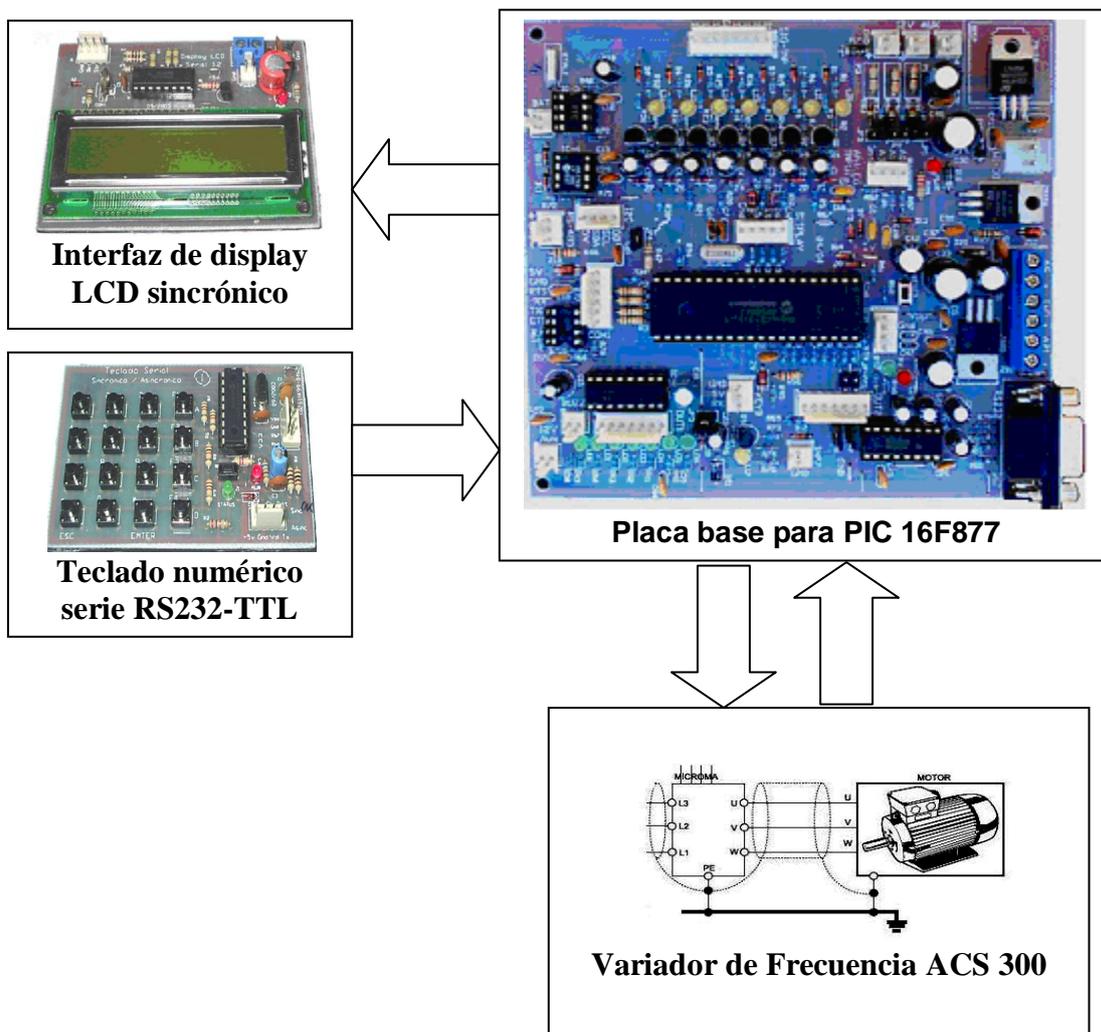
## CAPITULO III

### RESULTADOS Y PRUEBAS EXPERIMENTALES

#### 3.1 Descripción del sistema

Para describir el uso y funcionamiento de este proyecto, se lo dividió en dos partes; Manejo del Control Local y Manejo Control Remoto:

##### 3.1.1 Control Local



**Figura 3.1 Control Local**

La figura 3.1 describe el control local, el cuál consta de 4 elementos:

- Placa base para PIC 16F877.
- Interfaz de display LCD síncrono.
- Teclado numérico serie RS232-TTL.
- Variador de Frecuencia ACS 300.

A continuación se explicará más detalladamente cada una de las opciones que presenta éste.

En la figura 3.2 se tiene un esquema de los botones del teclado numérico serie RS232-TTL.

1	2	3	▲
4	5	6	▼
7	8	9	◀
Frenar	0	Enter	↻

**Figura 3.2 Botones de control**

Donde:

**1**

(0,1,2...9) Teclas numéricas utilizadas para selección de submenús de cada página, ingreso de frecuencias máximas (FMAX), mínimas (FMIN), Tiempos para rampas Ascendentes (TACC), Descendentes (TDCC) y Variables de los Procesos.



Desplazamiento hacia arriba para visualizar las diferentes opciones de cada página, también permite incrementar la frecuencia.



Desplazamiento hacia abajo para visualizar las diferentes opciones de cada página, también permite decrementar la frecuencia.



Regresar o Atrás, presionar este botón para salir de los submenús como por ejemplo de las rampas, de las frecuencias constantes, del control remoto o de los procesos.



Botón para finalizar el ingreso de un valor como por ejemplo FMAX, FMIN, TACC, etc.



Presionar para Invertir el giro del motor.



Presionar para Frenar el motor.

Después de ingresar la frecuencia y velocidad nominal del motor a utilizar, en la figura 3.3 en el LCD se muestra dos páginas disponibles para el operador.



**Figura 3.3. Páginas**

El acceso a cada página se realiza por el teclado numérico (Presionar 1 para página 1 ó 2 para página 2).

Cada página tiene opciones diferentes con relación a la otra.

Página 1:

1. Incrementar o Decrementar Frecuencia.
2. Visualización de la Velocidad del Motor y Corriente aproximada del Variador de Frecuencia.
3. Ingreso de la Frecuencia Mínima y Máxima para realizar las rampas.

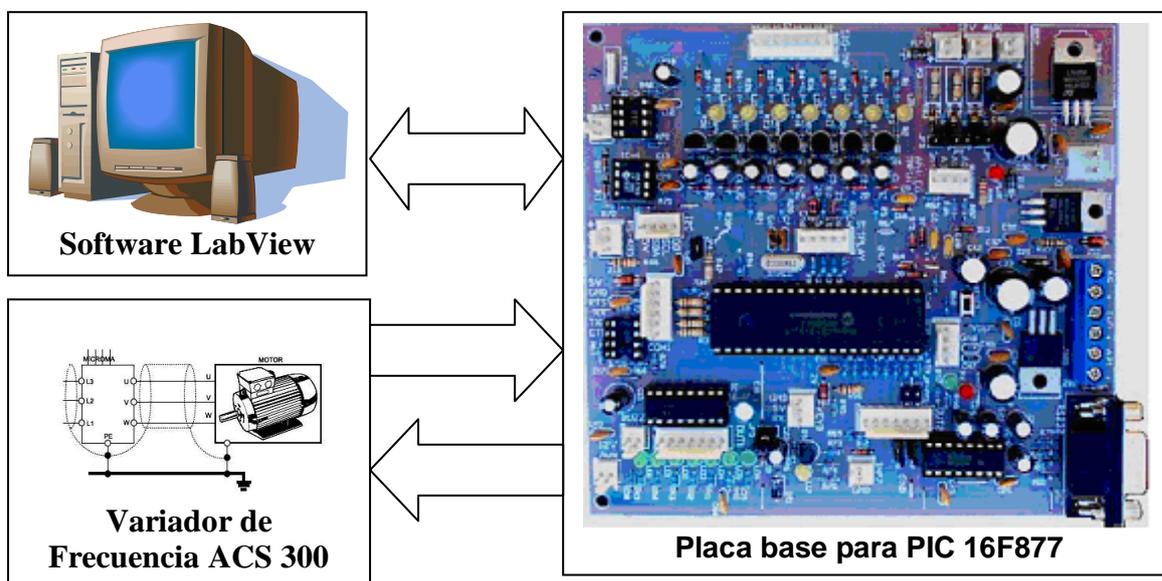
4. Ingreso de Tiempos para la Rampa Ascendente y para la Rampa Descendente.

Página 2:

1. Visualización de la Máxima Velocidad y Frecuencia del Motor.
2. Acceso al Control Remoto y Visualización del Voltaje de control suministrado al Variador.
3. Ingreso de Frecuencias Constantes 1 y 2.
4. Iniciar Rampas (Ascendente o Descendente) o Frecuencias Constantes.
5. Ingresar al Proceso 1 ó 2.

La selección, operación y función que realiza cada una, se detalla en el Manual del Usuario para el Control Local y Remoto del Variador de Frecuencia ABB<sup>10</sup>.

### 3.1.2 CONTROL REMOTO



**Figura 3.4 Control Remoto**

La figura 3.4 describe el control remoto, el cuál consta de 3 elementos:

- Placa base para PIC 16F877.

<sup>10</sup> Manual del Usuario para el Control Local y Remoto del Variador de Frecuencia ABB ver Anexo I

- PC con el Software LabVIEW.
- Variador de Frecuencia ACS 300.

Para el Control Remoto se utilizó la plataforma de Desarrollo LabVIEW, el cual da una gran versatilidad para la visualización y el control del motor.

El uso del control Remoto es mucho más sencillo que el Control Local; ya que tiene un ambiente más amigable para maniobrar al variador de frecuencia.

A continuación se indica en orden las ventanas que posee el programa desarrollado para el Control Remoto:

1. Ingreso de Password
2. Ingreso de Frecuencia y Velocidad Máxima
3. Menú Visualización – Control – Proceso
4. Visualización
5. Visualización y Control
6. Proceso
7. Comparar Gráficas

La selección, operación y función que realiza cada una, se detalla en el Manual del Usuario para el Control Local y Remoto del Variador de Frecuencia ABB ya especificado en el Control Local.

### **3.2 Pruebas experimentales**

Para esta parte, se tomará como referencia el manual de usuario para el control local y remoto del variador de frecuencia ABB.

#### **3.2.1 Control Local**

Se va realizar pruebas en el Módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW, para ello colocar el variador de frecuencia en modo remoto.

Primero realizar las conexiones respectivas para el montaje de los elementos, mostradas en el manual, encender e ingresa los valores frecuencia y velocidad nominales del motor a utilizar, para este caso los valores son 60 Hz y 1630 RPM, constatar que se visualiza en el LCD.

Se ingresar a la página1, se selecciona la opción 1 (Incrementar o Decrementar Frecuencia).

A continuación seguir con el procedimiento:

**a) Incremento de frecuencia.**

Se realizará pruebas de incremento de frecuencia, para ello, al seleccionar **Incrementar o Decrementar Frecuencia** de la página1, con la tecla de incremento (flecha hacia arriba) se varia la frecuencia y se verifica que coincide este valor con el indicado en el variador de frecuencia, se llena la tabla 3.1. prueba de incremento de frecuencia, para un análisis posterior.

Frec.Módulo (Hz)	Frec.Variador (Hz)	Diferencia entre Frec. Módulo y Frec. Variador (Hz)
0.10	0.10	0.00
10.00	10.00	0.00
15.11	15.00	0.11
20.06	20.20	-0.14
40.06	40.00	0.06
59.70	59.70	0.00

**Tabla. 3.1. Prueba de Incremento de Frecuencia**

**b) Decremento de frecuencia.**

Para las pruebas de decremento de frecuencia, es un procedimiento similar al anterior, en cambio con la tecla de decremento (flecha hacia abajo) se varia la frecuencia en el modulo local y se verifica que coincide este valor con el indicado

en el variador de frecuencia, llenar la tabla 3.2. prueba de decremento de frecuencia, para su posterior análisis.

Frec.Módulo (Hz)	Frec.Variador (Hz)	Diferencia entre Frec. Módulo y Frec. Variador (Hz)
59.70	59.70	0.00
40.06	40.0	0.06
20.06	20.20	-0.14
15.11	15.0	0.11
10.00	10.0	0.00
0.10	0.10	0.00

**Tabla. 3.2. Prueba de Decremento de Frecuencia**

**c) Realizar Rampas.**

En la realización de estas pruebas, se va a la página1, se selecciona la opción 3 (Ingreso de la Frecuencia Mínima y Máxima para realizar las rampas); en esta opción se realiza rampas ascendentes y descendentes, para ello se ingresa los valores de frecuencia mínima y frecuencia máxima, además los tiempos para cada una de las rampas, para lo cual en la página1, se selecciona la opción 4 (Ingreso de Tiempos para la Rampa Ascendente y para la Rampa Descendente), con los valores detallados en la tabla 3.3 y tabla 3.4 se realiza esta prueba, según sea el caso.

**c.1) Rampa Ascendente.**

Para iniciar a realizar las rampas, en la página2, se selecciona la opción 4 (Iniciar Rampas, Ascendente o Descendente o Frecuencias Constantes), se llena la tabla 3.3 prueba rampa ascendente, se verifica con los valores de frecuencia indicados en el variador de frecuencia y los valores de tiempo cronometrados.

VALORES INGRESADOS			VALORES REALES		
Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Ascendente (seg)	Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Ascendente (seg)

5.00	50.00	30.00	5.00	49.20	≈30.00
15.00	40.00	10.00	14.40	39.30	≈10.00
20.00	50.00	5.00	19.60	49.20	≈5.00

**Tabla 3.3 Prueba Rampa Ascendente**

c.2) Rampa Descendente.

Para las pruebas de rampas descendentes, se realiza el mismo procedimiento que el anterior, la diferencia es que se selecciona iniciar rampa descendente de la opción 4 (Iniciar Rampas, Ascendente o Descendente o Frecuencias Constantes), se llena la tabla 3.4 prueba rampa descendente, se comprueba con los valores indicados en el variador de frecuencia y los tiempos cronometrados, para realizar un análisis posterior.

VALORES INGRESADOS			VALORES REALES		
Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Descendente (seg)	Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Descendente (seg)
20.00	50.00	5.00	19.90	49.23	≈5.00
15.00	40.00	10.00	14.50	39.30	≈10.00
5.00	50.00	30.00	5.20	19.70	≈30.00

**Tabla 3.4 Prueba Rampa Descendente**

**d) Realizar Procesos.**

Para realizar estas pruebas, en la página 2, se selecciona la opción 5 (Ingresar al Proceso 1 o 2), se ha elaborado una tabla de datos para cada proceso, con esto se ayuda para la realización de esta prueba.

d.1) Proceso 1.

Al elegir proceso 1, se ingresa los valores pedidos (Frecuencias y Tiempos), correspondientes según los valores especificados en la tabla 3.5 prueba proceso1, seleccionar COMENZAR, se toma nota de

las frecuencias y tiempos reales y se los compara con los ingresados, se completa la tabla correspondiente para realizar un análisis.

	Valor Ingresado (Hz o seg )	Valor Visualizado o Médido (Hz o seg )	Valor Ingres. - Valor Visualiz. o Medido (Hz o seg )
<b>Frec. Inicial1</b>	0.00	0.03	-0.30
<b>Frec. Final1</b>	30.00	29.40	0.60
<b>Tiempo Subida1</b>	10.00	≈10.00	≈0.00
<b>Tiempo Bajada1</b>	20.00	≈20.00	≈0.00
<b>Frec. Inicial2</b>	0.00	0.30	-0.30
<b>Frec. Final2</b>	40.00	39.30	0.70
<b>Tiempo Subida2</b>	10.00	≈10.00	≈0.00
<b>Tiempo Bajada2</b>	20.00	≈20.00	≈0.00
<b>Tiempo Alto</b>	5.00	≈5.00	≈0.00
<b>Tiempo Bajo</b>	5.00	≈5.00	≈0.00

**Tabla 3.5 Prueba Proceso1.**

d.2) Proceso 2.

El procedimiento de ésta prueba, es similar al anterior, se elige proceso 2, se ingresa los valores pedidos y que están dados en la tabla 3.6 prueba proceso2 (Frecuencias y Tiempos), después se elige COMENZAR; se toma nota de las frecuencias y tiempos reales, se llena la tabla correspondiente, para realizar un análisis de los datos obtenidos.

	Valor Ingresado (Hz o seg )	Valor Visualizado o Médido (Hz o seg )	Valor Ingres. - Valor Visualiz. o Med. (Hz o seg )
<b>Frec_Inicial 1</b>	10.00	9.70	0.30
<b>Frec. Final 1</b>	20.00	19.40	0.60
<b>Tiempo1</b>	15.00	≈15.00	≈0.00
<b>Tiempo2</b>	25.00	≈25.00	≈0.00
<b>Frec. Inicial 2</b>	10.00	9.70	0.30

<b>Frec. Final 2</b>	50.00	49.30	0.70
<b>Tiempo3</b>	15.00	≈15.00	≈0.00
<b>Tiempo4</b>	25.00	≈25.00	≈0.00
<b>Tiempo alto</b>	10.00	≈10.00	≈0.00
<b>Tiempo bajo</b>	15.00	≈15.00	≈0.00

**Tabla 3.6 Prueba Proceso2.**

### **3.2.2 Control Remoto**

Para realizar las pruebas en el Módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW, se coloca el variador de frecuencia en modo remoto y en el módulo página 2 opción 2 habilitar modo remoto.

En la PC, se ejecuta el programa para el control remoto, desarrollado en la Plataforma de Desarrollo LabVIEW, previamente cargado, se conecta la interfase RS-232 y se sigue los pasos que se muestran en el programa.

Ingresar los valores frecuencia y velocidad nominal del motor, para este caso son 60 Hz y 1630 RPM. A continuación se sigue con el procedimiento:

#### **Prueba Visualización.**

Para la realización de esta prueba se ingresa a la opción Visualización, verificar si hay comunicación de la PC al variador de frecuencia.

Visualmente se comprueba que la gráfica de frecuencia que se muestra en la PC, corresponde con la frecuencia que se indica en el variador.

#### **Prueba Visualización y Control.**

b.1) Se Ingresa a la opción Visualización y Control, se verifica que hay comunicación de la PC al Variador de frecuencia.

Se puede observar gráficamente el estado del variador, comprobando que la frecuencia del variador corresponde con la

frecuencia indicada en la gráfica. En la parte de Control, se puede comprobar que efectivamente se puede variar el estado del variador de frecuencia, de una manera fácil y rápida.

Para las siguientes pruebas, se sigue un procedimiento similar al utilizado en las pruebas de Control Local.

### **b.2) Incremento de frecuencia.**

Para realizar la prueba de incremento de frecuencia, se fija la frecuencia mínima y con el botón de incremento se varia la frecuencia hasta la máxima frecuencia, se verifica que coincide este valor con el indicado en el variador de frecuencia y con la gráfica mostrada, se llena tabla 3.7 Prueba de Incremento de Frecuencia, para un análisis posterior.

<b>Frec. Modulo (Hz)</b>	<b>Frec. Variador (Hz)</b>	<b>Diferencia entre F. Módulo y Frec. Variador (Hz)</b>
0.10	0.10	0.00
10.00	10.00	0.00
15.11	15.00	0.11
20.06	20.20	-0.14
40.06	40.00	0.06
59.70	59.70	0.00

**Tabla. 3.7 Prueba de Incremento de Frecuencia**

### **b.3) Decremento de frecuencia.**

Al contrario de la prueba anterior, se fija la frecuencia máxima se pulsa el botón de decremento, se varia la frecuencia hasta llegar a la frecuencia mínima, se verifica que coincide este valor con el indicado en el variador de frecuencia, se llena la tabla 3.8 Prueba de Decremento de Frecuencia, para realizar un análisis posterior.

<b>Frec. Modulo (Hz)</b>	<b>Frec. Variador (Hz)</b>	<b>Diferencia entre F. Módulo y Frec. Variador (Hz)</b>
59.70	59.70	0.00
40.06	40.00	0.06
20.06	20.20	-0.14
15.11	15.00	0.11

10.00	10.00	0.00
0.10	0.10	0.00

**Tabla. 3.8. Prueba de Decremento de Frecuencia**

**b.4) Realizar Rampas.**

En la realización de estas pruebas, se fija las frecuencias mínimas y máximas y los tiempos ascendente y descendente en los lugares especificados, siguiendo la tabla correspondiente para cada una de estas pruebas.

**b.4.1) Rampa Ascendente.**

Se llena la tabla 3.9 Prueba de Rampa Ascendente, pulsando el botón de rampa ascendente, para los valores indicados en la tabla.

VALORES INGRESADOS			VALORES REALES		
Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Ascendente (seg)	Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Ascendente (seg)
5.00	50.00	30.00	6.00	49.70	≈30.00
15.00	40.00	10.00	14.70	39.60	≈10.00
20.00	50.00	5.00	19.20	49.70	≈5.00

**Tabla 3.9 Prueba Rampa Ascendente**

**b.4.2) Rampa Descendente.**

Al contrario de la prueba anterior, se llena la tabla 3.10 Prueba de Rampa Descendente, pulsando el botón de rampa Descendente, para los valores indicados en la tabla.

VALORES INGRESADOS			VALORES REALES		
Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Descendente (seg)	Frec. Min. (Hz)	Frec. Máx. (Hz)	Tiempo Descendente (seg)
20.00	50.00	5.00	19.60	49.70	≈5.00
15.00	40.00	10.00	14.50	39.10	≈10.00
5.00	50.00	30.00	5.20	49.50	≈30.00

**Tabla 3.10 Prueba Rampa Descendente**

### **b.5) Frecuencias constantes.**

Para la realización de estas pruebas, se va llenar las tablas correspondientes para cada una.

#### **b.5.1) Frecuencia Constante1.**

Se fija la frecuencia indicada en la tabla 3.11 Prueba Frecuencia Constante1, con el botón Frecuencia Constante1 se realiza esta prueba, se llena la tabla para su posterior análisis.

<b>Frec. Program (Hz)</b>	<b>Frec. Variador (Hz)</b>	<b>Diferencia entre F. Programa y Frec. Variador (Hz)</b>
0.10	0.10	0.00
25.00	24.90	0.10
55.00	54.70	0.30

**Tabla 3.11 Prueba Frecuencia Constante1.**

#### **b.5.2) Frecuencia Constante2.**

Al contrario de lo anterior, con el botón Frecuencia Constante2, se realiza esta prueba, se llena la tabla 3.12 Prueba Frecuencia Constante2, con los valores correspondientes, se verifica que coincide este valor con el indicado en el variador de frecuencia, para su posterior análisis.

<b>Frec. Program (Hz)</b>	<b>Frec. Variador (Hz)</b>	<b>Diferencia entre F. Programa y Frec. Variador (Hz)</b>
59.00	58.50	0.50
30.00	29.60	0.40
5.00	5.00	0.00

**Tabla 3.12 Prueba Frecuencia Constante2.**

**Nota1:** Se debe tener en cuenta que en este tipo de control, el programa permite en cualquier instante guardar registros de gráficas, ya sea en formato .xml(Excel) o para comparar en el mismo programa, dando la opción de hacerlo con 4 gráficas.

**Nota2:** Luego de realizar todas las pruebas, se puede establecer un análisis y la confiabilidad del Módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW, para concluir con el desarrollo del presente proyecto.

### **3.3 Análisis de Resultados**

Se va realizar un análisis del desempeño del Módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW, con los valores indicados por variador de frecuencia, y tiempos cronometrados.

#### **3.3.1 Análisis del Control Local**

A continuación se realizará un análisis de los valores obtenidos en las tablas correspondientes a estas pruebas.

Se obtuvo los siguientes resultados para cada una de ellas:

##### Análisis del literal 3.2.1.a) Incremento de frecuencia.

Las frecuencias incrementadas en el módulo, coinciden en su mayoría, cabe resaltar que en algunos valores se tiene un error menor o igual a  $\pm 0.14$  Hz, en la lectura.

##### Análisis del literal 3.2.1.b) Decremento de frecuencia.

Se observa que al igual que en caso anterior se tiene un error menor o igual a  $\pm 0.14$  HZ, en la lectura de los datos, pero la mayoría coincide con el valor de frecuencia especificado.

#### Análisis del literal 3.2.1.c) Realizar Rampas.

De la tabla3.3 Rampa Ascendente, se obtuvo que en los valores extremos se tiene una mayor precisión en la lectura de los valores de frecuencia y para los valores medios de frecuencia se tiene un error menor a  $\pm 0.8$  Hz, en cambio los tiempos coinciden.

Para la tabla3.4 Rampa Descendente, los valores en sus frecuencias extremas se tiene una mayor precisión en la lectura y para los valores medios de frecuencia se tiene un error menor a 0.8 Hz, teniendo en cuenta que los tiempos de descenso coinciden, al igual que en las rampas ascendentes.

#### Análisis del literal 3.2.1.d) Realizar Procesos.

Al realizar estas pruebas, de la tabla3.5 Proceso1, se obtuvo un proceso confiable, el error en la lectura de los datos de frecuencia es menor o igual a 0.7 Hz, los tiempos son aproximadamente los mismos, la inversión de giro es fiable; se consiguió un proceso continuo del cual se puede salir en el instante que se desee.

Para la tabla3.6 Proceso2, al igual que el anterior los valores en sus frecuencias se tiene un error menor o igual a  $\pm 0.7$  Hz en la lectura, los tiempos e inversión de giro fueron confiables, dando un proceso continuo para diversas aplicaciones.

### **3.3.2 Análisis del Control Remoto.**

De los datos conseguidos en estas pruebas, obtenidos en las tablas correspondientes se realizó el siguiente análisis.

#### Análisis del literal 3.2.2.a) Visualización.

Se verificó que la gráfica de frecuencia que se muestra en la PC por el programa desarrollado en la Plataforma de Desarrollo LabVIEW, corresponde con la frecuencia que se indica en el variador.

#### Análisis del literal 3.2.1.b) Visualización y Control.

Se puede concluir que hay comunicación de la PC al Variador de frecuencia, las graficas de frecuencia mostradas corresponden con los valores de frecuencia del variador.

También se observar que se tiene un mando al variador de frecuencia mediante la PC, se puede variar la frecuencia y realizar acciones específicas.

#### Análisis del literal 3.2.2.b.2) Incremento de frecuencia.

De los valores obtenidos en la tabla3.7 Incremento de Frecuencia, se obtiene que la lectura de los datos coincida con los realizados en el Control Local, obteniendo un error en la lectura menor o igual a  $\pm 0.14$  Hz, pero en la mayoría de datos se tiene mayor precisión.

#### Análisis del literal 3.2.2.b.3) Decremento de frecuencia.

Con los valores obtenidos de la tabla3.8 Decremento de Frecuencia, se observa que la lectura coincide como en el caso anterior, se obtiene resultados de lectura menor a  $\pm 0.16$  Hz.

#### Análisis del literal 3.2.2.b.4) Realizar Rampas.

Se puede concluir, que de la tabla3.9 Rampa Ascendente, en los valores extremos se tiene una mayor precisión en la lectura de los valores de frecuencia y para los valores medios de frecuencia se

tiene un error menor o igual a  $\pm 1$  Hz, en los tiempos hay un error aproximado de cero.

Para la tabla 3.10 Rampa Descendente, los valores en sus frecuencias extremas se tiene una mayor precisión en la lectura y para los valores medios de frecuencia se tiene un error menor o igual a  $\pm 0.7$  Hz, los tiempos coinciden aproximadamente, al igual que en las rampas ascendentes.

#### Análisis del literal 3.2.2.b.5) Realizar Frecuencias Constantes.

Al realizar el análisis de los valores obtenidos de la tabla 3.11 Frecuencia Constante 1, se concluye que el error aumenta en la lectura, es menor o igual a  $\pm 0.3$  Hz, en sus valores medios, lo contrario en la lectura de sus valores extremos.

Para los valores obtenidos de la tabla 3.12 Frecuencia Constante 2, se tiene valores similares al Proceso 1, un error menor o igual a  $\pm 0.5$  Hz, en la lectura de sus valores medios, para más confiable en sus frecuencias extremas.

### **Conclusión Control Local**

De las pruebas realizadas en el Control Local, se tiene un buen margen de confiabilidad en el manejo del variador de frecuencia desde el módulo y en la lectura de la frecuencia de este; teniendo en cuenta que el error es menor a  $\pm 0.15$  Hz, para los incrementos de frecuencia por incrementos o decrementos, y aumenta el error para las otras pruebas aproximado a  $\pm 0,7$  Hz, los tiempos para puntos específicos de las pruebas coinciden.

### **Conclusión Control Remoto**

Para las pruebas realizadas en el Control Remoto, se concluye que la comunicación desde la PC al variador de frecuencia es eficiente, al

igual que en el Control Remoto se tiene un error de lectura menor a  $\pm 0.15$  Hz en los incrementos y decrementos de frecuencia por pasos, este error aumenta para las pruebas siguientes a  $\pm 0.7$  Hz, se tiene en cuenta que los tiempos coinciden en las pruebas correspondientes; los datos obtenidos de frecuencia son los mismos a los graficados por el software desarrollado en LabVIEW, en tiempo real. Inversión de giro y guardando las gráficas para generar registros para análisis posteriores.

### **Conclusión General**

Se concluye que de la parte de Control Local y Control Remoto, estos coinciden en frecuencias y tiempos, se tiene un buen margen de confiabilidad, se debe resaltar que la señal analógica de corriente de salida del variador, no es completamente lineal, por lo cual este puede ser una de las razones por la que no se consigue una precisión total sobre la lectura de los datos de frecuencia del variador. Las acciones de inversión de giro, incremento y decremento de frecuencia, generación de rampas, frecuencias constantes y realización de procesos en control local las realizan de una manera confiable.

### **3.4 Alcances y Limitaciones**

En la actualidad la gran mayoría de industrias incorporan en sus respectivos procesos una infinidad de elementos sean estos mecánicos, eléctricos, electrónicos o neumáticos, los mismos que cumplen determinadas funciones para llegar al cumplimiento de su cometido final, esto es el de realizar una acción concreta de acuerdo al tipo de proceso a tratarse.

La velocidad del motor asíncrono depende de la forma constructiva del motor y de la frecuencia de alimentación; un variador de frecuencia es un elemento seguro en el control de un motor de corriente alterna; como la frecuencia de alimentación que entregan las Compañías de electricidad es

constante, la velocidad de los motores asincrónicos es constante, salvo que se varíe el número de polos, el deslizamiento o la frecuencia.

### **3.4.1 Alcances.**

Por lo expuesto se puede decir que entre los alcances del proyecto se tiene:

- Ya que se esta realizando un control electrónico de un variador de frecuencia para un motor AC, se puede manejar sus parámetros como velocidad, sentido de giro y frenado, localmente desde el módulo electrónico o en forma remota a través de una PC, obteniendo un sistema de control SCADA que puede aplicase con fines de investigación para un variador de frecuencia.
- El módulo de Control Local y Remoto para el Variador de Frecuencia y la Comunicación con la Plataforma de Desarrollo LabVIEW, esta realizado para que el estudiante, de una manera sencilla se familiarice con el funcionamiento del variador de frecuencia y a su vez con la plataforma de Desarrollo LabVIEW.
- En el Control Local y Remoto se puede generar rampas de aceleración, desaceleración con tiempos definidos, frecuencias constantes, y realizar una secuencia específica o proceso en tiempo real, optimizando el desempeño del variador de frecuencia.
- El Control Local y Remoto permite visualizar la frecuencia que suministra el variador, la velocidad del motor.
- A través de una señal analógica de 0 a 10 voltios se maneja la frecuencia del variador y por una señal de salida de este de 0 a 20 miliamperios de éste, se comprueba que la frecuencia suministrada al motor sea la correcta; estas variables serán visualizadas en el Control Local y Remoto.

- El módulo de Control Local y Remoto permite un mando seguro hacía el variador de frecuencia, permitiendo modificar la frecuencia y simultáneamente el voltaje aplicado al motor AC a ser utilizado.
- Ya que se cuenta con una comunicación con la Plataforma de Desarrollo LabVIEW, se tiene ventajas que permite ésta, como la facilidad de guardar registros del desempeño del variador de frecuencia.
- En la Plataforma de Desarrollo LabVIEW, se puede visualizar rampas de aceleración y desaceleración, frenado, inversión de giro y frecuencias constantes en tiempo real, permitiendo un análisis del desempeño del variador de frecuencia inmediato, o puede ser registrados para un análisis posterior.
- La comunicación vía RS-232 entre la placa base del microcontrolador PIC 16F877 y la PC, utilizando la Plataforma de Desarrollo LabVIEW permite un mando remoto del variador de frecuencia en procesos en los cuales el ambiente en el que se desarrolla puede ser inaccesible ya sea por la presencia de agentes como el ruido, gases, ácidos, etc, entonces es imprescindible establecer un control por mando remoto dada la situación física incómoda especialmente para el operario, al ofrecer una gran versatilidad para el monitoreo y mando del motor.
- El Microcontrolador (PIC) es una herramienta muy poderosa en el diseño de este proyecto, las características técnicas que tiene, resulta adecuado para simplificar el control local y remoto del variador de frecuencia, logrando generar una secuencia para un proceso determinado por el usuario.
- La Placa Base permite reprogramar fácilmente el microcontrolador PIC 16F877 mediante una comunicación ICP con el programador PIC Programmer II, sin desmontar dicho microcontrolador de la placa.

- El manejo del módulo es sencillo ya que fue creado con un ambiente amigable para el usuario.

### 3.4.2 Limitaciones.

Las limitaciones del proyecto son:

- El modulo de Control Local y Remoto fue creado específicamente para el variador de frecuencia ABB ACS 300.
- Para realizar una nuevo proceso se debe reprogramar el microcontrolador para que realice la secuencia requerida.
- La salida analógica de corriente de 4 a 20 mA del variador de frecuencia, no es completamente lineal, por lo cual limita en la lectura de los datos del estado del variador de frecuencia, generando un error menor a  $\pm 0.3$  Hz para algunos valores de frecuencia.
- El Proceso no podrá ser muy extenso debido a la escasa memoria estática del microcontrolador PIC 16F877.
- La resolución del conversor Digital – Análogo es de 8 bits lo cual no permite un control mas exacto de la señal analógica que ingresa al variador de frecuencia, y por consiguiente la frecuencia que se va a suministrar al motor.

## 3.5 Análisis Técnico – Económico

### 3.5.1 Introducción.

En la tabla se detalla el costo de los elementos que conforman el módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW. El cuál es necesario para la realización de este proyecto.

Item	Equipo	Precio	Cant.	Unid.	P. Total
------	--------	--------	-------	-------	----------

		Unitario			
1	Placa base para PIC 16F877	\$80	1	U	\$80
2	Convertor D/A	\$42	1	U	\$42
3	Teclado numérico serie RS232-TTL	\$23	1	U	\$ 23
4	Interfaz de display LCD sincrónico	\$55	1	U	\$55
5	Programador para PICs	\$18	1	U	\$18
6	Varios: Caja, Conductores y Conectores	\$41	---	---	\$41
TOTAL 1					\$259

**Tabla 3.13 Módulo de Control Local Electrónico.**

Realizando un análisis, de la tabla anterior, se observa que el costo total no es muy elevado, comparando con el equipo que va controlar (variador de frecuencia ACS 300), el cuál ya tiene la Escuela Politécnica del Ejercito-Latacunga.

El costo de los elementos que intervienen en el proyecto, al comparar con un equipo que se puede encontrar en mercado (PLC's, un cerebro electrónico, u otro), no varía mucho del precio total expuesto, la ventaja de haber realizado el presente proyecto, fue llenar todos los objetivos propuestos con el módulo, lo que talvez no se podía haber hecho con un solo equipo adquirido.

### 3.5.2 Conclusión Técnico – Económico.

Al concluir con el análisis del presupuesto, se detalla los costos de los elementos que conforman el “Módulo de Control Local Electrónico y Remoto para el Variador de Frecuencia y Comunicación con la Plataforma de Desarrollo LabVIEW”. Los costos para elaborar son reducidos, las placas son fáciles de adquirir al igual que elaborarlas.

**Nota1:** Cabe destacar que el computador, el software, el motor y el variador de frecuencia, los tiene la Escuela Politécnica del Ejercito-Latacunga los cuales no es necesario adquirir.

## **CONCLUSIONES**

- Luego de realizar las pruebas del caso y ver el Módulo funcionando, se concluye que se logró el objetivo propuesto:
  - Implementar un control local y remoto de un Variador de Frecuencia para un motor AC empleando el microcontrolador PIC 16F877 y la plataforma de desarrollo LabView.
  - Analizar los parámetros que intervienen en la operación de un Variador de Frecuencia.
  - Analizar los parámetros que intervienen en la operación de un motor AC.
  - Analizar la arquitectura y programación del microcontrolador PIC 16F877 y periféricos auxiliares.
  - Evaluar las herramientas de hardware y software con las que se cuenta para el desarrollo de aplicaciones para el presente proyecto de investigación.
  - Cumplir con los requisitos establecidos para obtener el título de Ingeniero Terminal en Electrónica e Instrumentación.
- El Microcontrolador PIC 16F877 es una herramienta muy poderosa en el diseño de este proyecto, las características técnicas que tiene, hace fácil el manejo de entradas y salidas analógicas y digitales, para el control local electrónico del variador de frecuencia.
- El Microcontrolador PIC 16F877 permitió de una manera sencilla y confiable la comunicación vía RS-232 con la PC utilizando para ello LabVIEW con lo que se consigue el mando remoto del variador de frecuencia.

- El proyecto permite el mando local y remoto de un variador de frecuencia permitiendo visualizar curvas: de rampas de aceleración y desaceleración, de frenado, de inversión de giro y de frecuencias constantes casi en tiempo real.
- El proyecto permite a través de la Plataforma de Desarrollo LabVIEW un análisis de curvas de aceleración y desaceleración, en el instante del funcionamiento del variador de frecuencia.
- El proyecto permite a través de la Plataforma de Desarrollo LabVIEW crear registros y guardarlos en el disco duro del PC, los cuales se pueden recuperar y volver a obtener las curvas de aceleración y desaceleración, del funcionamiento del variador de frecuencia para fines específicos para el profesor como para el alumno.
- El proyecto utiliza los equipos de hardware como motor, variador de frecuencia, la PC, etc. y software como el programa de desarrollo LabVIEW, necesarias con las que se cuenta, en las instalaciones de la Escuela Politécnica del Ejército – Latacunga, lo que permitió el desarrollo del presente proyecto de investigación, de una manera económica.
- La gran diversidad de software dedicada para sistemas SCADA permite garantizar tanto el mando como el monitoreo del módulo local y remoto del proyecto.
- El sistema de programación por eventos utilizado en este proyecto, es más sencillo y mucho más eficiente que la programación del PIC tradicional o ensamblador, ya que el computador responderá al evento realizado.
- Ya que el variador de frecuencia consta de una señal estándar de corriente (4–20 mA), permite colocar el módulo a distancias

considerables ya que no hay pérdidas por caídas de tensión, la cual va a la par con la frecuencia de trabajo de éste y que permitió visualizar las diferentes curvas y monitorear el desempeño del variador de frecuencia.

- La poca no linealidad de la salida analógica de corriente de 4 a 20 mA del variador de frecuencia, demostrada en laboratorio, no permitió tener medidas exactas de lectura de frecuencia, ya que para la curva de esta no coincidió exactamente con la curva de frecuencia, por lo cuál se presento un error de décimas de hertz, en la visualización de los datos tanto en la parte local y remota del proyecto.
- Este módulo podría manejar otros dispositivos que tengan señales analógicas de corriente de 4 a 20 mA como salida.
- Proyectos como este, realizados a conciencia incrementan los conocimientos, ocasionando investigar profundamente sin importar muchas veces el tiempo, considerando que, el objetivo de un profesional no es obtener un título buscando el camino sin obstáculos, sino la satisfacción de haberlo realizado, cumpliendo objetivos y llegando a la meta, a costa de sacrificio, esfuerzo y dedicación.

### **RECOMENDACIONES**

- Se recomienda linealizar la señal de salida analógica de corriente de 4 a 20 mA, en este caso del variador de frecuencia, esto se puede realizar por software en la programación mediante fórmulas matemáticas o comparaciones, o hardware utilizando circuitos integrados (circuitos operacionales), ya que se tiene una mejor exactitud en el manejo y control del variador de frecuencia,
- Antes de utilizar el módulo para prácticas, el estudiante u operador debe haber leído el manual de operación de éste, para evitar algún problema con su manejo.

- Se recomienda buscar nuevas formas de mando para muchos elementos de control existentes y no tener que importar las ya hechas.
- Ya que la presentación del proyecto tanto en la parte local y remota es amigable, se da la facilidad al operador que utilice todas las ventajas de una manera rápida para el mando del variador de frecuencia.
- Se recomienda que la operación y mantenimiento sea responsabilidad del laboratorio al cuál se halla confinado el módulo.
- Este módulo debe ser utilizado por personas que tienen conocimientos teórico prácticos de electrónica o electromecánica, bajo supervisión del profesor guía.
- Se debe cuidar de no golpear, mojar o dejar el equipo en lugares húmedos o corrosivos para un mejor mantenimiento de los elementos.
- El módulo debe dejarse apagado después de cada práctica.
- Tener cuidado con trabajar con alta tensión, ya que podría causar daño a la persona que opera el equipo como al equipo mismo, también puede introducir ruido y dar valores errados.
- En el desarrollo de nuestra investigación se utilizó un motor de  $\frac{1}{4}$  de HP para las pruebas realizadas, se recomienda utilizar el módulo para manejar motores de mayor potencia, dependiendo de las limitaciones del variador de frecuencia que presente para incrementar la eficiencia de este proyecto.
- Se recomienda que se elaboren más proyectos de este tipo, permitiendo esto, generar tecnologías propias y económicas de uso múltiple para las industrias o empresas.

- Como parte de nuestra formación profesional es dar soluciones a los problemas en la industria, recomendamos mayor acercamiento del estudiante a este medio, realizando giras técnicas serias con las que se pueda diseñar o dar soluciones concretas a los problemas puntuales existentes en la industria y así no recurrir al profesional extranjero para realizar tareas con lo que cuenta el país.

## GLOSARIO

**A / D:** Análogo / Digital.

**ADC:** Conversor Analógico Digital.

**Amplificador Operacional:** Amplificador de corriente continua de alta ganancia, que constituye la base de los controladores electrónicos.

**Bit:** Unidad de información o dígito binario, mínima información digital.

**C:** Lenguaje de programación de nivel medio.

**C.I.:** Circuito Integrado.

**CONTROL LOCAL:** Control ubicado en un lugar diferente al variador de frecuencia.

**CONTROL REMOTO:** Control ubicado en un lugar diferente al variador de frecuencia.

**Convertidor:** Instrumento que recibe una señal estándar y la envía modificada en otra de tipo de señal que también es estándar.

**DAC:** Convertidor Digital Analógico.

**D/A:** Digital / Analógico.

**E/S:** Entrada / Salida.

**Frecuencia:** Número de ciclos por unidad de tiempo, su cantidad de medida es el hertz.

**HMI:** Interfase Hombre Máquina.

**I/O:** Entradas Salidas.

**LABVIEW:** Software de programación gráfica.

**MODEM:** Modulador demodulador, interfaz análoga digital

**PLC:** Controlador Lógico Programable.

**Proceso:** Conjunto de instrucciones o funciones realizadas por el equipo, en el cuál es controlada una variable, sin incluir los equipos de control.

**RAM:** Memoria de Acceso Randómico, memoria del tipo no permanente.

**ROM:** Memoria Solamente de Lectura, memoria del tipo permanente.

**Ruido:** Cualquier perturbación eléctrica o señal accidental no deseada, que modifica los datos deseados.

**SCADA:** Sistema de adquisición de datos control y supervisión.

**Señal:** Salida o información que emana un instrumento. Información representativa de un valor cuantificado.

**Señal de Salida Analógica:** Señal de salida del instrumento que es una función continua de la variable medida.

**Señal de Salida Digital:** Señal de salida que representa la magnitud de la variable medida en forma de una serie de cantidades discretas codificadas en un sistema de notación.

# ANEXOS

# **Anexo I**

**Manual del Usuario para el Control Local y Remoto del Variador  
de Frecuencia ABB**

Para describir el uso de este control se dividirá en dos partes, la primera el manejo del Control Local y la segunda el Control Remoto:

### CONTROL LOCAL

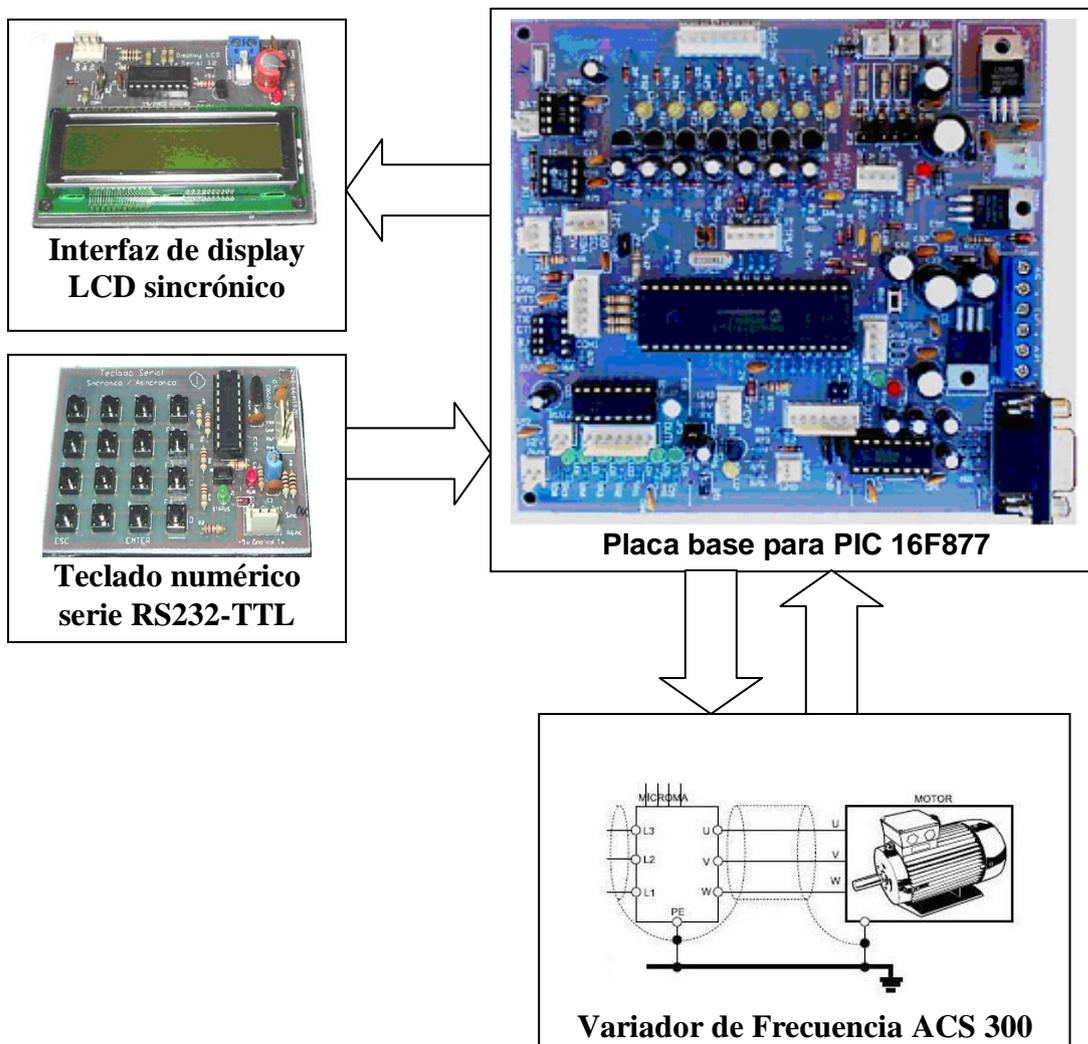


Figura 1. Control Local

En el siguiente diagrama (Fig.2) se detalla la secuencia de ingreso por teclado a las diferentes opciones del Sistema Local:

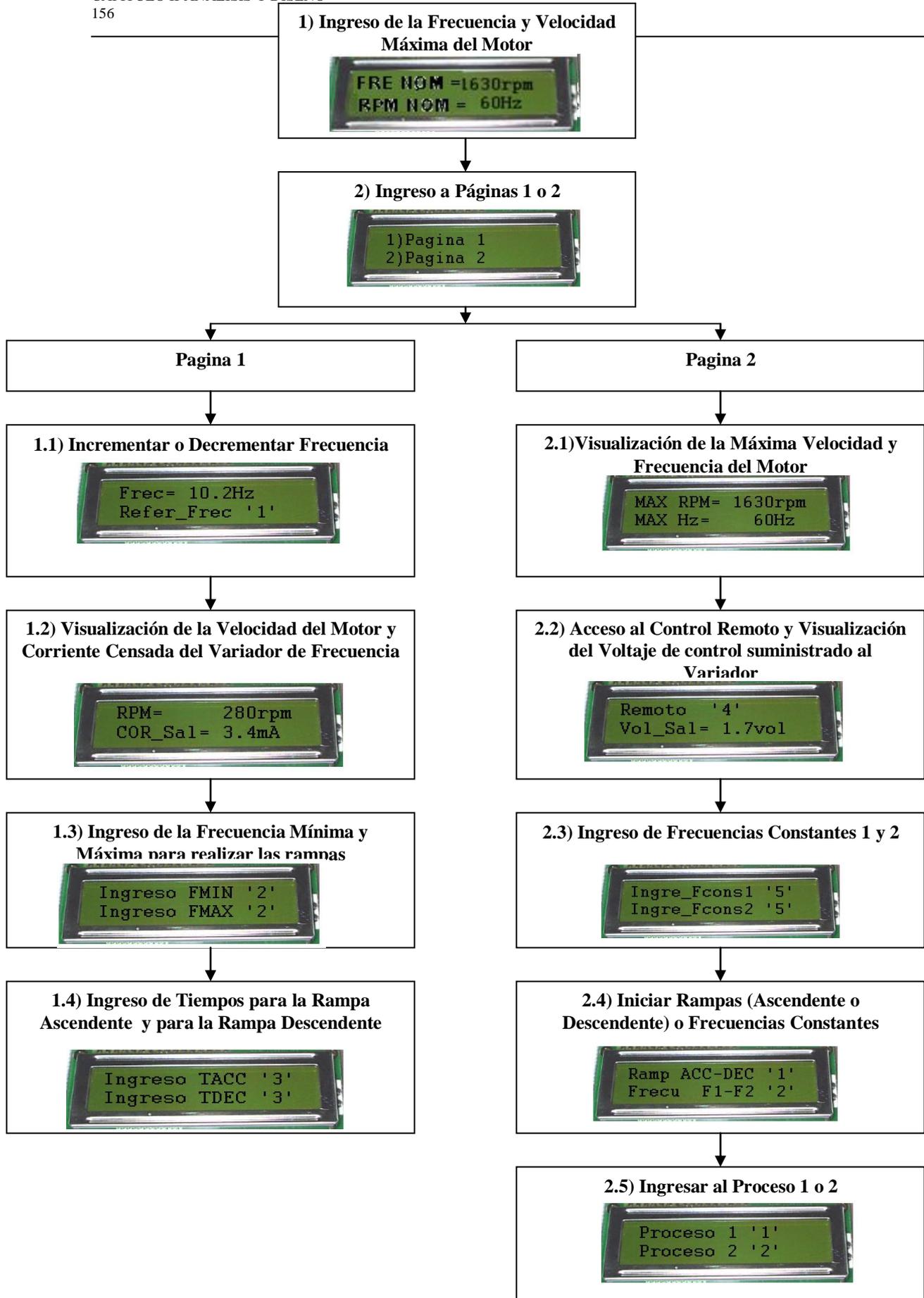


Figura 2. Secuencia de ingreso por Teclado

A continuación se explicará más detalladamente cada una de las opciones del Control Local:

El Teclado consta de los siguientes botones de control:

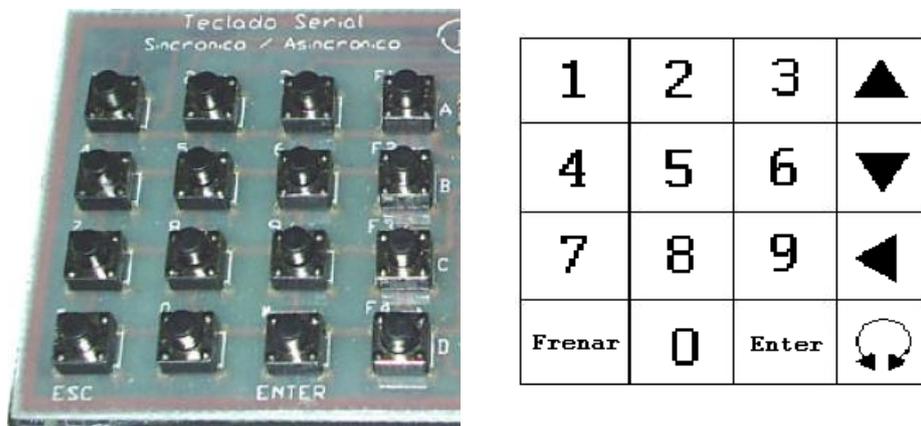


Figura 3. Botones de control

**1** (0,1,2...9) Teclas numéricas utilizadas para selección de submenús de cada página, ingreso de frecuencias máximas (FMAX),mínimas (FMIN), Tiempos para rampas Ascendentes (TACC), Descendentes (TDCC) y Variables de los Procesos.

▲ Desplazamiento hacia arriba para visualizar las diferentes opciones de cada página, Incrementar la Frecuencia.

▼ Desplazamiento hacia abajo para visualizar las diferentes opciones de cada página, Decrementar la Frecuencia .

◀ Regresar o Atrás, presionar este botón para salir de los submenús como por ejemplo de las rampas, de las frecuencias constantes, del control remoto o de los procesos.



Botón para finalizar el ingreso de un valor como por ejemplo FMAX, FMIN, TACC, etc.



Presionar para Invertir el giro del motor.



Presionar para Frenar el motor.

Después de ingresar la frecuencia y velocidad máxima del motor a utilizar, se visualiza en el LCD las dos páginas disponibles:



**Figura 4. Páginas**

El acceso a cada página se realiza por el teclado numérico (Presionar 1 para página 1 o 2 para página 2).

Cada página tiene opciones diferentes con relación a la otra.

En la página 1 tenemos:

5. Incrementar o Decrementar Frecuencia.
6. Visualización de la Velocidad del Motor y Corriente censada del Variador de Frecuencia.
7. Ingreso de la Frecuencia Mínima y Máxima para realizar las rampas.
8. Ingreso de Tiempos para la Rampa Ascendente y para la Rampa Descendente.

En la página 2 tenemos:

6. Visualización de la Máxima Velocidad y Frecuencia del Motor.

7. Acceso al Control Remoto y Visualización del Voltaje de control suministrado al Variador.
8. Ingreso de Frecuencias Constantes 1 y 2.
9. Iniciar Rampas (Ascendente o Descendente) o Frecuencias Constantes.
10. Ingresar al Proceso 1 o 2.

Algunas de las opciones de cada página tienen submenús para el ingreso de datos, como frecuencias, tiempos, variables de Procesos, etc.

A continuación se explica cada submenú:

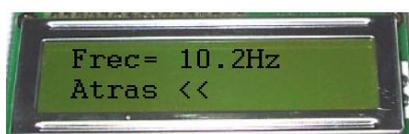
## **Página 1:**

### **1.1) Incrementar o Decrementar Frecuencia :**



**Figura 5. Incrementar o Decrementar Frecuencia**

Al ingresar en esta opción (presionando botón 1) aparece la siguiente pantalla:



**Figura 6. Submenú 1.1.1**

1.1.1) En este submenú podemos incrementar, decrementar y visualizar la frecuencia del Variador; si deseamos salir de este submenú presionamos el botón regresar.

### **1.2) Visualización de la Velocidad del Motor y Corriente censada del Variador de Frecuencia.**



**Figura 7. Visualización de la Velocidad del Motor y Corriente**

Esta opción solo nos permite visualizar los parámetros indicados.

### 1.3) Ingreso de la Frecuencia Mínima y Máxima para realizar las rampas.



Figura 8. Ingreso de la Frecuencia Mínima y Máxima para realizar las rampas

Al ingresar en esta opción (presionando botón 2) nos presenta la siguiente pantalla:

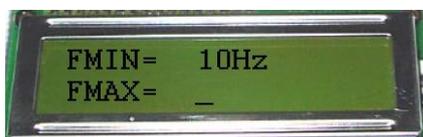


Figura 9. Submenú 1.3.1

1.3.1) Este submenú nos permite ingresar la frecuencia mínima (FMIN) y máxima (FMAX) de la rampa que deseemos generar; estos valores servirán para la rampa ascendente y descendente.

### 1.4) Ingreso de Tiempos para la Rampa Ascendente y para la Rampa Descendente



Figura 10. Ingreso de Tiempos para las Rampas

Al ingresar en esta opción (presionando botón 3) nos presenta la siguiente pantalla:

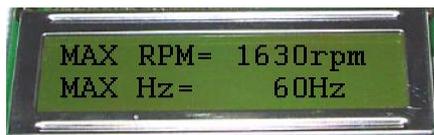


Figura 11. Submenú 1.4.1

1.4.1) Este submenú nos permite ingresar el tiempo que va a durar la rampa ascendente (TACC) y descendente (TDEC).

## **Página 2:**

### **2.1) Visualización de la Máxima Velocidad y Frecuencia del Motor**



**Figura 12. Visualización de la Máxima Velocidad y Frecuencia del Motor**

Esta opción solo nos permite visualizar los parámetros indicados.

### **2.2) Acceso al Control Remoto y Visualización del Voltaje de control suministrado al Variador.**



**Figura 13. Acceso al Control Remoto y Visualización del Voltaje de control**

En esta pantalla podemos visualizar el voltaje de control (0-10Vdc) aplicado al Variador de Frecuencia o ingresar al Control Remoto (presionando botón 4), en donde se presenta la siguiente pantalla:



**Figura 14. Submenú 2.2.1**

2.2.1) Este submenú nos permite Iniciar (presionando botón 1) el Control Remoto o Regresar a la Página 2 (presionando botón Regresar).

Al Iniciar el Control Remoto aparecerá la pantalla siguiente,



**Figura 15. Submenú 2.2.1.1**

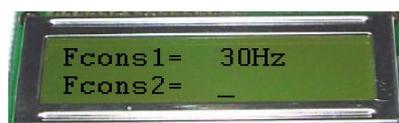
en la cual podemos visualizar las variaciones de Frecuencia.

### **2.3) Ingreso de Frecuencias Constantes 1 y 2.**



**Figura 16. Ingreso de Frecuencias Constantes 1 y 2**

Ingresando a esta opción (presionando botón 5) nos presenta la siguiente pantalla:



**Figura 17. Submenú 2.3.1**

2.3.1) Este submenú nos permite ingresar la frecuencias constantes1 y 2 (Fcons1 y Fcons2).

### **2.4) Iniciar Rampas (Ascendente o Descendente) o Frecuencias Constantes**



**Figura 18. Iniciar Rampas o Frecuencias Constantes**

Esta pantalla nos presenta dos opciones, ingresar a generar las rampas (presionando botón 1) o establecer frecuencias constantes (presionando botón 2).

2.4.1) Al ingresar a generar las rampas se muestra la siguiente pantalla:



**Figura 19. Submenú 2.4.1**

Este submenú nos permite generar la Rampa Ascendente (presionando botón 1) o Descendente (presionando botón 2).

En cualquiera de las rampas se presenta la pantalla que se indico en la Figura 14. y a continuación la pantalla de la Figura 15.

2.4.2) Al ingresar a establecer frecuencias constantes aparecerá la siguiente pantalla:



**Figura 20. Submenú 2.4.2**

El submenú nos permite fijar a una frecuencia constante 1 (presionando botón 1) o a una frecuencia constante 2 (presionando botón 2).

Al presionar cualquiera de las dos opciones se mostrará la pantalla que se indico en la Figura 14. y a continuación la pantalla de la Figura 15.

## **2.5) Ingresar al Proceso 1 o 2**



**Figura 21. Ingresar al Proceso 1 o 2**

Esta pantalla nos presenta dos opciones, ingresar al Proceso1 (presionando botón 1) o al Proceso 2 (presionando botón 2).

Al ingresar al **Proceso 1** se muestran las siguientes pantallas:

### 2.5.1) Variables 1 y 2



**Figura 22. Submenú 2.5.1**

Si deseamos ingresar a este submenú presionamos el botón 2 caso contrario presionamos los botones de desplazamiento para visualizar las demás variables.

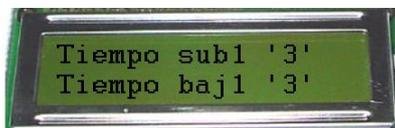
Este submenú nos permite ingresar dos variables, la Frecuencia Inicial 1 y la Frecuencia Final 1; Estas variables se utilizarán en la ejecución o secuencia del Proceso1. Lo mismo sucede con las siguientes variables del Proceso 1:

### 2.5.2) Variables 3 y 4



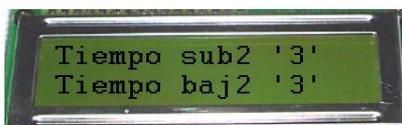
**Figura 23. Submenú 2.5.2**

### 2.5.3) Variables 5 y 6



**Figura 24. Submenú 2.5.3**

### 2.5.4) Variables 7 y 8



**Figura 25. Submenú 2.5.4**

### 2.5.5) Variables 9 y 10



Figura 26. Submenú 2.5.5

Se ingresará a cada variable presionando el botón indicado por ejemplo '3'. El ingreso de datos se lo realizará como se ha explicado en pantallas anteriores.

2.5.6) Comenzar o ejecutar el Proceso 1



Figura 27. Comenzar o ejecutar el Proceso 1

Esta pantalla nos permite comenzar o ejecutar el proceso1 (presionando botón 4) o salir (presionando botón 5).

El Proceso 1 tiene la siguiente secuencia de operación:}

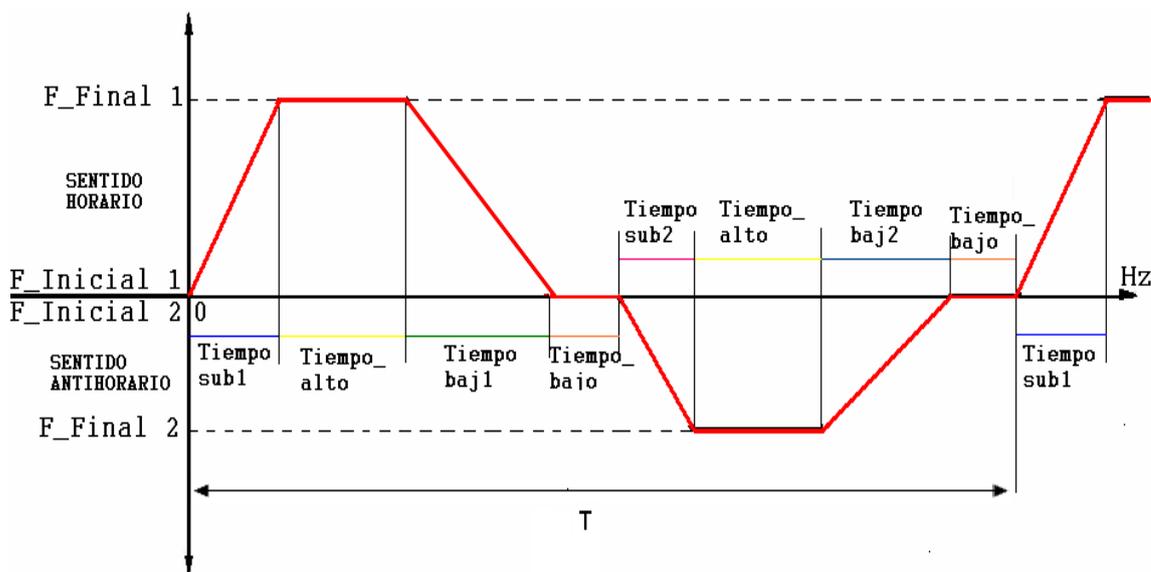


Figura 28. Secuencia de operación del Proceso 1

VARIABLE	DEFAULT
F_Inicial 1	1 Hz
F_Final 1	40 Hz
F_Inicial 2	1 Hz
F_Final 2	30 Hz
Tiempo sub1	20 seg
Tiempo baj1	15 seg
Tiempo sub2	10 seg
Tiempo baj2	5 seg
Tiempo_alto	10 seg
Tiempo_bajo	20 seg

Tabla 1. Variables del Proceso 1

Al ingresar al **Proceso 2** se muestran pantallas similares a las indicadas en el Proceso 1; el ingreso de datos y visualización es el mismo.

El Proceso 2 tiene la siguiente secuencia de operación:

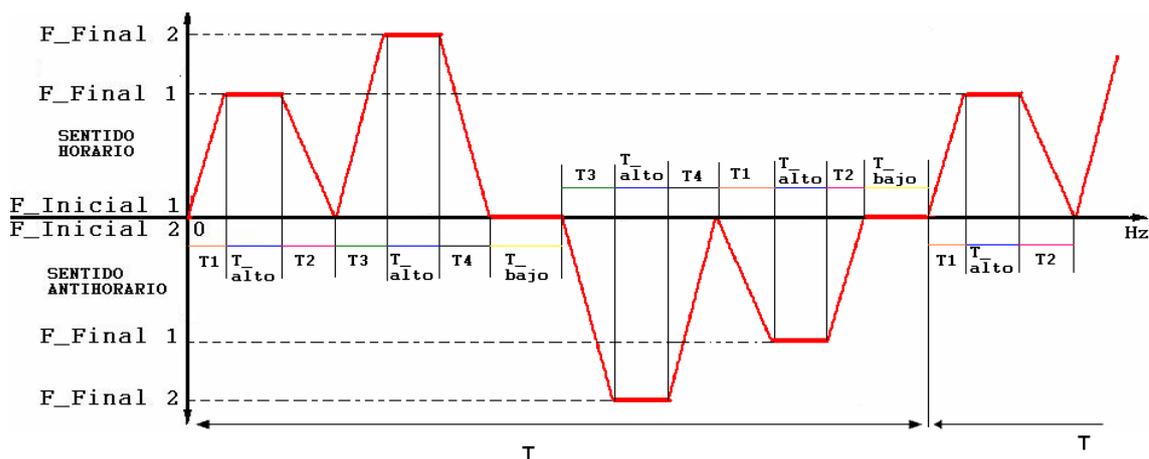


Figura 29. Secuencia de operación del Proceso 2

VARIABLE	DEFAULT
F_Inicial 1	1 Hz
F_Final 1	30 Hz
F_Inicial 2	1 Hz
F_Final 2	40 Hz
T1	20 seg
T2	15 seg
T3	10 seg
T4	5 seg
T_alto	10 seg
T_bajo	20 seg

Tabla 1. Variables del Proceso 1

## CONTROL REMOTO

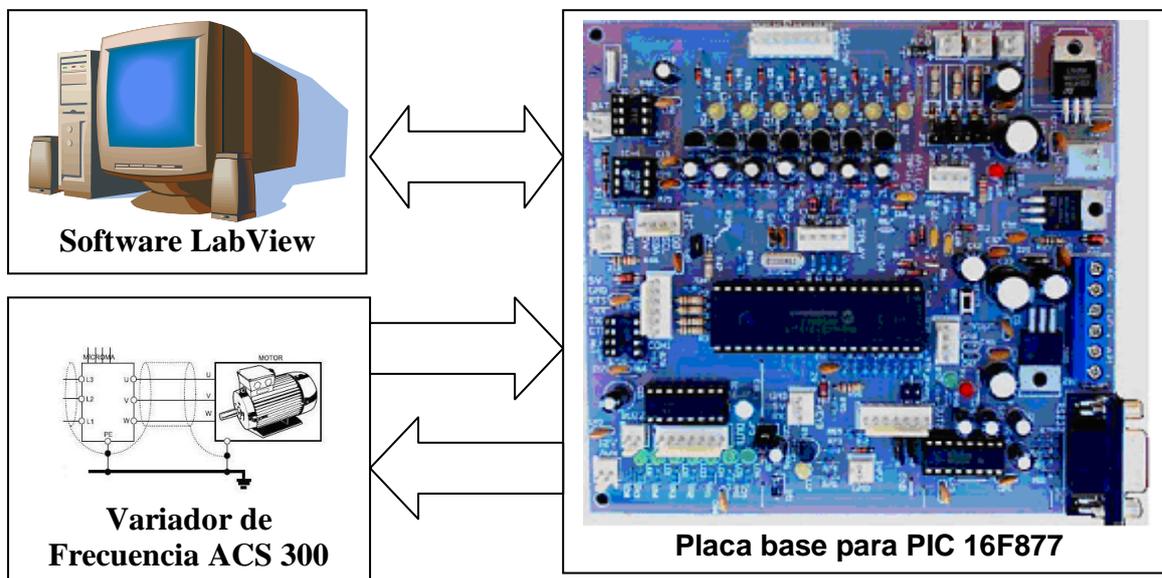


Figura 30. Control Remoto

El Control Remoto se realizó en la plataforma de Desarrollo Labview el cual nos da una gran versatilidad para la visualización y el control del motor.

El uso del control Remoto es mucho más sencillo que el Control Local; a continuación indicamos en orden las ventanas que posee el programa desarrollado para el Control Remoto:

8. Ingreso de Password
9. Ingreso de Frecuencia y Velocidad Máxima
10. Menú Visualización – Control – Proceso
11. Visualización
12. Visualización y Control
13. Proceso
14. Comparar Gráficas

### 1) Ingreso de Password

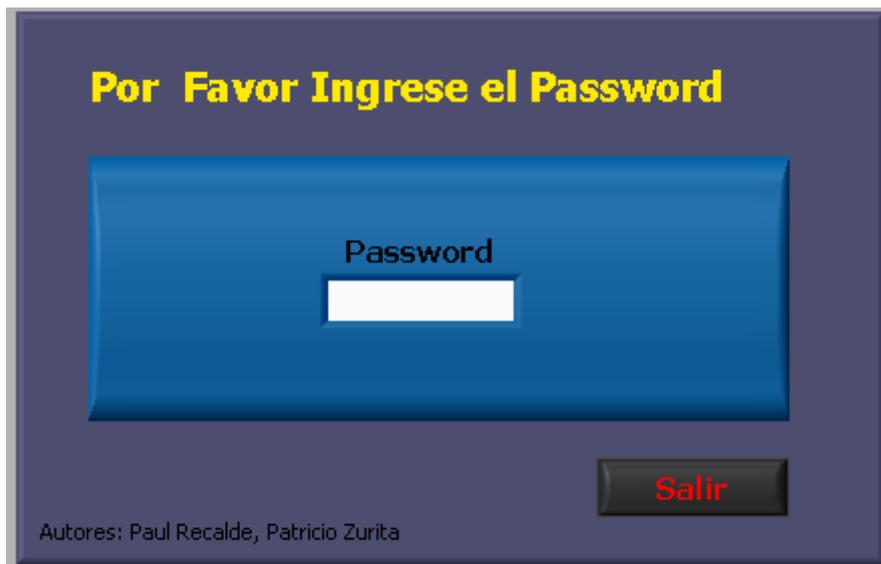


Figura 31. Pantalla para clave de ingreso

Esta es la primera pantalla que se muestra al ejecutar el programa o software; en la cual ingresamos nuestra clave para acceder a dicho programa de Control Remoto.

## 2) Ingreso de Frecuencia y Velocidad Máxima

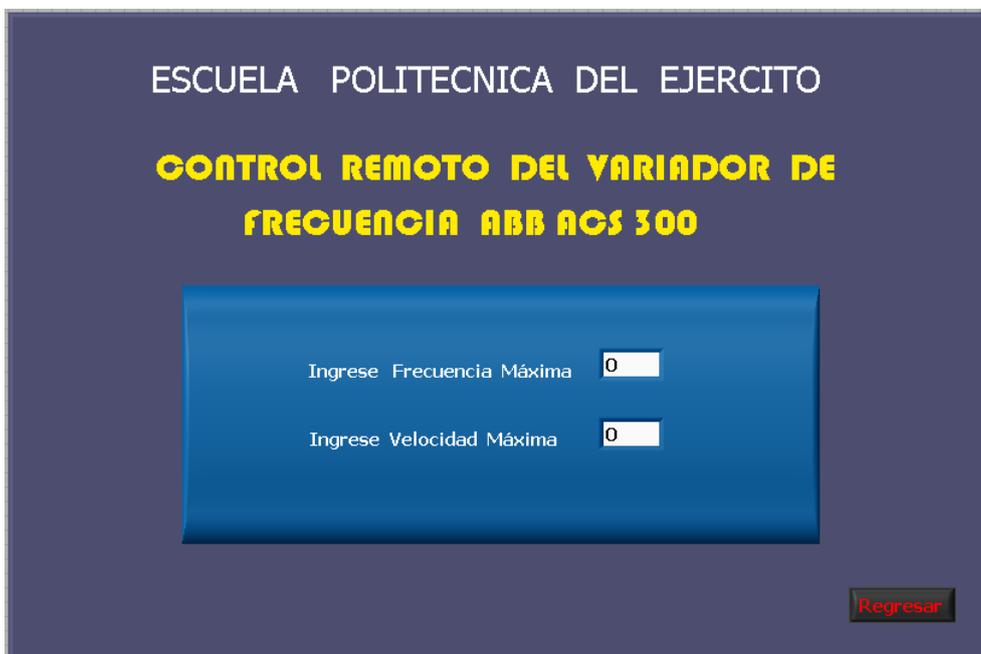


Figura 32. Pantalla para ingreso de

Frecuencia y Velocidad Máxima

En esta ventana ingresamos la Frecuencia y Velocidad Máxima del motor a utilizarse.

### 3) Menú Visualización – Control – Proceso



Figura 33. Pantalla Menú: Visualización – Control – Proceso

La ventana nos presenta un menú con tres opciones que son Visualización, Control + Visualización y Proceso; podemos acceder a cualquiera de ellas mediante un clic en el botón correspondiente.

### 4) VISUALIZACIÓN.

En la figura 34, se muestra la ventana nos permite la visualización de variables que se modifican en el Control Local como la Frecuencia que suministra el Variador, la Velocidad del Motor, el voltaje de control (0-10Vdc) aplicado al Variador, y la corriente censada del Variador de Frecuencia.

También nos permite la visualización de curvas de Frecuencia con respecto al Tiempo real, guardar graficas o curvas para luego ser comparadas y analizadas, y Guardar Registros.



Figura 34. Pantalla Visualización

### 5) Visualización y Control



Figura 35. Pantalla Control

Esta ventana tiene las mismas características u opciones que la ventana de Visualización, pero además nos permite controlar el Variador de Frecuencia.

Podemos Incrementar o Decrementar la frecuencia que suministre el Variador al motor, generar Rampas Ascendentes o Descendentes y fijar Frecuencias constantes todo esto con las características de visualización de variables y curvas.

## 6) Comparar Gráficas

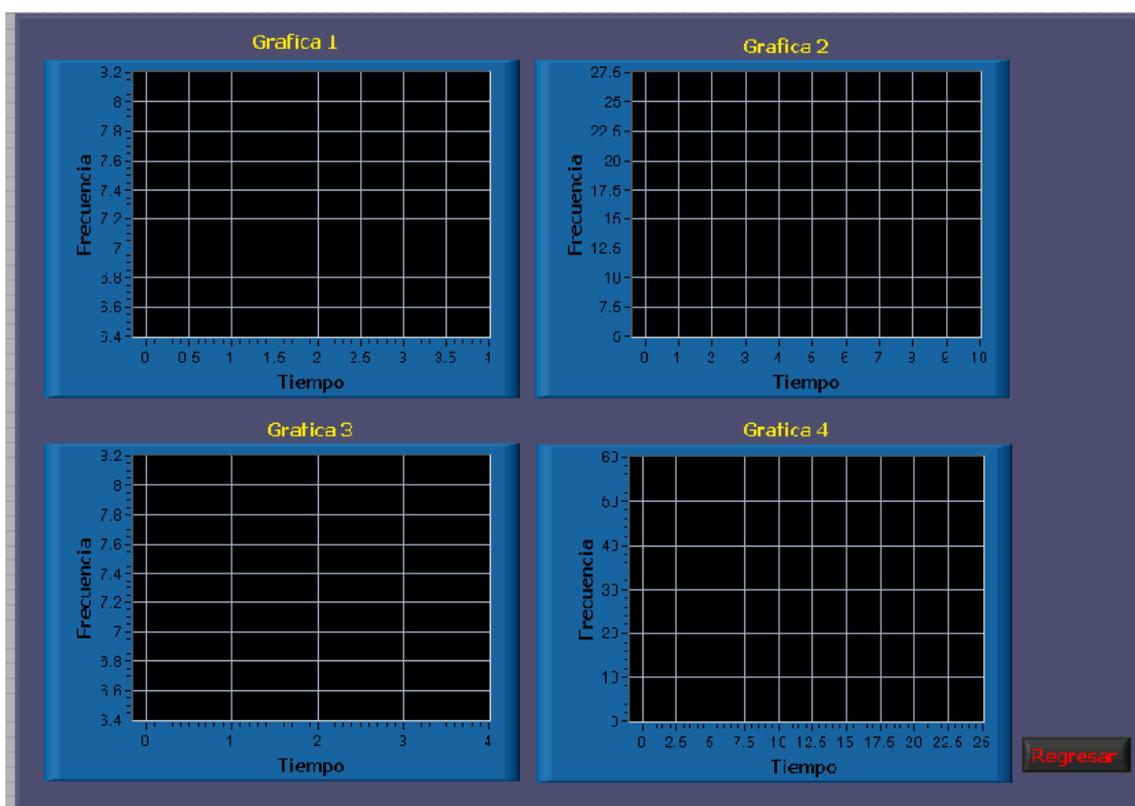


Figura 37. Pantalla Comparar Gráficas

En esta pantalla podemos comparar las últimas 4 gráficas que se han guardado para su comparación y análisis.

## Diagrama de Conexiones

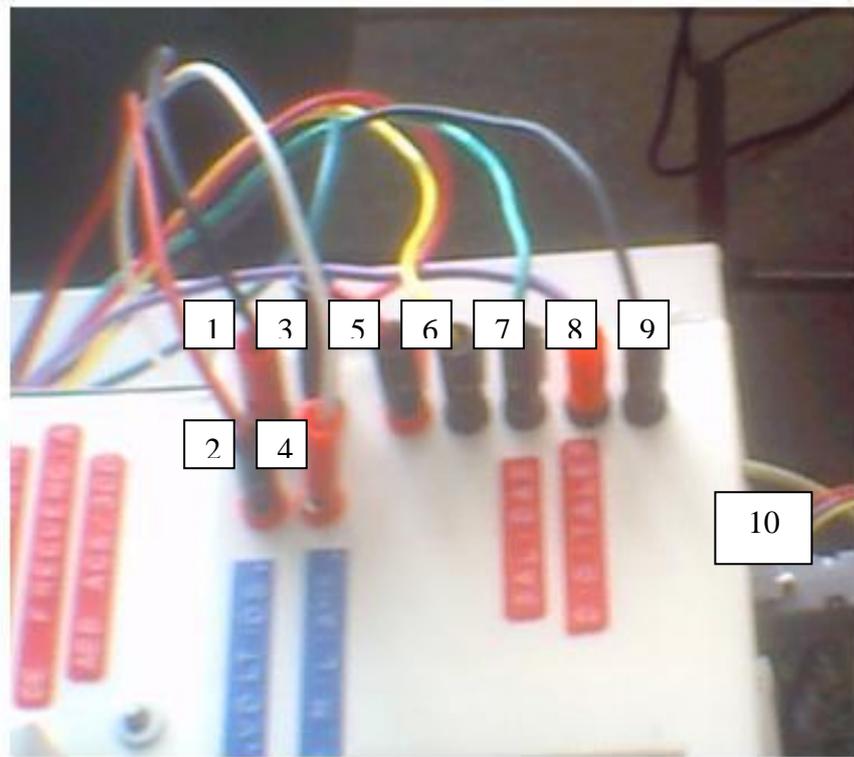


Figura 38. Diagrama de Conexiones en el Módulo

Como se observa en figura 38, para las conexiones del variador de frecuencia hasta el módulo se debe tener en cuenta las siguientes consideraciones:

- 1 (Cable Rojo) Y 2 (Cable Azul); Salida Analógica de Voltaje de 0 a 10 V hacia el Variador de Frecuencia (Cables fijos en el variador de frecuencia).
- 3 (Cable Verde) Y 4 (Cable Blanco); Entrada Analógica de Corriente de 4 a 20 mA (Cables fijos en el variador de frecuencia).
- 5 (Cable Rojo); Entrada de Voltaje de 24 Vdc, cable proveniente del Variador de Frecuencia.
- 6 (Cable Amarillo); Salida de 1 lógico (24Vdc), indica inicio del control local; 0 Lógico (0Vdc), apaga control local, cable proveniente del Variador de Frecuencia.

7 (Cable Celeste); Salida de inversión de giro, cable proveniente del Variador de Frecuencia.

8 (Cable Lila) Y 9 (Cable Negro); Frecuencias constantes 1 y 2 respectivamente, cables provenientes del Variador de Frecuencia.

10 Entrada RS-232, para comunicación entre la PC y el Módulo.

**NOTA:** Las conexiones de alimentación del Variador de Frecuencia, el Motor se las debe realizar como indique el profesor guía.

## **Anexo II**

### **Librerías para Programación de Interfases**

## 16F877.h

```
#device PIC16F877
```

```
#nolist
```

```
#define PIN_A0 40
```

```
#define PIN_A1 41
```

```
#define PIN_A2 42
```

```
#define PIN_A3 43
```

```
#define PIN_A4 44
```

```
#define PIN_A5 45
```

```
#define PIN_B0 48
```

```
#define PIN_B1 49
```

```
#define PIN_B2 50
```

```
#define PIN_B3 51
```

```
#define PIN_B4 52
```

```
#define PIN_B5 53
```

```
#define PIN_B6 54
```

```
#define PIN_B7 55
```

```
#define PIN_C0 56
```

```
#define PIN_C1 57
```

```
#define PIN_C2 58
```

```
#define PIN_C3 59
```

```
#define PIN_C4 60
```

```
#define PIN_C5 61
```

```
#define PIN_C6 62
```

```
#define PIN_C7 63
```

```
#define PIN_D0 64
```

```
#define PIN_D1 65
```

```
#define PIN_D2 66
```

```
#define PIN_D3 67
#define PIN_D4 68
#define PIN_D5 69
#define PIN_D6 70
#define PIN_D7 71
```

```
#define PIN_E0 72
#define PIN_E1 73
#define PIN_E2 74
```

```
//////////////////////////////////// Useful defines
```

```
#define FALSE 0
#define TRUE 1
```

```
#define BYTE int
#define BOOLEAN short int
```

```
#define getc getch
#define fgetc getch
#define getchar getch
#define putc putchar
#define fputc putchar
#define fgets gets
#define fputs puts
```

```
//////////////////////////////////// Control
```

```
// Control Functions: RESET_CPU(), SLEEP(), RESTART_CAUSE()
// Constants returned from RESTART_CAUSE() are:
```

```
#define WDT_FROM_SLEEP 0
#define WDT_TIMEOUT 8
#define MCLR_FROM_SLEEP 16
#define NORMAL_POWER_UP 24
```

```
//////////////////////////////////// Timer 0
// Timer 0 (AKA RTCC)Functions: SETUP_COUNTERS() or SETUP_TIMER0(),
// SET_TIMER0() or SET_RTCC(),
// GET_TIMER0() or GET_RTCC()
// Constants used for SETUP_TIMER0() are:
#define RTCC_INTERNAL 0
#define RTCC_EXT_L_TO_H 32
#define RTCC_EXT_H_TO_L 48

#define RTCC_DIV_1 8
#define RTCC_DIV_2 0
#define RTCC_DIV_4 1
#define RTCC_DIV_8 2
#define RTCC_DIV_16 3
#define RTCC_DIV_32 4
#define RTCC_DIV_64 5
#define RTCC_DIV_128 6
#define RTCC_DIV_256 7

#define RTCC_8_BIT 0

// Constants used for SETUP_COUNTERS() are the above
// constants for the 1st param and the following for
// the 2nd param:

//////////////////////////////////// WDT
// Watch Dog Timer Functions: SETUP_WDT() or SETUP_COUNTERS()
// RESTART_WDT()

#define WDT_18MS 8
#define WDT_36MS 9
#define WDT_72MS 10
#define WDT_144MS 11
```

```
#define WDT_288MS    12
#define WDT_576MS    13
#define WDT_1152MS   14
#define WDT_2304MS   15
```

```
//////////////////////////////////// Timer 1
```

```
// Timer 1 Functions: SETUP_TIMER_1, GET_TIMER1, SET_TIMER1
```

```
// Constants used for SETUP_TIMER_1() are:
```

```
#define T1_DISABLED          0
#define T1_INTERNAL          0x85
#define T1_EXTERNAL          0x87
#define T1_EXTERNAL_SYNC     0x83
```

```
#define T1_CLK_OUT          8
```

```
#define T1_DIV_BY_1          0
#define T1_DIV_BY_2          0x10
#define T1_DIV_BY_4          0x20
#define T1_DIV_BY_8          0x30
```

```
//////////////////////////////////// Timer 2
```

```
// Timer 2 Functions: SETUP_TIMER_2, GET_TIMER2, SET_TIMER2
```

```
// Constants used for SETUP_TIMER_2() are:
```

```
#define T2_DISABLED          0
#define T2_DIV_BY_1          4
#define T2_DIV_BY_4          5
#define T2_DIV_BY_16         6
```

```
//////////////////////////////////// CCP
```

```
// CCP Functions: SETUP_CCPx, SET_PWMx_DUTY
```

```
// CCP Variables: CCP_x, CCP_x_LOW, CCP_x_HIGH
```

```
// Constants used for SETUP_CCPx() are:
```

```
#define CCP_OFF          0
#define CCP_CAPTURE_FE          4
#define CCP_CAPTURE_RE          5
#define CCP_CAPTURE_DIV_4        6
#define CCP_CAPTURE_DIV_16       7
#define CCP_COMPARE_SET_ON_MATCH 8
#define CCP_COMPARE_CLR_ON_MATCH 9
#define CCP_COMPARE_INT          0xA
#define CCP_COMPARE_RESET_TIMER  0xB
#define CCP_PWM                  0xC
#define CCP_PWM_PLUS_1           0x1C
#define CCP_PWM_PLUS_2           0x2C
#define CCP_PWM_PLUS_3           0x3C
long CCP_1;
#define CCP_1 = 0x15
#define CCP_1_LOW= 0x15
#define CCP_1_HIGH= 0x16
long CCP_2;
#define CCP_2 = 0x1B
#define CCP_2_LOW= 0x1B
#define CCP_2_HIGH= 0x1C

////////////////////////////////////// PSP
// PSP Functions: SETUP_PSP, PSP_INPUT_FULL(), PSP_OUTPUT_FULL(),
// PSP_OVERFLOW(), INPUT_D(), OUTPUT_D()
// PSP Variables: PSP_DATA

// Constants used in SETUP_PSP() are:

#define PSP_ENABLED          0x10
#define PSP_DISABLED        0

#define PSP_DATA=          8
```

```
////////////////////////////////////// SPI
// SPI Functions: SETUP_SPI, SPI_WRITE, SPI_READ, SPI_DATA_IN
// Constants used in SETUP_SSP() are:
#define SPI_MASTER          0x20
#define SPI_SLAVE          0x24
#define SPI_L_TO_H         0
#define SPI_H_TO_L         0x10
#define SPI_CLK_DIV_4      0
#define SPI_CLK_DIV_16     1
#define SPI_CLK_DIV_64     2
#define SPI_CLK_T2         3
#define SPI_SS_DISABLED   1

#define SPI_SAMPLE_AT_END 0x8000
#define SPI_XMIT_L_TO_H   0x4000

////////////////////////////////////// UART
// Constants used in setup_uart() are:

// FALSE - Turn UART off
// TRUE  - Turn UART on
#define UART_ADDRESS 2
#define UART_DATA    4
// TRUE  - Turn UART on

////////////////////////////////////// ADC
// ADC Functions: SETUP_ADC(), SETUP_ADC_PORTS() (aka
SETUP_PORT_A),
// SET_ADC_CHANNEL(), READ_ADC()

// Constants used in SETUP_ADC_PORTS() are:

#define NO_ANALOGS          0x86    // None
```

---

```
#define ALL_ANALOG                0x80    // A0 A1 A2 A3 A5 E0
#define ANALOG_RA3_REF            0x81    // A0 A1 A2 A5 E0 E1
#define A_ANALOG                  0x82    // A0 A1 A2 A3 A5
#define A_ANALOG_RA3_REF          0x83    // A0 A1 A2 A5
#define RA0_RA1_RA3_ANALOG        0x84    // A0 A1 A3 Ref=Vdd
#define RA0_RA1_ANALOG_RA3_REF    0x85    // A0 A1 Ref=A3
#define ANALOG_RA3_RA2_REF        0x88    // A0 A1 A5 E0 E1 E2
#define ANALOG_NOT_RE1_RE2        0x89    // A0 A1 A2 A3 A5 E0
#define A_ANALOG_RA3_RA2_REF      0x8C    // A0 A1 A5 Ref=A2,A3
#define RA0_ANALOG                0x8E    // A0
#define RA0_ANALOG_RA3_RA2_REF    0x8F    // A0 Ref=A2,A3

// Constants used for SETUP_ADC() are:
#define ADC_OFF                    0        // ADC Off
#define ADC_CLOCK_DIV_2            1
#define ADC_CLOCK_DIV_8            0x41
#define ADC_CLOCK_DIV_32           0x81
#define ADC_CLOCK_INTERNAL          0xc1    // Internal 2-6us

// Constants used in READ_ADC() are:
#define ADC_START_AND_READ          7
#define ADC_START_ONLY              1
#define ADC_READ_ONLY               6

////////////////////////////////////// INT
// Interrupt Functions: ENABLE_INTERRUPTS(), DISABLE_INTERRUPTS(),
// EXT_INT_EDGE()
// Constants used in EXT_INT_EDGE() are:

#define L_TO_H                      0x40
#define H_TO_L                      0

// Constants used in ENABLE/DISABLE_INTERRUPTS() are:

#define GLOBAL                       0x0BC0
```

```
#define INT_RTCC          0x0B20
#define INT_RB           0x0B08
#define INT_EXT          0x0B10
#define INT_AD           0x8C40
#define INT_TBE         0x8C10
#define INT_RDA         0x8C20
#define INT_TIMER1      0x8C01
#define INT_TIMER2      0x8C02
#define INT_CCP1        0x8C04
#define INT_CCP2        0x8D01
#define INT_SSP         0x8C08
#define INT_PSP         0x8C80
#define INT_BUSCOL      0x8D08
#define INT_EEPROM      0x8D10
#define INT_TIMER0      0x0B20
```

## #list Teclado Serie 4x4 salida RS232

```
//Configuración general
#include <16F84.h>
#fuses XT,NOWDT,NOPROTECT,PUT
#use delay(clock=4000000)
#use rs232(baud=9600, xmit=PIN_B5)
#use fast_io(A)
#use fast_io(B)

//Definiciones del hardware
#define led_on    output_low(PIN_A4)
#define led_off  output_high(PIN_A4)
#define valid_on  output_high(PIN_B4)
#define valid_off output_low(PIN_B4)
#define pw       70 //Ancho de pulso en ms del Valid data antes de enviar el
caracter

#define enter    13 //codigo del enter (o poner #)

/*
Hardware del Interfaz 4x4 de teclado

Uso del puerto

TX: Ra0/Ra1/Ra2/Ra3: filas  0123 Salidas
RX: Rb0/Rb1/Rb2/Rb3: columnas 0123 Entradas}
Ra4: led de status activo en alto
Rb4: Valid
Rb5: TXdata
```

## Rb6 y Rb7 programador ICP

```
 /-----Tx -----\  
 Ra0 Ra1 Ra2 Ra3  
 c1 c2 c3 c4  
 1 2 3 A f1 Rb0  
 4 5 6 B f2 Rb1 -> |RX  
 7 8 9 C f3 Rb2  
 * 0 # D f4 Rb3  
*/  
  
main()  
{  
  char dato,fi,fil,col,acu,aux,asc;  
  
  // Definición de puertos  
  set_tris_a (0b00000);  
  set_tris_b (0b11001111);  
  
  led_off;  
  valid_off;  
  
  while (TRUE)  
  {  
  
    delay_ms(50);  
  
    asc='X';  
    fi=1;  
  
    //Barro transmitiendo por columnas de C1 (Ra0) a C4(Ra3)  
  
    for (col=1;col<=4;col++)
```

```
{

    output_a(fi+16); //El 16 para que mantenga el Ra4=1 (off)
                    //Activo linea de columna

    fi=fi<<1;

    //Leo las FILAS (portb) y lo cargo en Aux

    aux=(input_b() & 0b00001111); // sólo me interesa la parte baja

    if (aux!=0) // Si leo algo en las filas es que alguien pulso una tecla
    {

        fil=aux;      // es la 1 o 2 fila
        if (aux==4) fil=3; // es la tercera?
        if (aux==8) fil=4; // es la cuarta?
        dato=(fil-1)*3 + col; //calcula la formula
        asc='0'+dato;    // deajo el numero en ascii

        if (dato==10) asc='*'; // es el asterisco?
        if (dato==11) asc='0'; // es el cero?
        if (dato==12) asc=enter; // es el numeral?

        if (col==4) asc='@' + fil; // Flechas ?

        led_on;
        valid_on;
        delay_ms(pw);
        valid_off;
        delay_ms(1);
        putc(asc);
        led_off;
```

```
delay_ms(70); //Retardo antes de comenzar repeticion de caracteres
```

```
if (input_b() & 0b00001111 != 0) delay_ms(250);
```

```
while (input_b() & 0b00001111 != 0)
```

```
{
```

```
    led_on;
```

```
    valid_on;
```

```
    delay_ms(pw);
```

```
    valid_off;
```

```
    delay_ms(1);
```

```
    putc(asc);
```

```
    led_off;
```

```
    delay_ms(70); //Retardo entre caracteres si mantengo tecla
```

pulsada

```
    }
```

```
  }
```

```
}
```

```
}
```

```
}
```

## lcdser\_plc.h

/\*

Libreria para tarjeta IT&T LCD Serial

Funciones disponibles

lcd\_cursor\_on();

lcd\_cursor\_off();

lcd\_ini();

lcd\_linea1(int8 offset);

lcd\_linea2(int8 offset);

lcd\_putc(char o string)

lcd\_clear();

printf. Ej printf(lcd\_putc,"Total= %d",total);

lcd\_luz\_on(); enciende luz de display

lcd\_luz\_off(); apaga luz de display

\*/

//Definiciones de CK, DATA y STROBE en plantilla

#define ck0 output\_low(PIN\_e0)

#define ck1 output\_high(PIN\_e0)

#define data0 output\_low(PIN\_e1)

#define data1 output\_high(PIN\_e1)

#define strobe1\_off output\_low(PIN\_e2)

#define strobe1\_on output\_high(PIN\_e2)

//Definiciones del hardware LCD serial

#define set\_inst bit\_clear(estado1,1)

#define set\_dato bit\_set(estado1,1)

#define set\_enable bit\_set(estado1,3)

#define set\_disable bit\_clear(estado1,3)

```
//Variables globales
```

```
int8 luz_lcd=0;
```

```
int8 estado1;
```

```
//Rutinas
```

```
txcomm1()
```

```
{
```

```
int8 i,e;
```

```
e=estado1 | luz_lcd;
```

```
strobe1_off;
```

```
ck0;
```

```
for (i=1;i<=8;i++)
```

```
{
```

```
data0;
```

```
if (bit_test(e,7)==1)
```

```
data1;
```

```
e=e<<1;
```

```
ck1;
```

```
delay_cycles(10);
```

```
ck0;
```

```
}
```

```
strobe1_on;
```

```
delay_cycles(10);
```

```
strobe1_off;
```

```
}
```

```
lcd_luz_on()
{
    estado1=0;
    luz_lcd=1;
    txcomm1();
}
```

```
lcd_luz_off()
{
    estado1=0;
    luz_lcd=0;
    txcomm1();
}
```

```
write_i(int8 x)
{
    estado1=0;
```

```
    //setear RS y Enable
```

```
    set_inst;
    set_enable;
    txcomm1();
```

```
    // Enviar primero parte alta de x
```

```
    estado1=x & 0b11110000; //Poner parte baja en cero
```

```
    //Cargo estado1 parte alta con dato XH
```

```
    set_inst;
    set_enable;

    txcomm1();
```

```
set_disable; //aqui graba  
txcomm1();
```

```
// Enviar segundo parte baja de x  
//setear RS y Enable
```

```
set_inst;  
set_enable;  
txcomm1();
```

```
estado1=swap(x) & 0b11110000;
```

```
set_inst;  
set_enable;  
txcomm1();
```

```
set_disable;  
txcomm1();
```

```
delay_ms(5);
```

```
}
```

```
lcd_putc(int8 x)
```

```
{  
estado1=0;
```

```
//setear RS y Enable
```

```
set_dato;  
set_enable;  
txcomm1();
```

```
// Enviar primero parte alta de x
estado1=x;
estado1=estado1 & 0b11110000; // ;Poner parte baja en cero

//Cargo estado1 parte alta con dato XH

set_dato; //Mantengo
set_enable; //Mantengo
txcomm1();

set_disable; //aqui graba
txcomm1();

// Enviar segundo parte baja de x

//setear RS y Enable

set_dato;
set_enable;
txcomm1();

estado1=swap(x) & 0b11110000; //;Poner parte baja en cero

set_dato;
set_enable;
txcomm1();

set_disable;
txcomm1();

delay_ms(5);

}
```

```
lcd_clear()
```

```
{  
write_i(0b00000001);  
}
```

```
lcd_set(char x)
```

```
{  
estado1=x;  
set_enable;  
txcomm1();
```

```
set_disable;  
txcomm1();
```

```
delay_ms(5);  
}
```

```
// Funciones de display
```

```
lcd_ini()
```

```
{
```

```
//;Funcion:
```

```
//;0 0 0 0 0 0 0 1 Clear Display
```

```
//;0 0 0 0 0 0 1 x Return Home
```

```
//;0 0 0 0 0 1 I/D S Sent.de desplaz. 1:izq 0:der / S=1 Shift ON
```

```
//;0 0 0 0 1 D C B Display,Cursor, Blinking= 0:Off 1:On
```

```
//;0 0 0 1 S/C R/L x x Shift= 1:Screen 0:Cursor Sentido= 1:Right 0:Left
```

```
//;0 0 1 DL N F x x DL= 0/1: 4/8 bits, N= 0/1: 1/2 lineas, F=0/1 5x7, 5x10
```

```
(font)
```

```
//;1 0 < offset > Ubica el cursor en la linea 1 + offset
```

```
//;1 1 < offset > Ubica el cursor en la linea 2 + offset
```

```
set_disable;  
delay_ms(100);  
//envio 03h 3 veces y luego 02h
```

```
lcd_set(0x30);  
lcd_set(0x30);  
lcd_set(0x30);  
lcd_set(0x20);
```

```
write_i(0x28); //4 bits 2 lineas  
write_i(0x08);  
write_i(0x01);  
write_i(0x06);  
write_i(0x0C);
```

```
}
```

```
lcd_linea1(int8 offset)
```

```
{
```

```
offset=0b10000000 + offset;  
write_i(offset);
```

```
}
```

```
lcd_linea2(int8 offset)
```

```
{
```

```
offset=0b11000000 + offset;  
write_i(offset);
```

```
}
```

```
lcd_cursor_on()
```

```
{
```

```
write_i(0b00001111);
```

```
}
```

```
lcd_cursor_off()
```

```
{
```

```
write_i(0b00001100);
```

```
}
```

```
//;0 0 0 0 1 D C B Display,Cursor, Blinking= 0:Off 1:On
```

## EEPROM

```
#ifndef EEPROM_SDA
#define EEPROM_SDA PIN_b4
#define EEPROM_SCL PIN_b5
#endif

#include i2c(master,sda=EEPROM_SDA, scl=EEPROM_SCL)
#define EEPROM_ADDRESS long int
#define EEPROM_SIZE 16384

void init_ext_eeprom()
{
    output_float(EEPROM_SCL);
    output_float(EEPROM_SDA);
}

void write_ext_eeprom(long int address, byte data)
{
    short int status;
    i2c_start();
    i2c_write(0xa0);
    i2c_write(address>>8);
    i2c_write(address);
    i2c_write(data);
    i2c_stop();
    i2c_start();
    status=i2c_write(0xa0);
    while(status==1)
    {
        i2c_start();
        status=i2c_write(0xa0);
    }
}
```

```
    }  
}  
byte read_ext_eeprom(long int address) {  
    byte data;  
    i2c_start();  
    i2c_write(0xa0);  
    i2c_write(address>>8);  
    i2c_write(address);  
    i2c_start();  
    i2c_write(0xa1);  
    data=i2c_read(0);  
    i2c_stop();  
    return(data);  
}
```

## **Anexo III**

**Plantillas para Programación de la Placa Base para PIC 16F877**

## Base para PIC 16F877

```
//Configuración general
//Definición del PIC a utilizar y configuración del conversor A/D a 10 bits
#device PIC16F877 adc=10
//Inclusión de las definiciones del PIC a utilizar
#include <16F877.h>
#fuses HS,NOWDT,NOPROTECT,PUT,NOBROWNOUT,NOLVP,NOCPD,WRT
/*
HS
NOWDT: No use watch dog
NOPROTECT: No proteger el código
PUT: Power up timer (booto retardado)
NOBROWNOUT:no habilitar reset por baja tensión
NOLVP: no habilita programación con baja tensión
NOCPD: no proteja memoria de datos eeprom
WRT: Habilita la programación de la FLASH
NODEBUG: no habilitar modo debug
*/
//Rej a utilizar 20 mhz
#use delay(clock=20000000)
//Configuración del puerto RS485
#use rs232(baud=9600, xmit=PIN_c6,rcv=PIN_c7,errors,stream=RS485)
#define rts_485_on output_high(PIN_d7)
#define rts_485_off output_low(PIN_d7)
#define RX485_on output_low(PIN_c5)
#define RX485_off output_high(PIN_c5)
#define tx_485_on output_high(PIN_c6)
//Configuración del puerto RS232
```

```
#use rs232(baud=9600, xmit=PIN_b3,rcv=PIN_b2,errors,stream=RS232)
#define rts_232_on output_high(PIN_b4)
#define rts_232_off output_low(PIN_b4)
#define cts_232 input(PIN_b5)
#define tx_232_on output_high(PIN_b3)
//Entrada de teclado serie RS232 TTL
#use rs232(baud=9600, rcv=PIN_b0,errors,stream=kb)
#define Kb input(PIN_b0)

//Bus I2C
#define EEPROM_SDA PIN_c4
#define EEPROM_SCL PIN_c3

//Configuración de los puertos en modo manual (set_tris, en main)
#use fast_io(A)
#use fast_io(B)
#use fast_io(C)
#use fast_io(D)
#use fast_io(E)

//Salidas digitales
#define buzzer_on output_high(PIN_e0)
#define buzzer_off output_low(PIN_e0)
#define out0_on output_high(PIN_d0)
#define out0_off output_low(PIN_d0)
#define out1_on output_high(PIN_d1)
#define out1_off output_low(PIN_d1)
#define out2_on output_high(PIN_d2)
#define out2off output_low(PIN_d2)
#define out3_on output_high(PIN_d3)
#define out3_off output_low(PIN_d3)
#define out4_on output_high(PIN_d4)
#define out4_off output_low(PIN_d4)
#define out5_on output_high(PIN_d5)
```

```
#define out5_off output_low(PIN_d5)
#define out6_on  output_high(PIN_d6)
#define out6_off output_low(PIN_d6)
#define out7_on  output_high(PIN_c5)
#define out7_off output_low(PIN_c5)
#define out8_on  output_high(PIN_b2)
#define out8_off output_low(PIN_b2)
#define out9_on  output_high(PIN_b3)
#define out9_off output_low(PIN_b3)
#define out10_on output_high(PIN_b4)
#define out10_off output_low(PIN_b4)
#define out11_on output_high(PIN_b5)
#define out11_off output_low(PIN_b5)
#define out12_on output_high(PIN_b6)
#define out12_off output_low(PIN_b6)
#define out13_on output_high(PIN_b7)
#define out13_off output_low(PIN_b7)

//Entradas digitales
#define inp0 input(PIN_a2)
#define inp1 input(PIN_a4)
#define inp2 input(PIN_a5)
#define inp3 input(PIN_c0)
#define inp4 input(PIN_c1)
#define inp5 input(PIN_c2)
#define inp6 input(PIN_d7)
#define inp7 input(PIN_b1) //Int1 18Fxxx

//Definiciones para el display LCD (usa librería lcdser_plc.h>
#define ck0 output_low(PIN_a7)
#define ck1 output_high(PIN_a7)
#define data0 output_low(PIN_a6)
#define data1 output_high(PIN_a6)
#define strobe1_off output_low(PIN_a2)
```

```
#define strobe1_on output_high(PIN_a2)
```

```
/*
```

```
Entradas Analógicas de 10 bits (0 a 1023)
```

```
RA0: Entrada analógica, Jp1=On Input: 4 a 20mA / JP1=Off Input:0 a 5V
```

```
RA1: Entrada analógica, Jp2=On Input: 4 a 20mA / JP2=Off Input:0 a 5V
```

```
RA3: Entrada analógica, Jp3=On Input: 4 a 20mA / JP3=Off Input:0 a 5V
```

```
Configuración de las entradas analógicas
```

```
setup_adc_ports(RA0_RA1_RA3_ANALOG);
```

```
setup_adc(ADC_CLOCK_DIV_8);
```

```
Selección del canal analógico a leer (0,1 o 3)
```

```
set_adc_channel(0);
```

```
delay_us(20); //retardo necesario para estabilizacion del canal seleccionado
```

```
x0 = read_adc(); //lectura del canal, x0 variable tipo unsigned int16
```

```
*/
```

```
/* Hardware PIC16F877
```

```
;RA0 <..... An0 Entrada analógica 0
```

```
;RA1 <..... An1 Entrada analógica 1
```

```
;RA2 <..... Digital Input 0
```

```
;RA3 <..... An1 Entrada analógica 3
```

```
;RA4 <..... Digital Input 1
```

```
;RA5 <..... Digital Input 2
```

```
;RE0 .....> Ck LCD
```

```
;RE1 .....> Data LCD
```

```
;RE2 .....> Strobe LCD
```

```
;RB0 <..... Keyboard Input (RS232 TTL, Int0)
```

```
;RB1 <.....> Digital Input 8 (Int1 18FXXX)
```

```
;RB2 .....> Digital Output 8 / (Inp) RX Rs232 (Int2 18FXXX)
```

```
;RB3 .....> Digital Output 9 / (Out) TX Rs232
```

```
;RB4 .....> Digital Output 10 / (Out) RTS Rs232 JP6=ON
```

```
;RB5 .....> Digital Output 11 / (Inp) CTS Rs232 JP7=ON
```

```
;RB6 .....> Digital Output 12 / ICP
;RB7 .....> Digital Output 13 / ICP
;RC0 <..... Digital Input 3
;RC1 <..... Digital Input 4
;RC2 <..... Digital Input 5
;RC3 .....> SCL I2C (Serial Clock)
;RC4 <.....> SDA I2C (Serial Data)
;RC5 .....> CTS COM1 (Enable RX RS485) / Digital Output 7 (JP5=ON sin
chip 485)
;RC6 .....> TX COM1 (RS485)
;RC7 <..... RX COM1 (RS485)
;RD0 .....> Digital Output 0 / Buzzer
;RD1 .....> Digital Output 1
;RD2 .....> Digital Output 2
;RD3 .....> Digital Output 3
;RD4 .....> Digital Output 4
;RD5 .....> Digital Output 5
;RD6 .....> Digital Output 6
;RD7 <..... RTS COM1 RS485(JP4=OFF) / Digital Input 6 (JP4=ON, sin chip
485)
```

```
*/
```

```
//Globales
```

```
main()
```

```
{
```

```
    //Variables locales del main
```

```
    // Definición de puertos 0=out 1=inp
```

```
    set_tris_a (0b111111); //Define el puerto A
```

```
    set_tris_b (0b11000001); //Define el puerto B
```

```
    set_tris_c (0b10000111); //Define el puerto C
```

```
    set_tris_d (0b10000000); //Define el puerto D
```

```
    set_tris_e (0b000); //Define el puerto E
```

```
    delay_ms(1000);
```

```
//set_tris_b (0b00000001); //Define el puerto B

//Reset general
OUTPUT_A(0);
OUTPUT_B(0);
OUTPUT_C(0);
OUTPUT_D(0);
OUTPUT_E(0);

//Estado inicial pin Tx=high puertos RS232 y RS485
tx_485_on;
tx_232_on;

//Inicio
delay_ms(100);
//Ciclo de SCAN
while (TRUE)
{
//Programación del proyecto
}
}
```

## **Anexo IV**

Características Generales del Microcontrolador PIC 16F877



# PIC16F8XX

## EEPROM Memory Programming Specification

This document includes the programming specifications for the following devices:

- PIC16F83
- PIC16CR83
- PIC16F84
- PIC16CR84
- PIC16F84A
- PIC16F877

### 1.0 PROGRAMMING THE PIC16F8X

The PIC16F8X is programmed using a serial method. The serial mode will allow the PIC16F8X to be programmed while in the users system. This allows for increased design flexibility. This programming specification applies to PIC16F8X devices in all packages.

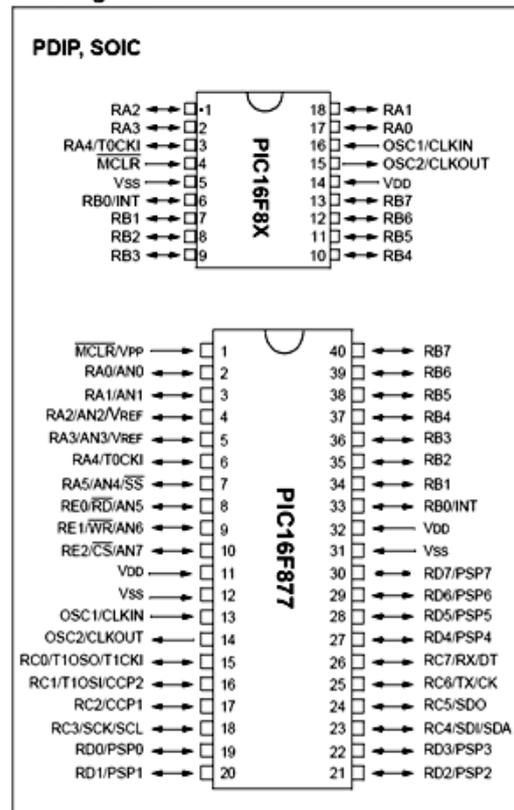
#### 1.1 Hardware Requirements

The PIC16F8X requires one programmable power supply for VDD (4.5V to 5.5V) and a VPP of 12V to 14V. Both supplies should have a minimum resolution of 0.25V.

#### 1.2 Programming Mode

The programming mode for the PIC16F8X allows programming of user program memory, data memory, special locations used for ID, and the configuration word.

### Pin Diagram



### PIN DESCRIPTIONS (DURING PROGRAMMING): PIC16F8X

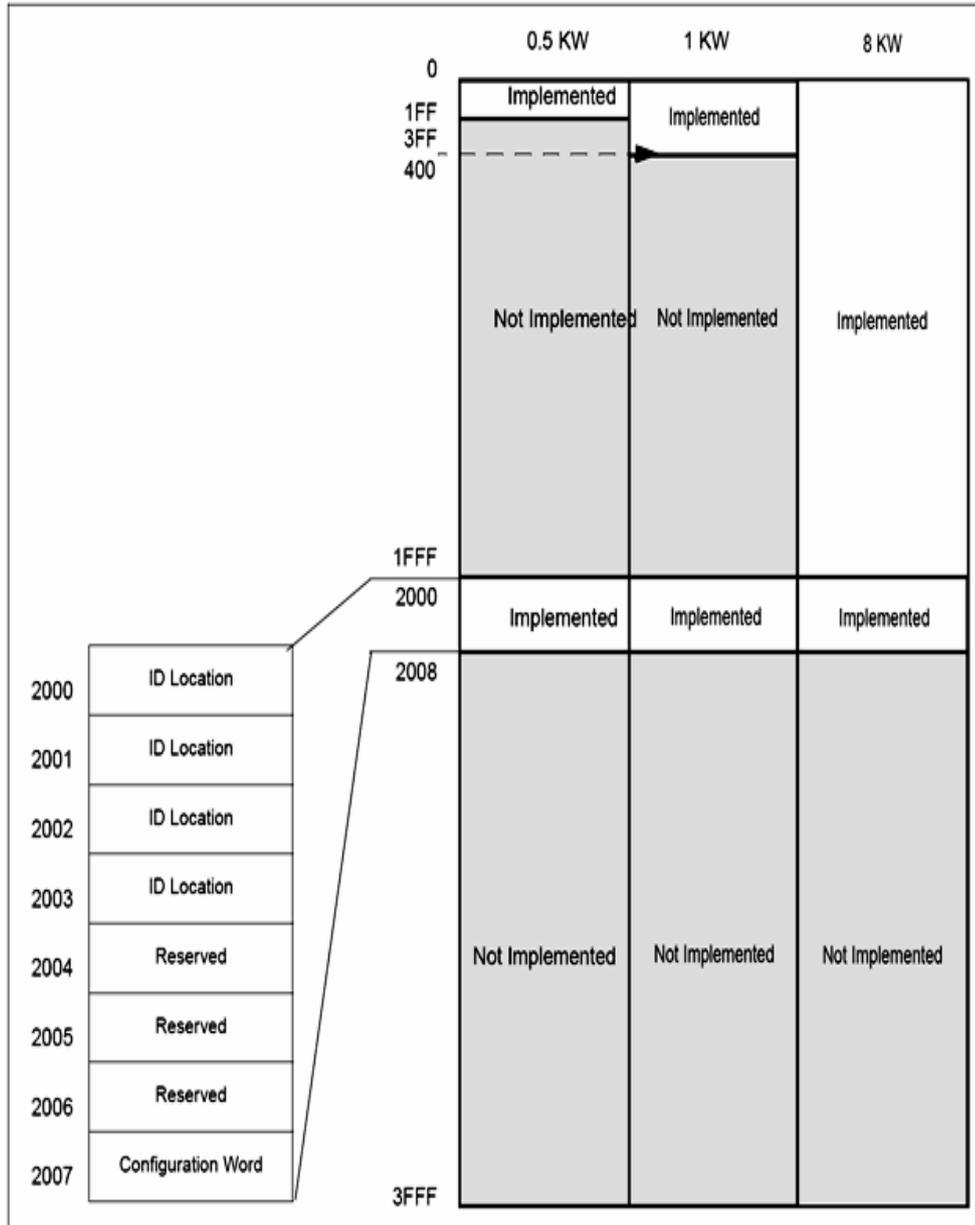
Pin Name	During Programming		
	Function	Pin Type	Pin Description
RB6	CLOCK	I	Clock input
RB7	DATA	I/O	Data input/output
MCLR	VTEST MODE	P*	Program Mode Select
VDD	VDD	P	Power Supply
Vss	Vss	P	Ground

Legend: I = Input, O = Output, P = Power

\*In the PIC16F8X, the programming high voltage is internally generated. To activate the programming mode, high voltage needs to be applied to MCLR input. Since the MCLR is used for a level source, this means that MCLR does not draw any significant current.

# PIC16F8XX

**FIGURE 2-1: PROGRAM MEMORY MAPPING**



**FIGURE 3-2: CONFIGURATION WORD FOR PIC16F877**

CP1	CP0	BKBUG	-	WRT	CPD	LVP	BODEN	CP1	CP0	PWRT $\overline{\text{E}}$	WDTE	FOSC1	FOSC0	Register: CONFIG
													Address 2007h	
bit13													bit0	
bit 13-12:														
bit 11: <b>BKBUG:</b> Background Debugger Mode (This bit documented as reserved in data sheet)														
1 = Background debugger functions not enabled														
0 = Background debugger functional.														
bit 5-4: <b>CP1:CP0:</b> Flash Program Memory Code Protection bits <sup>(2)</sup>														
11 = Code protection off														
10 = 1F00h to 1FFFh code protected														
01 = 1000h to 1FFFh code protected														
00 = 0000h to 1FFFh code protected														
bit 11: <b>Reserved:</b> Set to '1' for normal operation														
bit 10: <b>Unimplemented:</b> Read as '1'														
bit 9: <b>WRT:</b> Flash Program Memory Write Enable														
1 = Unprotected program memory may be written to by EECON control														
0 = Unprotected program memory may not be written to by EECON control														
bit 8: <b>CPD:</b> Data EE Memory Code Protection														
1 = Code protection off														
0 = Data EE memory code protected														
bit 7: <b>LVP:</b> Low voltage programming Enable bit														
1 = RB3/PGM pin has PGM function, low voltage programming enabled														
0 = RB3 is digital I/O, HV on $\overline{\text{MCLR}}$ must be used for programming														
bit 6: <b>BODEN:</b> Brown-out Reset Enable bit <sup>(1)</sup>														
1 = BOR enabled														
0 = BOR disabled														
bit 3: <b>PWRT<math>\overline{\text{E}}</math>:</b> Power-up Timer Enable bit <sup>(1)</sup>														
1 = PWRT disabled														
0 = PWRT enabled														
bit 2: <b>WDTE:</b> Watchdog Timer Enable bit														
1 = WDT enabled														
0 = WDT disabled														
bit 1-0: <b>FOSC1:FOSC0:</b> Oscillator Selection bits														
11 = RC oscillator														
10 = HS oscillator														
01 = XT oscillator														
00 = LP oscillator														
Note 1: Enabling Brown-out Reset automatically enables Power-up Timer (PWRT) regardless of the value of bit $\overline{\text{PWRT\text{E}}}$ . Ensure the Power-up Timer is enabled anytime Brown-out Reset is enabled.														
2: All of the CP1:CP0 pairs have to be given the same value to enable the code protection scheme listed.														

## **Anexo V**

Fotos del Módulo

**Foto 1. Módulo del Control Local y Remoto de un Variador de Frecuencia para el Manejo de un Motor AC Utilizando el MICROCONTROLADOR PIC 16f877 y la Plataforma de Desarrollo Labview”**



**Foto 2. Motor ¼ HP**



**Foto 3. Variador de Frecuencia**



**Foto 4. Montaje Completo del Proyecto**



## REFERENCIAS BIBLIOGRÁFICAS

- Rubén Herrán Sainz; Ingeniería Técnica Industrial, Electricidad. Dpto. de Ingeniería Eléctrica y Energética UNIVERSIDAD DE CANTABRIA
- Lisjak Leandro; Principios De Los Controladores De Motores A Inducción
- Andrés Cánovas López; Manual De Usuario Del Compilador Pcw De Ccs
- Microchip; Future PICmicro Microcontroller Products Guide.
- OCEANO / CENTRUM; Encyclopedía de la Electrónica Ingeniería y Teoría
- Papenkort Franz; Diagramas Eléctricos de Control y Protección Papenkort Franz
- Ing. Marco Singaña; Diseño y construcción de un laboratorio para microcontroladores PIC.
- Dawes Chester L, Electricidad Industrial Baterías Electricidad Electromagnetismo Motores Electricos.
- Ceballos Francisco Javier, Programación orientada a objetos con C++.
- GOTTFRIEND Cyron, Programación en C, McGraw Hill.
- Alberto Llorente; Visualizadores de Información. Pantallas de cristal Líquido.
- BELOVE Charleo, Enciclopedia de la Electrónica Ingeniería y Técnicas.
- Curtis Jonson, Process Control Instrumentation Technology, PRENTICE HALL.
- MACKAY, S.G. "Data Communication for Instrumentation and Control", IDC Techbooks, Australia 2000.
- SHINSKEY F. GREG, Sistemas de Control de Procesos, Aplicación Diseño y Sintonización. Tomo I.
- CREUS S. Antonio, Instrumentación Industrial. Ediciones Marcombo 1995.
- Cooper William, Instrumentación Electrónica y Mediciones.
- National Instrumenst, Lab-PC-1200/AI USER MANUAL.
- National Instrumenst , LabVIEW Data Acquisition.
- MALONEY, Timmothy, "Electrónica Industrial Moderna", Tercera Edición, Prentice-Hall Hispanoamericana, México 1996.

- [www.datex-ohmeda.es/aula-bioingenieria/numero8/Tecnologia\\_LCD.htm](http://www.datex-ohmeda.es/aula-bioingenieria/numero8/Tecnologia_LCD.htm);  
LCD.
- [http://www.zator.com/Hardware/H2\\_5\\_1.htm](http://www.zator.com/Hardware/H2_5_1.htm); Información Puerto serie
- [www.caveo.com.ar](http://www.caveo.com.ar); Interfases del proyecto
- <http://www.microchip.com/10/search/sitemap/submaps/appmap/index.htm>;  
Descripción PIC 16f877
- <http://www.ni.com>, Referencia de desarrollo de sistemas SCADA
- <http://www.terra.es/personal2/vistaatx/domotica/web/arquitec.htm>
- <http://www.geocities.com/automatizacionuda/TrabMod.htm>, Información automatización.
- [http://www.geocities.com/ResearchTriangle/Campus/6495/SISTEMAS\\_SCADA.htm](http://www.geocities.com/ResearchTriangle/Campus/6495/SISTEMAS_SCADA.htm), Introducción a los Sistemas SCADA.
- <http://www.monografias.com/trabajos11/sisco/sisco.shtml>, Sistemas SCADA.

Latacunga Julio del 2005

Elaborado por:

---

Cristian Paúl Recalde Tapia

---

Patricio Sebastián Zurita Armijos

EL DIRECTOR DE LA CARRERA DE INGENIERÍA  
ELECTRÓNICA ESPECIALIDAD EN INSTRUMENTACIÓN

---

Ing. Nancy Guerrón

EL SECRETARIO DE LA ESPE – LATACUNGA

---

Ab. Eduardo Vásquez Alcázar