UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

FACULTAD DE INGENIERÍA

REALIZACIÓN DE UN PLC DIDÁCTICO

T E S I S

QUE PARA OPTAR EL GRADO DE:

MAESTRO EN INGENIERÍA

INGENIERÍA ELÉCTRICA – SISTEMAS ELECTRÓNICOS

P R E S E N T A:

ROBERTO MOLERO MILO

TUTOR:

M. en I. MIGUEL ANGEL BAÑUELOS SAUCEDO

2006

JURADO ASIGNADO:

Presidente: M. en I. Luis Arturo Haro Ruiz

Secretario: M. en I. Lauro Santiago Cruz

Vocal: M. en I. Miguel A. Bañuelos Saucedo

1$^{er.}$ Suplente: M en I. Sergio Quintana Thierry

2$^{do.}$ Suplente: M. en I. José Castillo Hernández

Lugar donde se realizó la tesis:

CCADET, Ciudad Universitaria **México, D.F.**

Tutor de tesis:

M. en I. Miguel Ángel Bañuelos Saucedo

Agradezco el inmenso apoyo y la asesoría brindada para el desarrollo de este trabajo de tesis a mi tutor, M. en I. Miguel Angel Bañuelos Saucedo; a los miembros del jurado: M. en I. José Castillo Hernández, M. en I. Sergio Quintana Thierry, M en I. Luis Arturo Haro Ruiz y al M. en .I Lauro Santiago Cruz por su notable profesionalismo y valioso tiempo brindado en la revisión de este trabajo.

A todos mis compañeros y amigos del laboratorio de electrónica y de la maestría (Arturo Nevárez, Cuauhtemoc González, Francisco Hernández, Gerardo Rayo, Jorge Cruz, Jorge Morales, Nadia Launizar y Ricardo Damián), por su gran apoyo, valiosos comentarios y por todo lo mucho que aprendí de ellos en esos años.

También agradezco al CONACyT por la beca otorgada para la realización de la maestría, a la Facultad de Ingeniería por brindarme excelentes bases para mi formación profesional, al CCADET y a la UNAM por su excelente apoyo y haberme brindado el mejor espacio para mi superación personal.

Por supuesto reconozco que gracias a la lindísima Adriana Pelusi de Icaza el formato y muchos pequeñitos pero muy importantes detalles de este trabajo no se hubieran completado como debe, lo bien hecho de este trabajo es gracias a tí xxx.

Es obvio que sin el inmensurable apoyo y compresión de mi familia nada de esto y más se pudo haber logrado, Gracias.

ÍNDICE

INTRODUCCIÓN

El acelerado desarrollo tecnológico, tan característico en el mundo globalizado de estos últimos años, hace imperativo el mantenerse debidamente actualizado e informado acerca de las características más novedosas e importantes de las tecnologías aplicadas en la producción industrial de hoy en día. Las fuentes de actualizaciones se encuentran principalmente en la industria y en las grandes universidades donde se realizan investigaciones, y es ahí donde la evolución de la tecnología comienza.

Actualmente la automatización ha alcanzado un nivel tal que prácticamente no se puede pensar en desarrollo tecnológico sin considerar procesos o sistemas en dónde ésta se ausente. Y esto es debido a que la automatización reduce significativamente tanto los tiempos de manufactura como el material empleado [1] [2], además de reducir las mermas y las horas hombre teniéndose un producto final de mayor calidad.

La enseñanza de la automatización y de la programación de los PLC (acrónimo del inglés *Programmable Logic Controller*) ha entrado de una manera preponderante en casi todas las disciplinas técnicas de los institutos técnicos y profesionales. Es por ello que resulta obligado facilitar los medios y técnicas para una instrucción fácil, económica y coherente a las tecnologías existentes.

Crear y organizar un laboratorio o taller eficiente, funcional y flexible, intentando limitar al máximo las inversiones en términos económicos y de instalación logística, representa para cualquiera un auténtico problema. De hecho, la realización de un ejercicio de automatización consiste en la adecuación de puntos de trabajo que incluyan un PLC completo de sistema de desarrollo (que la mayoría de las veces funciona basándose en una PC), y uno o más paneles para la simulación de la instalación que es objeto del ejercicio en cuestión [8] [12].

OBJETIVOS

- Con la finalidad de facilitar la comprensión, el estudio y el entrenamiento de la utilización de los PLC´s mediante ejercicios completos de automatización, se propone diseñar y construir un PLC didáctico que conforme un sistema de desarrollo inspirado en los dispositivos comerciales.

- Para fomentar el aprendizaje de la programación de un PLC se creará un lenguaje de programación (en formato lista de instrucciones) inspirado en los estándares actuales y que soporte gran variedad de instrucciones, contadores y temporizadores.

- Debido que el PLC será un dispositivo de aprendizaje, el diseño de éste estará orientado a usuarios inexpertos y tendrá semejanza a dispositivos comerciales, además la memoria del PLC tendrá que ser no volátil para no perder el programa por un corte de energía.

ANTECEDENTES

Hubo una época en la que el control de procesos industriales de diversos tipos y niveles de complejidad se implementaba mediante el uso de relevadores y una intrincada red de cables [27]. En la industria automotriz se basaban en dichos esquemas para proveer una solución a su producción donde el problema radicaba en el momento en que se cambiaba de modelo de automóvil, lo que representaba el cierre de la planta hasta por un mes, ya que el reacondicionamiento y acomodo de los cables y paneles resultaba un proceso largo que consumía tanto tiempo como recursos humanos [1] [2] [3]. Además, cuando los cambiantes requisitos de la producción en un sistema de control se manifestaban, se tenía la consecuencia de que el costo se elevara considerablemente. Tan solo el hecho de realizar pequeños cambios se traducía en retrasos significativos.

Puesto que los relevadores son dispositivos electromecánicos, estos tienen vida limitada dado que exigen una buena adherencia entre sus terminales eléctricas (que frecuentemente se carbonizan) y el uso los deteriora, lo que conduce a que la localización de averías resultara un proceso tedioso y muy extenso. Además, claro está, que el cableado inicial resulta muy complicado y elaborado. Imperaba tener una alternativa a los circuitos secuenciales constituidos con relevadores y temporizadores [32].

Un PLC es un dispositivo que fue inventado para reemplazar los circuitos con relevadores y temporizadores para el control de máquinas en diversos procesos [22]. Con este dispositivo los cambios en el cableado son por mucho menores, ya que el cambio de las interconexiones se realiza principalmente mediante software; además, los temporizadores y contadores son implementados por software. Los PLC´s trabajan de acuerdo a sus entradas y dependiendo del estado de éstas, se enciende o apagan sus salidas. El usuario introduce un programa, obteniendo los resultados deseados.

De una manera general y más formal podemos definir al controlador lógico programable como toda máquina electrónica diseñada para controlar en tiempo real y en un medio industrial procesos secuenciales de control. El estándar NEMA (*National Electrical Manufacturers Association*), define a un PLC como un aparato electrónico digital que utiliza una memoria programable para el almacenaje interno de instrucciones para ejecutar funciones específicas tales como lógicas, secuenciales, de temporización, conteo y

aritméticas para controlar varios tipos de máquinas o procesos a través de módulos digitales o analógicos de entrada-salida [35].

Las fábricas automatizadas deben proporcionar en sus sistemas, alta confiabilidad, gran eficiencia y flexibilidad. Una de las bases principales de tales fábricas es un PLC. Este dispositivo fue inicialmente introducido en los años setenta con el objetivo tal de diseñar un aparato que reemplazara los complicados sistemas de control existentes en aquella época. Como respuesta ante tal demanda, Bedford Associates propuso algo llamado MODICON (Modular Digital Controller) o controlador modular digital, al mayor fabricante de automóviles en EE.UU. [45] mientras que otras compañías propusieron esquemas basados en computadoras, una de las cuales fue basada sobre el PDP-8 [29] (Computadora multiusuario basada en tarjetas intercambiables). El MODICON 084 trajo al mundo el primer PLC a la producción comercial.

La capacidad de comunicarse con otros PLC´s comenzó a aparecer aproximadamente en 1973. El primer sistema era el Modbus de Modicon. El PLC podría ahora comunicarse con otro PLC y podían estar en otra localidad distinta de la máquina que controlan. Podían también ser utilizados para transmitir y recibir voltajes variables lo cual los introdujo al mundo analógico. Desafortunadamente, la carencia de la estandardización [2], reunida con tecnología que cambiaba continuamente, ha hecho las comunicaciones del PLC una pesadilla de protocolos incompatibles y de redes físicas.

El PLC fue evolucionando con nuevos componentes electrónicos, tales como microprocesadores de alta velocidad [10], agregándole funciones especiales para el control de procesos más complejos. Hoy los controladores programables son diseñados usando lo último en diseño de microprocesadores y circuitos electrónicos, lo cual proporciona una mayor confiabilidad en su operación en aplicaciones industriales donde existen peligros debido al medio ambiente, alta repetibilidad, altas temperaturas, ruido de ambiente y/o eléctrico, suministro de potencia eléctrica no confiable, vibraciones mecánicas, etc.

Su programación y manejo pueden ser realizados por personal con conocimientos eléctricos o electrónicos, sin previos conocimientos sobre informática.

La programación del PLC puede ser hecha por una unidad de programación que suele ser en forma de calculadora [1]. Es la forma más simple de programar el equipo, y se suele reservar para pequeñas modificaciones del programa o la lectura de datos en el lugar

de colocación del equipo. También se puede usar una consola de programación, la cual es una terminal que proporciona una forma más cómoda de realizar el programa de usuario y observar parámetros internos del PLC de forma similar que con una PC.

El modo más empleado para programar un PLC es mediante una computadora tipo PC. Permite programar desde una computadora personal estándar, con todo lo que ello supone: herramientas más potentes, posibilidad de almacenamiento en soporte magnético, impresión, transferencia de datos, monitorización mediante software SCADA (del inglés Supervisory Control And Data Acquisition), etc.

Dentro de las ventajas de un PLC se enumeran las siguientes:

- Un solo PLC puede controlar simultáneamente diferentes máquinas con un programa independiente para cada una de ellas.
- En un PLC cuando se desea modificar o corregir un diseño, es solo cuestión de minutos ya que dichos cambios se corrigen cambiando la secuencia del programa sin necesidad de realambrar.
- Los PLC´s poseen prácticamente cualquier número de contactos por cada bobina ya que estos se implementan por software, el límite seria la memoria disponible. Un cambio de contactos se realizaría notoriamente en un tiempo breve, ya que no sería un cambio principal en un panel de control, sino en el programa de instrucciones.
- La tecnología creciente lo hace posible, ya que cada vez la capacidad de los circuitos integrados es mayor tanto en procesamiento como en almacenamiento, siendo el costo de la producción de estos cada vez más barata; además, el tamaño se ha reducido notablemente lo que conduce también a la reducción de encapsulados y empaques.
- Un PLC programado tiene la posibilidad de simular el comportamiento del programa introducido antes que entre en funcionamiento.

Para hacer notar la ventaja de un PLC, la tabla 1 muestra algunas características que lo diferencian de un sistema cableado [21].

CARACTERÍSTICAS	SISTEMA CABLEADO	PLC
Flexibilidad de adaptación al proceso	Baja	Alta
Hardware estándar para distintas aplicaciones	No	Sí
Posibilidades de ampliación	Bajas	Altas
Interconexiones y cableado exterior	Mucho	Poco
Tiempo de desarrollo del proyecto	Largo	Corto
Posibilidades de modificación	Difícil	Fácil
Mantenimiento	Difícil	Fácil
Herramientas de prueba	No	Sí
Stocks de mantenimiento	Medios	Bajos
Modificaciones sin parar el proceso (“on-line”)	No	Sí
Costo para series pequeñas	Alto	Bajo
Estructuración en bloques independientes	Difícil	Fácil

Tabla 1. Características entre sistemas.

1. FUNDAMENTOS DE UN PLC

1.1 Lógica alambrada

Durante muchos años las funciones de automatización y control fueron realizadas empleando lo que se denomina lógica alambrada con relevadores, que se basa en la activación o cierre de contactos o interruptores que a su vez se encargan de encender bobinas de relevadores que cierran otros contactos [31].

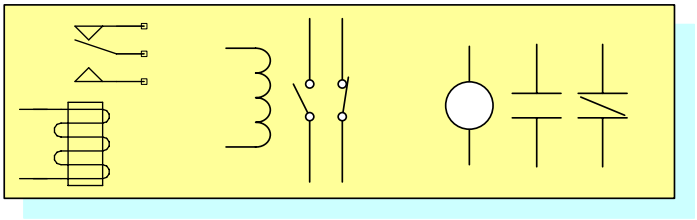


Figura 1. Diversos símbolos de un solo relevador.

Por tanto, la acción de encendido y apagado de algunos relevadores puede estar condicionada por otros eventos individuales; entonces las condiciones exactas necesarias para un proceso específico dependen de la forma en que los contactos y relevadores se encuentren conectados entre sí.

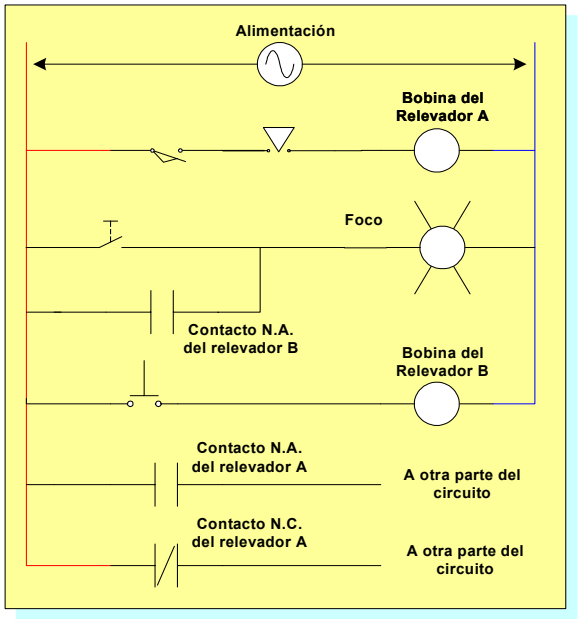


Figura 2. Diagrama de un sistema de lógica alambrada.

1.2 Partes de un circuito de control

Un circuito de control eléctrico para controlar un sistema industrial puede dividirse en tres partes diferentes: entrada, lógica y salida.

La sección de entrada, en ocasiones conocida como sección de recolección de información, consiste en todos los dispositivos que proporcionan parámetros del operador humano y del sistema de información a los circuitos. Algunos de los dispositivos de entrada más comunes son botones, interruptores de límites mecánicos, interruptores de presión y fotoceldas.

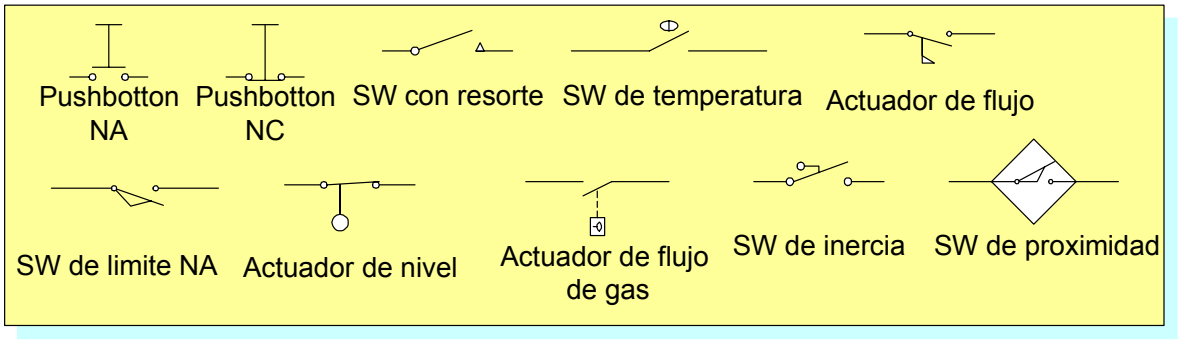


Figura 3. Dispositivos de la sección de entrada.

La sección lógica, en ocasiones llamada sección de toma de decisiones, es aquella parte del circuito que actúa sobre la información proporcionada por la sección de entrada. Toma decisiones con base en la información recibida y envía órdenes a la sección de salida [3]. Los circuitos de la sección lógica por lo general se construyen con relevadores magnéticos, circuitos de transistores discretos o circuitos de transistores integrados.

La sección de salida consiste en los dispositivos que toman las señales de salida de la sección lógica y las convierten o amplifican en una forma útil destinadas para el accionamiento de algún dispositivo actuador. Los dispositivos actuadores más comunes son bobinas, motores, focos, etc.

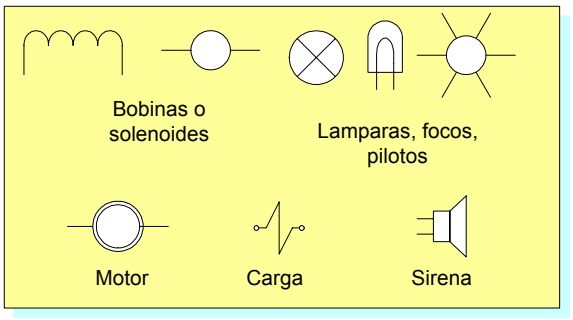


Figura 4. Dispositivos de la sección de salida.

1.3 Método diferente de un sistema de control

La relación entre las partes de un sistema de control basado en relevadores conforma en si un controlador que puede realizar diversas funciones, su único defecto desde el punto de vista de un usuario industrial es que no es modificable fácilmente. Si es necesario realizar modificaciones, se necesita cambiar las conexiones reales del cable. La figura 5 muestra qué se puede generalizar para cualquier sistema de control industrial sea alambrado con relevadores o no.

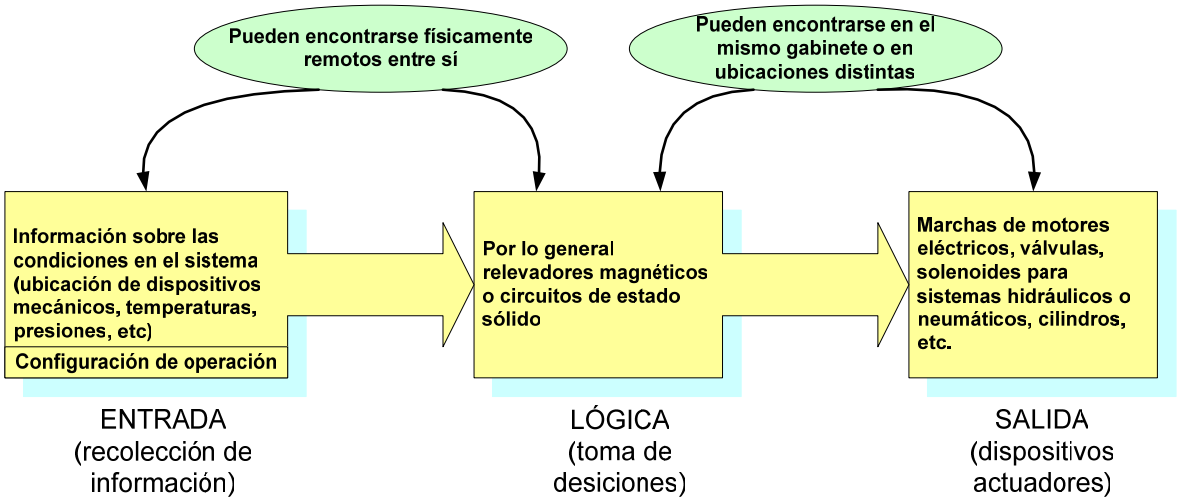


Figura 5. Sistema de control industrial.

En la actualidad se utiliza un método diferente y popular para la construcción de sistemas lógicos de control industriales en donde la toma de decisiones del sistema se lleva a cabo por instrucciones codificadas (la secuencia de instrucciones codificadas que controla el desempeño del sistema se conoce como un programa), que se almacenan en un chip de memoria y se ejecutan en un microprocesador [45]. Ahora, si el sistema de control se requiere cambiar solo es necesario modificar el programa del chip de memoria; por tanto, a

dicho sistema se le conoce como programable. Si todos los componentes necesarios de control se ensamblan y venden como una unidad completa, a dicha unidad se le conoce como PLC [37].

Dado que un PLC vino a sustituir a los sistemas alambrados con lógica de relevadores, este heredó de sus predecesores muchas características tales como los diagramas lógicos de comportamiento del sistema, así como las representaciones gráficas de las entradas y de las salidas, además de las mismas secciones de entrada y de salida (refiriendo al mismo hardware).

1.4 Partes de un PLC

Se puede considerar que los PLC´s tienen tres partes [3]: la sección de Entrada/Salida, el procesador y el dispositivo de programación o terminal. Las partes se ilustran en la figura 6 estando entre líneas punteadas.

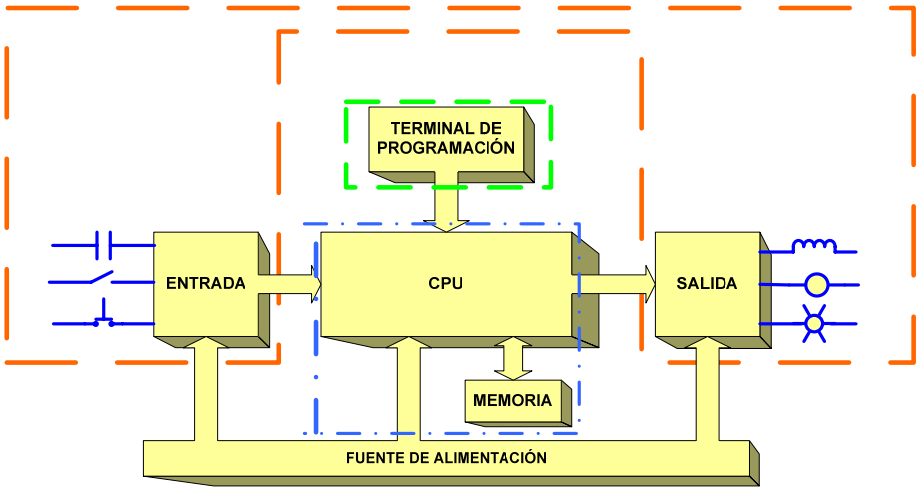


Figura 6. Partes de un PLC.

1.5 Sección de Entrada/Salida

La sección de Entrada/Salida de un PLC tiene a su cargo la función de interconectar de manera segura y eficiente a los dispositivos industriales de alta potencia al sistema de circuitos de baja potencia (el procesador o CPU,de las siglas en inglés *Central Processing Unit*) que almacena y ejecuta el programa de control. Dicha sección está constituida separadamente por los módulos de entrada y de salida [31]. Típicamente hay 4, 8, 12, o 16 terminales en cada módulo y pueden corresponder al mismo número de terminales de

entradas y de salidas de estos. Aunque no es regla general, ya que puede haber pequeños sistemas con 8 entradas y 4 salidas, mientras que en otros mucho más grandes pueden haber decenas de módulos de entrada y otros tantos de salida agrupados en racks.

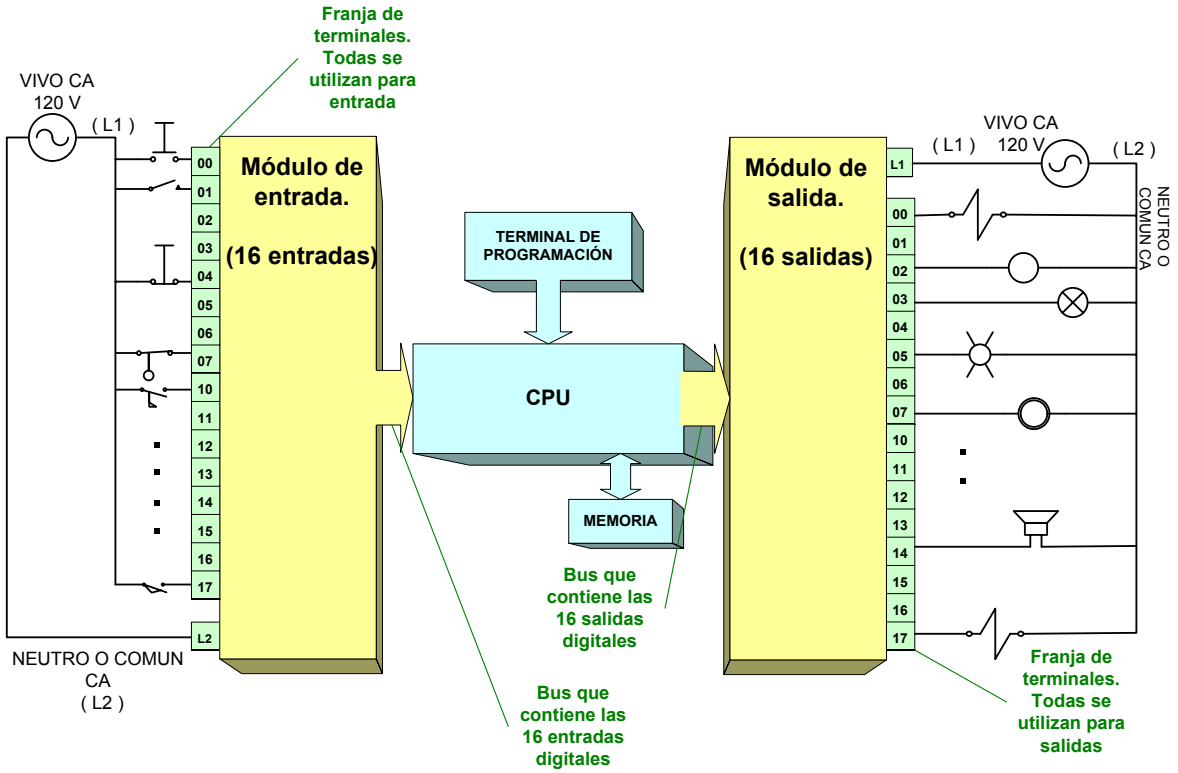


Figura 7. Representación de la conexión de los módulos E/S del PLC.

En cada una de las terminales del módulo de entrada, se hallan acondicionadores de señal cuya función única es sensar la presencia o ausencia de señal de entrada. Dado que cada terminal recibe por lo general señales cuya tensión puede ser de 120 VCA o bien 12 o 24 VCD según sea el modelo del PLC, y éstas tienen que ser transformadas a tensiones pequeñas comparables con la siguiente etapa electrónica a la cual se van a acoplar. Para ello se utilizan diversos circuitos como rectificadores y optoacopladores para aislar y proteger galvánicamente la señal de cada entrada segura y eficazmente [1].

Cada dispositivo de la sección de entrada está conectado a una terminal de entrada en una franja de terminales de un módulo (numeradas algunas veces en base octal para identificación entre módulos diferentes), por tanto, si algún interruptor (de los sensores de entrada), se encuentra cerrado, 120 VCA aparecerán en la terminal de entrada correspondiente y el acondicionador de señal de entrada convertirá tal tensión en un "1" digital, que posteriormente llegará a la siguiente etapa junto con las demás señales lógicas

ya identificadas. De forma análoga se tiene un "0" digital cuando se encuentran abiertos los interruptores.

Por otra parte, los módulos de salida operan de forma opuesta a los de entrada, teniéndose previamente en cada terminal un acondicionador de salida que es comandado desde la etapa anterior (el procesador o CPU). Dicho módulo se encarga de manejar los actuadores de salida correspondientes a la sección de salida. Debido a que en el módulo de salida pueden estar conectados dispositivos de potencia, generalmente dicha etapa está constituida por dispositivos eléctricamente aislados como pueden ser optotriacs o relevadores tanto de estado sólido como electromecánicos.

Cada dispositivo en cada una de las terminales de la sección de salida gobierna su funcionamiento dependiendo de la existencia de un "1" lógico proveniente del procesador; dicho "1" a la entrada del módulo se encarga de activar (o en el caso de un "0" de desactivar) el actuador correspondiente a esa terminal; es decir, si el actuador se tratase de un relevador electromecánico, el "1" se encargaría de energizar la bobina y por tanto de cerrar el contacto lográndose así energizar la carga de salida [11]. Existen muchos tipos de actuadores, siendo los más comunes los relevadores electromecánicos, relevadores de estado sólido y optotriacs, lográndose con estos un aislamiento entre las tensiones internas del procesador y las tensiones que alimentarán las cargas [59].

1.6 Sección del procesador

El procesador es la parte computarizada del CPU que ejecuta el programa de control. Este manipula los datos almacenados en la memoria del sistema y determina el estado de las salidas basándose en las condiciones dadas por las entradas. Esta sección también se le conoce como CPU.

El procesador es básicamente constituido por dos partes:

- La memoria del sistema.
- El procesador. (en algunos sistemas se le integra la fuente de alimentación).

La memoria del sistema almacena el programa de control del PLC, así como los datos recibidos y mandados desde y hacia la sección de Entrada/Salida generando unos archivos o registros de imagen/salida en donde se mantienen las condiciones actuales del sistema. En el momento que el usuario ingresa el programa estarán automáticamente

almacenados en ubicaciones secuenciales dentro de la memoria del programa de usuario. Esta ubicación secuencial de las instrucciones del programa está regulada por el procesador, sin que sea necesaria la intervención del usuario.

Las condiciones de entrada se almacenan en el archivo imagen de entrada, el cual es otra porción de la memoria del procesador (esta memoria es del tipo de lectura-escritura comúnmente conocida como RAM), es decir a cada terminal del módulo de entrada se le ha asignado una ubicación particular dentro del archivo imagen de entrada. Esta ubicación particular está dedicada únicamente a la tarea de mantener un registro de la última condición de su terminal de entrada.

De la misma manera que las condiciones de entrada, las de salida se almacenan en el archivo imagen de salida, el cual funciona exactamente de la misma forma difiriendo únicamente en lo que concierne a la dirección del flujo de información.

Las tareas del procesador son (ver figura 8): (1) obtener las instrucciones de la memoria de programa de usuario al CPU, (2) obtener la información de Entrada/Salida de los archivos imágen e información numérica de la memoria de información variable y (3) ejecutar las instrucciones. La ejecución de las instrucciones implica (4) tomar decisiones lógicas respecto a los estados adecuados de las salidas ocasionando que estos estados se presenten en el archivo imagen de salida, y (5) calcular los valores de la información variable y almacenar estos valores en memoria de información variable.

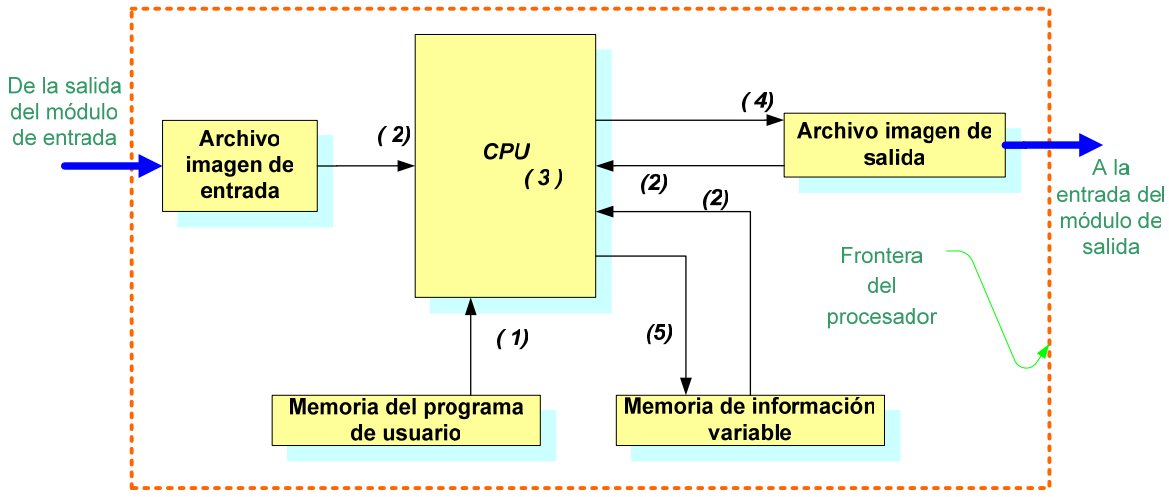


Figura 8. Tareas del procesador.

La fuente de alimentación provee de potencia o energía tanto a la memoria del sistema como al procesador y en algunas ocasiones a módulos de entrada/Salida adyacentes para el correcto funcionamiento de estos.

La tensión disponible en la mayoría de las industrias es de 120 VCA o 240 VCA a 60 Hz. Y dado que la electrónica del PLC funciona con corriente directa, se requiere que dicha fuente suministre el correcto nivel de tensión, además de proveer la correcta regulación y filtrado al PLC en ambientes tanto controlados como ruidosos, así como, proveer de energía en caso de falla eléctrica. Actualmente existen esquemas ampliamente utilizados que van desde fuentes lineales hasta sistemas conmutados.

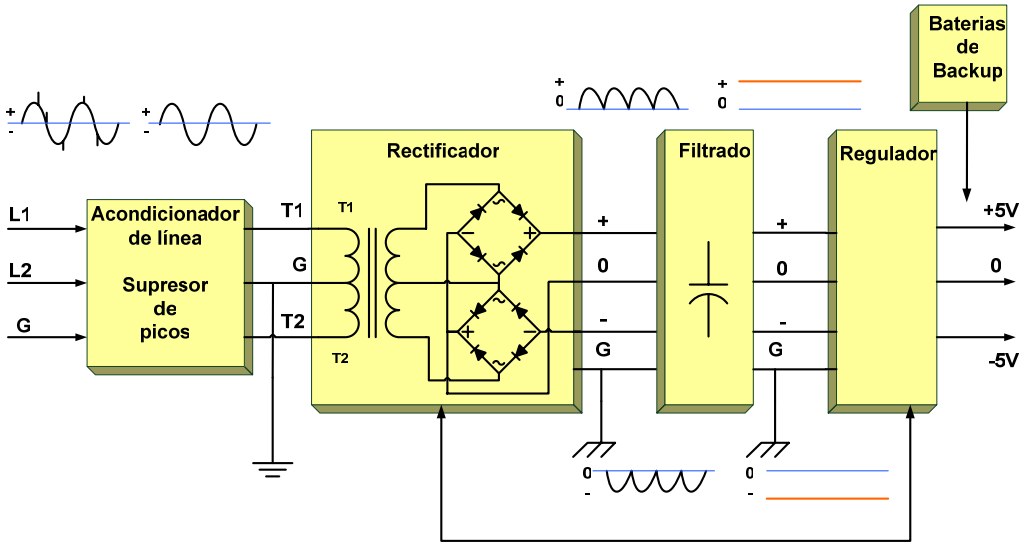


Figura 9. Fuente de alimentación típica.

1.7 Dispositivo de programación

Dispositivo de programación o terminal, también conocido sencillamente como programador. Algunos PLC´s están equipados con un dispositivo de programación dedicado, fabricado por la misma compañía que elabora el PLC, pero en muchas instalaciones, dicho programador es una computadora personal de escritorio o portátil con una tarjeta de interfaz de comunicación instalada en una ranura de expansión. Un cable de comunicación serial se conecta a la tarjeta de interfaz uniéndola con el procesador del PLC. Con software especial instalado en la computadora, las teclas pueden representar instrucciones del programa de usuario, las cuales son convertidas al código apropiado o bien utilizando los diferentes lenguajes de programación, se ingresa el programa de manera

gráfica o escrita; en el monitor se presenta el programa en forma gráfica o de texto pudiéndose observar el desarrollo del programa paso a paso.

En la figura 10 se muestra la forma más convencional de disponer los diferentes módulos que conforman un PLC. Se observa la manera modular de acomodar las diferentes secciones debido a que se pueden conectar un mayor número de módulos entradas/salida sobre un solo riel metálico, así como, otros diferentes tipos de módulos. También, se facilita en gran medida el mantenimiento y ubicación de fallas en un solo sitio; mientras que la colocación de buses y conductores resulta más cómoda para el usuario.

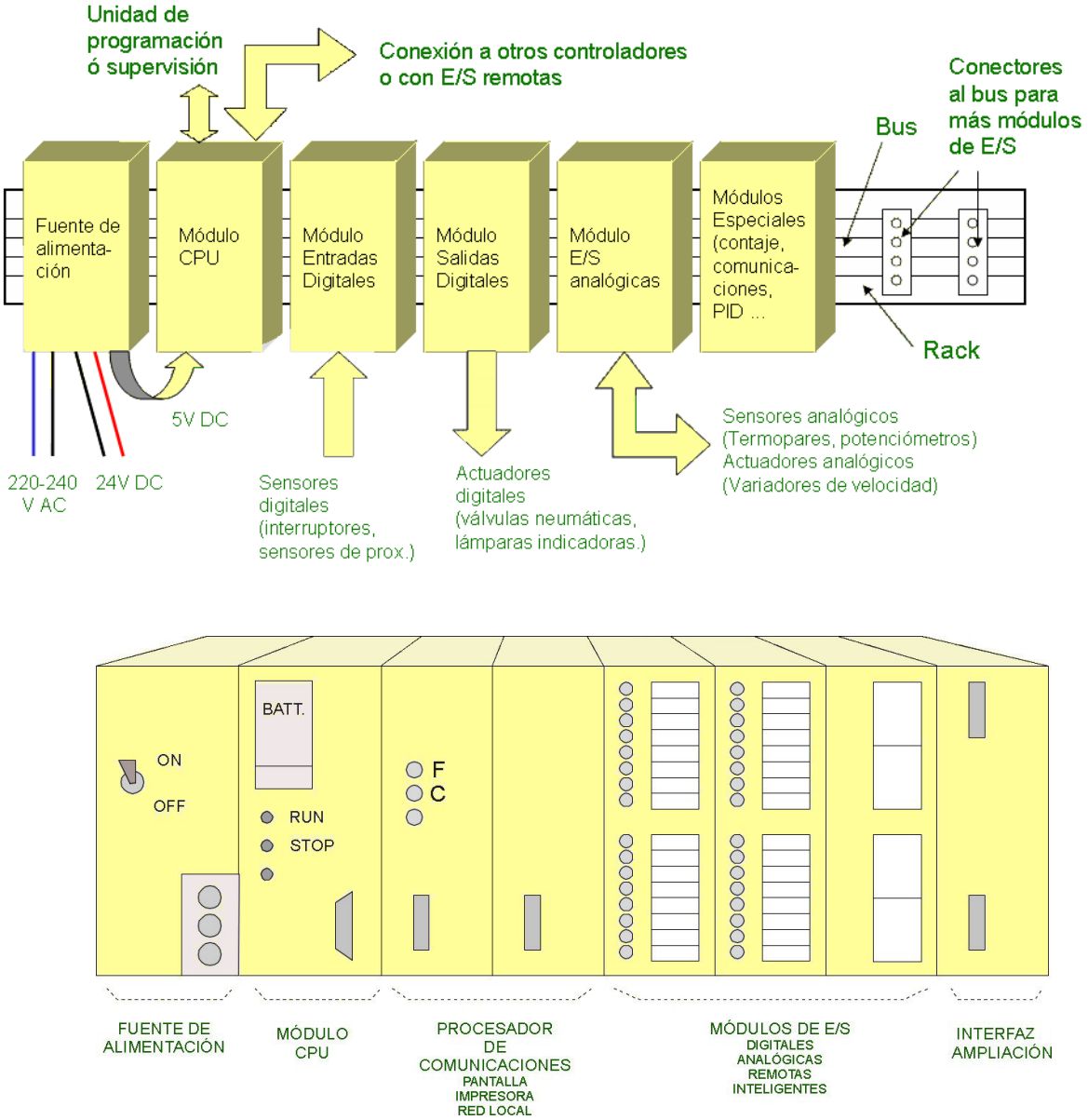


Figura 10. Módulos en un PLC.

2. DISEÑO ELECTRÓNICO DE UN PLC

2.1 Propuesta de diseño

Se propuso diseñar un PLC implementado con componentes de fácil acceso y mantenimiento, además, que el producto final sea lo más parecido a los modelos comerciales en funcionamiento, modo de operación y semejanza física, con el objetivo que el usuario final tenga la posibilidad de realizar ejercicios de automatización y control de sistemas industriales. Para ello, se tomaron como referencia modelos comerciales de pequeños y micro PLC´s.

Para tener una gama más amplia de ejercicios de automatización, el PLC tendrá la característica de poder sensar señales en sus puertos de entrada de corriente directa así como alterna, además de contar con de aislamiento galvánico para seguridad del operador y del dispositivo.

La forma en que se programará el PLC será mediante una computadora personal y el lenguaje del programa de usuario tendrá que asemejarse en gran medida a los lenguajes ya existentes para estos dispositivos, además, tendrá la posibilidad futura de ser un sistema flexible, el cual pueda manejar alternativas en cuanto a la forma de programación.

2.2 Análisis de modelos comerciales

El sistema de desarrollo PLC que se propuso diseñar está, tanto interna como externamente, implementado teniendo como inspiración los sistemas ya existentes y comercialmente disponibles [40] [41] [46] [47]. Como referencia a este diseño, se comparó con un PLC comercial marca Siemens modelo Logo! 230RC [46] (8 entradas y 4 salidas a relevador) aunque también se analizaron otros dispositivos con el fin de buscar un producto que tenga una enorme similitud con los comerciales con el fin de familiarizar al usuario con un sistema no muy distinto a los demás. La siguiente tabla muestra algunos PLC´s de los que se denominan gama baja o micro PLC´s de diferentes marcas y modelos que resultan idóneos para pequeños sistemas industriales y como modelo a seguir de nuestro PLC.

MARCA	MODELO	ENTRADAS	SALIDAS
Allen-Bradley	MicroLogix 1000	16	16
Siemens	S7-221	6	4
Siemens	Logo!	8	4
Klöckner-Moeller	Sucos PS3	16	16
Mikroelektronika	PICPLC16	16	8
Schneider Telemecanique	TSX17-10	12	8
Omron	Sysmac CPM	6	4
Festo	FPC 404	8	8
SCM	MINI14PLC	8	6

Tabla 2. Comparación entre modelos.

Una característica en común de todos los modelos de PLC´s, es que se componen de las mismas etapas, y un punto de donde se comenzó el diseño fue distinguiendo dichas etapas, implementándolas una por una, y realizando una elección de componentes y de circuitos adecuados. La figura 11 muestra las etapas comunes citadas.

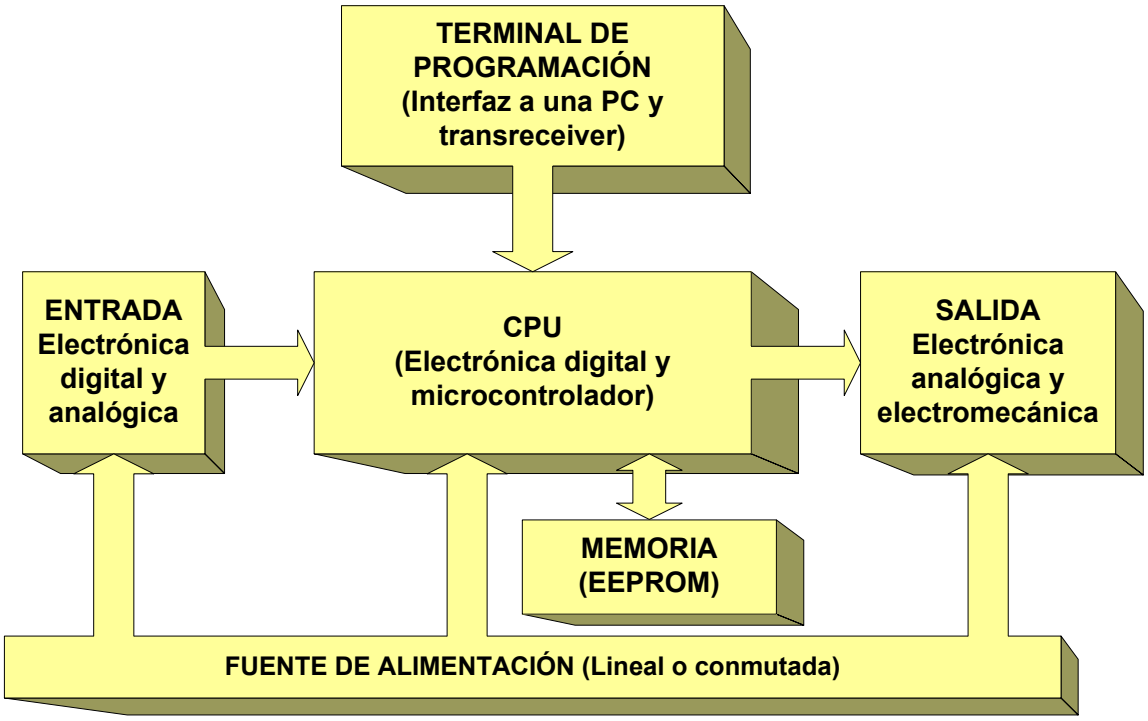


Figura 11. Etapas comunes en los PLC´s y la forma en que se implementan.

2.3 Elección del microcontrolador

Para la implementación del PLC se tuvieron varias opciones en cuanto a la elección del microcontrolador que fungiría como CPU, pero finalmente se eligió un microcontrolador PIC de la gama media, el modelo corresponde al PIC16F877A, el cual es uno de los más versátiles de dicha gama [54] [57].

La razón principal por la que se eligió éste dispositivo sobre otros microcontroladores, es que brinda la posibilidad de una expansión futura del sistema por la riqueza y potencial del dispositivo, además de contar con herramientas y programas para el desarrollo de múltiples aplicaciones (depuradores, grabadores, compiladores en C, etc.), lo que resulta en un ahorro considerable de tiempo y dinero, además por supuesto, de que se contaba con un "know-how" en el desarrollo de sistemas utilizando este dispositivo. A continuación se presentan algunas características generales de este tipo de dispositivos.

Los microcontroladores de la familia PIC siguen una arquitectura Harvard [18] en la cual la unidad central de procesos (refiriéndonos al CPU interno del microcontrolador no al del PLC) se conecta de forma independiente y con buses distintos, con la memoria de instrucciones y con la de datos, dicha arquitectura permite al CPU interno acceder simultáneamente a las dos memorias.

La segmentación ("pipeline") permite al procesador realizar al mismo tiempo la ejecución de una instrucción y la búsqueda del código de la siguiente. De esta forma se permite ejecutar una instrucción en un ciclo de instrucción, salvo las de salto que emplean dos ciclos.

Los microcontroladores PIC son del tipo RISC (Computadora con Juego de Instrucciones Reducido); para la gama media se tienen 35 instrucciones con 14 bits de longitud cada una [57]; esta característica proporciona una gran ventaja en la optimización de la memoria de instrucciones y facilita enormemente la construcción y utilización de ensambladores y compiladores. Dado que las instrucciones son ortogonales, cualquiera de ellas puede manejar los elementos de la arquitectura como origen o como destino. En este tipo de microcontroladores, todos los objetos (puertos de E/S, temporizadores, posiciones de memoria, etc.) están implementados físicamente como registros, es decir, la organización corresponde a un "banco de registros".

Además, para los microcontroladores PIC de la gama media es posible, a diferencia de los de la gama baja, admitir interrupciones, lo cual hace que la comunicación entre periféricos de distintos dispositivos tenga una adecuada sincronización. Por otro lado, cuentan con comparadores de magnitudes analógicas, convertidores A/D, puertos serie y diversos temporizadores [57].

Otras características importantes por la que fue elegido este microcontrolador son por su fácil programación y depuración de errores utilizando un "depurador-en circuito" (en inglés *In-Circuit-Debugger* o ICD) marca Microchip [56], dispositivo que resulta crucial en la actualización del firmware del PLC como hablaremos más adelante. También, fue importante la fácil manera en que una memoria EEPROM 24AA32A [55] puede conectársele utilizando el protocolo I^2C [44] (Inter-Integrated Circuit), para que en ella se coloque el programa del usuario.

Dado que el sistema a desarrollar deberá satisfacer de igual manera las especificaciones de diseño básicas de un PLC comercial, se siguieron los estándares establecidos tanto para las entradas como para las salidas del sistema [2] [28] [31]. Para ello se consultaron manuales, hojas de especificaciones e inclusive se desarmaron PLC´s para tener una mejor perspectiva para el diseño a desarrollar.

2.4 Entorno de programación

El lenguaje empleado para la realización del firmware fue el "C" [5] [6] [7], utilizando un compilador [18] [19] [20], para la obtención del archivo binario el cual se grabará en el microcontrolador PIC. Para ello nos auxiliamos de la compatibilidad con la plataforma de desarrollo y depuración que brinda la compañía Microchip, el Mplab IDE [53].

Mplab IDE es un software que funciona en una computadora personal para desarrollar aplicaciones para los microcontroladores Microchip. Este es llamado IDE (del inglés *Integrated Development Environment*), porque éste provee un ambiente único integrado para desarrollar código y simulación de sus microcontroladores PIC.

Para la grabación y depuración del programa se utilizó una herramienta denominada "depurador-en-circuito" marca Microchip [56]. El Mplab ICD 2 es un depurador (debugger) y un programador serial "en-circuito" (in-circuit serial programmer (ICSP)); este último término se refiere a que es un dispositivo tal que puede programar

microcontroladores aún cuando éstos ya se hallen en su base en el PCB del producto final. El ICD está orientado para ser usado como auxiliar en la depuración y programado de una Unidad MicroControladora (MCU) en un entorno de laboratorio.

El Mplab ICD fue elegido debido a la gran facilidad de su uso, además de poder permitir tanto la grabación del dispositivo como la depuración del programa de una manera muy rápida y eficiente en el mismo entorno de trabajo con el cual se programa, y claro, por su disponibilidad en el laboratorio donde fue desarrollado el proyecto.

2.5 Diseño de la entrada

En la sección de entrada, en particular a lo que se refiere al módulo de ésta [1] [22], se implementó una topología optoaislada que brindará el aislamiento galvánico necesario para evitar posibles daños e interferencias al sistema [4]. Además mediante el uso de este acoplamiento óptico se trabaja igualmente ya sea en señales de alta tensión de corriente directa o alterna; así mismo, el empleo de estos dispositivos utilizando electrónica de estado sólido puede soportar de una manera muy eficiente vibraciones y choques mecánicos en ambientes industriales.

Para el diseño de una entrada se empleó un optoacoplador [49], tal que cuando el led interno se encienda (condición que se traduce en que una entrada se active) el transistor se sature, lo que se traducirá como un "uno" lógico en el emisor de éste, de otra manera el transistor permanecerá en corte (cerca de 0.1 V) o cero lógico en el emisor. Para encontrar la condición de corte-saturación del transistor se ajustó el resistor de emisor dada la siguiente condición [15] [17] (ver figura 12):

$$V_{CC} < R_E \cdot I_C$$

Esto es realmente, llevar la tensión del resistor del emisor a valores un poco inferiores a la polarización del transistor, lo cual se logra con "elevadas" pero seguras corrientes en el led del optoacoplador.

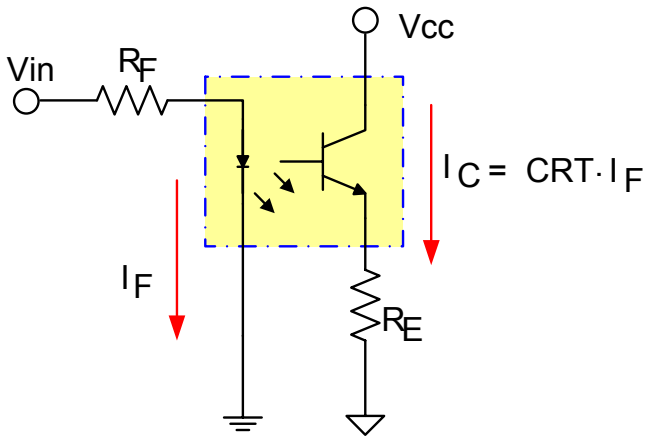


Figura 12. Disposición del optoacoplador.

Aunque consideremos que la corriente I_C dependerá de la corriente de la base, en los optoacopladores no existe una relación entre estas dos corrientes como tal, en lugar de ello está el CTR que es la relación de la corriente de salida con respecto a la corriente del led de entrada (CTR proviene del inglés *Current Transfer Ratio*) [49], para este caso en particular del H11A4 o 4N35 [50] se tiene una CTR mínima de 100%.

Entonces para una corriente de 1mA (valor de corriente propuesto que se encuentra dentro de un intervalo seguro del optoacoplador) proveniente de la entrada de la línea de corriente alterna despreciando la caída de tensión del led se tendra un valor para el resistor del led:

$$R_F = \frac{125V_{RMS}}{1mA_{RMS}} = 125k\Omega$$

Dónde los 125V representan el valor nominal actual de tensión presente en el lugar donde se realizaron las pruebas (pudiéndose éste variar de una localidad a otra).

Realizando un ajuste experimental para obtener 1 mA, esto es, implementando el circuito y modificando valores para el resistor, se obtuvo un valor de resistencia de:

$$R_F = 82k\Omega$$

Y dado que el resistor R_E limita la corriente de saturación se tendrá aproximadamente en el colector del transistor al menos 1 mA (recordemos que el CTR mínimo de este dispositivo es de 100% lo que significa que aun podemos obtener corrientes mayores que 1 mA pero aun seguras para no dañar el transistor) que corresponde a una pequeña corriente lo suficientemente manejable para las siguientes etapas (recordemos que el optoacoplador empleado asemeja su comportamiento al del transistor, y por ello se

realizaron los cálculos haciendo gran similitud a éste). Entonces dado que se polarizara el sistema con 5V, el voltaje V_{CEsat} es de 0.1V. Se tendrá una tensión en el resistor de emisor máxima en el momento de saturación del transistor de:

$$V_E = R_E \cdot I_C = V_{CC} - V_{CEsat} = 5 - 0.1 = 4.9V$$

Y proponiendo un valor de resistencia para el emisor de $1.5k\Omega$ se tiene:

$$I_C = \frac{V_E}{R_E} = \frac{4.9V}{1.5k\Omega} = 3.26mA$$

Cabe hacer notar la importancia que tiene el resistor R_F, dado que dependiendo de su valor, siempre y cuando limite a valores seguros para el optoacoplador en cuanto a corriente se refiere, se puede utilizar cualquier valor de tensión de entrada sea de corriente directa o alterna. En el diseño del PLC se consideró esta situación, por ello se podrán utilizar entradas tanto de corriente directa como de alterna, tan solo accionando un interruptor que modifique el valor de la resistencia para la tensión utilizada. Dicha característica es muy útil dado que algunos sistemas industriales no solo trabajan con las tensiones provenientes de la línea de corriente alterna, sino que utilizan buses de tensión de corriente directa que pueden ser típicamente 5, 12 o 24 Volts [1].

A continuación se muestra el diagrama del circuito de entrada con su correspondiente oscilograma. Se puede observar que los valores de tensión en el colector se acoplan perfectamente con cualquier sistema digital [14] [17].

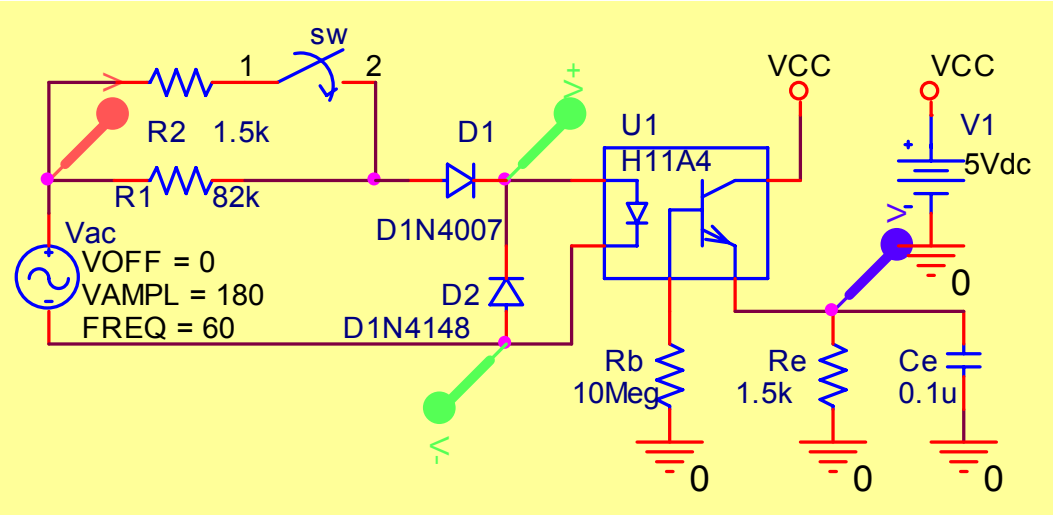


Figura 13. Circuito simulado.

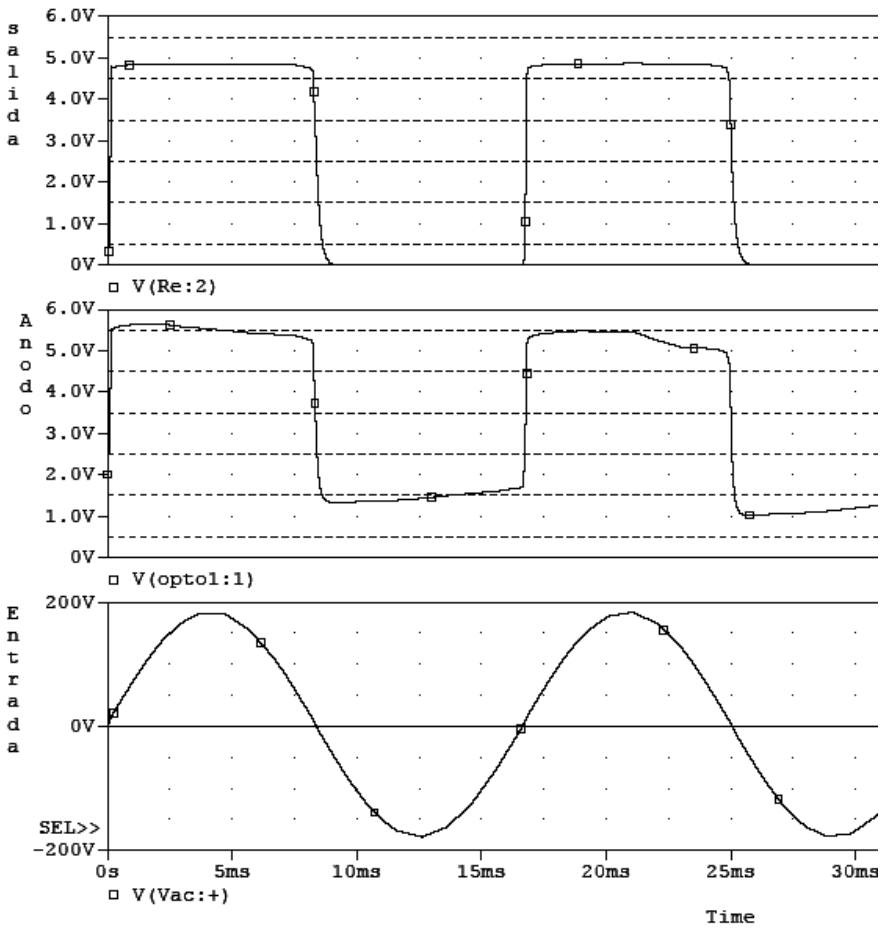


Figura 14. Oscilograma de la simulación.

En la solución propuesta para el acoplamiento en la entrada, se encontraron las tensiones de entrada así como el correcto acoplamiento. Para bajas frecuencias o pulsos de corriente directa, dicho sistema funciona perfectamente, pero al excitar el módulo de entrada con corriente alterna proveniente de la línea, al cerrar o abrir los sensores de entrada (dichos sensores básicamente son interruptores que sensan la presencia o ausencia de eventos), la situación es muy diferente, ya que cada cambio de polaridad negativa en la línea a 60 Hz se entenderá como un cero lógico, situación en la que un sensor típico (interruptor) jamás obedecerá a tal rapidez, además de que aun cerrado el interruptor o sensor de entrada se entendería que éste estuviese abriéndose y cerrándose intermitentemente (esto es si fuese el caso de poder vencer la inercia mecánica), por ello, lo que se debe tomar como un cambio lógico es la existencia o no de corriente alterna, es decir, se sensarán pulsos de corriente alterna.

La forma de abordar dicho problema es mediante la utilización de un circuito multivibrador monoestable o "one-shot" [48] redisparable, tal que cuando se le introduzca a la entrada un tren de pulsos, éste se encargue de generar un solo pulso. En el "one-shot" su salida temporalmente pasa a "uno" lógico cuando el circuito se dispara; luego regresa a "cero" después de cierto tiempo fijo. La manera en que éste se configura es tal que genere un pulso de duración poco mayor a un periodo de la señal de entrada, y dado que dicho periodo se trata de 16 ms (ya que la línea es de 60 Hz) resulta conveniente ajustarlo a unos 20 ms como mínimo. La facultad de redisparo aparece si dentro de cierto intervalo en que se esté generando el pulso de salida aparece o persiste la señal de entrada. En la figura siguiente se muestra ya el circuito implementado y calculado para 48 ms. donde las ecuaciones de diseño son:

$$tw[ns] = K \cdot R[k\Omega] \cdot C[pF]$$

Dónde tw es el ancho del pulso. El tiempo entre dos entradas consecutivas para producir un redisparo es de 0.3·tw.

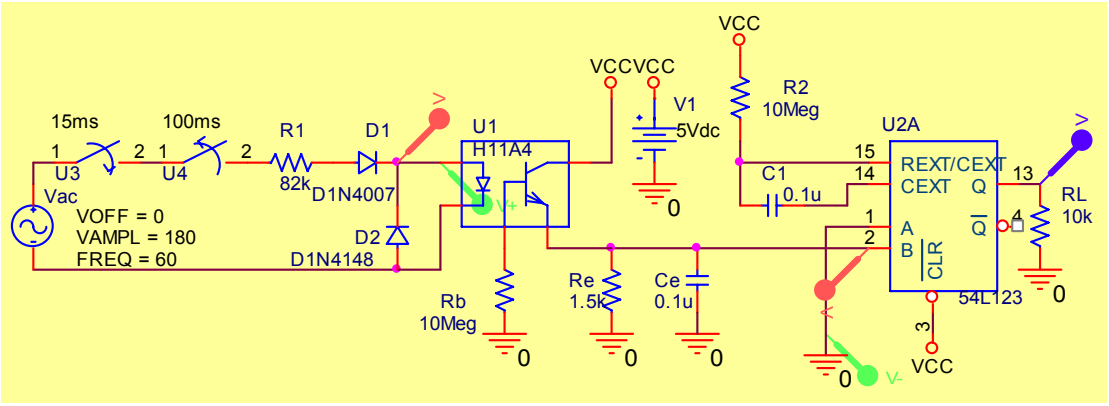


Figura 15. Diagrama eléctrico de la simulación de la etapa de entrada.

Enseguida se ilustra el resultado de la simulación (Figura 16) correspondiente al circuito de la figura 15, donde se ajustó el ancho de pulso para 20ms, y se observa que es suficiente tiempo para lograr el redisparo y obtener un pulso final (86ms) utilizable para la electrónica que sigue. En dicha simulación se pretendió emular un cierre de 85 ms de un interruptor de entrada y se observa precisamente que 20 ms antes que acabase el pulso de salida había acabado la excitación de entrada. También se muestra la señal vista desde el

osciloscopio. El circuito monoestable está implementado de tal manera que reaccione ante los flancos de subida provenientes del emisor del optoacoplador.

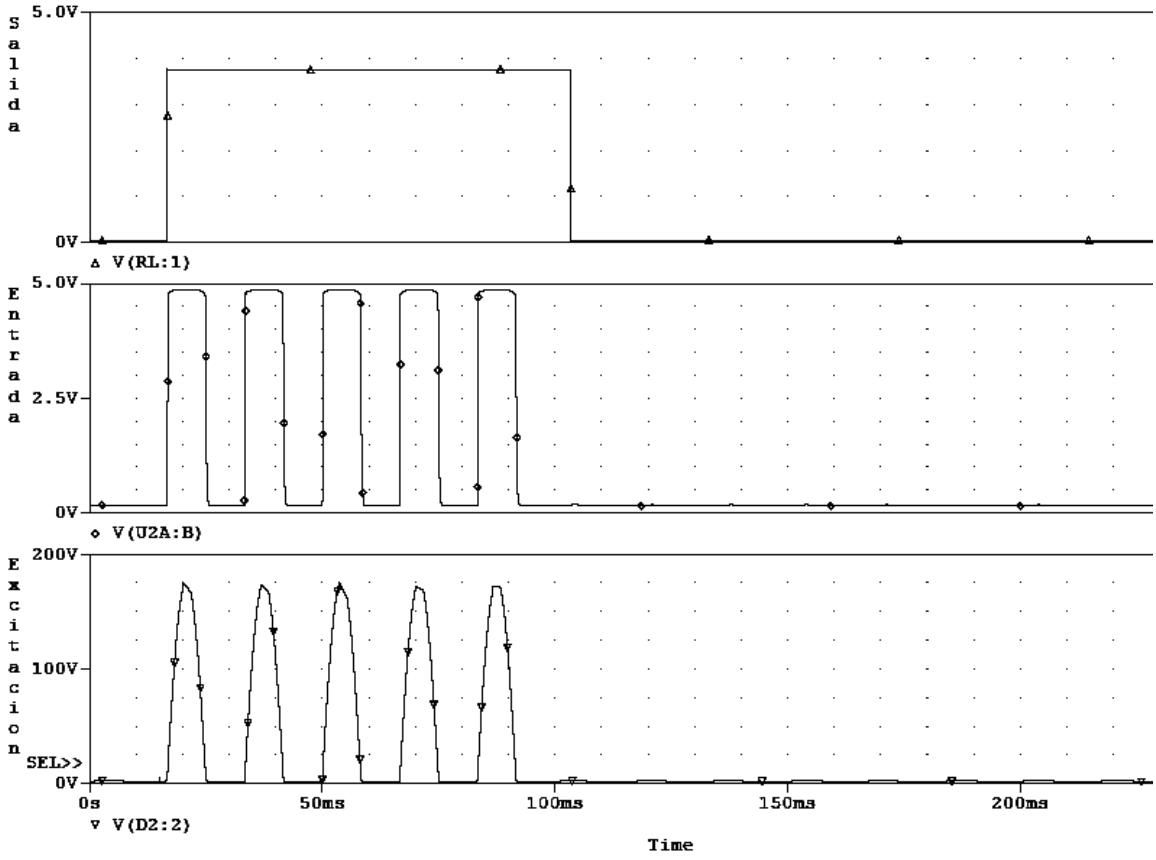


Figura 16. Oscilograma de la simulación.

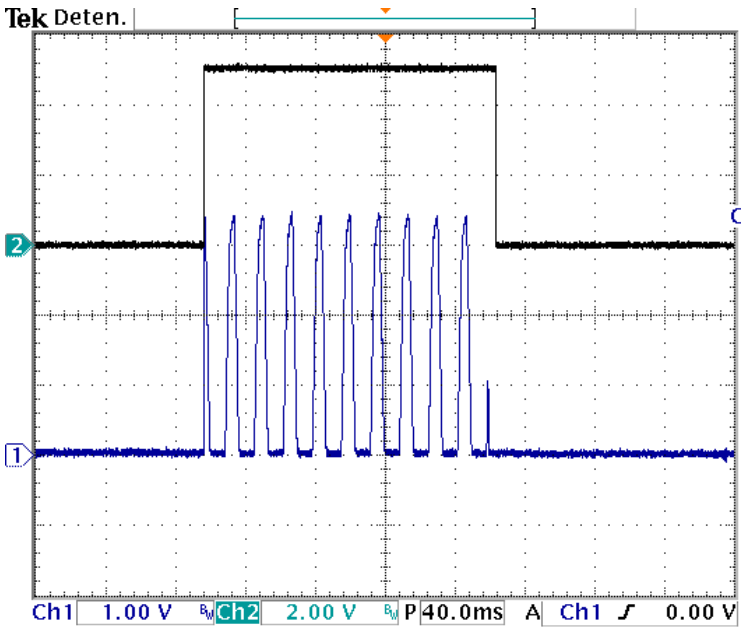


Figura 17. Captura de pantalla del osciloscopio digital.

Se tiene una ventaja adicional utilizando dicho circuito monoestable que resulta en proveer un mecanismo de filtrado antirebotes que pudiesen presentarse en la terminal del módulo de entrada ocasionando falsas entradas encendidas [4]. El circuito de entrada se completó añadiendo una etapa acondicionadora para corriente directa, ya que el multivibrador solo redispara en la presencia de flancos positivos dentro de un intervalo establecido (modo de funcionamiento de corriente alterna), y en condiciones dónde la entrada sea con corriente directa, este no funcionaría. Como acondicionador y fijador de nivel se empleó una compuerta lógica "OR" [60], dónde a su salida se colocó un led que sirve de testigo de la señal de entrada.

En la figura 18 se observa el diagrama de una de las ocho entradas diseñadas en su versión definitiva. Se nota también que se utilizan dos resistores R5 y R10, que con un interruptor se pondrán en paralelo o no según se escoja el modo de corriente directa o alterna como excitación de entrada. También se puede notar tanto la compuerta "OR" como el led indicador de señal.

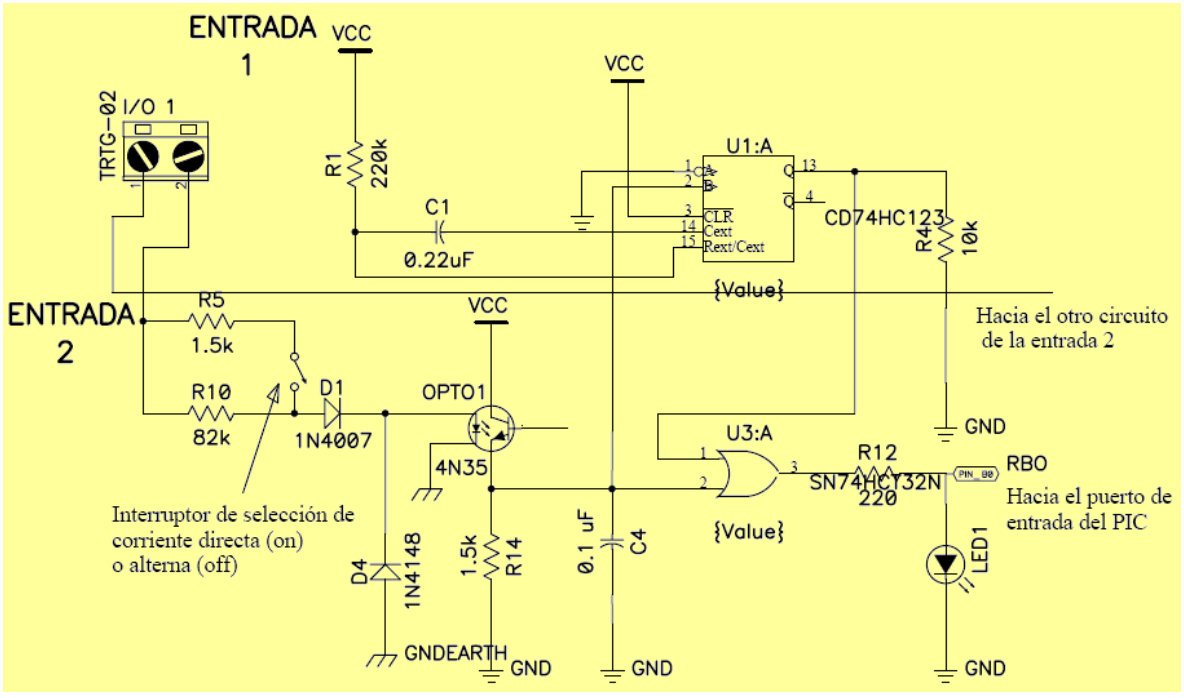


Figura 18. Diagrama de la entrada.

Para la constitución del procesador o CPU del PLC se pensó en la maximización del uso de los recursos de entrada/salida pensando en posibles expansiones así como en adaptaciones o mejoras futuras. El circuito que alberga el CPU está básicamente poniendo a

entera disposición los puertos de entrada/salida para los respectivos módulos, también aloja la interfase de comunicación serial RS-232 [9] [42] [43] [51], la memoria EEPROM I^2C, la fuente de alimentación y la entrada para el depurador en circuito para la programación del firmware.

2.6 Diseño de la salida

El módulo de salida está compuesto por tres secciones (ver figura 19), la primera se encarga de recibir las señales provenientes de los puertos de salida del microcontrolador indicando mediante el encendido de un led la ausencia o presencia de señal para cada una de las salidas, enseguida la segunda etapa es un driver que se encarga de encender o apagar los relevadores [58] [59] los cuales conforman la tercera sección.

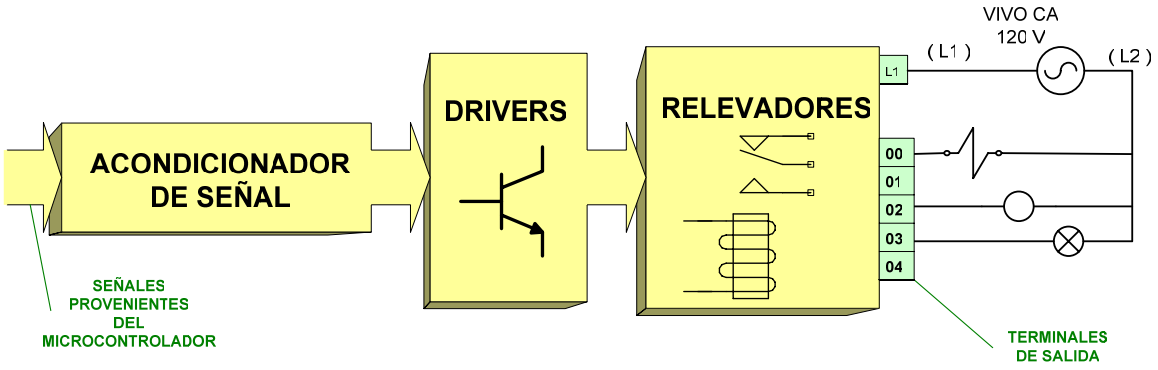


Figura 19. Módulos que integran la etapa de salida.

2.7 La memoria ROM

La idea de mantener la memoria EEPROM I^2C externa al microcontrolador, donde residirá el programa de usuario surge de la posibilidad futura de extender las capacidades de éste sin sacrificar espacio interno de la ROM del PIC, de esta forma se podría actualizar el firmware del CPU sin preocuparse por el tamaño del programa de usuario. Otro motivo es que se pretende a futuro idear una forma alternativa de programar el programa de usuario sin depender directamente del PIC como puede ser extrayendo la memoria, o simplemente diseñar un programador en el mismo sistema teniendo abiertas diferentes opciones en cuanto a la programación de ésta. Cabe señalar que por las capacidades intrínsecas del microcontrolador utilizado, es posible utilizar únicamente los recursos propios, pero debido a la naturaleza del proyecto que puede resultar en constante cambio o evolución se decidió

optar por la elección de un sistema ligeramente distribuido. La manera en que se estableció la comunicación entre la memoria y el CPU fue utilizando las características del puerto MSSP (Master Synchronous Serial Port) en modo I^2C del PIC [44].

2.8 Módulo de programación

En cuanto al módulo de programación, se aprovechó la característica interna del microcontrolador PIC de comunicarse mediante la interfase de comunicación serial (SCI) utilizando el módulo USART (Universal Synchronous Asynchronous Receiver Transmitter) interno, con esto se podrá programar la memoria EEPROM desde una PC a través de un puerto serial; para ello, tras programar el puerto correspondiente, se adecuaron un par de pines de este para realizar tal comunicación de manera asíncrona. Para el acondicionamiento de dicha señal dado que el PIC maneja niveles de tensión de 5V o TTL se implementó un transmisor-receptor (transreceiver) con un circuito max232 para adecuar el canal de transmisión a niveles de ±3 a ±15 V que constituye el estándar dictaminado por la especificación EIA-232-D o CCTT V24/28 para la comunicación RS232 (ahora EIA232 por Electronic Industries Association) [9] [42] [43] [51].

2.9 Ingreso del programa de usuario

La forma en que se ingresa el programa de usuario a la memoria EEPROM 24AA32A (no confundir con el firmware del PIC) es básicamente mediante dos técnicas. La primera que resulta bastante cruda, y se utilizó para realizar pruebas y ajustes, fue extrayendo el 24AA32A del sistema y colocarlo en una unidad programadora de memorias I^2C para que esta lo grabe mediante el software asociado, en donde la forma en que se ingresa el programa es, o bien byte por byte en formato hexadecimal, o ASCII [61]. La otra técnica es mediante el puerto serial de la PC, utilizando el programa contenido en todas las versiones del sistema operativo Windows llamado hyperterminal de Windows, esta última técnica es la que finalmente constituyó el medio de programación que será el definitivo, aunque se proyecta en un futuro crear un programa exclusivamente diseñado para este propósito junto con un analizador léxico sintáctico utilizando el mismo recurso.

Para introducir el programa de usuario se procede a abrir la Hyperterminal y se debe configurar de acuerdo a la figura. Aunque los parámetros pueden cambiar, siempre deben

ser congruentes a los programados en el firmware. Para asegurarse que el PLC esté en modo de programación debe cerciorarse que el pin de programación esté propiamente conectado a la línea de alimentación (Vcc) del sistema, así como todo polarizado correctamente y el puerto serial conectado al PLC.

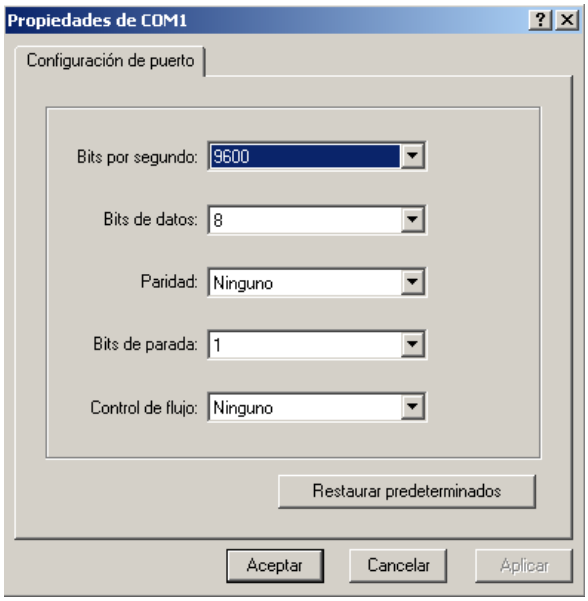


Figura 20. Ventana de diálogo de las propiedades en la hyperterminal.

Después de haber configurado la Hyperterminal se procede a "llamar" al dispositivo, para ello se presiona el botón cuyo ícono es un teléfono, enseguida aparecerá en dicha ventana la leyenda "MODO DE PROGRAMACIÓN AL PLC", la cual indicará que el dispositivo está listo para recibir las instrucciones del usuario de una forma "pseudo ladder texto" o lista de instrucciones. Una vez terminada la programación se pasa al PLC a modo "run" poniendo el pin de programación a tierra. Como aclaración cabe mencionar que tanto la forma de programación, como el lenguaje de programación están en fase de prueba y evaluación (no por ello carece de validez) debido a que más adelante se pretende llevar la técnica de programación a niveles gráficos avanzados usando como base de programación la lista de instrucciones, meta que resulta fuera de los objetivos establecidos de esta tesis. La manera en que se programará se explicará en detalle más adelante.

Figura 21. Ventana de la Hyperterminal.

2.10 Acabado final

Recordemos que la filosofía y giro de esta tesis es el desarrollo de un sistema didáctico, por lo que el diseño e implementación física fue orientado a que dicho sistema además fuese seguro, versátil y con enorme semejanza a los PLC´s comerciales. La figura 22 resume las características logradas (interior de las líneas punteadas en la figura) y muestra la interacción entre las partes que lo constituyen.

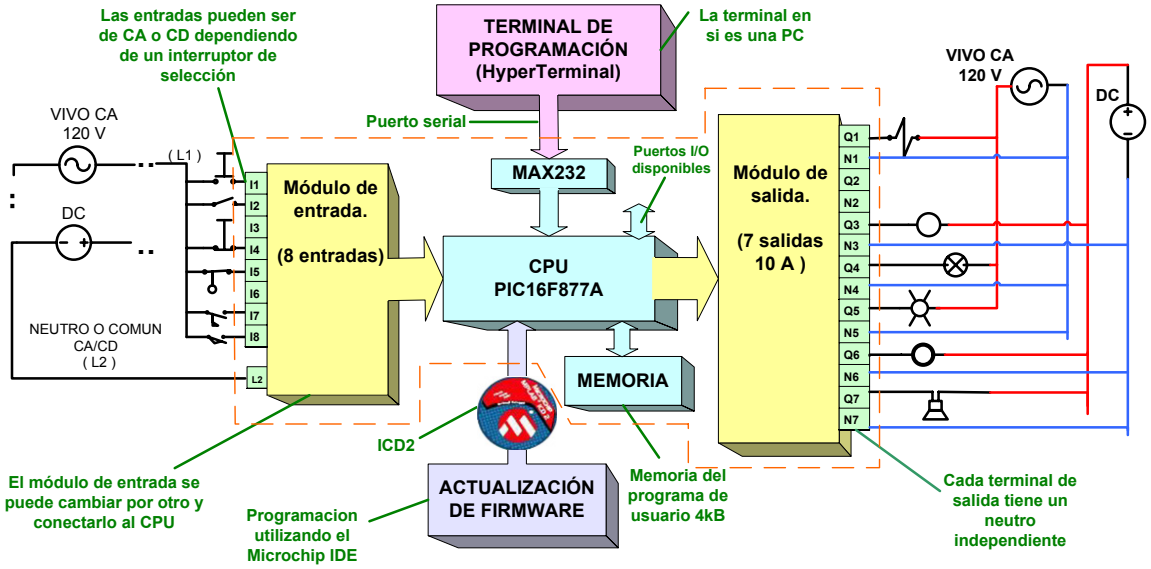


Figura 22. Diagrama de bloques del sistema implementado.

A continuación se muestran algunas fotografías del PLC ya terminado.

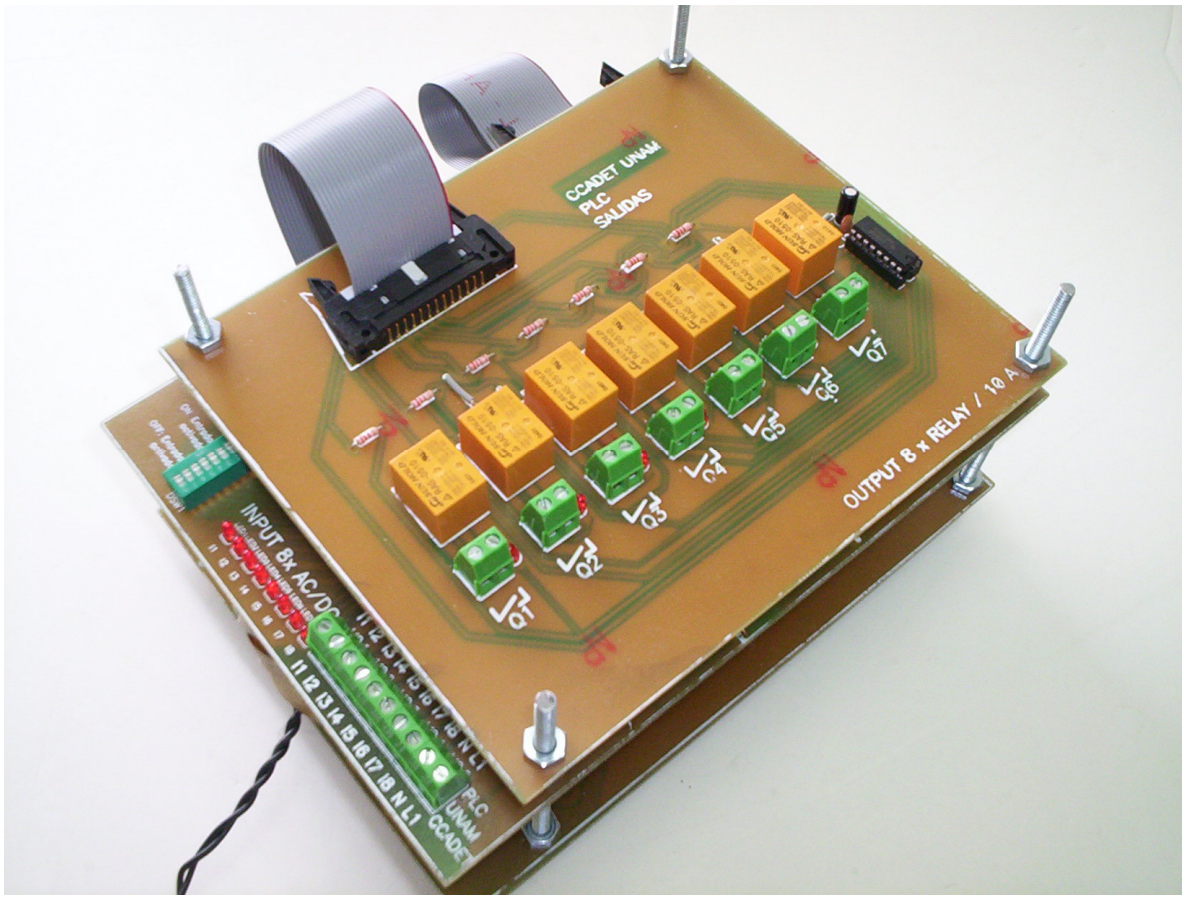


Figura 23. Fotografía del PLC implementado.

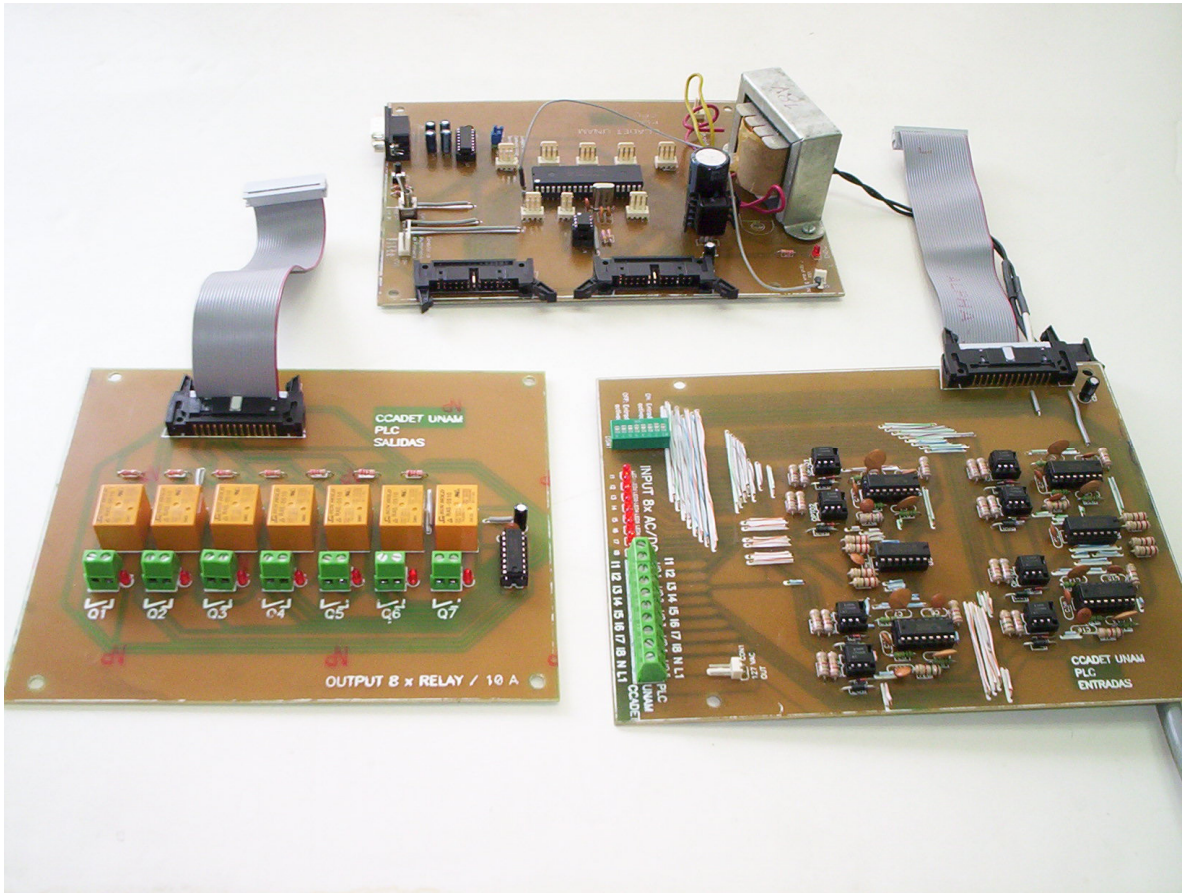


Figura 24. Fotografía del PLC desarmado.

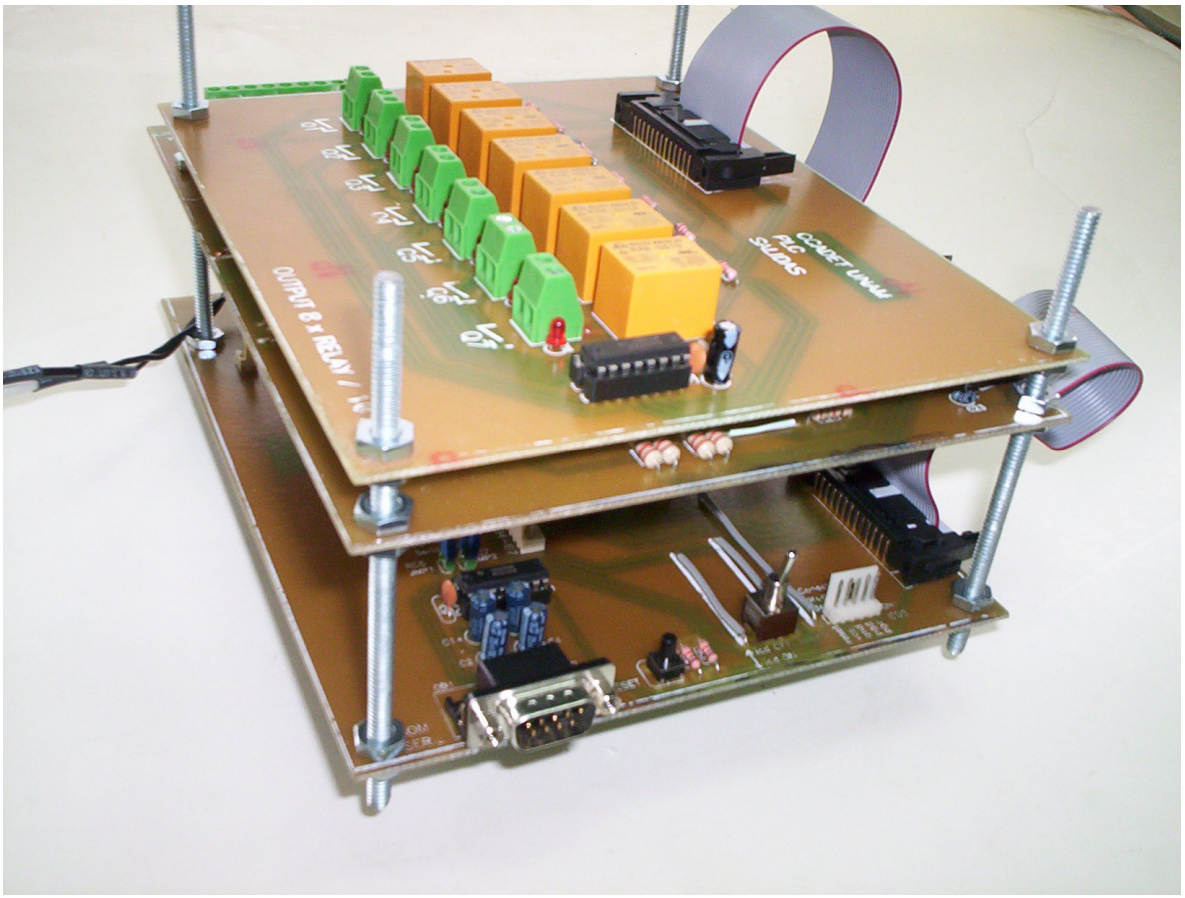


Figura 25. Vista en donde se observa el puerto serial, el reset, el interruptor y la conexión del ICD.

3. LENGUAJES DE PROGRAMACIÓN

3.1 Los estándares en la programación de un PLC

La estandarización de los lenguajes de programación en los PLC ha surgido de la imperiosa necesidad de mejorar las técnicas de programación para los sistemas de control industrial, no sólo para acrecentar la calidad, sino también para incrementar la productividad en general. Este capítulo pretende introducir al lector en las ideas básicas de los estándares IEC 1131, evidenciar que este estándar, a pesar de sus años es de actualidad, ya que en el mundo industrial los cambios resultan mucho más lentos.

El crecimiento acelerado en el uso de controladores lógicos programables para aplicaciones industriales trajo consigo una gran diversidad de lenguajes de programación en todas sus variantes. Diferentes soluciones proponían cada vez mejoras individuales, acrecentando la diversidad. Esta situación resulta un tanto incómoda para las empresas, ya que tienen que invertir recursos económicos cada vez que se realiza un cambio o actualización de la línea de producción, además para los involucrados en sistemas industriales de control, trabajar con estas diferencias resultaba ineficiente. El cada vez más exigente consumidor demanda mejor calidad y productos más económicos, amén de la enorme competencia que se suscita por apoderarse del mercado, lo que requiere de las empresas una política de constante actualización de las líneas de producción.

Para lograrlo, optimizando todo tipo de recursos, es fundamental contar con sistemas productivos ágiles y flexibles. Una empresa tendrá éxito según sea su capacidad de adaptarse rápidamente a los nuevos tiempos. Sin duda, una de las soluciones en el área de ensamblaje y producción es el uso de estándares y soluciones abiertas que permitan la interoperabilidad entre productos de diferentes proveedores además de simplificar la forma de programar y de reutilizar código generado. La adopción del estándar IEC 1131 es sólo un pequeño pero importante eslabón en la cadena de soluciones abiertas, y puede brindar beneficios inmediatos en los sistemas automáticos de toda empresa [2].

El incremento de complejidad en la programación de los PLC´s requiere más que nunca de la estandarización. Bajo la dirección de la IEC (*Internacional Electro-technical Commission*), fue definído el estándar IEC 1131 para la programación de PLC, que alcanzó la categoría de estándar en agosto de 1992. Con la idea de desarrollar el estándar adecuado para una gran diversidad de aplicaciones, se definieron cuatro lenguajes: dos textuales (lista

de instrucciones (LI, o AWL o IL), texto estructurado (TE o ST)), y dos gráficos (Diagrama de Bloques Funcionales (DBF o en inglés FBD) y lenguaje escalera (LE, LADDER, LD o KOP)), y se decidió incluir el gráfico secuencial de funciones (Grafcet) como herramienta para auxiliar en el desarrollo de aplicaciones [2].

El estándar es el resultado del gran esfuerzo realizado por 7 multinacionales a los que se añaden muchos años de experiencia en el campo de la automatización industrial. Incluye 200 páginas de texto aproximadamente, con más de 60 tablas. Son las especificaciones de la sintaxis y semántica de un lenguaje de programación, incluyendo el modelo de software y la estructura del lenguaje [28].

Partes del estándar IEC 1131:

1. Información general: Define la terminología básica y conceptos del estándar.
2. Requisitos de equipo y pruebas: Construcción mecánica y eléctrica y pruebas de verificación.
3. Lenguajes de programación: Estructura del software, lenguajes y ejecución de programas en PLC.
4. Líneas de guía al usuario: Guías para la selección, instalación y mantenimiento de los PLC.
5. Servicios de comunicación: Servicios de comunicación para facilitar la interacción con otros dispositivos basada en servicios de mensajería en sistemas de manufactura.
6. Comunicaciones vía buses de campo: Servicios de comunicación utilizando IEC buses de campo (posteriormente esta parte fue retirada).
7. Programación de control difuso: Software, incluyendo bloques de funciones estándares para manejo de lógica difusa con PLC.
8. Pautas para la implementación de lenguajes para controles programables: aplicación e implementación de pautas para los lenguajes del IEC 61131-3.

La parte tres es la que principalmente se tomó en cuenta para implementar la forma de programación, aunque para desarrollar un sistema completo y coherente con el estándar se tienen que tomar en cuenta todas las partes. Se decidió empezar por la parte que se

refiere a la sintaxis y semántica del lenguaje de programación, ya que el sistema didáctico PLC posteriormente evolucionará principalmente en la forma de programación [2] [28].

Debido a que la forma de programación estará inspirada en la parte tres de este estándar y que el trabajo futuro en lo que respecta a la interfase de programación se cimentará en gran medida a la manera en que se apegue a dicha parte, podría ser conveniente describir ampliamente dicha parte, pero debido a su extensión, resulta más favorable y suficiente tan solo mencionar en qué consiste y sus características básicas.

El IEC 1131-3 trata de lo que el operador del PLC ve en la pantalla o sea que lo a la interfase le concierne entre el programador y el sistema de control; en dicha interfase la forma en que se programa al PLC puede ser muy diversa como se mencionó anteriormente, este estándar es la base real para estandarizar los lenguajes de programación en la automatización industrial, haciendo el trabajo independiente de cualquier compañía. Pero en lo que se refiere al estándar, se sustenta básicamente de dos principales maneras: los elementos comunes y los lenguajes de programación.

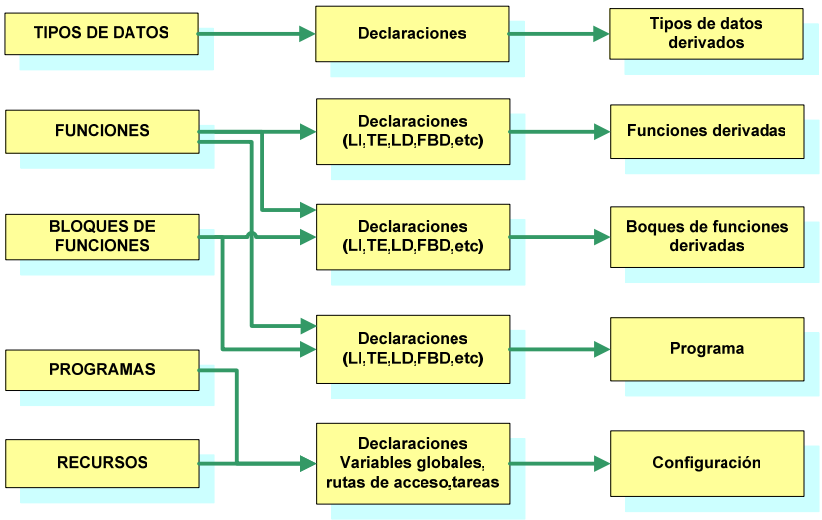


Figura 26. Diagrama de bloques indicando los elementos que intervienen en la IEC 1131-3.

El estándar IEC 1131-3 posteriormente se actualizó en IEC 61131-3 que es una armonización internacional de todos los estándares del IEC y las versiones locales (significando la traducción en diversos idiomas). Además de eso no hay ningún cambio en el contenido.

En cuanto a los elementos comunes, se refiere a los tipos de datos que el sistema maneja, así como las variables, configuración, recursos, tareas, y unidades de organización de programa. Mientras que para los lenguajes de programación, trata de su implementación o ingreso al sistema. En la figura 26 se muestran los elementos que intervienen en dicha parte.

Lenguajes de Programación

Para que un PLC haga diversas funciones o tareas lo primero que procede es introducirle en su memoria un programa, tarea que se realiza ya sea por medio de una PC y un puerto o por medio de una pequeña interfase utilizando algún software asociado al PLC. El programa en si es la serie de instrucciones o comandos en algún lenguaje que se ejecuta secuencialmente y que gobierna el estado de las salidas del sistema a controlar. La selección del lenguaje a utilizar para un desarrollo puede estar basada en la naturaleza de la aplicación, y viene determinada por las preferencias del programador. Sin embargo, aparte de los gustos y preferencias de éste, existen ciertas características de los diferentes lenguajes que deben tenerse en consideración en momentos cruciales, como lo es la selección del lenguaje.

Existen diversas maneras de programar un PLC donde las principales son [2]:

- El Gráfico de Funciones Secuenciales o Grafcet.
- Diagramas de bloques lógicos (FUP).
- Diagramas de contactos en escalera (Ladder o KOP).
- Texto estructurado (Basic, C, Pascal, etc.).
- Lista de instrucciones (AWL).

A continuación describiremos brevemente cada uno de los diferentes lenguajes.

3.2 Gráfico de Funciones Secuenciales

El Gráfico de Funciones Secuenciales, (GFS o en inglés SFC) o Grafcet, es un lenguaje gráfico que ofrece estructura general y coordinación a las secuencias del programa. Ha sido derivado de las técnicas que se usan para describir comportamiento secuencial. Describiendo el comportamiento de un sistema en términos de estados y transiciones, descritos como círculos y arcos, fue originalmente definido en una metodología llamada Petri-net. Petri-net es ahora una técnica bien establecida particularmente en diseño de sistemas computarizados, para formalmente describir el comportamiento de programas que

tienen múltiples estados. El Grafcet fue desarrollado del Petri-net como una forma de metodología industrial.

Una secuencia en El Grafcet es descrita como una serie de pasos mostrados como cajas rectangulares conectadas por líneas verticales como se muestra en la figura 27. Cada paso representa un particular estado del sistema a controlar. Cada flecha conectora posee una barra horizontal representando una transición. Una transición es asociada con una condición la cual, cuando resulta verdadera, causa que el estado anterior a la transición se desactive y que el que le sigue a la transición sea activado. El flujo de control es generalmente hacia abajo, sin embargo, pueden ser usadas ramas para dirigirse a pasos anteriores.

Las principales características del Grafcet son:

- Soporta selecciones alternativas y secuencias paralelas.
- Las etapas o estados implican acciones asociadas.
- Las transiciones gobiernan los cambios de estado
- Las flechas indican la dirección del cambio
- Pueden darse esquemas menos lineales.

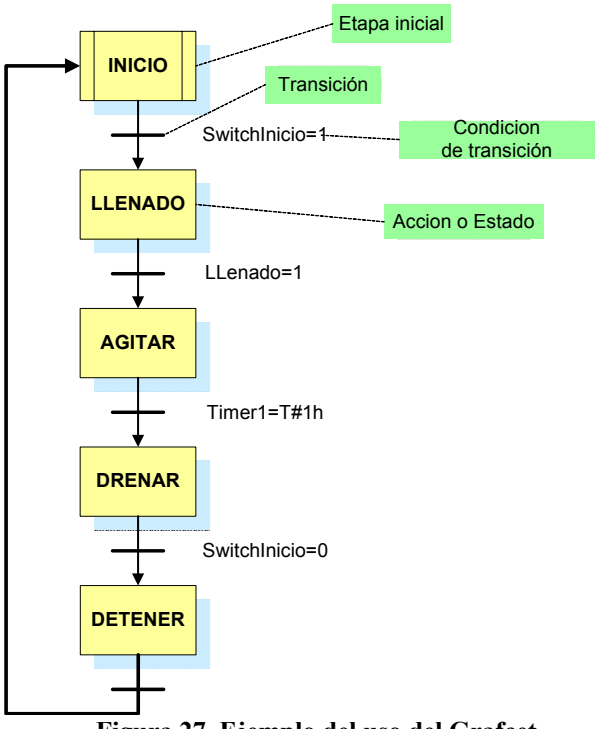


Figura 27. Ejemplo del uso del Grafcet.

3.3 Diagramas de Bloques Funcionales

Los Diagramas de Bloques Funcionales (DBF, en inglés FBD o en alemán FUP) son un lenguaje gráfico usado para construir procedimientos complejos a partir de una biblioteca de funciones, bloques de funciones y programas así como un juego de bloques interconectados.

Pueden ser usados donde el problema involucre un flujo de señales entre bloques de control. Una red de DBF puede considerarse análogo a un diagrama de un circuito eléctrico dónde las conexiones eléctricas describen trayectorias entre componentes. Usos típicos de los DBF incluyen la descripción de lazos de control y lógica Booleana.

Dentro de las convenciones gráficas que aplican los DBF establecen que una función es descrita como un bloque rectangular con entradas en la izquierda y salidas a su derecha. El tipo de la función, así como los nombres de las entradas y salidas, van escritas dentro del bloque mientras que el nombre del bloque arriba de éste.

Para prevenir el encimamiento de DBF, el nombramiento tanto de las entradas como de las salidas se realiza utilizando los "números de pins" en cada bloque, de tal manera que no se puedan repetir y en el momento de listar se visualicen correctamente.

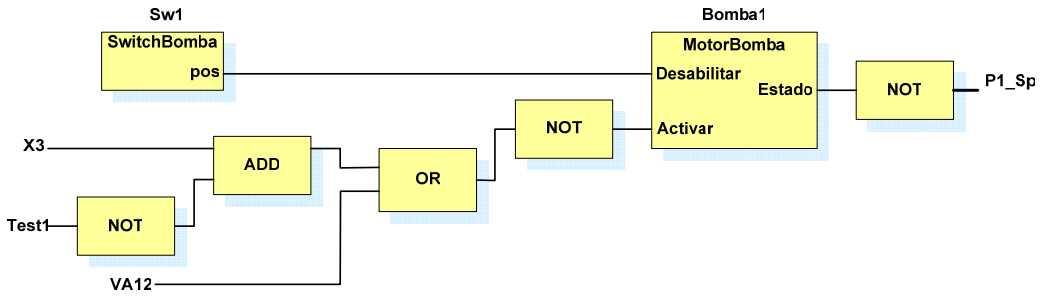


Figura 28. Ejemplo de uso del DBF.

3.4 Diagrama de Contactos

El Diagrama de Contactos (LADDER, LD o KOP) es un excelente lenguaje gráfico para lógicas discretas. También tiene la habilidad de incluir instrucciones de bloques funcionales dentro de una línea. Los contactos y bobinas del diagrama de contactos pueden ser usados en el lenguaje diagrama de bloques funcionales para implementar control discreto o funciones.

El lenguaje de diagrama de contactos es considerado el más comúnmente usado, está basado en el diseño lógico con relevadores. Un diagrama de escalera o ladder siempre

tiene a su izquierda la barra de alimentación que provee de energía a los dispositivos, la trayectoria de la corriente eléctrica se distribuye horizontalmente hacia la derecha en donde se halla el neutro. Dichos dispositivos pueden ser contactos, interruptores, bobinas, etc. donde se tienen símbolos asociados a estos, como se muestra en la figura 29.

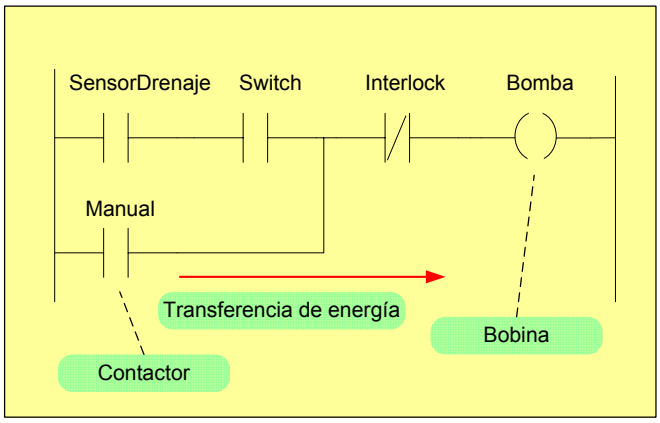


Figura 29. Ejemplo del diagrama de contactos.

3.5 Texto Estructurado

El Texto Estructurado (ST o TE) es un lenguaje similar al Pascal, Basic o C (desde bajo hasta alto nivel) que se emplea para procedimientos complejos o cálculos que no pueden ser implementados fácilmente utilizando lenguajes gráficos. En este tipo de programación se pueden expresar funciones de diversos comportamientos, bloques de funciones y programas. El estándar define un intervalo de operadores tanto para operadores aritméticos como booleanos como se muestra en la tabla siguiente

Operador	Descripcion
(…)	Expresion en parentesis
Funcion()	Lista de parametros de una función
**	Exponenciación
- NOT	Negación Complemento booleano Signo negativo
* / MOD	Multiplicación Division Modulo
+ -	Adicion Subtracción
<,>,>=,<=	Operadores de comparación
= <>	Igualdad Desigualdad
AND, &	And booleano
XOR	Or exclusiva booleana
Or	Or booleana

Tabla 3. Algunos operadores del texto estructurado.

Un ejemplo de este tipo es el siguiente, donde la forma de programación es muy similar a cualquier programa realizado con algún compilador, nótese que es de fácil comprensión debido a que muchas funciones especiales ya están implementadas.

```
Define# I001,005 = salida1;
Abierta:= 1;
Cerrada:=0;
Area := PI * R * R;
V:= K ** (-W * T);
Volts:= Amps * Ohms;
Start := OilPress AND Stream AND Pump;
Status:= (Valvula1 = Abierta) XOR (Valvula1 = Cerrada);
entrada1:= ConvAD((I001,005),5ms,10,3);
if(entrada1 >= 15)  salida1 == Abierta;
else { (Valvula1 & salida1 )== Cerrada; }
```

Figura 30. Ejemplo de texto estructurado.

3.6 Lista de Instrucciones

La lista de instrucciones (AWL, IL o LI) es un lenguaje de bajo nivel, similar al código máquina o ensamblador, y puede ser usado para expresar el comportamiento de funciones, bloques de funciones y transiciones en diagramas de funciones secuenciales. Es útil para pequeñas aplicaciones que requieran de una rápida ejecución para optimizar el tiempo de proceso. La estructura básica de la lista de instrucciones es muy simple y fácil de aprender.

El lenguaje de LI consiste en una serie de instrucciones donde cada instrucción está sobre una nueva línea. Una instrucción consiste en un operador seguido por uno o más operandos que pueden estar separados por comas. Cada instrucción puede cambiar o usar algún valor almacenado en un solo registro. El estándar refiere a este registro como el resultado de esta instrucción. El resultado puede ser sobrescrito con un nuevo valor, modificado o almacenado en alguna variable. Algunas veces a este registro, donde se halla dicho resultado, se le denomina acumulador [24].

El estándar establece que el comportamiento general o semántica de una instrucción LI para la mayoría de los operandos puede ser descrita por la siguiente expresión [2]:

Nuevoresultado := ResultadoActual Operador Operando

ResultadoActual. Representa el valor actual contenido en el registro de resultado antes de que la instrucción esté siendo ejecutada; es el valor que queda de la instrucción previa.

OPERADOR	MODIFICADOR	OPERANDO	COMENTARIO
LD	N (negacion)	cualquiera	Cargar operando en el registro de resultado
ST	N	cualquiera	Guardar el registro de resultado en el operando
S		booleana	Pone al operando verdadero (set)
R		booleana	Pone al operando falso (reset)
AND	N, (	cualquiera	Operación And booleana
OR	N, (	cualquiera	Operación Or booleana
ADD	(	cualquiera	Operación suma
MUL	(	cualquiera	Operación multiplicación
GT	(	cualquiera	Comparacion >
NOT		cualquiera	Negación lógica (complemento a uno)
JMP	C(condicion), N	etiqueta	Salto hacia la etiqueta
=		cualquiera	asignar
T		cualquiera	Habilitar temporizador
)		cualquiera	Cierre de paréntesis, ejecutar ultimo operador

Tabla 4. Ejemplo de operadores de la lista de instrucciones.

Nuevoresultado. Es el nuevo valor para ser cargado en el registro de resultado, creado de ejecutar la instrucción que está siendo realizada. Es el valor producido por el *Operador* actuando sobre los valores de *ResultadoActual* y el *Operando.* El *Nuevoresultado* se convierte en *ResultadoActual* para la siguiente instrucción.

Muchos fabricantes de PLC´s ofrecen sistemas que soportan la programación de lista de instrucciones prefiriendo éste sobre el texto estructurado. Para un diseñador de PLC´s, una de las principales ventajas de LI sobre el TE es que es mucho más fácil de implementar y desarrollar un PLC, que pueda interpretar tanto programación gráfica como listas de instrucciones directamente. Asimismo, con algunos sistemas es posible descargar programas en LI al PLC sin compilar el programa dada su simpleza además de su estandarización. Contrastando al TE que generalmente tiene que ser compilado al ensamblador nativo del microprocesador del PLC.

Las listas de instrucciones son consideradas algunas veces como el lenguaje en el cual todos los otros lenguajes IEC (*Internacional Electro-technical Commission*) pueden ser traducidos, aunque con limitantes, destinadas a pequeños PLC´s.

4. FUNCIONAMIENTO DE UN PLC

Una vez cargado el PLC con el programa de usuario o secuencia de control, el funcionamiento interno es muy similar entre distintas marcas comerciales. Como referencia consideremos el esquema de programación gráfica ladder o diagrama de contactos (LADDER, LD o KOP).

Para explicar el funcionamiento de la lógica con que funciona un programa encontramos conveniente agrupar las instrucciones en pasos de instrucciones, a menudo solo denominados pasos o escalones (un grupo de instrucciones que afecta a una sola terminal de salida, con base en los estados de ciertas terminales de entrada y de salida). La palabra paso o escalón se deriva del hecho de que estos grupos de instrucciones recuerdan los escalones de una escalera cuando el programa de usuario está representado en un formato lógico en escalera [3] [25].

En la siguiente figura se muestran dos diagramas, el de la izquierda es la representación lógica en relevador para PLC´s, mientras que el de la derecha es el que corresponde al programa en ladder. En este caso se pueden observar tres escalones en cada una de las representaciones gráficas, cada escalón o rama es un paso de instrucción, dicho paso es una porción del programa completo de usuario almacenado en la sección de memoria del programa de usuario.

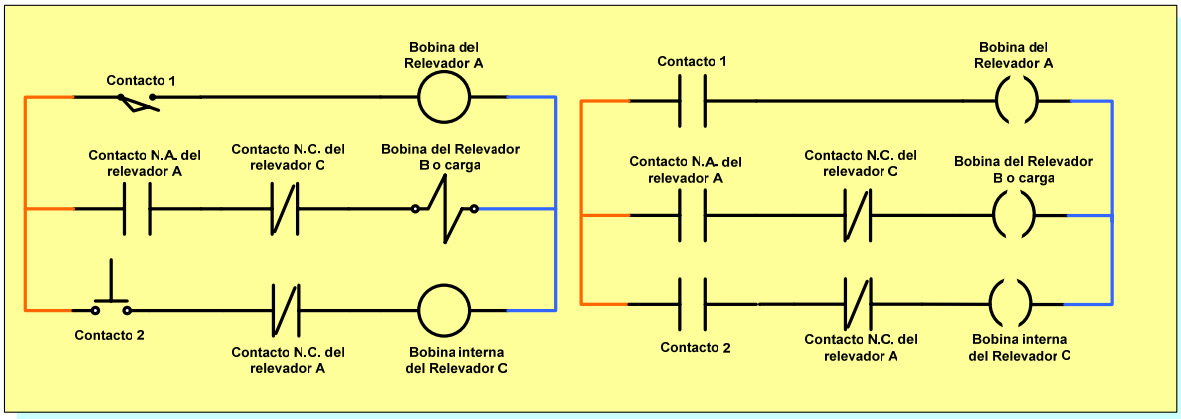


Figura 31. Comparación entre representación lógica y Ladder.

4.1 Continuidad y discontinuidad lógica

También se puede notar que en el primer diagrama se encuentran dos interruptores o contactos (Contacto 1, Contacto 2), dichos contactos corresponden a dispositivos físicos de entrada ubicados en las terminales del módulo de entrada, así también se tiene un contacto asociado a otra entrada física (contacto N.C. del relevador interno C). Además se tienen presentes otros contactos de relevadores que son internos del PLC (Contactos N.A y N.C. del Relevador A), donde a cada uno de estos se le asocia su estado de encendido o apagado con el cambio de una salida física del PLC (bobina del relevador A). También se tiene una salida interna o marca (bobina del relevador C) que obedece según su correspondiente contacto (contacto N.C. del relevador interno C).

Por otra parte, el diagrama de la derecha muestra el equivalente ladder del programa, donde se puede observar que los símbolos se simplificaron notablemente, y esto es debido a los efectos que el PLC genera al manejar las entradas y las salidas como relevadores "virtuales". Por ejemplo, el símbolo asociado al contacto 1 representa una instrucción llamada "examinar-encendido", "examinar-si-cerrado" o "examinar-N.A." (o en inglés "examine-on", "examine-if-on","examine-if-closed"), en tal instrucción si la terminal de entrada asociada a dicha instrucción (contacto 1) se le aplica una tensión (es decir, se cierra el contacto, ya que recordemos que un extremo de éste estará conectado a algún suministro de voltaje); entonces, el paso de instrucción global considera la instrucción como si produjera continuidad lógica, al igual que un contacto eléctrico cerrado. El símbolo que le sigue a la derecha semejante a un paréntesis que representa una salida (en ocasiones la simbología no cambia) cambia su estado lógico a verdadero ocasionando que dicha salida se prenda (dicho encendido se puede asociar a la energización de alguna bobina externa o alguna marca interna). Sin embargo, si a dicha entrada no se le aplica ninguna tensión, el paso de instrucción global considera la instrucción como si produjera discontinuidad lógica, al igual que un contacto eléctrico abierto. Entonces se puede considerar que el PLC logra una equivalencia entre un paso de instrucción en un programa ladder con un circuito lógico relevador [26].

También existe la instrucción llamada "examinar-apagado", "examinar-si-abierto" o "examinar-N.C." (o en inglés "examine-off", "examine-if-off","examine-if-open"), que

opera de manera exactamente opuesta a la de “examinar-encendido” , como es el contacto N.C. del relevador A.

En la figura 32, se muestra la utilidad de un PLC debido a una reducción notable de relevadores, ya que como se observa se pueden implementar tanto funciones como relevadores internos, aprovechando la característica de continuidad lógica en un paso de instrucción programado en la memoria interna, lo cual se traduce en un ahorro significativo tanto en tiempo como en el uso de dispositivos externos [30] [31].

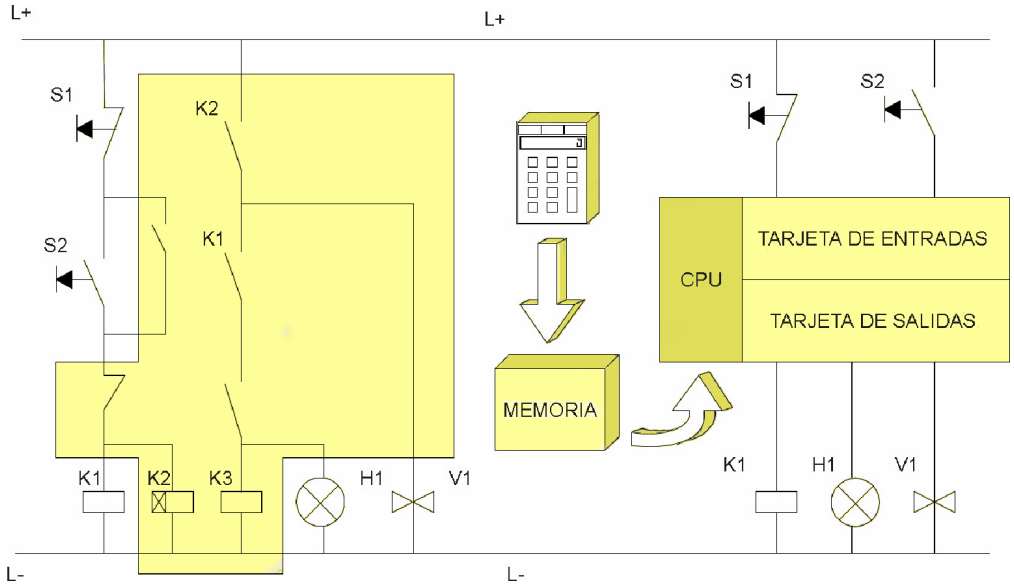


Figura 32. Uso de relevadores internos en un PLC.

4.2 Ejecución del paso de instrucción

Para ejecutar el programa de usuario, el procesador o CPU maneja un paso de instrucción a la vez. La figura 33 muestra los eventos involucrados en la ejecución de un paso de instrucción en las operaciones AND y OR. Dicho diagrama de flujo de igual manera se puede utilizar para otras operaciones booleanas tan solo cambiando o negando banderas [3].

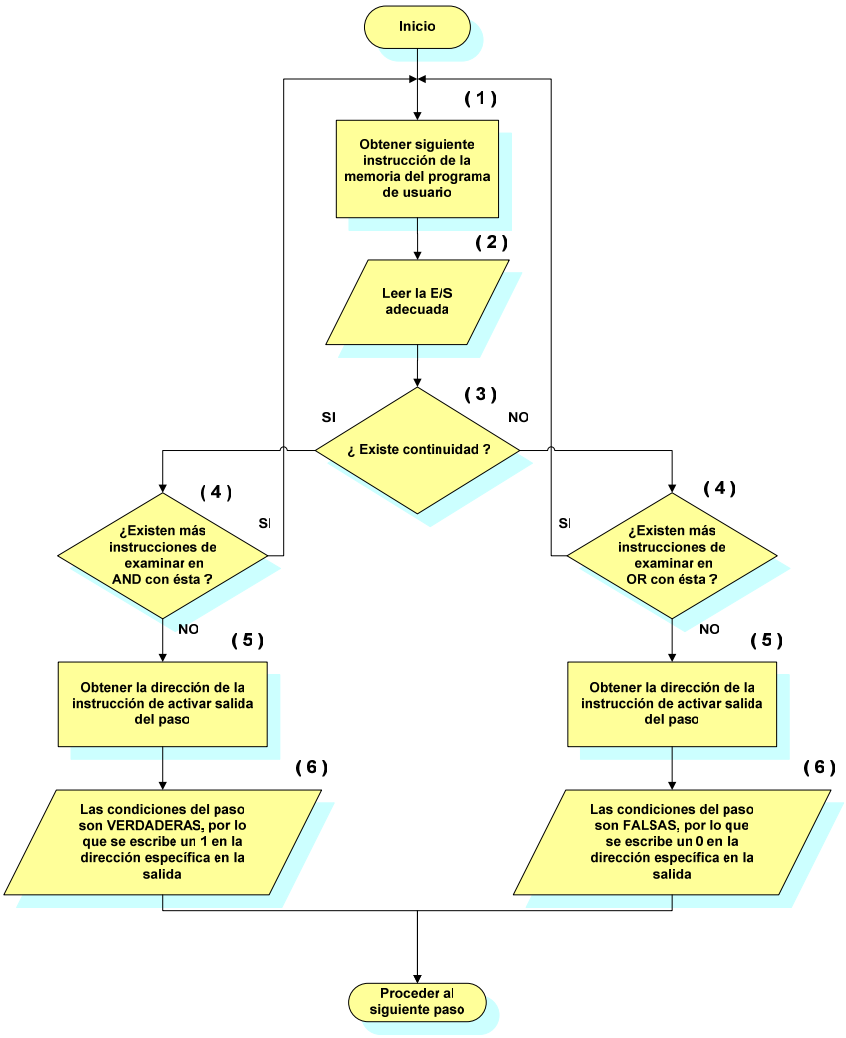


Figura 33. Operaciones AND y OR

(1) El CPU, obtiene la siguiente instrucción secuencial de la memoria de programa de usuario.

(2) La instrucción que el CPU acaba de obtener está obligada a ser una instrucción de tipo “examinar” (citada anteriormente) que se le asocia tanto a una función booleana AND como a una OR (que adelante explicaremos más en detalle) debido a que se requiere que cada paso comience con una instrucción de este tipo en dónde se pretende inspeccionar alguna entrada o marca. El CPU obtiene la información requerida del archivo imagen de salida o entrada con el fin de evaluar la instrucción.

(3) El CPU lleva a cabo una prueba interna mediante la combinación de la instrucción del punto (1) con la información de E/S del punto (2). Este examen determina si la instrucción produce continuidad o discontinuidad lógica.

(4) El CPU busca en la memoria de programa de usuario para ver si la siguiente instrucción es otra instrucción del tipo "examinar" o una instrucción de activar salida. Si es una instrucción del tipo "examinar", el CPU observa si está lógicamente en AND o en OR con la instrucción previa. Si está lógicamente en AND (en serie en la representación lógica en ladder), entonces ambas instrucciones deben producir continuidad para que el paso mantenga continuidad hasta aquí. Si la siguiente instrucción está lógicamente en OR con la anterior (aparecen rutas paralelas en la representación ladder), entonces será suficiente que cualquier instrucción produzca continuidad con el fin de que el paso mantenga continuidad hasta aquí.

Puede suceder que el CPU pueda tomar su decisión inmediata respecto a la verdad o falsedad de las condiciones del paso o escalón. Una decisión inmediata se expresa por alguna de las dos bifurcaciones "no" que salen de los cuadros de decisión señalados con (4) en el diagrama de flujo. Estas bifurcaciones llevan al paso (5), la cual trae la dirección de la última dirección de ese paso, la instrucción de activar salida.

Por otro lado, puede suceder que el CPU no pueda tomar su decisión de verdadero o falso de forma inmediata, pero debe obtener la siguiente instrucción del tipo "examinar" para una validación posterior de la continuidad. La situación se expresa por alguna de las dos bifurcaciones "si" que salen de los cuadros de decisión señalados con (4). Estas bifurcaciones regresan al paso (1) en el diagrama de flujo; las cuales ocasionan que el CPU repita los pasos del (1) al (4).

(5) Eventualmente, el CPU progresará a través del paso o escalón hasta el punto en que pueda decidir si las condiciones generales del paso son verdaderas o falsas. Luego se obtiene la instrucción de activar salida de la memoria del programa de usuario, de manera que se pueda conocer a qué dirección afectar.

(6) El CPU ahora conoce la condición del paso y la correcta dirección de salida, de forma que envía la señal digital adecuada al archivo de imagen de salida, el cual a su vez lo transfiere a la terminal de salida asociada.

4.3 El Scan

Cuando el procesador ha terminado de ejecutar un paso de instrucción, se desplaza hacia la siguiente ubicación secuencial en la memoria del programa de usuario, obtiene la siguiente instrucción (la primera instrucción del siguiente escalón), y repite los pasos del (1) al (6). Continúa de esta forma hasta que cada instrucción haya sido ejecutada. En ese punto el programa de usuario habrá sido ejecutado completamente una vez.

El método en que el PLC evalúa el programa de usuario en ladder determina el orden en el cual las instrucciones programadas serán ejecutadas. Entonces la forma en que se barre y evalúa el programa tiene un profundo efecto sobre la manera en que éste y las máquinas o procesos asociados operan. El ciclo operativo del PLC siempre que se deje en el modo ejecutar o "run", y que se procese y ejecute libremente el programa de usuario una y otra vez se denomina "scan" [3] [22].

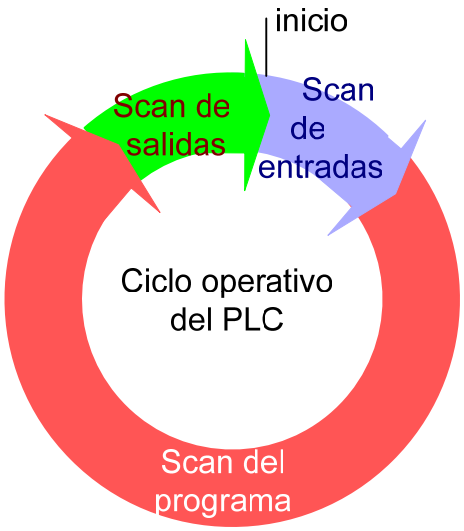


Figura 34. Ciclo operativo o Scan.

Comenzando en la parte superior del círculo que representa el ciclo operativo de barrido, la primera instrucción es el barrido de entrada. Durante éste, el estado actual de cada terminal de entrada se almacena en el archivo de imagen de entrada, actualizándolo. Al igual que todas las operaciones del PLC, el barrido de entrada es muy rápido.

Después del barrido de entrada, el procesador ingresa a su ejecución del programa de usuario, algunas veces denominado scan o barrido de programa. La ejecución implica comenzar en el primer paso o escalón de instrucción del programa, realizando la secuencia de ejecución de seis pasos descrita anteriormente. Después pasará al segundo paso realizando su secuencia de ejecución y así hasta el último paso de instrucción del programa. El tiempo de ejecución del programa dependerá de la longitud del programa, la complejidad de los escalones de instrucción y de la rapidez del CPU.

A través de la ejecución del programa de usuario, el procesador continuamente mantiene su archivo de imagen de salida actualizado. Sin embargo, las terminales de entrada en sí mismas no se mantienen continuamente actualizadas. En lugar de ello el archivo imagen de salida completa se transfiere a las terminales de salida durante el barrido o scan de salida posterior a la ejecución del programa.

5. DISEÑO DEL JUEGO DE INSTRUCCIONES

EL PLC pone a disposición del usuario diferentes elementos en el modo de programación, esto es, un juego o set de instrucciones con los cuales se va a realizar el programa de usuario. Dicho juego siempre está presente y está constituido mediante las diferentes formas de programación (dichos lenguajes de programación en sus diversas variaciones o representaciones), a grandes rasgos de manera gráfica o en modo texto.

Aunque existen equivalencias entre funciones u operaciones en los juegos de instrucciones de diferentes lenguajes, existe lo que se denomina "reversibilidad" [2] [3], dicho término se refiere a la capacidad del programa intérprete con el cual se programa al PLC para convertirlo en lenguaje lista de instrucciones (LI o AWL); Por ejemplo, a los programas de aplicación escritos en lenguaje Ladder y viceversa. Nosotros aprovecharemos tal característica pese a la advertencia de que no siempre es posible obtener una equivalencia exacta en algunos casos.

Para entender el concepto de reversibilidad es importante conocer la relación que existe entre el "circuito", conjunto de instrucciones de programación en lenguaje de contactos que constituyen una expresión lógica, y la "sentencia", el conjunto de instrucciones de programación en lista de instrucciones que realizan la misma función.

Para tener un mejor entendimiento acerca de las equivalencias entre lenguajes de programación, es recomendable entender el juego de instrucciones desde un entorno gráfico ya que con esto se tendrá una mejor visualización en el momento de programar sea en el modo gráfico por supuesto o en algún modo de texto. Por ello, resulta imperativo conocer la representación de las instrucciones junto con las capacidades del PLC en cuanto a ejecución de determinados procedimientos [26].

El juego de instrucciones gráficas representa:

- las entradas/salidas del PLC (botones pulsadores, sensores, relevadores, indicadores de funcionamiento, interruptores, etc.)
- las funciones avanzadas (temporizadores, contadores...),
- las operaciones matemáticas y lógicas (suma, división, y, o exclusiva...),
- los operadores de comparación y otras operaciones numéricas (A < B, A = B, desplazamiento, circular...)
- las variables internas o marcas del autómata (bits, palabras, etc.).

Las funciones o instrucciones básicas son en su mayoría elementos lógicos sencillos del álgebra de Boole donde en algunas ocasiones se agrupan formando funciones más complejas, aunque también existen funciones de mayor nivel de complejidad.

Las instrucciones las podemos agrupar en operaciones lógicas, timers y contadores, mismas que se describen a continuación.

5.1 Operaciones lógicas

Las operaciones lógicas con bits operan con dos dígitos, 1 y 0. Estos dos dígitos constituyen la base de un sistema numérico denominado sistema binario. Los dos dígitos 1 y 0 se denominan dígitos binarios o bits. En el ámbito de los contactos y bobinas, un 1 significa activado (”conduce”) y un 0 significa desactivado (”no conductor”) como lo habíamos establecido anteriormente con la continuidad y discontinuidad lógica [3] [26].

Las operaciones lógicas con bits interpretan los estados de señal 1 y 0, y los combinan de acuerdo con la lógica de Boole. Estas combinaciones producen un 1 ó un 0 como resultado y se denominan “resultado lógico”. Las instrucciones booleanas pueden ser comparadas con los elementos de lenguaje de contactos [33].

5.2 Temporizadores o Timers

Estos dispositivos cambian el estado de alguna determinada salida dependiendo del tiempo preprogramado en ellos, y tal funcionamiento comienza a partir del cambio de alguna entrada. La operación de este tipo de sistemas suele describirse con la ayuda de un diagrama de tiempos, que no es más que una gráfica de el estado de entradas y salidas a lo largo del tiempo. Suelen inicializarse indicando el tiempo y la forma en que va a funcionar, también se le indican cuáles serán los parámetros, entradas y salidas que serán involucrados [34].

5.3 Contadores

Todos los PLC´s incorporan funciones que reemplazan a la aplicación de contadores en el sistema de control. Además del obvio uso de estos contadores para contar, por ejemplo, piezas o ciclos de trabajo, la combinación de varios de ellos, quizás con el uso de algunas funciones de tipo aritmético, permite reemplazar contadores a leva y realizar funciones que

de otra forma resultarían complicadas. Todos los contadores tienen una entrada de pulsos a contar, una entrada de reset, que cuando es activada lleva al contador a su estado inicial y una salida que se activa cuando la cuenta llega a su valor final. El tipo más común de contador es el ascendente, en el que el estado inicial es: cuenta cero con la salida desactivada. Al ir recibiendo pulsos en la entrada de conteo, la cuenta aumenta siempre manteniendo la salida desactivada, hasta el momento en que la cuenta llega al valor establecido en el programa y el contador deja de contar. Podemos encontrarnos también con contadores descendentes, en los que se programa un valor inicial distinto de cero y la salida se activa cuando luego de realizar un conteo descendente la cuenta llega a cero [34].

En la figura 35 se muestra en forma de tabla un juego de instrucciones básico, en el cual se muestran cuatro diferentes maneras de programar una instrucción. Cada columna corresponde a un lenguaje diferente de programación o a la representación de alguna instrucción. Para la segunda columna se muestra la forma de diagramas lógicos con relevadores, en la tercera columna se muestra la programación en escalera o ladder (se puede observar la similitud con la lógica de relevadores), en la cuarta columna se tienen los bloques de instrucciones lógicos (donde se hace ver la sencillez de los bloques) y en la quinta y última se tiene la lista de instrucciones. En esta última forma de programar se utilizaron los operandos del juego de instrucciones que corresponden al PLC Siemens donde más adelante comentaremos en detalle

.

5.4 Lista de Instrucciones como opción elegida

La forma en que se decidió abordar las instrucciones del programa de usuario, fue utilizando el método de texto, más específicamente la lista de instrucciones, ya que se asemeja al ensamblador en cuanto a la estructura, sintaxis y comportamiento, y resulta un pilar para realizar más adelante una programación avanzada, como se explicó anteriormente. Además, el desarrollo del juego de instrucciones de este PLC está inspirado en el actual estándar IEC 1131-3, por lo que en sí resultará en un lenguaje válido y a su vez cumplirá con el propósito de ser una herramienta didáctica y funcional [2].

El juego de instrucciones creadas fue elaborado para poder desarrollar funciones cada vez más complejas con elementos sencillos. Dicho compendio de instrucciones para tener congruencia con el estándar y seguir conservando la temática de sistema didáctico

PLC, fue elaborado teniendo como modelo el AWL o lenguaje de lista de instrucciones del PLC Simatic S7/S5 de Siemens [47]. Por ello, podemos encontrar equivalencia entre operandos en este juego de instrucciones, fomentando así, tanto un mejor entendimiento de la programación en lista de instrucciones como una buena referencia comercial con que comparar.

5.5 Instrucciones creadas

A continuación, la figura 35 enlista las instrucciones válidas para el PLC creado. Como referencia y para una mejor comprensión se muestra también su equivalente en lenguaje gráfico. Recordemos que la forma en que se introduce el programa de usuario es por medio de listas de texto donde cada operador y operando se introduce línea por línea. Para facilitar la lectura se adicionará también la instrucción con formato. Profundizaremos en la forma de programar al PLC más adelante.

Convenciones:

El símbolo #: equivale a un número decimal que puede ser del cero al siete que corresponden a las entradas o a las letras de D a la K que pertenecen a las 8 marcas internas.

El símbolo %: equivale a un número decimal que puede ser del cero al seis que corresponden a las salidas o a las letras de D a la K que pertenecen a las 8 marcas internas.

El símbolo < indica el número de decenas que van del cero al 9.

El símbolo > indica el número de unidades que van del cero al 9.

El símbolo * indica un caracter alfanumérico cualquiera y no importa de cual se trate ya que su existencia es por legibilidad de formato, normalmente se repite el caracter de la entrada.

El parámetro de salida indica que función resultará afectada por las operaciones anteriormente efectuadas.

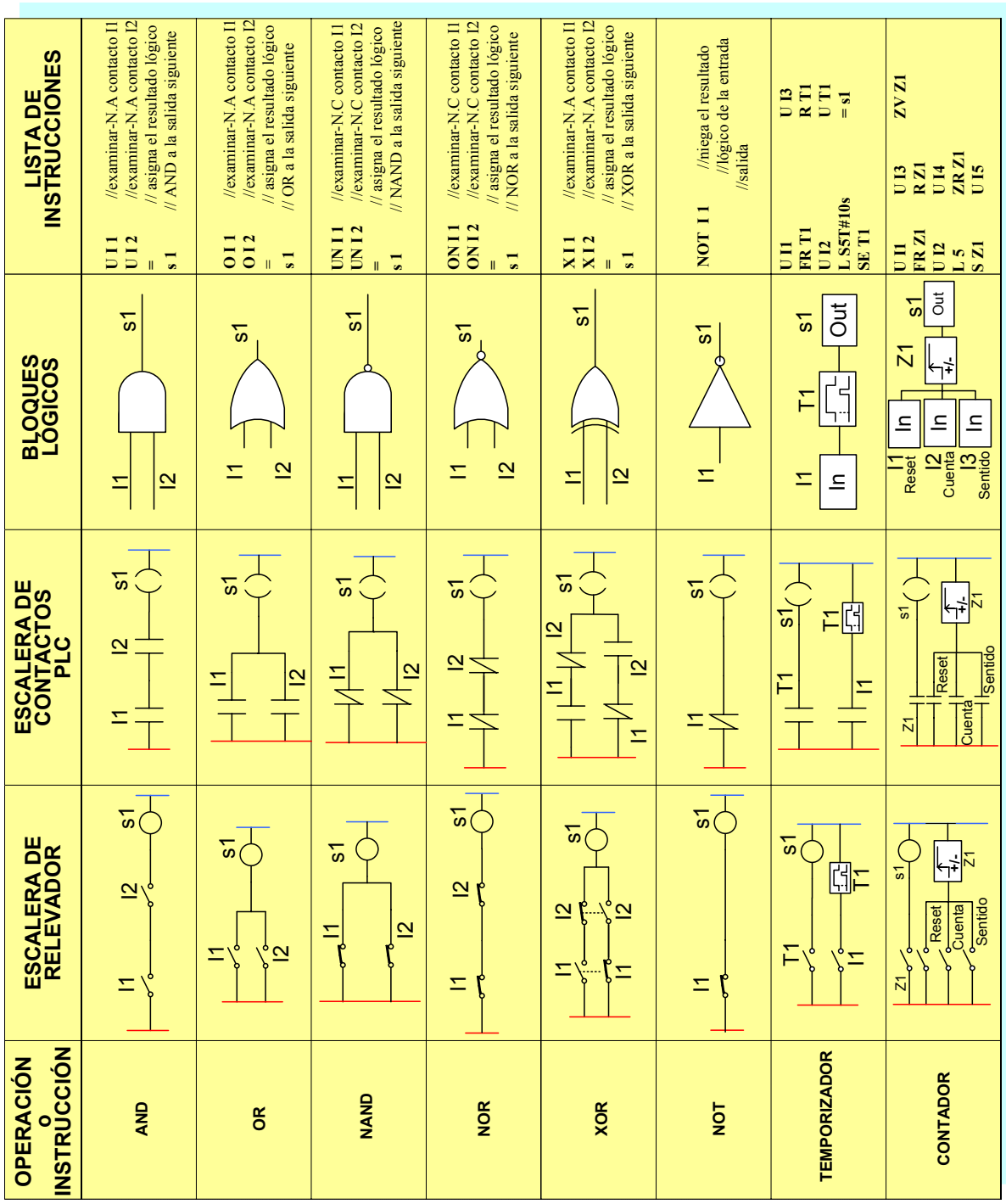
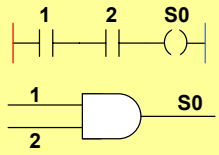


Figura 35. Comparación de operandos entre diferentes lenguajes de programación.

5.6 Función AND

Instrucción: “A” (equivalente Siemens: “U”)

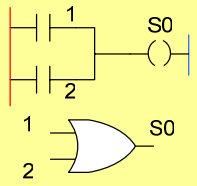
Realiza la función lógica AND entre el acumulador o resultado lógico RLO y el operando especificado, almacenando el resultado en RLO (bandera AND). Se puede operar con el negado si se le antecede con el operador NOT (N).

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
N (negación)	A	#	S% Siguientes A,O,o,a
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS**
A 1 A 2 S 0 =	A1 A2 S0 =		U I 1 U I 2 = A 0

5.7 Función OR

Instrucción: “O” (equivalente Siemens: “O”)

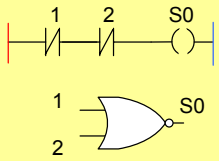
Realiza la función lógica OR entre el acumulador o resultado lógico RLO y el operando especificado, almacenando el resultado en RLO (bandera OR). Se puede operar con el negado si se le antecede con el operador NOT (N).

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
N (negación)	O	#	S% Siguientes A,O,o,a
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS**
O 1 O 2 S 0 =	O1 O2 S0 =		O I 1 O I 2 = A 0

5.8 Función NOR

Instrucción: “o” (equivalente Siemens: inexistente, solo composición.)

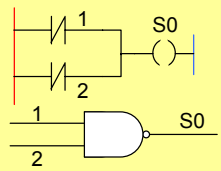
Realiza la función lógica NOR entre el acumulador o resultado lógico RLO y el operando especificado, almacenando el resultado en RLO (bandera NOR). Se puede operar con la negación si se le antecede con el operador NOT (N). Es una función derivada de la OR que simplifica la sintaxis del programa de usuario.

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
N (negación)	o	#	S% Siguientes A,O,o,a
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS**
o 1 o 2 S 0 =	o1 o2 S0 =		ON I 1 ON I 2 = A 0

5.9 Función NAND

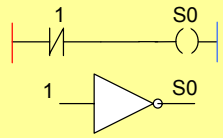
Instrucción: “a” (equivalente Siemens: inexistente, solo composición).

Realiza la función lógica NAND entre el acumulador o resultado lógico RLO y el operando especificado, almacenando el resultado en RLO (bandera NAND). Se puede operar con el negado si se le antecede con el operador NOT (N). Es una función derivada de la AND que simplifica la sintaxis del programa de usuario. También se utiliza para realizar la comprobación antes mencionada de “examinar-apagado”, esto es, para comprobar si una entrada o marca esta en “1” o en “0”.

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
N (negación)	a	#	S% Siguientes A,O,o,a
CÓDIGO PROPIO (ejemplo)	CÓDIGO CON FORMATO	EQUIVALENCIA GRÁFICA	EQUIVALENCIA SIEMENS
a 1 a 2 S 0 =	a1 a2 S0 =		UN I 1 UN I 2 = A 0

5.10 Función NOT

Instrucción: “N” (equivalente Siemens: “NOT” (invierte el RLO actual al no activarse el bit OR)). Realiza la inversión en el acumulador o resultado lógico RLO tanto de operadores como de parámetros de salida activando la bandera NOT invirtiendo los operandos.

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
	N	S% A#, O#, o#, a#	Bandera Not en A#, O#, o#, a#
CÓDIGO PROPIO (ejemplo)	CÓDIGO CON FORMATO	EQUIVALENCIA GRÁFICA	EQUIVALENCIA SIEMENS
N A 1 S 0 =	NA 1 S0 =		NOT A 0

5.11 Función fin de rama o bloque

Instrucción: "=" (equivalente Siemens: "BEB", "BE" fin de bloque en un módulo).

Dicha instrucción indica el fin de una rama en ladder o de un bloque de instrucciones. Restablece todas las banderas a su estado inicial. No tiene ningún modificador y opera tras toda la secuencia de instrucciones que le precede.

5.12 Función OUT salida

Instrucción "S". (Equivalente Siemens: "=" asignación.)

Extrae el valor del contenido del acumulador RLO ("1-activado o 0 desactivado") asignándolo al operando correspondiente (la salida o marca).

MODIFICADOR	OPERADOR	OPERANDO	PARÁMETRO DE SALIDA
N	S	%	Continuidad/ discontinuidad lógica %
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS**
A 1 S 0 =	A 1 S0 =		U I1 = A 0

Funciones temporizador o timer

5.13 Temporizador tipo U (Flip-flop con retardo a la salida temporizado).

En un PLC, la función de memoria se elabora mediante un flip-flop SR, el cuál dispone de dos entradas: una para la instrucción de activación SET (S) y otra para la instrucción de desactivación. Con un "1" en la entrada del SET, la salida Q alcanza un valor de "1", mientras que con un "1" en la entrada del RESET (R) la salida Q se pone en "0".

Después de haber transcurrido un tiempo parametrizable Ta (de 0 a 99 segundos) que constituye la condición inicial de arranque, un impulso de entrada del disparador S (cualquier entrada o marca que cambia de "0 a 1") activará la salida Q (cualquier salida o marca). A través de la entrada de reset R (cualquier entrada o marca que cambia de "0 a

1"), el tiempo para el retardo a la conexión y la salida se ponen a cero (Reset tiene preferencia respecto al SET). Si la entrada S pasa del estado "0" al "1"durante el transcurso del tiempo aplicado Ta y aún no ha arrancado el flip-flop, la salida Q se activará en el instante en que nuevamente la entrada S pasa del estado "0" al "1", siempre y cuando la entrada R permanezca en "0". La figura 36 muestra el diagrama de tiempos asociado y enseguida se muestra una tabla descriptiva de esta función. El ejemplo ilustra un flip-flop temporizado ajustado a 25 segundos, que se activa con la entrada número uno y el reset con la entrada número dos, quedando como salida la número cero.

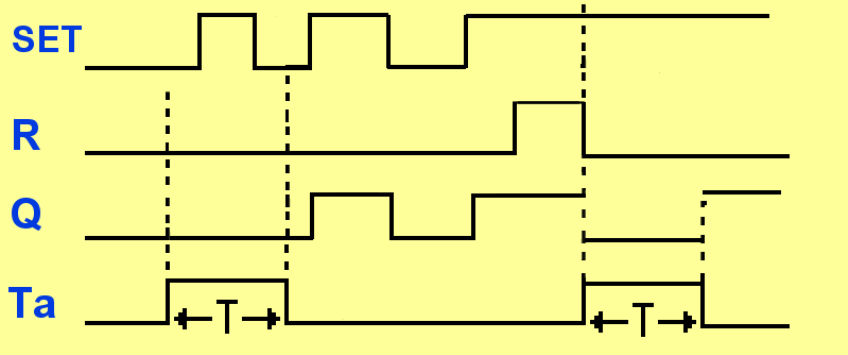
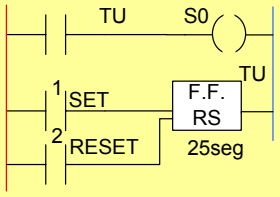


Figura 36. Oscilograma del temporizador tipo U.

FORMATO	ELEMENTOS INVOLUCRADOS	INTERVALO	PARÁMETRO DE SALIDA
TU#<>#S% (SET, RESET)	Entradas: # Salidas:%	0 a 99 segundos	Continuidad/ discontinuidad lógica %
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS (aprox.)**
T U 1 2 5 2 S 0 =	TU 1 25 2 S0 =		U I 1 S A 0 U I 2 R A 0

5.14 Temporizador tipo "V" (de retardo a la conexión).

Equivalente Siemens: SE. Con este temporizador, la salida Q (cualquier salida o marca) se activa una vez que ha transcurrido un periodo de tiempo parametrizable T (de 0 a 99 segundos). La cuenta de tiempo comienza cuando el disparador Trg (cualquier entrada o marca) cambia de "0 a 1", si este disparador cambia en un tiempo Ta (tiempo aplicado) a "0" antes de haber transcurrido el intervalo programado, el temporizador se detiene.

La figura 37 muestra el diagrama de tiempos asociado y enseguida se muestra una tabla descriptiva de esta función. El ejemplo ilustra un temporizador ajustado a 25 segundos, que se activa con la entrada número uno quedando como salida la número cero.

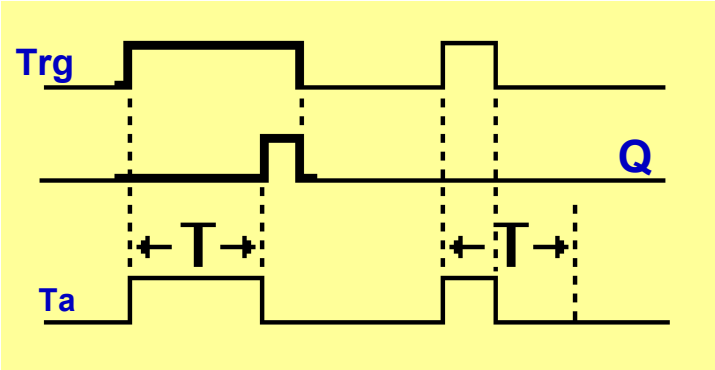
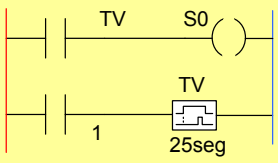


Figura 37. Oscilograma del temporizador tipo V.

FORMATO	ELEMENTOS INVOLUCRADOS	INTERVALO	PARÁMETRO DE SALIDA
TV#<>*S%	Entradas: # Salidas:%	0 a 99 segundos	Continuidad/ discontinuidad lógica %
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS (aprox.)**
T V 1 2 5 1 S 0 =	TV 1 25 1 S0 =		U I 1 FR T1 U I 2 L S5T#25s SE T1 U I 3 R T1 U T1 = A 0

5.15 Temporizador tipo "W" (retardo a la conexión memorizado).

Equivalente Siemens: SS. Tras un impulso de entrada del disparador Trg (cualquier entrada o marca que cambia de "0 a 1") transcurre un tiempo parametrizable T (de 0 a 99 segundos) que constituye el retardo a la conexión, después del cual se activa la salida Q (cualquier salida o marca). A través de la entrada de reset R (cualquier entrada o marca que cambia de "0 a 1"), el tiempo para el retardo a la conexión y la salida se ponen a cero (Reset tiene preferencia respecto a Trg). Remanencia activada en la salida (memoria): el estado de Q se guarda de forma remanente. Si la entrada Trg pasa del estado "0 al 1", comienza a transcurrir el tiempo actual Ta. Una nueva conexión de la entrada Trg no tiene efecto sobre Ta. La salida y el tiempo Ta se restablecen nuevamente a 0 cuando la entrada R toma estado 1.

La figura 38 muestra el diagrama de tiempos asociado y enseguida se muestra una tabla descriptiva de esta función. El ejemplo ilustra un temporizador ajustado a 25 segundos, que se activa con la entrada numero uno y el reset con la entrada número dos, quedando como salida la número cero.

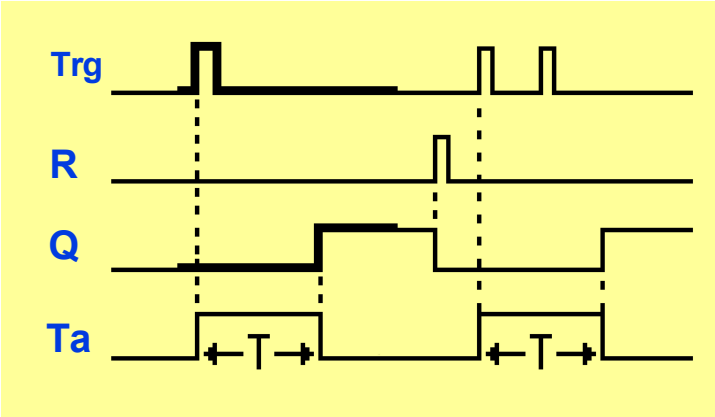
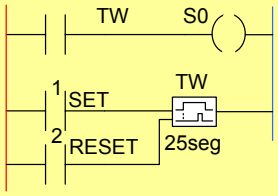


Figura 38. Oscilograma del temporizador tipo W.

FORMATO	ELEMENTOS INVOLUCRADOS	INTERVALO	PARÁMETRO DE SALIDA
TW#<>#S% SET RESET	Entradas: # Salidas:%	0 a 99 segundos	Continuidad/ discontinuidad lógica %
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS (aprox.)**
T W 1 2 5 2 S 0 =	TW 1 25 2 S0 =		U I 1 FR T1 U I 2 L S5T#25s SS T1 U I 3 R T1 U T1 = A 0

5.16 Función contador

Instrucción: "Z" equivalente Siemens: combinación de "ZV" y "ZR"

Según la parametrización, un impulso de entrada SET (cualquier entrada o marca que cambia solamente de "0 a 1") incrementa o decrementa un valor de cómputo interno. Cuando se alcanzan los valores umbral parametrizables (números de eventos para la conexión y para la desconexión), la salida Q (cualquier salida o marca) se activa o se reinicia. La dirección o sentido en que se cuenta (ascendente/descendente) puede cambiarse a través de la entrada de dirección Dir (cualquier entrada o marca puesta en "0" o en"1"), se define el sentido de cuenta: Dir = 0: cuenta de avance Dir = 1: cuenta de retroceso. Actualmente se disponen de dos contadores iguales, el U y el V.

En la figura 39 se muestran primero el diagrama de tiempos asociado, con un límite de conexión y de desconexión igual a cinco eventos, y luego una tabla descriptiva de esta función de conteo. El ejemplo en la tabla ilustra un contador ajustado a 13 eventos a la conexión y 25 a la desconexión. La cuenta está asociada con la entrada SET número uno, el RESET con la entrada número dos, el sentido o dirección de la cuenta con la entrada tres y quedando como salida la número cero.

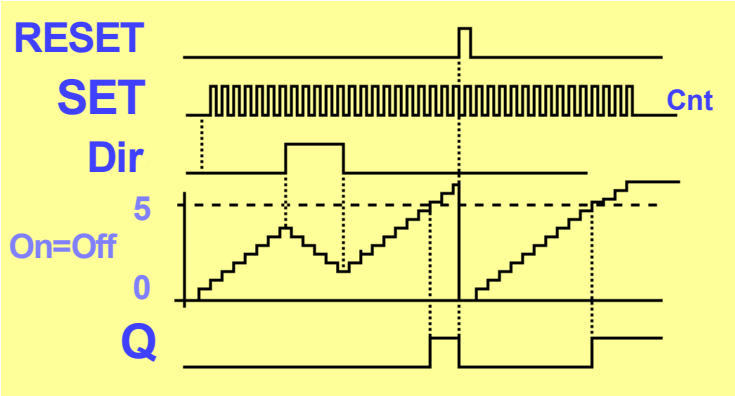
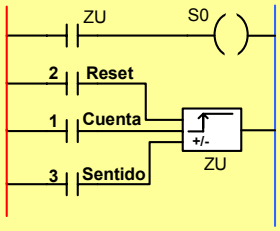


Figura 39. Oscilograma del contador.

FORMATO	ELEMENTOS INVOLUCRADOS	RANGO	PARÁMETRO DE SALIDA
ZU#<><>##S% ZV#<><>##S% SET conexión desconexión RESET dirección	Entradas: # Salidas:%	0 a 99 Eventos	Continuidad/ discontinuidad lógica %
CÓDIGO PROPIO (ejemplo)	**CÓDIGO CON FORMATO**	**EQUIVALENCIA GRÁFICA**	**EQUIVALENCIA SIEMENS (aprox.)**
ZU1132523S0= ZU 1 13 25 2 3 S0 =		U I4 L Z1 ZV Z1 T MW4 U I5 LC Z1 ZR Z1 T MW7 U I3 U Z1 L C#20 = A0 S Z1 U I7 R Z1

5.17 Función FIN DE PROGRAMACIÓN

Instrucción: “FF” o “ÿ” (tecleando ALT+0255) sin equivalente.

Esta instrucción le indica al PLC el final del programa de usuario en la memoria EEPROM, el apuntador que direcciona el programa se pone en cero y comienza el “scan” del programa completo nuevamente.

Esta función o instrucción sólo se introduce una vez al final de la secuencia de programación, seguido de esto se mostrará el mensaje en la Hyperterminal “FIN DE PROGRAMACIÓN”.

El caracter “FF” (hexadecimal) o “ÿ” (ASCII [61]) revela al CPU donde terminan las instrucciones del programa de usuario y donde comienza el área en blanco de la memoria EEPROM, ya que en esta última, cuando está en blanco todas sus localidades están con ese caracter. La figura 40 muestra el contenido de la memoria EEPROM, ya programada, se puede ver el espacio en blanco con el caracter “ÿ”o “FF” al final de la secuencia del programa.

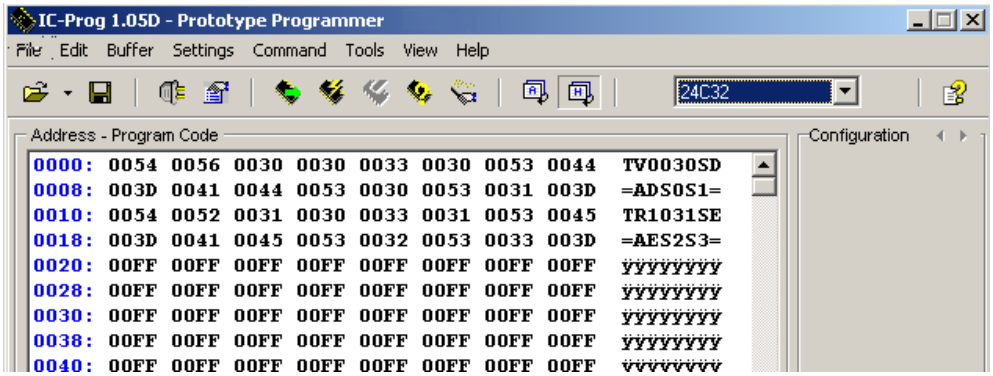


Figura 40. Ventana del programa para leer y grabar memorias y PIC

6. EJEMPLOS DE PROGRAMACIÓN

6.1 Ingresando el programa

En capítulos anteriores se mencionó acerca de cómo configurar la Hyperterminal de Windows para establecer la correcta disposición del puerto serial con el PLC. En este capítulo se mostrará cómo se introducen los programas en el lenguaje de lista de instrucciones conformado por el juego citado anteriormente.

Debido al grado de desarrollo de la interfase y a las otras formas de programación de la memoria EEPROM que resultan básicas, el pequeño intérprete implementado en el firmware muestra en la pantalla de la Hyperterminal el número hexadecimal equivalente al caracter ASCII [61] de la instrucción o comando introducido para un mejor control en la depuración avanzada necesaria para la interfase final.

Recordemos que para empezar a programar, el pin de programación debe estar en "1" y para finalizar la secuencia de programación deberemos introducir el caracter ASCII que corresponde al número hexadecimal FF, seguido de esto aparecerá la leyenda: "FIN DE PROGRAMACIÓN AL PLC", después de esto deberemos de colocar ahora el pin de programación en "0". Finalmente reiniciamos el PLC y en la pantalla aparecerá la leyenda: "MODO RUN". Para la situación en donde se cometa un error, se tendrá que iniciar nuevamente la secuencia de programación tras haber dado reset a la unidad PLC.

Para ingresar un programa en formato de lista de instrucciones se procederá a anotar de forma secuencial línea por línea los operadores y operandos de acuerdo a la sintaxis o formato previamente establecido. La siguiente tabla muestra los caracteres de la programación mostrando el caracter ASCII que se debe introducir directamente por el teclado.

FUNCIÓN	Valor ASCII	Valor Hexadecimal
AND	A	41
NAND	a	61
OR	O	4F
NOR	o	6F
NOT	N	4E
OUT	S	53
TEMPORIZADOR	T	54
CONTADOR	Z	5A
Fin de rama	=	3D
Fin de programa	ÿ	FF

Tabla 5. Equivalencia entre caracter y operando.

A continuación se presentan algunos ejemplos básicos de programación utilizando nuestro juego de instrucciones.

6.2 Ejemplo 1: Compuertas lógicas

A manera de ejemplo se procederá a ingresar paso a paso un pequeño programa de muestra con el fin de familiarizarse con la forma de entrada del programa. Dicho programa representará las ecuaciones resultantes en modo lista de instrucciones AWL[24]. En la figura 41 se muestran en dos representaciones gráficas.

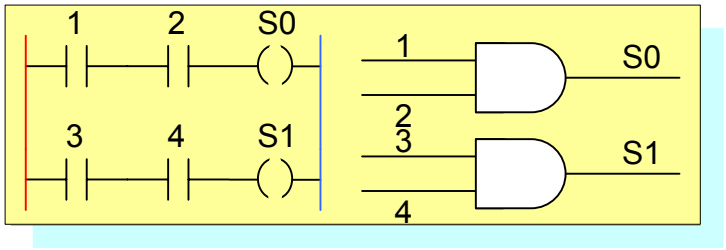


Figura 41. Dos diferentes representaciones de la función AND.

Recordaremos que en la sintaxis del AWL "propio" dicho programa resultaría de la siguiente manera como muestra la tabla siguiente.

Renglón	Instrucción	Descripción
1	A	Realizará una "and" entre el operando siguiente usualmente una entrada (en este caso la 0) y con la siguiente "and", si existe, y además con el resultado anterior (a menos que se tenga un "="al final)
2	0	Entrada 0 (cero)
3	A	Realizará una "and" entre el operando siguiente usualmente una entrada (en este caso la 1) y con la siguiente "and", si existe, y además con el resultado anterior (a menos que se tenga un "="al final)
4	1	Entrada 1
5	S	Despliega el resultado de la operación(es) precedente(s) y lo saca en el número de salida indicada en la siguiente línea de instrucción
6	0	Salida 0 (cero)
7	=	Fin de rama que indica que el stack de resultados se restablece
8	A	Realizará una "and" entre el operando siguiente usualmente una entrada (en este caso la 2) y con la siguiente "and", si existe, y además con el resultado anterior (a menos que se tenga un "="al final)
9	2	Entrada 2

10	A	Realizará una “and” entre el operando siguiente usualmente una entrada (en este caso la 3) y la siguiente “and”, si existe, y además con el resultado anterior (a menos que se tenga un “=”al final)
11	3	Entrada 3
12	S	Despliega el resultado de la operación(es) precedente(s) y lo saca en el número de salida indicada en la siguiente línea de instrucción
13	1	Salida 1
14	=	Fin de rama que indica que el stack de resultados se restablece
15	ALT+0255	FIN DE PROGRAMACIÓN AL PLC, le indica al CPU el final del programa en la memoria EEPROM

Tabla 6. Programa de dos AND´s.

La manera en que el programa es introducido a través de la Hyperterminal se muestra en la figura 42.

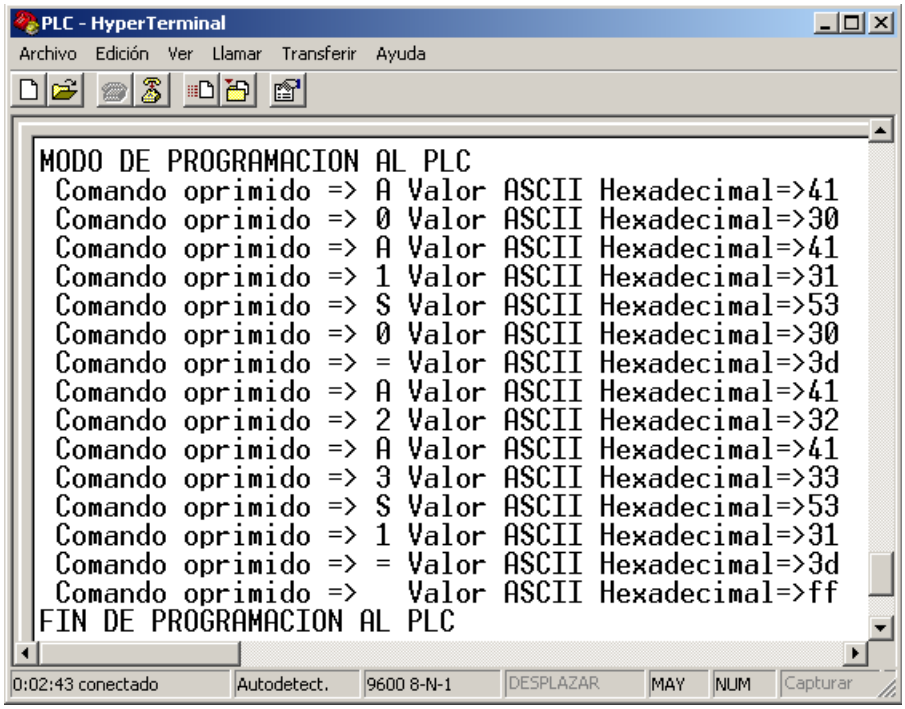


Figura 42. Hyperterminal mostrando el programa introducido.

En dicha figura se muestran los caracteres que secuencialmente se ingresan, además de mostrar el valor hexadecimal de la instrucción, con ello se logra una mejor apreciación de lo que debe contener la memoria EEPROM y como se organizan los datos dentro de ella.

6.3 Ejemplo 2: Contador avance-retroceso

En la figura 43 se muestra dicho contador, donde la entrada corresponde a la número uno, el número de eventos para la conexión es de cuatro mientras que para la desconexión son diez, el reset es la entrada dos y el sentido de la cuenta es la entrada tres, finalmente la salida es la uno.

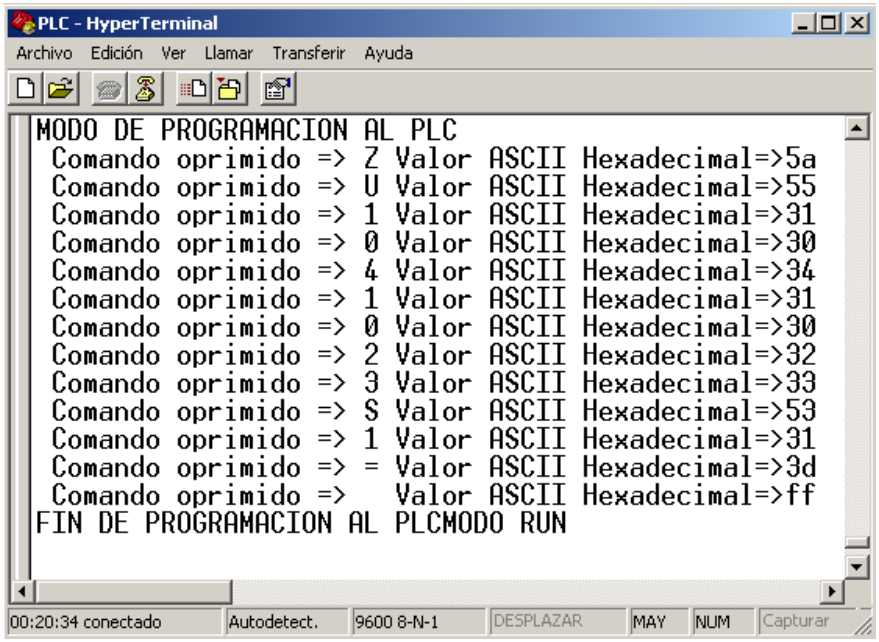


Figura 43. Programa de un contador avance retroceso.

6.4 Ejemplo 3: Compuerta XOR

La programación de dicha compuerta lógica es implementada mediante una combinación de funciones AND y NOT, aunque es posible también implementarla con funciones NAND. La siguiente tabla muestra la secuencia de instrucciones necesaria para implementar un XOR entre la entrada cero y la uno.

Renglón	Instrucción	Descripción
1	A	Realizará una “and” entre el operando siguiente usualmente una entrada (en este caso la 0) y la siguiente “and”, si existe, y además con el resultado anterior (a menos que se tenga un “=”al final)
2	0	Entrada 0 (cero)
3	N	Niega o invierte el resultado del siguiente operando
4	A	Realizará una “and” entre el operando siguiente usualmente una entrada (en este caso la 1) y la siguiente “and”, si existe, y además con el resultado anterior (a menos que se tenga un

		"="al final)
5	1	Entrada 1 (uno)
6	S	Despliega el resultado de la operación(es) precedente(s) y lo saca en el número de salida indicada en la siguiente línea de instrucción
7	0	Salida 0 (cero)
8	=	Fin de rama que indica que el stack de resultados se resetea
9	N	
10	A	
11	0	
12	A	
13	1	
14	S	
15	0	
16	=	
17	alt + 0255	FIN DE PROGRAMACIÓN AL PLC

Tabla 7. Programa de una XOR.

6.5 Ejemplo 4: Control de un motor

El siguiente programa controla el sentido de rotación de un motor de corriente directa dándole un tiempo muerto de 3 segundos en cada cambio de dirección, y mediante un interruptor se controla el arranque rápido y con otro el encendido. La tabla siguiente lista dicho programa.

Renglón	Instrucción	Descripción
1	A	Realiza la comprobación "examinar-encendido"
2	2	Para el interruptor general en la entrada dos
3	S	Si la comprobación anterior es verdadera, el interruptor
4	6	de la salida seis se cierra.
5	=	fin de la rama
6	A	Realiza la comprobación "examinar-encendido"
7	3	Para el interruptor de velocidad normal en la entrada tres
8	S	Si la comprobación anterior es verdadera, el interruptor
9	5	de la salida cinco se cierra.
10	=	fin de la rama
11	T	Hace un llamado a la función temporizador T
12	V	el tipo elegido es el V (de retardo a la conexión)
13	0	la entrada de TV es la cero
14	0	se ingresan tres segundos

15	3	
16	0	se repite el número de la entrada
17	S	instrucción que direcciona la salida a la marca interna D
18	D	
19	=	fin de la rama
20	A	realiza la comprobación “examinar-encendido”
21	D	si la marca D es "0" o "1" también lo serán las salidas
22	S	Cero y uno
23	0	
24	S	
25	1	
26	=	fin de la rama
27	T	Hace un llamado a la función temporizador T
28	R	el tipo elegido es el R (idéntico al V)
29	1	la entrada de TR es la uno
30	0	se ingresan tres segundos
31	3	
32	1	se repite el número de la entrada
33	S	instrucción que direcciona la salida a la marca interna E
34	E	
35	=	fin de la rama
36	A	realiza la comprobación “examinar-encendido”
37	E	si la marca E es "0" o "1" también lo serán las salidas
38	S	Dos y tres
39	2	
40	S	
41	3	
42	=	fin de la rama
43	alt + 0255	le indica al PLC fin de la programación

Tabla 8. Programa de control de un motor.

El diagrama del programa se muestra a continuación (figura 44); además se incluyen los diagramas de escalera y los de la lógica alambrada empleando el PLC didáctico (figuras 45 y 46). Obsérvese que la salida Q1 corresponde a S0 y así sucesivamente, esto es para que haya correspondencia entre la serigrafía del circuito impreso con los diagramas lógicos.

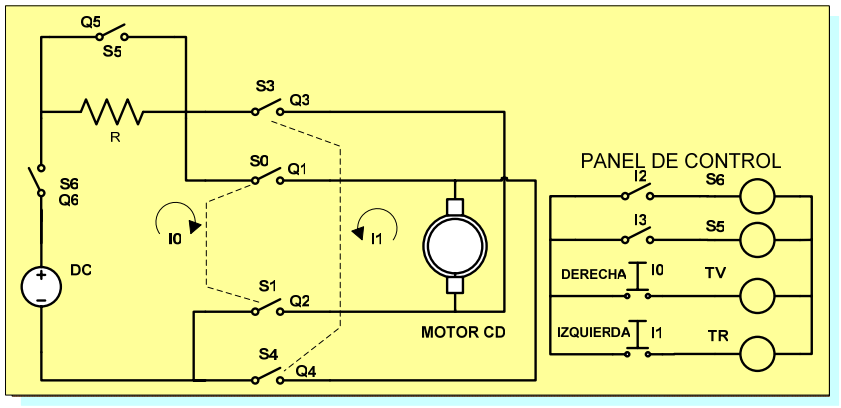


Figura 44. Diagrama del control eléctrico de la conexión al PLC con el motor.

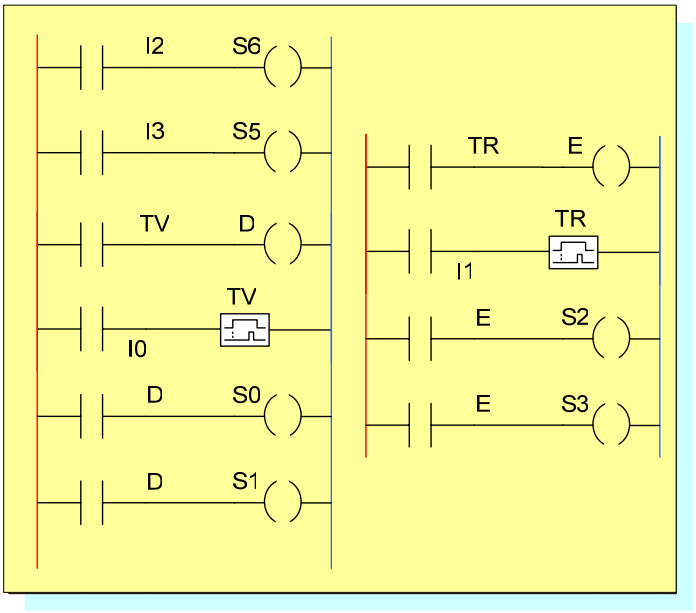


Figura 45. Diagrama de escalera del control del motor.

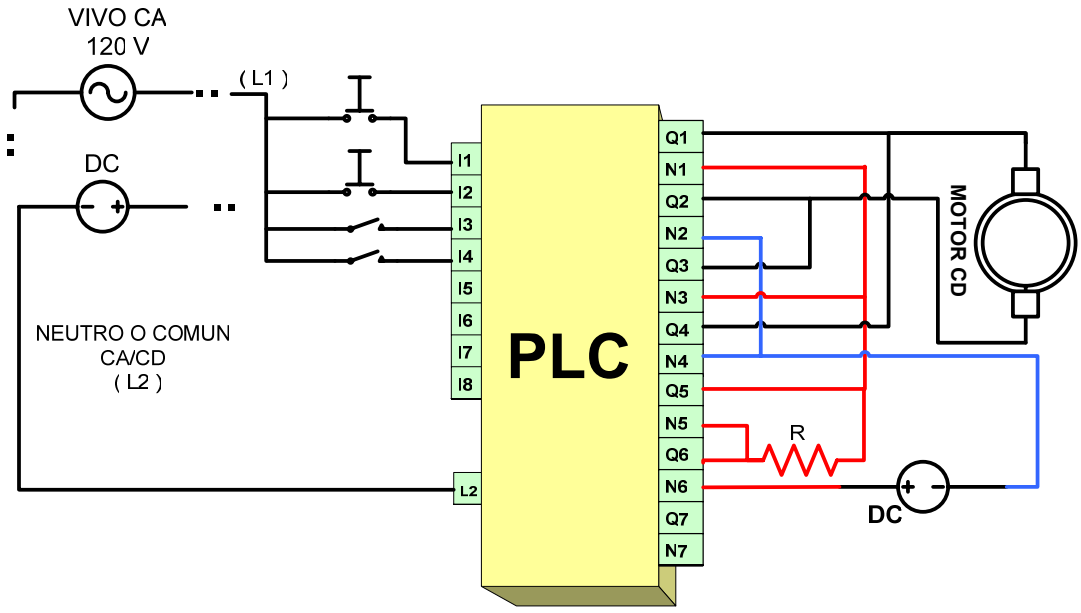


Figura 46. Disposición física del PLC con el motor.

7. ALGORITMO DE FUNCIONAMIENTO

Para que internamente el microcontrolador realice el ciclo de barrido del programa o "Scan" fue implementando un algoritmo que es casi idéntico a los empleados en PLC´s comerciales, la diferencia primordial es que realiza las funciones de atender tanto a las salidas como a las entradas de forma inmediata durante el paso de instrucción [3].

En esta forma de atender y despachar el programa de usuario, los pasos de instrucción en el "Scan" son divididos a su vez en "scan de entrada" y "scan de salida". En el primero se monitorea el estado de la entrada o entradas indicadas en el paso de instrucción del programa actualizando de forma inmediata a las variables o partes involucradas en dicho paso de instrucción, mientras que en el segundo, dan acceso a la salida o salidas involucradas activándolas o desactivándolas.

Las figuras 47 y 48 muestran el proceso de ejecución del programa de usuario en un paso de instrucción cualquiera. Se observa cómo una vez que el programa indica atender una entrada, ésta se procesa enseguida e inmediatamente prosigue con el proceso de atender a la salida.

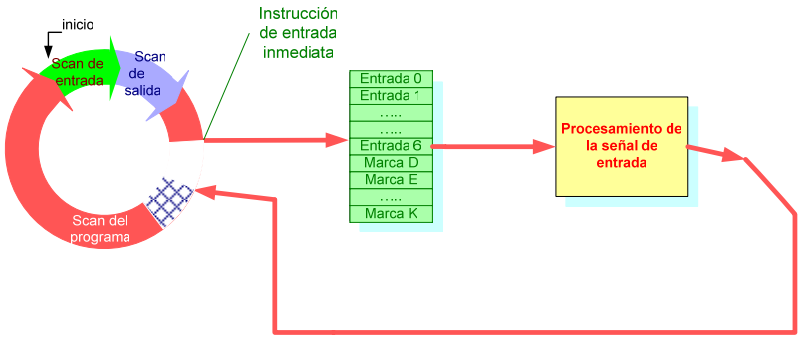


Figura 47. Scan de entrada.

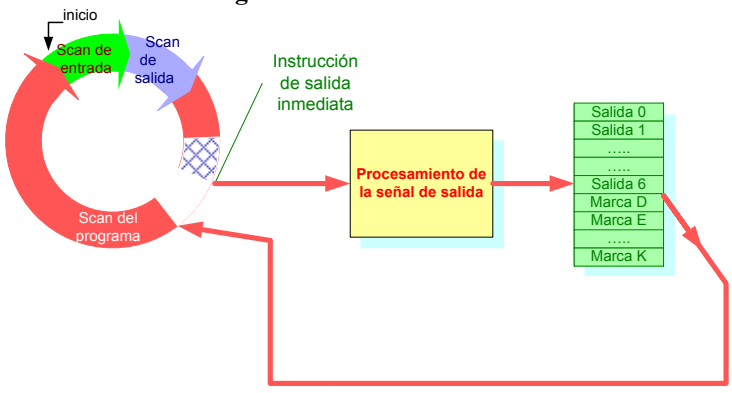


Figura 48. Scan de salida.

La ejecución completa de estos dos "Scans" constituye un paso de instrucción dentro del programa de usuario (ver figura 49). En el paso de instrucción se hallan todas las instrucciones de una rama o escalón dentro del programa de usuario y un programa completo es una secuencia de pasos de instrucciones.

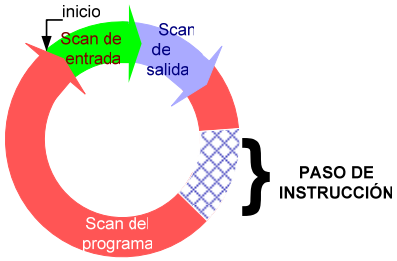


Figura 49. Paso de instrucción en el Scan.

En la figura 50 se muestra primero un diagrama que recuerda la programación en escalera seguido del diagrama del "Scan". Se observa en este último que el flujo del programa se adecua a los requerimientos de la programación en forma de escalera y a su vez en la de lista de instrucciones como anteriormente establecimos.

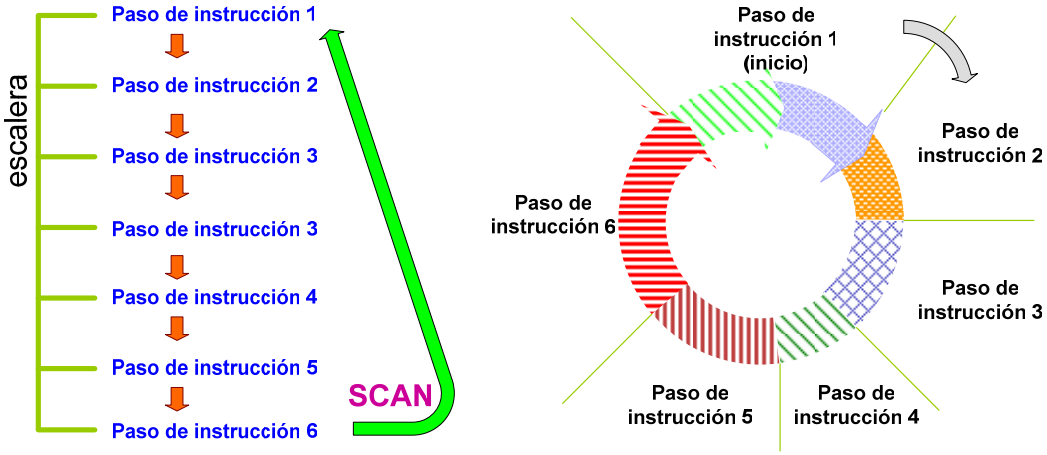


Figura 50. Comparación entre un diagrama de escalera y el Scan.

7.1 El Firmware

El juego de instrucciones creado para este PLC, dependiendo del tipo de instrucción que se trate, se comporta como un "Scan" de entrada o salida internamente en el firmware del microcontrolador, por ello, tienen tanto estructura como comportamiento semejante y elementos o funciones en común.

El firmware comienza tras inicializar registros, variables internas y puertos I/O, con una rutina de poleo que verifica el modo del PLC, es decir, corriendo el programa de usuario (modo "RUN"), o si está esperando el ingreso del programa (modo "MODO DE PROGRAMACIÓN AL PLC"), y dependiendo del modo en que esté, la ejecución del firmware seguirá un camino u otro. A continuación de muestran los diagramas de flujo asociados al firmware del microcontrolador.

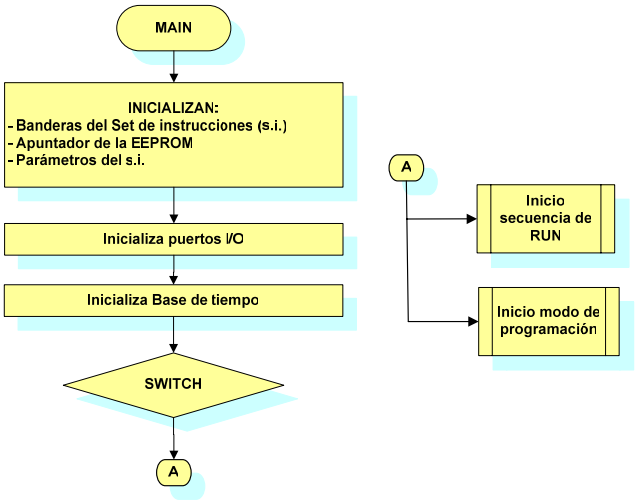


Figura 51. Diagrama global de funcionamiento.

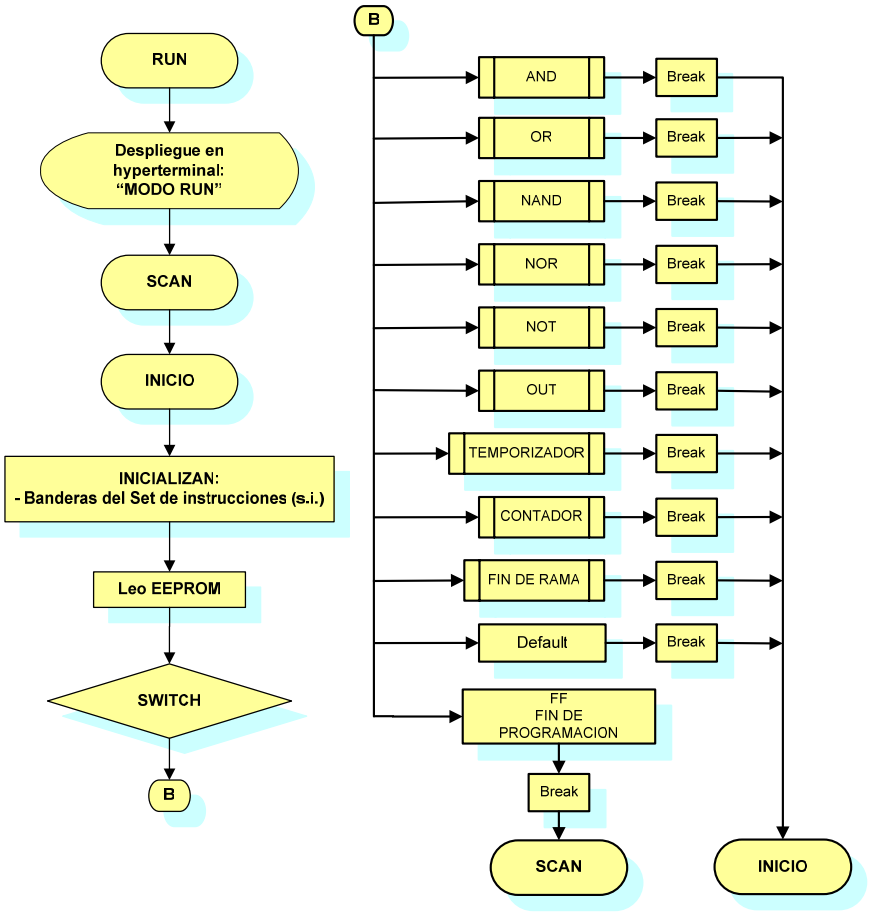


Figura 52. Inicio de secuencia RUN.

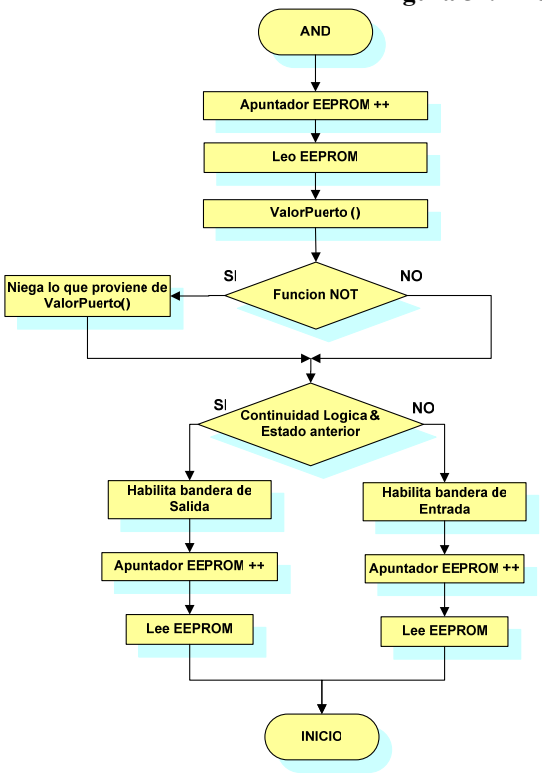


Figura 53. Instrucción AND.

Este diagrama de flujo muestra la instrucción AND (A) (figura 53), que junto con otras instrucciones (OR, NAND, NOT) comparten una estructura y funcionamiento casi idéntico, las diferencias son sutiles y van asociadas con las banderas empleadas y la forma en que se evalúa la continuidad lógica.

Para el caso de los temporizadores y contadores no cambia mucho la situación, la continuidad lógica se compara con una variable incrementada con una base de tiempo, mientras que la cuenta con una variable de conteo.

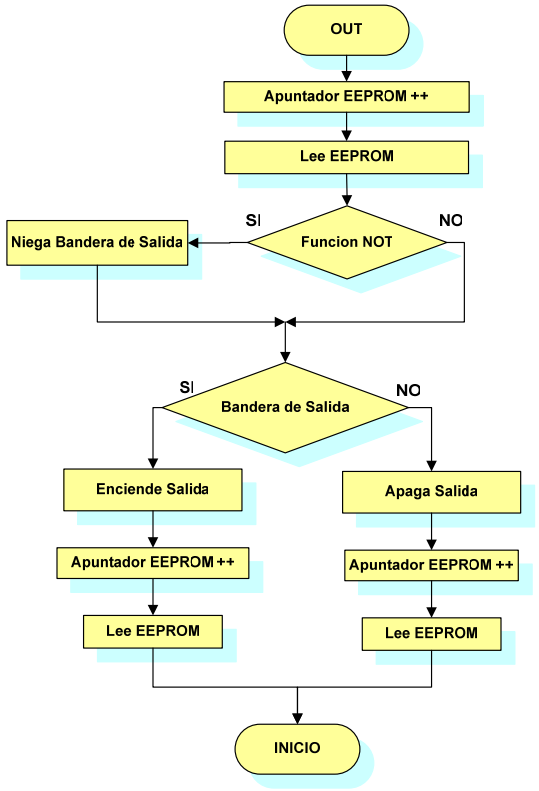


Figura 54. Instrucción OUT

Aquí el diagrama de flujo de la instrucción OUT (S) (figura 54), se ilustra ya que a pesar que comparte características muy similares con las otras funciones, no resulta muy evidente la forma en que funciona y atiende las salidas. Se puede notar la forma inmediata que procesa la salida, lo que determina el "Scan" de salida.

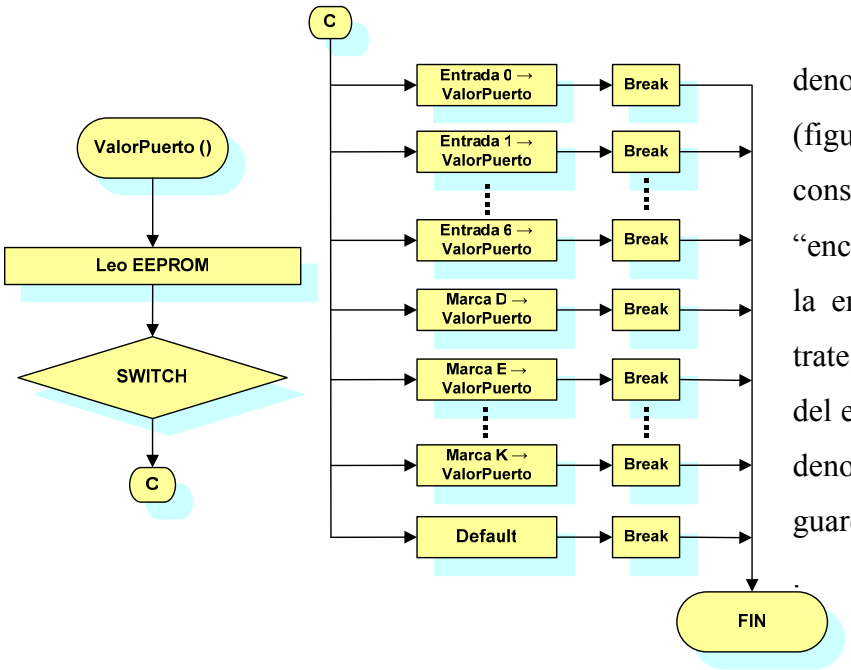


Figura 55. Función ValorPuerto.

En la función interna denominada ValorPuerto (figura 55), su trabajo consiste en adquirir el estado "encendido" o "apagado" de la entrada cualquiera que se trate, sea física proveniente del exterior del PLC o interna denominada marca y guardarlo en una variable del mismo nombre.

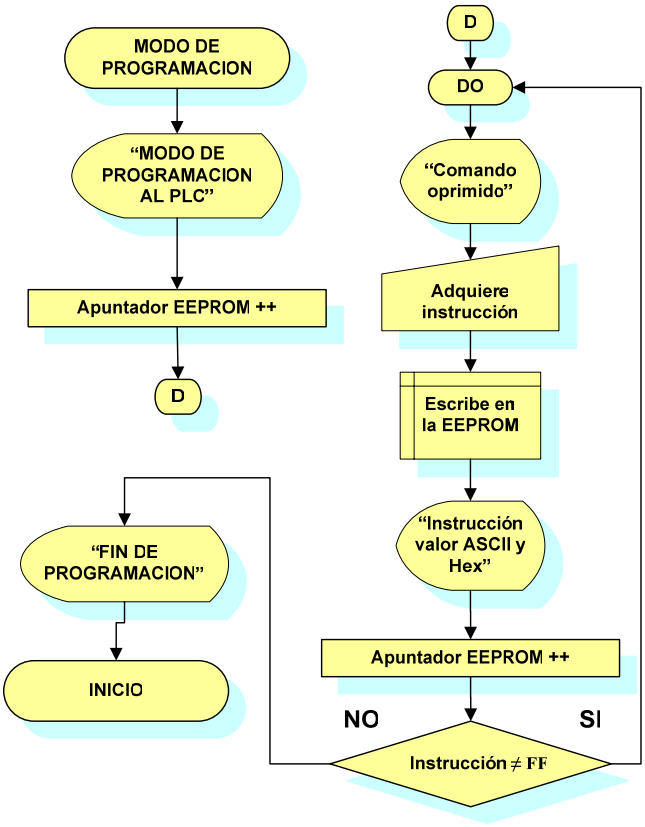


Figura 56. Inicio modo de programación.

La manera en que trabaja el PLC cuando se escoge el MODO DE PROGRAMACIÓN (ver figura 56), resulta en un ciclo continuo de escritura en la memoria EEPROM, en donde se despliega una a una la instrucción introducida. El final de la programación la dictamina un caracter ya establecido como "FF" en hexadecimal.

8. RESULTADOS Y CONCLUSIONES

Los resultados obtenidos satisfacen los requerimientos planteados. Se ha desarrollado un PLC que con base en los estándares, su funcionamiento y desempeño se puede comparar en forma razonable a los comercialmente disponibles. Aunque, si bien, la forma de programarlo no es del todo amigable (considerando que así es el lenguaje de lista de instrucciones), representa un comienzo en el desarrollo de un sistema integral de aprendizaje de PLC´s. Por ello, para darle continuidad al proyecto, actualmente se está desarrollando un entorno de programación gráfico en donde se pudiese programar en cualquier lenguaje estándar; así mismo, la manera en que se establece la comunicación con la computadora podría ser además utilizando la conectividad USB [52].

El PLC cuenta actualmente con ocho entradas aisladas ópticamente, capaces de responder a estímulos de corriente alterna o directa, acción que resulta del cambio de un interruptor de selección. Consta también de siete salidas a relevador capaz de drenar hasta 10 Amperes a 120 VCA o 24 VCA. Posee una entrada serial del tipo RS-232 con la cual el programa de usuario se ingresa tras poner el PLC en modo de programación mediante un interruptor, y para la actualización del firmware se conecta mediante un conector del tipo RJ-12 al ICD de Microchip.

La justificación del uso del ICD es la manera sencilla de actualizar el firmware del PLC totalmente ensamblado y con carga en sus salidas sin necesidad de desarmarlo, tan solo con modificar de posición un interruptor que habilita o deshabilita el empleo del ICD y conectar el PLC al Mplab ICD y este último a la computadora, enseguida actualizar mediante el Mplab IDE figura 57.

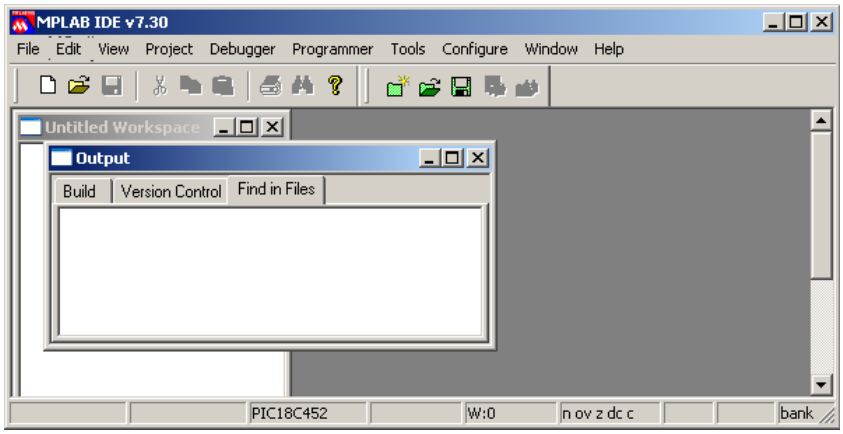


Figura 57. Ventana del Microchip Mplab IDE.

A continuación se muestran las características generales del PLC desarrollado.

8 Entradas	@ 120VCA/24VCD max
Voltaje de aislamiento	5300 VCA rms
Resistencia de aislamiento	10^{11} Ω
7 Salidas	10A @ 120VCA, 7A @ 240VCA, 10A @24VCD
Resistencia de contactos	50 mΩ
Tiempo de encendido	10 ms Máx.
Tiempo de apagado	5 ms Máx.
Aislamiento	Separación galvánica, 100M Mín .(500VCD)
Material del contacto	Aleación de plata.
Vida mecánica esperada	10, 000,000 operaciones. Mín.
Vida eléctrica esperada	100,000 operaciones. Mín.
Tensión de alimentación	120 VCA @ 60Hz
Memoria de usuario	8 kBytes
Temp. de funcionamiento	0 – 55 °C
Humedad relativa	10- 90 % sin condensación
Dimensiones	21x10x15.5 cm.

Tabla 9. Características obtenidas.

Debido que este sistema es una versión prototipo, no está exenta de cambios, expansiones y mejoras. Dentro de los posibles cambios está el uso de una fuente conmutada para la alimentación interna para la reducción en el tamaño y una batería de respaldo, así mismo un cambio en el número de las entradas y salidas de acuerdo a las necesidades o aplicaciones específicas. También, para mejorar el PLC se pretende adicionar un número mayor de entradas y salidas de forma modular, así como, formar módulos analógicos. En cuanto al firmware, un juego más extenso de funciones e instrucciones podría brindar más funcionalidad y de la misma manera seria conveniente desarrollar un algoritmo de seguridad contra fallas del sistema.

Por el momento, el juego de instrucciones resulta limitado, aunque con éste se pueden implementar funciones más complejas y resultaría idóneo tener una gama más amplia para programar. Se demostró que es posible la viabilidad de la programación de

acuerdo al método de lista de instrucciones apegándose a los estándares de programación con los cuales está inspirado este PLC.

El diseño electrónico industrial, es un trabajo que requiere de tiempo, dedicación, investigación y muchos recursos; además, el nivel de calidad con el que actualmente cuenta la industria se logró con la experiencia de años de investigación y desarrollo de equipo. Sin embargo, esto no significa que sea imposible para nosotros adentrarnos en este campo, ya que si en este trabajo se pudo alcanzar un nivel comparable al de un equipo diseñado por grupos de profesionales, con el trabajo de una sola persona, entonces es más probable que un grupo de ingenieros sea capaz de adentrarse poco a poco al campo de diseño electrónico, a niveles cada vez más competitivos.

Es evidente que para fomentar el entrenamiento de programación, así como facilitar la comprensión y el estudio de los PLC´s, se necesita de un proyecto que involucre un gran número de colaboradores que enriquezca y corrija el producto final. Una tarea multidisciplinaria como ésta no es sencilla, sin embargo, aquí se expuso una parte de esa enorme tarea, que se espera sea útil en breve.

En este trabajo se demostró que con relativamente pocos recursos puede ofrecerse una alternativa [36] que si bien no soluciona la totalidad de las deficiencias en el aprendizaje de la programación de un PLC, puede beneficiar a la enseñanza y entrenamiento en el uso de estos.

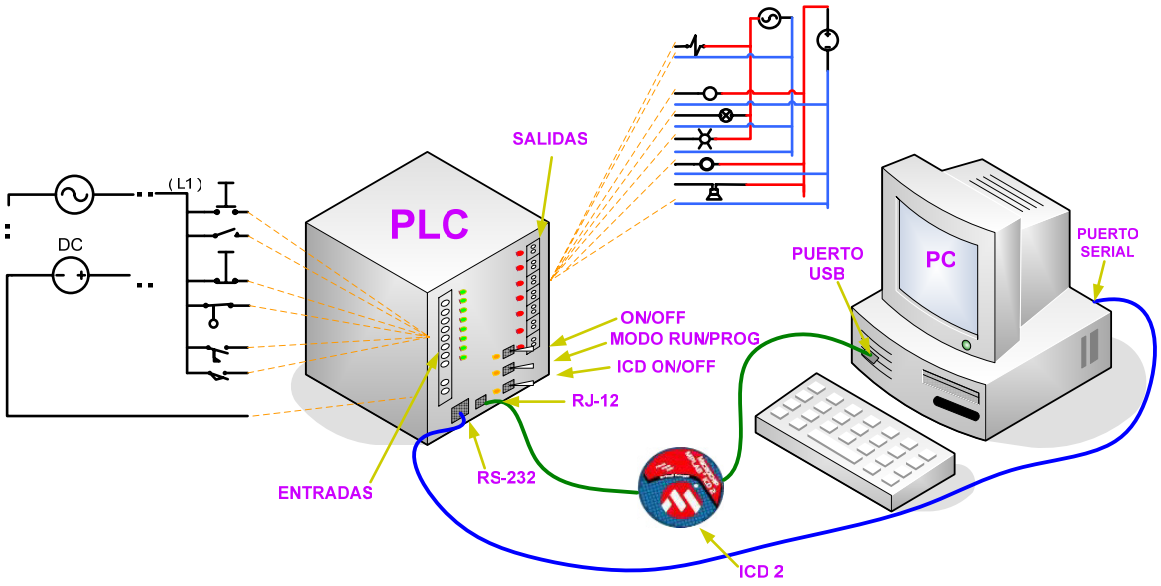
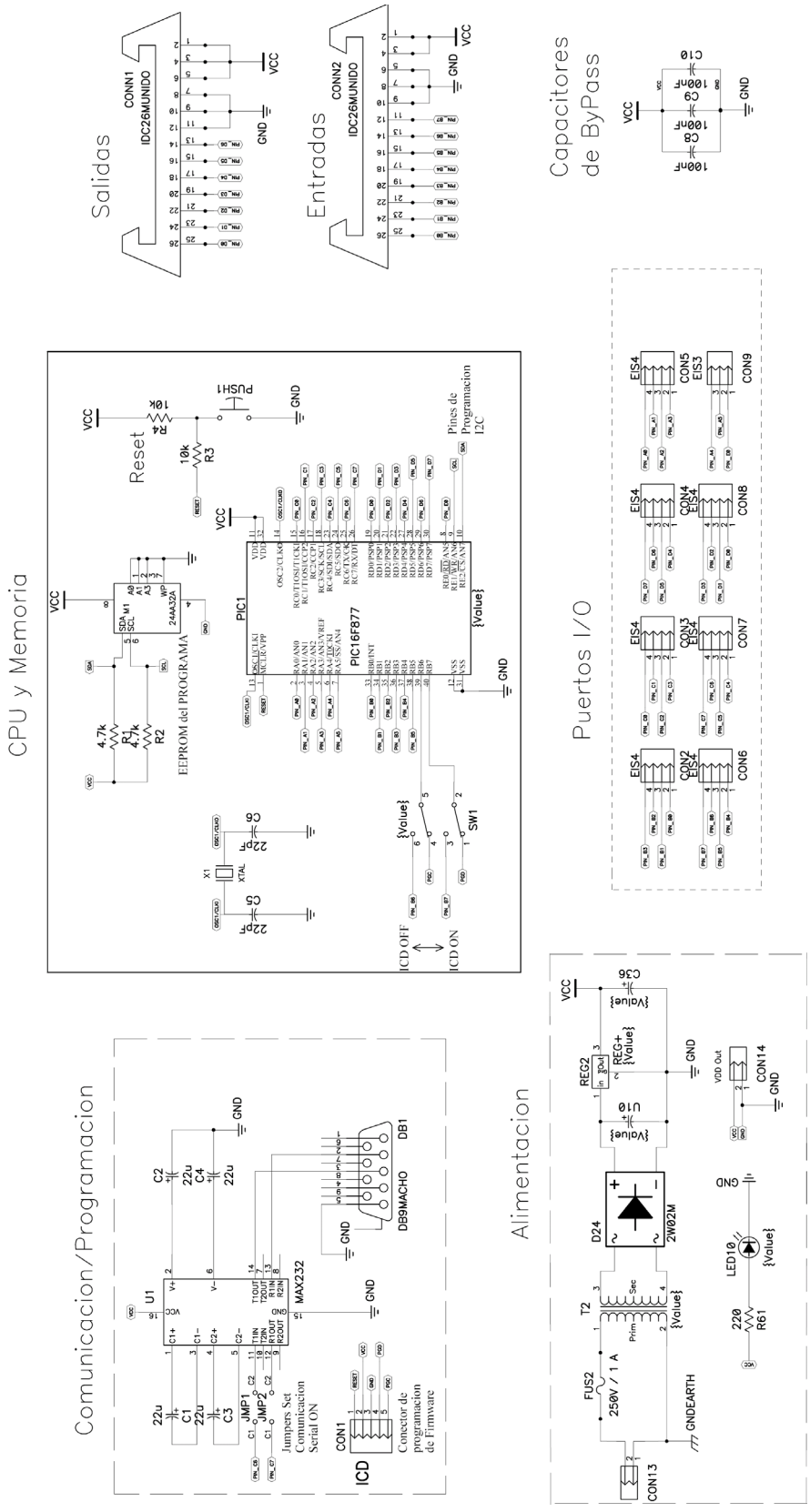
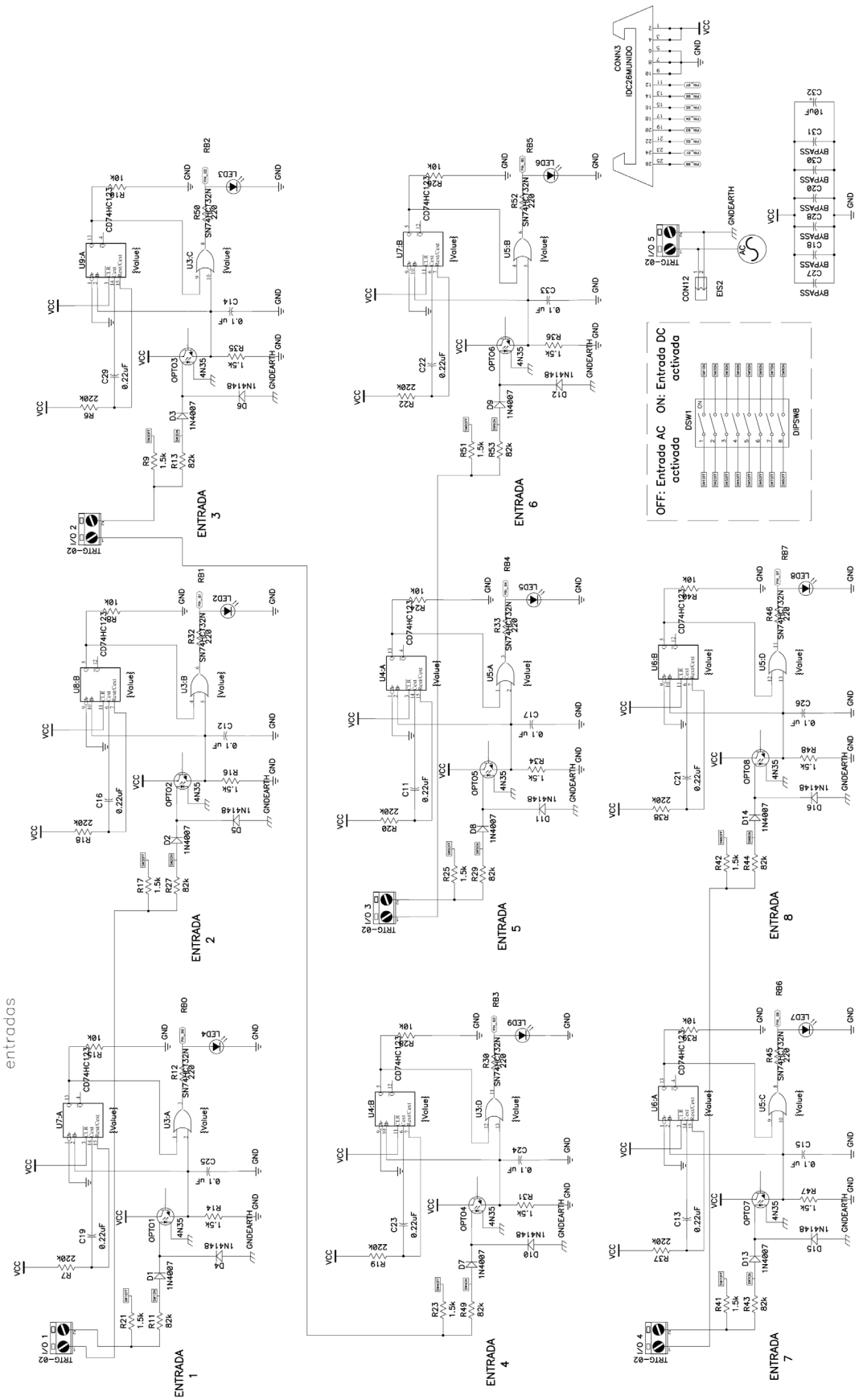
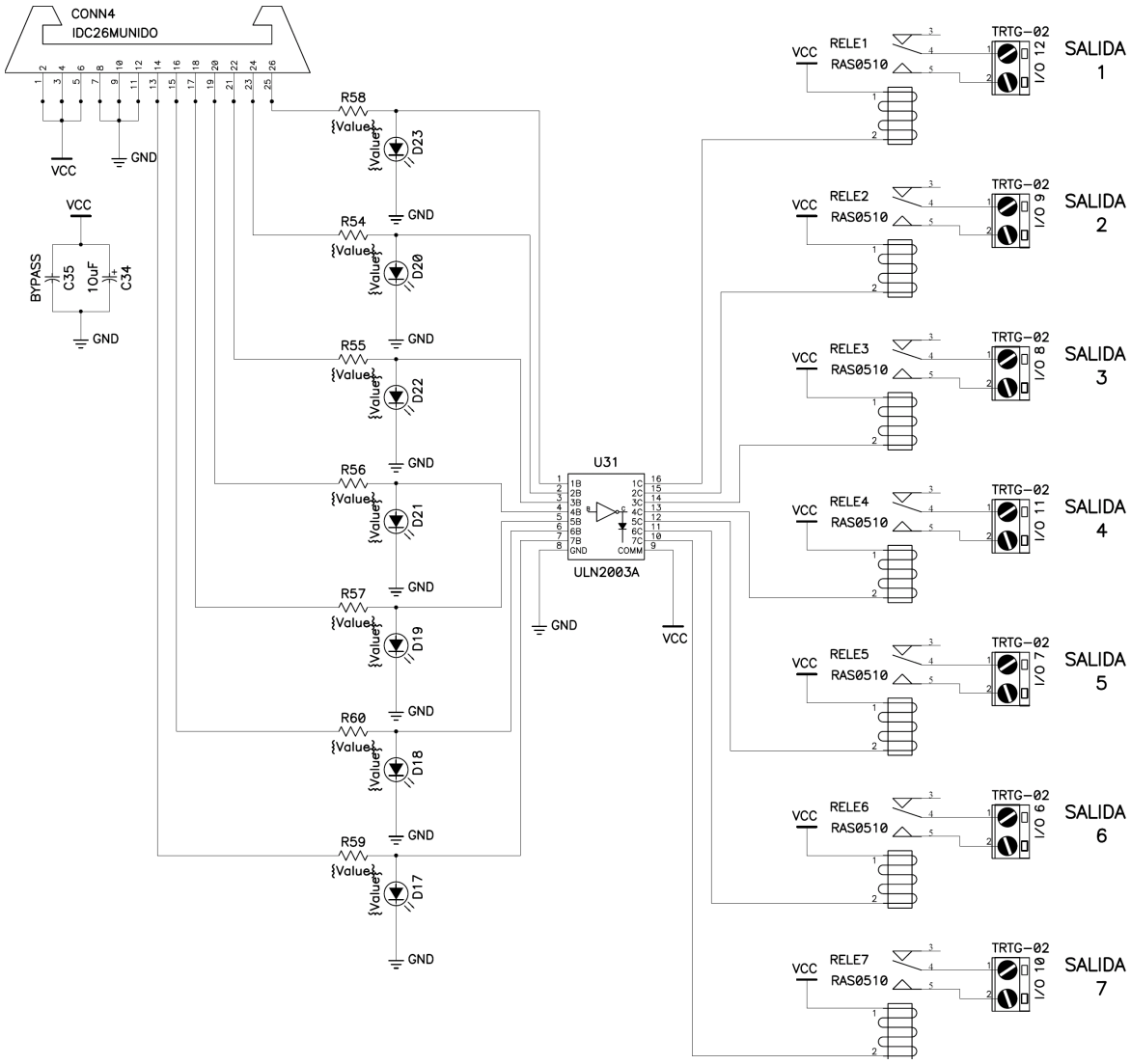


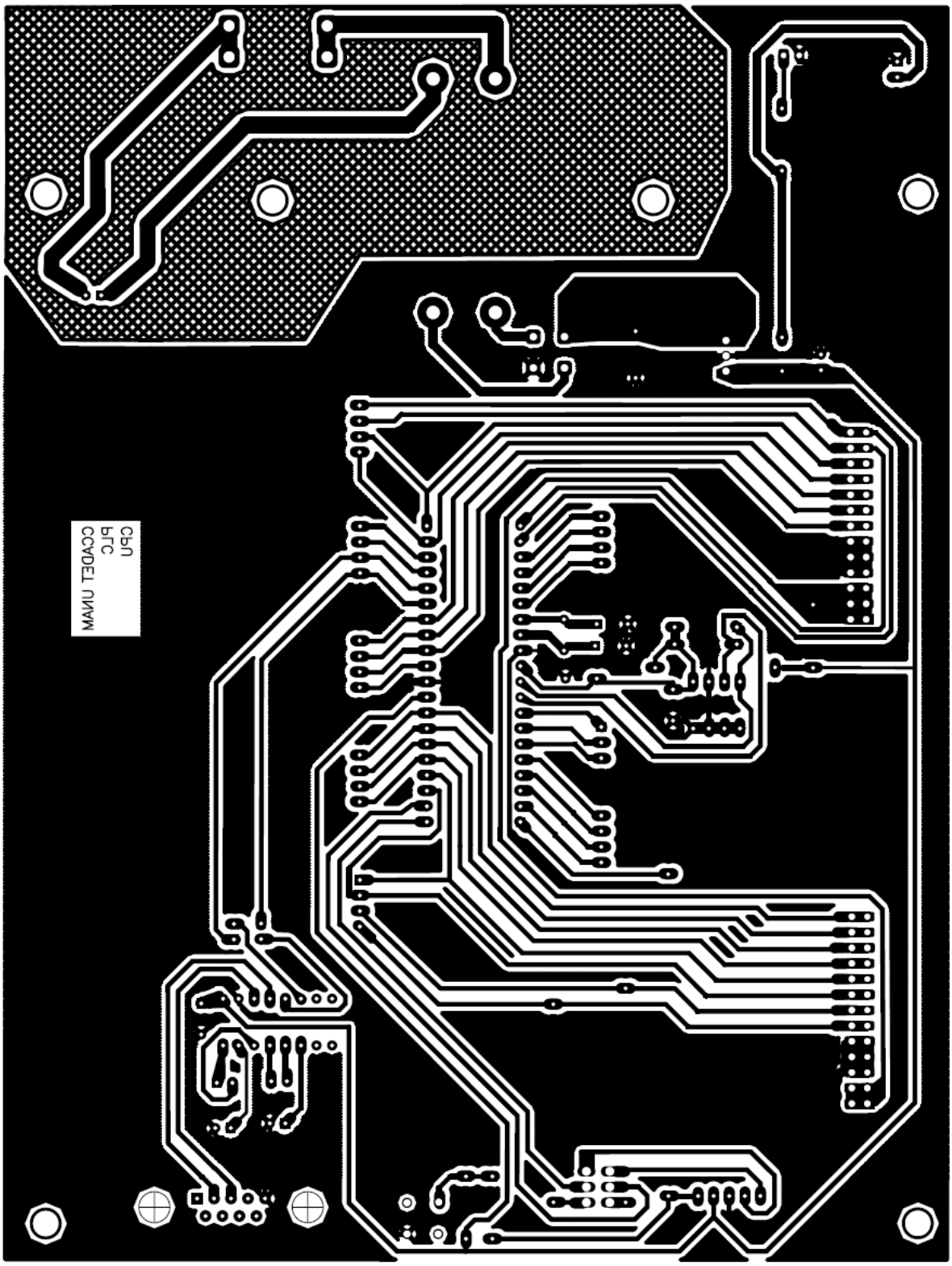
Figura 58. Disposición final del PLC y su entorno.

DIAGRAMAS Y CIRCUITOS IMPRESOS

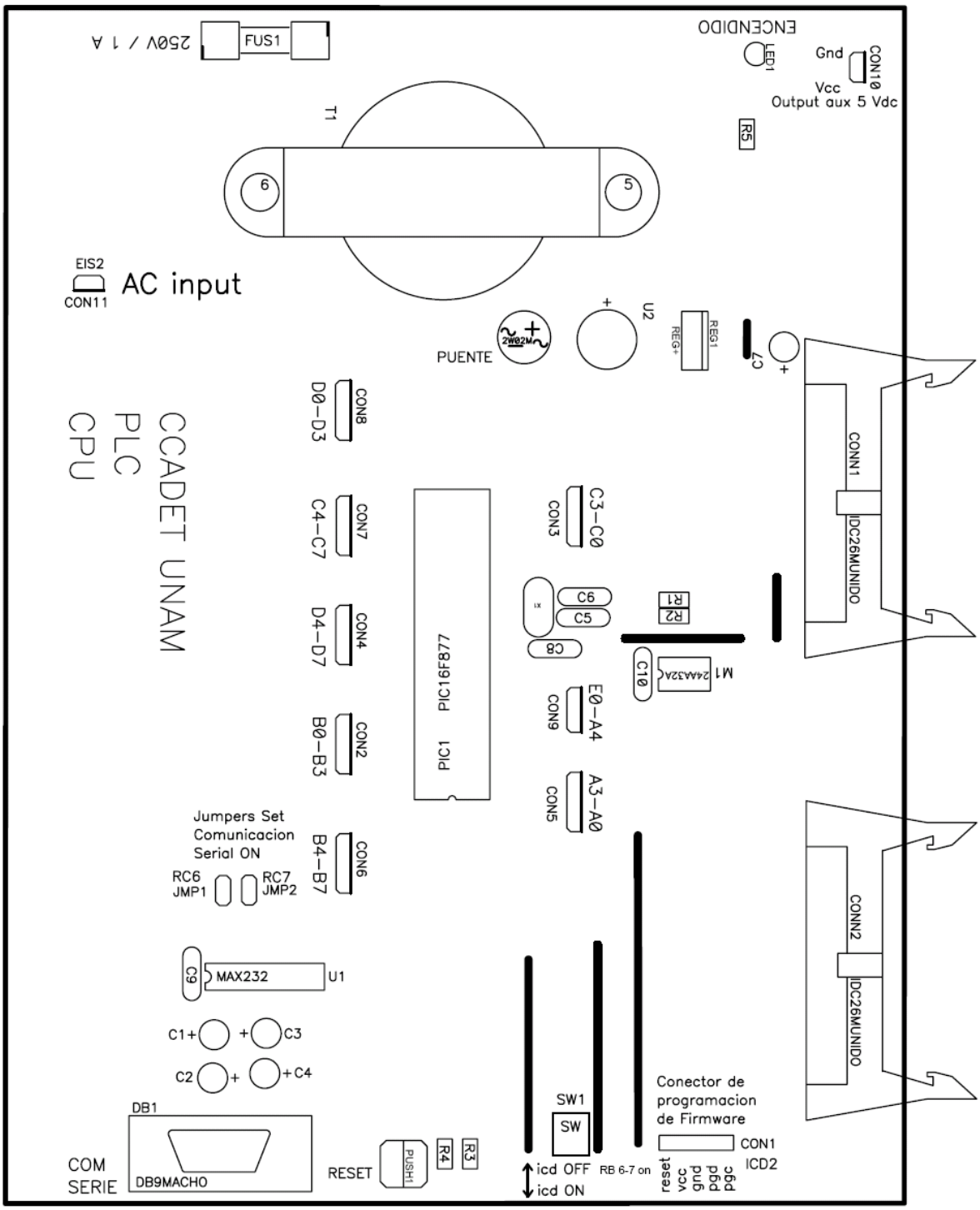



entradas
ENTRADA 1
ENTRADA 2
ENTRADA 3
ENTRADA 4
ENTRADA 5
ENTRADA 6
ENTRADA 7
ENTRADA 8
OFF: Entrada AC ON: Entrada DC activada
DSW1
DIPSW8
CONN3
IDC26MUNIDO
CON12
EIS2
CD74HC123
SN74HC32N
4N35
1N4148
1N4007
TRIG-02
VCC
GND
GNDEARTH

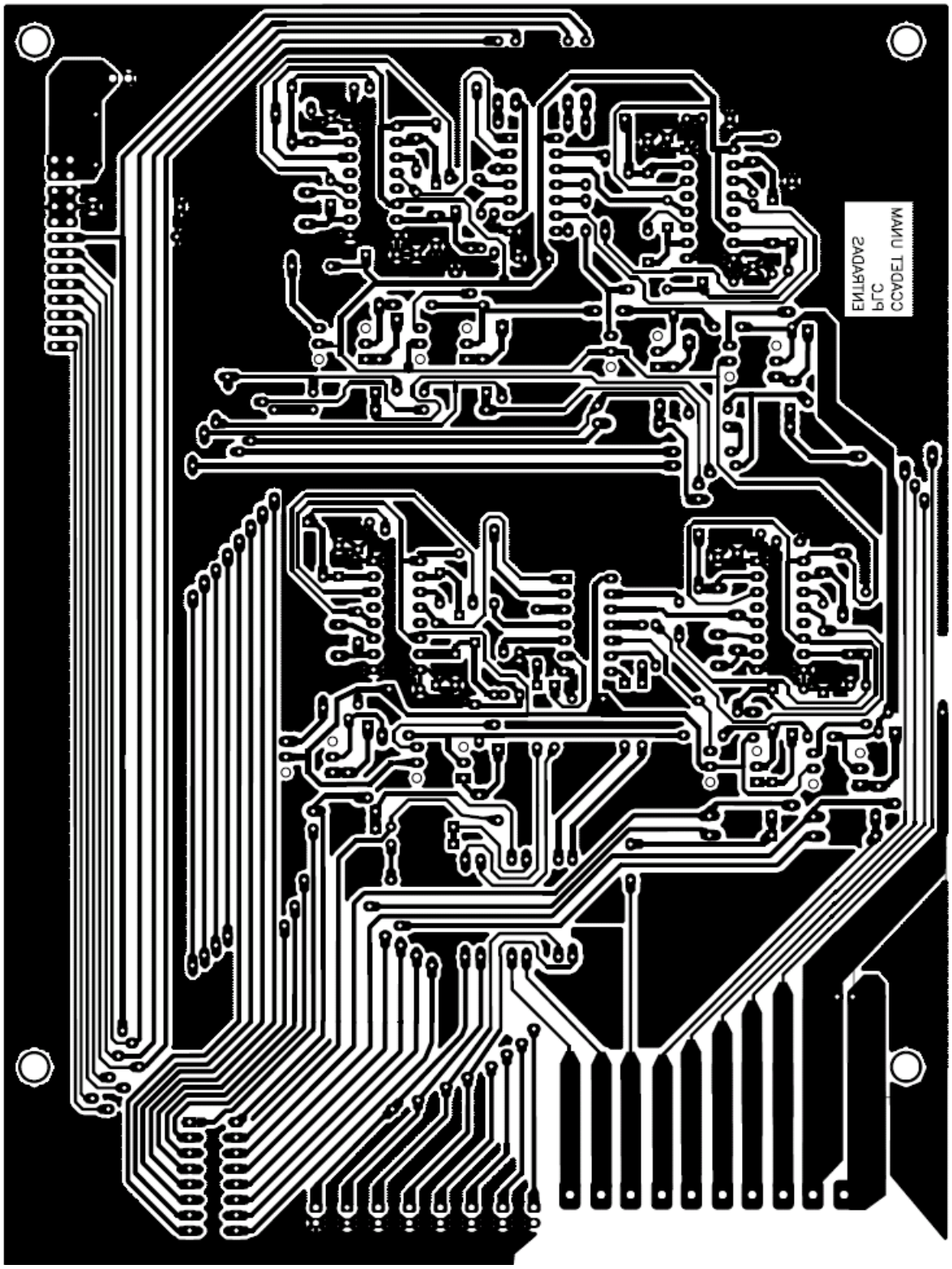

CONN4
IDC26MUNIDO
VCC
GND
BYPASS
C35
10uF
C34
R58
R54
R55
R56
R57
R60
R59
{Value}
D23
D20
D22
D21
D19
D18
D17
U31
1B
2B
3B
4B
5B
6B
7B
GND
1C
2C
3C
4C
5C
6C
7C
COMM
ULN2003A
RELE1
RELE2
RELE3
RELE4
RELE5
RELE6
RELE7
RAS0510
TRTG-02
I/O 12
I/O 9
I/O 8
I/O 11
I/O 7
I/O 6
I/O 10
SALIDA 1
SALIDA 2
SALIDA 3
SALIDA 4
SALIDA 5
SALIDA 6
SALIDA 7



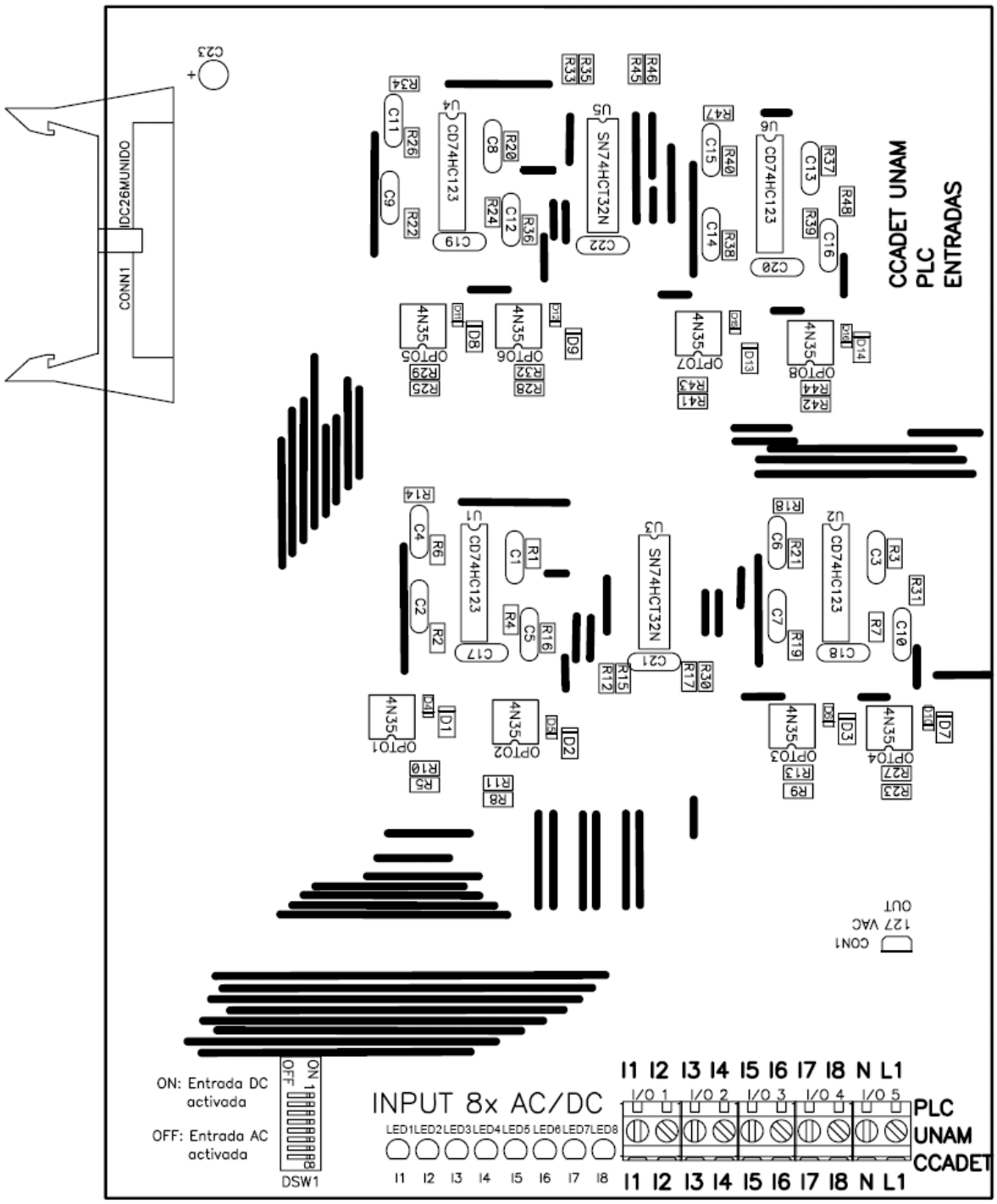
PCB de la etapa CPU y memoria.



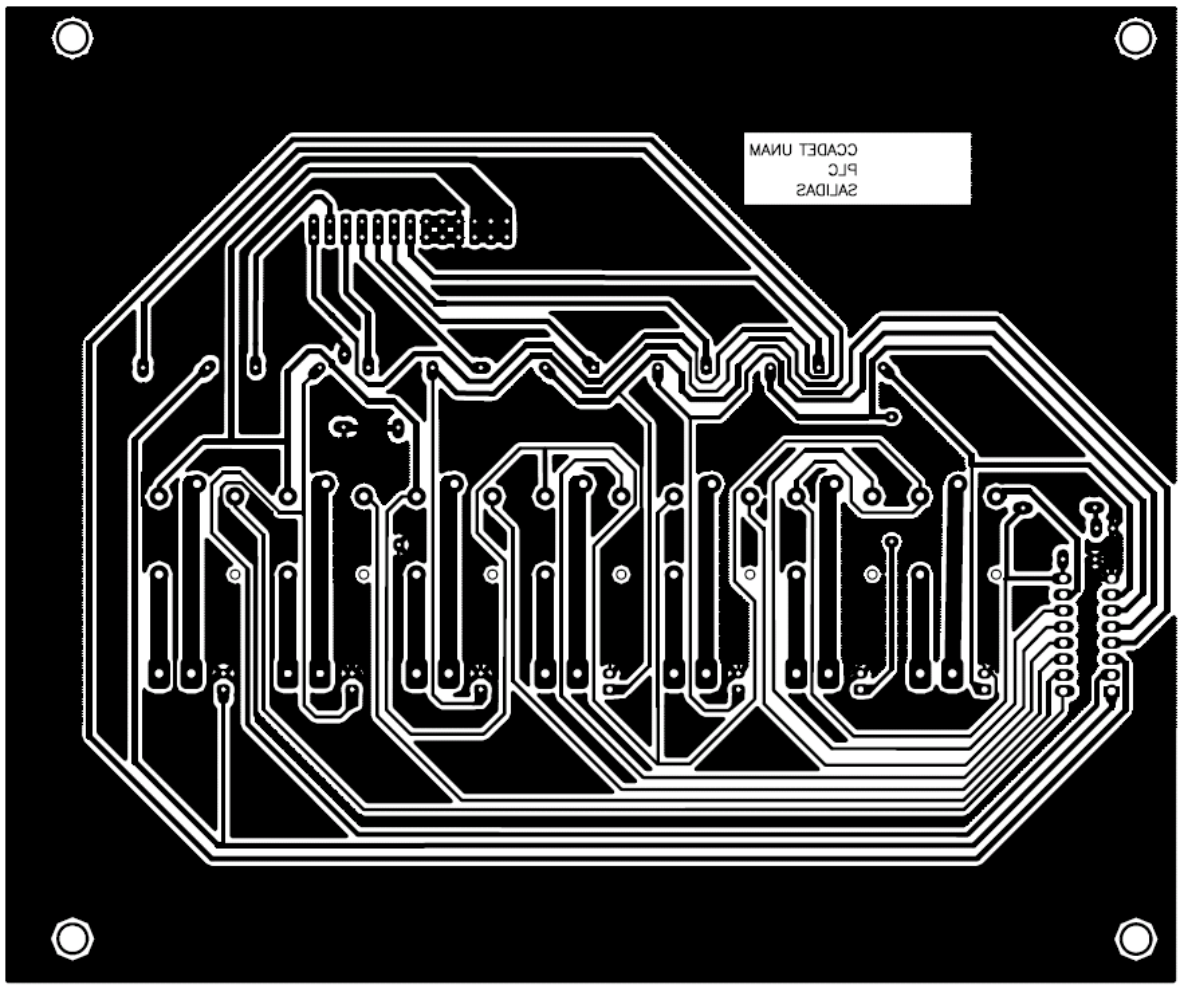
Máscara de componentes de la etapa CPU y memoria.



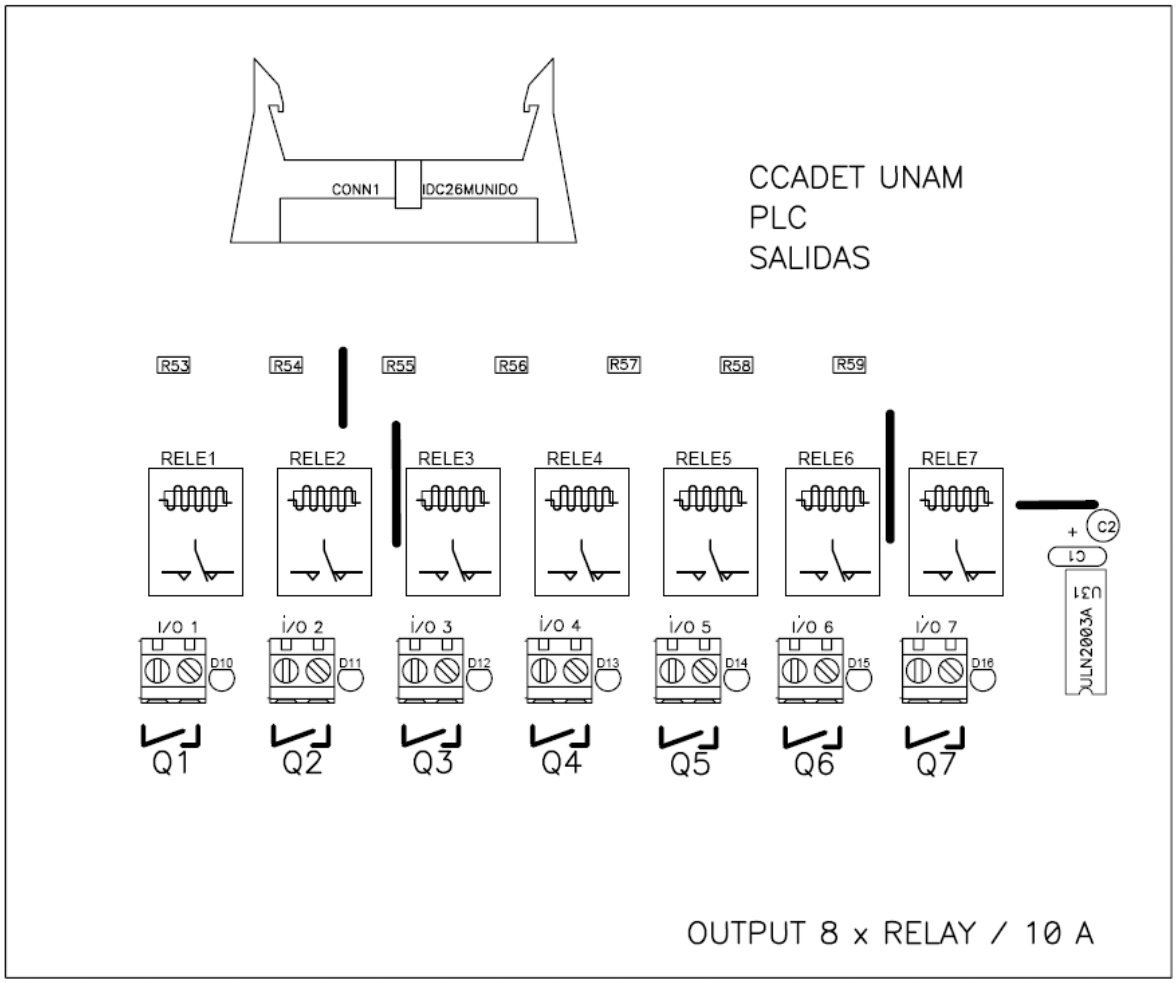
PCB de la etapa de entrada.



Máscara de componentes de la etapa de entrada



PCB de la etapa de salida.



Máscara de componentes de la etapa de salida

REFERENCIAS

[1] Webb, J. (1992). *Programmable logic controllers: principles and applications*. Merrill: Canada.

[2] Lewis, R.W. (1998). *Programming industrial control systems using IEC 1131-3 Revisend edition*. The institution of electrical engineers: United Kingdom.

[3] Maloney T.J. (2003). *Electrónica industrial moderna*. Monroe County Community Collage: USA.

[4] Balcells J., et al. (1992). Interferencias electromagnéticas en sistemas electrónicos. Alfaomega: España.

[5] Coplien, j. (1992). *Advanced C++: Programming Styles and Idioms*. Addison-Wesley Publishing Company: USA.

[6] Stroustrup,B. (1991). *The C++ Programming Language*. Addison-Wesley Publishing Company: USA

[7] Stephen, C. et al. (1989). *Programming in C++*. Prentice-Hall: USA

[8] Balcells, J. Romeral, J.L. (1997). *Autómatas Programables*. MARCOMBO: España.

[9] Couch L.W. (1997). *Sistemas de comunicación digitales y analógicos*. Pearson Educación: Mexico.

[10] Tubbs, S.P. Allen-Bradley. (2005). *Programmable Logic Controller (PLC) Tutorial: Circuits and Programs for Allen-Bradley Micrologix and SLC 500 Programmable Controllers*. Allen-Bradley: USA

[11] Petruzella F.D. (2004). *Programmable Logic Controllers*. McGraw-Hill College: USA

[12] Clements-Jewery, K., Jeffcoat, W. (1995). *Plc Workbook: Programmable Logic Controllers Made Easy*. Prentice Hall: USA

[13] Gorton, R. (1988). *Noise and Distortion in transient waveform recorders: Instrumentation and Measurement Technology Conference*, IMTC-IEEE: USA

[14] Franco, S. (1988). *Design with Operational Amplifiers and Analog Integrated Circuits.* McGraw-Hill: USA

[15] Horowitz, P., Hill, W. (1989). *The Art of Electronics.* Cambridge University Press: USA

[16] Mano M., (1982), *Logic and Computer Design Fundamentals*. Prentice Hall: USA.

[17] Boylestad, R. L., Nashelsky, L., (1997). *Electrónica: Teoría de Circuitos*. Prentice Hall: México

[18] Gardner, N., (2002). *PICmicro MCU C: An Introduction to Programming the Microchip PIC in CCS C*. Bluebird Electronics: USA.

Medios electónicos

[19] Custom Computer Services Incorporated. (2005). "C Compiler Reference Manual November 2005". [electrónico]. C Compiler for Microchip picmicro MCUs. http://www.ccsinfo.com/picc.shtml [Recuperado en mayo 2005].

[20] Custom Computer Services Incorporated. (2005). "PICmicro MCU C: An introduction to programming The Microchip PIC in CCS C". [electrónico]. C Compiler for Microchip picmicro MCUs. http://www.ccsinfo.com/picc.shtml [Recuperado en mayo 2005].

[21] Rios J.F., (2002). "Control y automatización con PLC´s del sistema de transmisores del radio observatorio de Jicamarca" [electrónico]. Radio Observatorio de Jicamarca. http://jro.igp.gob.pe/spanish/ [Recuperado en junio 2005].

[22] Ingeniería de Sistemas y Automática. (2000). "Introducción a los autómatas programables Estructura externa" . [electrónico] Bienvenido al Servidor Web del Área de Ingeniería de Sistemas y Automática ISA-UMH. http://lorca.umh.es/isa/es/ [Recuperado en abril 2005].

[23] Digital electronics corporation. (2005). "Ladder logic instruction list". [electrónico]. Pro-face Human machine interface. http://www.pro-face.com/ [Recuperado en abril 2005].

[24] Siemens AG. (2002). "Lista de instrucciones (AWL) para S7-300 y S7-400 Manual de referencia Referencia 6ES7810-4CA06-8DR0". [electrónico]. Local Partners. http://www.siemens.com/automation/partner [Recuperado en abril 2005].

[25] Rodrigo, A. Musalem, M. (2001). "Programación en escalera". [electrónico]. Introducción a la programación de PLC´s. http://jmtirabasso.galeon.com/cursoplc.htm [Recuperado en mayo 2005].

[26] Schneider electric. (2003). "Descripción de lenguajes Lista y de contactos". [electrónico]. Schneider electric. http://www.schneiderelectric.es/index.html [Recuperado en mayo 2005].

[27] Jimmy Córdova, J. (2000). "¿Que es un autómata programable?". [electrónico]. PCL ó Autómatas programables http://www.geocities.com/cordova_jimmy/plcs.htm [Recuperado en mayo 2005].

[28] PLCopen. (2005). "TC1 - Standards". [electrónico]. Welcome to PLCopen. http://www.plcopen.org/ [Recuperado en mayo 2005].

[29] Jones, D.,Computer Science, University of Iowa. (1994). "Frequently Asked Questions about the DEC PDP-8 computer". [electrónico]. alt.sys.pdp8,alt.answers,news.answers. ftp://rtfm.mit.edu/pub/usenet/alt.sys.pdp8/PDP-8_Frequently_Asked_Questions_(posted_every_other_month) [Recuperado en mayo 2005].

[30] Henry, C. (2004). "infrastructure technologies". [electrónico]. Cal Poly Pomona Co. http://www.csupomona.edu/~hco/Strategy/ [Recuperado en mayo 2005]

Allen-Bradley MicroLogix. (2005)."Architecture and operation".[electrónico]. MicroLogix 1000 Controller. http://www.ab.com/plclogic/micrologix/1000/ [Recuperado en mayo 2005].

[31] Allen-Bradley MicroLogix. (2005)."Input - Output system".[electrónico]. MicroLogix 1000 Controller. http://www.ab.com/plclogic/micrologix/1000/ [Recuperado en mayo 2005].

[32] Allen-Bradley MicroLogix. (2005)."Basic Relay Instructions".[electrónico]. MicroLogix 1000 Controller. http://www.ab.com/plclogic/micrologix/1000/ [Recuperado en mayo 2005].

[33] Allen-Bradley MicroLogix. (2005)."comparison instructions and flow control instructions".[electrónico]. MicroLogix 1000 Controller. http://www.ab.com/plclogic/micrologix/1000/ [Recuperado en mayo 2005].

[34] Allen-Bradley MicroLogix. (2005)."Timing, counting, and data-handling instructions".[electrónico]. MicroLogix 1000 Controller. http://www.ab.com/plclogic/micrologix/1000/ [Recuperado en mayo 2005].

[35] NEMA. (2004). "2004 Electrical Standars & Product guide". [electrónico]. National Electrical Manufacturers Association. http://www.nema.org/ [Recuperado en mayo 2005].

[36] Festo. (2005). "Programa Seminarios Festo 2005". [electrónico]. Festo Didactic. http://www.festo-didactic.com/didactic/?nation=MX&lang=es&data=home.html
[Recuperado en abril 2005].
[37] Vallejo, H.D., TodoPic. (2005). "PLC Los controladores lógicos programables".[electrónico]. Saber electronica No. 166 http://www.todopic.com.ar/
[Recuperado en mayo 2005].
[38] National Instruments. (2005). "¿Cómo puedo comunicar LabVIEW con un PLC?". [electrónico]. Módulo de Control Supervisorio y Registro de Datos para LabVIEW. http://www.ni.com/niglobal/americas.htm [Recuperado en mayo 2005].
[39] Rodríguez, J.M. (2002). "PIC-PLC II".[electrónico]. Proyecto de automata programable de 8 E/S. http://inicia.es/de/juanmarod/main.htm [Recuperado en mayo 2005].
[40] SCM International. (2005). "MINI14PLCs". [electrónico]. MINI14PLCs. http://scmstore.com/plcs/MINI14PLCs/
[Recuperado en mayo 2005].
[41] SCM International. (2005). "PLCs de bajo costo, alta perfomance y flexibilidad". [electrónico]. MINI14PLCs. http://scmstore.com/plcs/default.asp [Recuperado en mayo 2005].
[42] Strangio, C.E., CAMI Research Inc. (2005). "The RS232 standard". [electrónico]. The RS232 standard, a tutorial with signals names and definitions. http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html [Recuperado en junio 2005].
[43] ARC Electronics. (2005). "RS485 Data Interface". [electrónico]. A tutorial. http://www.arcelect.com/RS485_info_Tutorial.htm [Recuperado en mayo 2005].
[44] Coquet, E. (2005). "El bus I2C". [electrónico]. Notas y Artículos Técnicos. http://www.comunidadelectronicos.com/articulos/i2c.htm [Recuperado en mayo 2005].
[45] Matic, N. MikroElektronika. (2003). "Introduction to PLC controllers". [electrónico]. Introduction to PLC controllers on-line FREE!.
http://www.mikroelektronika.co.yu/english/product/books/PLCbook/plcbook.htm
[Recuperado en mayo 2005].
[46] Siemens AG. (2003). "LOGO! Manual,Manual edición 06/2003 Ref. A5E00228594-01". [electrónico]. Micro automation.

http://www.automation.siemens.com/logo/index_76.html [Recuperado en mayo 2005].

[47] Siemens AG. (2000). "Aplicaciones en la industria y sector terciario y residencial Ref. 6ZB5310-0JA04-0BA0". [electrónico]. Micro automation. http://www.automation.siemens.com/logo/index_00.html [Recuperado en mayo 2005].

[48] Texas Instruments. (1999). "Designing With the SN74AHC123A and SN74AHCT123A Ref. SCLA014". [electrónico]. Application Notes. http://focus.ti.com/docs/apps/catalog/resources/appnoteabstract.jhtml?abstractName=scla014 [Recuperado en junio 2005].

[49] Fairchild Semiconductor. (2005). "Definitions". [electrónico]. Optocoupler Products Specifications. http://www.fairchildsemi.com/index.html [Recuperado en mayo 2005].

[50] Fairchild Semiconductor. (2005). "4N25/4N26/4N27/4N28/4N35/4N36/4N37/H11A1/H11A2/H11A3/H11A4/H11A5 General Purpose 6-pin Phototransistor Optocouplers". [electrónico]. Product tree - Optocouplers. http://www.fairchildsemi.com/products/opto/oi/index.html [Recuperado en mayo 2005].

[51] Maxim Integrated Products. (2004). "+5V-Powered, Multichannel RS-232 Drivers/Receivers". [electrónico]. RS-232 Line Driver/Receivers. http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2008 [Recuperado en mayo 2005].

[52] Texas Instruments. (2003). "TUSB3410, TUSB3410I USB to serial port controller Ref. SLLS519C". [electrónico]. USB Peripherals . http://focus.ti.com/docs/prod/folders/print/tusb3410.html [Recuperado en junio 2005].

[53] Microchip Technology Inc. (2002). "MPLAB IDE simulator, editor user´s guide Ref.DS51025E" [electrónico]. Development Software http://www.microchip.com/ [Recuperado en abril 2005].

[54] Microchip Technology Inc. (2001). "PIC16F87X Data Sheet Ref.DS30292C" [electrónico]. PICmicro Microcontrollers. http://www.microchip.com/ [Recuperado en abril 2005].

[55] Microchip Technology Inc. (2002). "24AA32A/24LC32A Data Sheet Ref.DS21713B" [electrónico]. 32K I2C Serial EEPROM. http://www.microchip.com/ [Recuperado en abril 2005].

[56] Microchip Technology Inc. (2005). "MPLAB ICD 2 User´s guide Ref.DS5133B" [electrónico]. Development Tools. http://www.microchip.com/ [Recuperado en abril 2005].

[57] Microchip Technology Inc. (1997). "PICmicro Mid-Range MCU Family Reference Manual Ref.DS33023A" [electrónico]. PICmicro Microcontrollers. http://www.microchip.com/ [Recuperado en abril 2005].

[58] Sun hold Electric inc. (2005). "RAS Series Datasheet". [electrónico]. Relays. http://www.sunhold.com/ras2.html [Recuperado en junio 2005].

[59] Tyco International Ltd. (2005). "Relays application notes, Suppressing Relay Coil Transients ". [electrónico]. Relays.
http://relays.tycoelectronics.com/kilovac/appnotes/transients.stm
[Recuperado en junio 2005].

[60] Philips semiconductors. (2003). "74HC32; 74HCT32 Quad 2-input OR gate Ref. 9397 750 12488" [electrónico]. Product Information. http://www.semiconductors.philips.com/pip/74HC32.html [Recuperado en mayo 2005].

[61] American Standard Code for Information Interchange. (1979). "ASCII Table and description" [electrónico]. ASCII Table and description. http://www.lookuptables.com/ [Recuperado en junio 2005].

Algunos sitios de interés

http://www.pic-c.com

http://www.piclist.com

http://www.pics.net

http://www.sitrain.com

http://www.lookuptables.com/ codigo ASCCI

www.metexcorporation.com

www.oakton.ca

www.myronl.ca

www.lakewoodinstruments.ca

www.walchem.ca

www.osmonics.ca