1. INTRODUCCIÓN.

Los PLC[1] son dispositivos electrónicos muy usados en la automatización industrial.

La historia del los PLC se remonta a finales de la década de 1960, cuando la industria estaba en busca de nuevas tecnologías electrónicas para reemplazar y hacer más eficiente los sistemas de control basados en circuitos eléctricos con relés, interruptores y otros componentes comúnmente utilizados para el control de los sistemas de lógica combinacional.

El propósito de este proyecto es diseñar e implementar un controlador lógico programable (PLC) basado en un microcontrolador PIC el mismo que proporcionara un sistema de control de potencias en sus salidas y en sus entradas poder censar o realizar alguna función específica, este dispositivo podrá programarse fácilmente, tendrá una pantalla de LCD con la finalidad de visualizar según el programa lo que en este se ejecute, también contará con un sistema de comunicación hacia una computadora pudiendo utilizarse para monitorear las funciones del controlador lógico programable (PLC).

Este diseño es un aporte tecnológico para aquellas personas que hagan uso del presente proyecto, así se ha cumplido con el método investigativo que al comenzar a desarrollar la tesis se ha planteado, mismo que ya esta diseñado.

La Universidad Nacional de Loja como organismo de Educación e Investigación a través de la carrera de Tecnología en Electrónica, en su afán de mejorar los procesos de enseñanza y aprendizaje, ha permitido diseñar e implementar un controlador lógico programable (PLC) basado en un microcontrolador PIC, como una herramienta de apoyo para facilitar y ayudar tanto a los docentes como alumnos que se educan en esta Institución.

[1] **PLC** (Programmable Logic Controller) Controlador Lógico Programable.- Sirve para controlar y Automatizar Sistemas Industriales gracias a que puede manejar niveles elevados de voltaje y corriente.

OBJETIVO GENERAL.

Diseñar e implementar un controlador lógico programable (PLC) basado en un microcontrolador PIC.

OBJETIVOS ESPECIFICOS.

- Aportar experiencias en el campo de los sistemas micro procesados, instrumentación y electrónica de potencia.

- Desarrollar el software necesario para la implementación del proceso de automatización basados en el PLC planteado.

- Construir con una opción de bajo costo, los procesos de automatización y control que se generan en la Carrera, Área y la Universidad como aporte a la sociedad.

- Aportar al Área de Energía, las Industrias y los Recursos Naturales no Renovables con una solución técnica viable en la que a prácticas pre profesionales se refiere, en los campos de la electrónica de potencia, microcontroladores, programación y automatización.

2. REVISIÓN DE LA LITERATURA

2.1. DEFINICIÓN DE UN CONTROLADOR LÓGICO PROGRAMABLE (PLC) A BASE DE UN MICROCONTROLADOR PIC16F877A.

EL Controlador Lógico Programable (PLC) construido a base de un microcontrolador para el cual se Utilizo el PIC16F877A, consiste en controlar y automatizar sistemas que requieren o consumen grandes cantidades de potencia.

La automatización o control de sistemas de potencia se realizan utilizando un microcontrolador que es un circuito integrado cuyas características técnicas permiten programarlo para realizar tareas especificas, pudiendo también borrar el programa para luego cargar uno nuevo, el microcontrolador funciona con 5 Voltios de corriente continua, en sus pines se pueden registrar dos estados lógicos según la electrónica digital que son: El estado alto o 1 lógico equivalente a 5 VDC, y el estado bajo u 0 lógico equivalente a 0 VDC.

El PLC esta compuesto por:

- 8 Entradas que soportan 12 Voltios de corriente continua.
- 8 Salidas que en corriente continúa soportan desde 0V hasta 40V a 5 Amperios, y en corriente alterna soporta desde 0V hasta 125V a 12 Amperios desde 130V hasta 250V a 7 Amperios.
- Interface para comunicación serial PLC a PC mediante interface serial RS-232.
- Display LCD 2x16.
- Interface de Programación del PLC.
- 8 Puertos para pruebas soportan Tensiones desde 0 a 5VDC.

2.1.1. Ventajas de un PLC a base de un Microcontrolador.

Sencillez y Rapidez: Es un dispositivo de automatización a nivel industrial muy compacto y que se adapta a cualquier necesidad en la Industria, además es muy fácil de utilizar y montar ya que solo es necesario cargarle el programa y colocar las entradas y salidas según lo requerido.

Fácil Programación: Para hacer el programa que se vaya a cargar al microcontrolador, se utilizó una herramienta para programar los micro Pic llamada Microcode Studio, ésta herramienta permite programar en lenguaje Basic que es bastante fácil de aprender.

Aplicación en la Domótica: Este dispositivo puede ser utilizado para el control de grandes potencias desde la facilidad de su PC a través del puerto serie por lo que podría controlar fácilmente una casa.

Herramienta de Laboratorio: Este dispositivo puede ser muy útil como herramienta de laboratorio ya que además de servir como programador del PIC16F877A sirve para probar o ejecutar el programa cargado en el microcontrolador sin tener que retirar el Pic del dispositivo agilizando las prácticas y ayudando a comprender al estudiante la infinidad de proyectos que se pueden realizar con el microcontrolador y como se podría ajustar este a sus necesidades.

Menor Costo: Este sistema aunque un poco complejo es muy económico en relación a otros sistemas de automatización como por ejemplo los temporizadores, relés, contactores, etc.

2.2. COMPONENTES UTILIZADOS PARA LA CONSTRUCCION DE UN CONTROLADOR LÓGICO PROGRAMABLE (PLC) A BASE DE UN MICROCONTROLADOR.

Los componentes utilizados en el Diseño y construcción de este proyecto se detallan a continuación:

2.2.1 Microcontrolador PIC16F877A.

Un microcontrolador es un circuito integrado programable que contiene todos los componentes de un computador.

Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporando en el propio dispositivo al que gobierna.

El microcontrolador es un computador dedicado. En su memoria sólo reside un programa destinado a gobernar una aplicación determinada, sus líneas de entrada/salida soportan el coleccionado de los sensores y actuadores del dispositivo a controlar y todos los recursos complementarios son disponibles y tienen como única finalidad atender sus requerimientos. Una vez programado y configurado el microcontrolador solamente sirven para gobernar la tarea asignada.

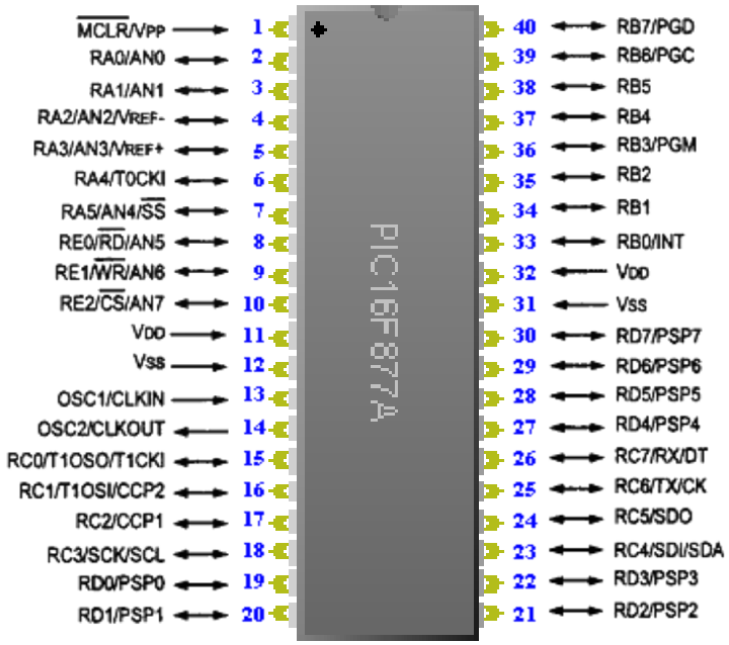

FIGURA 1: PIC16F877A, descripción de pines.

Voltaje DC	+5 V
Corriente	8 mA
Frecuencia de Operación	DC - 20 MHz
Resets	POR, BOR (PWRT, OST)
Memoria FLASH (14 Bit - Words)	8K
Memoria de Datos (RAM) - Bytes	368
Memoria EEPRON	256
Interrupciones	14
Puertos I/0	Puertos A,B,C,D,E
Temporizadores	3
Modulador de anchura de Pulsos PWM	2
Comunicación Serial	MSSP, USART
Comunicación Paralela	PSP
Modulo de 10 Bits Analógico – Digital	8 canales de entrada
Configuración de Instrucciones	35 instrucciones

TABLA 1: DATOS TECNICOS DEL PIC16F877A.

Descripción.

Frecuencia de operación del PIC16F877A.- El microcontrolador utilizado para este proyecto utiliza cristales de frecuencia de 4, 8, 12,16 y 20 MHz estos respectivamente deberán ir acompañados por dos capacitores cerámicos de 22 pf.

Memoria FLASH.- Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos y es más pequeña.

Memoria de Datos (RAM[2]).- Estos dispositivos son de poca capacidad pues sólo deben contener las variables y los cambios de información que se produzcan en el transcurso del programa. Por otra parte, como sólo existe un programa activo, no se requiere guardar una copia del mismo en la RAM pues se ejecuta directamente desde la ROM[3].

Memoria EEPRON[4].- Se trata de memorias de sólo lectura, programables y borrables eléctricamente

Tanto la programación como el borrado, se realizan eléctricamente desde el propio grabador y bajo el control programado de un PC. Es muy cómoda y rápida la operación de grabado y la de borrado.

Interrupción.- Cuando la CPU acepta una solicitud de interrupción ejecuta un salto a la dirección 0004h, por lo cual a esta se le conoce como "**dirección del vector de interrupción**". El registro PCLATH (Contador de Programa Alto) no es modificado en esta circunstancia, por lo cual habrá que tener cuidado al manipular el PC dentro de la Rutina de Atención a la Interrupción.

[2] **RAM:** (Random Access Memory) Memoria de Acceso Aleatorio.
[3] **ROM:** (Read Only Memory) Memoria de sólo lectura.
[4] **EEPROM** (Electrical Erasable Programmable Read Only Memory) ROM Programable y Borrable Eléctricamente.

Temporizadores-. Se emplean para controlar periodos de tiempo y para llevar la cuenta de acontecimientos que suceden en el exterior (contadores).

Para la medida de tiempos se carga un registro con el valor adecuado y a continuación dicho valor se va incrementando o decrementando al ritmo de los impulsos de reloj o algún múltiplo hasta que se desborde y llegue a 0, momento en el que se produce un aviso.

Cuando se desean contar acontecimientos que se materializan por cambios de nivel o flancos en alguna de las patitas del microcontrolador, el mencionado registro se va incrementando o decrementando al ritmo de dichos impulsos.

Modulador de anchura de Pulsos PWM.- Son circuitos que proporcionan en su salida impulsos de anchura variable, que se ofrecen al exterior a través de las patitas del encapsulado.

Comunicación Serial.- El microcontrolador tiene en el propio hardware una función especial llamada USART, esta función es la de comunicar al microcontrolador con otros dispositivos como otros microcontroladores o con una PC mediante la interface serial Asíncrona.

La interfaz USART incorporada en el Pic se encuentra a través de los pines 25= C6 como TX (Transmisor) y 26=C7 como RX (Receptor).

Comunicación Paralela.- (PSP): de 8 bits con líneas de protocolo.

Los datos viajan simultáneamente a través de 4 hilos, tiene la ventaja de que la transferencia de datos es mas rápida, pero el inconveniente es que necesitamos un cable por cada BIT de dato, lo que encarece y dificulta el diseño de las placas, otro inconveniente es la capacitancia que genera los conductores por lo que la transmisión se vuelve defectuosa a partir de unos pocos metros.

Modulo de 10 Bits Analógico – Digital.- El ADC es un convertidor de aproximaciones sucesivas de 10 Bits, el cual puede realizar la conversión de una de las 8 entradas o canales analógicos AN0 hasta AN7 multiplexadas por la lógica interna que utiliza como líneas de selección de canal los Bits CHS2 - CHS0, en donde se coloca el número en binario del canal a convertir.

Conversor A/D (CAD).- Los microcontroladores que incorporan un Conversor A/D (Analógico/Digital) pueden procesar señales analógicas, tan abundantes en las aplicaciones. Suelen disponer de un multiplexor que permite aplicar a la entrada del CAD diversas señales analógicas desde las patitas del circuito integrado.

Conversor D/A (CDA).- Transforma los datos digitales obtenidos del procesamiento del computador en su correspondiente señal analógica que saca al exterior por una de las patitas de la cápsula. Existen muchos efectores que trabajan con señales analógicas.

Perro Guardián o "WATCHDOG".- Cuando el computador personal se bloquea por un fallo del software u otra causa, se pulsa el botón del reset y se reinicializa el sistema. Pero un microcontrolador funciona sin el control de un supervisor y de forma continuada las 24 horas del día. El Perro guardián consiste en un temporizador que, cuando se desborda y pasa por 0, provoca un reset automáticamente en el sistema.

Protección ante fallo de Alimentación o "BROWNOUT".- Se trata de un circuito que resetea al microcontrolador cuando el voltaje de alimentación (VDD) es inferior a un voltaje mínimo ("brownout"). Mientras el voltaje de alimentación sea inferior al de brownout el dispositivo se mantiene reseteado, comenzando a funcionar normalmente cuando sobrepasa dicho valor.

Estado de Reposo o de Bajo Consumo.- Son abundantes las situaciones reales de trabajo en que el microcontrolador debe esperar, sin hacer nada, a que se produzca algún acontecimiento externo que le ponga de nuevo en funcionamiento. Para ahorrar energía, (factor clave en los aparatos

portátiles), los microcontroladores disponen de una instrucción especial (SLEEP en los PIC), que les pasa al estado de reposo o de bajo consumo, en el cual los requerimientos de potencia son mínimos. En dicho estado se detiene el reloj principal y se "congelan" sus circuitos asociados, quedando sumido en un profundo "sueño" el microcontrolador. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

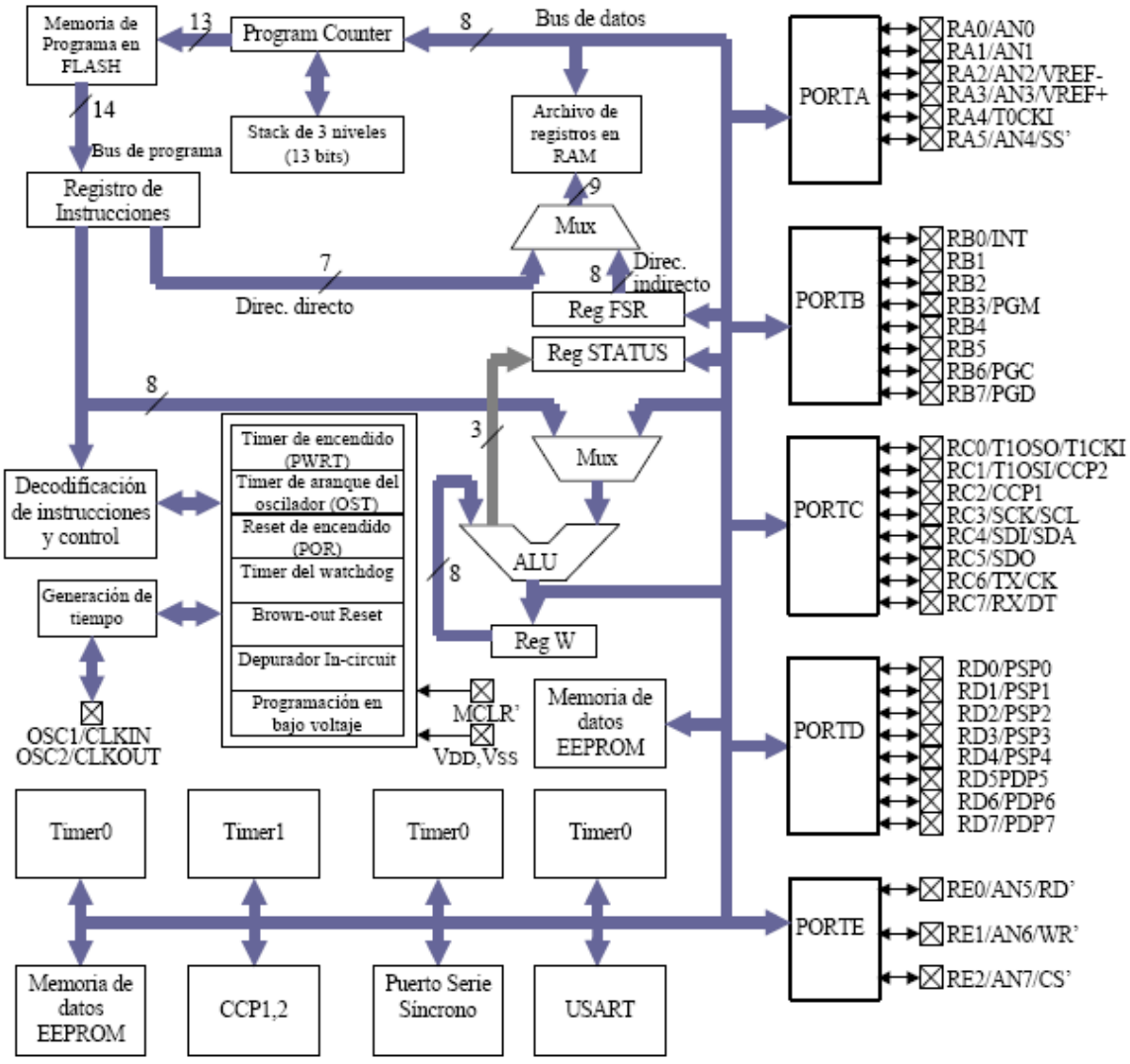

FIGURA 2: Diagrama de Bloques del PIC16F877A.

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	13	14	30	I	ST/CMOS(4)	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	14	15	31	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	1	2	18	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
						PORTA is a bi-directional I/O port.
RA0/AN0	2	3	19	I/O	TTL	RA0 can also be analog input0.
RA1/AN1	3	4	20	I/O	TTL	RA1 can also be analog input1.
RA2/AN2/VREF-	4	5	21	I/O	TTL	RA2 can also be analog input2 or negative analog reference voltage.
RA3/AN3/VREF+	5	6	22	I/O	TTL	RA3 can also be analog input3 or positive analog reference voltage.
RA4/T0CKI	6	7	23	I/O	ST	RA4 can also be the clock input to the Timer0 timer/ counter. Output is open drain type.
RA5/$\overline{SS}$/AN4	7	8	24	I/O	TTL	RA5 can also be analog input4 or the slave select for the synchronous serial port.
						PORTB is a bi-directional I/O port. PORTB can be soft-ware programmed for internal weak pull-up on all inputs.
RB0/INT	33	36	8	I/O	TTL/ST(1)	RB0 can also be the external interrupt pin.
RB1	34	37	9	I/O	TTL	
RB2	35	38	10	I/O	TTL	
RB3/PGM	36	39	11	I/O	TTL	RB3 can also be the low voltage programming input.
RB4	37	41	14	I/O	TTL	Interrupt-on-change pin.
RB5	38	42	15	I/O	TTL	Interrupt-on-change pin.
RB6/PGC	39	43	16	I/O	TTL/ST(2)	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock.
RB7/PGD	40	44	17	I/O	TTL/ST(2)	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data.

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

TABLA 2: Descripción de los pines para el PIC16F877A.

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
						PORTC is a bi-directional I/O port.
RC0/T1OSO/T1CKI	15	16	32	I/O	ST	RC0 can also be the Timer1 oscillator output or a Timer1 clock input.
RC1/T1OSI/CCP2	16	18	35	I/O	ST	RC1 can also be the Timer1 oscillator input or Capture2 input/Compare2 output/PWM2 output.
RC2/CCP1	17	19	36	I/O	ST	RC2 can also be the Capture1 input/Compare1 output/PWM1 output.
RC3/SCK/SCL	18	20	37	I/O	ST	RC3 can also be the synchronous serial clock input/ output for both SPI and I^2C modes.
RC4/SDI/SDA	23	25	42	I/O	ST	RC4 can also be the SPI Data In (SPI mode) or data I/O (I^2C mode).
RC5/SDO	24	26	43	I/O	ST	RC5 can also be the SPI Data Out (SPI mode).
RC6/TX/CK	25	27	44	I/O	ST	RC6 can also be the USART Asynchronous Transmit or Synchronous Clock.
RC7/RX/DT	26	29	1	I/O	ST	RC7 can also be the USART Asynchronous Receive or Synchronous Data.
						PORTD is a bi-directional I/O port or parallel slave port when interfacing to a microprocessor bus.
RD0/PSP0	19	21	38	I/O	ST/TTL(3)	
RD1/PSP1	20	22	39	I/O	ST/TTL(3)	
RD2/PSP2	21	23	40	I/O	ST/TTL(3)	
RD3/PSP3	22	24	41	I/O	ST/TTL(3)	
RD4/PSP4	27	30	2	I/O	ST/TTL(3)	
RD5/PSP5	28	31	3	I/O	ST/TTL(3)	
RD6/PSP6	29	32	4	I/O	ST/TTL(3)	
RD7/PSP7	30	33	5	I/O	ST/TTL(3)	
						PORTE is a bi-directional I/O port.
RE0/$\overline{RD}$/AN5	8	9	25	I/O	ST/TTL(3)	RE0 can also be read control for the parallel slave port, or analog input5.
RE1/$\overline{WR}$/AN6	9	10	26	I/O	ST/TTL(3)	RE1 can also be write control for the parallel slave port, or analog input6.
RE2/$\overline{CS}$/AN7	10	11	27	I/O	ST/TTL(3)	RE2 can also be select control for the parallel slave port, or analog input7.
VSS	12,31	13,34	6,29	P	—	Ground reference for logic and I/O pins.
VDD	11,32	12,35	7,28	P	—	Positive supply for logic and I/O pins.
NC	—	1,17,28, 40	12,13, 33,34		—	These pins are not internally connected. These pins should be left unconnected.

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

TABLA 3: Descripción de los pines para el PIC16F877A, Continuación.

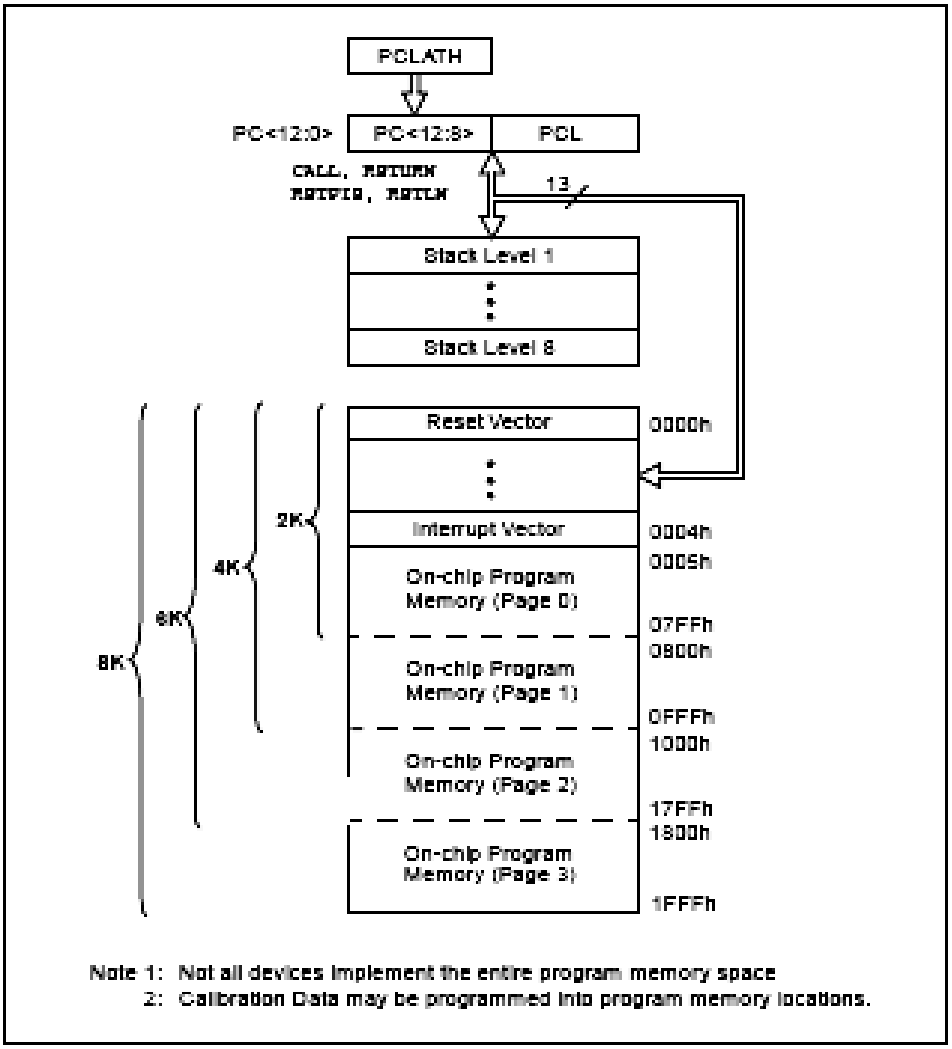

FIGURA 3: PIC16F877A organización de la memoria.

2.2.2 Max 232.

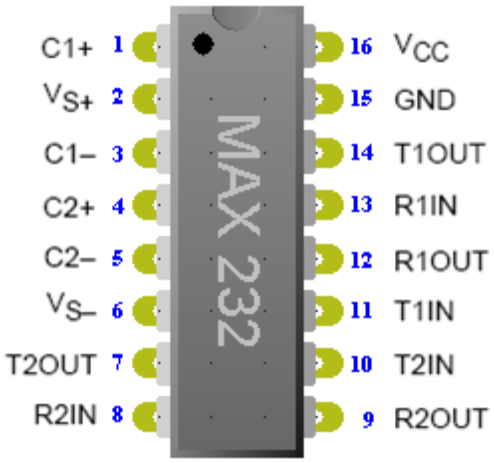

FIGURA 4: MAX232 descripción de sus pines

El CI. MAX232.- Es la solución para transmitir a mayor distancia, ya que incrementa los niveles de voltaje de 5V. a ± 10V gracias a un juego de capacitores que le ayudan a doblar los voltajes por lo que para su alimentación solo requiere una fuente de 5V. Que puede ser la misma que utiliza el PIC. El MAX232 dispone de 2 juegos de transmisores y receptores, de los cuales en este proyecto solo se ocupó un par de ellos.

Este chip dispone internamente de 4 conversores de niveles TTL al bus Standard RS232 y viceversa, permitiendo conectar un PC con un microcontrolador utilizando la comunicación serie. Sólo es necesario este chip y 4 condensadores electrolíticos de 10 micro-faradios vea el esquema siguiente:

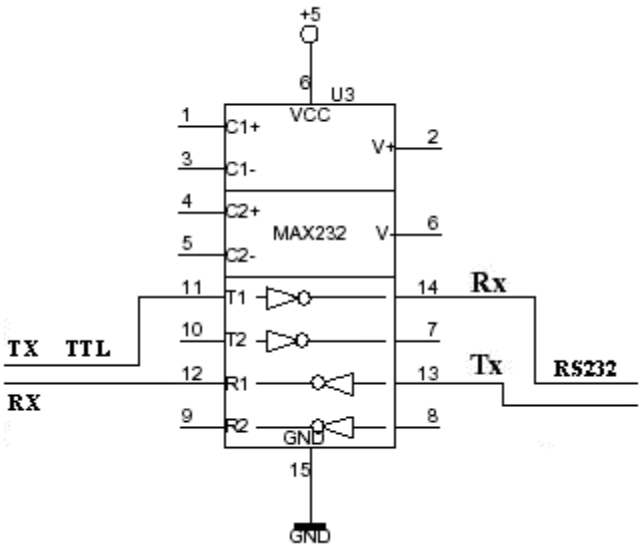

FIGURA 5: Diagrama de Conexión interna del MAX232

Usos: Este integrado es usado para comunicar un microcontrolador o sistema digital con un PC o sistema basado en el bus serie RS232.

Input supply voltage range, V_{CC} (see Note 1)	–0.3 V to 6 V
Positive output supply voltage range, V_{S+}	V_{CC} – 0.3 V to 15 V
Negative output supply voltage range, V_{S-}	–0.3 V to –15 V
Input voltage range, V_I: Driver	–0.3 V to V_{CC} + 0.3 V
Receiver	±30 V
Output voltage range, V_O: T1OUT, T2OUT	V_{S-} – 0.3 V to V_{S+} + 0.3 V
R1OUT, R2OUT	–0.3 V to V_{CC} + 0.3 V
Short-circuit duration: T1OUT, T2OUT	Unlimited
Package thermal impedance, θ_{JA} (see Note 2): D package	73°C/W
DW package	57°C/W
N package	67°C/W
NS package	64°C/W
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds	260°C
Storage temperature range, T_{stg}	–65°C to 150°C

TABLA 4: Hoja de Datos Técnicos del CI MAX232.

COMUNICACIÓN SERIAL RS232.- (También conocido como Electronic Industries Alliance RS-232) es una interfaz que designa una norma para el intercambio serie de datos binarios entre un DTE[5] y un DCE[6], aunque existen otras situaciones en las que también se utiliza la interfaz RS-232.

En particular, existen ocasiones en que interesa conectar otro tipo de equipamientos, como pueden ser computadores. Evidentemente en el caso de interconexión entre los mismos, se requerirá la conexión de un DTE con otro DTE.

Construcción física.- La interfaz RS-232 está diseñada para distancias cortas, de unos 15 metros o menos, aunque cuando se utiliza el Driver Max 232 en comunicaciones entre PC y microcontroladores, se puede extender distancias de hasta 200 metros, y para velocidades de comunicación bajas, de no más de 20 KB. A pesar de ello, muchas veces se utiliza a mayores velocidades con un resultado aceptable. La interfaz puede trabajar en comunicación asíncrona o síncrona y tipos de canal simplex[7], half dúplex[8] o full dúplex[9].

Los circuitos y sus definiciones.- Las UART o U(S)ART[10] se diseñaron para convertir las señales que maneja la CPU y transmitirlas al exterior. Las UART deben resolver problemas tales como la conversión de voltajes internos del DCE con respecto al DTE, gobernar las señales de control, y realizar la transformación desde el bus de datos de señales en paralelo a serie y viceversa. Debe ser robusta y deberá tolerar circuitos abiertos, cortocircuitos y escritura simultánea sobre un mismo pin, entre otras consideraciones. Es en la UART en donde se implementa la interfaz.

Para los propósitos de la RS-232 estándar, una conexión es definida por un cable desde un dispositivo al otro. Hay 9 conexiones en la especificación

[5] **DTE**: Equipo terminal de datos
[6] **DCE**: Equipo de terminación del circuito de datos
[7] **Simplex:** Los datos siempre viajarán en una dirección, por ejemplo desde DCE a DTE.
[8] **Half Dúplex.-** Los datos pueden viajar en una u otra dirección, pero sólo uno a la vez; luego la línea debe ser conmutada antes que los datos puedan viajar en la otra dirección.
[9] **Full Dúplex**.- Los datos pueden viajar en ambos sentidos simultáneamente.
[10] **UART o U(S)ART.-** Transmisor y Receptor [Síncrono] Asíncrono Universal.

completa, pero es muy probable que se encuentren menos de la mitad de éstas en una interfaz determinada. La causa es simple, una interfaz full dúplex puede obtenerse con solamente 3 cables.

También la dirección de la flecha indica que dispositivo, (DTE o DCE) origina cada señal, a excepción de las líneas de tierra (---).

Sobre los circuitos, todos los voltajes están con respecto a la señal de tierra.

Las convenciones que se usan son las siguientes:

Voltaje	Señal	Nivel Lógico	Control
+3 a +25	Espacio	0	On
-3 a –25	Marca	1	Off

Los valores de voltaje se invierten desde los valores lógicos. Por ejemplo, el valor lógico más positivo corresponde al voltaje más negativo. También un 0 lógico corresponde a la señal de valor verdadero o activado. Por ejemplo si la línea DTR está al valor 0 lógico, se encuentra en la gama de voltaje que va desde +3 a +25 V, entonces DTR está listo (ready).

El canal secundario a veces se usa para proveer un camino de retorno de información más lento, de unos 5 a 10 bits por segundo. Si el módem usado acepta esta característica, es posible para el receptor aceptar o rechazar un mensaje sin tener que esperar el tiempo de conmutación, un proceso que usualmente toma entre 100 y 200 milisegundos.

Comunicaciones serie asíncronas

Primero se envía un bit de start, a continuación los bits de datos (primero el bit de mayor peso) y finalmente los bits de STOP.

El número de bits de datos y de bits de Stop es uno de los parámetros configurables, así como el criterio de paridad par o impar para la detección de errores. Normalmente, las comunicaciones serie tienen los siguientes parámetros: 1 bit de Start, 8 bits de Datos, 1 bit de Stop y sin paridad.

El protocolo RS232 consta de tres líneas: una de transmisión, una de recepción y el común entre las dos interfases de comunicación.

Para que los dos equipos se entiendan, ambos equipos deben de estar sincronizados a una misma velocidad de transmisión de datos, que en este caso específico será de 9600 baudios.

La transmisión es relativamente sencilla, simplemente se transmite primero un bit de inicio de comunicación, luego se transmiten los 8 bits de datos (1 byte) y finalmente un bit de parada. El tiempo en que se transmite un bit y otro es:

$$\frac{1}{9600} = 0.104ms$$

.

Todo este proceso se realiza por cada byte, es decir por cada bit que se desea transmitir, ver la siguiente Figura 6.

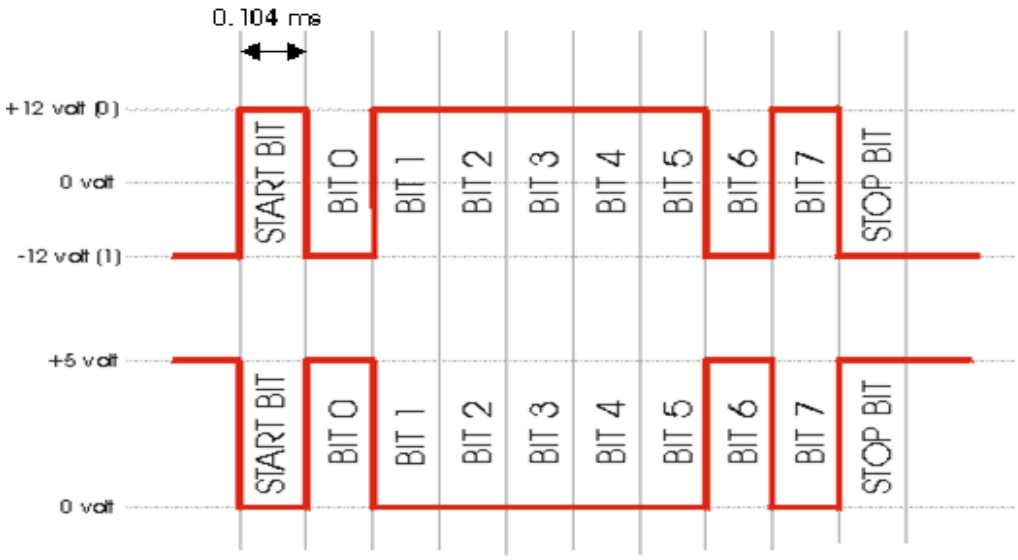

FIGURA 6: Diferencia entre la estrucura de una trama normal y otra utilizando el protocolo RS232.

Señales de comunicación del protocolo serial RS232

La recepción, se ejecuta solo cuando se detecta un bit de inicio; inmediatamente después de éste, se deberá de esperar 1.5 bits (0.156ms), para de esta forma leer cada uno de los bits con un nivel lógico ya establecido, tratando así de eliminar errores de lectura que se generarían si leemos los bits justamente cuando éstos recién se hayan transmitido.

El protocolo RS232 funciona en una lógica inversa es decir que para la señal, 1 lógica los valores correspondientes serán (-5V a -15V) en le transmisor y (-3V a -25V) en el receptor, para señal 0 lógica los valores correspondientes serán (+5V a +15V) en le transmisor y (+3V a +25V) en el receptor.

El microcontrolador tiene en el propio hardware una función especial llamada USART, esta función es la de comunicar a el microcontrolador con otros dispositivos como son otros microcontroladores o con una PC mediante la interface serial Asíncrona.

2.2.3 Opto Transistor 4N27.

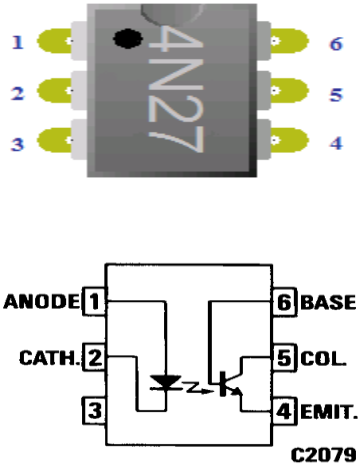

FIGURA 7: Opto Transistor 4N27, Imagen real, descripción Interna y de los pines

TOTAL PACKAGE	
*Storage temperature	−55°C to 150°C
*Operating temperature at junction	−55°C to 100°C
*Lead temperature (soldering, 10 sec)	260°C
*Total package power dissipation at 25°C ambient (LED plus detector)	250 mW
*Derate linearly from 25°C	3.3 mW/°C
INPUT DIODE	**OUTPUT TRANSISTOR**
*Forward DC current continuous — 80 mA	*Collector emitter voltage (BV_{CEO}) — 30 V
*Reverse voltage — 3.0 V	*Collector base voltage (BV_{CBO}) — 70 V
*Peak forward current (300 μs, 2% duty cycle) — 3.0 A	*Emitter collector voltage (BV_{ECO}) — 7 V
*Power dissipation at 25°C ambient — 150 mW	*Power dissipation at 25°C ambient — 150 mW
*Derate linearly from 25°C — 2.0 mW/°C	*Derate linearly from 25°C — 2.0 mW/°C
*Indicates JEDEC Registered Data.	

TABLA 5: Hoja de Datos Técnicos del Opto Transistor 4N27

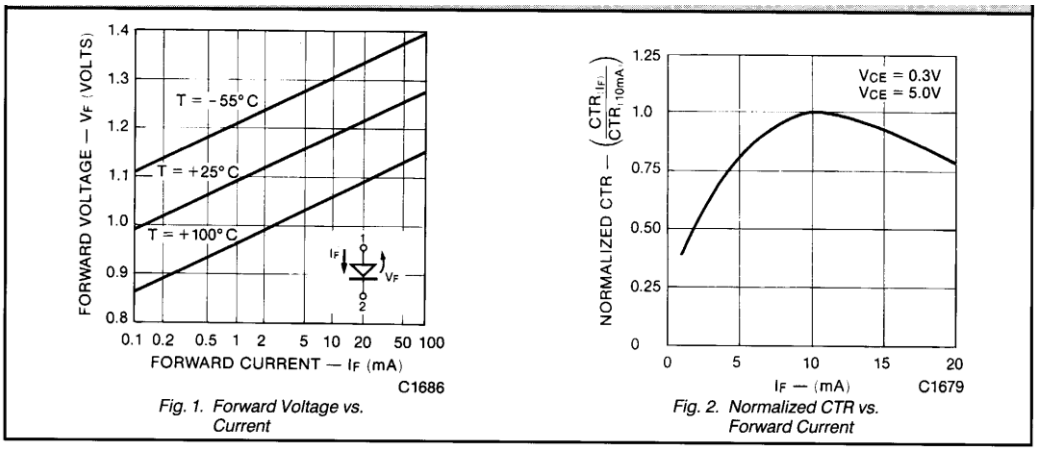

Fig. 1. Forward Voltage vs. Current

Fig. 2. Normalized CTR vs. Forward Current

FIGURA 8: Curvas Características del Opto transistor.

2.2.4 Display LCD 2x16.

FIGURA 9: Imagen de un LCD 2X16

CARACTERISTICAS TECNICAS.

Tensión de Alimentación.	12 V. D.C.
Consumo mínimo.	20 mA.
Consumo máximo.	300 mA.
Carga máx aplicable al relé.	5 A.
Vida media de la memoria.	1 Millón ciclos escritura aprox.
Visualización del LCD / Altura de carácter.	2 Líneas de 16 caracteres. / 9,4 mm.
Activación de mensajes, (conexión matricial de 7 x 7).	Cierre de contactos libres de potencial.
Función Luminiscencia.	Led.
Protección contra inversión de polaridad, (P.I.P.).	Si.
Medidas de la placa base.	107 x 76,25 x 30 mm.
Medidas del display.	84 x 44 x 20 mm.
Medidas área visible del display.	61 x 15,8 mm.

TABLA 6: Hoja de Datos Técnicos del LCD 2x16

2.2.5 Resistencia Eléctrica.

El símbolo de una resistencia es: ó

FIGURA 10: Imagen de una resistencia y su simbologìa

RESISTENCIA ELÉCTRICA.

Resistencia eléctrica es toda oposición que encuentra la corriente a su paso por un circuito eléctrico cerrado, atenuando o frenando el libre flujo de circulación de las cargas eléctricas o electrones. Cualquier dispositivo o consumidor conectado a un circuito eléctrico representa en sí una carga, resistencia u obstáculo para la circulación de la corriente eléctrica, esta se representa por el símbolo Omega: **Ω**

2.2.6 Condensador.

Condensador o capacitor es un dispositivo que almacena energía eléctrica, es un componente pasivo. Está formado por un par de superficies conductoras en situación de influencia total[11], generalmente en forma de tablas, esferas o láminas, separados por un material dieléctrico.

La capacidad de 1 faradio es mucho más grande que la de la mayoría de los condensadores, por lo que en la práctica se suele indicar la capacidad en micro- μF = 10^{-6}, nano- F = 10^{-9} o pico- F = 10^{-12} -faradios.

[11] todas las líneas de campo eléctrico que parten de una van a parar a la otra

Condensador Electrolítico.

Simbología.

FIGURA 11: Imagen de un Condensador Electrolìtico y su Simbologìa

Los condensadores electrolíticos deben su nombre a que el material dieléctrico que contienen es un ácido llamado electrolito y que se aplica en estado líquido. La fabricación de un condensador electrolítico comienza enrollando dos láminas de aluminio separadas por un papel absorbente humedecido con ácido electrolítico. Luego se hace circular una corriente eléctrica entre las placas para provocar una reacción química que producirá una capa de óxido sobre el aluminio, siendo este óxido de electrolito el verdadero dieléctrico del condensador. Para que pueda ser conectado en un circuito electrónico, el condensador llevará sus terminales de conexión remachados o soldados con soldadura de punto. Por último, todo el conjunto se insertará en una carcasa metálica que le dará rigidez mecánica y se sellará herméticamente, en general, con un tapón de goma, que evitará que el ácido se evapore en forma precoz.

Condensador Cerámico.

Simbología

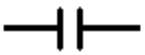

FIGURA 12: Imagen de un condensador ceramico y su Simbologìa

Son buenos aislantes térmicos y eléctricos, se usa en circuitos en los que se necesita alta estabilidad y bajas perdidas en alta frecuencia.

Utiliza cerámicas de varios tipos para formar el dieléctrico. Existen tipos formados por una sola lámina de dieléctrico, pero también los hay formados por láminas apiladas. Dependiendo del tipo, funcionan a distintas frecuencias, llegando hasta las microondas. Su unidad es el faradio, aunque normalmente se los encuentra en valores de microfaradios, nano faradios, picofaradios “µf, nf, pf”.

2.2.7. Cristal.

Simbología.

FIGURA 13: Imagen de un cristal y su Simbologia

El oscilador, a veces abreviado XTAL u OSC en los esquemáticos, es un circuito electrónico que usa resonancia mecánica de un material cristalino para que generar una señal con una frecuencia precisa. Las frecuencias típicas de oscilación son 4Mhz, 8Mhz, 10Mhz, 20Mhz, 32Mhz y 40 MHz El microcontrolador toma la señal que emite el cristal oscilador y ejecuta una instrucción (o una parte de una) en cada ciclo.

La velocidad máxima de oscilación del cristal que se puede utilizar va a depender del microcontrolador que se esté utilizando. Para saber cual es la máxima velocidad del microcontrolador que escogió se debe revisar la hoja de datos o datasheet[12].

Para que el oscilador funcione es necesario en algunos casos utilizar dos condensadores los cuales dependen del cristal escogido. Usualmente se utilizan condensadores cerámicos de 22pF.

2.2.8. CI 7805 y 7812.

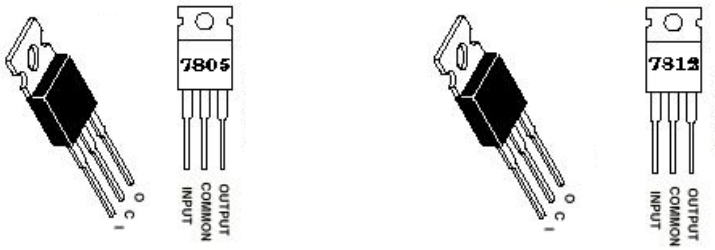

FIGURA 14: Imagen y descripción del CI 7805 y 7812

Circuito Equivalente.

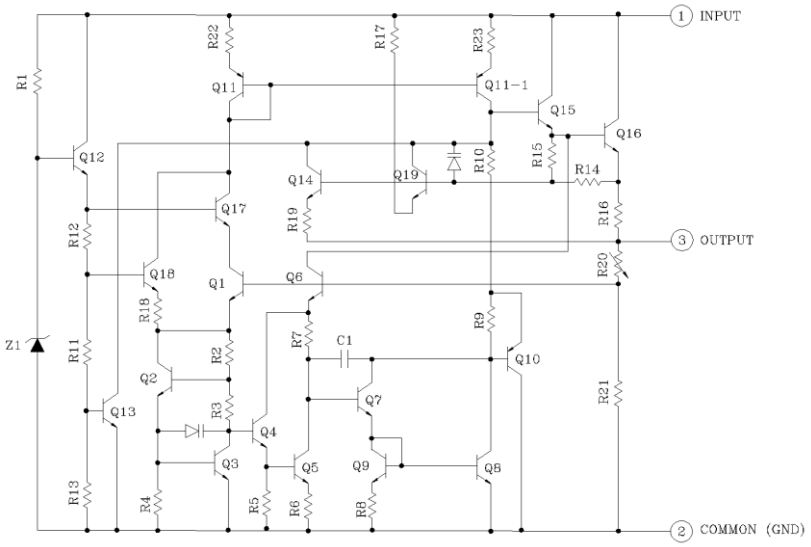

FIGURA 15: Circuito equivalente tanto del CI 7805 y 7812

[12] La hoja de datos es el documento oficial que entrega el fabricante del microcontrolador en donde aparecen las especificaciones técnicas.

CHARACTERISTIC	SYMBOL	TEST CIRCUIT	TEST CONDITION		MIN.	TYP.	MAX.	UNIT
Output Voltage	V_{OUT}	1	T_j=25℃, I_{OUT}=100mA		4.8	5.0	5.2	V
Input Regulation	Reg line	1	T_j=25℃	7.0V≦V_{IN}≦25V	-	3	100	mV
				8.0V≦V_{IN}≦12V	-	1	50	
Load Regulation	Reg load	1	T_j=25℃	5mA≦I_{OUT}≦1.4A	-	15	100	mV
				250mA≦I_{OUT}≦750mA	-	5	50	
Output Voltage	V_{OUT}	1	7.0V≦V_{IN}≦20V 5.0mA≦I_{OUT}≦1.0A, Po≦15W		4.75	-	5.25	V
Quiescent Current	I_B	1	T_j=25℃, I_{OUT}=5mA		-	4.2	8.0	mA
Quiescent Current Change	ΔI_B	1	7.0V≦V_{IN}≦25V		-	-	1.3	mA
Output Noise Voltage	V_{NO}	1	Ta=25℃, 10Hz≦f≦100kHz I_{OUT}=50mA		-	50	-	μV_{rms}
Ripple Rejection Ratio	RR	1	f=120Hz, 8.0V≦V_{IN}≦18V, I_{OUT}=50mA, T_j=25℃		62	78	-	dB
Dropout Voltage	V_D	1	I_{OUT}=1.0A, T_j=25℃		-	2.0	-	V
Short Circuit Current Limit	I_{SC}	1	T_j=25℃		-	1.6	-	A
Average Temperature Coefficient of Output Voltage	TC_{VO}	1	I_{OUT}=5mA, 0℃≦T_j≦125℃		-	-0.6	-	mV/℃

TABLA 8: Hoja de Datos Técnicos del CI 7805.

CHARACTERISTIC	SYMBOL	TEST CIRCUIT	TEST CONDITION		MIN.	TYP.	MAX.	UNIT
Output Voltage	V_{OUT}	1	T_j=25℃, I_{OUT}=100mA		11.5	12.0	12.5	V
Input Regulation	Reg line	1	T_j=25℃	14.5V≦V_{IN}≦30V	-	10	240	mV
				16V≦V_{IN}≦22V	-	3	120	
Load Regulation	Reg load	1	T_j=25℃	5mA≦I_{OUT}≦1.4A	-	12	240	mV
				250mA≦I_{OUT}≦750mA	-	4	120	
Output Voltage	V_{OUT}	1	14.5V≦V_{IN}≦27V 5.0mA≦I_{OUT}≦1.0A, Po≦15W		11.4	-	12.6	V
Quiescent Current	I_B	1	T_j=25℃, I_{OUT}=5mA		-	4.3	8.0	mA
Quiescent Current Change	ΔI_B	1	14.5V≦V_{IN}≦30V		-	-	1.0	mA
Output Noise Voltage	V_{NO}	1	Ta=25℃, 10Hz≦f≦100kHz I_{OUT}=50mA		-	90	-	μV_{rms}
Ripple Rejection Ratio	RR	1	f=120Hz, 15V≦V_{IN}≦25V I_{OUT}=50mA, T_j=25℃		55	71	-	dB
Dropout Voltage	V_D	1	I_{OUT}=1.0A, T_j=25℃		-	2.0	-	V
Short Circuit Current Limit	I_{SC}	1	T_j=25℃		-	0.7	-	A
Average Temperature Coefficient of Output Voltage	TC_{VO}	1	I_{OUT}=5mA, 0℃≦T_j≦125℃		-	-1.6	-	mV/℃

TABLA 9: Hoja de Datos Técnicos del CI 7812.

2.2.9. TRANSISTOR NPN "2N3904".

Simbología.

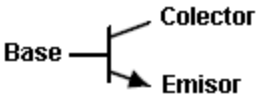

FIGURA 16: Imagen y Descripcion de los pines del Transistor 2N3904

Symbol	Parameter	Value	Units
V_{CEO}	Collector-Emitter Voltage	40	V
V_{CBO}	Collector-Base Voltage	60	V
V_{EBO}	Emitter-Base Voltage	6.0	V
I_C	Collector Current - Continuous	200	mA
T_J, T_{stg}	Operating and Storage Junction Temperature Range	-55 to +150	°C

TABLA 10: Características Técnicas del Transistor 2N3904

h_{FE}	DC Current Gain	I_C = 0.1 mA, V_{CE} = 1.0 V I_C = 1.0 mA, V_{CE} = 1.0 V I_C = 10 mA, V_{CE} = 1.0 V I_C = 50 mA, V_{CE} = 1.0 V I_C = 100 mA, V_{CE} = 1.0 V	40 70 100 60 30	300	
$V_{CE(sat)}$	Collector-Emitter Saturation Voltage	I_C = 10 mA, I_B = 1.0 mA I_C = 50 mA, I_B = 5.0 mA		0.2 0.3	V V
$V_{BE(sat)}$	Base-Emitter Saturation Voltage	I_C = 10 mA, I_B = 1.0 mA I_C = 50 mA, I_B = 5.0 mA	0.65	0.85 0.95	V V

TABLA 11: Características Técnicas del Transistor 2N3904, continuación

t_d	Delay Time	V_{CC} = 3.0 V, V_{BE} = 0.5 V,		35	ns
t_r	Rise Time	I_C = 10 mA, I_{B1} = 1.0 mA		35	ns
t_s	Storage Time	V_{CC} = 3.0 V, I_C = 10mA		200	ns
t_f	Fall Time	I_{B1} = I_{B2} = 1.0 mA		50	ns

TABLA 12: Características de Interrupción.

2.2.9.1 Diodo rectificador 1N4148.

Simbología.

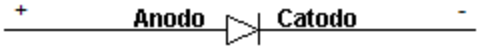

FIGURA 17: Imagen y simbología del diodo 1N4148

Parameter	Test Conditions	Type	Symbol	Min	Typ	Max	Unit
Forward voltage	I_F=5mA	1N4448	V_F	0.62		0.72	V
	I_F=10mA	1N4148	V_F		0.86	1	V
	I_F=100mA	1N4448	V_F		0.93	1	V
Reverse current	V_R=20V		I_R			25	nA
	V_R=20V, T_j=150?		I_R			50	μ A
	V_R=75V		I_R			5	μ A
Breakdown current	I_R=100μ A,t_p/T=0.01,t_p=0.3ms		$V_{(BR)}$	100			V
Diode capacitance	V_R=0, f=1MHz, V_{HF}=50mV		C_D			4	pF
Rectification efficiency	V_{HF}=2V, f=100MHz		? $_R$	45			%
Reverse recovery time	I_F= I_R=10mA, i_R=1mA		t_{rr}			8	ns
	I_F=10mA, V_R=6V, i_R=0.1× I_R, R_L=1000		t_{rr}			4	ns

TABLA 13: Características Eléctricas del diodo 1N4148

Parameter	Test Conditions	Type	Symbol	Value	Unit
Repetitive peak reverse voltage			V_{RRM}	100	V
Reverse voltage			V_R	75	V
Peak forward surge current	t_p=1μ s		I_{FSM}	2	A
Repetitive peak forward voltage			I_{FRM}	500	mA
Forward current			I_F	300	mA
Average forward current	V_R=0		I_{FAV}	150	mA
Power dissipation	l=4mm T_L= 25?		P_V	500	mW
Junction temperature			T_j	175	?
Storage temperature range			T_{stg}	-65~+175	?

TABLA 14 Valores máximos soportados en el diodo 1N4148

2.2.9.1 Diodo Zener 1N4733.

Simbología.

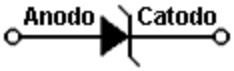

FIGURA 18: Imagen y simbología del diodo zener 1N4733

Tensión Nominal	Test de Corriente	Máxima Corriente de fuga inv		Aumento de Corriente	Máxima corriente regulatoria[2]
		I_R µA	V_R V		
5.1V	49	10	1	890mA	178Ma

TABLA 15: Características Técnicas del Diodo Zener 1N4733

2.3 ANÁLISIS ECONÓMICO DEL TEMA.

GASTOS DE MATERIALES E INSTRUMENTOS.					
ITEM	**DESCRIPCION**	**UNIDAD**	**CANTIDAD**	**PRECIO UNITARIO ($)**	**PRECIO TOTAL($)**
01	Gastos de Materiales.	Global	1	400	400
02	Gastos de Materiales para pruebas.	Global	1	250	250
03	Gastos logístico	Global	1	50	50
04	Fotocopias	Global	500	0.02	10
05	Horas de uso en Internet.	Global	50	0.80	40
06	Impresión de consultas.	Global	100	0.07	7
07	Impresiones del documento.	Global	600	0.15	90
08	Gastos en transporte	Global	1	50	50
				Total:	$ 897

TABLA 16: Gastos para materiales e instrumentos para la construcción del PLC.

2.4 MATERIALES.

1 PIC16F877 y Zócalo 40 Pines.

FIGURA 19: PIC16F877A y un Socalo de 40 Pines

8 Opto transistores 4N27 y Zócalos 6 Pines.

FIGURA 20: Opto Transistor 4N27 y un Zocalo de 6 Pines

9 Transistores 2N3904.

2 DB9 Hembra.

FIGURA 21: Transistor 2N3904 y DB9 Hembra para Circuitos Impresos

Percloruro Férrico.

FIGURA 22: Percloruro Ferrico.

Baquelita de Fibra de vidrio.

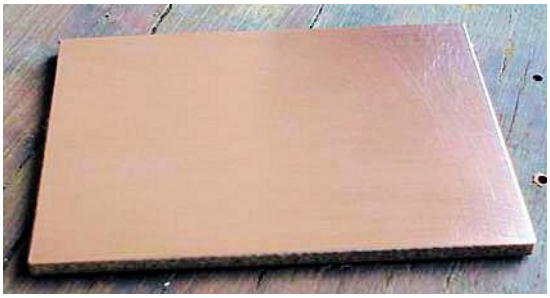

FIGURA 23: Baquelita De Fibra de Vidrio

10 Resistencia de 10K, 4.7K, 10Ω.

2 Tiras de Pines.

FIGURA 24: Recistencia y Tira de Pines

Cable Para Comunicación Serial Norma RS232.

FIGURA 25: Cable Serial Para la Norma RS232

2 Condensadores Cerámicos de 22pF y 5 Electrolíticos de 100, 10µF.

FIGURA 26: Condensador Ceramico y Condensador Electrolitico

1 MAX232 y Zócalo de 16 Pines.

FIGURA 27: CI MAX232 y Socalo de 16 Pines

1 Cristal de 4Mhz

1 Pantalla LCD.

FIGURA 28: Un Cristal y una Pantalla LCD

2 Fusibles y 2 Porta fusible.

FIGURA 29: Fusibles Y Porta Fusibles

2 Diodo Zener 5.1V

2 Diodo Rápido 1N4148.

FIGURA 30: Un diodo Zener 5.1V y un Diodo de respuesta Rapida

1 Relé 12V Doble Contacto.

8 Relés 12V

FIGURA 31: Un Relé Doble contacto y un rele de un contacto NA y NC

3. PROCESO METODOLOGICO.

3.1. METODOLOGÍA PARA EFECTUAR EL DISEÑO, CONSTRUCCION Y PROGRAMACION DE UN PLC A BASE DE UN MICROCONTROLADOR PIC16F877.

3.1.1 Método utilizado para realizar el diseño del proyecto.

Para el diseño del proyecto se procedió con el método investigativo, el cual consistió en buscar información acerca de los PLC y de los microcontroladores para con esto tener una idea de cómo se podría comenzar.

3.1.1.1 Características del PLC a Construir.

- 8 Entradas Digitales estas saldrán del Puerto B del Microcontrolador, en cada entrada deberá ingresar una tensión de 12V DC.
- 8 Salidas Digitales estas saldrán del Puerto D del Microcontrolador, las salidas deberán soportar tensiones tanto continuas como Alternas de hasta 120V 10A como Máximo en cada Salida.
- 8 Entradas o Salidas Auxiliares analógicas y digitales también pudiendo ser utilizadas como comparadores, estos Funcionan a 5V DC
- 1 Interface serial con la cual se pueda comunicar el PLC con una PC.
- 1 Modulo LCD.
- 1 Programador integrado en el PLC para poder cargarle el programa que necesitemos en el microcontrolador.
- 2 Módulos estabilizadores de Tensión uno a 5V DC 2 A y otro a 12V DC 2 A.

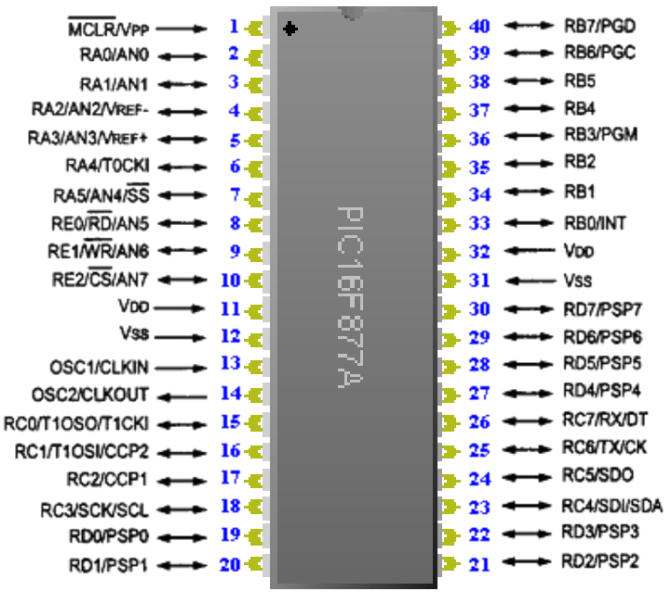

FIGURA 32: PIC16F877A, descripción de pines.

3.1.1.2 Microcontrolador a Utilizar.

El microcontrolador que se utilizó de acuerdo a las especificaciones del PLC es el PIC16F877 por tener todas las características que se necesitan para construir el PLC, estas son:

- 3 Puertos Digitales de 8 bits cada uno de ellos, B, C y D pudiendo ser estos utilizados para entradas o salidas, comunicación con el módulo LCD, en caso de comunicación serial los pines apropiados para esta función están en el Puerto C y los pines son: C6 como transmisor y C7 como Receptor, y para el control de motores paso a paso serán en los pines C0 y C1.

- 2 Puertos Comparadores pudiendo también ser utilizados como digitales, el Puerto A es de 8 Bits y el Puerto E es de 4 Bits.

- 1 Memoria FLASH (14 Bit - Words) con una capacidad de 8 KB.

- 1 Memoria de Datos (RAM) – con una capacidad de 368 Bytes.

- 1 MCLR Para reinicializar las funciones del microcontrolador.

- 2 Entradas de alimentación de 5V DC estas consumen 8 mA entre las dos ya que es la corriente que consume el microcontrolador.
- 2 Pines destinados para un oscilador externo pudiendo soportar este un cristal de hasta 20Mhz.
- 1 Memoria EEPRON con una capacidad de 256 Bytes.
- 14 Interrupciones.
- 3 temporizadores.
- 35 Instrucciones posibles a configurar.

3.1.1.3 Proceso del Diseño.

Luego de haber escogido el tipo de microcontrolador a utilizar y que tipo de PLC será el que se construya, se procedió a acoplar el microcontrolador según las características del PLC a construir.

Entradas Digitales.- En el microcontrolador se utilizó el puerto B el cual tiene 8 bits es decir 8 pines como entradas digitales, cada pin admite un máximo de 5V DC y el PLC debe admitir en cada entrada 12V DC, para reducir los 12 V que deben ingresar al circuito a 5 V que es la tensión que soporta el microcontrolador, se utilizó una resistencia para atenuar de 12V a 5V con lo cual podrá funcionar correctamente el microcontrolador, además se usó un OPTO TRANSISTOR, el 4N27 cuya función será la de proteger al PIC de tensiones mal rectificadas y sobre tensiones en caso de que se utilice una fuente externa para las entradas, y como señalización de tensión por cada entrada se utilizó un diodo Led, por cada una de ellas, a continuación se verá el esquema de cómo va ubicada una de las entradas del PLC.

Datos:
T.requerida=3V
I. requerida=10mA
R.requerida=?
T.total=12V

$$V1=Vt-V2 \Rightarrow V1=12V-3V \Rightarrow V1=9V$$

$$R=\frac{V1}{I1} \Rightarrow R=\frac{9V}{0.01A} \quad R=900\Omega$$

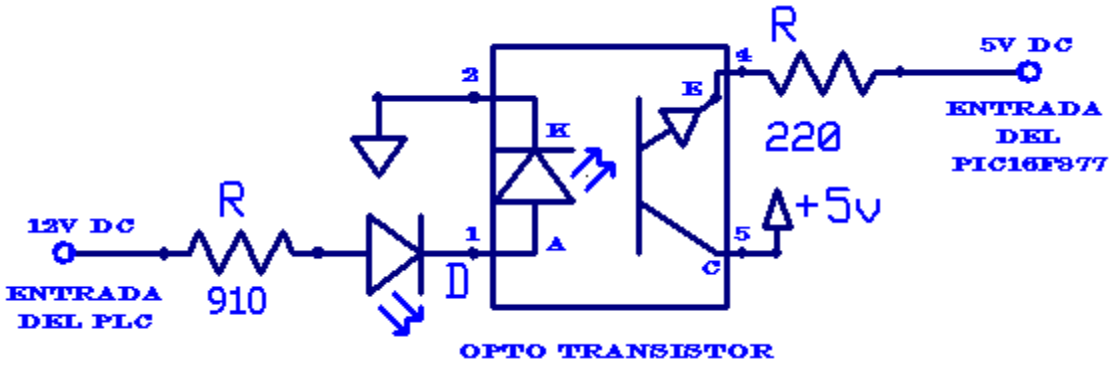

FIGURA 33: Esquema de conexión de una entrada digital del PLC

Salidas Digitales.- En el microcontrolador se utilizó el puerto D el cual tiene 8 bits es decir 8 pines utilizados como salidas digitales, en cada pin vamos a tener una tensión de 5V DC, pero de acuerdo a las especificaciones del PLC en estas salidas pueden variar la tensión en un rango desde 0V hasta 120V pudiendo esta ser Alterna o Continua y además que soporte una corriente máxima de 10A, para lograr esto en la investigación efectuada se encontró que podría utilizar relés que sirvieran como un switch controlado por una tensión DC que lo activa y hace que se cierre el circuito con lo que la tensión externa al circuito suministrada en el extremo A del relé circule libremente hacia el extremo B para activar algún tipo de sistema o circuito al que necesitemos activar. El relé que se utilizó necesitó una tensión de activación de 12V DC debido a que no se encontró relés a 5V, que es la tensión nominal en cada pin del microcontrolador, para activar dicho relé se acopló un transistor por sus especificaciones técnicas fue el 2N3904 de respuesta rápida y muy utilizado para este tipo de proyectos, este utiliza una tensión de saturación en su base de 1mA y dándole al PIC un promedio de tensión de salida de 4.7V que es la tensión mínima que arroja en cada pin, se calculó la resistencia apropiada para activar el transistor basándose en la formula de la ley de OHM a continuación expuesta.

$$R = \frac{V}{I}$$

$$R = \frac{4.7V}{1mA} = \quad R = \frac{4.7V}{0.001A} = \quad R = 4.700\Omega \quad R = 4.7K\Omega$$

Luego de haber calculado el valor de la resistencia se la colocó en serie entre el pin del microcontrolador y la base del transistor. El cálculo para hallar el valor de la resistencia es debido a que la tensión de saturación de la base del transistor es de 0.85V y el estado lógico cero en los pines del microcontrolador es de 0 a 1V que es lo necesario para que el transistor se sature y active al relé. A continuación se verá como quedará el circuito en una de las salidas del PLC.

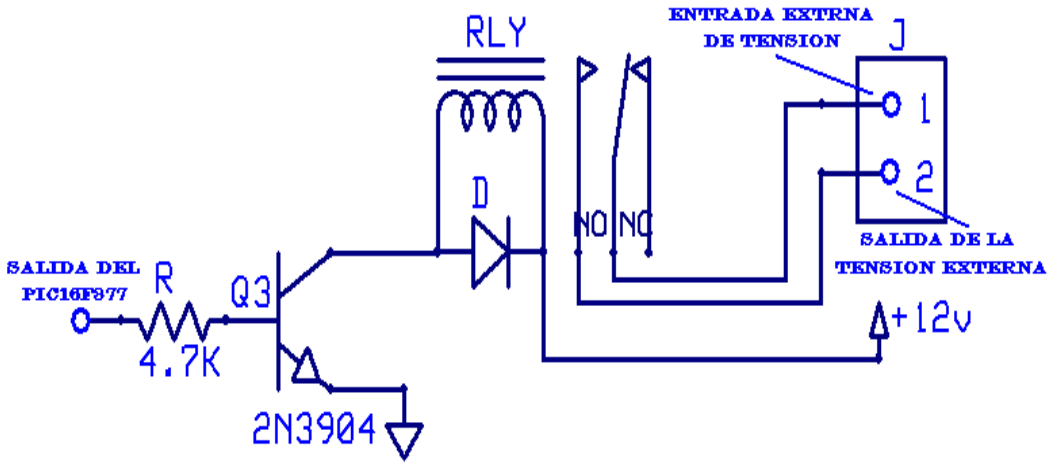

FIGURA 34: Esquema de conexión de una salida digital del PLC

Interface Serial.- El microcontrolador posee una interface de comunicación serial con la PC llamada USART, fue diseñada para convertir las señales que maneja la CPU y transmitirlas al exterior. Las USART deben resolver problemas tales como la conversión de voltajes internos del DCE con respecto al DTE, gobernar las señales de control, y realizar la transformación desde el bus de datos de señales en paralelo a serie y viceversa. Debe ser robusta y deberá tolerar circuitos abiertos, cortocircuitos y escritura simultánea sobre un mismo pin.

El microcontrolador emite señales de lógica digital TTL que emite voltajes de +5V cuando esta en un nivel alto (1 lógico) y 0V cuando está en un estado lógico bajo (0 lógico), por lo que necesariamente se debe implementar un driver que cumpla la función de convertir niveles de la norma RS232 a la norma TTL, el CI. MAX232 es el driver que cumple con estas características ya que internamente cambia los niveles de tensión gracias a un juego de capacitores, además tiene inversores para que lleguen valores verdaderos al microcontrolador es decir cuando el Driver MAX232 posee a la entrada +12VDC, le arrojará al PIC 0VDC y cuando posea -12VDC la entrada del PIC será 5VDC, además de esto para poder comunicar al microcontrolador con la PC existe el modo de comunicación asíncrona que consiste encapsulados en tramas como se lo indica en el siguiente figura 35.

FIGURA 35: Encapsulado de las Tramas al enviar datos

Primero se envía un bit de start, a continuación los bits de datos (primero el bit de mayor peso) y finalmente los bits de STOP.

El número de bits de datos y de bits de Stop es uno de los parámetros configurables, así como el criterio de paridad par o impar para la detección de errores. Normalmente, las comunicaciones serie tienen los siguientes parámetros: 1 bit de Start, 8 bits de Datos, 1 bit de Stop y sin paridad.

A continuación se verá un grafico en el cual se muestra como esta compuesta una información de 8 Bits de datos, las diferentes tensiones y como se invierten los datos en este proceso de comunicación.

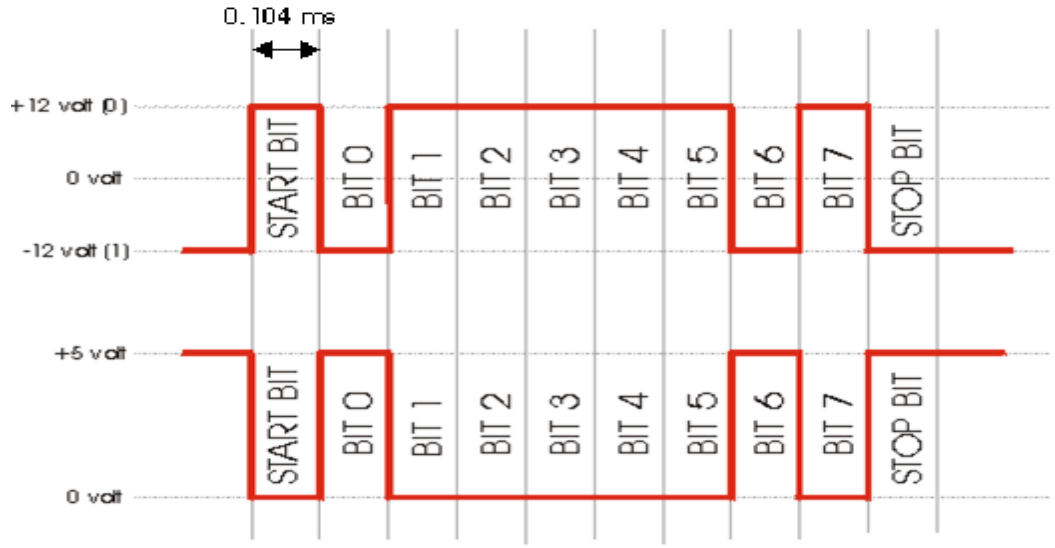

FIGURA 36: Diferencia entre la estrucura de una trama normal y otra utilizando el protocolo RS232.

PROTOCOLO RS-232.- Está diseñado para distancias de hasta 15 metros o menos, aunque cuando se utiliza el Driver Max 232 en comunicaciones entre PC y microcontroladores, se puede extender distancias de hasta 200 metros, y para velocidades de comunicación bajas, de no más de 20 KB. A pesar de ello, muchas veces se utiliza a mayores velocidades con un resultado aceptable. La interfaz puede trabajar en comunicación asíncrona o síncrona y tipos de canal simplex, half dúplex o full dúplex.

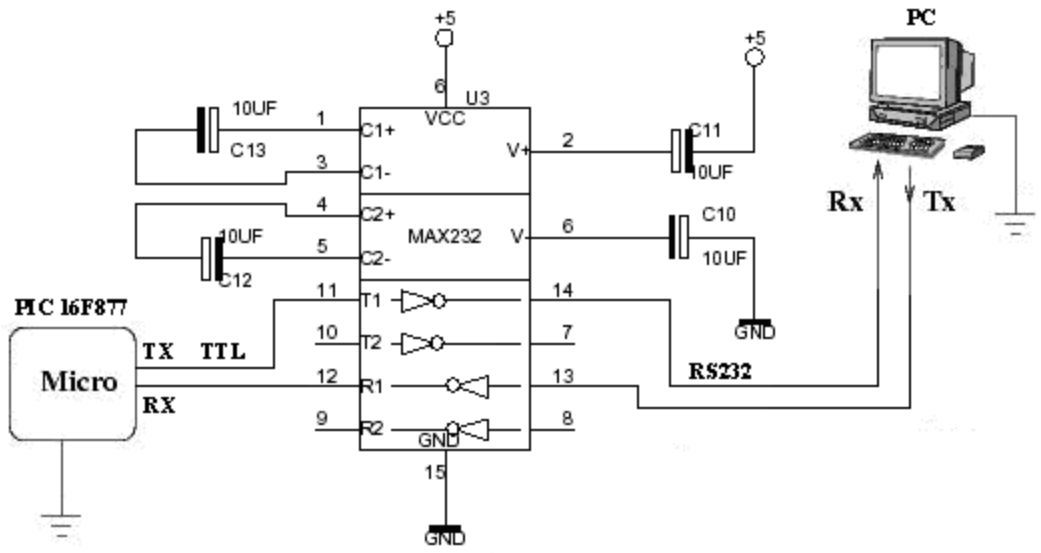

FIGURA 37: Diagrama de Conexión interna del MAX232

Entradas o Salidas Auxiliares analógicas, convertidor A/D.- El puerto A del PIC16F877A se caracteriza por tener 6 pines (pin 2=A0, 3=A1, 4=A2, 5=A3, 6=A4, 7=A5) los cuales pueden ser utilizados como entradas o salidas para señales analógicas este tipo de puerto se lo utiliza mucho como comparadores de señales, este internamente tiene un convertidor Analógico/Digital, pudiendo ser utilizado como entradas o salidas digitales.

Modulo LCD.- En el microcontrolador se utilizó los pines 15, 16, 17, 18, 23, 24 del puerto C cuya función es la de enviar información al LCD para poder visualizarla, utilizado un LCD 2X16 con Backlight (luz color Azul) letras Blancas, este tiene 16 pines. Los pines 1(GND), 5(R/W) y 16(K) están conectados a tierra, mientras que el pin 2 se conecta a la fuente de 5V DC, el pin 3 (Vo) es el ajuste de contraste de cristal liquido de contrastando de 0 a +5V, para lograr el contraste es necesario utilizar un potenciómetro de 10KΩ (recomendado por el Fabricante) teniendo en un extremo del potenciómetro +5V, en el otro extremo conectado a GND y el extremo central irá conectado con el pin 3. En el pin 4(Selección del registro, control/datos) irá conectado al pin 24 del microcontrolador, el pin 6(E) sirve para habilitar o deshabilitar el módulo y está conectado con el pin 23 del microcontrolador. La comunicación

con el microcontrolador se puede realizar a 8 o 4 Bits en este caso se utilizó solo 4 bits para ahorrar pines en el PIC estos 4 pines del modulo LCD son 11(D4) conectado al pin 15 del microcontrolador, 12(D5) conectado al pin 16 del microcontrolador, 13(D6) conectado al pin 17 del microcontrolador, 14(D7) conectado al pin 18 del microcontrolador. El pin 15(Ánodo) es Alimentación del Backlight este va conectado a una resistencia de 10Ω la cual va conectada a +5V. A continuación en la figura 38 se muestra el esquema de conexión de un Modulo LCD 2X16 con el PIC16F877A.

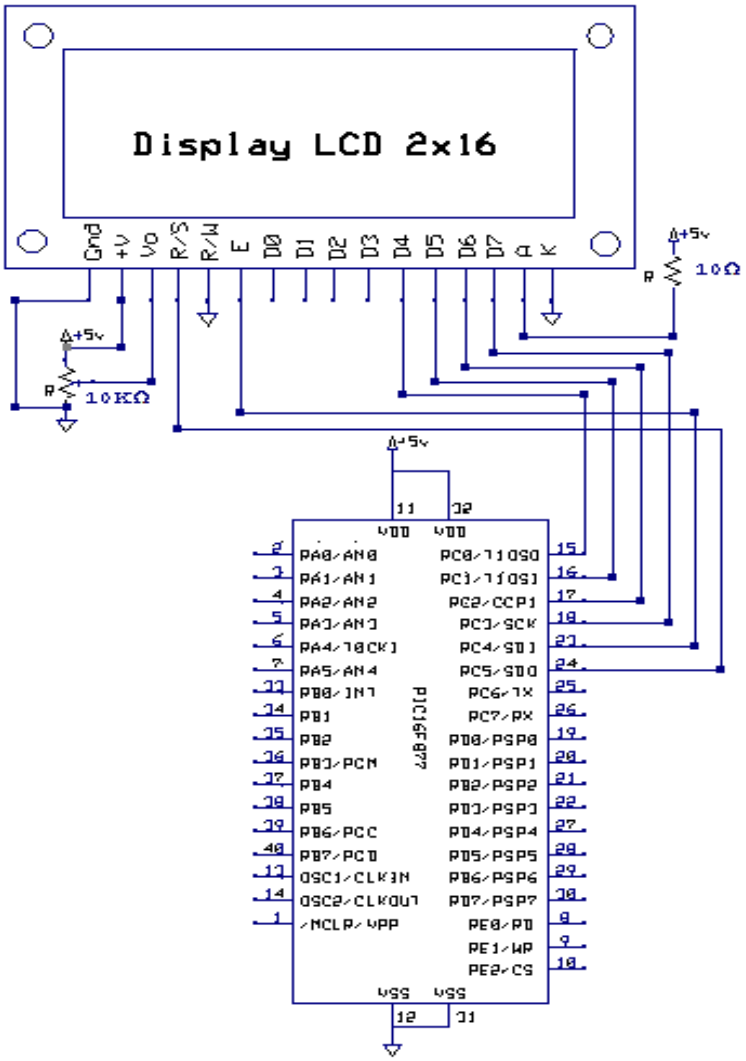

FIGURA 38: Esquema de Conexión de un Modulo LCD 2X16

Programador.- Para poder cargarle un programa al microcontrolador se necesitó un circuito que lo programe y además que este integrado al PLC de tal manera que no se necesite retirar el PIC del circuito con todas estas características se investigó acerca de los diferentes tipos de programadores con lo que se llegó a la conclusión de utilizar un programador de tipo ISP[13] este funciona utilizando la HVP[14] que consiste en aplicar un voltaje VIHH comprendido entre los 12 a 14V, al pin Vpp/MCLR Luego de llevar al PIC al estado de programación, se comienza la transmisión serial por medio de los pines **PGC**[15] correspondiente al pin 39 y **PGD**[16] correspondiente al pin 40 del microcontrolador, aparte de las patillas ya mencionadas se necesita energizar el microcontrolador por medio de los pines **VDD**[17] (32) y **VSS**[18] (31).

Para poder acoplar este circuito programador al PLC se optó por utilizar diodos de silicio de respuesta rápida cuya función es de evitar que el voltaje de Programación alto de 12 a 14V dañe a la fuente cuyo valor sin programar será de 0V, además se utilizó un relé para separar el voltaje del circuito con el voltaje de programación. A continuación se presenta un gráfico de la figura 39, de cómo va ubicado el programador en el microcontrolador.

[13] **ISP:** (In-Circuit Serial Programming) Programación Serial en el Circuito.
[14] **HVP:** (High Voltaje Programming) Programación por Alto Voltaje.
[15] **PGC:** Señal de reloj para la programación serial.
[16] **PGD:** Señal de datos para la programación serial.
[17] **VDD:** Voltaje en corriente directa.
[18] **VSS:** Masa, Polo con 0 Voltios o tierra.

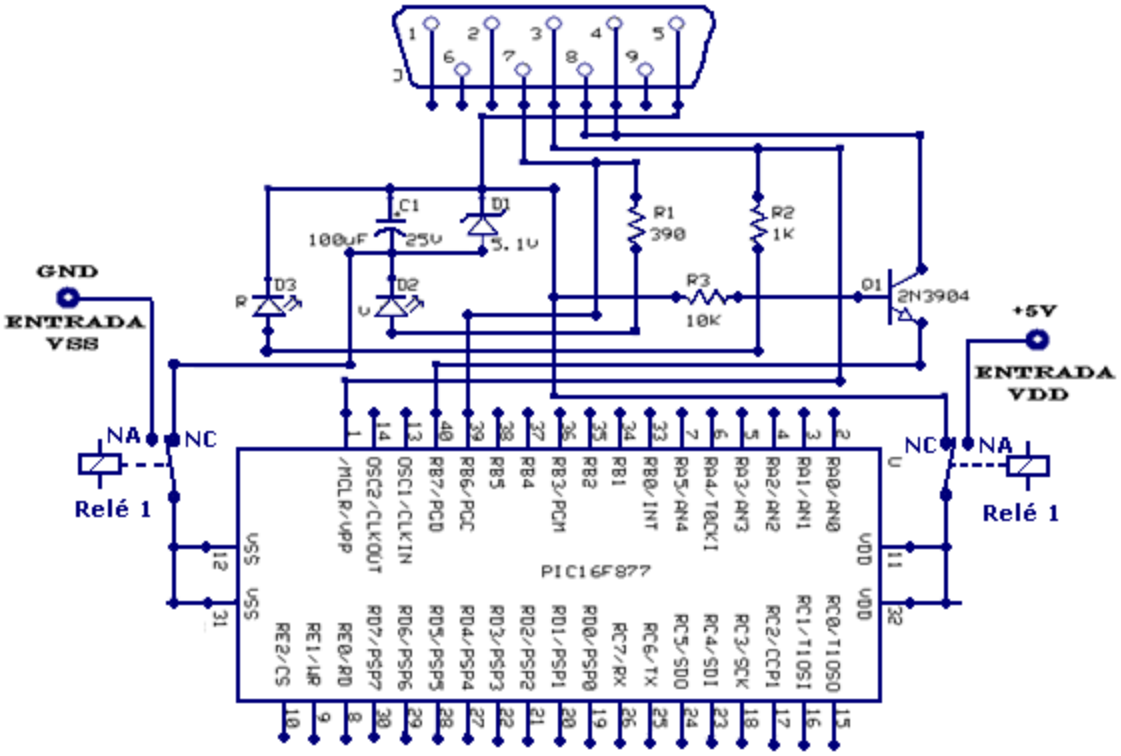

FIGURA 39: Esquema de conexión del Programador de un microcontrolador.

Módulos Estabilizadores de Tensión.- Para estabilizar la tensión del circuito y de esta forma proteger todos sus componentes, se utilizaron dos estabilizadores en paralelo estos son los 7805, este integrado cumple la función de estabilizar la tensión en +5V 1A, en este caso como son 2 integrados se podría manipular corrientes de hasta 2A. El pin 1 es donde va colocada la entrada de tensión, el pin 2 deberá ir a tierra (GND), el pin 3 es la salida de tensión estabilizada, se utilizó un juego de capacitores en paralelo entre el pin 3 y 2 con el fin de que la tensión sea totalmente continua.

En el caso de estabilizar una tensión de 12V, se hace exactamente lo mismo solo se cambia el circuito integrado por el 7812 (Estabilizador a +12V). A continuación se presenta el esquema de conexión de los circuitos estabilizadores de 12 y 5V.

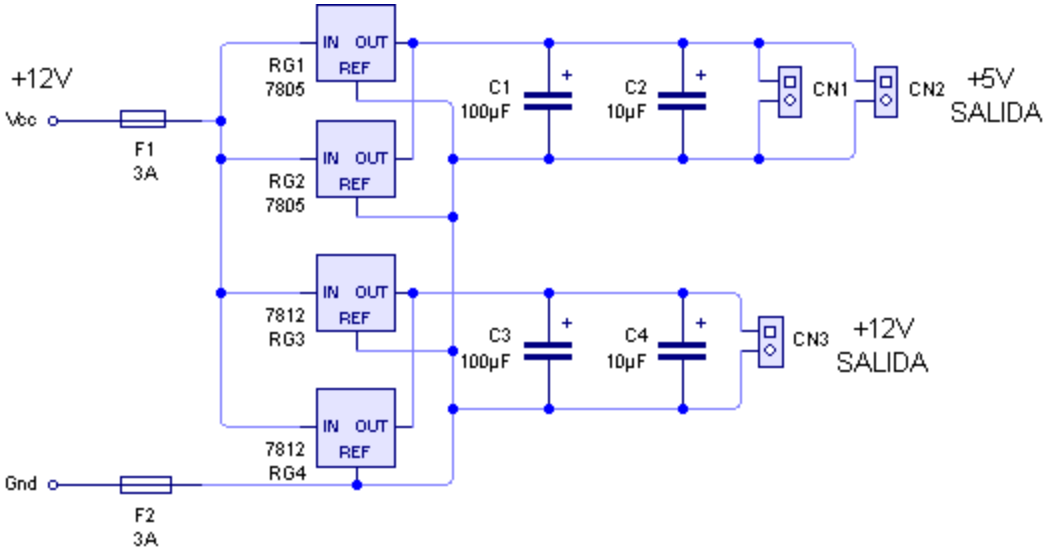

FIGURA 40: Conexión de circuitos estabilizadores de 12 y 5V.

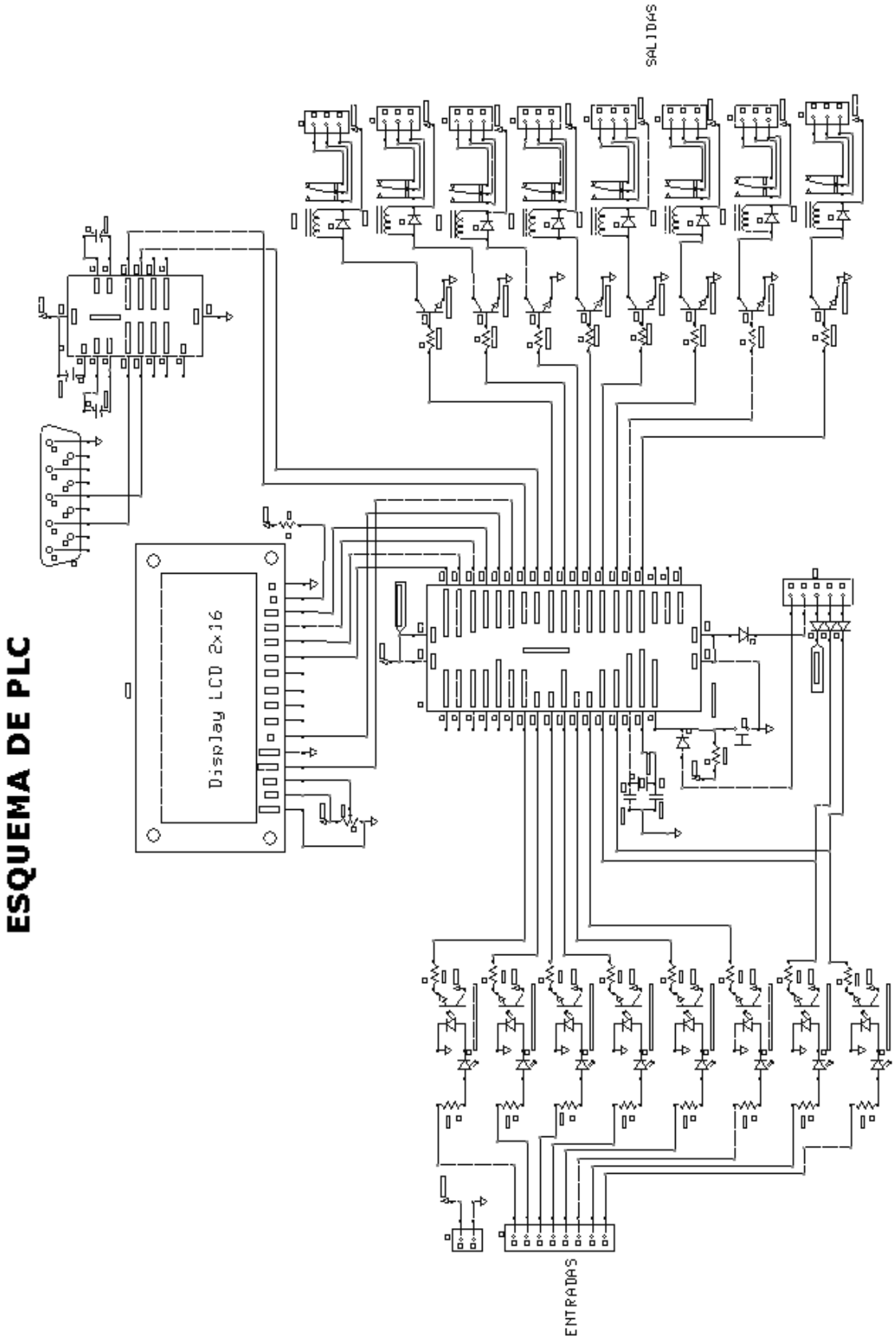

FIGURA 41: esquema completo de conexión del PLC en dos gráficos.

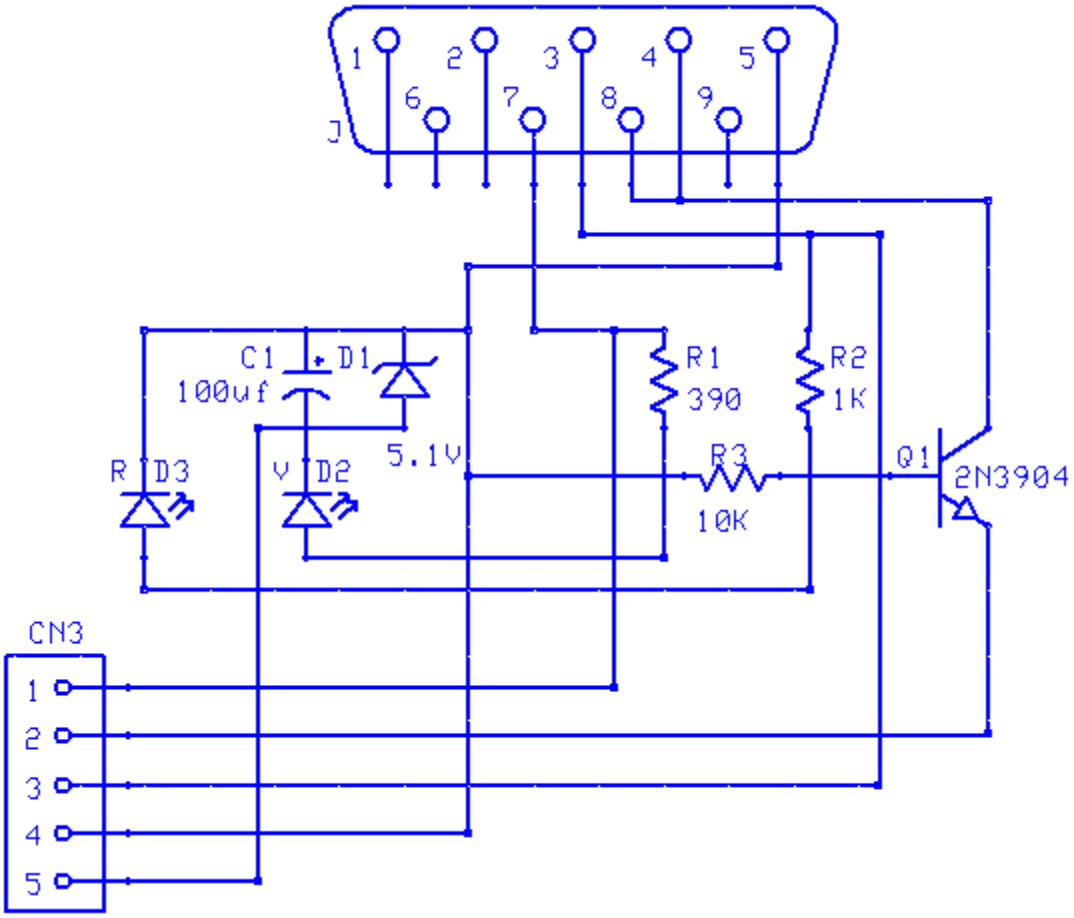

FIGURA 42: Esquema de conexión del Programador utilizado en el PLC.

3.1.1.4 Diseño de las Placas.

Para el diseño de las placas se decidió utilizar el Programa PCB Wizard, este es ideal para el diseño de cualquier tipo de placa además tiene la particularidad de permitirnos ver (Simulación) de cómo nos quedaran las placas con los componentes soldados a esta.

Recomendaciones Técnicas:

- Para evitar que se produzcan arcos voltaicos y cortocircuitos entre líneas se debe realizar una separación mínima entre pistas de 0.05mm para un voltaje de 0 a 15V, y de 0.7mm para un voltaje de 101 a 150V.

- La máxima corriente que circulará por las pistas conductoras determina el ancho de las mismas, por ejemplo, una cinta de 0.5 mm. de ancho soporta aproximadamente 1 amperio, en estos parámetros se determinó el grosor para cada pista según el amperaje que se debe suministrar en cada una de ellas.

- Los discos de cobre para la conexión de las patitas (pines) de los componentes deben ser redondos con diámetro de 3 mm. cuánto mayor sea el área de este punto, será más difícil que se desprenda por el calor. En el caso de circuitos integrados, pueden ser rectangulares y de un ancho adecuado.

- Los orificios para insertar los componentes deben quedar al centro del disco de conexión, con un diámetro de .75 mm. Es recomendable utilizar una broca de 1 mm. y una mayor para componentes que lo requieran.

- Agujeros para los tornillos que fijarán la placa al chasis (fijación horizontal o vertical).

- Evitar formar ángulos rectos para así evitar que se genere ruido ya que este podría afectar seriamente al circuito, además la estética en el diseño de la trama metálica no siempre es lo mejor desde el punto de vista eléctrico.

Tomando en cuenta todas estas características técnicas se procedió a unir los componentes del circuito basándonos en el esquema de conexión previamente realizado.

A continuación se muestran las tres placas diseñadas en PCB Wizard, por cada una de estas se mostrará como quedaran las pistas de cobre, la ubicación de los componentes, la combinación de las anteriores y una simulación como quedara la placa con sus componentes ya soldados.

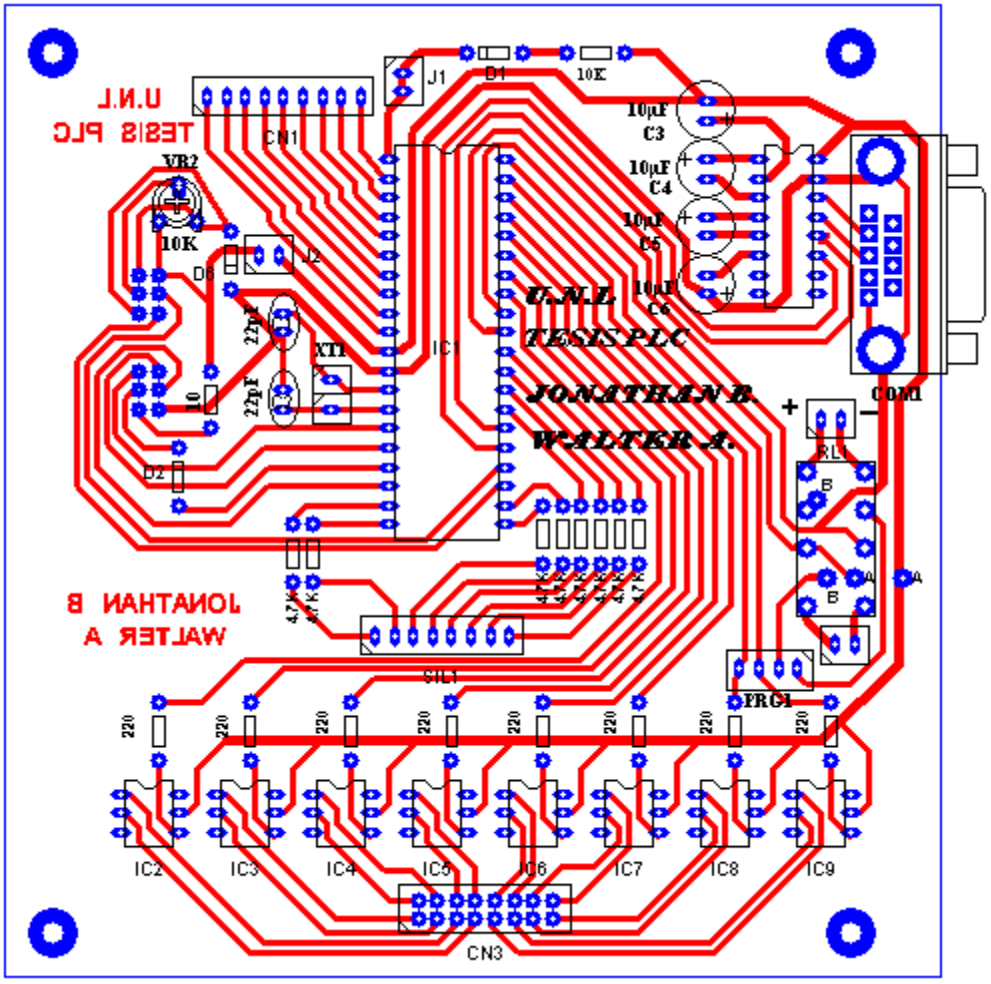

FIGURA 43: Placa CPU del PLC Vista de pistas y componentes.

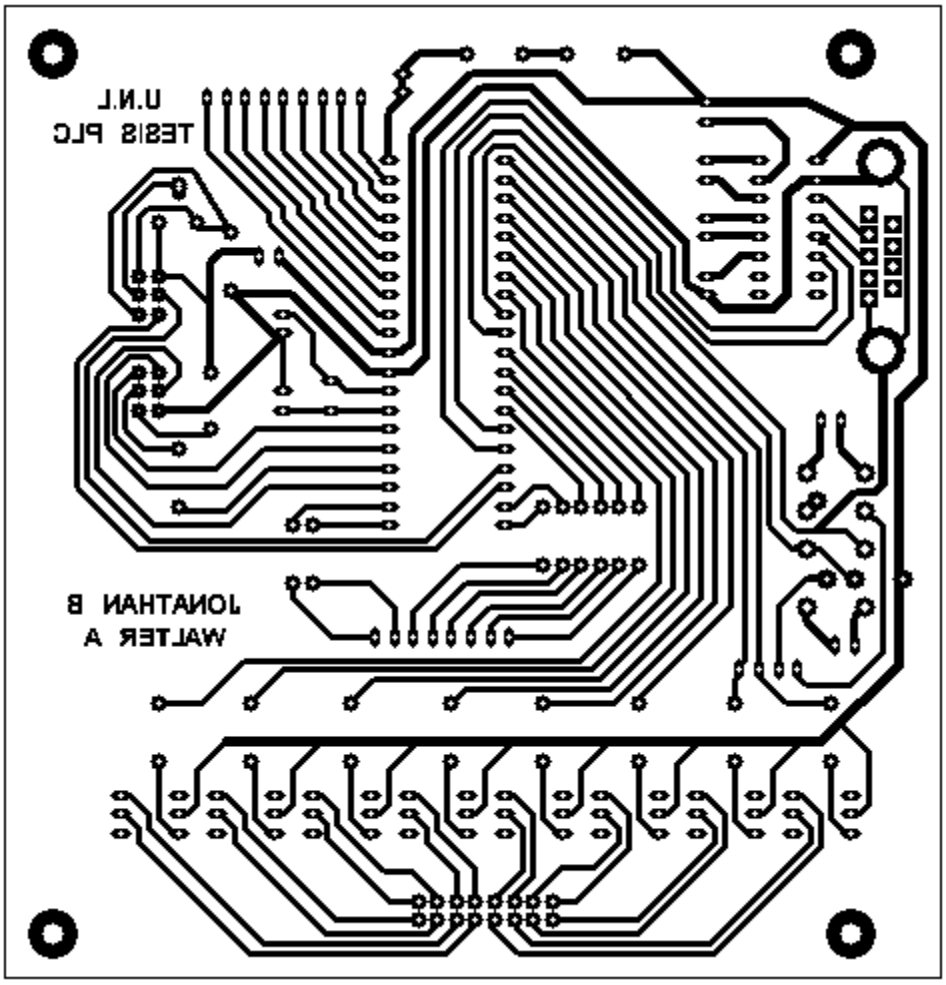

FIGURA 44: Placa CPU del PLC Vista de Pistas.

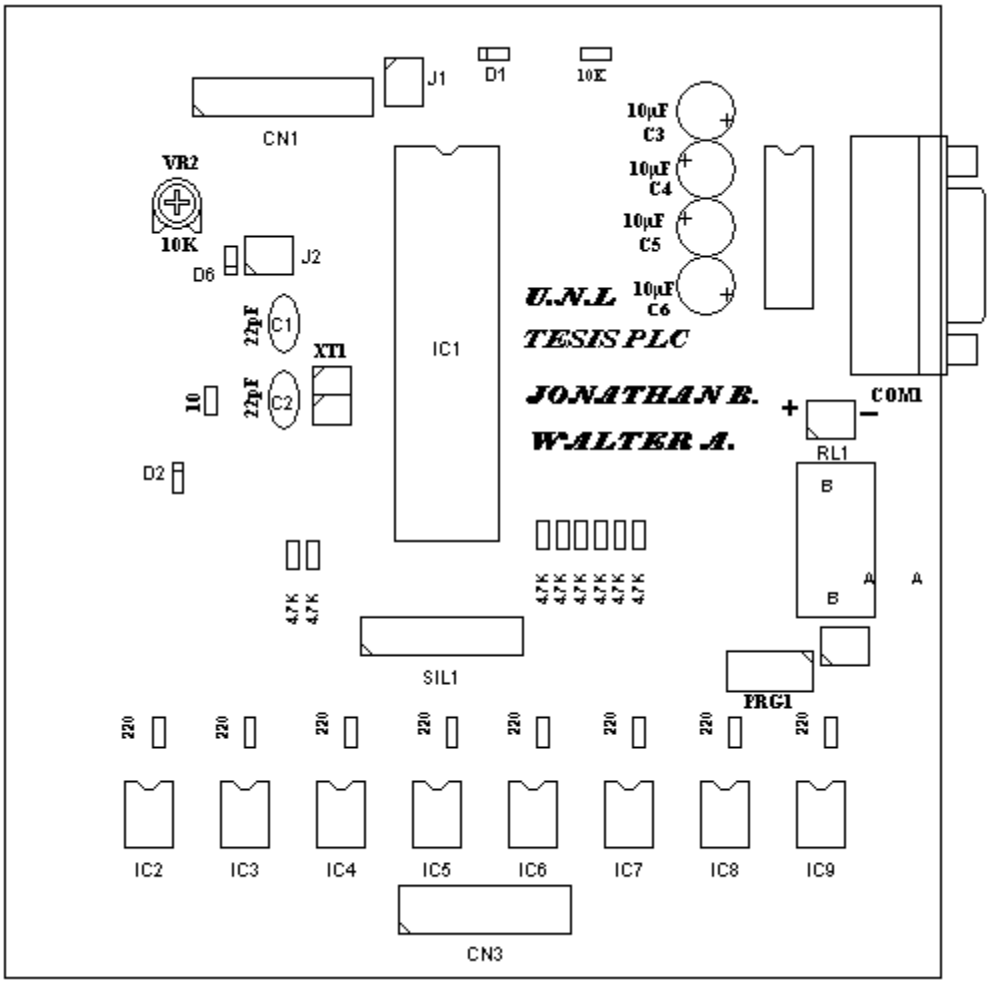

FIGURA 45: Placa CPU del PLC Vista (frontal) de componentes.

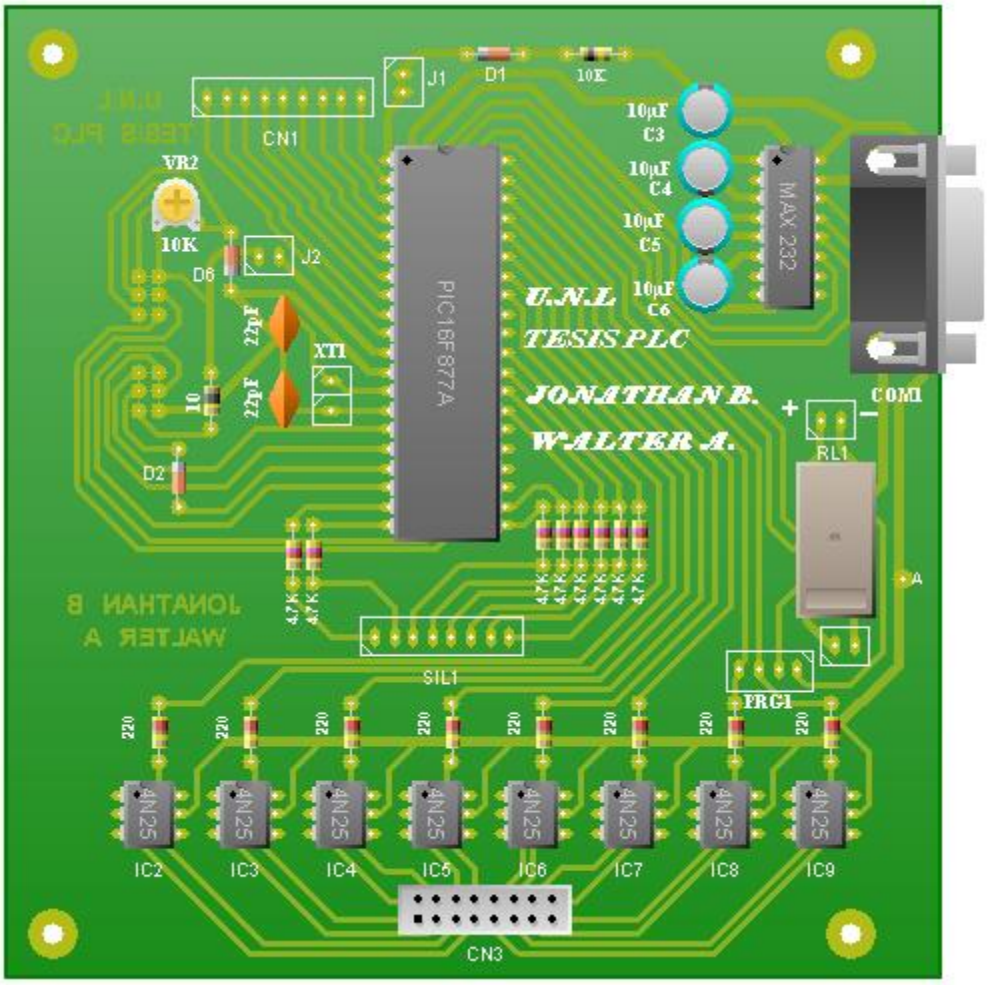

FIGURA 46: Placa CPU del PLC Vista Simulada.

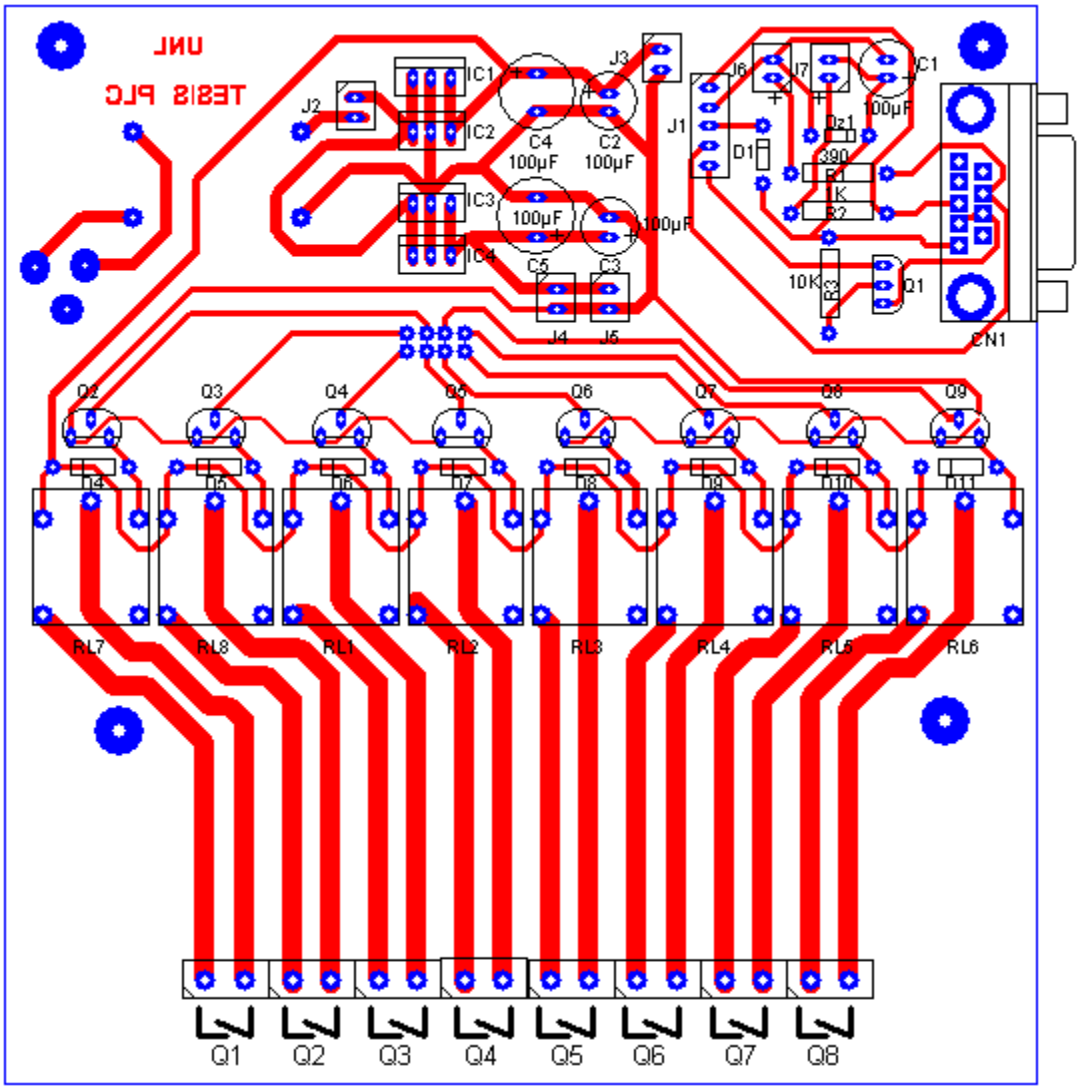

FIGURA 47: Placa Fuente, Programador y Etapa de alto Voltaje del PLC Vista de pistas y componentes.

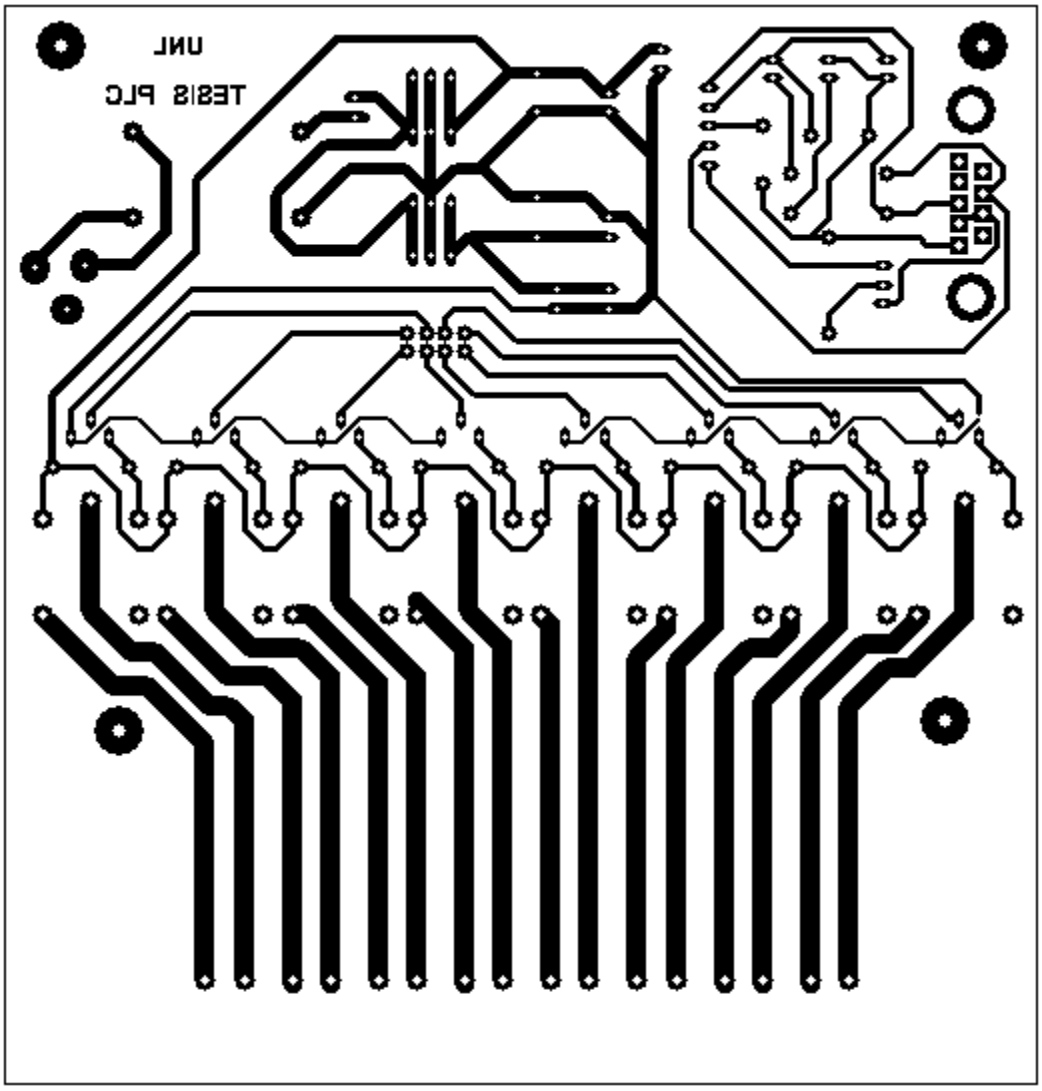

FIGURA 48: Placa Fuente, Programador y Etapa de alto Voltaje del PLC Vista de pistas.

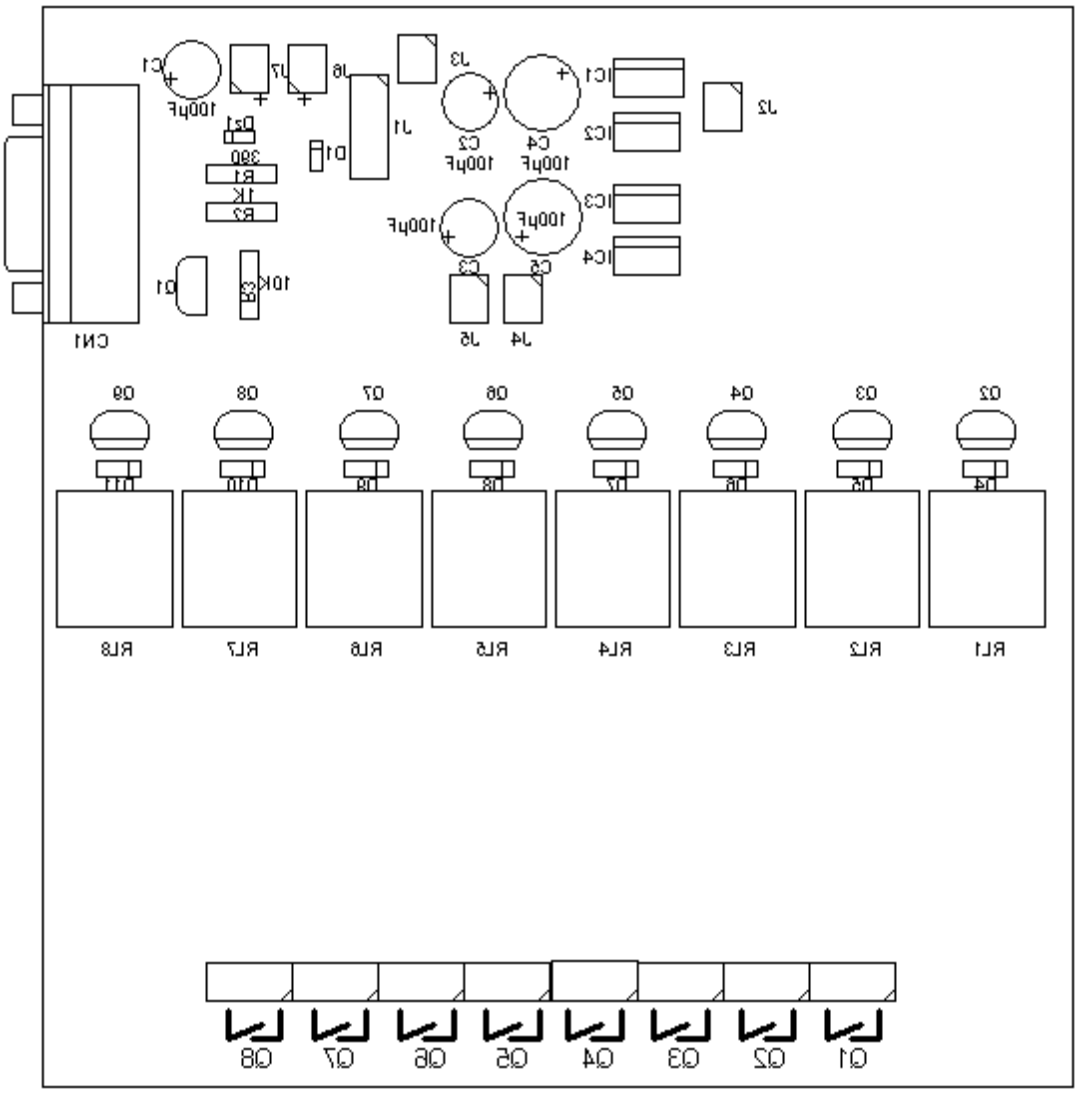

FIGURA 49: Placa Fuente, Programador y Etapa de alto Voltaje del PLC, Vista (frontal) de componentes.

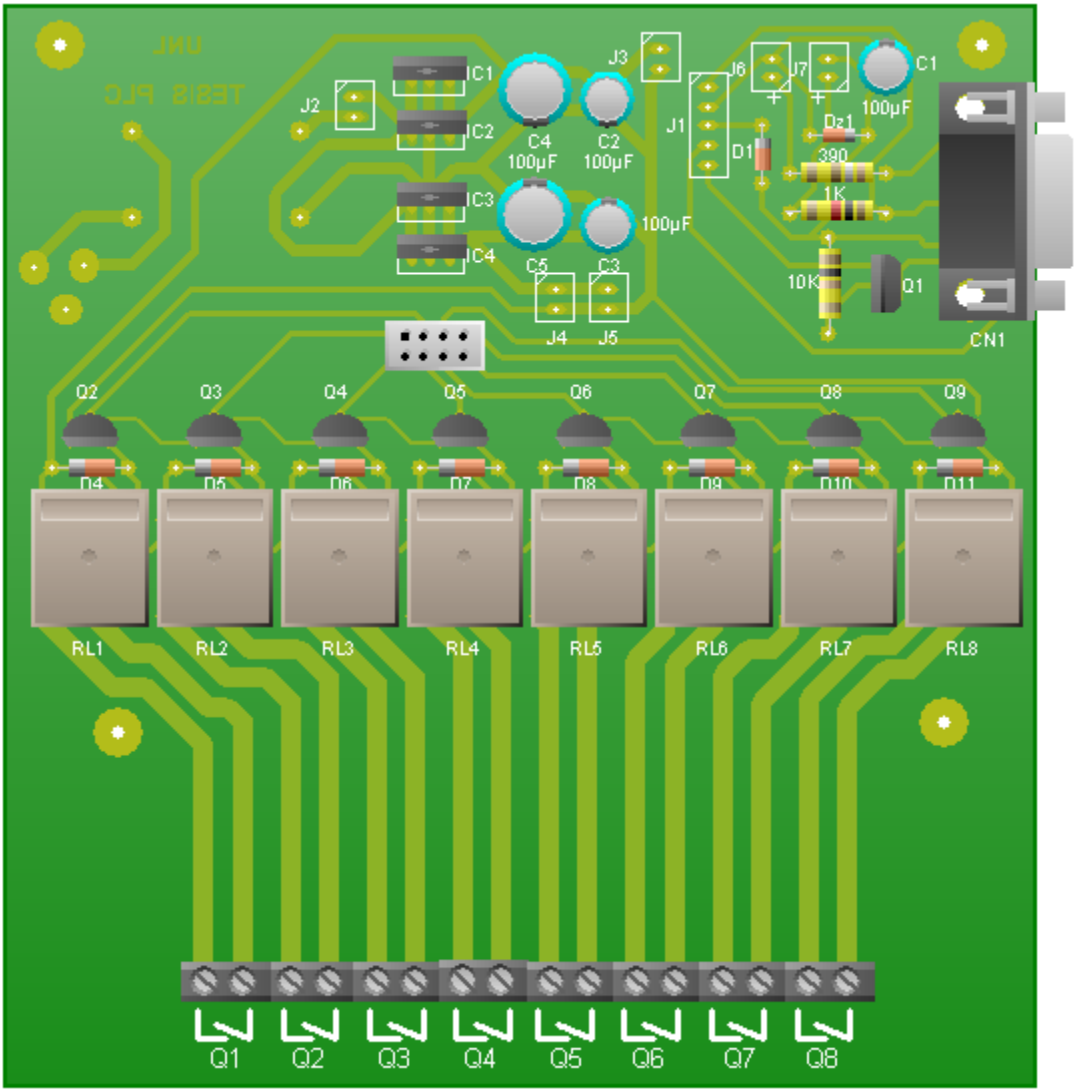

FIGURA 50: Placa Fuente, Programador y Etapa de alto Voltaje del PLC, Vista Simulada.

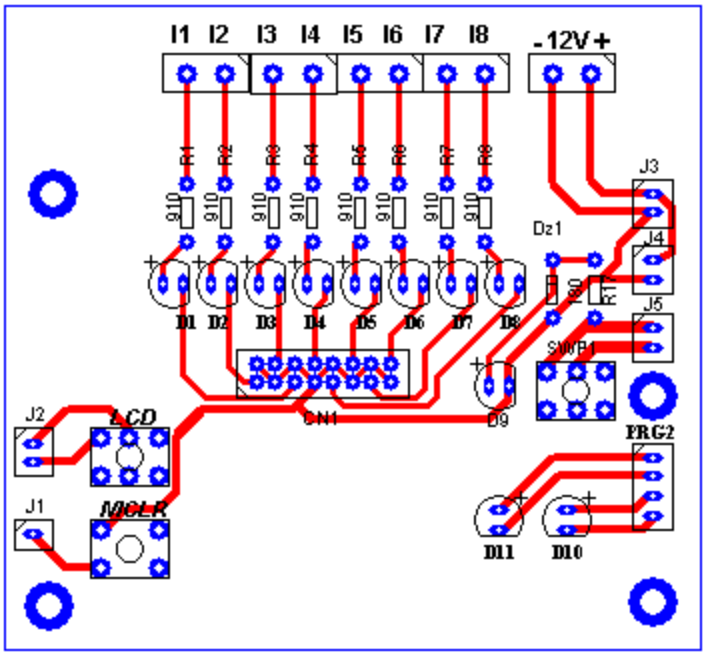

FIGURA 51: Placa de Control del PLC Vista de pistas y componentes.

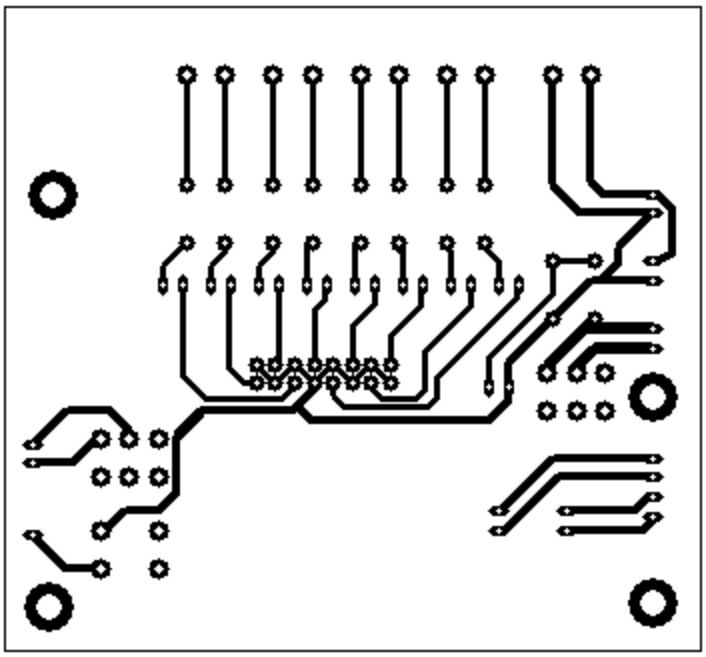

FIGURA 52: Placa de Control del PLC Vista de pistas.

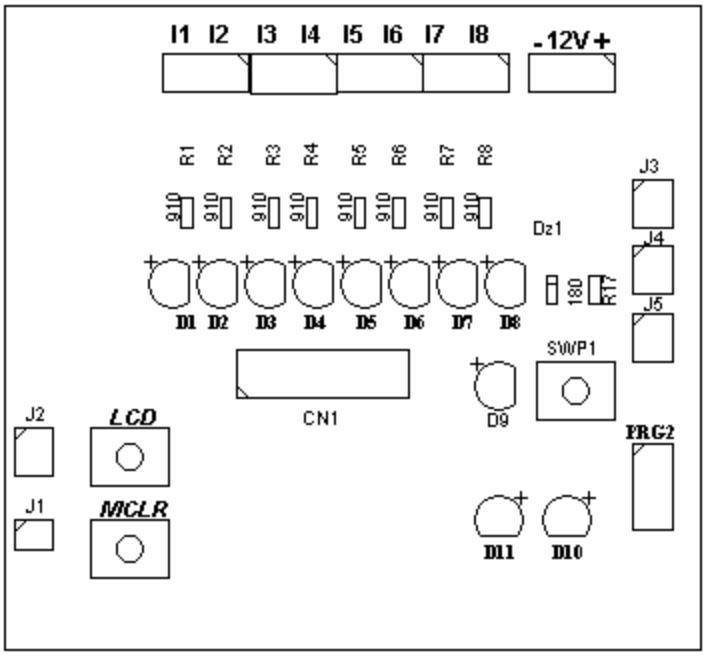

FIGURA 53: Placa de Control del PLC Vista de componentes.

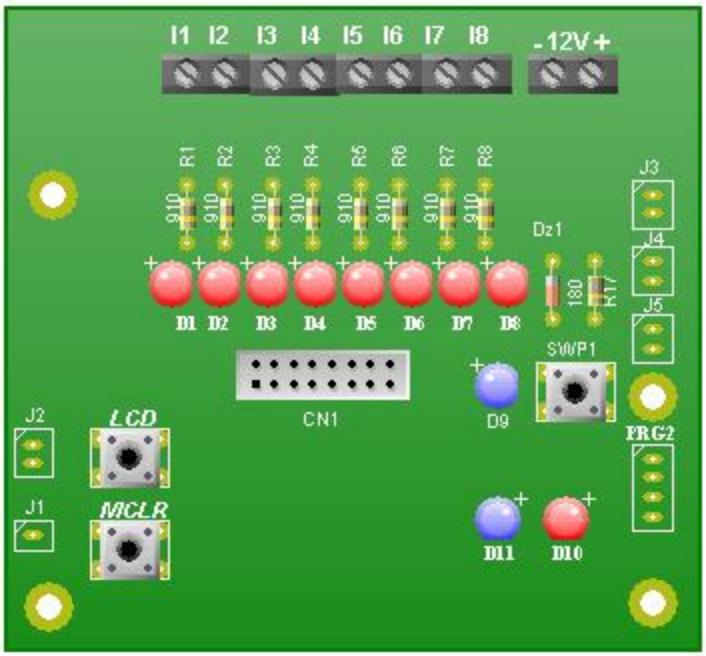

FIGURA 54: Placa de Control del PLC Vista Simulada.

3.1.2 Método utilizado para la construcción del proyecto.

Después de haber hecho el diseño y las pruebas respectivas se procedió a la construcción del circuito impreso con el método del papel fotográfico.

Esto consiste en utilizar una baquelita de fibra de vidrio, imprimir con tinta laser en el papel fotográfico el circuito o pistas a impregnarse en la baquelita, una vez impreso se ubicó la cara impresa del papel en la parte de cobre de esta, luego con una plancha caliente se procedió a dar unas pasadas a la hoja de papel hasta que la tinta quede impregnada en el cobre quedando así lista para ser expuesta al percloruro férrico, este es un ácido que carcomerá todo el cobre que no esté protegido por la tinta impregnada en el mismo, luego de algunos minutos y dependiendo de la efectividad del acido se obtuvo ya la placa con las pistas lista para ser perforada utilizando para ello brocas de 1, 1.2 y 2 milímetros.

Una vez hecha la placa se procedio a ubicar respectivamente los componentes en el sitio adecuado, luego se realizó algunas pruebas con el circuito ya montado para ver si todo está en orden antes de ubicar las placas en la caja donde va ubicado todo el circuito realizado.

3.1.3 Programación del PLC.

Características de IC-Prog.-

Es un programa que corre bajo Windows y que permite la programación de muchos dispositivos y que está probado por un buen número de programadores Hardware.

Los parámetros de la línea de mando pueden ser usados como interfaz con Compiladores externos.

Configuración De ICprog.

1) DESCOMPRIMIR Y CORRER EL .EXE

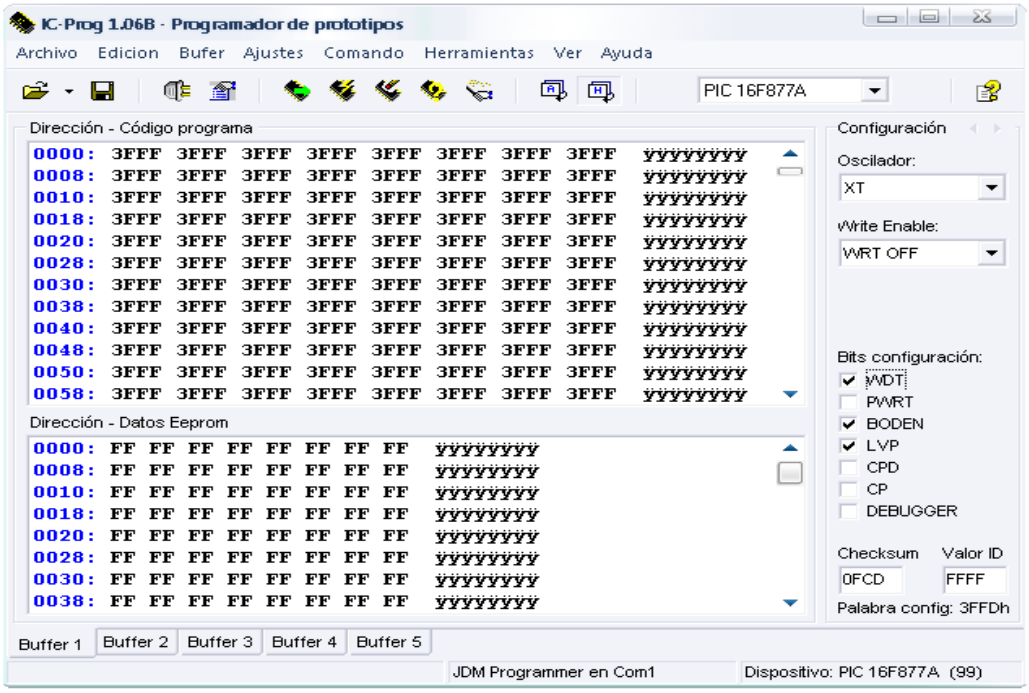

FIGURA 55: Descomprimir y correr el .exe

2) UNA VEZ ABIERTO EL PROGRAMA CONFIGURAR LO SIGUIENTE:

AJUSTES->TIPO DE HARDWARE

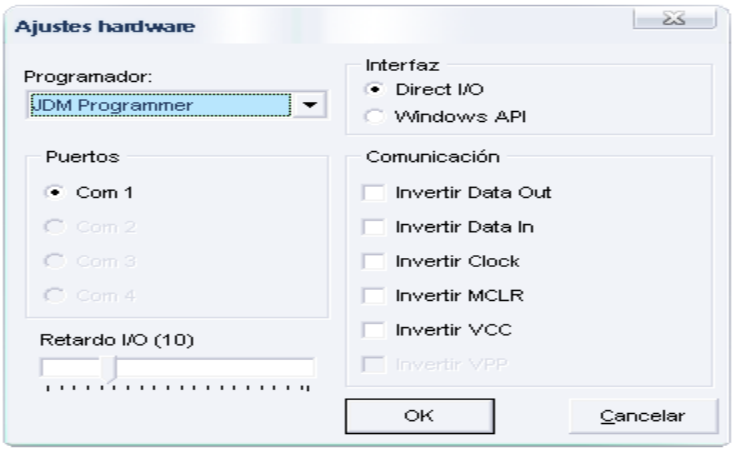

FIGURA 56: Ajustes, tipo de HARDWARE.

AJUSTES->OPCIONES->

IDIOMA.

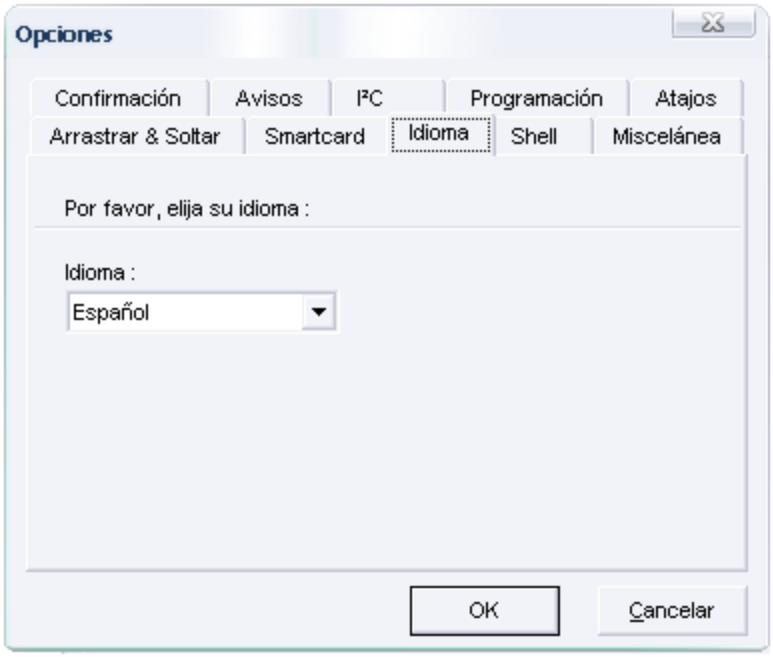

FIGURA 57: Ajustes, Opciones, IDIOMA.

MISCELÁNEA.

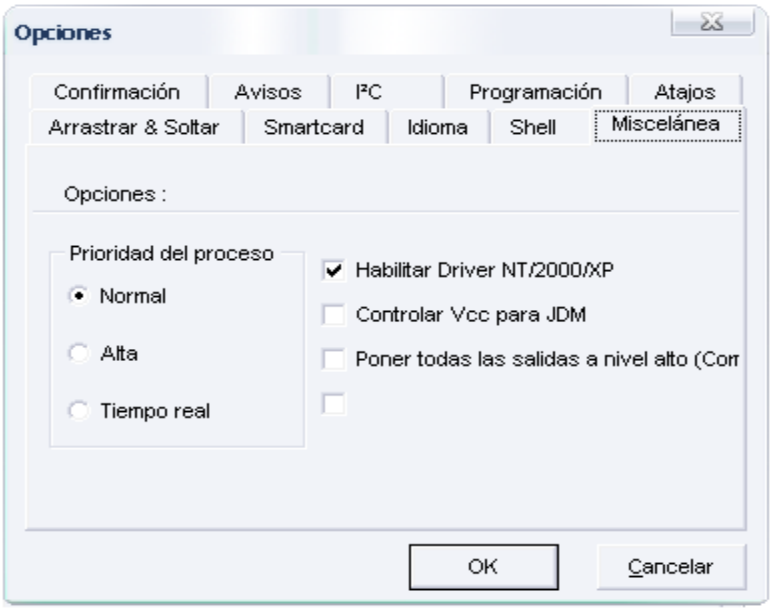

FIGURA 58: Ajustes, Opciones, MISCELÁNEA.

PROGRAMACIÓN.

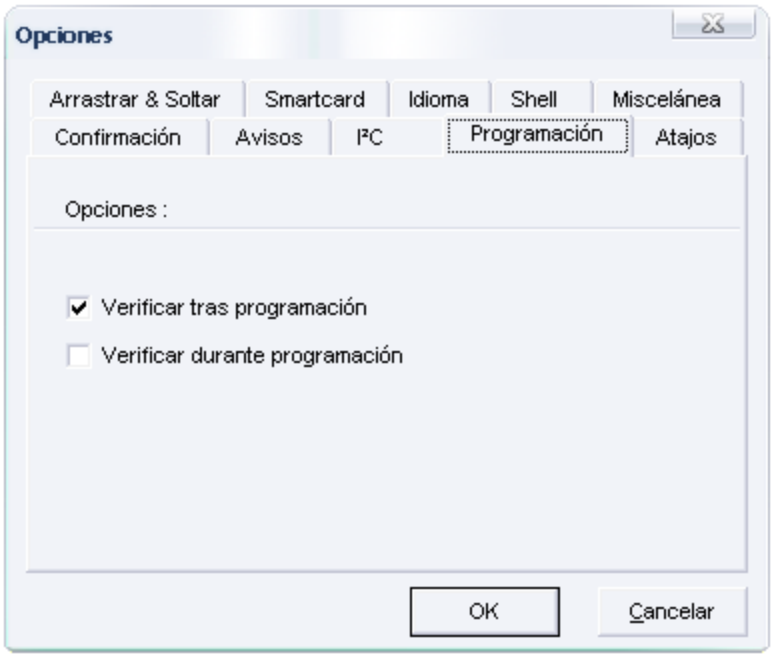

FIGURA 59: Ajustes, Opciones, PROGRAMACIÓN.

Compilador PicBasic Pro.

El software PBP debe ser copiado a su disco rígido antes de usarlo. Se debe crear un subdirectorio en su disco llamado PBP u otro nombre a su elección, tipeando md PBP en el prompt de DOS. Se copia todos los archivos desde el diskette adjunto, en el subdirectorio recientemente creado, tipeando:

xcopy a:*.* PBP /s

La opción /s asegurará que todos los subdirectorios necesarios serán creados dentro del subdirectorio PBP. Si el archivo es comprimido (.ZIP) o ejecutable (.EXE), necesita descomprimirlo (unzip) usando la opción -d para asegurarse que los subdirectorios sean recreados. Asegúrese que FILES y BUFFERS sean fijados en por lo menos 50 en su archivo CONFIG.SYS. Dependiendo de cuantos FILES y BUFFERS ya estén en uso por su sistema, puede ser necesario aumentar su número.

3.1.3.1 BASES DEL PBP.

ETIQUETAS DE LÍNEA (LABELS).- Para marcar líneas que el programa puede desear referenciar con comandos GOTO ó GOSUB, PBP usa etiquetas de línea. PBP no permite número de línea y no requiere que cada línea sea etiquetada. Cualquier línea PBP puede comenzar con una etiqueta de línea que es simplemente un identificador seguido por un punto y coma (;)

here: Serout 0, N2400, [“Hello, World!”, 13, 10]

Goto here

VARIABLES.- Es donde se guardan datos en forma temporaria en un programa PBP. Son creadas usando la palabra clave VAR. Pueden ser bits, bytes ó word. Espacio para cada variable es automáticamente destinado en la memoria del micro controlador por PBP. El formato para crear una variable es el siguiente:

Etiqueta VAR tamaño (.modificadores)

CONSTANTES.- Las llamadas constantes pueden ser creadas de manera similar a las variables. Puede ser más conveniente usar un nombre de constante en lugar de un número constante. Si el número necesita ser cambiado, únicamente puede ser cambiando en un lugar del programa donde se define la constante. No pueden guardarse datos variables dentro de una constante. Etiqueta con expresión constante.

COMENTARIOS. - Un comentario de PBP comienza con la palabra clave REM o el apóstrofe (´ o ;). Todos los demás caracteres de esa línea se ignoran.

DEFINE.- Algunos elementos, como el oscilador y las ubicaciones de los pin LCD, están predefinidos en PBP. DEFINE le permite a un programa PBP cambiar estas definiciones si así lo desea. Define puede ser usado para cambiar el valor predefinido del oscilador, los pines de DEBUG y el baud rate y

las ubicaciones de los pin LCD además de otras cosas. Estas definiciones deben estar en mayúsculas

Operadores de comparación.- Se usan en declaraciones IF... THEN para comparar una expresión con otra .Los operadores soportados son:

Operador	Descripción
= o ==	Igual
<> o !=	No igual
<	Menor
>	Mayor
<=	Menor o igual
>=	Mayor o igual

TABLA 17: Operadores de comparación.

Operadores lógicos.- Los operadores lógicos difieren de las operaciones de bit inteligente. Entregan un resultado CIERTO / FALSO de su operación Valores 0 son tratados como falso cualquier otro valor es cierto se usan junto a operadores de comparación en una declaración IF... THEN .Los operadores soportados son:

Operador	Descripción
AND o &&	AMD logico
OR o I I	OR logico
XOR o ^ ^	OR exclusivo logico
NOT AND	NAND logico
NOT OR	NOR logico
NOT XOR	NXOR logico

TABLA 18: Operadores lógicos.

3.1.3.2 REFERENCIA DE DECLARACIONES PBP.

@	Inserta una linea de codigo ensamblador
ASM...ENDASM	Inserta una seccion de codigo ensamblador
BRANCH	GOTO computado(equiv. a ON..GOTO)
BRANCHL	BRANCH fuera de pagina(BRANCH largo)
BUTTON	Anti-rebote y auto-repeticion de entrada en el pin especificado
CALL	Llamada a subrutina de ensamblador
CLEAR	Hace cero todas las variables
COUNT	Cuenta el numero de pulsos en un pin
DATA	Define el contenido inicial en un chip EEPROM
DEBUG	Señal asincronica de salida en un pin fijo y baud
DISABLE	Deshabilita el procesamiento de ON INTERRUPT
DTMFOUT	Produce tonos en un pin
EEPROM	Define el contenido inicial en un chip EEPROM
ENABLE	Habilita el procesamiento de ON INTERRUPT
END	Detiene la ejecución e ingresa en modo de baja potencia
FOR...NEXT	Ejecuta declaraciones en forma repetitiva
FREQOUT	Produce hasta 2 frecuencias en un pin
GOSUB	Llama a una subrutina BASIC en la etiqueta especificada
GOTO	Continua la ejecución en la etiqueta especificada
HIGH	Hace alto la salida del pin
HSERIN	Entrada serial asincrónica (hardware)
HSEROUT	Salida serial asincrónica (hardware)
I2CREAD	Lee bytes de dispositivo I2C

I2CWRITE	Graba bytes en dispositivo I2C
IF..THEN..ELSE..ENDIF	Ejecuta declaraciones en forma condicional
INPUT	Convierte un pin en entrada
(LET)	Asigna el resultado de una expresión a una variable
LCDOUT	Muestra caracteres en LCD
LOOKDOWN	Busca un valor en una tabla de constantes
LOOKDOWN2	Busca un valor en una tabla de constantes o variables
LOOKUP	Obtiene un valor constante de una tabla
LOOKUP2	Obtiene un valor constante o variable de una tabla
LOW	Hace bajo la salida de un pin
NAP	Apaga el procesador por un corto periodo de tiempo
ON INTERRUPT	Ejecuta una subrutina BASIC en un interrupt
OUTPUT	Convierte un pin en salida
PAUSE	Demora (resolución 1mseg.)
PAUSEUS	Demora (resolución 1 µseg.)
PEEK	Lee un byte del registro
POKE	Graba un byte en el registro
POT	Lee el potenciómetro en el pin especificado
PULSIN	Mide el ancho de pulso en un pin
PULSOUT	Genera pulso hacia un pin
PWM	Salida modulada en ancho de pulso a un pin
RANDOM	Genera numero pseudo-aleatorio
RCTIME	Mide el ancho de pulso en un pin
READ	Lee byte de un chip EEPROM
RESUME	Continua la ejecución después de una interrupción
RETURN	Continua en la declaración que sigue al ultimo GOSUB

REVERSE	Convierte un pin de salida en entrada o uno de entrada en salida
SERIN	Entrada serial asincrónica (tipo BS!)
SERIN2	Entrada serial asincrónica (tipo BS2)
SEROUT	Salida serial asincrónica (tipo BS1)
SEROUT2	Salida serial asincrónica (tipo BS2)
SHIFTIN	Entrada serial sincrónica
SHIFTOUT	Salida serial sincrónica
SLEEP	Apaga el procesador por un periodo de tiempo
SOUND	Genera un tono o ruido blanco en un pin
STOP	Detiene la ejecución del programa
SWAP	Intercambia los valores de dos variables
TOGGLE	Hace salida a un pin y cambia su estado
WHILE..WEND	Ejecuta declaraciones mientras la condición sea cierta
WRITE	Graba bytes a un chip EEPROM
XIN	Entrada X - 10
XOUT	Salida X - 10

TABLA 19: Referencia de declaraciones PBP.

@.- Cuando se usa al comienzo de una línea, provee un atajo para insertar una declaración en lenguaje ensamblador en un programa PBP. Este atajo se puede usar libremente para unir código ensamblador con declaraciones PBP.

ASM..ENDASM.- Estas instrucciones le dicen a PBP que el código entre estas dos líneas esta en lenguaje ensamblador y no debe ser interpretado como declaraciones PBP. Se puede usar estas dos instrucciones libremente para mezclar código ensamblador con declaraciones PBP.

El tamaño máximo para una sección de texto ensamblador es 8 K. Este es el tamaño máximo para la fuente actual, incluyendo comentarios, no el

código generado .Si el bloque de texto es mayor, divídalo en múltiples secciones ASM ENDASM o inclúyalo en un archivo separado.

ASM resetea a 0 el registro de página. Debe asegurarse que el registro de página sea 0 antes de ENDASM si el código de ensamblador lo ha alterado.

BRANCH índex, [etiqueta { ,etiqueta .....}].- Causa que el programa salte a una posición diferente, basada en una variable indexada. Es similar al ON...GOTO de otros BASIC.

Índex selecciona una etiqueta de una lista. La ejecución comienza en la etiqueta especificada .Por ejemplo, si Índex es 0, el programa salta a la primer etiqueta especificada en la lista, si Índex es 1, salta a la segunda y así sucesivamente. Si Índex es mayor ó igual al número de etiquetas, no se toma ninguna acción y la ejecución continúa con la declaración siguiente al BRANCH. Se pueden usar hasta 256 etiquetas en un BRANCH.

BRANCHL Índex, [etiqueta {, etiqueta....}].- Trabaja en forma similar a BRANCH, haciendo que el programa salte a una localización determinada, basándose en una variable indexada. Las principales diferencias son que puede saltar a una etiqueta ubicada en otra página de código y que genera un código dos veces mayor en tamaño al de BRANCH

BRANCHL. Se pueden usar hasta 128 etiquetas en un BRANCHL.

BUTTON, Pin , Down , Delay , Rate , Bvar , Action , Etiqueta.

Lee Pin y opcionalmente ejecuta anti-rebote y auto-repetición. Pin automáticamente se toma como entrada.

Down	Estado del pin cuando se oprime el pulsador (0 ..1)
Delay	Contador de ciclos antes de que comience la auto-repetición (0...255). Si es 0, no se efectúa anti-rebote ni auto repetición .Si es 255 se eliminan rebotes, pero no auto-repetición.
Rate	Valor de auto-repetición (0..255)
Bvar	Variable con tamaño de byte usada internamente para conteo de demoras y repeticiones, Debe ser inicializada a 0 antes de ser usada y no ser usada en cualquier lugar del programa.
Action	Estado del pulsador a ser actuado.
Etiqueta	La ejecución comienza en esta etiqueta si es cierto Acción.

TABLA 20: Comandos de antirrebote y autorepeticion

CALL.- Ejecuta la subrutina ensamblador llamada etiqueta.

Normalmente se usa GOSUB para ejecutar una subrutina PBP. La principal diferencia entre GOSUB y CALL, es que con ésta última no se chequea la existencia de etiquetas hasta el momento de ensamblar.

CLEAR.- Coloca en cero todos los registros en cada banco. Coloca en cero todas las variables, incluyendo las del sistema. Por lo general, las variables deben ser colocadas en un estado inicial apropiado por el programa, y no usando CLEAR.

COUNT COUNT Pin,Period,Var.- Cuenta el numero de pulsos en un Pin, durante un período Periodo, y guarda el resultado en Var .Pin es automáticamente colocado como entrada.

DATA { @ location , } constante { ,constante }.- Guarda constantes en un chip EEPROM cuando este dispositivo se programa por primera vez. Si se omite el valor opcional location, la primer declaración de DATA comienza a almacenarse en la dirección 0 y las declaraciones siguientes, en las direcciones siguientes. Si existe un valor location este indica la dirección de comienzo donde se almacenará la información.

Una etiqueta opcional se le puede asignar a la dirección de comienzo, para futuras referencias del programa,

DEBUG item{ ,item ..}.- Envía uno ó más items a un pin predefinido con un baud rate predefinido en formato estándar asincrónico, usando 8 bits de datos, sin paridad y con 1 bit de parada (stop bit) (8N1) .El pin, automáticamente se convierte en salida. Si un signo (#) precede a un item, se envía serial mente la representación ASCII para cada dígito. Puede ser usada para enviar información de depuración (variables, posición de marcadores, etc.

DISABLE.- Interrumpe el procesamiento siguiente a la instrucción. Pueden ocurrir otras Interrupciones, pero el manipulador de interrupciones del BASIC en el PBP, no se ejecutará hasta que se encuentre un ENABLE.

DTMFOUT Pin, { Onms ,Offms,} [Tone {,Tone}].- Produce una secuencia DTMF Touch Tone en Pin, Pin automáticamente se convierte en salida .

On ms es el número de milisegundos que suena cada tono y Off ms es el número de milisegundos de pausa entre cada tono. Si no están especificados, por defecto On ms es 200 ms y Off ms es 50 ms.

DTMFOUT trabaja mejor con un oscilador de 20 MHz También puede trabajar con uno de 10 MHz y aún con uno de 4 MHz, aunque será muy difícil de filtrar y tendrá muy baja amplitud.

EEPROM {Location,} [constante {, constante...}].- Guarda constantes en un chip EEPROM. Si se omite el valor opcional localización, la primera declaración se guarda en la dirección 0 del EEPROM y las subsiguientes en las siguientes direcciones del mismo. Si se indica un valor localización, éste indica la dirección de comienzo para guardar los datos.

Constante puede ser una constante numérica ó una sarta de constantes. Solo se guardan los bytes menos significativos de los valores numéricos. Las sartas

son guardadas como bytes consecutivos de valores ASCII. No se agregan automáticamente terminadores, ni se completa el largo.

ENABLE.- Interrumpe el procesamiento que fue previamente deshabilitado con DISABLE, siguiendo a esta instrucción.

END.- Detiene la ejecución del proceso y entra en modo de baja potencia.

FREQOUT.- Produce la ó las frecuencias especificadas en el Pin, este se convierte automáticamente en salida. Puede producir una ó dos frecuencias de 0 a 32767 Hz al mismo tiempo.

FREQOUT genera tonos usando una forma de modulación de ancho de pulso. Usualmente se necesita algún tipo de filtro para suavizar la señal hasta una forma de onda senoidal quitándole algunas armónicas generadas:

FREQOUT trabaja mejor con un oscilador de 20 Mhz. También puede trabajar con uno de 10 MHz y aún con uno de 4 MHz, aunque será muy difícil de filtrar y tendrá muy baja amplitud. Cualquier otra frecuencia causará que FREQOUT genere una frecuencia proporcional al oscilador comparado a 20 Mhz .

GOSUB.- Salta a la subrutina indicada en la etiqueta, guardando su dirección de regreso en la pila (stack). A diferencia del GOTO, cuando se llega a un RETURN, la ejecución sigue con la declaración siguiente al último GOSUB ejecutado. Se puede usar un número ilimitado de subrutinas en un programa y pueden estar anidadas.

GOTO.- La ejecución del programa continúa en la declaración de la etiqueta.

GOTO send ´ salta a la declaración etiquetada send

send: serout 0,N2400, [” Hi”] ´ envía “ Hi” como salida al Pin0 en forma serial

HIGH.- Hace de valor alto el Pin especificado y lo convierte automáticamente en salida.

HIGH PORTA.0 ´ convierte PORTA, Pin0 en salida y lo coloca en valor

Alto (5 volt).

HSERIN {ParityLabel , }{Timeout ,Label ,} [Item { . . . }].- Recibe uno ó más Items de un port serial (de hardware) en dispositivos que soportan comunicaciones seriales asincrónicas por hardware .

HSERIN es una de varias funciones seriales asincrónicas pre-construidas. Sólo puede ser usada en dispositivos que posean hardware USART. Vea la hoja de datos del dispositivo para información de los pin seriales de entrada y otros. Los parámetros seriales y el baud-rate son especificados usando DEFINE:

DEFINE HSER_RCSTA 90h´ coloque el registro receptor en receptor habilitado

DEFINE HSER_TSTA 20h ´ coloque el registro de transmisión en transmisión habilitada

DEFINE HSER_BAUD 2400 ´ coloque baud rate

HSERIN asume un oscilador de 4 MHz cuando calcula el baud rate. Para mantener una relación de baud rate apropiada con otros valores de oscilador, use DEFINE para especificar el nuevo valor OSC.

El formato por defecto de los datos seriales es 8N1, 8 bits de datos, sin paridad y 1 stop bit.

Dado que la recepción serial se realiza por hardware, no es posible invertir los niveles para eliminar un driver RS - 232. Por esto debe usarse un driver adecuado con HSERIN.

HSEROUT [Item {,Item }].- Envía uno ó más Items al port serial de hardware en dispositivos que soportan comunicaciones seriales asincrónicas por hardware .

HSEROUT es una de varias funciones seriales asincrónicas pre-construidas. Sólo puede ser usada en dispositivos que posean hardware USART. Vea la hoja de datos del dispositivo para información de los pin seriales de entrada y otros. Los parámetros seriales y el baud-rate son especificados usando DEFINE:

DEFINE HSER_RCSTA 90h ´ coloque el registro receptor en receptor habilitado

DEFINE HSER_TSTA 20h ´ coloque el registro de transmisión en transmisión habilitada

DEFINE HSER_BAUD 2400 ´ coloque baud rate

HSEROUT asume un oscilador de 4 MHz cuando calcula el baud rate. Para mantener una relación de baud rate apropiada con otros valores de oscilador, use DEFINE para especificar el nuevo valor OSC.

El formato por defecto de los datos seriales es 8N1, 8 bits de datos, sin paridad y 1 stop bit.

Dado que la recepción serial se realiza por hardware, no es posible invertir los niveles para eliminar un driver RS - 232. Por esto debe usarse un driver adecuado con HSEROUT.

I2CREAD DataPin ,ClockPin,Control,{Address,} [Var {,Var ...}] { . Label }.- Envía los bytes de Control y opcionalmente los de Address, a través del ClockPin y el DataPin y guarda los bytes recibidos dentro de Var.

I2CREAD y I2CWRITE pueden ser usados para leer y grabar datos de un EEPROM serial usando una interface I2C de 2 cables

I2CWRITE DataPin ,ClockPin,Control,{Address,}[Value {,Value ...}] { . Label }.- Envía los bytes de Control y opcionalmente los de Address, a través del ClockPin y el DataPin seguidos por Value. ClockPin y DataPin. El tamaño de dirección enviado (byte ó Word) es determinado por el tamaño de la variable usada. Si se usa una variable con tamaño byte se envía una dirección de 8 bits. Si se envía una variable de tamaño word, se envía una dirección de 16 bits. Cuando se escribe un EEPROM serial, es necesario esperar 10 ms (dependiendo del dispositivo) para completar la grabación, antes de intentar comunicarse nuevamente con el dispositivo.

IF...THEN

IF Comp { AND/OR Comp ... } THEN Label

IF Comp { AND/OR Comp ... } THEN

Declaración

ELSE

Declaración

ENDIF

Efectúa una ó más comparaciones. Cada término Comparado puede relacionar una variable con una constante ú otra variable e incluye uno de los operadores listados anteriormente.

IF... THEN evalúa la comparación en términos de CIERTO o FALSO. Si lo considera cierto, se ejecuta la operación posterior al THEN. Si lo considera falso, no se ejecuta la operación posterior al THEN. Las comparaciones que dan 0 se consideran falso, cualquier otro valor es cierto. Todas las comparaciones son sin signo, ya que PBP solo soporta operaciones sin signo.

Se utilizó paréntesis para especificar el orden en que se deben realizar las operaciones. De otra manera, la prioridad de los operadores lo determina y el resultado puede no ser el esperado.

INPUT.- Convierte el Pin especificado en una entrada.

```
INPUT PORTA.0 ‘convierte el PORTA, pin 0 en entrada
```

En forma alternativa, el pin puede ser colocado como entrada de una forma más rápida y corta (desde un código generado standpoint):

```
TRISB.0 =1 ‘Setea el PORTB, pin 0 como entrada
```

Todos los pines en un port pueden ser colocados como entradas seteando el registro TRIS completo de una sola vez:

```
TRISB = %11111111 ‘Setea todo el PORTB como entrada
```

{LET}.- Asigna un Value a una Variable. El Value puede ser una constante, otra variable o el resultado de una expresión. Refiérase a la sección previa acerca de operadores para más información. La palabra clave LET, por sí misma, es opcional.

```
LET B0 = B1 * B2 + B3

B0 = Sqr W1
```

LCDOUT Item{ , Item...}.- Muestra Items en un visor de cristal líquido inteligente (LCD). PBP soporta módulos LCD con un controlador Hitachi 44780 o equivalente. Estos LCD, usualmente, tienen un cabezal de 14 o 16 pines simples o duales en un extremo.

Los comandos son enviados al LCD, enviando un $FE seguido por el comando. Algunos comandos útiles se muestran en la siguiente tabla:

Comando	Operación
$FE, 1	Limpia visor
$FE, 2	Vuelve a inicio (comienzo de la primera línea)
$FE, $0C	Cursor apagado
$FE, $0E	Subrayado del cursor activo
$FE, $0F	Parpadeo del cursor activo
$FE, $10	Mueve cursor una posición hacia la izquierda
$FE, $14	Mueve cursor una posición hacia la derecha
$FE, $C0	Mueve cursor al comienzo de la segunda línea

TABLA 21: Algunos comandos útiles para controlar un LCD.

Existe un comando para mover el cursor al comienzo de la segunda línea en un visor de dos líneas. Para muchos visores 16x2, la primera línea comienza en $0 y la segunda, en $40. El comando:

LCDOUT $FE, 1, “Hello” ‘limpia el visor y muestra “Hello”

El LCD puede estar conectado al micro Pic, usando un bus de 4 bit o uno de 8 bit. Si se usa un bus de 8 bit, todos los 8 bits deben estar en un port. Si se usa un bus de 4 bit, debe estar conectado o a los 4 bit inferiores o a los 4 bit superiores de un port. Enable y Register Select deben estar conectados a algún pin del port. R/W debe estar colocado a tierra, ya que el comando de LCDOUT solamente es de grabación.

PBP supone que el LCD está conectado a pines específicos, a menos que se le diga de otra manera. Asume que el LCD va a ser usado con un bus de 4 bits, con las líneas de data DB4 - DB7 conectadas en el micro Pic a PORTA.0 - PORTA.3, Register Select a PORTA.4 y Enable a PORTB.3. Además, inicializa el LCD como un visor de dos líneas.

Para cambiar este seteo, coloque uno o más de los siguientes DEFINEs, todos en mayúsculas, en el comienzo de su programa PBP:

```
DEFINE LCD_DREG PORTC      ; define pines del LCD C0 a C3

DEFINE LCD_DBIT 0          ; empezando desde el puerto C0 a C3

DEFINE LCD_RSREG PORTC     ; define pin para conectar el bit RS

DEFINE LCD_RSBIT 5         ; en el puerto C5

DEFINE LCD_EREG PORTC      ; define pin para conectar el bit Enable

DEFINE LCD_EBIT 4          ; en el puerto B4
```

LOOKDOWN Search, [Constant{ , Constant...}] , Var.- La declaración LOOKDOWN busca en una lista de 8 bit los valores Constante que coincidan con un valor. Si se encuentra, el índice de la constante es guardado en Var así, si el valor es el primero de la lista, Var = 0. Si es el segundo, Var = 1 y así, sucesivamente. Si no se encuentra, no se toma ninguna acción y Var permanece sin cambios.

LOOKDOWN2 Search, {Test} [Value{, Value...}] , Var.- La declaración LOOKDOWN2 busca un valor en una tabla. Si lo encuentra, el índice de la constante es guardado en Var así, si el valor es el primero de la lista, Var = 0. Si es el segundo, Var = 1 y así, sucesivamente. Si no se encuentra, no se toma ninguna acción y Var permanece sin cambios.

LOOKUP Index, [Constant{ , Constant...}] , Var.- LOOKUP puede ser usado para obtener valores de una tabla de constantes de 8 bits, Si Index es cero, Var toma el valor de la primer Constante. Si Índex es 1, Var toma el valor de la segunda Constante y así sucesivamente. Si Índex es mayor ó igual que el número de entradas en la lista de constantes, no se toma ninguna acción y Var permanece sin cambios.

LOOKUP2 Index,[Value [,Value ...}] , Var.- Puede ser usado para obtener entradas de una tabla de Valores, Si Índex es cero, Var toma el valor del primer Value. Si Índex es 1, Var toma el valor del segundo Value y así sucesivamente. Si Índex es mayor ó igual que el número de entradas en la lista, no se toma ninguna acción y Var permanece sin cambios. La lista de Values puede ser una mezcla de constantes numéricas y sartas en 16 bits y variables.

LOW.- Coloca el pin especificado en valor bajo y automáticamente lo convierte en salida.

NAP.- Coloca al micro controlador en modo de baja potencia por períodos de tiempo reducidos. Durante este NAP, se reduce al mínimo el consumo de energía. Los períodos indicados son solo aproximados, porque el tiempo se deriva del Watchdog Timer que está controlado por R/C y puede variar de chip a chip y también con la temperatura. Como NAP usa el Watchdog Timer es independiente de la frecuencia del oscilador.

Period	Demora (aprox.) en milisegundos
0	18
1	36
2	72
3	144
4	288
5	576
6	1152
7	2304

TABLA 22: Periodo y demora aproximada en milesegundos.

NAP 7 ´ pausa en baja potencia por aprox. 2,3 segundos

ON INTERRUPT.- Permite el manejo de las interrupciones del micro controlador por medio de una subrutina PBP. Existen dos formas de manejar interrupciones usando PBP. La primera es escribir una subrutina de interrupción en Lenguaje ensamblador. Esta es la forma de manejar interrupciones con la menor latencia. El segundo método es escribir un handler (manejador) de interrupciones PBP. Es similar a una subrutina PBP, pero termina con un RESUME. Cuando ocurre una interrupción, se marca con una bandera .Cuando la ejecución de la declaración PBP que se estaba ejecutando termina, el programa salta al handler de interrupciones indicado en Label. Una vez que termina el trabajo del handler, una declaración RESUME envía el programa de vuelta a donde estaba cuando ocurrió la interrupción, tomando todo como lo dejó.

OUTPUT.-Convierte el pin especificado en salida.

OUTPUT PORTA.0 ´ convierte PORTA pin 0 en salida

En forma alternativa, el pin puede ser convertido en salida de una manera más rápida y corta (con un código generado standpoint).

TRISB.0 = 0 ´ setea PORTB pin 0 como salida

Todos los pines de un port pueden ser seteados simultáneamente como salida usando el registro TRIS completo :

TRISB = %00000000 ´ setea todos los pines de PORTB como salidas

PAUSE.- Detiene el programa por Periodo milisegundos .Periodo tiene 16 bit, por lo que los retardos pueden ser de hasta 65.535 milisegundos. (Un poco más de 1 minuto). Tiene la misma precisión que el clock. PAUSE asume la frecuencia de 4 MHz del oscilador. Si se usa un oscilador de otra frecuencia, se debe indicar usando el comando DEFINE OSC.

PAUSEUS.- Detiene el programa por Periodo milisegundos .Periodo tiene 16 bit, por lo que los retardos pueden ser de hasta 65.535 milisegundos.

OSC	Demora mínima
3(3.58)	20us
4	24us
8	12us
10	8us
12	7us
16	5us
20	3us

TABLA 23: Oscilador y su demora minima.

PAUSEUS Asume la frecuencia de 4 MHz del oscilador. Si se usa un oscilador de otra frecuencia, se debe indicar usando el comando DEFINE OSC. Vea la sección sobre velocidad para mayores detalles.

PAUSEUS 1000 demora de 1 segundo

PEEK Address, Var.- Lee el registro del micro controlador en la dirección Address especificada y guarda la lectura en Var. Opciones especiales del microPIC, como convertidores A/D y puertos adicionales pueden ser leídos usando PEEK.

POKE Address, Value.- Graba Value en el registro del micro controlador en la dirección Address especificada. Opciones especiales del microPIC, como convertidores A/D y ports adicionales pueden ser leídos usando PEEK.

POT Pin, Scale, Var.- Lee un potenciómetro (ú otro dispositivo resistivo) en Pin. La resistencia se mide tomando el tiempo de descarga de un capacitor a través de un resistor. (5 K a 50 K). Scale se usa para ajustar distintas constantes RC. Para constantes RC grandes, Scale debe ser baja (valor mínimo 1). Para constantes RC pequeñas, Scale debe ser máxima (255).

PULSIN Pin,State,Var.- Mide el ancho del pulso en Pin. Si State es cero se mide el ancho de un pulso bajo. Si State es uno, se mide el ancho de un pulso alto. El ancho medido se coloca en Var. Si el flanco del pulso no llega, ó el ancho del pulso es demasiado grande para ser medido, Var=0. Si se usa una variable de 8 bit, solo se usan los bits menos significativos de la medición de 16 bits. La resolución de PULSIN depende de la frecuencia del oscilador. Si se usa un oscilador de 4 MHz, el ancho de pulso se obtiene en incrementos de 10 µs. Si se usa un oscilador de 20 MHz, el ancho de pulso tendrá una resolución de 2 µs. Definir un valor de OSC no tiene efectos sobre PULSIN. La resolución siempre cambia con la velocidad del oscilador en uso.

PULSEOUT Pin, Period.- Genera un pulso en Pin, con un Periodo especificado. El pulso se genera activando dos veces el pin, por lo que la polaridad del pulso depende del estado inicial del pin. La resolución de PULSOUT depende de la frecuencia del oscilador. Si se usa un oscilador de 4 MHz, el Periodo del pulso generado estará en incrementos de 10 µs .Si se usa un oscilador de 20 MHz, Periodo una resolución de 2 µs. Definir un valor de OSC no tiene efectos sobre PULSOUT.

RANDOM .- Efectúa una interacción pseudo-aleatoria en Var, esta debe ser una variable de 16 bit. No se pueden usar variables de array con índice variable, pero se permite usar variables de array con índice constante. Var se usa tanto como origen, como para guardar el resultado.

RCTIME Pin, State, Var.- Mide el tiempo que un Pin permanece en un estado determinado Básicamente es la mitad de un PULSIN. RCTIME puede usarse para leer un potenciómetro (ó cualquier dispositivo resistivo).

READ.- Lee el EEPROM incorporado en la dirección Address, y guarda el resultado en Var. Esta instrucción solo puede ser usada con un microPIC que tenga un EEPROM incorporado como el PIC16F877.

RESUME { Label }.- Vuelve al lugar del programa que se abandonó, después que termina de procesarse una interrupción. RESUME es similar a RETURN, pero es usado al final de un handler de interrupciones PBP.

RETURN.- Vuelve desde una subrutina. Retoma la ejecución en la declaración que sigue al GOSUB que llamó la subrutina. Gosub sub1 ´ va a la subrutina denominada sub1

sub1: Serout 0, N2400,[“Lunch”] ´ envía “Lunch” al pin 0 en forma serial

RETURN ´ vuelve al programa principal después del gosub.

REVERSE.- Si Pin es entrada, lo convierte en salida. Si es salida, lo convierte en entrada. Output 4 ´ convierte pin 4 en salida

SERIN: SERIN Pin,Mode, {Timeout,Label,}{[Qual...],} {Item...}.- Recibe uno ó más Items en Pin, en formato standard asincrónico, usando 8 bit de datos, sin paridad y un stop bit (8N1). SERIN es similar al comando Serin de BS1 con el agregado de Timeout. Pin automáticamente se convierte en entrada Pin puede ser una constante, 0 - 15, ó una variable que contenga un número de 0-15 (p.ej. B0) ó un número de Pin (p.ej. PORTA.0)

Los nombres Mode (p.ej. T2400) están definidos en el archivo MODEDEFS.BAS. Para los, agregue la línea: Include "modedefs.bas".

Al comienzo del programa PBP. BS1DEFS, BAS y BS2DEFS.BAS ya incluyen MODEDEFS.BAS, No lo incluya si ya está usando alguno de estos archivos. Los números Mode pueden ser usados sin incluir este archivo.

Mode	Mode N°	Baud rate	State
T2400	0	2400	CIERTO
T1200	1	1200	
T9600	2	9600	
T300	3	300	
N2400	4	2400	FALSO
N1200	5	1200	
N9600	6	9600	
N300	7	300	

TABLA 24: Rangos de velocidad en datos verdadero u invertido para la comunicación serial

SERIN2 DataPin {FlowPin}, Mode, {ParityLabel,} {Timeout,Label,} [Item...].- Recibe uno ó más Items en el Pin especificado en formato standard asincrónico. SERIN2 es similar al comando Serin de BS2. DataPin es colocado como entrada en forma automática. FlowPin es opcional y

es automáticamente colocado como salida. DataPin y FlowPin pueden ser una constante, 0 - 15, ó una variable que contenga un número de 0-15 (p.ej. B0) ó un número de Pin (p.ej. PORTA.0)

Algunos baud rate estándar se muestran en la tabla siguiente:

Baud rate	Bits 0 - 12
300	3313
600	1646
1200	813
2400	396
4800	188
9600	84
19200	32

TABLA 25: Algunos baud rate estándar.

Bit 13 selecciona paridad par (bit13=1) ó sin paridad (bit13=0). Normalmente, las transmisiones seriales son 8N1 (8 bit de datos, sin paridad, 1 stop bit) .Si se selecciona paridad, los datos son recibidos como 7E1 (7 bit de datos, paridad par, 1 stop bit).

Bit 14 selecciona el nivel de los pines de datos y de control de flujo. Si bit 14=0, se reciben los datos en forma normal, para usar con los drivers RS-232. Si bit 14=1, los datos se reciben invertidos. Esto se puede usar para evitar usar drivers RS-232

SEROUT Pin,Mode,[Item[,Item...}].- Envía uno ó más Items a Pin, en formato standard asincrónico usando 8 bits de datos, sin paridad y 1 stop bit (8N1). SEROUT es similar al comando Serout de BS1 Pin es automáticamente colocado como salida. Los nombres Mode (p.ej. T2400) están definidos en el archivo:

Include "modedefs.bas"

Mode	Mode N°	Baud rate	Estado
T2400	0	2400	LLEVADO A CIERTO
T1200	1	1200	
T9600	2	9600	
T300	3	300	
N2400	4	2400	LLEVADO A INVERTIDO
N1200	5	1200	
N9600	6	9600	
N300	7	300	
OT2400	8	2400	ABIERTO CIERTO
OT1200	9	1200	
OT9600	10	9600	
OT300	11	300	
ON2400	12	2400	ABIERTO INVERTIDO
ON1200	13	1200	
ON9600	14	9600	
ON300	15	300	

TABLA 26 Rangos de velocidad en datos verdadero u invertido para la comunicación serial

SEROUT soporta 3 tipos distintos de datos, que pueden ser combinados libremente dentro de una declaración SEROUT.

SEROUT asume un valor de oscilador de 4 MHz cuando genera sus tiempos de bit. Para mantener los valores de baud rate adecuados con otro oscilador, asegúrese de usar DEFINE OSC con el nuevo valor de oscilador.

SEROUT 0,N2400,[#B0,10] ´ envía el valor ASCII de B0 ,seguido por un LF al pin 0 , en forma serial

SEROUT2 DataPin {Flipan}, Mode,{Pace,} {Timeout,Label,} [Item...].- Envía uno ó más Items al Pin especificado en formato estándar asincrónico. SEROUT2 es similar al comando Serout de BS2. DataPin es colocado como salida en forma automática. FlowPin es opcional y es automáticamente colocado como entrada. DataPin y FlowPin pueden ser una constante, 0 - 15, ó una variable que contenga un número de 0-15 (p.ej. B0) ó un número de Pin (p.ej. PORTA.0)

SHIFTIN DataPin,ClockPin,Mode, [Var{}...].- El ClockPin, desplaza en forma sincrónica los bits en DataPin y guarda los bytes recibidos en Var (en forma opcional) especifica el número de bits a ser desplazado. Si no se especifica se desplazan 8 bits, independientemente del tipo de variable.

Los nombres Mode (p.ej. MSBPRE) están definidos en el archivo:

Include “modedefs.bas”

SHIFTOUT DataPin,ClockPin,Mode, [Var{}...].- Desplaza en forma sincrónica el contenido de Var sobre DataPin y ClockPin. (En forma opcional) específica el número de bits a ser desplazado. Si no se especifica, se desplazan 8 bits, independientemente del tipo de variable.

Los nombres Mode (p.ej. LSBFIRST) están definidos en el archivo:

Include “modedefs.bas”

SLEEP Periodo.- Coloca al micro controlador en modo de baja potencia por Periodo segundos. Periodo tiene 16 bit, por lo que los retardos pueden ser de hasta 65535 segundos (aprox. 18 horas).

SLEEP usa el WatchDog Timer, por lo que es independiente de la frecuencia del oscilador utilizado.

SLEEP 60 ´ duerme por aprox. 1 minuto

SOUND Pin,[Note,Duration{,Note,Duration...}].- Genera un tono y/o ruido blanco en el Pin especificado. Pin es automáticamente colocado como salida. Pin puede ser una constante, 0 - 15, ó una variable que contenga un número de 0-15 (p.ej. B0) ó un número de Pin (p.ej. PORTA.0)

Note 0 es silencio Nte 1-127 son tonos Notes 128-255 son ruido blanco. Los tonos y el ruido blanco están en una escala ascendente (p.ej. 1 y 128 son las frecuencias menores, 129 y 266 las mayores). Note 1 es aprox. 78,74 Hz y Note 127 es aprox. 10000 Hz.

Duración es 0-255 y determina el largo de la nota, en incrementos de 12 milisegundos.

STOP.- Detiene la ejecución del programa, ejecutando un loop sin fin, No coloca al micro controlador en modo de baja potencia. El micro controlador trabaja igual que siempre.

STOP ´ envía el programa a vía muerta

SWAP Variable, Variable.- Intercambia los valores de dos variables. Normalmente intercambiar los valores de dos variables es un proceso tedioso. SWAP lo hace con una sola declaración, sin variables intermedias. Puede ser usado con variables de bit, byte y Word.

TOGGLE.- Invierte el estado del Pin especificado. Pin es automáticamente colocado como salida. Pin puede ser una constante, 0 - 15, ó una variable que contenga un número de 0-15 (p.ej. B0) ó un número de Pin (p.ej. PORTA.0)

Low 0 ´ comienza Pin0 como bajo

TOGGLE 0 ´ cambia a alto el estado del pin 0

5.62. WHILE...WEND

WHILE Condition

Statement

WEND.- Ejecuta las declaraciones Statement en forma repetida, mientras la condición sea cierta. Cuando la condición deja de ser cierta, la ejecución continúa con la declaración siguiente al WEND. Condición puede ser cualquier expresión de comparación.

WRITE.- Graba valores Value en el EEPROM incorporado en la dirección Address especificada. Esta instrucción solo puede ser usada con un microPIC que tenga un EEPROM incorporado como el PIC16F877.

Es usado para colocar datos en el EEPROM durante el momento de la ejecución. Para grabar datos en el EEPROM durante la programación, se usan las declaraciones DATA y EEPROM.

Cada WRITE se auto regula en tiempo y toma aproximadamente 10 milisegundos ejecutarlo en un microPIC

CODIGO GENERADO PBP.- Un programa PBP compilado se construye en varias etapas. Primero PBP crea el archivo ASM. Luego construye un archivo. MAC que contiene solo las macros (tomadas de la librería) usadas en el archivo ASM. Si hasta ese momento no existen errores, va al ensamblador.

El ensamblador genera su propio juego de archivos. Estos incluyen el archivo final. HEX y como opcionales, archivos de listado y depuración.

Oscilador Externo.- Por defecto PBP genera programas sobre la base de un microPIC con un cristal de 4 MHz ó un resonador cerámico. Todas las instrucciones sensibles al tiempo, asumen un tiempo de instrucción de 1 microsegundo para sus demoras.

Para utilizar un cristal de mayor frecuencia se deberá agregar la siguiente declaración:

OSC	Oscilador usado
3	3.58 MHz
4	4 MHz
8	8 MHz
10	10 MHz
12	12 MHz
16	16 MHz
20	20 MHz

TABLA 27: Declaración de los diferentes rangos de Frecuencia de Oscilacion (Velocidad de Procesamiento).

INTERRUPCIONES EN GENERAL.- Cuando ocurre una interrupción, el microPIC guarda la dirección de la próxima instrucción que debería ejecutar en el stack (pila) y salta a la dirección 4. Esto significa que se necesita una dirección extra en el stack de hardware, que solamente tiene 8.

Las librerías de rutinas de PBP pueden usar hasta 4 direcciones del stack, ellas solas. Las 4 restantes están reservadas para CALLs y GOSUBs anidados. Debe asegurarse de que sus GOSUB no estén anidados en más de tres niveles, y sin CALL dentro de ellos, para tener una dirección de stack disponible para la dirección de regreso Si su handler de interrupciones usa el

stack (haciendo un CALL ó un GOSUB ,p.ej.) necesitará espacio adicional disponible en el stack .

Después, usted necesita habilitar las interrupciones apropiadas. Eso significa dar valores al registro INTCO. Setee los bits de habilitación necesarios junto con el Global Interupt Enable. Por ejemplo;

INTCON = %10010000

Habilita la interrupción para RB0/INT. Dependiendo de la interrupción deseada, puede necesitar setear el registro PIE.

Refiérase a los manuales de Microchip para información adicional acerca de cómo usar las interrupciones. Hay ejemplos del contexto general del procesador y de cómo habilitar una interrupción en particular.

INTERRUPCIONES EN BASIC.- La forma más fácil de escribir un handler de interrupción, es escribirlo en PBP junto a una declaración ON INTERRUPT, la cual le indica a PBP que active su handler interno de interrupción (el de PBP) y salte tan pronto pueda a su handler de interrupción BASIC (creado por el usuario) después de recibir una interrupción. Usando ON INTERRUPT, cuando ocurre una interrupción, PBP simplemente marca el evento y vuelve a la tarea que estaba realizando. No salta inmediatamente al handler. Como las declaraciones de PBP no son re-entrantes (PBP debe terminar con la declaración en curso antes de ejecutar otra) puede haber considerable demora (latencia) antes de manejar a la interrupción.

Como ejemplo, el programa PBP recién comenzó la ejecución de PAUSE 10000 cuando ocurre una interrupción. PBP marca la interrupción y continúa con el PAUSE, pueden transcurrir 10 segundos antes de que se ejecute el handler de interrupción. Si se están acumulando caracteres en un port serial, muchos de ellos pueden perderse. Para minimizar este problema, use declaraciones que no tomen mucho tiempo de ejecución, Por ejemplo en lugar de PAUSE 1000 use PAUSE 1 dentro de un loop FOR...NEXT. Esto

permite completar cada declaración más rápidamente y manejar cualquier interrupción pendiente.

Es necesario un proceso de interrupción más rápido que el provisto por ON INTERRUPT, se deben usar interrupciones en lenguaje ensamblador. Lo que sucede cuando se usa ON INTERRUPT es lo siguiente: un corto handler de interrupción es colocado en la dirección 4 del microPIC. Este handler de interrupción es simplemente un RETURN. Esto envía el programa de vuelta a lo que estaba haciendo antes de ocurrir la interrupción. No requiere guardar ningún contexto del procesador. Lo que esto no hace es re-habilitar el Global Interrupts, como hace el Retfie. Un Call a una pequeña subrutina es colocado después de cada declaración en el programa PBP cuando se encuentra un ON INTERRUPT. Esta pequeña subrutina verifica el estado del bit de Global Interrupt Enable. Si está apagado hay una interrupción pendiente, por lo que apunta al handler de interrupción del usuario. Si no el programa continúa con la próxima declaración BASIC, después de lo cual se verifica nuevamente el bit GIE, y así sucesivamente.

Cuando se encuentra una declaración RESUME después del handler de interrupción BASIC, setea el bit GIE para rehabilitar las interrupciones y vuelve donde estaba el programa cuando ocurrió la interrupción. Si se indica en RESUME una etiqueta hacia donde saltar, la ejecución continuará en esa dirección. En ese caso, se pierden todas las direcciones previas de regreso.

DISABLE detiene PBP insertando un Call al verificador de interrupción después de cada declaración. Esto permite que se ejecuten secciones de código sin la posibilidad de ser interrumpidas. ENABLE permite la inserción para continuar.

DISABLE debe ser colocado antes que el handler de interrupción para que éste no sea arrancado cada vez que se chequee el bit GIE. Si por alguna razón se desea apagar las interrupciones después que se encuentra un ON INTERRUPT, no debe apagar el bit GIE. Apagando este bit, se le indica a

PBP que ha sucedido una interrupción y esto ejecutará el handler de interrupción por siempre. En su lugar haga:

INTCON = $80

Esto deshabilita todas las interrupciones individuales, pero deja seteado el bit GIE.

Una nota final acerca de las interrupciones en BASIC. Si rl programa usa:

loop: goto loop

Espera ser interrumpido, eso no sucederá. Recuerde que la bandera de interrupción es chequeada después de cada instrucción. Realmente no hay un lugar donde chequear después de un GOTO. Inmediatamente salta al loop, sin chequear la interrupción. Debe colocarse alguna declaración dentro del loop, para que haya un chequeo de interrupciones.

I / O DIGITAL.- Los programas PBP operan directamente en los registros PORT y TRIS .Aunque esto da ventajas de velocidad y tamaño de RAM / ROM, también tiene sus contras.

INSTRUCCIONES DE BAJA POTENCIA.- Cuando el Watch Dog Timer despierta al microPIC del modo de sueño (SLEEP), la operación recomienza sin modificar el estado de los pines de I/O. Por razones desconocidas, cuando BASIC Stamp recomienza la ejecución después de una instrucción de baja potencia (NAP ó SLEEP) los pines de I / O quedan con disturbios por aproximadamente 18 mseg. Los programas PBP hacen uso de la coherencia de los I / O de los microPIC. NAP y SLEEP no alteran los pines de I/O.

SIN VARIABLES AUTOMATICAS.- El compilador PBP no crea variables en forma automática, como B0 ó W0 .Deben ser definidas usando VAR. Se entregan dos archivos: BS1DEFS.BAS y BS2DEFS.BAS que definen las variables estándar BS1 y BS2 .Sin embargo, es recomendable que usted asigne sus propias variables.

OPERADORES MATEMATICOS.- Las operaciones matemáticas tienen precedencia de operación. Significa que no son evaluadas estrictamente de izquierda a derecha como están originalmente en BASIC Stamp y en el compilador PBP. Esta precedencia significa que multiplicación y división son realizadas antes que suma y resta, por ejemplo. Se debe usar paréntesis para agrupar las operaciones en el orden en que deben ser realizadas.

Operador matemático	Descripción
+	Suma
-	Resta
*	Multiplicacion
**	16 bits superiores de la multiplicacion
*/	16 nits medios de la multiplicacion
/	Division
//	Resto (módulo)
<<	Desplazamiento izquierdo
>>	Desplazamiento derecho
ABS	Valor absoluto
COS	Coseno
DCD	2m decodificador
DIG	Digito
MAX	Maximo *
MIN	Minimo *
NCD	Codificar

REv	Invertir bits
SIN	Seno
SQR	Raiz cuadrada
&	Bit inteligente AND
÷	Bit inteligente OR
^	Bit inteligente EXCLUSIVE OR
~	Bit inteligente NOT
& /	Bit inteligente NOT AND
÷ /	Bit inteligente NOT OR
^ /	Bit inteligente NOT EXCLUSIVE OR

TABLA 28: Muestra el orden jerárquico de los operadores.

DEPURACION.- El depurador DEBUG en PBP no es un caso especial de SEROUT como en Stamp .Tiene su propia rutina que trabaja con un pin fijo y un baud rate. Puede usarse para enviar información de depuración a un terminal ó a un dispositivo serial. Los signos de interrogación (?) en las declaraciones DEBUG se ignoran. No se debe usar el modificador ASC?

IF...THEN.- BASIC Stamp y el compilador PBP original solo permitían especificar un label después de un IF...THEN .PBP permite la construcción de IF...THEN...ELSE...ENDIF ,así como la ejecución de un código como resultado de un IF ó un ELSE.

MAX y MIN.- Las funciones MAX y MIN han sido bastante modificadas respecto a las de Stamp y el compilador original PBP. MAX devuelve el máximo de dos valores. MIN devuelve el mínimo de dos valores. Es más parecido a los demás BASIC y no tiene los problemas de límites en 0 y 65535 del Stamp. En la mayoría de los casos basta intercambiar MAX y MIN en los programas Stamp para que funcionen en PBP.

3.1.3.3 Ejemplos de programación.

Programa para controlar un Semáforo.

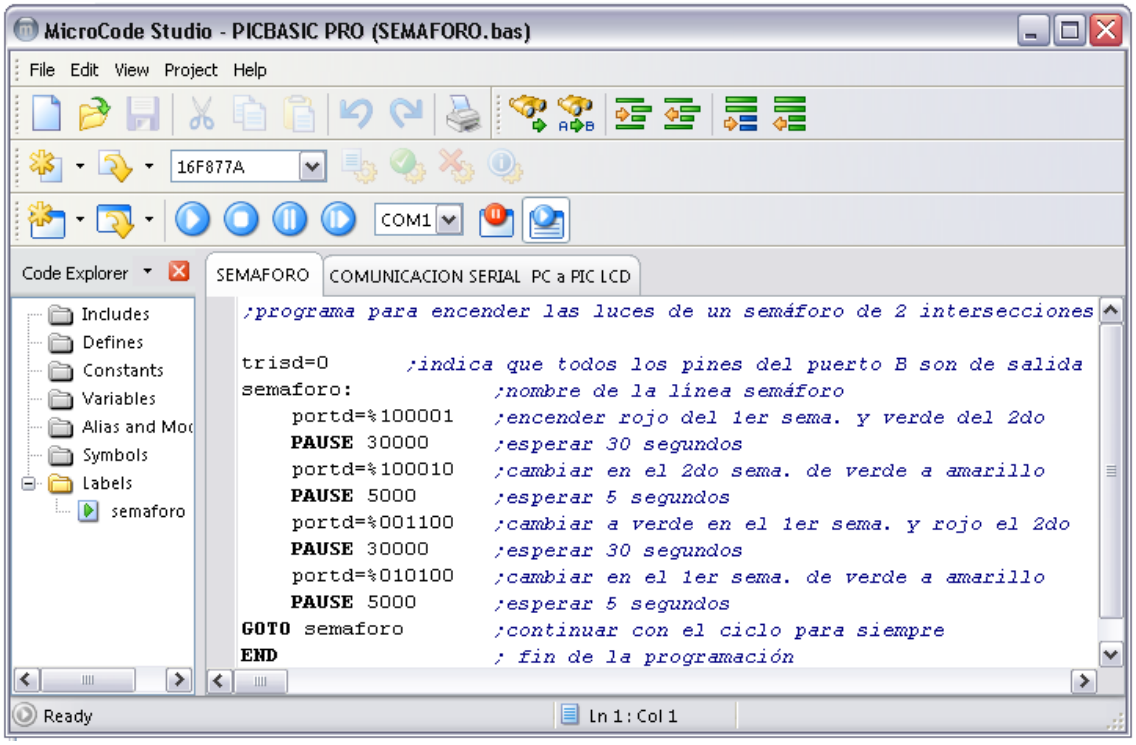

FIGURA 60: Programa de un semáforo de dos intercepciones.

Programa: Control de 8 Relés desde una PC.

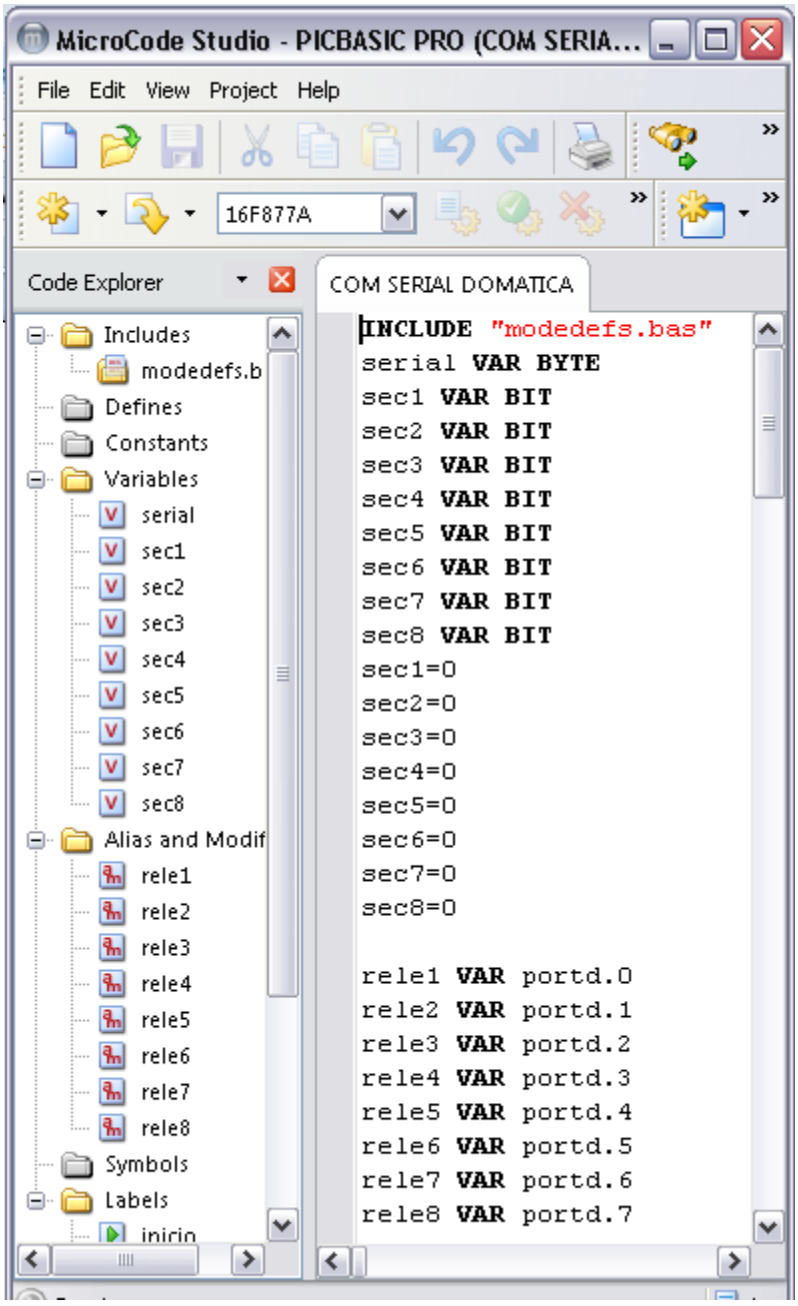

FIGURA 61: Programa de un Control de 8 Relés desde una PC.

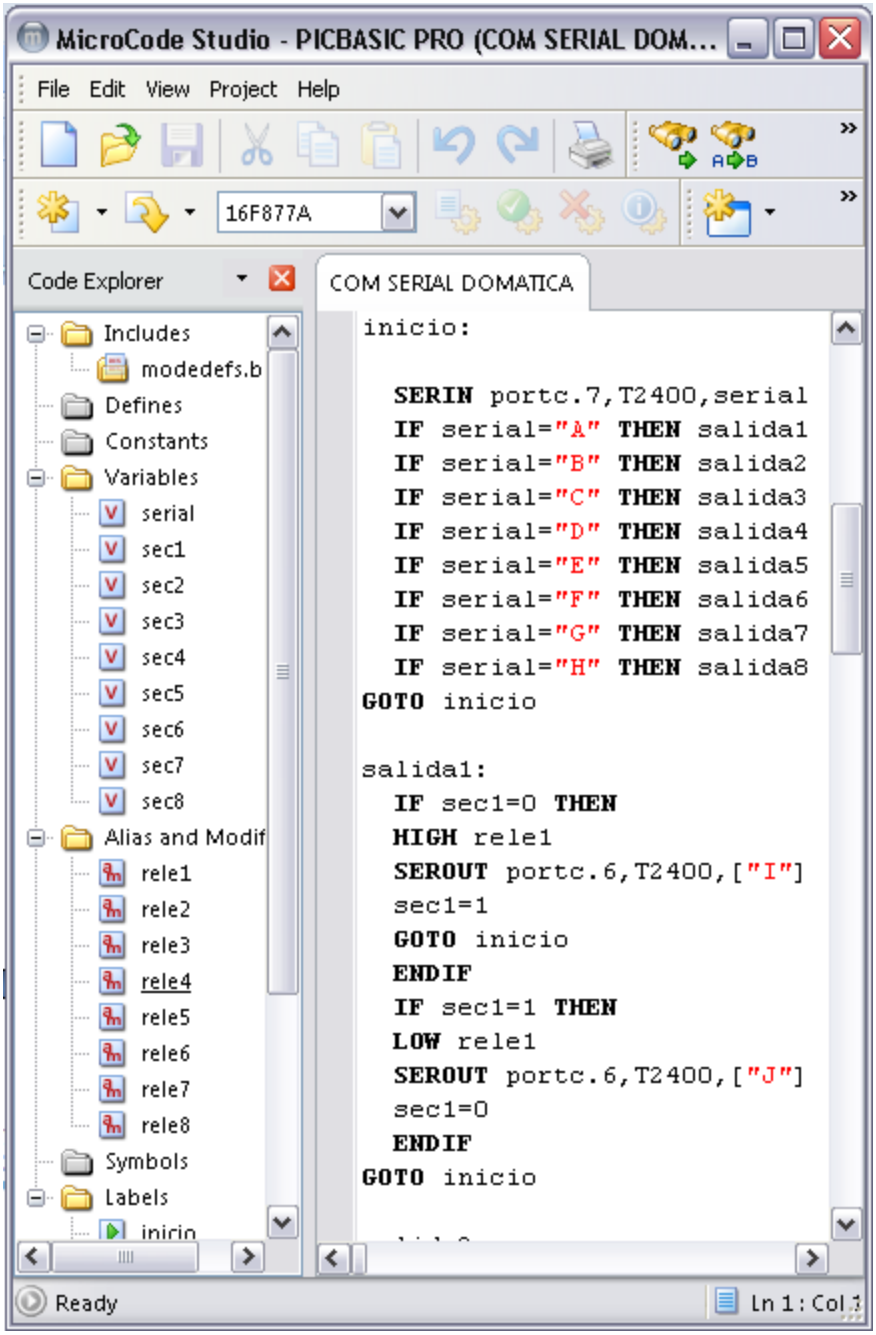

FIGURA 62: Programa de un Control de 8 Relés desde una PC.

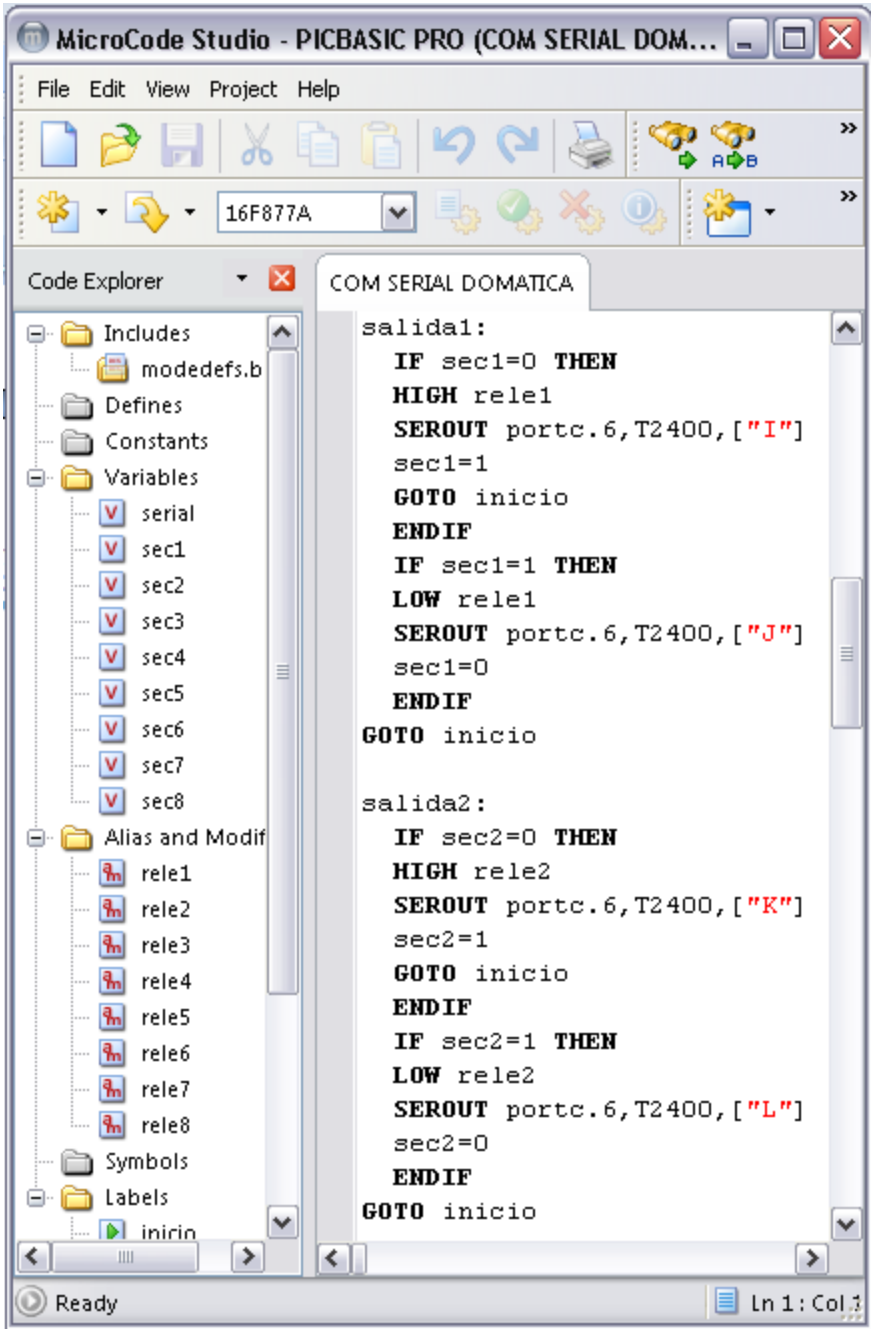

FIGURA 63: Programa de un Control de 8 Relés desde una PC.

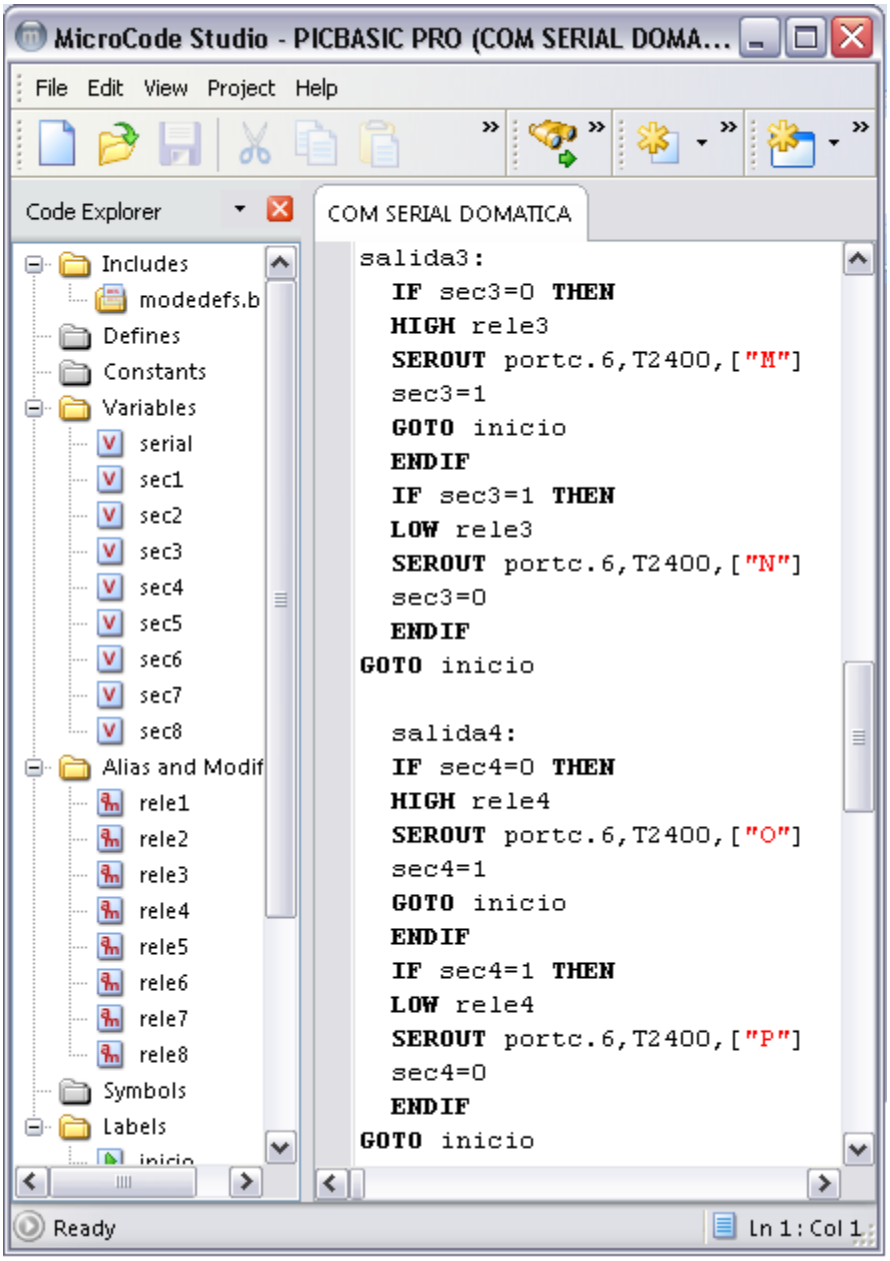

FIGURA 64: Programa de un Control de 8 Relés desde una PC.

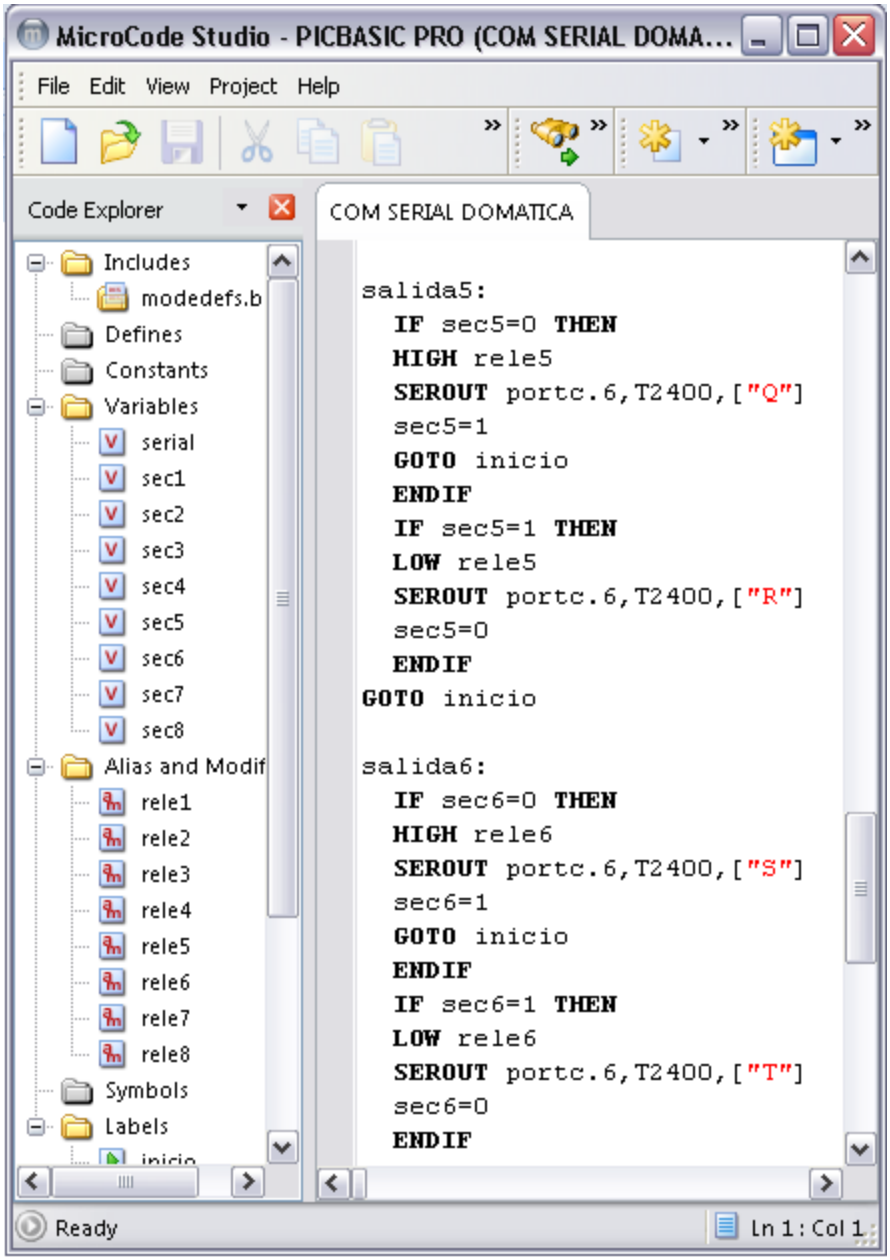

FIGURA 65: Programa de un Control de 8 Relés desde una PC.

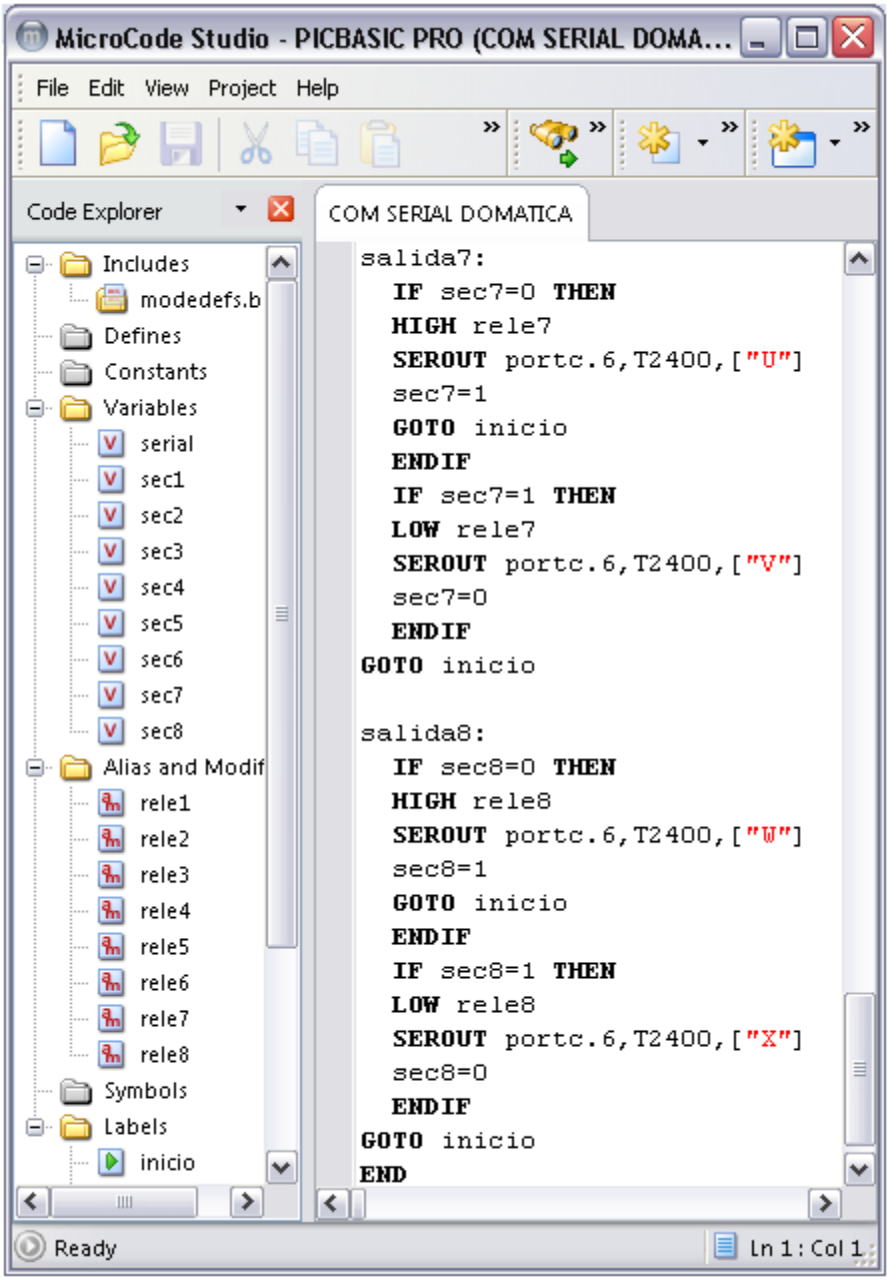

FIGURA 66: Programa de un Control de 8 Relés desde una PC.

En las figuras 61 al 66 vemos un programa cuyo funcionamiento es de controlar 8 Relés del PLC desde una PC enviando desde esta las 8 primeras letras mayúsculas del alfabeto para encender los relés y además envíen una letra del alfabeto (Mayúsculas) al PC por cada relé para saber que esta encendido cada uno de ellos o a su vez apagados.

Programa de comunicación serial de PC al PLC.

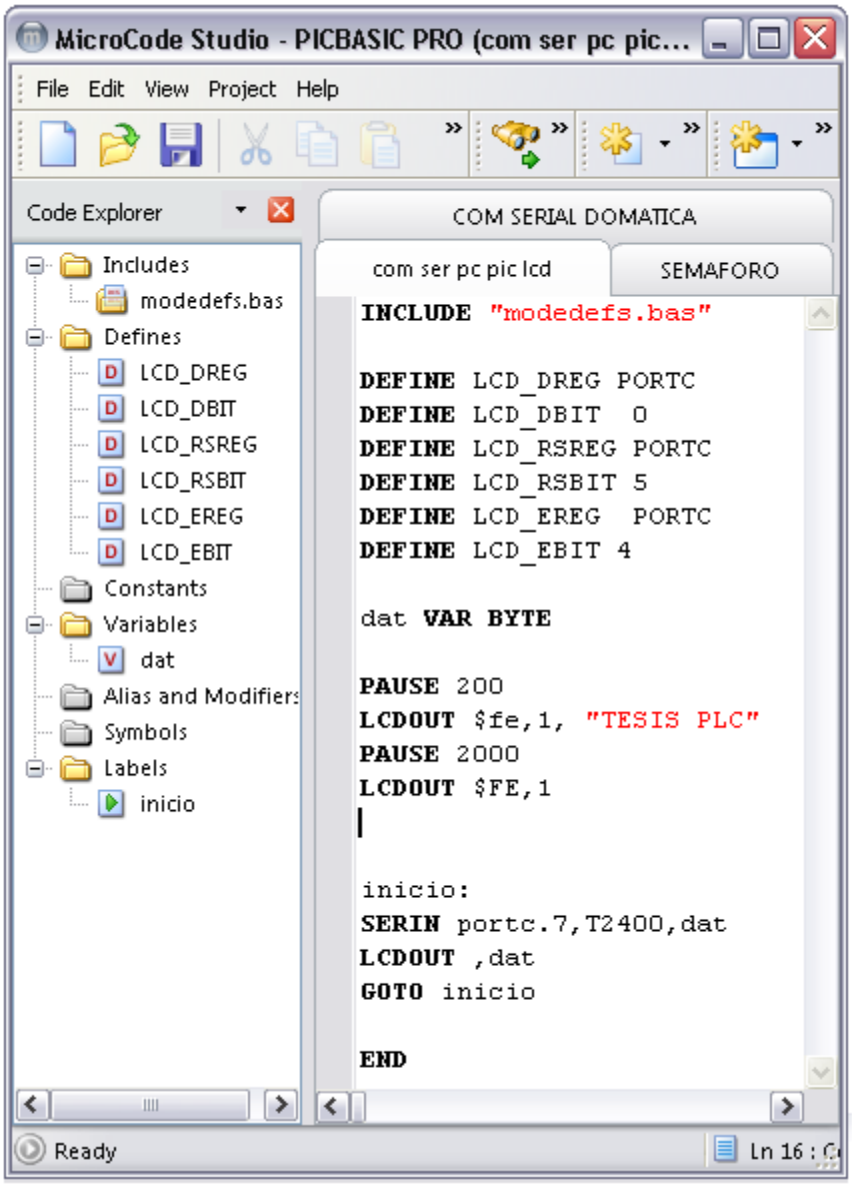

FIGURA 67: Programa cuyo funcionamiento es de enviar un mensaje al PLC y que en este se visualize mediante la pantalla LCD.

4 Resultados.

4.1.1 Requerimiento mínimo del sistema.

- Tensión de alimentación al PLC de 12VDC.
- Corriente de alimentación al PLC 1.5A.
- Procesamiento mínimo del PLC 4Mhz.
- Sistema Operativo del ordenador Windows 2000 o XP (SP1, SP2 o SP3) para Programación del dispositivo y comunicación serial.

4.1.2 Pruebas con el dispositivo.

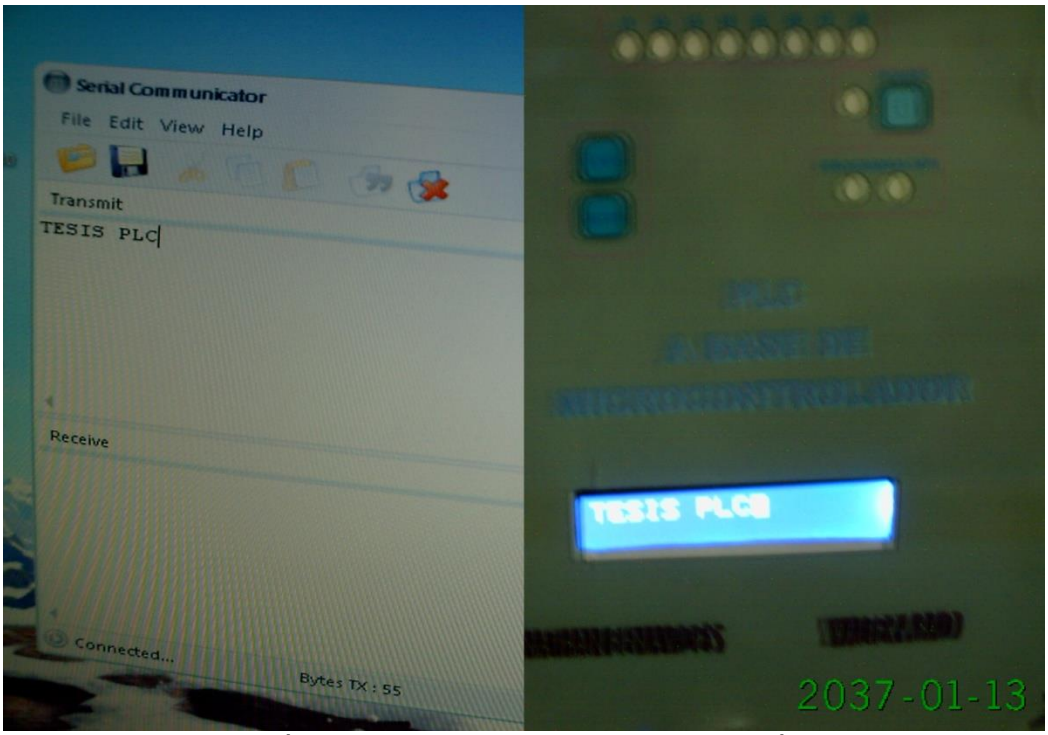

FIGURA 68: Comunicación serial del computador con el dispositivo (Envía un mensaje desde la PC al LCD del PLC).

Comunicación serial del computador con el dispositivo (Envía un mensaje desde la PC al LCD del PLC).

5 Conclusiones

Se ha verificado que los sistemas automatizados, hoy en día tienen una función muy importante como es el facilitar el trabajo al hombre, aumentar la producción, así como la calidad de la misma, además mejora las condiciones de trabajo aumentando la seguridad para los operadores, etc.

El Diseño de un controlador lógico programable (PLC) basado en un microcontrolador PIC facilita a la persona para ya no aplicar su fuerza para realizar su trabajo.

La amplia gama de aplicaciones de los PLC. Tradicionalmente utilizados en el control secuencial, se concluye que son igualmente efectivos en el control realimentado, extendiendo todas sus capacidades y ventajas a esta rama del control automático.

Para programar el PIC se necesita el programa Microcode Studio esta herramienta permite programar en lenguaje Basic que es bastante fácil de aprender.

Se observó que el microcontrolador emite señales de lógica digital TTL cuyos voltajes son de +5V (Estado lógico Alto) y 0V (Estado lógico Bajo), por lo que necesariamente se implementó un driver que cumpla la función de convertir niveles de la norma RS232 a la norma TTL, el CI. MAX232 es el driver que cumple con estas características, el mismo que dispone de 2 juegos de transmisores y receptores, de los cuales en este proyecto solo se ocupò un par de ellos.

Se comprobó que el microcontrolador PIC16F877A en su memoria sólo reside un programa destinado al gobernar una aplicación determinada; sus líneas de entrada / salida soportan el coleccionado de los sensores y actuadores del dispositivo a controlar y todos los recursos complementarios son disponibles tienen como única finalidad atender sus requerimientos.

6 Recomendaciones.

Se recomienda dar mantenimiento preventivo al controlador lógico programable (PLC) basado en un microcontrolador PIC con el fin de evitar desgaste y daño.

Se debe tener un conocimiento bien claro del controlador lógico programable (PLC) para poderlo utilizar de la mejor manera ya que si no los tienen ocasionaran daño en el equipo.

Al microcontrolador PIC se debe manejar con mucho cuidado ya que tiende a dañarse o quemarse por ser muy sensible al tacto humano

Otro aspecto al que se le debe poner atención, cuando se utiliza el PLC, es al nivel de

Voltaje, ya que dichos niveles debe ser adecuado a las especificaciones antes ya dichas.

Con el fin de prevenir daños al equipo y accidentes, el programa que se le aplique al PLC, debe estar protegido de tal manera que no pueda ser modificado por personal no calificado.

Cuando se le suspende la alimentación al PLC es importante suplir de energía suplementaria durante una suspensión de energía no programada. Esto con el fin de que el PLC sea capaz de llevar el sistema de manera controlada a un estado seguro.

7 BIBLIOGRAFÍA.

LIBROS:

1) Angulo Usategui, José María; Angulo Martínez, Ignacio. "Microcontroladores PIC. Diseño práctico de aplicaciones. 2ª edición" (1999). Editorial McGraw Hill. Madrid.

2) Manual de laboratorio de electrónica de Industrial II (Electrónica de Potencia). ENPES La Habana CU 1985. Por CORREA SAURA ARTURO.

3) Fundamento de electrónica Industrial 2da Edición. MARCOMBO, BOXAREU. Por MORRIS N. M.

4) Manual de laboratorio de electrónica de Industrial III (Electrónica de Potencia). ENPES La Habana CU 1986. Por MORALES ACOSTA MIGUEL; FLORENCIA PUERTO, ALDO ALVAREZ RUBLOS ISIDRO.

5) Aprenda rápidamente a programar Microcontroladores PIC. Por CARLOS A. REYES.

6) Diseño de equipos electrónicos. EMPES La Habana CU 1990. DIANA UNGUEL MARIA.

SITIOS WEB:

1) http://www.comunidadelectronicos.com/foros.htm

2) http://www.datasheetcatalog.com/datasheets_pdf/M/A/X/2/MAX232.shtml

http://www.datasheetcatalog.net/es/datasheets_pdf/2/N/3/9/2N3904.shtml

http://www.datasheetcatalog.com/datasheets_pdf/4/N/2/7/4N27.shtml

http://www.datasheetcatalog.com/datasheets_pdf/4/N/2/7/4N27.shtml

3) http://electronikdtrujillo.blogspot.com/2009/01/microcontrolador-pic16f877-en-espanol.html

http://electronred.iespana.es/condensador.htm

http://es.wikipedia.org/wiki/Resistencia_el%C3%A9ctrica

4) http://www.forosdeelectronica.com/forum-19.html

5) http://www.lawebdelprogramador.com/news/new.php?id=58&texto=Microcontroladores

http://r-luis.xbot.es/edigital/ed04.html

6) www.microchip.com

http://www.modtronix.com/product_info.php?products_id=250

http://ww1.microchip.com/downloads/en/DeviceDoc/30292c.pdf

7) http://perso.wanadoo.es/pictob/icprog_jdm.htm#programacion_de_pic_con_ic_prog

http://www.portalpmr.com/articulos/resistencias-electricas-vp22.html

8) http://www.scribd.com/doc/101179/pic16f877-en-espanol2

9) http://todopic.mforos.com/

http://www.todopic.com.ar/pbp_sp.html#introduccion

http://www.todopic.com.ar/

10) http://usuarios.lycos.es/patricio/ensam/ensam1.htm

http://usuarios.lycos.es/patricio/ensam/ensam2.htm

http://usuarios.lycos.es/patricio/ensam/ENSAM3.HTM

http://usuarios.lycos.es/patricio/ensam/ENSAM5.HTM

http://www.unicrom.com/tutoriales.asp

11) http://www.yoreparo.com/foros/digital/index.html