19/11/2015

18/12/17

ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"Simulación de un Sistema de Control tipo PLC para el control de cargas monofásicas, diseño e implementación de una Interfase por Software, diseño del sistema de control de una Interfase de potencia".

TESIS DE GRADO

Previa a la Obtención del Título de:

INGENIERO EN COMPUTACION

Presentado por:

Pablo Salvatierra Villavicencio

Guayaquil – Ecuador

1999

AGRADECIMIENTO

Al **ING.CARLOS DEL POZO CAZAR** por su valiosa ayuda disposición y colaboración para la realización de esta tesis.

DEDICATORIA

A las personas a quienes más quiero: mis Padres Luis y Yolanda, a mis hermanos Luis, Alexandra, José, y con mucho cariño a la bebé Karen Adriana . Gracias Padres por ser mi guía espiritual, de todas sus enseñanzas viviré eternamente agradecido.

Subdecano de la Facultad de Ingeniería en Electricidad y Computación
Ing. Armando Altamirano

Director de Tesis
Ing. Carlos Del Pozo C

Miembro del Tribunal
Ing. Guido Caicedo

Miembro del Tribunal
Ing. Hugo Villavicencio

DECLARACION EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestos en esta Tesis, me corresponden exclusivamente; y, el patrimonio de la misma a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL"

(Reglamento de Exámenes y Títulos Profesionales de la ESPOL)

PABLO SALVATIERRA VILLAVICENCIO

RESUMEN

El trabajo de investigación realizado en esta tesis tiene por finalidad dar una visión diferente a la perspectiva cognoscitiva de control, se refiere básicamente al diseño de los sistemas de control automatizado por computadoras, estos siempre los hemos visto como sistemas muy complejos, difíciles de comprender y complicados en su funcionalidad.

OBJETIVOS :

Nuestro objetivo de investigación se centra en diseñar un sistema, que provisto de un computador, el puerto centronics, así como de la apropiada *interfase por software*, y de la *interfase electrónica* de potencia, permita en su conjunto obtener un *sistema automático de control* para el manejo de cargas monofásicas de potencia.

El sistema de mando de control es el computador, del cual vamos a aprovechar el puerto centronics que dispone un PC, y por medio de las señales de control que proporciona los pines del puerto centronics del computador podemos controlar una carga monofásica o bifásica. El manejo apropiado de los puertos externos de un PC es componente fundamental para controlar cualquier periférico, este diseño propuesto aquí tiene muchas aplicaciones reales, funcionales y prácticas; no solamente es útil para el control de interfases de potencia, sino también para aplicaciones electrónicas de alta precisión en la automatización de las industrias; como es el control de estados sobre dispositivos sensores.

La excelente versatilidad del lenguaje C, permite realizar una interfase que sea amigable con el usuario, esta interfase de tipo visual se muestra en el monitor del operador indicando el estado o estados del circuito de carga que se dispone en ese instante, estas características visuales hacen que el sistema sea muy fácil de interactuar con el usuario,

no se necesita tener un previo entrenamiento para su manejo, los parámetros son ingresados en las ventanas respectivas por el usuario, y una vez finalizado el ingreso de los mismos entonces se ejecuta la acción de control automatizada sobre la carga.

Como podemos darnos cuenta, estamos enfocando dos ramas de la ingeniería que son: la ELECTRONICA y la COMPUTACION me permito sugerir las mejores alternativas de diseño en ambas ramas de la ingeniería para la obtención de óptimos resultados funcionales; en un sistema de control automático de cargas.

El diseño de la interfase de software lo dispongo bajo el lenguaje más funcional y apropiado que existe; como lo es el lenguaje C, y en lo referente a la interfase electrónica igualmente se justifica la circuitería electrónica que gobierna la interfase de potencia y la utilización de dispositivos de alto rendimiento. El diseño de la interfase electrónica de potencia tiene como objetivo operar de manera eficiente el circuito de carga, así como disponer de una manera segura el puerto centroncis del sistema principal (CPU) con la interfase aisladora.

En el área de electrónica hemos estado acostumbrados a diseñar sistemas de control basados en microprocesadores y por supuesto hay que tener conocimiento del lenguaje de programación Asembler; característica principal para lograr éxito en nuestros objetivos, existe entonces una barrera limitadora para las personas que poco o nada conozcan en el área de la electrónica, pero si conocen hoy en día la funcionalidad y operatibilidad de un computador personal, los inconvenientes para el desconocedor de la electrónica sería por ejemplo, reconocer cual es el uso apropiado de cada uno de los pines de un microprocesador, y como lograr su disposición de eficiencia. Es aquí que nace la idea de diseñar un sistema automático de control de cargas, que involucre a un computador como *hardware*, la interfase visual de control como *software*, el *puerto*

centronics del PC y la circuitería electrónica *optoaisladora* como interfase externa de salida, la interfase electrónica de *potencia*, y la carga.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

INDICE DE ABREVIATURAS

CPU	Unidad de Proceso Central
IECO	Interfase Electrónica de Control Optoaisladora
IP	Interfase de Prueba
PC	Computador Personal
PLC	Controlador de Lógica Programable
RAM	Memoria de Acceso Aleatorio
SAC	Sistema Automático de Control de Carga
STL	Lista de Instrucciones

INTRODUCCION

Podemos asegurar que el ser humano representa al sistema de control más perfecto que se haya diseñado en la naturaleza, otros sistemas de control semi-perfectos son los animales y demás seres vivos existentes. Mencionamos semi-perfectos porque no poseen la cualidad más importante que tiene el hombre y esta es la inteligencia en confinación con el razocinio. Las máquinas que el hombre ha desarrollado para elevar su nivel de vida, usan medios del tipo electrónico para implementar un sistema de control automático. El control se lo puede describir como la generación de una señal de salida en respuesta a una señal de entrada al sistema.

Este proceso puede ser en *"lazo abierto"* o en *"lazo cerrado"*. Apagar un calentador de agua de manera automática durante el día y encenderlo en horas de la noche, es un ejemplo de un control en *lazo abierto*. Apagar el calentador automáticamente cuando la temperatura del agua supere los 60° C grados sería entonces un control en *lazo cerrado*. El propósito de un sistema de control electrónico es de manejar con estabilidad un fenómeno físico, el cual se debe estar midiendo constantemente. Podemos mencionar a los fenómenos físicos como: temperatura, velocidad, posición, humedad, deformación, nivel, presión, intensidad luminosa, flujo, desplazamiento, voltaje, corriente y potencia.

La popularización del microprocesador produjo un cambio notable en los diseños de los sistemas de control. Un ***microcomputador*** puede implementar *cualquier ley de control* utilizando una arquitectura estándar (CPU, RAM, ROM, I/O) y acompañado de un programa almacenado (Software). Toda labor de control implementada por circuitos procesadores se lo puede desarrollar de una manera más eficiente utilizando un microcomputador o PC, el *sistema de control clásico* cambia por una configuración que involucra al PC. Para procesar un sistema implementado con microcomputadores se

necesita disponer solamente de variables del tipo digital; que toman dos valores definidos: nivel alto y nivel bajo. Las señales de tipo analógica varía su amplitud incrementalmente, a lo largo de la base de tiempo.

Las señales digitales, en cambio son de amplitud uniforme y están representadas por dos únicos valores posibles; utilizando lógica positiva representamos a un nivel alto de voltaje como "1" y a un nivel bajo de voltaje como "0". El microcomputador necesita procesar la información recogida por los sensores, es necesario por lo tanto realizar una *conversión* de una *señal análoga* a una *señal de tipo digital*. Los circuitos que realizan esta función se los denominan convertidores Analógico - Digital. De igual forma, la salida de característica digital que genera el microcomputador debe ser convertida a señal análoga esto es por medio de un circuito electrónico denominado convertidor Digital - Analógico.

Algunos sensores son de naturaleza digital por lo tanto no requieren de estos circuitos convertidores; un microinterruptor, un interruptor de fin de carrera, son ejemplos de entradas de naturaleza digital que no necesitan conversión. De igual forma, activar o desactivar un relé o un motor son salidas de naturaleza digital. Los microcomputadores modernos que se conocen hoy en día, utilizan microprocesadores Pentium 300 Mhz, MMX, y ofrecen una velocidad de procesamiento muy alta, estabilidad y almacenamiento masivo de datos. Estas características proporcionan extraordinarias capacidades matemáticas de análisis, despliegue gráfico, generación de reportes, control y el más importante de todos *"comunicaciones"* por medio de los *puertos I O*.

El computador personal, es la máquina que tiene mayor aceptación para el diseño de los sistemas de control y para el diseño de los sistemas modernos de adquisición de datos.

Existen varias formas de acoplar un sistema de control a un computador personal o PC, mencionaremos tres acoplamientos característicos:

A.- Uso del puerto Centronics o Paralelo *DB-25F* de la impresora.

B.- Conexión directa a la barra del bus interno del microprocesador, (barra de datos, barra de direcciones, barra de control de estado).

C.- Por medio del puerto de comunicación serial *RS-232*, *RS-422*, *RS-423*, o la más moderna interfase serial asincrónica conocida hoy en día como *RS-485*.

CAPITULO I

AUTOMATAS PROGRAMABLES

1.1 RESEÑA HISTORICA.-

Los computadores requeridos en la industria no solamente se utilizan para procesar información de tipo administrativo; también son utilizados para controlar directamente los procesos de fabricación. La automatización industrial dio sus primeros pasos en el desarrollo de sistemas automáticos de control apoyado en la lógica cableada; hoy avanza guiada de la mano de la lógica programable en computadores y en *PLC*. Los *PLC* son grandes exponentes de esta técnica de mando por programa en la industria moderna.

El desafío constante que toda industria moderna tiene planificada para ser competitiva ha sido el motor impulsor del desarrollo de las nuevas tecnologías; característica fundamental para lograr una mayor productividad y beneficios.

Muchos de los procesos de fabricación se realizan en ambientes que son nocivos para la salud, como son los gases tóxicos (por ejemplo: amoniaco), ruidos nocivos, temperaturas muy altas o muy bajas y muchos otros procesos requieren que un determinado parámetro como: temperatura, presión, relación aire/combustible, deba mantenerse constante para garantizar una buena calidad en el producto terminado.

Todo esto lleva a pensar en la posibilidad de desarrollar ciertas tareas repetitivas, peligrosas o de precisión a través de dispositivos especializados que reemplazarán al hombre. De lo mencionado nace entonces el concepto de máquina y con ella la **AUTOMATIZACION.**

La automatización como hoy la concebimos, surgió de la utilización de relés, temporizadores, contadores, pulsadores, válvulas, levas y otros elementos basados en los principios electromecánicos. Con estos se puede implementar cualquier aplicación particular; claro está teniendo previamente un diseño adecuado. Al aumentar la complejidad de las tareas y rutinas de un proceso a realizar, los armarios de maniobra que contienen a la circuitería electrónica se hacen cada vez más grandes, y es engorroso revisarlos o tomar una lectura, la probabilidad de avería aumenta y el mantenimiento se hacía cada vez más difícil. La aparición de los semiconductores como lo son los transistores, y los circuitos integrados electrónicos posteriormente, conllevó consigo a la sustitución de los relés por dispositivos electrónicos de funciones específicas.

Los sistemas de automatización que emplean las tarjetas electrónicas basadas en compuertas lógicas físicas (diseño de lógica programable PLA), se caracterizan por su mayor velocidad de respuesta, menor consumo de potencia, menor espacio, entre otras cualidades, pero aún conservan el problema de la flexibilidad de los sistemas construidos con relés debido a que su lógica es alambrada físicamente. Para el año de 1960 varios computadores fueron instalados en plantas industriales de producción de vidrio estas realizaban las tareas de control de los hornos. Su uso evidenció las ventajas y bondades de los sistemas de *"mando por programas"* para aplicaciones en el área industrial, como lo es la facilidad de diseño, mantenimiento, y flexibilidad, esto nos conlleva a marcar el inicio de la era de la automatización usando lógica "cableada" por programa (*software*) [INTE96].

La aparición de los microprocesadores identifica el inicio de la era de los microcomputadores, se genera entonces la migración de estos equipos hacia la industria con el objetivo de realizar tareas de automatización y control, permitiendo de esta forma

lograr un mayor grado de desarrollo de soluciones a problemas muy complejos tales como: reducción de costos, ahorros en la instalación y aumento de la productividad.

Inicialmente, la utilización de estos equipos evidenció que su diseño originalmente conceptualizado; era inapropiado para operar en ambientes de trabajo de tipo industrial. Los problemas más frecuentemente hallados fueron:

1. la incompatibilidad de los sistemas operativos no orientados hacia tareas de control y de mando.

2. la necesidad de diseñar tarjetas y circuitos electrónicos que posean un bajo nivel de rechazo con las interferencias electromagnéticas, características que hacen que los sistemas sean muy sensibles a las perturbaciones y fallas frecuentes.

3. circuitos inadecuados para el manejo de señales de entrada - salida y poco o nada desarrollo de *programas* y dispositivos de apoyo orientados a la automatización y control.

4. denotamos la falta de estandarización de los lenguajes de programación así como también de los sistemas de comunicaciones entre los microcomputadores.

5. la necesidad de utilizar el *lenguaje ensamblador;* para poder trabajar a tiempo real de operación en los procesos de alta velocidad de respuesta.

6. la necesidad de requerimientos de una adecuada atmósfera controlada en los salones donde operan los microcomputadores y por supuesto el desarrollo de programas de automatización implementados sobre sistemas operativos para este propósito.

generaron muchos inconvenientes, los mismos que desalentaron su uso en aplicaciones industriales.

Pero no todo eran inconvenientes y problemas de ajustes, también existían las ventajas de los mandos conocidos como *"programados"* estas eran relativas como para generalizar su utilización. Para fines de la década de los 70's, aparecen en el mercado los primeros *"Controladores Programables"* los mismos que tenían por misión reemplazar los paneles de relés y estaban basados en microprocesadores.

Su confección de diseño electrónico y mecánico eran de características muy robustas, de fácil instalación y adecuados para operar en áreas de trabajo de la industria generalmente en un ambiente electromagnético hostil. Todos estos nuevos equipos inicialmente utilizaron los *"esquemas de contactos"* (*Ladder Logic*) para su programación, por su facilidad de manejo por parte de los operadores o usuarios quienes eran los encargados del mantenimiento y diseño en las fábricas.

Posteriormente vinieron los nuevos diseños de los ya denominados *"Controladores Lógicos Programables" PLC* (son las iniciales que se conservan de su denominación en inglés: *Programable Logic Controller*) tienen las características que se pueden programar con *"lógica de contactos"* e incluyen nuevos lenguajes de programación diferentes al ensamblador.

En su mayoría todos los *PLC* incluyen un modo de programación denominada *"Lista de Instrucciones" (STL)* la cual debe cumplir con dos condiciones importantes.

- rápido.
- intuitivo.

Los equipos que se conocen hoy en dia presentan nuevos modos de programación conocimos como "esquemas de funciones" y otros de tipo gráfico de funciones secuenciales (por ejemplo *GRACET de Telemecanique*, el *PSM de Adatek* y el *GRAFH 5 de Siemens*) todas estas nuevas funciones lo que buscan es facilitar el trabajo de programación al operador, presenta la ventaja en poder digitar una instrucción; utilizando un computador personal (*PC*) por supuesto con la ayuda de un *software* que sea apropiado y que sea transmitido al *PLC* por la vía del puerto de comunicaciones paralelo.

En sus inicios los diseños de los *PLC* contenían un pequeño set o conjunto de instrucciones los mismos que han evolucionado hasta obtener un rango medio de instrucciones muy potentes, tales como:

- Operaciones aritméticas: suma, resta, multiplicación, división, raíz cuadrada y comparación.
- Operaciones de transferencia: de bits, bytes y tablas en memoria y unidades periféricas.
- Manipulación de bits: puesta a "1", borrado, chequeo de estado de bit.
- Desplazamiento de registros.
- Manejo de subrutinas, otros.

Paralelamente, todas estas innovaciones están acompañadas de los nuevos sistemas operativos, las tarjetas y circuitos electronicos también han evolucionado para mejorar sus condiciones de operación, el alcance y la programación de los *PLC*.

A nivel de los sistemas operativos cabe destacar las rutinas de alarma y de señalización para los circuitos de seguridad y de fallas, los mismos que se implementan parte en *software* y otra en *hardware*.

A nivel de los circuitos electrónicos, lo más relevante es el desplazamiento de los microcomputadores y en su lugar, el uso de los microcontroladores.

El incremento de la escala de integración para la construcción de los nuevos circuitos integrados electrónicos miniaturizados ha permitido diseñar y construir en un solo *chip* (integrado) un sistema de control compuesto por un microprocesador, contadores, circuitos generadores de bases de tiempo, circuitos de manejo de datos serie (comunicación serial), los conocidos puertos paralelos de entrada - salida, en general los circuitos electrónicos consistían que antes eran independientes ahora forman parte de un conjunto. Con todos estos elementos, integrados en un sólo integrado, toman el nombre genérico de MICROCONTROLADOR.

Así como los microprocesadores han evolucionado desde el 4004 hasta los más avanzados como el *Pentium 300 Mhz (arquitectura RISC)*, y el *MMX* de la familia *INTEL*, los microcontroladores han adquirido un desarrollo permanente de acuerdo a las exigencias del mercado cada vez mayores y al aumento de la escala de integración.

Hago referencia a microcontroladores característicos como:

- **8051 de INTEL**.
- **68HC11 de MOTOROLA**.
- **PIC de Microchip**.

Una de las cualidades más notables de los microcontroladores es la reducción de la sensibilidad a las interferencias de tipo electromagnéticas con respecto a los microprocesadores, ventaja que permite su utilización en ambientes industriales con menor probabilidad de que sean afectados por perturbaciones ajenas al ambiente.

En los últimos años, el desarrollo de los Autómatas programables ha estado ligado al desarrollo de nuevos microcontroladores y las nuevas versiones de los *PLC* están en la necesidad de implementarse microcontroladores cada vez más sofisticados y potentes en sus funciones operativas [INTE 96].

Para inicios de la década de los 80's se presenta en el mercado Autómatas de diferentes características y precios adaptables a las distintas gamas de necesidades de operatibilidad.

En el área industrial, sus prestaciones y servicios se tornan muy confiables y común por las características de su diseño especial para trabajar en esta clase de ambientes, sin las restricciones que presentan los computadores [INTE96].

A finales de los 80's, los fabricantes de *PLC* centran sus esfuerzos en mejorar sus diseños con respecto a la compatibilidad electromagnética *EMC*, de modo que les permita operar con sus equipos en ambientes eléctricamente y magnéticamente contaminados; sin introducir perturbaciones indeseables en su funcionamiento continuo.

Se han dictado normas para los fabricantes de aparatos y máquinas eléctricas con el objeto de reducir la emisión de interferencias a niveles muy bajos, de esta manera la solución a este problema se está desarrollando desde el punto de vista de las fuentes y los receptores de las interferencias [INTE96]

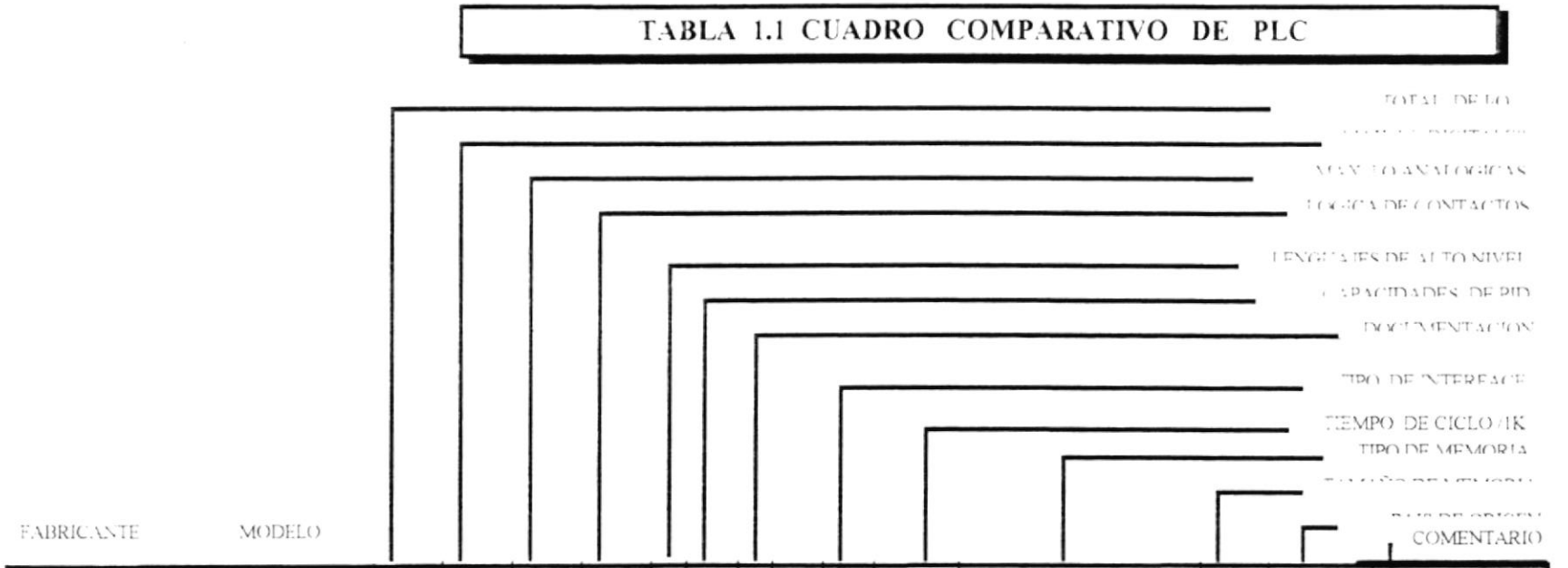
En la actualidad los autómatas programables se encuentran a medio camino entre los microcomputadores y los computadores de proceso y sus aplicaciones se extienden a procesos de maniobra de máquinas e instalaciones, señalización, sistemas de alarma y control de procesos.

Desde el punto de vista técnico puede afirmarse que el *PLC* simplificó la industria y la dotó de modernas herramientas para su desarrollo.

Actualmente se encuentra en el mercado equipos muy seguros en las gamas de baja, media y alta potencia, suministrados por distintos fabricantes.

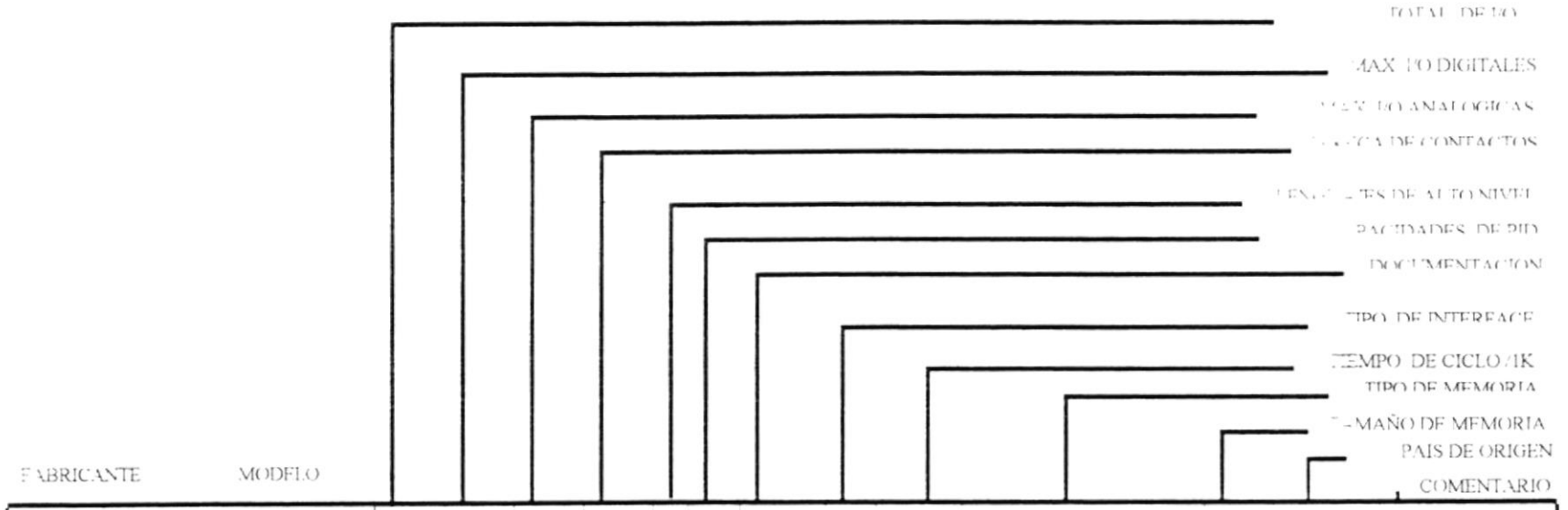
En la *Tabla 1.1* se muestra un cuadro comparativo de algunos de los *PLC's* más utilizados en bajas y medias potencias en aplicaciones industriales.

TABLA 1.1 CUADRO COMPARATIVO DE PLC



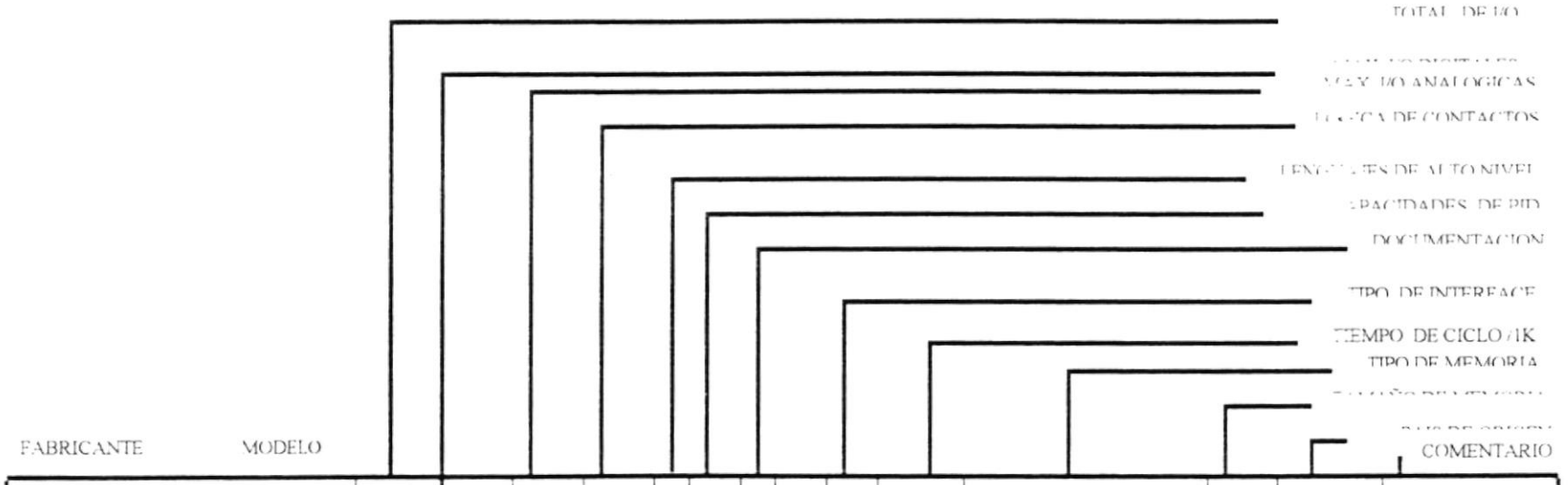
FABRICANTE	MODELO	TOTAL DE E/S	E/S DIGITALES	E/S ANALOGICAS	LOGICA DE CONTACTOS	LENGUAJES DE ALTO NIVEL	CAPACIDADES DE PID	DOCUMENTACION	TIPO DE INTERFACE	TIEMPO DE CICLO /1K	TIPO DE MEMORIA	TAMAÑO DE MEMORIA	PAIS DE ORIGEN	COMENTARIO
ALLAN - BRADLEY (Millwaukee,WI)	PLC -3	8192	8192	4096	Y	Y	Y	Y	Y	A	2.5 ms	RAM EDC	2M	USA	
	PLC - 3/10	4098	4098	2048	Y	Y	Y	Y	Y A	2.5ms	RAM EDC	64K			
	SLC 100	112	112	24	Y			Y	Y	J	*	EEPROM,CMOS RAM	885	USA,Japon	*15ms 500words
	SLC 150	112	112	24	Y			Y	Y	J	*	EEPROM CMOS RAM	1200	USA,Japon	* 2ms 500words
	PLC - 2/02	512	512	256	Y	Y	Y	Y	Y	A	12.5ms	RAM, EEPROM	1K	USA	
	PLC - 2/16	512	512	256	Y	Y	Y	Y	Y	A	12.5ms	RAM,EEPROM	3K	USA	
	PLC - 2/17	512	512	256	Y	Y	Y	Y	Y	A	12.5ms	RAM,EEPROM	6K	USA	
	PLC - 2/30	2688	2688	896	Y	Y	Y	Y	Y	A	5ms	RAM	16K	USA	
BRITISH BROWN (Telford, England)	K200	128	128			Y		Y	Y		5ms	EPROM	2K	Germany	
	Procontic B	1024	128	256	Y	Y	Y	Y	Y	B	2.5ms	RAM,EPROM	512	Germany	
	DIP800	4000	3000	1000		Y	Y	Y	Y	B	1ms	RAM,EPROM	inf.	U.K	
	p214	1164	1100*	64*		Y	Y	Y	Y	B,I	2ms	RAM,EPROM	700K	U.K	*por estacion
FOXBORO CO (Foxboro,MA)	3PC - 3A	64*	512	128	Y		Y	Y	Y	D	1ms	EEPROM,RAM	32K	USA	* Modulos
	3PC - 4A	256*	2048	128		Y	Y	Y	Y	D	1ms	EEPROM,RAM	48K	USA	* Modulos

TABLA 1.1 CUADRO COMPARATIVO DE PLC



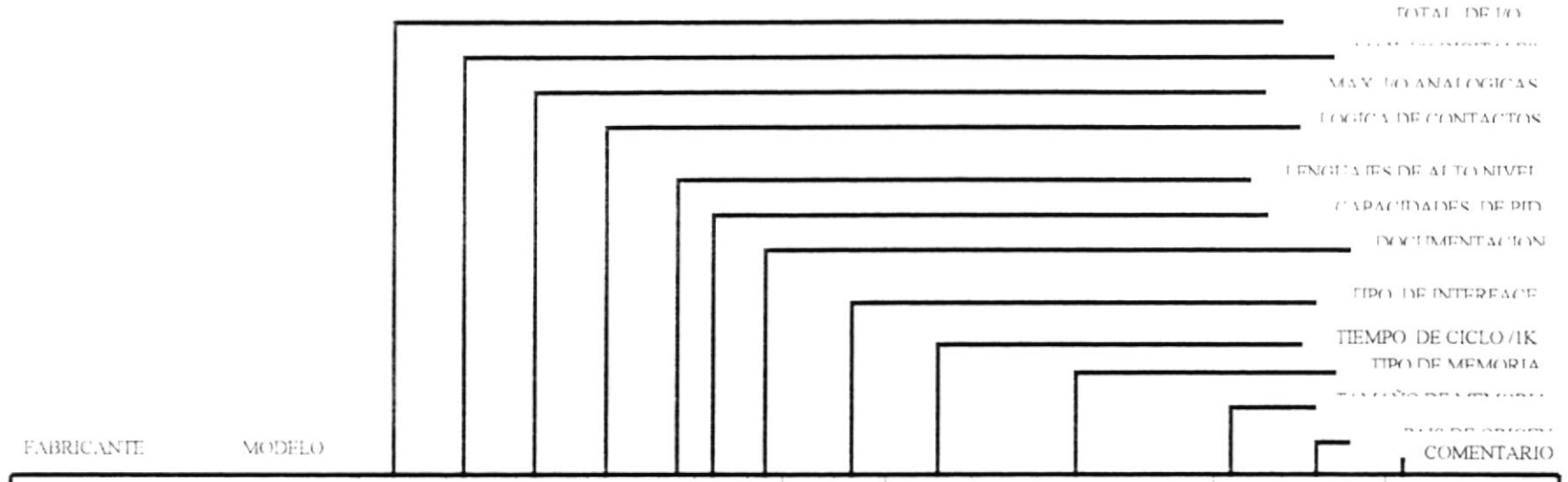
FABRICANTE	MODELO	TOTAL DE I/O	MAX I/O DIGITALES	MAX I/O ANALOGICAS	CANTIDAD DE CONTACTOS	LENGUAJES DE ALTO NIVEL	CAPACIDADES DE PID	DOCUMENTACION	TIPO DE INTERFACE	TIEMPO DE CICLO/1K	TIPO DE MEMORIA	TAMAÑO DE MEMORIA	PAIS DE ORIGEN	COMENTARIO
HONEYWELL INDUST. CONTROLS DIV. MICRO PAC	S20 - 30	512	512		Y	Y		Y	Y	B	10ms	CMOS,EPROM	4K	USA	
	S20 - 15	512	512	512	Y	Y	Y	Y	Y	B	10ms	CMOS,EPROM	4K	USA	
	S20 - 20	512	512	512	Y	Y	Y	Y	Y	B	3ms	CMOS	8K	USA	

TABLA 1.1 CUADRO COMPARATIVO DE PLC



FABRICANTE	MODELO	TOTAL DE I/O	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	TIPO DE INTERFACE	TIEMPO DE CICLO /1K	TIPO DE MEMORIA	[illegible]	[illegible]	COMENTARIO
MITSUBISHI ELCT SALES AMERICA (No Prospect IL)	A1CPU	256			Y	Y	Y	Y	Y	B	1.25ms	RAM,EPROM,EEPROM	6K	Japon	
	A2CPU	512			Y	Y	Y	Y	Y	B	1.25ms	RAM,EPROM	14K	Japon	
	A3CPU	2048			Y	Y	Y	Y	Y	B	1.25ms	RAM,EPROM	60K	Japon	
	F1-F2	32	32		Y			Y	Y	J	12ms	RAM,EPROM,EEPROM	1K	Japon	
	F1-20M	40	40	6	Y			Y	Y	J	12ms	RAM, EPROM,EEPROM	1K	Japon	
OMROM ELECTRONICS (Schaunburg,IL)	S6	64	64					Y	Y		10ms	RAM,EPROM	1024	Japon	
	C20	140	140		Y			Y	Y	H	10ms	RAM,EPROM	1194	Japon	
	C20K	84	84		Y			Y	Y	H	10ms	RAM,EPROM	1K	Japon	
	C120	256	256	22	Y			Y	Y	B,H	10ms	RAM,EPROM	2.6K	Japon	
	C500	512	512	64	Y		Y	Y	Y	B,H	5ms		8K	Japon	

TABLA 1.1 CUADRO COMPARATIVO DE PLC



FABRICANTE	MODELO	TOTAL DE I/O		MAX. I/O ANALOGICAS	LOGICA DE CONTACTOS	LENGUAJES DE ALTO NIVEL	CAPACIDADES DE PID	DOCUMENTACION	TIPO DE INTERFACE	TIEMPO DE CICLO /1K	TIPO DE MEMORIA			COMENTARIO
PHILIPS B.V. (Eindhoven, The Netherlands)														
MC20	120	120		Y	Y		Y	Y	B,D	2ms		RAM,EPROM,EEPROM	2K	Holanda
	PC20	2000	2000	300	Y	Y	Y	Y	Y	B,D	1ms	RAM,EPROM,EEPROM	16K	Holanda
PHOENIX DIGITAL CORP (Phoenix, AZ)														
	DPAC	832	320	512	Y	Y	Y	Y	Y	B,H	Var	CMOS,RAM,UV PROM	180K	USA

1.2 SIMULADOR DE PLC CON PC.-

Hemos descrito en el capítulo anterior el porque de la existencia de los autómatas programables (PLC), ahora nos proponemos hacer una descripción de como se encuentra compuesto un equipo simulador autómata.

El componente principal de un autómata programable PLC (Programmable Logic Controller) se centra en el *software de aplicación* que básicamente para nuestro diseño en particular se encuentra diseñado en el lenguaje C.

Las ventajas que permite este lenguaje con alta funcionalidad son: edición del texto del programa fuente (*.C, o *.CPP), compilación del programa fuente, creación del programa objeto de enlace (*.OBJ), creación del programa ejecutable (*.EXE), librerías estándares de cabecera (<.h>), las mismas que contienen funciones ya predeterminadas y facilitan al diseñador la creatividad del diseño por software, y entre otros como grabar en disco, imprimir en papel, ejecutar el programa escrito en listado de mnemónicos.

Gracias a las facilidades de diseño por medio de la interfases gráficas del lenguaje C, se puede realizar un diagrama de contactos, así también se puede disponer de entradas y salidas por medio de la utilización de una interfase física electrónica de adquisición de datos. El enlace con la interfase's electrónica se logra; gracias a la utilización del puerto centronics DB-25F del PC. La comunicación se la realiza a través de este puerto paralelo como medio de enlace de procesamiento de datos.

El software de programación C tiene características altamente funcionales - operativas, que permiten al ingeniero en computación diseñar y desarrollar programas de control vía puertos I/O; facilitar y sintetizar la simulación o control de un sistema externo.

Describimos los componentes necesarios para configurar un simulador de PLC con las siguientes características:

- Un Computador personal o PC de preferencia:

  - Microprocesador Intel Pentium II 233 Mhz - 333 Mhz.
  - Memoria de Acceso Aleatorio RAM: 64 MB.
  - 8 MB memoria de video.
  - Tarjeta Multipuertos, paralelo: LPT1, LPT2, LPT3, LPT4.
  - Disco Duro 6 Gigabytes.
  - Drive de 3 1/2".
  - Tarjeta aceleradora Gráfica de 64 bits.

- Sistema Operativo: Windows 98, MSDOS,
- Monitor a color UVGA o monocromático VGA.
- Impresora generador de reportes.
- Disco con el software simulador.
- Interfase Electrónica física (Tarjeta de control de puertos I/O)
- Protoboard o tablero de conexiones.
- Interfase electrónica externa, circuitería de control de potencia.

En la figura 1.1 se observa el equipo que es necesario para desarrollar un simulador de PLC.

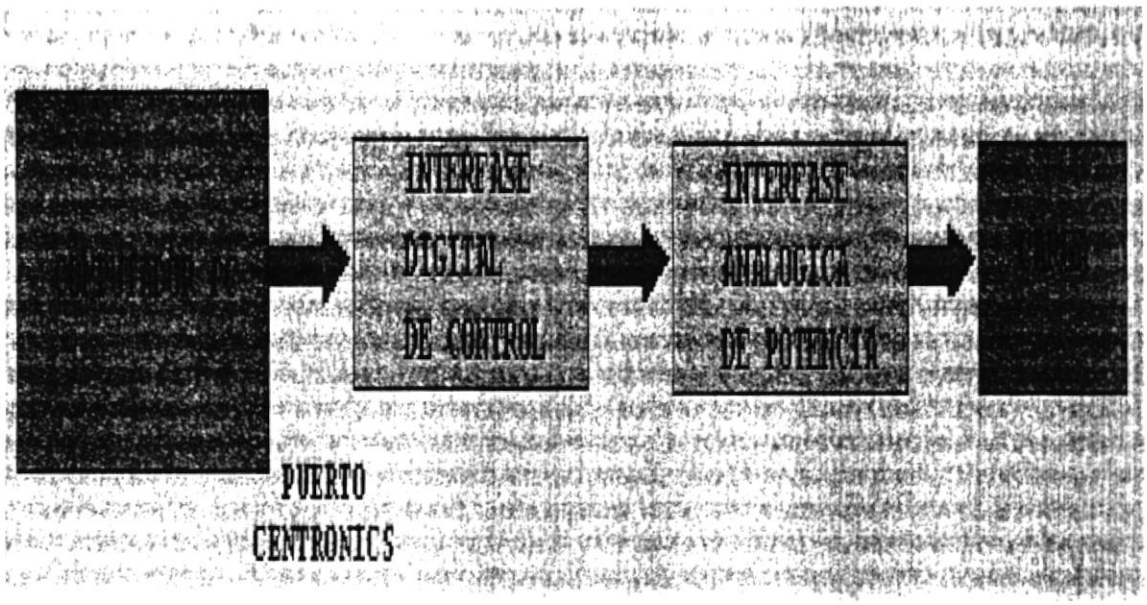


Figura 1.1 Modelo del Sistema Simulador

1.2.1 CARACTERISTICAS DE UN SIMULADOR AUTOMATA PLC.-

El simulador de autómata programable posee las siguientes características:

- Programación en mnemónicos o diagrama de contactos.
- Conversión de un diagrama a listado y de un listado a diagrama.
- Grabar, y recuperar el programa en disco.
- Configurar la tarjeta de adquisición de datos en 4 formas posibles:

  - ♦ entradas / 8 salidas
  - ♦ entradas / 12 salidas
  - ♦ entradas / 8 salidas
  - ♦ entradas / 16 salidas

- 16 Temporizadores de 0 a 999 segundos.
- 16 Contadores ascendentes de 0 a 999 eventos.
- 16 Marcas o bobinas digitales internas.
- Bobinas de SET / RESET
- Ejecución y visualización interactiva en pantalla, activación de una variable.
- Ejemplos básicos de programación.

En la figura 1.2 podemos apreciar los componentes de un autómata.

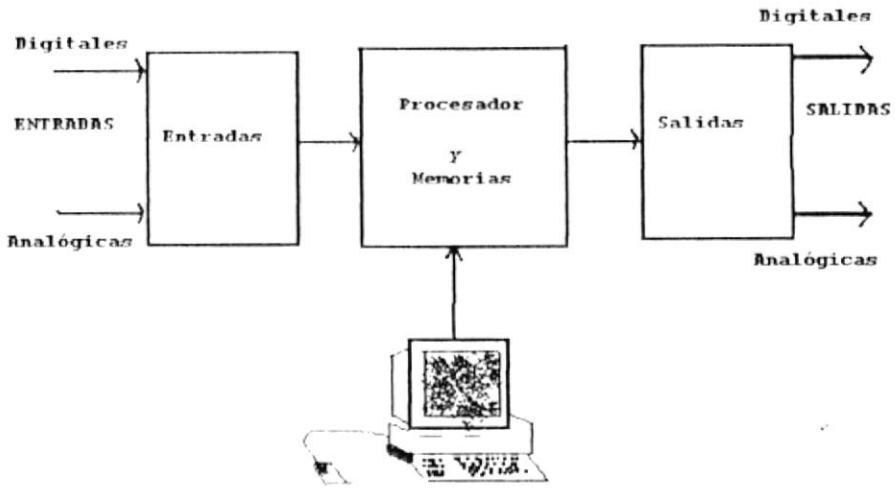


Figura 1.2 Modelo Entradas -Salidas en un Autómata Programable PLC

Se dispone también de otras ventajas adicionales tales como: cambiar la dirección base de la tarjeta de adquisición de datos, de esta manera se pueden realizar ajustes a las direcciones de los puertos I/O; cada uno de ellos se encuentran plenamente identificados en los computadores personales *PC*, estas direcciones se encuentran en base hexadecimal y pueden ser (300h a 31Fh).

Este tipo de ajuste por hardware se consigue; gracias al posicionado de los jumpers o puentes internos que proporcionan generalmente todas las tarjetas de los PC.

Ahora también algunas tarjetas permiten el redireccionamiento de las direcciones bases de salida de los puertos I/O, por medio del ajuste por software. Depende del grado de conocimiento y de la habilidad para disponer cualquiera de los dos tipos de ajuste.

El ciclo de ejecución de un autómata programable lo podemos representar como se muestra en la figura 1.3

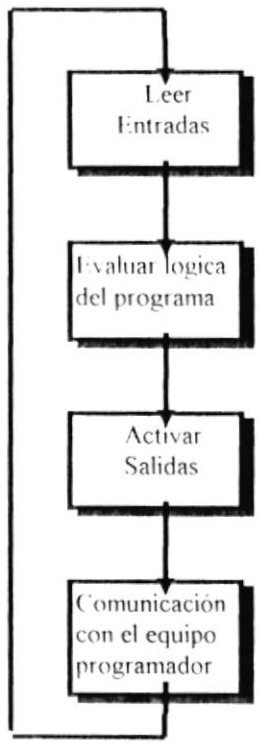


Figura 1.3 Ciclo de Ejecución de un Autómata Programable

1.2.2 HARDWARE UTILIZADO POR LOS AUTOMATAS PROGRAMABLES.-

Como se puede apreciar en la figura 1.2, el autómata programable se compone de tres partes fundamentales y una opcional; las tres partes básicas son:

a. Circuitos electrónicos de entrada:

Estos dispositivos electrónicos se encargan de convertir los niveles de voltajes provenientes de los sensores a niveles de lógica TTL.

b. Unidad de procesador:

Está encargada de leer las entradas, así como también de evaluarlas a cada una de ellas con el programa escrito por el usuario y activar las salidas que se habilitan por medio del uso de la lógica del programa, otra característica es de realizar la comunicación con el equipo programador, en la figura 1.3 se muestra el ciclo de ejecución del programa ejecutado por el procesador.

c. Circuitos electrónicos de salida:

Están encargados de convertir los niveles de lógica TTL (transistor transistor lógico) a niveles de voltaje y corriente necesarios para activar a los dispositivos de potencia.

El componente opcional lo podemos referir como el **equipo programador**, ya que una vez programado el autómata, éste se lo puede retirar físicamente, el autómata guarda el programa principal en su memoria RAM o EEPROM.

1.2.3 PROGRAMACION DE LOS AUTOMATAS.-

El corazón de la existencia de un autómata es el programa o software que es diseñado para uso del usuario.

Un programa es un conjunto de instrucciones que tiene por finalidad indicar al autómata lo que tiene que hacer; el programa puede ser desarrollado en notación ASCII (texto) o en forma gráfica.

Cualquiera que sea la forma que se utilice para la programación, el software que incluye el fabricante genera un programa en lenguaje de máquina entendible solamente por el autómata.

El software simulador de PLC es programable en lista de instrucciones y en diagrama de contactos.

Un ejemplo de un programa realizado en lista de instrucciones y en forma secuencial sería interpretar la siguiente instrucción:

$$Y = A.B.C + A.C + B.C \qquad \textbf{(ec. 1.1)}$$

Ejecutar esta función de tipo Booleana por medio de la interpretación por lista de instrucciones de programa; conlleva a ser cuidadoso en la programación ya que de lo contrario erraríamos en la ejecución de un proceso, para describir cada uno de los pasos de la programación de la ec. 1.1 lo podemos representar en la Tabla 1.2 :

Lista de instrucciones	Ecuación lógica Y = A B C + A C + B C
1 Load A	A
2 And B	.B
3 And C	.C
4 Or (	+
6 Load A	A
7 And C	.C
8)	
9 Or (	+
10 Load B	B
11 And C	.C
12)	
13 = Y	=Y

Tabla 1.2 Lista de Instrucciones y Ecuación Lógica

1.2.4 ELEMENTOS DE PROGRAMACION AUTOMATA.-

Un programa es la representación de un conjunto de instrucciones que lo conforman, y cada instrucción que conforma el cuerpo de un programa se compone en más de una operación de tipo lógica y en una o más variables de representación de estado[INTE96].

La operación lógica lo que hace es indicar a la unidad de proceso central (*CPU*) o procesador lo que tiene que realizar con la variable definida dentro del programa.

Las variables son las representaciones de celdas de memoria, en las cuales se almacena el valor de un dato conocido; para el simulador PC todas las variables son del tipo Bit.

Un bit es la mínima unidad de información del procesador, el bit puede contener la representación de un solo estado lógico binario '0' ó '1'. Por lo tanto acabamos de definir algo muy importante que es: el tipo de estado lógico en que se pueden encontrar las entradas, salidas y marcas digitales, '0' para el estado de apagado y '1' para el estado de encendido; por supuesto en notación de lógica positiva.

Dentro del programa, cada variable debe tener una dirección de memoria; característica que sirve para identificarla una de otra variable, así como la ubicación instantánea para algún requerimiento de búsqueda.

- **Entradas Digitales:**

  A las entradas de tipo digital se las representa por la letra **I**; cada una de las entradas posee un número de dirección, esto sirve para identificar con su correspondiente entrada de tipo digital real. Los dispositivos que se conectan a las entradas pueden ser: pulsadores, microsuiches, etc.

- **Salidas Digitales:**

  Las salidas de tipo digital se las representa por la letra **Q**, cada una de las salidas posee un número de dirección, característica que sirve para identificar su correspondiente salida de tipo digital real.

- **Marcas:**

  Ejecutan la funcionalidad de una bobina o relé auxiliar además sirve también para memorizar estado o estados de las salidas que no se necesitan activar físicamente, se las representa por la letra M, cada marca lleva una dirección que la identifica internamente. En la Tabla 1.3 se muestra las variables de un simulador.

TIPO	DESCRIPCION	RANGO DIRECCION	EJEMPLO
I	Entrada digital	0...15	I09
O	Salida digital	0...15	O08
M	Marca digital	0...15	M09
S	Habilitación de Temporizador	0...15	S00
T	Contacto digital Temporizado	0...15	T00
Z	Habilitación de Contador	0...15	Z01
P	Entrada de pulsos de Contador	0...15	P01
C	Contacto digital de Contador	0...15	C01

Tabla 1.3 Variables de un Simulador

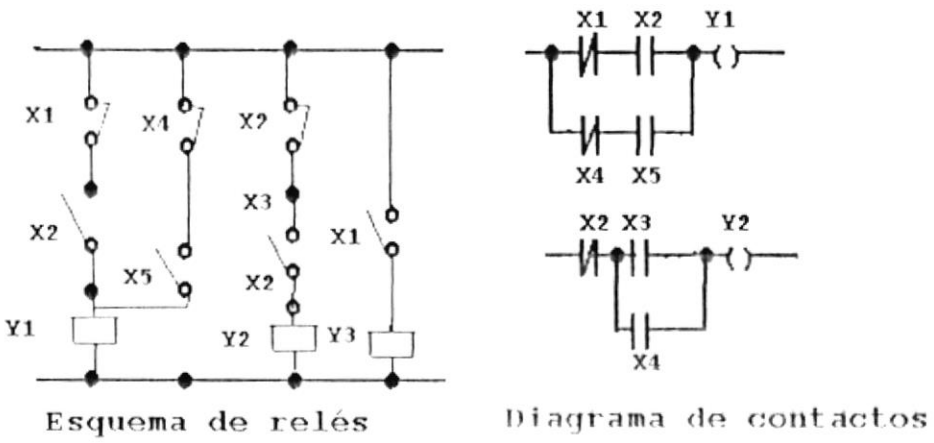


Figura 1.4 Esquemas de Relés y Diagrama de Contactos

1.2.5 FORMAS DE PROGRAMACION EN UN AUTOMATA.-

La función de un software de control es simular las formas de programación de la lista de instrucciones y del diagrama de contactos [INTE 96].

- **Por lista de instrucciones:**

  Es conocido también como lenguaje mnemónico, su abreviatura es *AWL* o *STL*, tiene mucha similitud con la lógica booleana del diseñador inglés BOOLE. En la Tabla 1.2 se ilustra un ejemplo de un programa realizado por *"lista de instrucciones"*; en el programa se visualizan todos los comandos completos de programación, pero en la realidad solo se debe escribir las primeras letras en negrillas. Un buen programador preferiría utilizar el diseño de mnemónicos completos, debido a la gran cantidad de comandos existentes; lo que implica memorizar y tenerlos presente en mente para disponer de ellos cuando se los necesite durante la programación [INTE 96].

- **Por Diagrama de contactos:**

  Son conocidos también con el nombre de *LAD* o *KOP* por sus siglas en inglés *Ladder Diagram* o diagrama de escalera, son muy similares a los esquemas relés o contactos utilizados en la lógica cableada, ver la figura 1.4, aquí se aprecia la similitud.

  Considero personalmente que la programación de un autómata programable *por "diagrama de contactos"* es de tipo más interactivo con el diseñador o el usuario, desde el punto de vista como programador aquí se visualiza la forma en que se desea diseñar un sistema autómata de control.

1.2.6 PROGRAMACION POR DIAGRAMA DE CONTACTOS.-

Un programador que se está iniciando en la programación de los autómatas programables o PLC debe tener presente algunas normas fundamentales cada vez que se inicia el diseño de un programa. Los pasos a seguir deben ser [INTE96]:

1. Comprender claramente el proceso que se va a ejecutar y las etapas de elaboración del mismo. Para ello hay que seguir las siguientes normas:

2. Elaborar un listado detallado de las variables utilizadas. Definimos la palabra "variables" como líneas disponibles de manejo de: entradas, salidas, marcas, temporizadores y contactores.

3. Realizar un diagrama de flujo primero en papel y luego en pantalla.

4. Comprobación de su correcto funcionamiento, esto se lo logra mediante:

   - Revisión de las entradas, el autómata (PLC) debe encontrarse en el modo STOP.
   - Revisión de las salidas, el autómata (PLC) debe encontrarse en el modo RUN.

5. Documentación final: Imprimir la lista de variables, el programa en su totalidad, y el diagrama de conexiones eléctricas.

- **Programación:** *LAD* o *KOP* (Ladder Diagram) conocidos como diagramas de contactos o diagramas de escaleras se basan en las definiciones iniciales de los antiguos sistemas de control secuencial a través de una gran cantidad de relés.
  La programación por diagrama de contactos se realiza a través de dos grafos básicos: representación de bobinas, y relés.

La unión lógica de contactos y de bobinas forman los segmentos o escalones, la conformación de varios segmentos forman un programa en diagrama de contactos.

Los contactos se empiezan dibujando siempre de izquierda a derecha, y la bobina se la fija del lado derecho del segmento del plano.

En la figura 1.5 se muestra un segmento de diseño completo. Este diagrama nos representa un estado de conexiones, el lado izquierdo del diagrama está conectado a una tensión de alto voltaje y el lado derecho del segmento se encuentra conectado a cero voltios.

En el instante que se cierre el contacto A normalmente abierto del circuito, y mientras el contacto B permanezca normalmente cerrado entonces se energizará la bobina C, esto se expresaría como: la tensión de alto voltaje llega hasta el terminal izquierdo del relé C y con su otro terminal conectado a tierra produciría la activación del mismo.

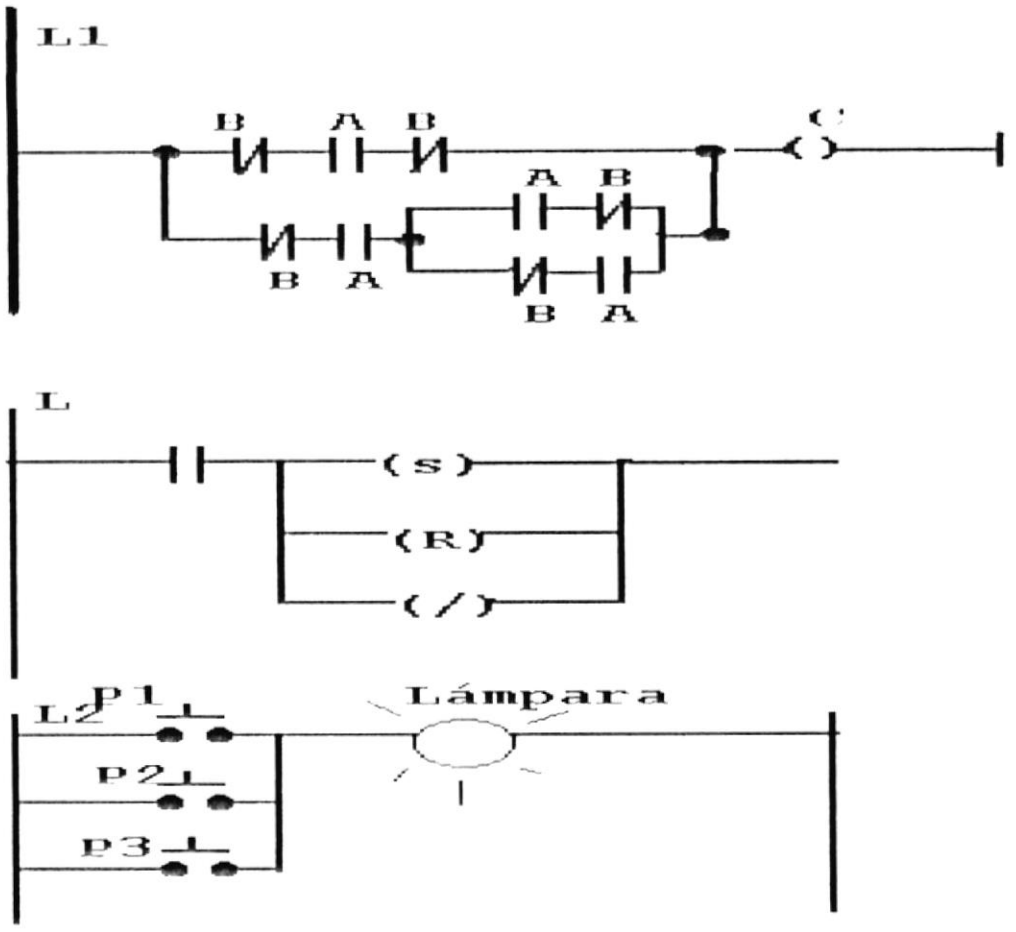


Figura 1.5 Programación por Diagrama de Contactos

Las bobinas disponibles pueden ser de tipo:

- Set (S)
- Reset (R)
- Negada (/)

Toda bobina tiene asociado contactos que pueden ser del tipo normalmente abiertos "NO" (Normal Open), o normalmente cerrados "NC" (Normal Close), los cuales se cierran o se abren instantáneamente; esto es cuando se energiza a un relé del tipo instantáneo.

Cuando la bobina de Set (S) es energizada, entonces activa la salida o marca, quedando en estado lógico '1' activado, así se abra el camino de los contactos que la activaron.

La única manera de desactivar las salidas o las marcas, es por medio de la energización de la bobina de Reset (R). Por lo tanto las salidas o las marcas pasarían a su estado lógico '0' desactivado.

En el diagrama de contacto podemos observar que se escribe junto al símbolo de los contactos el mnemónico que identifica el contacto perteneciente a una bobina, a este mnemónico o letra lo llamaremos variables, que puede ser simbólica o real, recordar que las variables tienen una dirección, la cual en el caso de entradas y salidas se las identifica con la conexión real física.

Una ventaja de poder trabajar con los *PLC* es que se pueden utilizar tantos contactos asociados a una variable como la aplicación lo requiera.

Como podemos observar es característica fundamental de un diseñador u operador, conocer como trabaja la lógica booleana para lograr una aplicación correcta en un diseño de lógica de programación de un autómata programable PLC.

1.2.7 DESCRIPCION DE LOS COMPONENTES POR SOFTWARE DE UN AUTOMATA.-

La disposición característica de un menú para un autómata la podemos describir como se muestra en la pantalla del menú principal, ver figura 1.6.

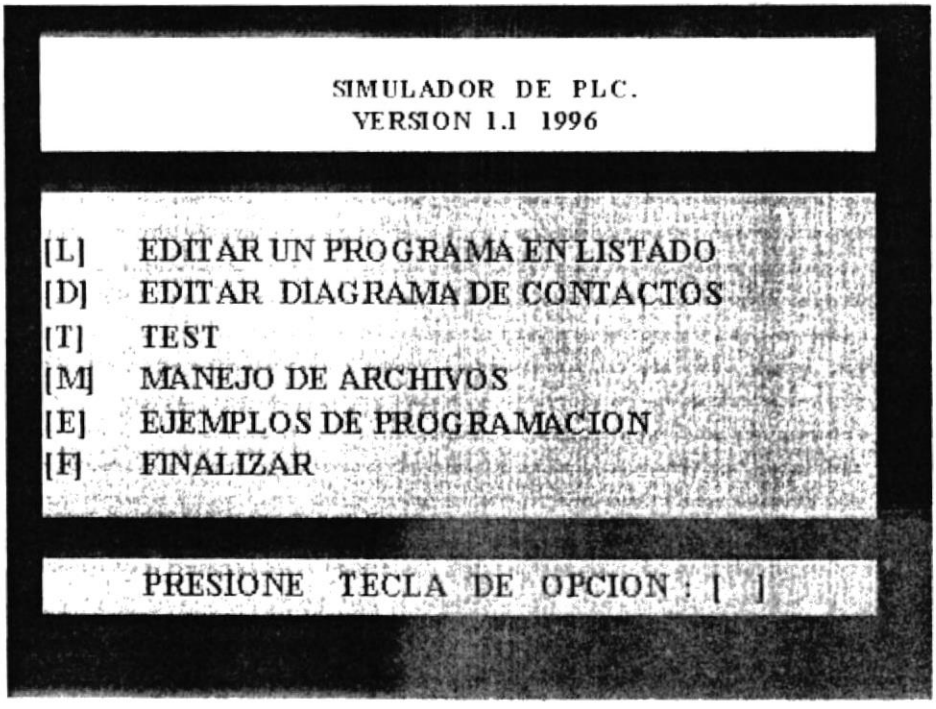
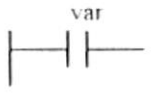
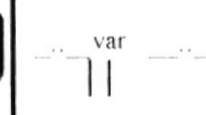
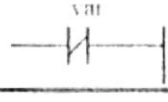


Figura 1.6 Pantalla del Menú Principal de un Autómata Programable

La letra que se encuentra encerrada entre corchetes dentro del menú principal, es la opción que deber ser escogida para seleccionar un ítem en particular.

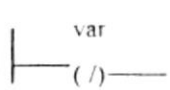
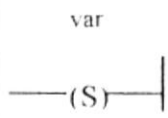
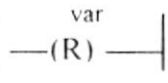
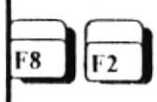
Para realizar la edición de un programa en la modalidad "lista por instrucciones" se debe utilizar la opción "**L**" del menú principal; una vez realizado esto se mostrará en la pantalla un conjunto de instrucciones necesarias para realizar la programación utilizando el simulador de PLC, aquí se describe detalladamente que función posee cada tecla que se selecciona para la elaboración de un diagrama de contactos.

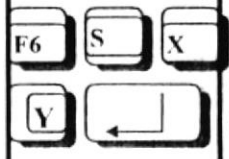
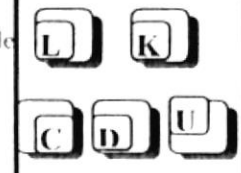
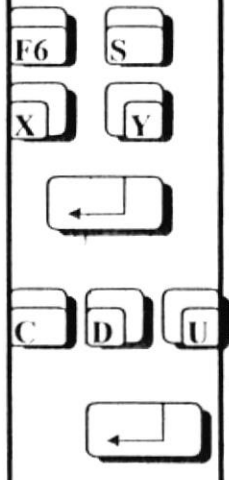
A continuación se detalla en la Tabla 1.4 las instrucciones de programación de un software simulador de un autómata programable.

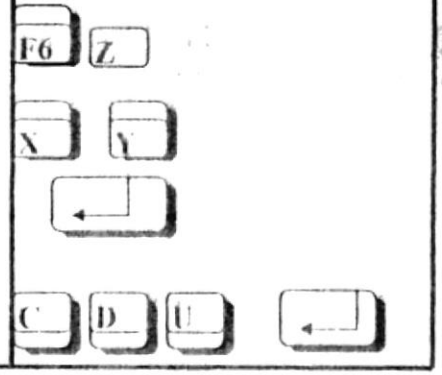
Instrucción de listado	Teclas de Listado	Instrucción de diagrama	Teclas de diagrama	Descripción
Load variable				Carga el valor la variable especificada en "var" en el evaluador de lógica.
Load variable negada				Carga el valor de "var" negado en el evaluador de lógica
And variable				Realiza un "Y" lógico entre la lógica anterior y la variable "var".
And var. Neg.				Realiza un "Y" logico entre la logica anterior y el valor negado de "var".
Or variable				Realiza un "O" logico entre la logica anterior y la variable "var".

Or variable negada	O Y	var	F4 F2	Realiza un "O" lógico entre la lógica anterior y el valor negado de "var"
Bloque And serie A (	A (			Abre un bloque de programa el cual estará en serie con la lógica del programa
)			F4	Se entiende por bloque de programa como aquel que esta compuesto por las instrucciones contenidas en esta tabla
Bloque Or paralelo O(	O (		F4	Abre un bloque de programa el cual estará en paralelo con lógica del programa
)				
Igualar variable = variable	L	var	F4	Iguala la evaluación de la lógica anterior a la variable "var" Para esta instrucción sólo son válidas las variables Q, M, S, Z, P

Tabla 1.4 Instrucciones de Programación

Instrucción de listado	Teclas de listado	Instrucción de diagrama	Teclas de diagrama	Descripción
Igualar variable con lógica negada = N variable				Iguala la evaluación de la lógica anterior, la niega y la lleva a la variable "var". Para esta instrucción sólo son válidas las variables: Q, M, S, Z, P.
Poner la variable en Set S variable				Si la lógica de programa es evaluada en "1", pone en set la variable "var" dejándola así, aún cuando cambie la evaluación lógica de programa a "0".
Desactivar la variable "var"				Si la lógica de programa es evaluada en "1", la variable "var" es desactivada

				
Entrar variable **WXY		* wxy cdu		Se escribe la variable que acompaña la instrucción lógica, donde "W" es la variable y "xy" es la dirección. W: Y, O, M, S, T, Z, P, C. X: 0..1 Y: 0..9
Entrar valor de temporización		WXY ( ) cdu		Carga el valor de contaje máximo al final de la cual se activa la variable Cxy c: centenas 0..9 d: decenas 0..9 u: unidades 0..9 EVENTOS

Entrar valor de contador LK CDU		* WXY ( ) cdu	

El editor de diagrama de contactos posee los comandos F1 a F8 por el medio del cual podemos realizar las depuraciones necesarias para obtener un diagrama de conexiones correcto. La figura 1.7 muestra las pantallas de configuración [INTE 96].

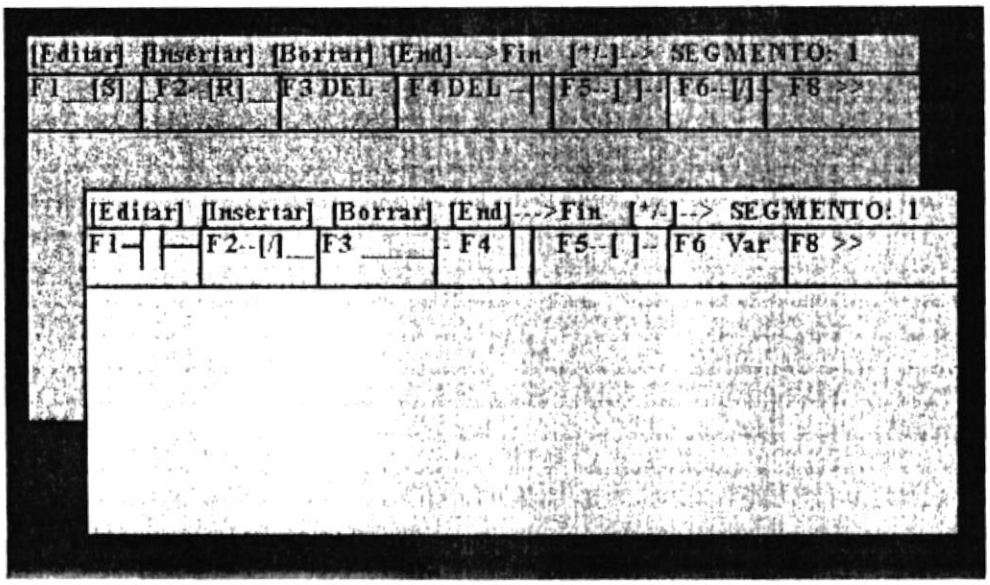


Figura 1.7 Pantalla Editora para un Diagrama de Contactos

Si deseamos realizar una edición de diagramas de contactos entonces se escoge el comando F1 y se posiciona en el plano diagramador del editor. Observar la figura 1.8.

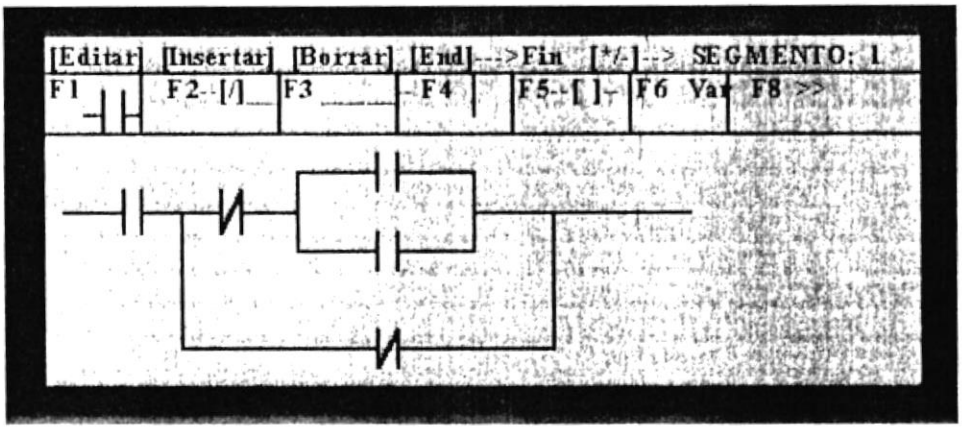


Figura 1.8 Edición de un Diagrama de Contactos

Veamos como se realiza una programación de un autómata PLC siguiendo los pasos de diagramación. La programación simplemente se la puede realizar presionando los comandos de las funciones requeridas, entonces se va realizando el diagrama, el mismo que es dibujado en la pantalla; cada diagrama realizado recibe el nombre de "segmento" el programa permite un máximo de 32 segmentos [INTE 96].

Las funciones de comando como podemos observar en la figura 1.7 permite al usuario editar, borrar, insertar entre segmentos. Otras características adicionales es que se puede recuperar y grabar información en el disco duro, esto es por medio del comando "M" que sirve para el manejo de archivos; este se lo invoca desde el menú principal [INTE 96].

Por ejemplo si se desea que se active una alarma sonora, y que se active la misma cuando tres interruptores se cierren o cuando se presiona un pulsador para verificar el estado de prueba de la alarma entonces los pasos a seguir son:

Como se muestra en la figura 1.6 "menu principal de un autómata PLC" se selecciona:

- comando "D" "editar un diagrama de contactos".
- comando "E" "ejemplos de programación".

Para cerrar los interruptores que van en serie, debemos dar valores a los mismos para identificarlos en el programa como:

ENTRADAS

- interruptor 1 : **I00**
- interruptor 2 : **I01**
- interruptor 3 : **I02**
- pulsador : **I03**

SALIDAS

- bocina de alarma : **Q00**

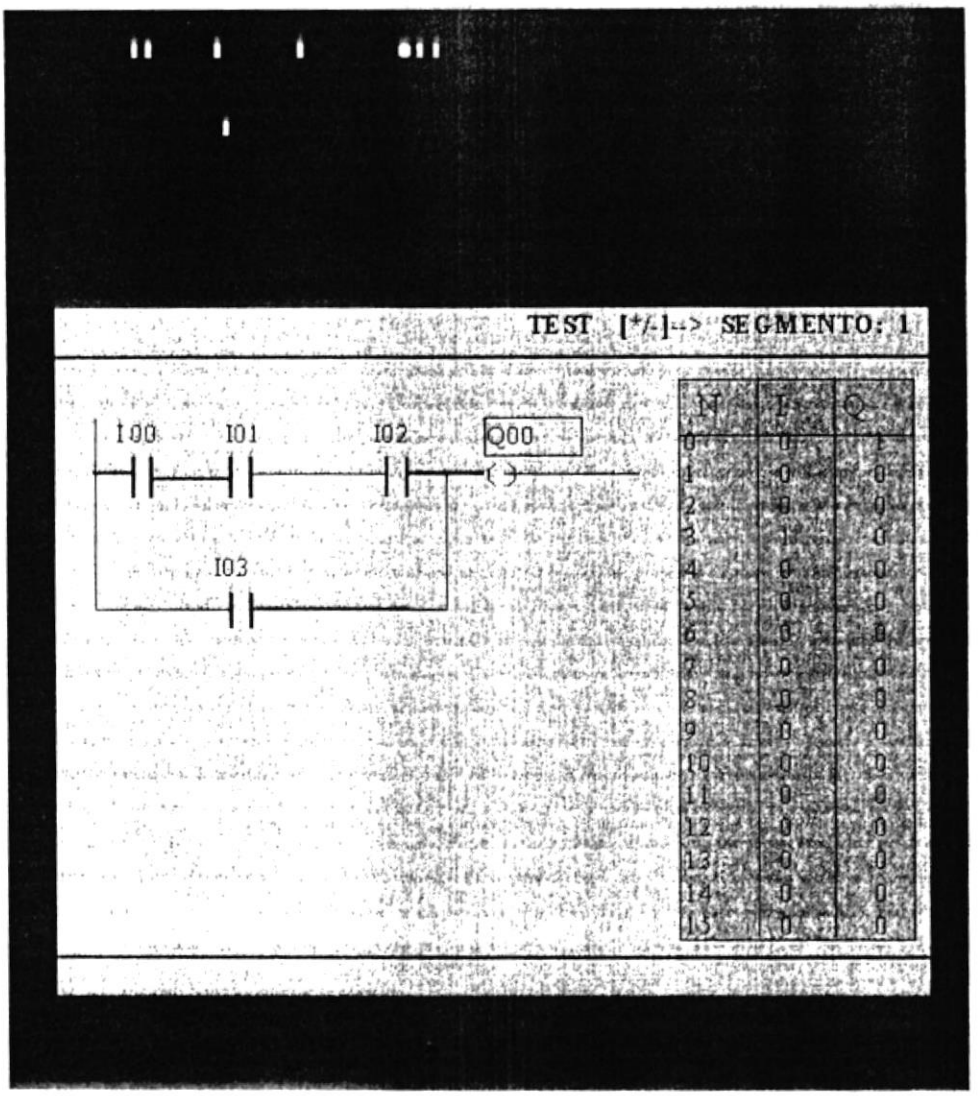


Figura 1.9 Diagrama Completo - Pantalla de Funcionamiento

En el editor se pulsa la tecla F1 tres veces, de manera que se obtiene la pantalla de edición de la figura 1.8. Recordar que igualmente se puede accionar por un pulsador a la alarma sonora esto es de manera independiente, es de ésta forma que debe existir una conexión paralela a los tres contactos en serie normalmente abiertos.

Pulsar F5 para construir una bobina, posicione luego el cursor en "*" y posteriormente pulse el comando F4, lo que se obtiene es el diagrama de contactos en el editor. Posicione el cursor en "*" y presione F1, luego presione F3 dos veces, lleve el cursor a "X" y pulse F6, escriba I00, haga lo mismo para las instrucciones de programa I01, I02, y para el pulsador I03 de prueba de acción de la alarma, observar la figura 1.9.

Para revisar el estado de funcionamiento del ejemplo anterior, podemos utilizar la opción de comando "TEST" del menú principal (ver figura 1.6) la misma que tiene por finalidad evaluar las entradas y generar sus respectivas salidas, esto es dependiendo de la lógica de programación utilizada por el programador.

No solamente es importante el software de control de un autómata programable sino también hay que tener en cuenta los circuitos electrónicos de entradas y salidas que ejercen un control sobre los tipos de cargas que deseamos operar. Un controlador simple generalizado; para los puertos de entrada y de salida para este diseño en particular lo podemos identificar en la figura 1.10.

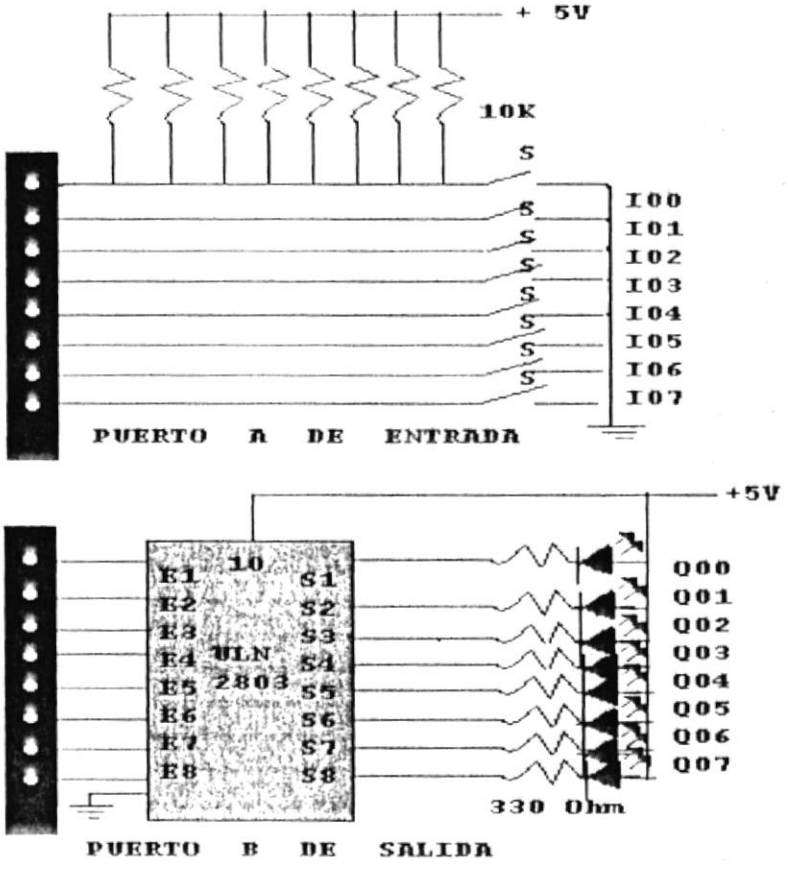


Figura 1.10 Interfase Entradas y Salidas

Este software que acabamos describir es un diseño en particular de una compañía proveedora de software para controles de PLC, por lo tanto cada diseñador tiene su propio diseño de interfases visuales que son por medio de las cuales el usuario puede interactuar con la máquina (PLC). Este diseño en particular de autómata está dirigido hacia una plataforma para MSDOS

CAPITULO II

SISTEMA AUTOMATICO DE CONTROL CON UN PC

2.1. DESCRIPCION DEL SISTEMA DE AUTOMATIZACION.-

Nuestro sistema se basa en la automatización de control, esto significa poder controlar periodos de energización y desenergización de cargas monofásicas-bifásicas como por ejemplo un motor DC o un motor de inducción, por medio del uso de un computador personal y el sistema electrónico de control.

Hemos visto como los diseños de los sistemas automatizados utilizan los microcomputadores, y en nuestro caso son los computadores personales los que facilitan nuestro desempeño en las labores de trabajo continuo. Este diseño en particular dispone el manejo de cuatro cargas monofásicas-bifásicas con una potencia en carga de ½ a 1.5 HP.

Para su correcto funcionamiento y desempeño es necesario que se realice la implementación de un software de control, labor que recae sobre nosotros como diseñadores.

La caracteristicas del software es controlar la programación de activación y desactivación de los periféricos externos al puerto Centronics DB-25F. De modo que este software deberá programar cuatro tiempos de conmutación, divididos en dos tiempos de encendido y dos tiempos de apagado para el control único de carga, sobre una base de tiempo de 24 horas de control continuo.

Muchos diseñadores de sistemas han demostrado que existe un gran poderío en la funcionalidad y disposición del puerto Centronics para el control de periféricos externos, y en realidad se lo está aprovechando muy poco en todas sus utilidades y ventajas que ofrece al usuario. Es por ello que el diseño automático de control está enfocado en el funcionamiento operativo de este puerto.

Si no conocemos programar en Lenguaje C, así como también del funcionamiento y operatibilidad del puerto Centronics DB-25F, y de las utilidades de la electrónica con optoacopladores; entonces no vamos a llegar muy lejos.

En los sistemas que utilizan microprocesadores; siempre se a diseñado sistemas de control mediante la conexión directa a los buses de direcciones y de datos del micro, como por ejemplo; el microprocesador SDK 8085.

El diseño va más allá de tener que utilizar un microprocesador de características antiguas, o tener que abrir el motherboard del computador para conectarnos directamente a la barra de direcciones y de datos del PC, esta molestia la podemos omitir simplemente por el aprovechamiento y manejo del puerto Centronics que se encuentra en la ranura posterior externa de un PC.

Los sistemas de características sencillas de adquisición de datos tipo análogos (no son críticas ni la velocidad y resolución de los datos), se han remodelado con los nuevos sistemas, ante el desarrollo continuo de los circuitos especializados que realizan la conversión de señales análogas a digitales (ADC), de esta manera los datos puedan ingresar al computador a través de los puertos paralelos y seriales, simplificándose la interfase electrónica y reduciendo la necesidad de construcción de fuentes muy complejas para los convertidores.

2.1.1 COMPONENTES DEL SISTEMA.-

El sistema automático de control de cargas esta diseñado inicialmente para cuatro cargas de características monofásicas-bifásicas y cada una de ellas son controladas independientemente una de la otra por el computador personal (PC). La circuitería de la interfase electrónica de control facilita la operación de manejo de los periféricos conectados externamente al PC y su característica es de tipo optoaisladora.

Los componentes de nuestro sistema automático de control son:

- **Computador Personal "PC":** Es la Unidad de Procesamiento Centralizado (CPU) la que controla el funcionamiento correcto del sistema. Los componentes operativos para este diseño en particular requiere; de un computador de las siguientes características o equivalentes:
- Microprocesador INTEL PENTIUM 233 Mhz
- Monitor SVGA 15"
- Memoria RAM 32 Mb
- Disco Duro 4.0 Gb
- Tarjeta Multipuertos "1 puerto paralelo - 2 puertos serial "

- **Puerto Centronics:** Conocido también como puerto paralelo, su conector tiene el estatus estándar DB-25F. Esto significa que posee 25 pines que sirven para interconectar cualquier periférico externo al computador, como por ejemplo, la impresora.

- **Interfase de control por Software:** Está conformada por líneas de código en lenguaje C, y en su conjunto es el programa que controla al sistema; y le indica como se debe ejercer el funcionamiento de un periférico. La interfase por software

es la componente interactiva con el usuario. Cualquier modificación en la estructura de los parámetros como datos del programa, significa un elemento correctivo en el funcionamiento de las cargas controladas.

- **Interfase de control Electrónica:** Está conformada por la circuitería de control fisico - lógica que enlaza al CPU con los periféricos externos conectados al computador. Del diseño electrónico depende el correcto funcionamiento operativo sobre una carga. Es lógico pensar que ésta interfase electrónica debe ser de característica digital TTL. La razón es sencilla debido a que el computador funciona con señales digitales TTL y debe existir compatibilidad con los niveles lógicos que el PC entrega al puerto paralelo DB-25F.

2.1.2 DISPOSITIVOS OPTOAISLADORES.-

Un optoacoplador también es llamado optoaislador o aislador óptico. Se encuentra conformado por un solo paquete que contiene un Led y un fotodetector. El optoacoplador o aislador optoelectrónico, contiene en un mismo encapsulado un emisor infrarrojo cuyo elemento se identifica como Galio - Arsénico (IR LED con emisión de 900 a 940 nm) y un fotodetector [BOYL94].

Dentro de este dispositivo podemos hacer la representación del mismo como: un diodo emisor de luz, y un fotoreceptor configurado como un par Darlington o un transistor; cuya función es de recibir la radiación luminosa proveniente del diodo emisor, otros dispositivos están conformados por: fotodiodos, fototransistor, DIAC, o SCR (Rectificador Controlado de Silicio).

El encapsulado está formado de un material opaco a la luz, de manera que el conjunto emisor-receptor queda protegido de la luz ambiental. Lo importante es que no existe una conexión eléctrica entre la entrada y la salida y que además los elementos internos no pueden invertir sus funciones, esto significa que la señal que se emite sólo puede pasar en una sola dirección.

Dependiendo de las características de aislamiento y capacitancia entre la entrada y la salida del optoacoplador, éste puede ser insensible a las señales de modo común y proporciona protección a los circuitos de entrada o de control con respecto a los voltajes altos en el circuito de salida o de potencia [BOYL94]

Existe una amplia gama de combinaciones de estos dispositivos con diferentes formas de encapsulados, obteniéndose así una gran variedad de posibilidades en la elección de características de entrada - salida y acoplamiento.

Sus principales aplicaciones residen en el acoplamiento entre dos etapas de un circuito electrónico, entre las que debe existir un "*elevado aislamiento eléctrico*". También se emplea, en algunos modelos para detectar el movimiento de motores o piezas que giran, accionadas por estos [INTE 96].

El modelo más generalizado consta de un diodo Led situado a muy corta distancia de un fototransistor, estando ambos dispuestos en un encapsulado común cuya forma exterior es la de un cuerpo rectangular con seis terminales dispuestos en dos filas paralelas (Dual in Line).

Dos de estos terminales corresponden al ánodo y al cátodo del Led y dos o tres de los restantes serán los puntos de conexión del fototransistor segun exista o no una conexión eléctrica con la base.

Otro tipo de cápsula, muy diferente de la anterior, es en la forma de una U, con los elementos emisor y receptor situados en los dos brazos verticales, de forma que pueda interrumpirse su acoplamiento óptico mediante cualquier objeto plano opaco que se introduzca entre ambos (por ejemplo una hoja de papel).

Los terminales de conexión aparecen, por separado en la zona inferior. Las principales características que se han de tomar en consideracion a la hora de elegir un tipo determinado son las siguientes [SEMI87]:

- **Tensión directa del LED V_F:** Es la diferencia de potencial que se produce entre los dos terminales del LED cuando por el atraviesa la corriente de excitación. Está comprendida entre 1.5V y 2.2 V para la mayoría de los modelos.

- **Tensión inversa máxima del LED V_R:** Corresponde al máximo voltaje inverso que se puede aplicar al LED emisor sin que entre a la región de avalancha.

- **Tensiones máximas del fototransistor BV_{CEO}, BV_{CBO}:** BV_{CEO} es el voltaje de ruptura entre el colector y emisor, voltaje medido entre el colector y el emisor con la base abierta. BV_{CBO} : BV_{CBO}, es el voltaje de ruptura entre el colector y la base, voltaje medido entre el colector y la base con el emisor abierto.

- **Corriente de oscuridad I_{CEO}:** Es la corriente que circula del colector al emisor con la base abierta y sin recibir radiación alguna de luz en el fototransistor.

- **Relación de transferencia de corriente LED - fototransistor CTR (%):** Relacionado con la ganancia de corriente entre salida para la entrada $CTR(\%) = I_O/I_F \times 100\% = I_{CEO}/I_F \times 100\%$

- **Voltaje de aislamiento LED - fototransistor Viso:** Voltaje DC de la fuente de aislamiento. Máxima relación de aislamiento del dieléctrico entre la entrada y la salida del optoacoplador.

- **Tiempo de encendido T_{ON}:** Es el tiempo requerido por el dispositivo de salida para pasar de su estado de no conducción a su estado de encendido.

- **Tiempo de apagado T_{OFF}:** Es el tiempo requerido por el dispositivo de salida para pasar de su estado encendido a su estado de no conducción.

- **Tensión DC de entrada V:** Corresponde al rango DC del voltaje de entrada permitido, para activar el LED emisor (previamente se debe disponer de una resistencia limitadora de corriente a la entrada del mismo).

- **Tensión DC de salida V_O:** Corresponde al máximo voltaje DC que el dispositivo puede entregar.

- **Potencia del dispositivo P_t:** Disipación de potencia máxima en todo el dispositivo.

De todas las características anteriores es de destacar la Relación de transferencia entrada salida (LED - fototransistor) CTR (del inglés Current Transfer Ratio). Se define por la relación entre la corriente de salida y la de entrada y se expresa como un porcentaje de esta última.

Para clasificar lo mejor posible este parámetro supongamos un optoaislador por el que circula una corriente de I_f = 5 mA a través del diodo IR-LED. Esta corriente genera otra en el fototransistor de I_{CEO} = 5 mA. El CTR será por lo tanto la relación entre estos dos valores, es decir: CTR = 5/5 = 1 o bien el 100%.

La fórmula anterior quiere decir que un optoacoplador con un CTR del 100% proporciona una corriente de salida de un miliamperio, cuando circula un miliamperio a través del diodo emisor IR-LED en la entrada. Las hojas de datos técnicos suministrados por los fabricantes entregan este parámetro especificado en función de la corriente directa de entrada I_f y en condiciones de salida determinadas, esto es con un voltaje conocido entre el colector y el emisor, si es el caso de un optoacoplador con fototransistor.

La respuesta de longitud de onda de cada dispositivo se ajusta para que sea lo más idéntica posible y permitir el mejor acoplamiento. Se diseñan con tiempos de respuesta tan pequeños que pueden emplearse para transmitir datos en el intervalo de los megahertz.

La medida de la capacidad de un optoacoplador para transferir una señal de manera eficiente se denomina "Razón de Transferencia de Corriente" o CTR (Current Transfer Ratio), y es dependiente de la eficiencia de emisión del diodo LED (IR-LED), del espacio entre el emisor y el receptor (para optoaisladores en forma de U), del área, sensibilidad y ganancia de amplificación del detector.

También está sujeta a la no linealidad (corriente, voltaje y temperatura) de los dispositivos internos (emisor - receptor), lo que produce una función de transferencia más compleja.

La capacidad de un optoacoplador para soportar una tensión entre la entrada y la salida, usualmente se expresa como "Voltaje de Aislamiento entre Fuentes" (Viso) y es esencialmente dependiente de las características dieléctricas del material que transfiere la luz emitida por el diodo infrarrojo IR-LED. Los límites típicos se encuentran entre 2500 y 7500V [SEMI87].

En la figura 2.1 (a-b) detallamos dos posibles configuraciones de integrados aisladores:

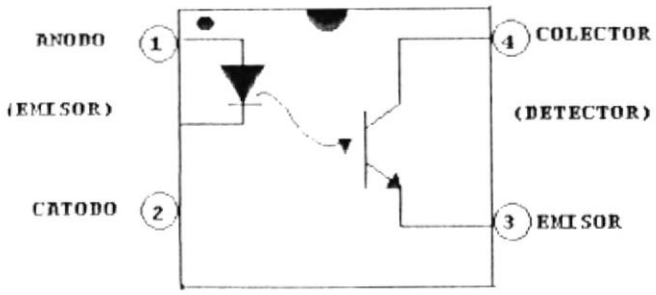


Figura 2.1 (a) Optoaislador LED - Transistor 4N26 - ECG3081

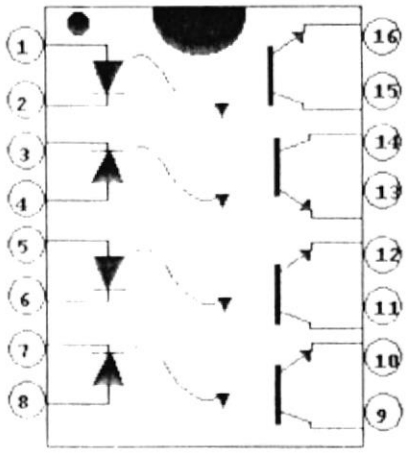


Figura 2.1 (b) Circuito Integrado Optoaislador L7743 Litronix

Los valores nominales máximos y las características eléctricas son indispensables conocerlas, ya que de ello depende el funcionamiento correcto del optoacoplador. Por ejemplo para el integrado L7743 la corriente $I_{(D)}$ está en el orden de los nanoamperios y la potencia del LED y del transistor de salida son aproximadamente iguales [BOYL94].

LED en cada canal (de arseniuro de galio) [BOYL94]	
Disipación de potencia @25°C	200mW
Degradar linealmente desde 25°C	2.6mW/°C
Corriente avance continua	150mA

Fototransistor (cada canal) de silicio detector [BOYL94]	
Disipación de potencia @25°C	200mW
Degradar linealmente desde 25°C	2.6mW/°C
Voltaje de falla de colector - emisor	30V
Voltaje de falla de emisor - colector	7V
Voltage de falla de colector - base	70V

Paquete [BOYL94]	
Disipación total de paquete en ambiente de 25°C (detector LED plus)	250mW
Degradar linealmente desde 25°C	3.3 mW/°C
Temperatura de almacenamiento	-55 a150°C
Temperatura de operación	-55 a 100°C

Tabla 2.1 Valores Máximos de Operación L7743

Parámetro [BOYL94]	Min	Tip	Máx	Unidades	Condiciones de prueba
Arseniuro de galio LED					
Voltaje directo		1.3	1.5	V	I_F=60mA
Corrriente inversa		0.1	10	µA	V_R=3.0V
Capacitancia		100		pF	V_R= 0V
Detector fototransistor					
B_{VCEO}		30		V	I_C=1mA
I_{CEO}		5.0	50	nA	V_{CE}=10V,I_F=10mA
Capacitancia colector - emisor		2.0		pF	V_{CE}=0V
BV_{ECO}		7			
Caracteristicas acopladas					
Proporción de transferencia de corriente dc	0.2	0.35			I_F=10mA,V_{CE}=10V
Capacitancia, entrada a salida		0.5			pF
Voltaje de falla	2500			V	DC
Resistencia entrada salida		100		GOhm	
Vsat			0.5	V	I_C=1.6mA,I_F=16mA
Demora de propagación					
t_D conectada		6.0		µs	R_L=2.4K,V_{CF}=5V
t_D desconectada		25		µs	I_F=16mA

Tabla2.2 Características Eléctricas por Canal L7743 a 25°C

2.2 FUNCIONAMIENTO DE LA INTERFASE ELECTRONICA DE BAJA POTENCIA.-

La interfase electrónica está diseñada para energizar y desenergizar las cargas de potencia. Mencionamos la palabra carga como un elemento que consume potencia y puede estar representado por dispositivos puramente resistivos como una resistencia de potencia o como un elemento resistivo inductivo como un motor AC. Indistintamente se puede manejar cargas DC o AC, la naturaleza de las mismas no perjudica el funcionamiento de la interfase optoaisladora.

El sistema de interfase electrónico diseñado es netamente de características digitales del tipo TTL "0v" "+5v". El seguimiento de operación del sistema para una sola etapa se inicia a partir de que una señal digital de activación de carga y en lógica negativa aparece a la salida del puerto, en realidad es un mandato proveniente del CPU al puerto DB-25F, de manera que se produce un nivel lógico "0" a la salida del puerto I/O.

De esta manera el cátodo del diodo optoacoplador pasa a tomar un valor lógico de "0" y como el ánodo del diodo emisor se encuentra conectado a un voltaje de "+5v" (valor lógico "1") entonces este elemento conduce, irradiando una luz infrarroja interna en el dispositivo semiconductor.

La resistencia externa conectada al cátodo del diodo emisor tiene por finalidad limitar la magnitud de la corriente en el dispositivo y asegurar su operatibilidad en un rango seguro.

Una vez que el diodo emisor infrarrojo entra en conduccion, este hace que exista una recombinación de los electrones en la base abierta del transistor optoacoplador, la corriente que pasa del colector al emisor del transistor activa al transistor Q1 existiendo

corriente de entrada en el emisor y corriente de salida en el colector, esta corriente de colector en Q1 se bifurca y energiza al microrelé colocando un extremo de la bobina a un voltaje de "+12v" y el otro extremo a un voltaje de "tierra virtual" de "0v" con referencia al segundo puente rectificador.

El terminal de "tierra virtual" no es la misma tierra del puerto DB-25F de aquí que radica una de las dos normas de seguridad para prevenir cualquier daño al CPU en condiciones de sobrecarga del periférico, o la dependencia de una carga inductiva.

La segunda norma de seguridad es el dispositivo optoacoplador como interfase de aislamiento entre CPU y los elementos de carga.

La ventaja de implementar esta interfase electrónica es que no existe "conexión física" entre la Unidad de Proceso Central, la interfase electrónica, y los elementos de carga.

La comunicación del CPU a través de su puerto centronics con la interfase electrónica es por medio de un rayo de luz infrarrojo y no existe cableado presente.

La máxima corriente de salida por cada toma, es de aproximadamente seis amperios, suficiente corriente para la mayoría de las aplicaciones con motores AC de baja potencia.

Si se requiere manejar cargas mayores en potencia, se tendrá que recurrir a dispositivos externos de control más sofisticados como triacs de conmutación.

Producida la energización del microrelé este procede instantáneamente a conmutar su contacto normalmente abierto que se encuentra en serie con la respectiva toma de carga, y ahora pasa a estar normalmente cerrado.

Es así como se produce la energización de la carga y se inicia ahora el estado de control por software.

El diagrama de la interfase electrónica de baja potencia, con todos sus componentes electrónicos se lo representa en la figura 2.2, este diagrama se encuentra conceptualizado para un funcionamiento en *lógica negativa*, por lo tanto depende de las habilidades del diseñador del software para disponer los niveles lógicos correctos a la salida del puerto centronics DB-25F.

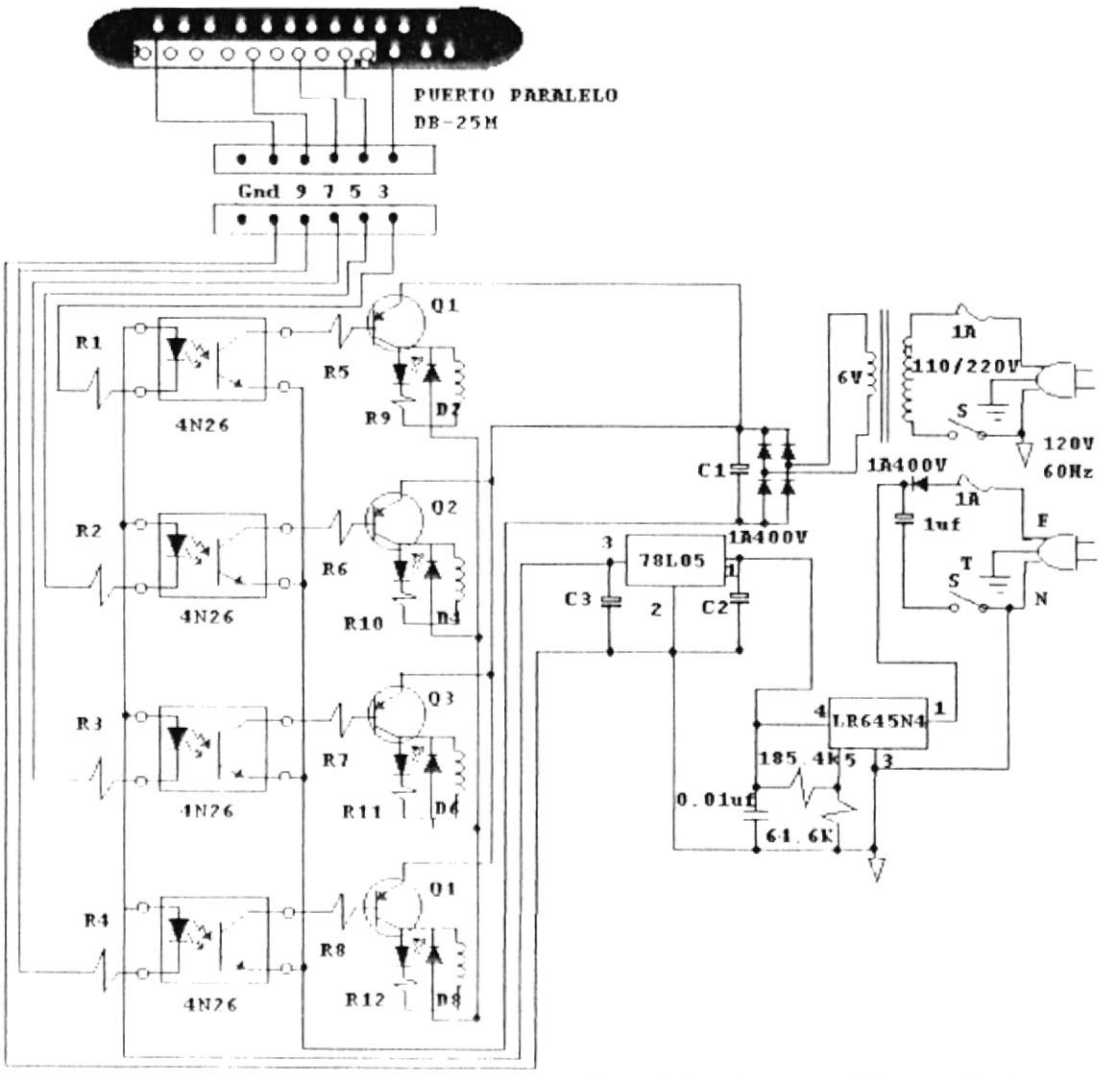


Figura 2.2 Interfase Electrónica de Control Optoaisladora

Los componentes de la interfase electrónica optoaisladora y de potencia son:

Resistencias (1/4 W, 5% tolerancia)

- R1 - R4 560 Ohmios
- R5 - R8 5.1 KOhmios
- R9 - R12 1.2 KOhmios

Condensadores

- C1 - 2200 µF/25V electrolítico
- C2 - 470 µF/25V electrolítico
- C3 - 0.01 µF/25V cerámico

Semiconductores

- D1 - D3 - D5 Diodos Led rojos (20mA)
- D2 - D4 - D6 - D8 Diodos rectificadores (1N4148)
- Q1 - Q4 Transistor BJT PNP 2N3906

Integrados

- IC1 - Regulador fijo de voltaje LM7805CT (ECG 960)
- Optoaislador 4N26 (ECG 3081)
- Inversor Hex Schmitt Trigger (74LS14)

Adicionales

- Transformador 110V/220V tomas

de 6V 100mA y 12V 300 mA en el secundario

- Porta fusibles de 5 A

- microrelés 12VDC 10A 1-2 contactos NA
- tomas de alimentación de cargas
- toma de alimentación alterna (120 Vac)

2.2.1 RELES DE ESTADO SOLIDO SSR.-

Los relés estáticos de estado sólido o llamados SSR (solid state relays), son dispositivos electrónicos que se utilizan como interruptores de conmutación de potencia como los tiristores y transistores, en vez de utilizar los contactores electromecánicos. Su funcionalidad se sintetiza en: activar el dispositivo de conmutación de la carga de potencia; una vez que es recibida las respectiva señal de habilitación de control [SCRM80].

Estas señales de control pueden provenir de circuitos lógicos de disparo los cuales tienen por función controlar motores, solenoides, calefactores etc. La conmutación en el circuito de fuerza puede realizarse por dispositivos especializados de potencia como: **TRIAC´s, DIAC´s, LASCR´s, SCR´s, IGBT´s, MOSFET´s o BJT´s [SCRM80].**

Los relés de estado sólido ofrecen varias ventajas notables en comparación con los ya comúnmente conocidos relés y contactores electromecánicos, estas características las podemos describir como [SCRM80]:

- aislamiento entre el circuito de entrada y el de salida
- soportan altas corrientes y voltajes sin producir arcos de terminal
- alta velocidad de conmutación
- resistente al desgaste
- inmunes a las vibraciones
- no producen ruido en las conmutaciones

Los relés SSR son dispositivos de un solo polo, una posición. Esto indica que un solo SSR no puede conmutar al mismo tiempo varios circuitos independientes de potencia.

Las características de encapsulamiento del módulo SSR se presentan como unidades completamente herméticas, e imposibles de reparar en caso de daño. Estos dispositivos pueden ser utilizados para aplicaciones industriales de modo que su costo es alto.

Las interfases optoaisladoras tomadas a consideración para el manejo de cargas de potencia podrían ser: MOC3010 o MOC3031, 4N33 (MOTOROLA).

En la Tabla 2.3 se relacionan las principales especificaciones de los relés de estado sólido "SSR" con sus respectivas interfases de potencia.

int.	TIPO	SALIDA	Viso: Voltaje de aislamiento entrada-salida =7500 V [SCRM 80]											
			I_{in} [mA]			Vin [V]			I_L [A]			V_L[V]		
			Min	Nom	Max	Min	Nom	Max	Min	Nom	Max	Min	Nom	Max
A	DC/AC	TRIAC	15	10	60	3	9	15	0.1	10	20	-	115	400
B	DC/DC	BJT	10	15	100	2	9	15		5	15		24	60
C	DC/DC	MOSFET	10	20	100	2	9	15	-	15	27		18	60

Tabla 2.3 Interfases SSR y Dispositivos de Potencia

Simbologías de convención:

- Corriente de entrada DC (I_m)
- Voltaje de entrada DC (V_m)
- Corriente de carga RMS o DC (I_L)
- Voltaje de carga RMS o DC (V_L)
- Voltaje de aislamiento entrada-salida (Viso)

Como podemos observar en la Tabla 2.3 los datos técnicos de entrada para cada una de las interfases; operan con un voltaje nominal DC de +9V y un voltaje máximo nominal de +15V, pueden ser fácilmente adaptados para el control de voltajes AC, además utilizan dispositivos optoacopladores como elementos "insolate" (de aislamiento) entre la entrada y la salida del circuito.

Otra ventaja es que están provistos de relés de protección contra voltajes excesivos de entrada, inversión de polaridad, sobrecargas, y transientes de alto voltaje. En la figura 2.3 (a) se muestran los componentes del SSR [SEMI87].

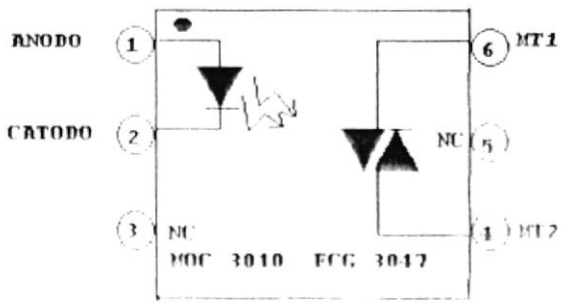


Figura 2.3 (a) Modelo del Optoaislador DIAC MOC 3010

La representación en diagramas de bloques de una interfase optoaisladora del tipo SSR se muestra en la figura 2.3 (b):

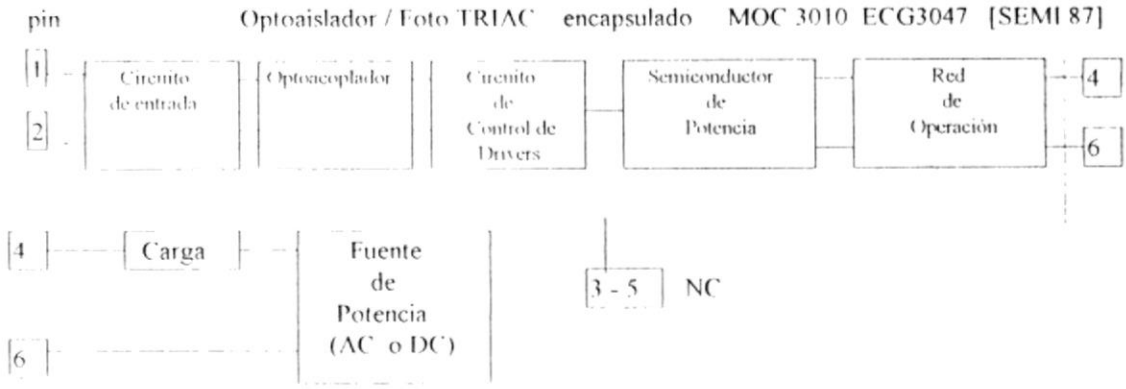


Figura 2.3 (b) Modelo de Bloques Optoaislador Triac MOC 3010

2.2.1.1 RELE DE ESTADO SOLIDO INTERFASE OPTOAISLADORA ESTANDAR MOTOROLA MOC 3010.-

El SSR MOC 3010 utiliza un triac aislado como dispositivo de conmutación previa, su aplicación está diseñada para voltajes alternos y frecuencias de operación de 60 Hz.

En la Tabla 2.4 se determina los datos de operación de la interfase optaisladora MOC 3010 (ECG3047) y MOC 3031 (ECG 3049).

La máxima corriente directa que puede soportar el dispositivo triac MOC 3010 es de 10 mA; de manera que se requerirá de la ayuda de un dispositivo de conmutación de potencia a la salida; controlado por la interfase optoaisladora [SEMI87].

Fototiristores		Tasa Operación		Máx. Oper. LED		Fototiristor Operación [SEMI 87]				
FCG TIPO (mA)	Config. de Salida	Voltaje aislamt. Viso(V)	Potencia Pt(mW)	Corriente Directa IF(mA)	Voltaje Inverso VR(V)	V_{DRM} (V)	I_T RMS (mA)	I_{FT} (mA)	$V_{T(on)}$ 100mA	I_{HOLD}
3047	TRIAC	7500	330	50	3	250	100	10	3.0	0.1
3049	TRIAC circuito de cruce por cero	7500	330	50	3	250	100	15	3.0	0.1

Tabla 2.4 Especificaciones de Operación de los Optoaisladores

En la Tabla 2.5 se obtienen los datos técnicos de operación del dispositivo triac de "conmutación de potencia en la carga" Q4015L5 (TRIACS de aislamiento de compuerta a terminal principal MT2 "Isolated Tab"). Para nuestro diseño se requiere un voltaje de operación nominal de 115Vac 10A.

Para el caso de que se necesite alimentar una carga que trabaje con voltajes de líneas como 220V alternos, entonces no existe impedimento para hacerlo; se debe tener presente no sobrepasar el requerimiento máximo nominal de 400Vac / 20A del triac.

VRRM (Amps) DC o Pico Volts (400 V)	IT RMS Máxima corriente Directa Q4015L5 ECG 56020 [SCRM 80]
25 A I_{GT} Min (mA) Cuadrante I & III	50
I_{GT} Min (mA) Cuadrante II & IV	50
V_{GT} Max (V)	2.5
I_{Surge} Max(A)	200
I_{hold} Min (mA)	70
V_{on} Max (V)	1.8
V_{GM} (V)	+ /- 10
P_G Av(W)	0.5
Temperatura Operación T_J °C	- 40 a +110
dv/dt V/usec	60
Cuadrante de Operación	I, II, III

Tabla 2.5 Especificaciones del TRIAC Q4015L5 - ECG56020

Al utilizar como dispositivo activo al triacs Q4015L5, es necesario proteger al elemento contra disparos forzados por picos de voltajes; por esta razón es conveniente incluir en el diseño una "red snubber" de protección contra transientes de voltajes. La naturaleza de la carga afecta indirectamente al dispositivo semiconductor más aún si es predominantemente inductiva como motores o solenoides [SCRM80].

Según las necesidades de control que se requiera a veces es necesario que se incorpore a la circuitería electrónica la "***detección de cruce por cero***" si ese fuese el caso entonces

se puede incluir fácilmente esta característica; sustituyendo el optoaislador de tipo estándar (MOC3010) por un MOC 3031.

Veamos un diagrama esquemático de como utilizar adecuadamente una interfase optoaisladora estándar SSR MOC3010 en combinación con la interfase de conmutación de potencia "TRIACS Q4015L5" y la "red snubber", esta interfase se encuentra diseñada para operar en lógica positiva (figura 2.4).

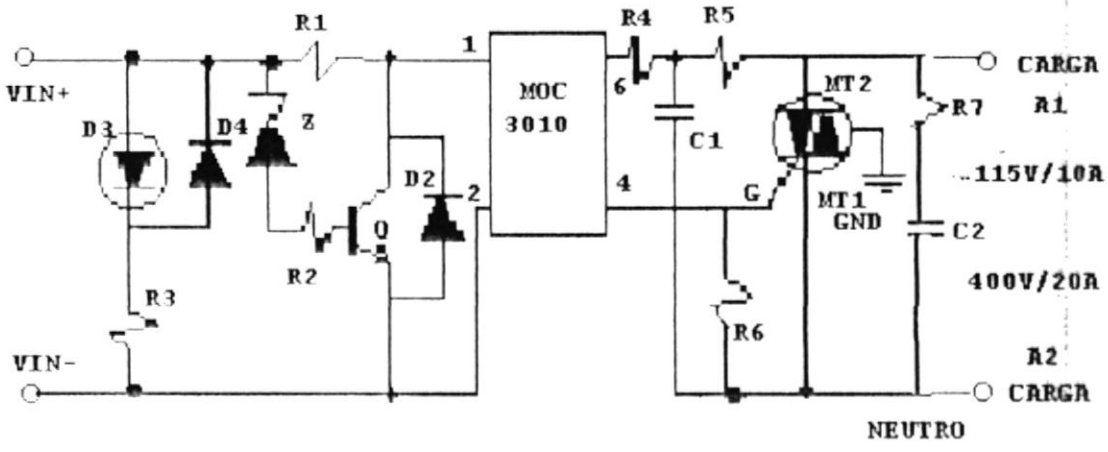


Figura 2.4 SSR DC/AC MOC3010 Optoaislador y TRIAC de Potencia

2.2.1.2 RELE DE ESTADO SOLIDO INTERFASE OPTOAISLADORA 4N33.-

El SSR 4N33 utiliza como dispositivo inicial de conmutación fototransistores. Dentro del encapsulado este dispositivo electrónico posee dos transistores en configuración Darlington (cascodo), el transistor de entrada es utilizado como driver "emisor regenerativo" y el segundo dispositivo es manejado como un transistor de conmutación a la salida.

El diodo infrarrojo emisor emite el haz de luz a la base del driver y este a su vez se encarga de que conduzca el transistor de salida.

Para poder manejar cargas de potencia es necesario de la presencia de un dispositivo semiconductor como por ejemplo: el transistor de potencia bipolar de juntura conocido como "BJT" y particularmente el 2N3055.

Se lo utiliza para realizar el control de paso de corriente en la carga, su uso es especializado para aplicaciones DC. Los datos técnicos proporcionados en la Tabla 2.3 muestran como se puede trabajar con este dispositivo con un voltaje de operación nominal de 24VDC/5A, así como también se lo puede utilizar para manejar corrientes nominales de hasta 15A y voltajes de carga nominales hasta 60VDC.

La frecuencia máxima de conmutación admisible está en el orden de los 1000 Hz, ésta frecuencia de operación está limitada por las capacitancias parásitas del optoaislador (4N33 - FCG 3083).

Un dispositivo como un "varistor" es un elemento siempre requerido como supresor de picos generalmente cuando se utiliza cargas inductivas.

Veamos un diagrama esquemático de como utilizar adecuadamente una interfase optoaisladora SSR 4N33 (ECG3083) en combinación con la interfase de conmutación de potencia "BJT 2N3055 y el "varistor", esta interfase se encuentra diseñada para operar en lógica positiva (figura 2.5).

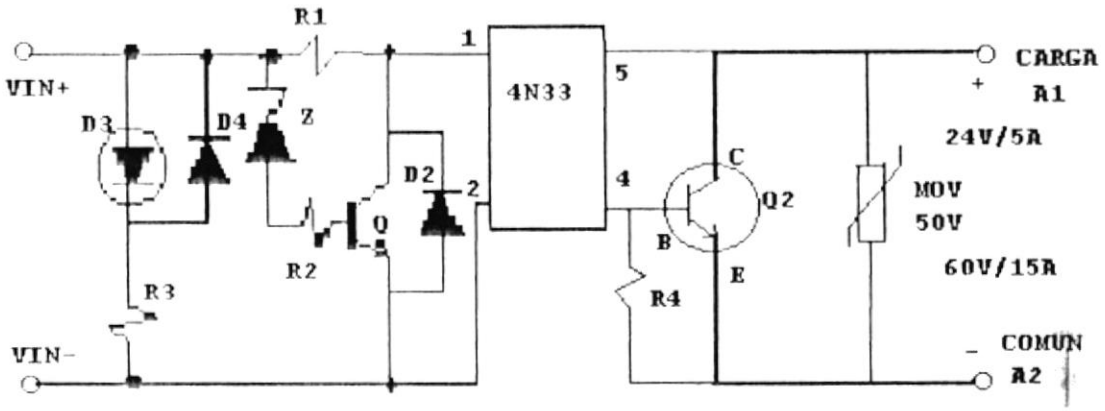


Figura 2.5 SSR DC/DC 4N33 con Transistor BJT de Potencia

Podemos mejorar las condiciones de salida en lo referente a voltaje y corriente continuo sin necesidad de modificar la interfase optoaisladora 4N33, la modificación a realizar es el transistor BJT por un MOSFET de enriquecimiento canal N.

Se utiliza un transistor del tipo MOSFET de potencia y es manejado como un dispositivo de conmutación, al igual que la interfase de potencia 2N3055, esta interfase electrónica de potencia es diseñada para trabajar bajo operaciones de carga DC.

La Tabla 2.5 especifica los datos técnicos nominales de operación 48V/15A, de ser necesario se lo puede utilizar para aplicaciones máximas nomimales de 60V/27A; aún bajo estas condiciones el dispositivo se encuentra en su área segura de operación.

Este dispositivo admite frecuencias de operación superiores a los 60 Khz, gracias a la incorporación del optoacoplador 4N33 (ECG 3083) en configuración "cascodo".

Es necesario para este diseño de disponer de las características de lógica Schmitt Trigger para la conformación de la señal de compuerta, el aislamiento de la compuerta en el transistor de canal N proporciona la ventaja de tener una caída de tensión muy baja para cuando el dispositivo se encuentra en estado de conducción (ver figura 2.6).

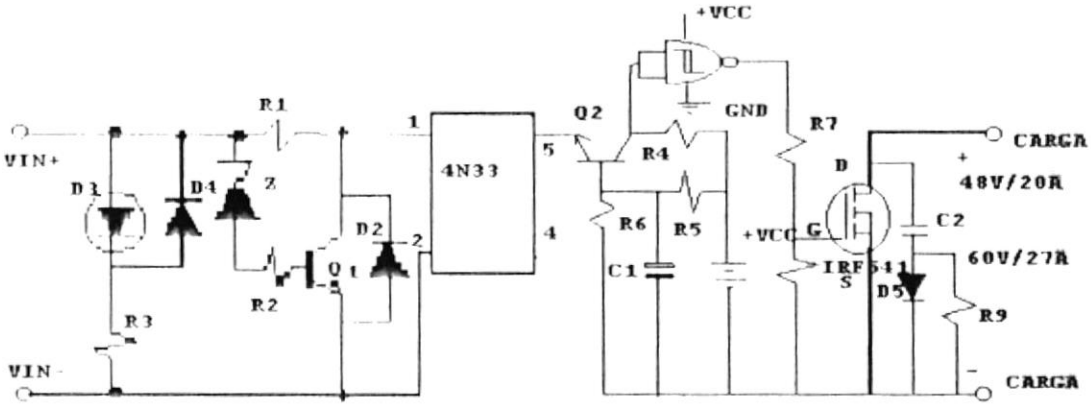


Figura 2.6 SSR DC/DC IRF541 con MOSFET de Potencia

2.3 FUNCIONAMIENTO DE LA INTERFASE ELECTRONICA SSR DC/AC MOC 3010 - Q4015L5 DE MEDIA POTENCIA.-

La interfase electrónica SSR DC/AC MOC 3010 - Q4015L5 mostrada en la figura 2.4 está implementada con un Triacs de compuerta aislada y controlado por el Diac del integrado MOC 3010.

Al conmutar un voltaje de control DC a la entrada de la etapa; involucra que conduzca el optoacoplador y este provoca a su vez una circulación de corriente de compuerta I_G y el triac conduce, permitiendo el paso de corriente a través de la carga.

El triac de aislamiento requiere de una corriente de compuerta I_{GT} mínima del orden de 50 mA para su disparo. El Diac MOC3010, por su parte, tiene una capacidad de manejo de corriente en el orden de los 100 mA, de modo que se inyecta suficiente corriente para disparar prácticamente cualquier Triac.

La resistencia R_1 tiene por función fijar la corriente de compuerta del Triac. La red snubber RC formada por la resistencia R_7 y el capacitor C_2 evita que los cambios instantáneos dv(t)/dt de la fuente alterna dispare al Triac; cuando este se encuentra en su estado de apagado.

La rapidez de variación máxima aceptada por el Triac Q4015L5 es típicamente de 60 V/μs

Si se conmuta sólo cargas puramente resistivas, entonces la red snubber puede ser omitida. La red de atraso de fase formada por la resistencia R_5 y el capacitor C_1, permite una conmutación confiable de cargas inductivas como motores, solenoides relés, transformadores, etc.

Sabemos que un Triac deja de conducir cuando la corriente de carga es cero o cuando la corriente de mantenimiento I_H es menor a la indicada en las hojas técnicas. Como la carga puede ser de naturaleza resistiva, resistiva - inductiva o inductiva, existirá un "cruce por cero" de la corriente de carga, razón por la cual el Triac pasa a su estado de apagado.

Cuando esto ocurre el Triac pasa de su estado de saturación (1.6 V entre terminales principales MT1 - MT2) a su estado de apagado 162.63 V_{pico} para un voltaje de alimentación de 115 V_{rms}.

Se puede disparar al Triac en cualquier semiciclo del voltaje de línea. La resistencia R_5 controla la constante de tiempo del circuito de control del diac.

Esto significa poder cambiar el ángulo "ø" de disparo en el punto al cual el diac conduce y dispara al Triac. Se tiene como resultado un control sobre el ángulo de conducción del Triac. De esta manera se puede regular la corriente en la carga.

En condiciones normales, el disparo del Triac no se encuentra "sincronizado" con los puntos de cruce por cero de la corriente o la tensión de la carga. De manera que la conducción de la interfase de potencia puede producirse en cualquier punto de la forma de onda de alimentación.

Si el disparo del Triac ocurre cerca de un pico positivo o negativo del voltaje de alimentación, punto en que la rapidez de respuesta es muy alto [dv(t)/dt] entonces repercute a través de la carga; circulando una corriente instantánea muy elevada. Esto perjudica y origina una gran cantidad de interferencia en el elemento.

Para prevenir que esto suceda, el MOC 3010 puede ser reemplazado por un optoacoplador con detector de cruce por cero tal como el MOC3031 (ECG 3049) o similares.

En este caso en particular el capacitor C_1 debe ser retirado y la resistencia R_5 suprimirse y colocar un puente de cero ohmios (ver figura 2.7).

La señal de control a la entrada de la interfase electrónica se la puede representar por una batería de 9 VDC, la salida de un circuito lógico TTL, CMOS, o en general el pulso de control de un microprocesador.

La carga debe energizarse cuando la señal de control (V_{IN}) se encuentre en un nivel alto "1" y desenergizarse cuando el voltaje de entrada sea menor al de umbral de disparo en el optoaislador (2 ó 3V), o voltajes negativos o un voltaje superior a los +14 V.

El circuito electrónico conformado por el diodo zener D_1 y el transistor Q1, protege al dispositivo MOC3031 contra voltajes de control mayores a 15 V. La presente interfase se encuentra diseñada para operar en lógica positiva.

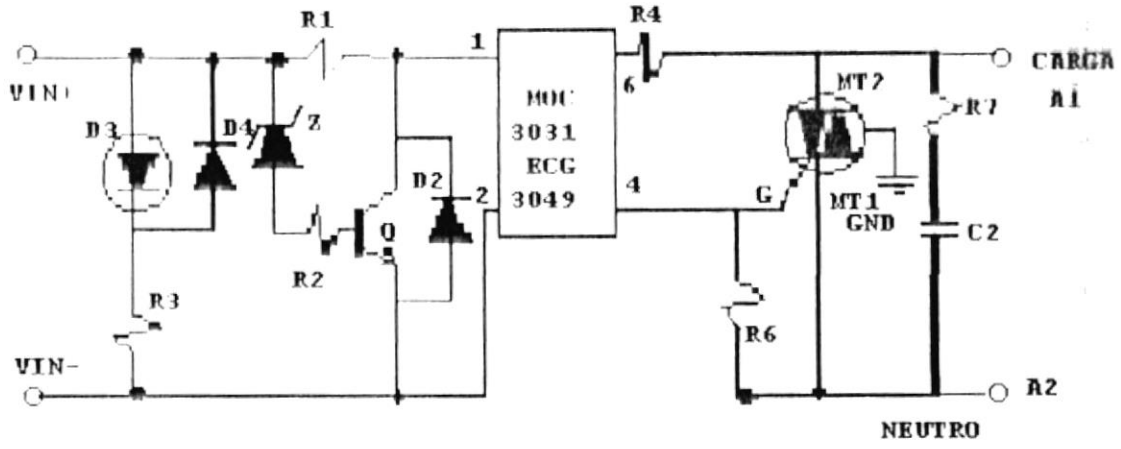


Figura 2.7 SSR DC/AC MOC3031 Optoaislador con Detector de Cruce por Cero e Interfase TRIAC de Media Potencia

2.3.1 ANALISIS DC DE LA INTERFASE ELECTRONICA SSR DC/AC MOC 3031 - Q4015L5.-

El diseño de la interfase electrónica SSR DC/AC MOC3031 (figura 2.7) está basado en las ecuaciones de carga y en las especificaciones técnicas de los manuales.

Al aplicarse un nivel lógico "1" V_{IN} = 5 voltios a la entrada del circuito, el LED D_3 se polariza directamente por lo tanto este se enciende e indica que una señal de entrada está presente en el circuito, la corriente que enciende al LED D_3 se la determina :

$$I_{R3} = \frac{V_{IN} - V_{LED}}{R_3} = \frac{5V - 1.5V}{2.2K} = 1.59\ mA \qquad \textbf{(ec.2.1)}$$

El voltaje de entrada V_{IN} puede estar en el rango de operación entre +5V a +14V dc. Permite utilizar lógica TTL y CMOS dependiendo de la interfase de acople que utilice el sistema.

En el caso de que se cometa un error de conceptualización sobre el funcionamiento del circuito electrónico esto es se aplique un voltaje de inverso a la entrada, entonces el circuito protege contra voltaje inverso al diodo infrarrojo emisor del MOC3031 y aplica una corriente de a través de la resistencia R_3 de la magnitud:

$$I_{R3} = \frac{-V_{IN} - V_{D1}}{R_3} = \frac{5V - 0.7V}{2.2K} = 1.95\ mA \qquad \textbf{(ec.2.2)}$$

En el caso de que el voltaje V_{IN} de entrada sea mayor a 15 voltios, entonces se enciende el diodo zener y el transistor se satura. Las ecuaciones que gobiernan este funcionamiento lo expresamos:

$$I_R = \frac{V_{D1} - V_Z - V_{BE}}{R_2} = \frac{15V - 14V - 0.7V}{0.27K} = 1.11\ mA \qquad \textbf{(ec.2.3)}$$

$V_{IN} - V_{CE} = I_{R1} R_1$ ecuación de línea de carga de **(ec.2.4)**

corriente de saturación: $I_C = I_{R1} = \left.\frac{V_{IN}}{R_1}\right|_{V_{CE} = 0} = \frac{5V}{0.35K} = 14.29mA$

voltaje de corte : $\left. V_{IN} \right|_{I_{R1} = 0} = V_{CE} = 5V$

corriente de disparo del optoaislador MOC 3031:

$$I_F = \frac{V_{IN} - VD_{3031}}{R_3} = \frac{5V - 1.2V}{0.33K} = 11.52\ mA \qquad \textbf{(ec.2.5)}$$

En la sección de APENDICES se realiza una simulación en PSPICE de la interfase *SSR DC/AC MOC3031* (figura 2.7), además se determina la lista de resultados de la simulación, diagrama de la interfase, y curvas de voltaje y corriente en la carga.

Lista de componentes de la interfase SSR DC/AC MOC3031 figura 2.7:

Resistencias (0.5% tolerancia)

- R1- 330 Ohmios / 1 W
- R2- 270 Ohmios / 0.25 W
- R3- 2.2 Kohmios / 0.25 W
- R4- 220 Ohmios / 0.25 W
- R6- 10 Kohmios / 0.25 W
- R7- 39 Ohmios / 0.25 W

Condensadores

- C1- 0.01 µF/160V poliéster
- C2- 0.01 µF/160V poliéster

Semiconductores

- D1- Diodo Zener de 14V/ 0.5 W (1N5244B ECG 5023A)
- D2- D4 - Diodos rectificadores (1N4004)
- D3 - LED indicador de entrada, rojo (20 mA)
- Q1 - Transistor BJT NPN 2N3904
- Q2 - Triac Q4015L5 (ECG 56020) Tab aislado 400V/25A

Integrados

- U1 - Optoaislador MOC 3031 (ECG 3049), detector de cruce por cero

Adicionales

- Placa disipadora térmica para el Triac Q4015L5
- Porta fusibles de 20 A

En la sección de APENDICES se diagrama una interfase digital de control que conectada apropiadamente al puerto I/O DB-25F proporciona las señales de energización - desenergización sobre 56 cargas. Esta interfase se encuentra simulada en el software digital LOGIC WORK. Al utilizar ésta interfase digital y en combinación con una interfase de potencia diseñada en lógica positiva, el aprovechamiento es mayor en lo que respecta al número de cargas, una restricción limitante del diagrama de la figura 2.2.

CAPITULO III

DESCRIPCION DEL PUERTO CENTRONICS DB-25F DE UN PC COMO INTERFASE CONTROLADORA

3.1 PUERTO CENTRONICS DB-25F DE UN PC.-

La utilización del puerto Paralelo o Centronics DB-25F de un PC; hoy en día se está utilizando muy frecuentemente para el control de sistemas electrónicos de potencia o para dispositivos de baja, media y alta tensión como motores DC o AC, estas cargas monofásicas necesitan de un sistema controlador de mando que se encargue de activar un dispositivo optoacoplador y este de disponer el disparo en la compuerta de un triac para ejercer control sobre una carga AC.

La ventaja que dispone utilizar el puerto paralelo de un PC, es que no necesitamos abrir el computador para conectarnos a los buses internos del microprocesador y luego disponer de un cableado interior - exterior, para simplificar todo esto, lo que hacemos es conectarnos directamente al puerto paralelo externo que posee el computador por medio de un conector DB-25 M. Me permito detallar algunas alternativas referentes al uso de los buses internos del computador si es nuestro interés de utilizarlo en vez del puerto centronics.

Los diseñadores que hayan trabajado con el puerto paralelo DB-25F leyendo o controlando señales de tipo digital o analógica en un computador personal, saben que en la gran mayoría de los casos esto se realiza por medio de la utilización de una tarjeta interna que tenga las características de "adquisición de datos", estas tarjetas tienen que ser fijadas internamente en el computador a través de las ranuras o "slots" de expansión que disponen los PC, claro está que esto lo hacían las personas que desconocen las

ventajas del uso del puerto paralelo, tales como sus direcciones, datos, control, status, y pines. Los microcomputadores de la clase PC disponen de *tres conjuntos de líneas o buses* por los cuales viaja información de tipo binaria.

Conociendo su modo de operación es posible *ingresar* y *extraer* "*datos digitales*" con el propósito de obtener un sistema de control sobre algún dispositivo.

3.1.1 TIPOS DE BUSES.-

1. Bus Interno.
2. Buses Externos.
3. Líneas de Datos.

1. **Bus Interno:** Los buses internos en un microcomputador están clasificados como *buses de direcciones*, de *datos* y de *control*, y otras señales que son generadas por otros dispositivos mismos del computador.

Las ventajas de conectar directamente una tarjeta de "adquisición de datos" a una ranura interna son [INTI-96]

A. alta velocidad de muestreo.
B. tamaño reducido.
C. aprovechamiento del uso de los buses **ISA, EISA, MCA,VL-VESA, PCI**.
D. reducir los costos operativos.

La energización para alimentar los circuitos de la tarjeta de "adquisición de datos" se obtiene directamente de una de las ranuras internas (slots de expansión) de los terminales

del bus, como por ejemplo pin B3 $+5V_{DC}$ en un Bus ISA de 8 bits, o EISA de 16 bits de datos. Los tipos de buses más utilizados son **ISA, EISA,VL-VESA, PCI y Microcanal**.

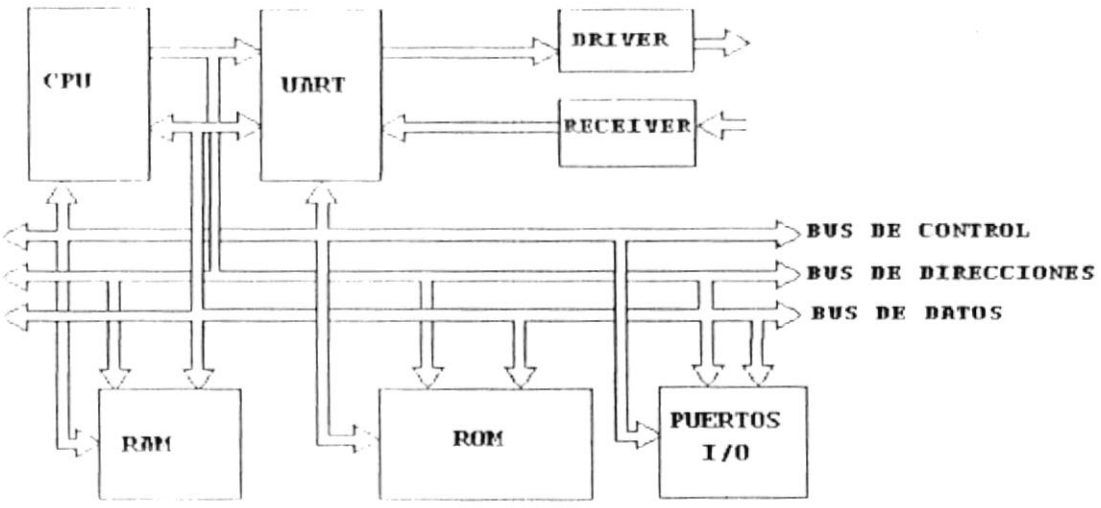


Figura 3.1 Configuración del Sistema de Buses de un PC

En la historia de los computadores IBM y compatibles, existen diferentes tipos de buses que han ido evolucionando, principalmente con el desarrollo de los microprocesadores.

El primer bus, ya legendario pero todavía utilizado, fue el bus ISA de 8 bits, luego apareció el bus ISA de 16 bits, después el MCA exclusivo de IBM (Microchannel), el EISA de 32 bits, el VL-BUS de 32 bits y ahora se habla del bus PCI de 64 bits.

Descripcion de los tipos de buses existentes [INTE96]

A. **Bus ISA 8 bits de datos:** El bus ISA conocido como Industry Standard Architecture, fue el primer bus utilizado en los equipos PC y XT de IBM como sistema de arquitectura abierta. El bus ISA de 8 bits se encuentra en forma de

conector en línea de doble lado y tiene 62 contactos, 31 por cada lado. Sus pines o contactos están enumerados por un lado como A1 hasta A31 y por el otro lado como B1 hasta B31.

Como se puede observar en la figura 3.2, en una ranura o slots se encuentra el bus de datos, el bus de direcciones, las señales de control, de reloj y los voltajes de alimentación. Casi todas estas señales van conectadas al microprocesador a través de circuitos acopladores (drivers o buffers) con el fin de protegerlo de posibles cortocircuitos o conexiones erróneas.

Cuando nos referimos a un bus ISA de 8 bits significa que son 8 bits de datos (de aquí su nombre), el bus ISA posee un bus de direcciones de 20 líneas lo que permite un direccionamiento hasta de 1 Mbyte = 2^{20} /(1024 x1024) MByte.

El bus posee 6 señales de interrupción (IRQ2 a IRQ7), 3 canales de DMA y una señal de reloj de 4.77 Mhz. Aunque cada conector en el bus se supone que trabaja de la misma forma, los primeros PC's fabricados con 8 ranuras requerían tener por lo menos una tarjeta conectada en el slots 8, con el fin de obtener la señal "card selected" (tarjeta seleccionada) en el pin B8. Esta señal era una línea diseñada para recibir un adaptador especial de IBM llamado 3270PC, pero la mayoría de los fabricantes de clones y compatibles no siguió esta configuración [INTE 96].

Señal	Pin	Pin	Señal
Ground	B1	A1	-I/O CHCK
Reset	B2	A2	Data Bit 7
+5VDC	B3	A3	Data Bit 6
IRQ2	B4	A4	Data Bit 5
-5VDC	B5	A5	Data Bit 4
DRQ2	B6	A6	Data Bit 3
12VDC	B7	A7	Data Bit 2
-Card Selected	B8	A8	Data Bit 1
+12VDC	B9	A9	Data Bit 0
Ground	B10	A10	-I/O CHRDY
-SMEMW	B11	A11	AEN
-SMEMR	B12	A12	Address Bit 19
-I/O W	B13	A13	Address Bit 18
-I/O R	B14	A14	Address Bit 17
-DACK 3	B15	A15	Address Bit 16
DRQ 3	B16	A16	Address Bit 15
-DACK 1	B17	A17	Address Bit 14
DRQ 1	B18	A18	Address Bit 13
-REFRESH	B19	A19	Address Bit 12
Clock (4.77 MHz)	B20	A20	Address Bit 11
IRQ 7	B21	A21	Address Bit 10
IRQ 6	B22	A22	Address Bit 9
IRQ 5	B23	A23	Address Bit 8
IRQ 4	B24	A24	Address Bit 7
IRQ 3	B25	A25	Address Bit 6
-DACK 2	B26	A26	Address Bit 5
T/C	B27	A27	Address Bit 4
BALE	B28	A28	Address Bit 3
+5VDC	B29	A29	Address Bit 2
Osc(14.3 MHz)	B30	A30	Address Bit 1
Ground	B31	A31	Address Bit 0

Figura 3.2 Ranura de Expansión Bus ISA 8 Bits de Datos

El oscilador se encuentra ubicado en el pin B30, y entrega una señal de 14.3 MHz (depende del microprocesador que se use). Cuando se debe realizar un reset en el PC, el pin RESET DRV (B2) reinicia todo el sistema. Cuando se recibe una dirección válida, el pin AEN (A11), le indica al sistema que se puede decodificar esta dirección [INTE 96].

El pin -I O CHCK o I O Channel Check (A1), comunica a los circuitos de la tarjeta principal (motherboard), que ha ocurrido un error en la tarjeta de expansión, el signo negativo indica que es una señal activa baja (lógica negativa) [INTE 96].

El pin -I/O CHRDY o I/O Channel Ready (A10) se activa cuando una tarjeta de expansión está lista. Si este pin está en nivel bajo (0), el microprocesador extiende el ciclo del bus generando estados de espera (wait states) [INTE 96].

Las 6 *señales de interrupción por hardware* (IRQ2 a IRQ7), se utilizan por las tarjetas de expansión para demandar atención por parte del microprocesador.

Las interrupciones 0 y 1 no están disponibles en el bus ya que ellas tienen las prioridades más altas del temporizador principal y el teclado. Las señales I/O Read y I/O Write indican que el microprocesador o el controlador de DMA desean transferir datos hacia o desde el bus de datos.

Las señales de lectura y escritura de memoria (-MEMR y -MEMW) le indican a la tarjeta de expansión que el CPU o el DMA van a leer o escribir datos a la memoria principal.

El bus de los antiguos computadores XT poseen tres señales de requisición de DMA y estas son (DRQ1 a DRQ3), permite a la tarjeta de expansión transferir datos hacia o desde la memoria.

B. Bus ISA AT 16 bits: El avance de la tecnología permite modificar el bus ISA de 8 bits por el de 16 bits de datos (microprocesador 80286 Intel)

Consiste en que se le agrega al bus ISA de 8 bits un segundo conector de 36 pines alineado con el primero con nuevas señales

En esencia se agregaron otros 8 bits de datos, más direcciones, cinco interrupciones y cuatro canales de DMA y algunas señales de control. Así mismo, se incrementó la velocidad del reloj a 8.33 Mhz [INTE 96]

Ver esquema de ranura del bus ISA AT 16 bits, slots de expansión figura 3.3.

Señal	Pin	Pin	Señal
Ground	B1	A1	-I/O CHCK
Reset	B2	A2	Data Bit 7
+5VDC	B3	A3	Data Bit 6
IRQ2	B4	A4	Data Bit 5
-5VDC	B5	A5	Data Bit 4
DRQ2	B6	A6	Data Bit 3
12VDC	B7	A7	Data Bit 2
-Card Selected	B8	A8	Data Bit 1
+12VDC	B9	A9	Data Bit 0
Ground	B10	A10	-I/O CHRDY
-SMEMW	B11	A11	AEN
-SMEMR	B12	A12	Address Bit 19
-I/O W	B13	A13	Address Bit 18
-I/O R	B14	A14	Address Bit 17
-DACK 3	B15	A15	Address Bit 16
DRQ 3	B16	A16	Address Bit 15
-DACK 1	B17	A17	Address Bit 14
DRQ 1	B18	A18	Address Bit 13
-REFRESH	B19	A19	Address Bit 12
Clock (4.77 MHz)	B20	A20	Address Bit 11
IRQ 7	B21	A21	Address Bit 10
IRQ 6	B22	A22	Address Bit 9
IRQ 5	B23	A23	Address Bit 8
IRQ 4	B24	A24	Address Bit 7
IRQ 3	B25	A25	Address Bit 6
-DACK 2	B26	A26	Address Bit 5
T/C	B27	A27	Address Bit 4
BALE	B28	A28	Address Bit 3
+5VDC	B29	A29	Address Bit 2
Osc(14.3 MHz)	B30	A30	Address Bit 1
Ground	B31	A31	Address Bit 0

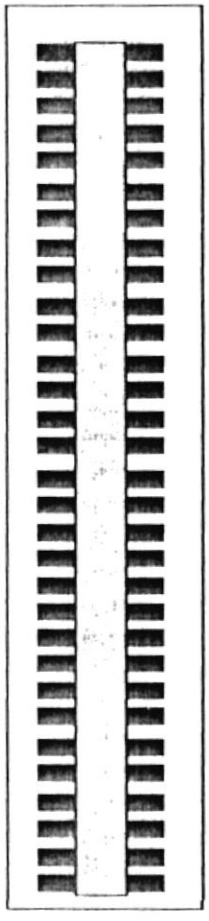


Figura 3.3 Ranura de Expansión Bus ISA AT 16 Bits de Datos

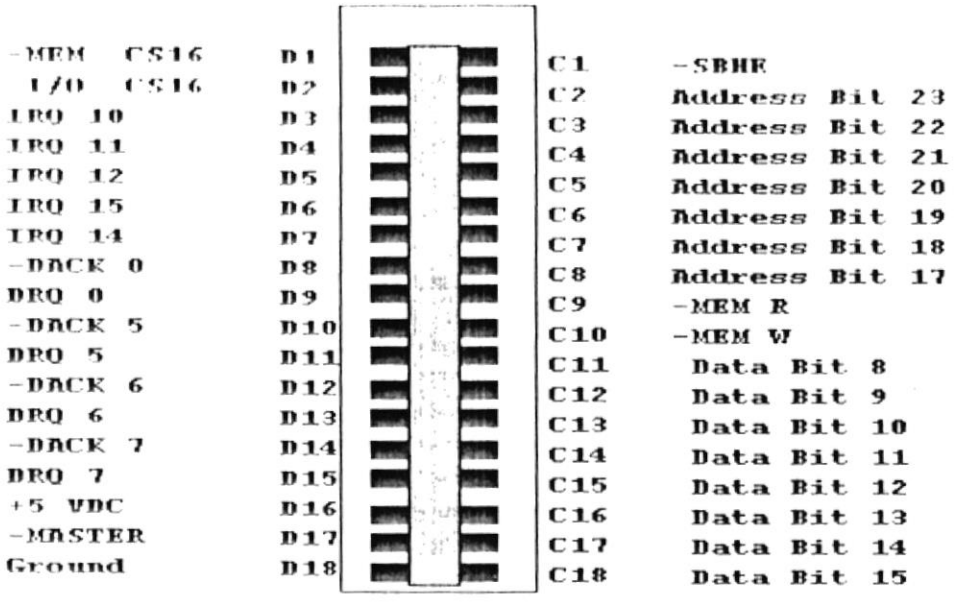


Figura 3.3 Ranura de Expansión Bus ISA AT 16 Bits de Datos

C. **Bus MCA de 32 bits:** Está presente en los microprocesadores de 32 bits de datos como el 80386 y el 80486 de Intel. Diseñado por IBM (Micro Channel Architecture) para la línea de computadores PS/2.

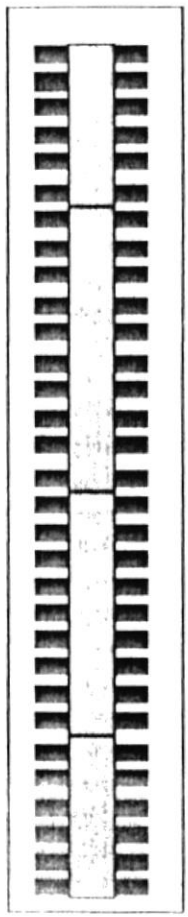
D. **Bus EISA de 32 bits:** El bus EISA (Extended Industry Standard Architecture) de 32 bits de datos para satisfacer las necesidades de mayor velocidad y desempeño que proporcionaban los microprocesadores 80386 y 80486 de Intel.

Este bus posee 30 líneas de direccionamiento, 32 bits de datos, 15 niveles de interrupción y 7 canales de DMA.

Una de las principales ventajas del bus EISA es su capacidad de manejo autónomo del bus (bus mastering), que en pocas palabras podría explicarse como la capacidad para permitir el intercambio de información entre dos periféricos sin la intervención del CPU o microprocesador (μP).

Ver esquema de ranura del bus EISA 32 bits de datos, slots de expansión figura 3.4.

32 bits	16 bits	Pin
Ground	Ground	B1
+5VDC	Reset	B2
+5VDC	+5VDC	B3
Reserved	IRQ 9	B4
Reserved	-5VDC	B5
Key	DRQ 2	B6
Reserved	-12 VDC	B7
Reserved	-0 WAIT	B8
+12VDC	+12 VDC	B9
M -I/O	Ground	B10
-LOCK	-SMEMW	B11
Reserved	-SMEMR	B12
Ground	-I/O W	B13
Reserved	-I/O R	B14
-BE3	-DACK 3	B15
Key	DRQ 3	B16
-BE2	-DACK1	B17
-BE0	DRQ 1	B18
Ground	-REFRESH	B19
+5VDC	Clock(8 33)	B20
-Address29	IRQ 7	B21
Ground	IRQ 6	B22
-Address26	IRQ 5	B23
-Address24	IRQ 4	B24
Key	IRQ 3	B25
Address16	IRQ 2	B26
-Address14	-DACK 2	B27
+5VDC	T/C	B28
+5VDC	BALE	B29
Ground	+5VDC	B30
Address10	Osc(14 3)	B31



Pin	16 bits
A1	-I/O CHCK
A2	Data Bit 7
A3	Data Bit 6
A4	Data Bit 5
A5	Data Bit 4
A6	Data Bit 3
A7	Data Bit 2
A8	Data Bit 1
A9	Data Bit 0
A10	-I/O CHRDY
A11	AEN
A12	Address Bit 19
A13	Address Bit 18
A14	Address Bit 17
A15	Address Bit 16
A16	Address Bit 15
A17	Address Bit 14
A18	Address Bit 13
A19	Address Bit 12
A20	Address Bit 11
A21	Address Bit 10
A22	Address Bit 9
A23	Address Bit 8
A24	Address Bit 7
A25	Address Bit 6
A26	Address Bit 5
A27	Address Bit 4
A28	Address Bit 3
A29	Address Bit 2
A30	Address Bit 1
A31	Address Bit 0

Figura 3.4 Ranura de Expansión Bus EISA 32 Bits de Datos

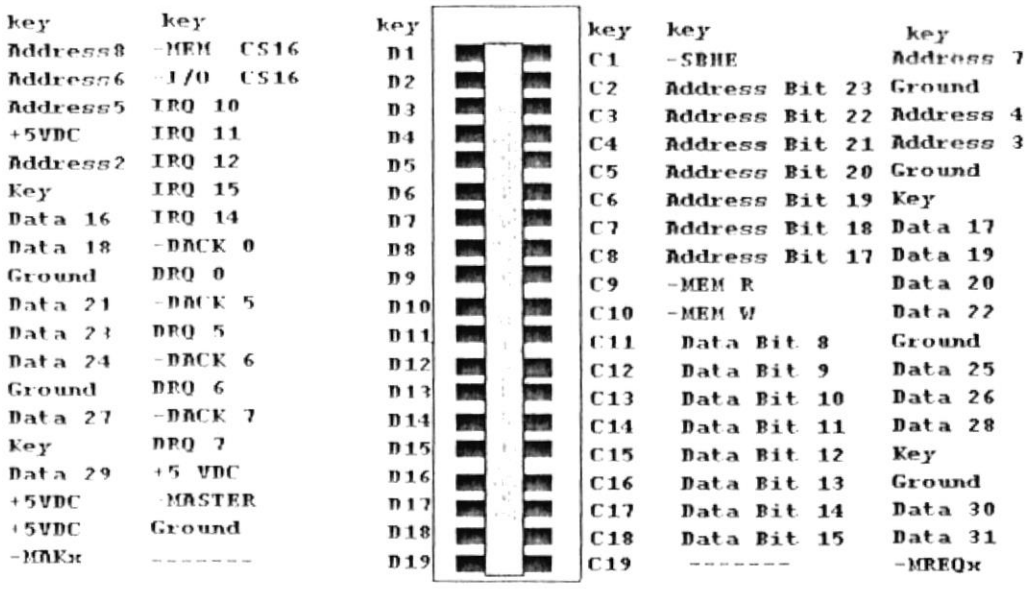


Figura 3.4 Ranura de Expansión Bus EISA 32 Bits de Datos

E. **Bus VL-VESA de 32 bits:** Diseñado por la Video Electronics Standard Association es conocido también como VESA Video Local Bus.

Al igual que el bus MCA utiliza 62 pines en su conector. El bus VL es expandible de 32 a 64 bits permitiendo su utilización en sistemas con el microprocesador Pentium, por medio de otro conector tipo MCA alineado con el primero. Posee dos conectores, uno en cada extremo y para cada bus.

Ver esquema de ranura del bus VL-VESA 32 bits de datos, slots de expansión figura 3.5.

64 bits	32 bits	Pin		Pin
	Data 00	A01		B01
	Data 02	A02		B02
	Data 04	A03		B03
	Data 06	A04		B04
	Data 08	A05		B05
	Ground	A06		B06
	Data 10	A07		B07
	Data 12	A08		B08
	+VCC	A09		B09
	Data 14	A10		B10
	Data 16	A11		B11
	Data 18	A12		B12
	Data 20	A13		B13
	Ground	A14		B14
	Data 22	A15		B15
	Data 24	A16		B16
	Data 26	A17		B17
	Data 28	A18		B18
	Data 30	A19		B19
	+VCC	A20		B20
Data 63	Address 31	A21		B21
	Ground	A22		B22
Data 61	Address 29	A23		B23
Data 59	Address 27	A24		B24
Data 57	Address 25	A25		B25
Data 55	Address 23	A26		B26
Data 53	Address 21	A27		B27
Data 51	Address 19	A28		B28
	Ground	A29		B29
Data 49	Address 17	A30		B30
Data 47	Address 15	A31		B31

64 bits	32 bits	Pin		Pin	32 bits	32 bits
	+VCC	A32		B32	Address 12	Data 44
Data 45	Address 13	A33		B33	Address 10	Data 42
Data 43	Address 11	A34		B34	Address 8	Data 40
Data 41	Address 9	A35		B35	Ground	
Data 39	Address 7	A36		B36	Address 6	Data 38
Data 37	Address 5	A37		B37	Address 4	Data 36
	Ground	A38		B38	-VBAK	
Data 35	Address 3	A39		B39	-BE 0	-BE 4
Data 34	Address 2	A40		B40	+VCC	
LBL 64	N/C	A41		B41	-BE 1	-BE 5
	-RESET	A42		B42	-BE 2	-BE 6
	D/-C	A43		B43	Ground	
	M/-I/O	A44		B44	-BE 3	-BE 7
	W/-R	A45		B45	ADS	

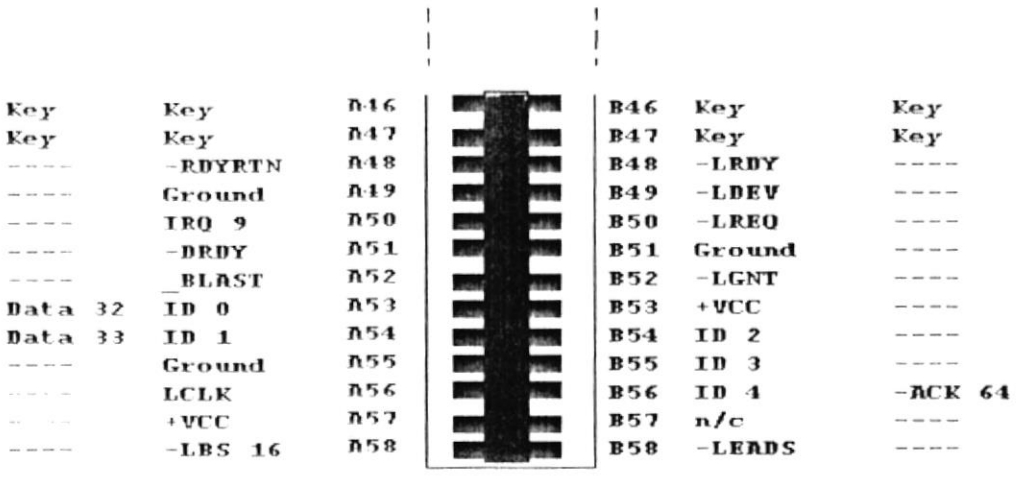


Figura 3.5 Ranura de Expansión Bus VL-VESA 32 Bits de Datos

F. **Bus PCI 64 bits:** Conocido con el nombre de "Componentes de interconexión de periféricos" (Peripheral Component Interconnect). Posee 188 pines, utiliza conectores de 62 pines tipo MCA que se instalan algunas veces alineados con las ranuras ISA o EISA

El bus PCI, de 32 bits es expandible a 64 bits y soporta niveles de voltajes de 3.3 voltios junto a la estándar de 5 voltios

Existen cuatro versiones las que describimos como: 5 voltios y 32 bits, 3.3 voltios y 32 bits, 5 voltios y 64 bits, 3.3 voltios y 64 bits

Las ranuras PCI están dedicadas para los periféricos más críticos que pueden estar conectados directamente a la tarjeta principal o por medio de tarjetas de interfase.

De todas maneras, los sistemas seguirán incorporando ranuras ISA, EISA y VL-VESA ya que el bus PCI es un complemento y no reemplaza totalmente los buses tradicionales.

En la figura 3.6 describimos detalladamente el bus local PCI, como opción a disponibilidad para el usuario como diseñador de sistemas de control, si fuere el caso en que la interfase de control se lo haga a través de los buses locales que dispone un computador personal.

5 Voltios	3.3 Voltios	Pin	Pin	3.3 Voltios	5 Voltios
-12VDC	-12VDC	B1	A1	-TRST	-TRST
TCK	TCK	B2	A2	+12VDC	+12VDC
Ground	Ground	B3	A3	TMS	TMS
TDO	TDO	B4	A4	TDI	TDI
+5VDC	+5VDC	B5	A5	+5VDC	+5VDC
+5VDC	+5VDC	B6	A6	-INTA	-INTA
-INTB	-INTB	B7	A7	-INTC	-INTC
-INTD	-INTD	B8	A8	+5VDC	+5VDC
-PRSNT1	-PRSNT1	B9	A9	Reserved	Reserved
Reserved	Reserved	B10	A10	+3.3VDC(I/O)	+5VDC
-PRSNT2	-PRSNT2	B11	A11	Reserved	Reserved
Ground	Key	B12	A12	Key	Ground
Ground	Key	B13	A13	Key	Ground
Reserved	Reserved	B14	A14	Reserved	Reserved
Ground	Ground	B15	A15	-REST	-RST
Clock	Clock	B16	A16	+3.3VDC	+5VDC
Ground	Ground	B17	A17	-GNT	-GNT
-REQ	-REQ	B18	A18	Ground	Ground
+5VDC	+3.3VDC	B19	A19	Reserved	Reserved
Adr/Dat31	Adr/Dat31	B20	A20	Adr/Dat30	Adr/Dat30
Adr/Dat29	Adr/Dat29	B21	A21	+3.3vdc	+5vdc
Ground	Ground	B22	A22	Adr/Dat28	Adr/Dat28
Adr/Dat27	Adr/Dat27	B23	A23	Adr/Dat26	Adr/Dat26
Adr/Dat25	Adr/Dat25	B24	A24	Ground	Ground
+5VDC	+3.3VDC	B25	A25	Adr/Dat24	Adr/Dat24
C/-BE3	C/BE3	B26	A26	IDSEL	IDSEL
Adr/Dat23	Adr/Dat23	B27	A27	+3.3VDC	+5VDC
Ground	Ground	B28	A28	Adr/Dat22	Dar/Dat22
Adr/Dat21	Adr/Dat21	B29	A29	Adr/Dat20	Adr/Dat20
Adr/Dat19	Adr/Dat19	B30	A30	Ground	Ground
+5VDC	+3.3VDC	B31	A31	Adr/Dat18	Adr/Dat18

Figura 3.6 Ranura de Expansión Bus PCI 64 Bits de Datos (Pág 1/3)

5 Voltios	3.3 Voltios	Pin
+5VDC	+3.3 VDC	B31
Adr/Dat17	Adr/Dat17	B32
C/-BE2	C/-BE2	B33
Ground	Ground	B34
-IRDY	-IRDY	B35
+5VDC	+3.3VDC	B36
-DEVSEL	-DEVSEL	B37
Ground	Ground	B38
-LOCK	-LOCK	B39
-PERR	-PERR	B40
+5VDC	+3.3VDC	B41
-SERR	-SERR	B42
+5VDC	+3.3VDC	B43
C/-BE1	C/-BE1	B44
Adr/Dat14	Adr/Dat14	B45
Ground	Ground	B46
Adr/Dat12	Adr/Dat12	B47
Adr/Dat10	Adr/Dat10	B48
Ground	Ground	B49
Key	Ground	B50
Key	Ground	B51
Adr/Dat8	Adr/Dat8	B52
Adr/Dat7	Adr/Dat7	B53
+5VDC	+3.3vdc	B54
Adr/Dat5	Adr/Dat5	B55
Adr/Dat3	Adr/Dat3	B56
Ground	Ground	B57
Adr/Dat1	Adr/Dat1	B58
+5VDC	+3.3VDC	B59
-ACK64	-ACK64	B60
+5VDC	+5VDC	B61
+5VDC	+5VDC	B62
Key	Key	Key
Key	Key	Key

Pin	3.3 Voltios	5 Voltios
A31	Adr/Dat18	Adr/Dat18
A32	Adr/Dat16	Adr/Dat16
A33	+3.3VDC	+5VDC
A34	-FRAME	-FRAME
A35	Ground	Ground
A36	-TRDY	-TRDY
A37	Ground	gROUND
A38	-STOP	-STOP
A39	+3.3VDC	+5VDC
A40	SDONE	SDONE
A41	-SBO	-SBO
A42	Ground	Ground
A43	PAR	PAR
A44	Adr/Dat15	Adr/Dat15
A45	+3.3vdc	+5VDC
A46	Adr/Dat13	Adr/Dat13
A47	Adr/Dat11	Adr/Dat11
A48	Ground	Ground
A49	Adr/Dat9	Adr/Dat9
A50	Ground	Key
A51	Ground	Key
A52	C/-BE0	C/-BE0
A53	+3.3VDC	+5VDC
A54	Adr/Dat6	Adr/Dat6
A55	Adr/Dat4	Adr/Dat4
A56	Ground	Ground
A57	Adr/Dat2	Adr/Dat2
A58	Adr/Dat0	Adr/Dat0
A59	+3.3VDC	+5VDC
A60	-REQ64	-REQ64
A61	+5VDC	+5VDC
A62	+5VDC	+5VDC
Key	Key	Key
Key	Key	Key

Figura 3.6 Ranura de Expansión Bus PCI 64 Bits de Datos (Pág 2/3)

5 Voltios	3.3 Voltios	Pin		Pin	3.3 Voltios	5 Voltios
Reserved	Reserved	B63		A63	Ground	Ground
Ground	Ground	B64		A64	C/-BE7	C/-BE7
C/-BE6	C/-BE6	B65		A65	C/-BE5	C/-BE5
C/-BE4	C/-BE4	B66		A66	+3.3VDC	+5VDC
Ground	Ground	B67		A67	PAR64	PAR64
Adr/Dat63	Adr/Dat63	B68		A68	Adr/Dat62	Adr/Dat62
Adr/Dat61	Adr/Dat61	B69		A69	Ground	Ground
+5VDC	+3.3VDC	B70		A70	Adr/Dat60	Adr/Dat60
Adr/Dat59	Adr/Dat59	B71		A71	Adr/Dat58	Adr/Dat58
Adr/Dat57	Adr/Dat57	B72		A72	Ground	Ground
Ground	Ground	B73		A73	Adr/Dat56	Adr/Dat56
Adr/Dat55	Adr/Dat55	B74		A74	Adr/Dat54	Adr/Dat54
Adr/Dat53	Adr/Dat53	B75		A75	+3.3VDC	+5VDC
Ground	Ground	B76		A76	Adr/Dat52	Adr/Dat52
Adr/Dat51	Adr/Dat51	B77		A77	Adr/Dat50	Adr/Dat50
Adr/Dat49	Adr/Dat49	B78		A78	Ground	Ground
+5VDC	+3.3VDC	B79		A79	Adr/Dat48	Adr/Dat48
Adr/Dat47	Adr/Dat47	B80		A80	Adr/Dat46	Adr/Dat46
Adr/Dat45	Adr/Dat45	B81		A81	Ground	Ground
Ground	Ground	B82		A82	Adr/Dat44	Adr/Dat44
Adr/Dat43	Adr/Dat43	B83		A83	Adr/Dat42	Adr/Dat42
Adr/Dat41	Adr/Dat41	B84		A84	+3.3VDC	+5VDC
Ground	Ground	B85		A85	Adr/Dat40	Adr/Dat40
Adr/Dat39	Adr/Dat39	B86		A86	Adr/Dat38	Adr/Dat38
Adr/Dat37	Adr/Dat37	B87		A87	Ground	Ground
+5VDC	+3.3VDC	B88		A88	Adr/Dat36	Adr/Dat36
Adr/Dat35	Adr/Dat35	B89		A89	Adr/Dat34	Adr/Dat34
Adr/Dat33	Adr/Dat33	B90		A90	Ground	Ground
Ground	Ground	B91		A91	Adr/Dat32	Adr/Dat32
Reserved	Reserved	B92		A92	Reserved	Reserved
Rreserved	Reserved	B93		A93	Ground	Ground
Ground	Ground	B94		A94	Reserved	Reserved

Figura 3.6 Ranura de Expansión Bus PCI 64 Bits de Datos (Pág 3/3)

2. **Bus Externo:** Existen dos buses externos los mismos que reciben el nombre de puertos; "*paralelo*" y "*serial*" y a través de estos se conectan periféricos externos.

A. El "puerto serial" es conocido como *RS-232* o "puerto de comunicaciones", este puerto maneja datos que pueden ingresar o salir del mismo en forma serial.

Aquí disponemos solamente de una línea de comunicaciones y los datos ingresan o salen de este puerto uno por uno, es decir uno a continuación del otro.

El puerto de comunicaciones es muy útil para cuando uno desea enviar información como archivos de textos y binarios, programas, imágenes, este puerto de adquisición de datos y de control requiere de pocas líneas de conexión, fácil de configurar para sistemas de cualquier de tamaño, los circuitos de adquisición de datos se pueden ubicar a cualquier distancia del computador personal y aún más si se dispone de un módem.

B. El "**puerto paralelo**" o puerto de impresora o conocido también con el nombre de puerto "Centronics", está compuesto por un conjunto de líneas digitales las que describimos a continuación:

- 5 entradas de protocolo.
- 3 salidas de protocolo.
- 8 salidas de datos.
- 8 líneas de tierra.

todas estas líneas en su conjunto permiten el correcto funcionamiento de cualquier interfase que se conecte al computador personal.

Identifiquemos los nombres de etiqueta de cada uno de los pines del puerto paralelo DB-25F de un computador personal, característica que es necesaria para poder operar correctamente un puerto. La Tabla 3.1 describe cada uno de los pines del puerto paralelo DB-25F.

PIN	SEÑAL	FUNCION	I/O	REGISTRO	BIT	LOGICA CAMBIADA
1	- STB	STROBE	I/O	CONTROL	0	Y
	DO	BIT 0	O*	DATA	0	N
3	D1	BIT 1	O*	DATA	1	N
4	D2	BIT 2	O*	DATA	2	N
5	D3	BIT 3	O*	DATA	3	N
6	D4	BIT 4	O*	DATA	4	N
7	D5	BIT 5	O*	DATA	5	N
8	D6	BIT 6	O*	DATA	6	N
9	D7	BIT 7	O*	DATA	7	N
10	-ACK	Acknowledge	I	STATUS	6	N
11	BUSY	Printer Busy	I	STATUS	7	Y
12	PE	Paper End	I	STATUS	5	N
13	SELECT	Printer Select	I	STATUS	4	N
14	-AUTOLF	Autom Line Feed	I/O	CONTROL	1	Y
15	-ERROR	Error	I	STATUS	3	N
16	-INIT	Initialice Printer	I/O	CONTROL	2	N
17	-SELECT I	NSelect Printer	I/O	CONTROL	3	Y
18	GND	GROUND	I	NONE	NONE	NONE
19	GND	GROUND	I	NONE	NONE	NONE
20	GND	GROUND	I	NONE	NONE	NONE
21	GND	GROUND	I	NONE	NONE	NONE
22	GND	GROUND	I	NONE	NONE	NONE
23	GND	GROUND	I	NONE	NONE	NONE
24	GND	GROUND	I	NONE	NONE	NONE
25	GND	GROUND	I	NONE	NONE	

Tabla 3.1 Identificación de las Señales del Puerto Paralelo DB-25F

3. **Líneas de Datos:** Las líneas de datos es el medio por donde el computador transporta la información hacia cualquier periférico externo.

   La información de los datos se agrupan en líneas de 8 bits o 1 byte y utilizan los pines del 2 al pin 7 del conector DB -25F, estos pines los denominamos pines de datos y son etiquetados como D0 (dato 0) hasta D7 (dato 7).

   Como podemos darnos cuenta en total son 8 salidas de datos "D", clasificadas en orden de prioridad significativa de izquierda a derecha, desde el bit más significativo "*MSB*" hasta el menos significativo "*LSB*":

   - DATO 7 (D7) BIT MAS SIGNIFICATIVO "MSB"
   - DATO 6 (D6)
   - DATO 5 (D5)
   - DATO 4 (D4)
   - DATO 3 (D3)
   - DATO 2 (D2)
   - DATO 1 (D1)
   - DATO 0 (D0) BIT MENOS SIGNIFICATIVO "LSB"

   Todas las líneas etiquetadas del puerto representan un estado de información lógico; estos estados están referenciados por niveles lógicos de voltajes TTL.

   Podemos representar a una señal proveniente del puerto paralelo como un estado lógico "alto" cuando existe nivel de voltaje entre el rango de +5 V a +2.5 V, y un estado lógico "bajo" cuando existe un nivel de voltaje entre el rango de +2.4 V a +0.8 V.

Las 8 líneas de tierra que posee el puerto DB-25F tienen dos funciones a cumplir:

A. La primera característica es unificar las tierras entre el computador y la interfase externa, por lo tanto define un potencial común de 0 voltios.

B. La segunda característica es de proveer un blindaje a las líneas de datos de información contra el ruido externo, de esta manera se evita las interferencias y la pérdida de información por efectos capacitivos a los hilos del cable ribbon.

3.2 ENTRADAS Y SALIDAS DE PROTOCOLO DE UN PC.-

La velocidad de procesamiento de datos de un computador es mucho más rápida que cualquier periférico que se conecte, el computador puede transmitir más datos de los que puede recibir.

Es por ello que un periférico necesita disponer de señales especiales de control de estado, esto es para indicarle al computador que detenga momentáneamente el envío de datos; hasta que se encuentre listo nuevamente la interfase para recibirlos y procesarlos nuevamente.

Es así como el computador dispone del tiempo de procesamiento de los datos y puede ejecutar otras tareas asignadas; permanece por lo tanto en un estado de mantenimiento "hold", hasta que la interfase externa le indique por medio de una señal de protocolo que se encuentra listo para recibir otra vez los datos.

Las señales de control conocidas como "señales de protocolo" son:

- strobe ——→ señal de "inicio"
- busy ——→ señal de "ocupado"
- acknowledge ——→ señal de "reconocimiento"

Cuando un computador posee datos que enviar por el puerto paralelo, lo hace acompañado de una señal de "strobe", entonces el periférico externo le responde al computador con una señal de "busy" esta señal está presente hasta que termina de aceptar o procesar todos los datos recibidos; de esta manera el periférico evita que el computador le envíe nuevos datos que no pueda procesar.

Cuando el periférico se encuentra listo nuevamente para recibir datos, entonces le envía al computador una señal "acknowledge" de reconocimiento y se inicia el envío del frame de datos.

3.2.1 DIRECCIONES DEL PUERTO CENTRONICS.-

La mayoría de los computadores personales o compatibles poseen una tarjeta interna multipuertos I/O, nos referimos específicamente a los puertos paralelos etiquetados como LPT1, LPT2, LPT3. Los diseñadores de computadores han tratado de estandarizar las direcciones de los puertos paralelos, no es de extrañarse que un computador difiera de otro en las características de las direcciones de los puertos.

Casi todos los computadores personales (PC) disponen de 1 puerto paralelo y 2 puertos seriales. Cada puerto de un computador está conformado por 12 líneas de salida de información hacia el mundo exterior y 5 líneas de entrada.

Podemos identificar fácilmente que se necesita de 3 bytes o de 24 bits para tener una representación completa del grupo de 17 líneas de información disponibles.

En la figura 3.7 se muestra la representación del grupo de las 17 líneas de información del puerto paralelo de un PC.

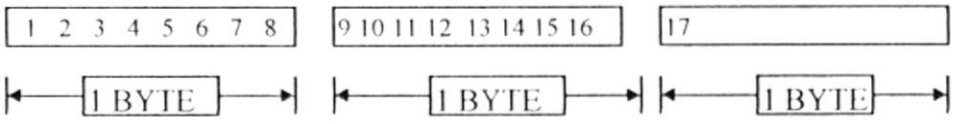


Figura 3.7 Distribución de los Bytes de Direccionamiento

Por lo tanto, a cada puerto paralelo se le deben asignar tres direcciones. En la Tabla 3.2 se hace referencia a las direcciones de los puertos paralelos de un microcomputador.

Se identifica además a cada una de las direcciones respectivas de los puertos LPT1, LPT2, LPT3, así como también los bits de cada byte disponibles, los pines del conector DB-25F, las entradas y salidas.

Puerto Paralelo Dirección Hexadecimal			Bits de 1 Byte (N)	DB-25F Pin	Entrada - Salida
LPT1	LPT2	LPT3			
3BC	378	278	0	2	SALIDA
			1	3	SALIDA
			2	4	SALIDA
			3	5	SALIDA
			4	6	SALIDA
			5	7	SALIDA
			6	8	SALIDA
			7	9	SALIDA
3BD	379	279	0	NA	----------
			1	NA	----------
			2	NA	----------
			3	15	ENTRADA
			4	13	ENTRADA
			5	12	ENTRADA
			6	10	ENTRADA
			7*	11	ENTRADA
3BE	37A	27A	0*	1	SALIDA
			1*	14	SALIDA
			2	16	SALIDA
			3*	17	SALIDA
			4	NA	----------
			5	NA	----------
			6	NA	----------
			7	NA	----------

Tabla 3.2 Puerto Paralelo Direcciones Bases Hexadecimales

En el sistema MS-DOS se hace referencia al puerto paralelo como LPT1 (Line Printer 1), LPT2 y LPT3 para puertos adicionales. Cada puerto paralelo posee una de tres direcciones base posibles: 3BCh, 378h, 278h.

Algunos puertos permiten sólo dos de las tres direcciones, y otros permiten la elección de cualquier dirección, incluída la no estándar.

Cuando un computador arranca lo primero que hace es verificar el estado de todas las direcciones disponibles del computador e inicializa la configuración del BIOS SETUP, para nuestro ejemplo se busca un puerto en cada una de las tres direcciones, en el orden 3BCh, 378h, 278h.

El BIOS determina si un puerto existe o no escribiéndole la dirección existente configurada por la tarjeta de puertos en memoria y luego leyendo lo que escribió en memoria. Si la lectura durante el arranque tiene éxito, entonces el puerto existe.

El primer puerto que se encuentra se lo llama LPT1, el segundo LPT2 y el tercero es LPT3. Dependiendo de la configuración que se haga en la tarjeta de puertos, LPT1 puede ubicarse en cualquiera de las tres direcciones; LPT2 en 378h o 278h y LPT3 únicamente puede estar en la dirección 278h.

La rutina BIOS (Basic Input Output System) almacena las direcciones de los puertos en la memoria RAM (Random Access Memory) del computador.

Estas direcciones de los puertos van especificadas en una tabla de direcciones de la memoria RAM, y están clasificadas desde las direcciones de memoria 0040:0008 hasta 0040:000F empezando con LPT1. Los otros 16 bits superiores compartidos para LPT4 raramente se los utilizan.

Se puede modificar las direcciones de la interfase Centronics, cambiando los valores en la tabla de memoria. Por lo tanto es posible modificar la configuración original.

El objetivo de esto es de configurar el hardware de la máquina, si es el caso de que un programa soporta sólo el puerto LPT1 y el periférico se encuentra conectado en el puerto LPT2.

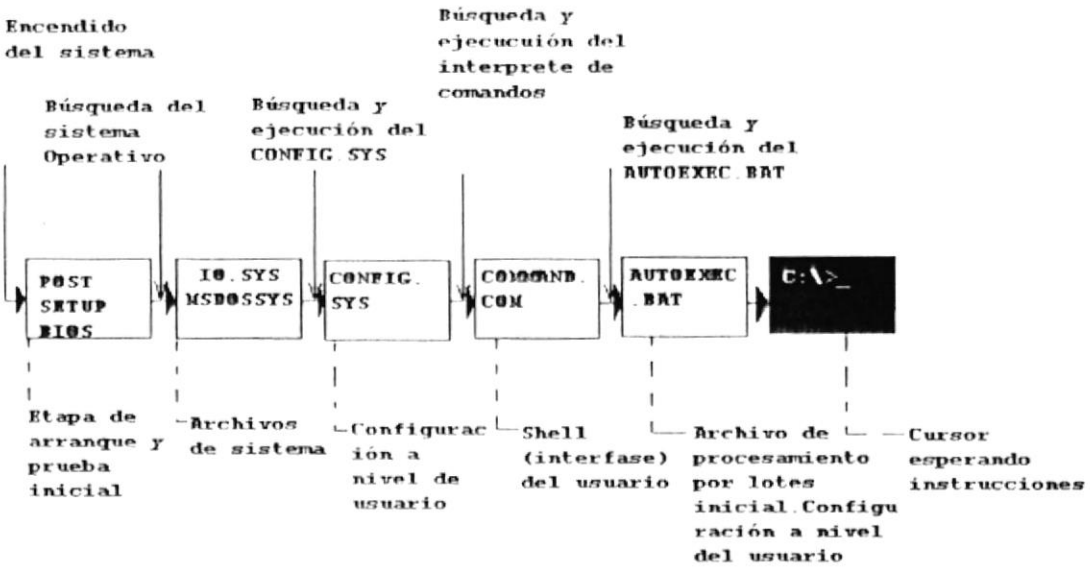


Figura 3.8 Proceso de Arranque de un PC

Dos interrupciones de hardware, IRQ 5 e IRQ 7, generalmente se asocian con los puertos paralelos. Algunos diseñadores de hardware permiten que en las tarjetas se pueda seleccionar una interrupción con un jumper ("puente") o algunas utilizan el setup por software, mientras que en otras ya vienen seleccionadas por seteo o fijación para una determinada interrupción.

Todo "nivel de solicitud de interrupción" se encuentra asociado a un "número de vector de interrupción", así como también a vectores de interrupción y estos indican la "dirección del puerto paralelo convencional".

Por convención estandarizada, *el puerto paralelo LPT1* utiliza la *interrupción IRQ 7* y el *puerto LPT2 la IRQ 5*; en el caso de los computadores antiguos tipo XT el disco duro ocupa la interrupción IRQ 5, de modo que no se pudiera utilizar el puerto paralelo por estar ya ocupado.

Sabemos que el software manejado por interrupción es rápido, pero es importante conocer que la mayoría de los puertos paralelos no utilizan las interrupciones.

En estos puertos, la línea que requiere la interrupción no tiene *seguro* ("latch") esto significa que si un pulso de entrada es corto, entonces es posible que el microprocesador no lo perciba; esto significaría no poder capturar ese dato.

3.2.2 ENTRADAS Y SALIDAS DEL PUERTO.-

Se puede tener acceso al puerto paralelo tanto a través del sistema MS-DOS como por el ROM BIOS.

Las funciones 00, 01 y 02 de la interrupción 17h del BIOS nos permite enviar un byte al puerto Centronics, inicializarla y obtener el estatus de la misma.

La interrupción 21h del DOS correspondiente a la función 05, también nos permite escribir un byte al puerto Centronics, la función 40h nos permite dirigir un bloque de datos a un puerto paralelo.

Para conseguir el acceso a las 17 señales del puerto paralelo, es necesario escribir directamente al puerto.

Para lograr esto, se debe ignorar las funciones del MSDOS y del BIOS y lo que se hace es leer y escribir directamente al puerto de datos; de estatus y al registro de control como se muestra en la figura 3.9.

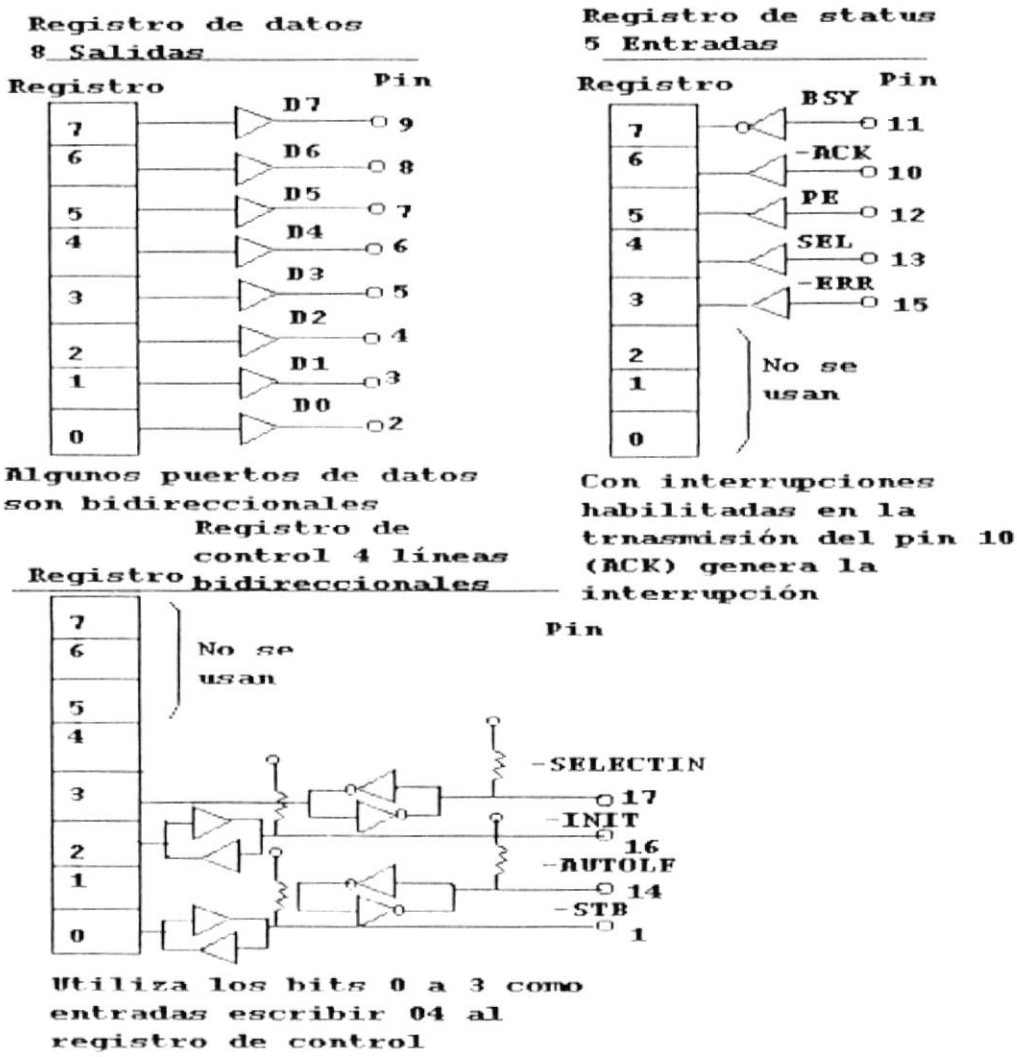


Figura 3.9 Configuración de los Registros del Puerto Paralelo

Conocemos que se puede leer y escribir directamente al puerto de datos, de estatus y al registro de control es por ello que los detallamos a continuación:

1. **Líneas de datos:** Hemos especificado que las líneas de datos son 8 y se encuentran etiquetadas como D7 "bit MSB" hasta D0 "bit LSB" y además están referidas a los pines 9 hasta 2 respectivamente del puerto DB-25F.

   Las líneas de datos como ya lo hemos dicho, su función es transportar los datos hacia el puerto paralelo Centronics. En otras aplicaciones es posible utilizar al puerto paralelo como salidas generales.

   Para poder controlar el estado de los pines 9 al 2 en el puerto, sólo se debe escribir los datos que se desean en el "registro de datos", y la "dirección base del puerto".

   Por ejemplo, para colocar un nivel lógico alto los bits: D0, D4, D5, D6, D7, y un nivel bajo a los bits D1, D2, D3, se escribiría en el registro de datos "0F1h".

   A continuación explicamos como se obtiene este dato hexadecimal. Sabemos que disponemos de 8 líneas de datos en un puerto paralelo, de manera que las identificamos en su orden jerárquico significativo como:

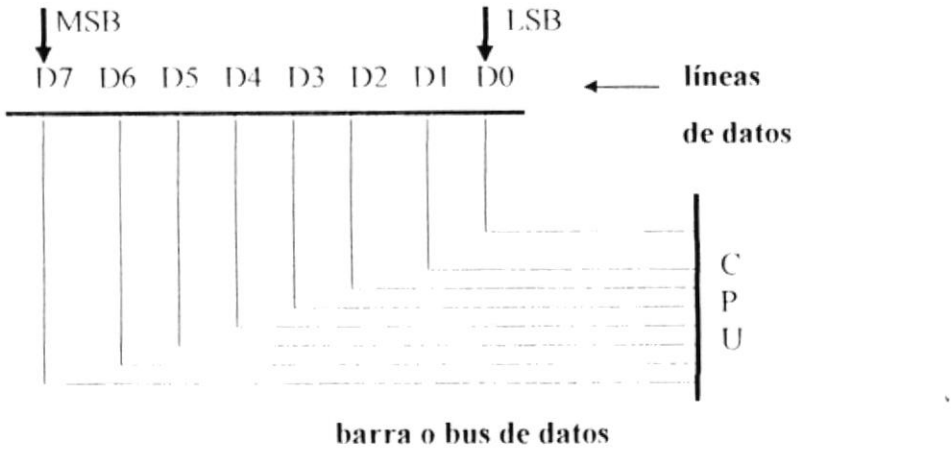


Figura 3.10 Representación de la Línea de Datos del Puerto DB-25F

Para poder representar los bits de datos del puerto paralelo es necesario realizar la conversión binaria de cada uno de los bits de datos, los niveles lógicos de voltaje requeridos son:

D7	D6	D5	D4	D3	D2	D1	D0	línea de datos
1	1	1	1	0	0	0	1_2	valor binario a escribir
7	6	5	4	3	2	1	0	n : número del bit

Si realizamos la conversión a base hexadecimal de este número binario, lo que tenemos es: " 0F1h". La pregunta que nos haríamos es ¿ Porqué usar un número hexadecimal?. La razón es porque para poder escribir una respuesta numérica a la salida del puerto Centronics de un PC, en el diseño del software del lenguaje C se debe especificar el dato en notación hexadecimal para transportarlo a la salida del

puerto, y porque la tabla de memoria se encuentra distribuida en notación hexadecimal.

Si conocemos cual es el puerto activo del computador (generalmente es el puerto paralelo LPT1) y su "dirección base" entonces es posible colocar un dato o un bit a la salida del puerto paralelo.

En el "lenguaje C" la instrucción de programación de salida es:

Outportb*(DataAddress, Data);*

si utilizamos la "dirección base" del puerto paralelo LPT1, y el dato hexadecimal "0F1h", la representación de la estructura de línea de programa es :

Outportb(3BC,HF1);

Dato hexadecimal

Dirección base del puerto

Para poder mostrar un dato a la salida del puerto paralelo LPT1; utilizando el "lenguaje de programación C", la estructura de las líneas de programa debe especificarse como:

Port = peek (Data Address_Memory, Data Position_Port);

outportb (Port, Data Position_Bit);

Describiremos en detalle de la programación en el lenguaje C de la interfase por software en el capítulo 4.

2. **Líneas de Estatus:** La línea de estatus está definida en un número de 5 entradas, las cuales se pueden leer en un registro de estatus que se localiza en la "dirección base +1", si estamos utilizando el puerto paralelo LPT2 su dirección base es 378h; de modo que la dirección base +1 sería 379h.

   El registro de estatus tiene por característica ser un registro únicamente de lectura; de tal manera que si escribimos sobre este registro, no se ve afectado por proceso alguno. Las cinco líneas de estatus utilizan los bits de registro 7, 6, 5, 4, 3, correspondientes a los pines 11, 10, 12, 13, 15 del conector DB-25F.

   Algunos puertos paralelos tienen líneas de datos bidireccionales, que pueden ser utilizados como entradas o salidas, estos reciben el nombre de "líneas de control".

3. **Líneas de Control:** Además de las líneas de datos y las líneas de estatus, el puerto paralelo contiene un puerto de control bidireccional. Se pueden utilizar 4 líneas como entradas o salidas, en cualquier orden de combinación. Si por ejemplo la dirección del registro de control es la dirección base + 2, y si nuestro puerto activo es LPT1 (3BCh) entonces la dirección base +2 es 3BEh.

   Las 4 líneas de control: $\overline{\text{SELECTIN}}$, $\overline{\text{INIT}}$, $\overline{\text{AUTOLF}}$, $\overline{\text{STROBE}}$, utilizan los bits de registro 3, 2, 1, 0 y que corresponden a los pines 17, 16, 14, 1 respectivamente del conector DB-25F.

El bit 4 del registro de control permite las interrupciones de hardware del puerto paralelo. Cuando este bit 4 del registro se encuentra en un nivel lógico alto, entonces un flanco de bajada a la entrada del registro de estatus en ACK (registro de bit 6, pin 10) genera la interrupción. En otros tipos de puertos activan la interrupción con flancos de subida de la señal ACK.

Para utilizar la interrupción proporcionada por el bit 4 del registro de control, se debe instalar una rutina de interrupciones que se encuentren asignadas al puerto paralelo. Si es el caso que no se esté utilizando la interrupción, entonces no hay que colocar en alto el bit 4 del registro de control.

Los bits 7, 6, 5 del registro de control no son utilizados en la mayoría de los puertos paralelos. Los puertos que posean líneas de datos bidireccionales, los bits 7 o 5 permiten la configuración del puerto como entrada o salida.

CAPITULO IV

INTERFASE DE CONTROL POR SOFTWARE

4.1 REGISTROS DE SEGMENTO.-

Para poder reconocer cual es el *puerto paralelo activo* por estatus de un computador con el objetivo de utilizarlo como interfase de control; es necesario previamente conocer como se interpretan los datos direccionados que se encuentran en la memoria de acceso aleatorio (RAM), esto es a partir de su posición de memoria.

No es perjudicial ni crítico si el bus de datos es de 8, 16, 32 o 64 bits como es el caso de la arquitectura PCI, todos ellos están en la capacidad de ejecutar las mismas intrucciones para la transferencia de los datos; a través de sus puertos de entrada - salida y direccionar un espacio mínimo de memoria de 1 Mbyte.

El bus de datos más pequeño que puede disponer un microprocesador es de 16 bits, de modo que sólo se podría direccionar 2^{16} posiciones diferentes de la memoria principal esto es (64 KB):

2^{16} = 65536 bytes ó 64 Kbytes **1Kbyte = 1024 bytes**

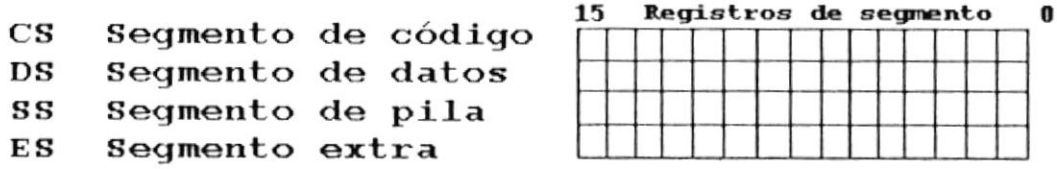


Figura 4.1 Estructura de los Registros de Segmento

Es posible expandir el rango de direcciones a 2^{20} posiciones esto es 1MB.

Esta capacidad de memoria se lo puede dividir en 16 segmentos de 64 KB cada uno representados por un dato de 16 bits.

La representación de una dirección de memoria tiene la siguiente forma:

Dirección Segmentada = Segmento: Offset

4.1.1 CONTENIDOS DE LA MEMORIA RAM.-

En la moyoría de los computadores existe un comando externo del MS-DOS, que nos permite realizar aplicaciones simples de visualización de las direcciones de memoria, a éste comando se lo conoce con el nombre de DEBUG.

Para utilizar el comando DEBUG, el usuario debe primero, direccionar la ruta raíz del path; esto significa encontrarse en la raíz del directorio: **C:/>_** luego se debe escribir la palabra *debug* desde el teclado y dar un ENTER.

Una vez realizado este proceso se visualizará en la pantalla del monitor un guión de línea " -" que es el cursor del "prompt" del editor DEBUG.

Este indicador de línea nos da la pauta de la existencia del prompt del editor DEBUG, nos indica por lo tanto que el sistema se encuentra listo para recibir los datos de direccionamiento desde el teclado, los mismos que se encuentran en el BIOS de la memoria RAM.

Para conocer cuales son los comandos disponibles del editor DEBUG, digitamos después del prompt de línea; la tecla de interrogación "?" y realizamos un ENTER.

El resultado de los pasos anteriores es la pantalla del editor DEBUG del MS-DOS, sus contenidos se muestran en la figura 4.2.

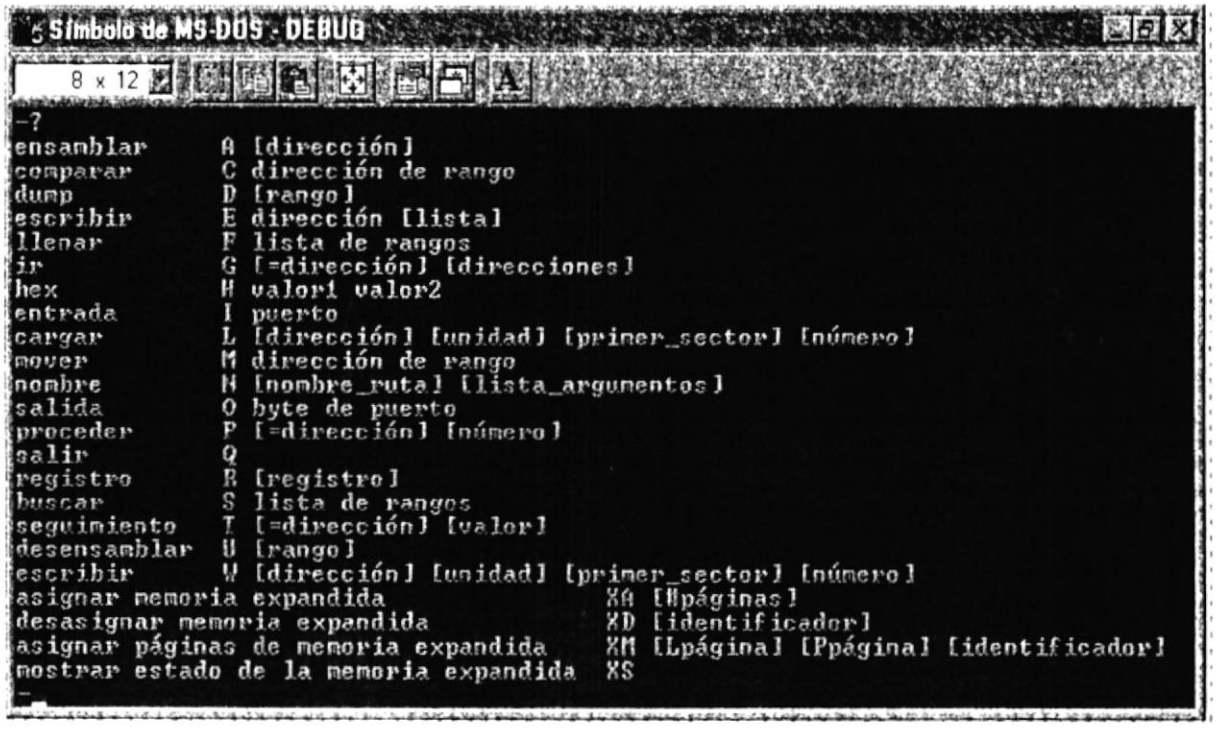


Figura 4.2 Comandos del Editor DEBUG Sistema Operativo MS-DOS

De esta lista detallada de comandos del editor; nos interesa el indicador "DUMP"; cuya función es mostrar los códigos existentes en la memoria RAM.

Es de esta manera como realmente se puede empezar identificando los ***"puertos"*** disponibles de un computador.

La presencia de los "*puertos*" comienza a partir de la posición de memoria 0040h:0000h.

Esta dirección segmentada es el resultado de las componentes "segmento" y "offset", ambas se encuentran representadas en notación hexadecimal.

Si deseamos conocer los contenidos de la memoria RAM a partir de esta posición; entonces debemos escribir desde el teclado:

C:\>debug [ENTER] y luego escribir -d40:0 [ENTER]

Este proceso se resume en la pantalla del MS-DOS, donde se obtiene una estructura similar a la que se muestra en la figura 4.3.

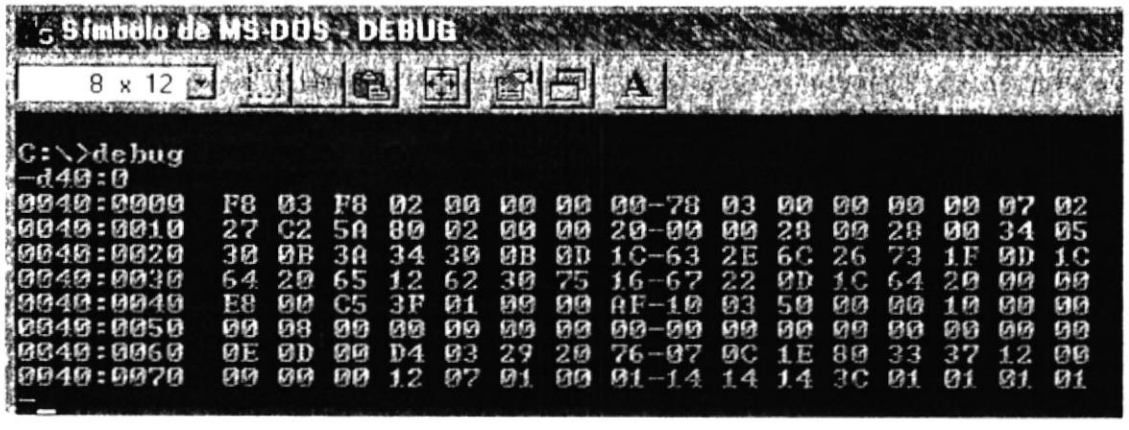
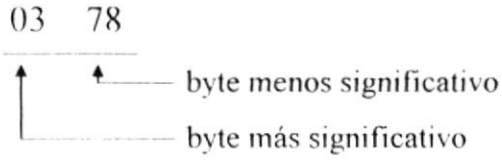


Figura 4.3 Contenido de la Memoria RAM

Como se puede observar en la figura 4.3 a partir de la posición de memoria 0040:0000 se encuentran los datos que conforman las direcciones de los dos puertos seriales [03F8 - 02F8] y del puerto paralelo [0378].

La disposición de la estructura de estos datos se encuentra conformada por pares. En la figura 4.3 el par de datos que define la dirección del puerto paralelo es [78 03] y su distribución en bytes se encuentra definida como: el primer byte es el menos significativo y el segundo byte es el más significativo.

De modo que para leer la dirección del puerto paralelo disponible, debemos escribirla realmente como:



Cada uno de los pares de datos se encuentra representados por sus etiquetas de identificación, definidas de izquierda a derecha como: COM1, COM2, COM3, COM4, LPT1, LPT2, LPT3, LPT4. Ver figura 4.4.

Esta dirección corresponde como sabemos al puerto paralelo LPT1 (Line Printer 1), las otras dos direcciones que se encuentran en la misma línea de la posición de memoria 0040:0000 corresponden a los dos puertos seriales que disponen generalmente todos los computadores personales.

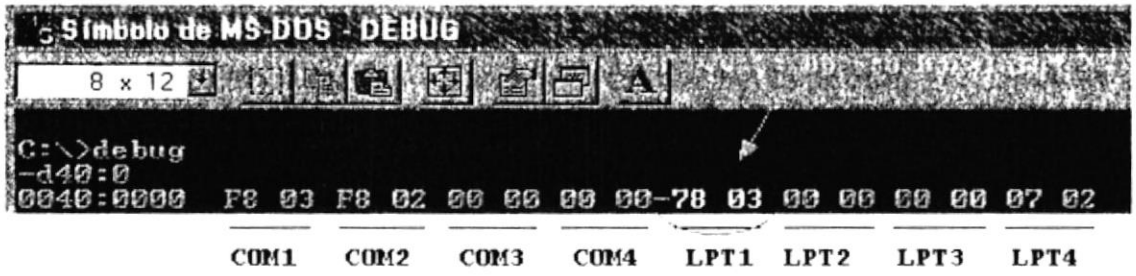


Figura 4.4 Dirección del Puerto Paralelo Activo en la Memoria RAM

Es de esta manera "técnica" y no empírica, como verificamos que el puerto que vamos a utilizar para controlar un periférico se trata efectivamente del puerto LPT1.

De este modo estamos preguntando directamente a la memoria RAM cual es la dirección del puerto paralelo activo por setup.

4.1.2 SIMULACION DE CONTROL SOBRE UN PERIFERICO.-

Ahora que conocemos como usar el editor de programación (DEBUG), podemos realizar una previa *simulación de control de periféricos* utilizando el puerto paralelo.

La instrucción que se debe escribir en la línea del prompt para obtener una respuesta de salida en el puerto I/O; se la obtiene de uno de los comandos del editor DEBUG, sea el sistema operativo MS-DOS o DOS 7.0 (Ver figura 4.2), este comando se lo conoce como: salida O byte de puerto.

Directamente del editor DEBUG se puede escribir una línea de instrucción para que sea interpretado por el sistema mismo del PC y permita tener el control de los 8 bits de datos del puerto paralelo DB-25F. Previamente antes de realizar la simulación de control del puerto paralelo, es necesario disponer de una *interfase digital de prueba*, esto es para poder visualizar claramente como se dispone el control de los datos en el puerto.
A continuación se muestra el esquema de la interfase digital de prueba:

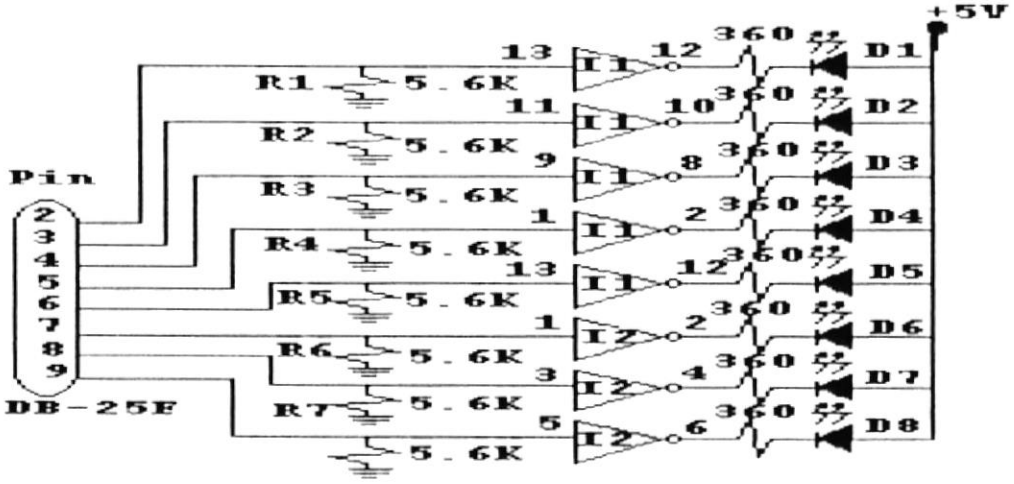


Figura 4.5 Interfase Digital TTL de Prueba para Control de Periféricos

4.1.3 FUNCIONAMIENTO DE LA INTERFASE DIGITAL.-

La interfase digital se encuentra conformada por dos circuitos integrados IC Schmitt Trigger 74LS14N, su funcionalidad es de operar como un inversor de lógica.

Se disponen de 8 inversores que son conectados a la salida del bus de datos del puerto paralelo. En el capítulo III Tabla 3.1 se describe claramente cuales son los pines correspondientes a los 8 bits de datos del conector DB-25M.

De esta tabla concluímos que los pines son: 2,3,4,5,6,7,8,9 y corresponden a los bits 0,1,2,3,4,5,6,7 respectivamente. El orden de prioridad significativo de los datos es de LSB a MSB (de menor valor a mayor valor significativo).

Al encender el computador; éste realiza el proceso de arranque o boteo, los niveles lógicos de cada uno de los bits de datos a la salida del puerto DB-25F; se pueden encontrar en un nivel lógico alto o bajo, de ello que es muy importante conocer el "estado" de estos bits; y esto lo determinamos una vez que concluya todo el proceso de arranque controlado por el sistema operativo que disponga el computador.

La disposición del estado de los bits de salida del puerto DB-25F corresponde únicamente al BIOS de cada computador, es por ello que esta *interfase digital* simuladora nos ayuda a "identificar", cual es el estado lógico final del puerto I/O después de que se realiza el arranque de un PC.

Para direccionar un bit a la salida del I/O puerto DB-25M, se escribe desde el editor DEBUG la sentencia:

_O[Dirección del puerto, Valor del bit]

Esta sentencia la interpretamos como: Direccionar un bit (cualquiera de los 8 bits de datos) a la salida del puerto paralelo. Por ejemplo colocar un nivel lógico "1" en el pin 2 (menos significativo, LSB) del puerto paralelo.

Esto lo realizamos escribiendo la sentencia en la línea del prompt del editor DEBUG:

_O378,1 ó _o378,1

Al ejecutar esta instrucción se puede observar que el LED1 que se encuentra conectado al pin 2 del puerto I/O se enciende, por lo tanto existe un nivel lógico "1" a la salida del bit del puerto.

Si deseamos apagar el LED1 debemos escribir nuevamente en la línea del prompt del editor la sentencia:

_O378,0 ó _o378,0

En el caso que se desee obtener un nivel lógico "1" en cualquiera de los demás bits se deberá escribir las sentencias :

_O378,1 ó _o378,1 → enciende el LED1, pin 2, bit LSB
_O378,2 ó _o378,2 → enciende el LED2, pin 3
_O378,4 ó _o378,4 → enciende el LED3, pin 4
_O378,8 ó _o378,8 → enciende el LED4, pin 5
_O378,10 ó _o378,10 → enciende el LED5, pin 6
_O378,20 ó _o378,20 → enciende el LED6, pin 7
_O378,40 ó _o378,40 → enciende el LED7, pin 8
_O378,80 ó _o378,80 → enciende el LED8, pin 9, bit MSB

Debemos destacar que el control de cada bit es único, esto significa que se puede "*encender*" un único bit de prueba a la vez, al ejecutar cualquiera de estas sentencias desde el editor DEBUG en un sistema operativo MS-DOS.

La acción de apagado de los LED's se la ejecuta igualmente como las sentencias anteriores excepto que el ***Valor del bit*** debe ser cero.

La razón de escribir estos valores de bit's como: 1, 2, 4, 8, 10, 20, 40, 80 tiene su lógica. Estos valores se encuentran en notación hexadecimal y su codificación binaria es lo que interesa para colocar un único bit en alto a la salida del puerto I/O.

La codificación binaria de estos valores hexadecimales las podemos escribir como:

Valor del bit hexadecimal		**bit activo en el I/O puerto**
1	→	**0 0 0 0 0 0 0 1**
2	→	**0 0 0 0 0 0 1 0**
4	→	**0 0 0 0 0 1 0 0**
8	→	**0 0 0 0 1 0 0 0**
10	→	**0 0 0 1 0 0 0 0**
20	→	**0 0 1 0 0 0 0 0**
40	→	**0 1 0 0 0 0 0 0**
80	→	**1 0 0 0 0 0 0 0**

De esta manera podemos verificar manualmente, como disponer de un bit a la salida del puerto I/O del PC.

Ahora estamos en capacidad de probar al sistema parcialmente estructurado esto es: la interfase electrónica de aislamiento, la interfase de potencia, el computador personal como dispositivo de control y la carga AC.

Mencionamos que el sistema se encuentra parcialmente estructurado porque aún falta de diseñar el "software" como *interfase de control automático* para que interactúe con el CPU.

4.2 PROGRAMACION EN LENGUAJE C.-

Hemos escogido el lenguaje C como medio de programación por ser un lenguaje altamente funcional, práctico, operativo a nivel de puertos, eficiente, y de bajo nivel, y entre otras muchas características por ser una herramienta fundamental para la elaboración de software de tipo profesional.

Características esenciales del lenguaje C:

1. **Eficiencia del lenguaje:** El lenguaje C proporciona de una gran ventaja en lo relacionado a eficiencia, su objetividad facilita la portabilidad con muchos campos de la aplicación, por la popularidad que ha ganado y por la amplia bibliografía disponible es por ello que ha sido escogido en muchos lugares del mundo como el lenguaje de programación ideal.

2. **Funcionalidad:** Lo más importante del lenguaje C es la funcionalidad y la estructuración. La funcionalidad implica que el programa puede constar de bloques de funciones que son invocados desde el programa principal o desde otras funciones que no se encuentren dentro del programa. Así también la funcionalidad del C

permite la elaboración de librerías de funciones, las mismas que pueden ser compiladas, e incorporadas en cualquier programa.

3. **Estructuración:** La estructuración significa que permite la conformación de las estructuras clásicas en los diagramas de control de flujo del programa; estas estructuras clásicas están conformadas por las conocidas sentencias: *for, while, do, if, switch*. Estas son las características que facilitan los lazos de control sobre instrucciones de código internos mediante los conocidos bloques condicionales, las instrucciones de control son reconocidas por lo general en cualquiera de las diferentes versiones de software en lenguaje C diseñadas.

4. **Acceso:** Otro detalle del lenguaje C es que puede acceder a la memoria del sistema y realizar modificaciones de bit's individualmente con operaciones booleanas o rotaciones sobre bytes.

5. **Portabilidad:** La portabilidad significa que el software escrito para un determinado tipo de computador, puede adaptarse a otro tipo. Por ejemplo, si un programa se ha escrito para un Apple II+ y puede llevarse fácilmente a un IBM PC, ese programa es portable [SCH190].

4.2.1 ELABORACIÓN DE UN PROGRAMA EJECUTABLE.-

Es conocido que todo programa ejecutable debe tener por lo menos una de las tres extensiones siguientes: .EXE, .COM, .BAT. El lenguaje C puede crear un archivo ejecutable con extensión .EXE.

Para poder crear un ejecutable se debe seguir los siguientes pasos:

1. **Escribir el programa en el editor de texto:** Todo software de programación está acompañado de un editor de texto ASCII para la elaboración de un programa. El editor de texto es el lugar donde el programador escribe las instrucciones y sentencias que conforman el cuerpo del programa.

   Una vez que se tiene el programa estructurado se procede a grabar el archivo en disco y el software automáticamente identifica el código fuente como una extensión ".C".

2. **Compilación del código fuente:** Ahora que ya tenemos el archivo fuente es necesario realizar la compilación, esto significa que el programa sea elaborado en código de máquina. El proceso de compilación crea un archivo objeto con el mismo nombre que el código fuente, excepto que la extensión termina en ".OBJ".

   Los compiladores únicamente son programas que operan sobre el código fuente de su programa. Un compilador lee el programa entero y lo convierte en código objeto, que es la traducción del código fuente del programa, a una forma que es directamente ejecutable por el computador.

3. **Archivo de enlace:** El archivo de enlace es el archivo objeto ".OBJ" que está encargado de convertir las referencias cruzadas a subrutinas en código ejecutable, en el proceso de compilación se crea automáticamente un archivo ejecutable con extensión .EXE.

4.2.2 ESTRUCTURA DE UN PROGRAMA EN C.-

Todos los programas en C tienen definido un orden de estructura que clasificamos en los siguientes grupos:

1. Directivas del preprocesador.
2. Declaración de funciones definidas por el usuario.
3. Declaración de variables globales.
4. Definición de la función principal del programa *main*.
5. Definición de las funciones propias del usuario.

1. **Directivas del preprocesador:** A través de estas directivas se establece los procedimientos en tiempo de compilación y no en tiempos de ejecución.
   La identificación del uso de una directiva del preprocesador se la reconoce cuando se escribe el carácter #.

   La directiva más conocida por su uso es por ejemplo #include <stdio.h>, que le indica al compilador que funciones habilite para su uso, de las muchas que tiene el lenguaje C. A los archivos terminados en extensión ".h" se los conoce como *archivos de cabecera* y cada uno de ellos tiene su propósito específico dentro de un programa. Como por ejemplo el archivo *stdio.h* es un archivo estándar de cabecera de entrada salida.

   Si este archivo va incluido en dobles comillas significa que el fichero especificado, debe ser buscado en el directorio actual de trabajo y si no se encuentra, la búsqueda debe continuar en el directorio estándar para los ficheros con extensión .h (directorio include) [CEVA91].

   Si el fichero especificado, en lugar de escribirlo entre comillas, lo escribimos entre ángulos, la búsqueda de dicho fichero se efectúa solamente en el directorio estándar para los ficheros con extensión .h (directorio include) [CEVA91].

Entre los archivos de cabecera más conocidos podemos mencionar los siguientes:

assert.h	dos.h	iomanip.h	setjmp.h	stdlib.h
bcd.h	errno.h	iostream.h	signal.h	stream.h
bios.h	fcntl.h	limits.h	stat.h	string.h
complex.h	float.h	locale.h	stdarg.h	time.h
conio.h	fstream.h	math.h	stddef.h	timeb.h
ctype.h	generic.h	mem.h	stdio.h	types.h
dir.h	graphics.h	process.h	stdiostr.h	

Tabla 4.1 Archivos de Cabecera del Lenguaje C

2. **Declaración de funciones definidas por el usuario:** Las declaraciones de funciones en C tiene por finalidad indicar que existe una función con cierto nombre y de un tipo especifico.

   Para declarar una función propia se debe establecer cual es el tipo de dato que la función retorna, así como también el nombre de la función y los tipos de datos que se toman como argumentos. También se puede hacer un procedimiento, esto significa que la función no retorna valor alguno.

   La declaración de una función o función prototipo, o declaración forward que es lo mismo; va ubicada después de los archivos cabecera del programa.

3. **Declaración de variables globales:** Las variables globales mantienen sus valores en todo el programa, y mientras dura su ejecución. Se crean declarándolas fuera de toda función. Pueden ser accedidas por cualquier expresión, independientemente de en qué función se encuentre [SCHI91].

4. **Definición de la función main:** Todo programa tiene una función llamada *main* donde se inicia la ejecución; los paréntesis que van después de *main* indican al compilador que se trata de una función. Las llaves encierran al cuerpo de la función; también se usan para agrupar varias proposiciones [KELL89].

   La función main es la función principal, o llamada también rutina principal, y es la que el sistema operativo invoca cuando se inicia la ejecución de un programa.

5. **Definición de las funciones propias del usuario:** Las funciones son bloques con los que se constituyen programas en C, y en ellos se lleva a cabo toda actividad del programa. Una vez que una función ha sido escrita y depurada, puede utilizarse una y otra vez. Es este uno de los aspectos más importantes del C como lenguaje de programación [SCHI 91].

   Otro punto importante a tener presente en la programación en C son las declaraciones de las variables las cuales nos referimos a continuación:

- **Variables:** Para que un computador pueda procesar los datos de entrada y de salida, se debe reservar un espacio de memoria para almacenar estos datos. Estos espacios de memoria lo ocupan las variables y se deben reservar (declarar) antes de utilizarse; al ser declarados se les asigna un nombre. De acuerdo al tipo de información de la variable que se requiera se debe especificar el tipo de dato y el nombre con el que se va a identificar cuando se la utilice en cualquier parte del programa.

- **Datos enteros:** Los números enteros se los representan en cantidades exactas, y para el lenguaje C un dato entero está comprendido en el rango: -32768 a 32767. Significa que su representación numérica exige de dos bytes o 16 bits.

- **Datos flotantes:** Los datos flotantes son las representaciones numéricas con decimales, en lenguaje C un dato flotante está comprendido en el rango: +/- 3.4 $x10^{38}$ a +/- 3.4 $x10^{-38}$. Significa que su representación numérica exige de cuatro bytes o 32 bits.

- **Caracteres:** Las variables que son definidas como carácter están representadas por valores enteros en el rango: 0 a 255. Significa que su representación numérica exige de un byte o 8 bits.

4.2.3 DIRECCION SEGMENTO OFFSET EN MEMORIA.-

Estamos familiarizados con el manejo de puertos a través del editor DEBUG, ahora nuestra tarea no es tan difícil para interpretarlo en lenguaje de programación C.

Igualmente el lenguaje C hace referencia a segmentos y offset para conseguir una dirección de memoria. Los segmentos definen bloques de memoria de 64 Kbytes y el offset define los desplazamientos y apunta a una dirección particular dentro de ese bloque. La dirección completa se encuentra en notación hexadecimal y es el resultado de la suma del segmento por una base decimal de diez más el offset. Esta dirección es en realidad la posición de memoria. Existe diversas formas de escribir un segmento y un offset tal que su suma represente una misma posición de memoria.

Por ejemplo hemos definido que una posición de memoria la podemos escribir como:

$$\textit{Dirección de Memoria} = \textit{Segmento} * 10 + \textit{Offset}$$

Si tenemos una dirección de memoria 400h, ésta la podemos referenciar escribiéndola como:

1) 40h * 10h + 00h =
400h + 00h =
400h

2) 04h * 10h + 3C0h =
040h + 3C0h =
400h

3) 00h * 10 h + 400h =
00h + 400h =
400h

El resultado final será escribir la posición de memoria en la notación segmento : offset.

1) [0040:0000] **2)** [0004:03C0] **3)** [0000:0400]

Todas estas posiciones direccionan a un mismo punto de memoria. De la referencia anterior se explica ahora el porque, poder direccionar a una posición de memoria de diferentes maneras; no existe un modo único.

En el lenguaje C estas posiciones de memoria se encuentran en notación hexadecimal y se las escribe como *argumentos* en la función de salida del puerto I/O separándolas por una coma.

Para hacer un requerimiento de una dirección en una posición de memoria definida, el lenguaje C permite el recurso de la función *peekb( )*: retorna el byte en la localización de memoria especificada por el segmento : offset.

La función *peekb( )* hace referencia al archivo de cabecera *dos.h*. C permite conocer la dirección del primer puerto paralelo disponible, obteniendo el valor de cada uno de los bytes y luego reconstruyendo la dirección del puerto; esto lo logramos multiplicando el contenido de la posición de memoria siguiente por la base decimal 256 y luego sumando el contenido de la dirección inicial.

La dirección del primer puerto paralelo la podemos conocer escribiendo la sentencia:

Puerto = peekb(0x0040,0x0009) * 256 + peekb(0x0040, 0x0008);

En C para representar un valor en notación hexadecimal es necesario primero escribir el código *0x* o *0X* y luego el segmento y offset que se desea direccionar.

C dispone de dos maneras para realizar un acceso de lectura a memoria:

- *peekb ( )* : Esta función hace un requerimiento de lectura a memoria de un byte.

Byte = peekb (Segmento, Offset);

- *peek ( )* : Esta función hace un requerimiento de lectura a memoria de una palabra.

  Palabra = peek (Segmento, Offset);

Conocido el acceso a las posiciones de memoria, podemos hablar de los puertos. Estos son indispensables ya que a través de ellos es como se ingresa la información al computador (por medio del uso de un módem, un scanner, un convertidor analógico a digital, etc) y a través de los mismos puertos es como sale la información (disco, impresora, modem, convertidor digital a analógico, etc).

Ahora que ya estamos familiarizados con el uso del puerto centronics DB-25F, y conocedores de sus estrategias y disposición del uso de sus entradas y salidas, pines de operación y bits, modo de simulación de control de salidas por medio del editor DEBUG, nuestra tarea se centra en escribir los códigos del programa que gobierne al sistema de control automático de cargas.

4.2.4 SENTENCIAS DE CONTROL DE PUERTO EN LENGUAJE C.-

Vamos a escribir las sentencias de un programa en lenguaje C, que consulta la dirección del primer puerto paralelo disponible en un PC, una vez conocido este puerto se direccionan los 8 bits de datos.

Cada bits de los datos son fijados a un nivel lógico alto "1" (lógica positiva) en un conteo ascendente en disposición como un contador binario de 00000000 a 11111111.

Estas líneas de programa proporciona una idea más clara de las ventajas del uso de los puertos. Para poder observar el correcto funcionamiento y desempeño a la salida del puerto centronics DB-25F es necesario disponer de la interfase digital TTL de prueba para control de periféricos (figura. 4.5).

Se debe tener muy presente realizar en forma correcta la interconexión de los pines de la interfase digital con los terminales y pines del puerto centronics DB-25F. Una vez realizado éste proceso; se procede a llamar al programa ejecutable PSV.EXE. Al inicializarse el programa inmediatamente se ejecuta el conteo por parte del contador binario interno codificado por los códigos de programación del software escrito aquí. El reloj del contador binario en el programa depende de la variable "Contador"

Programa PSV.C de prueba de control de puertos I/O.

```
/*****************************************************************/
/************ PROGRAMA DE PRUEBA DE CONTROL ***************/
/************ DEL PUERTO CENTRONICS DB-25 F  ***************/
/*****************************************************************/

/* El programa imprime en la pantalla la dirección hexadecimal del I/O  puerto de un
   computador personal, activa un nivel lógico 1 en el puerto centronics DB -25F, bit
   menos significativo (LSB) pin 2, bit dato 0. Además el programa cuenta en binario del
   00000000 a 11111111. Para poder apreciar su resultado es necesario disponer de la
   Interfase Digital de Prueba (Figura 4.5) para el Control de Periféricos a la salida del
   I/O puerto, utilizar un conector DB-25 M    PSV.C */


#include <stdio.h>   /* librería de cabecera estándar de entrada-salida */
#include <stdlib.h>
#include <dos.h>     /* librería de cabecera del DOS (Disk Operating System
                     *  la función peekb( ), outportb( ), delay ( ) */
#include <conio.h>   /* presencia de la librería conio.h  para  la habilitación
                     * de la funciones clrscr( ) , y getch ( ) */

void main (void)     /* inicio del programa principal */
                     /* la función principal no acepta argumento alguno ni
                      * retorna ningún valor */
{
```

```
int Valor_Puerto ;      /* declaración de la variable local a la función principal
                        * main */
int Contador = 0;       /* declaración de la variable local Contador e
                        *  inicialización */
system("cls");          /* borrado de la pantalla con comando DOS */
Valor_Puerto      peekb(0x0040,0x0009) * 256  + peekb(0x0040, 0x0008);

                        /* requerimiento de lectura a memoria de un byte */
outportb(Valor_Puerto,0x1);

do{
clrscr( );              /* borrado de la pantalla */
textcolor(10);          /* línea de texto en color ROJO */

                    /* convierte el valor del puerto en dato hexadecimal */
cprintf("Salida del puerto paralelo ACTIVADA, Puerto LPT1:%xH\n", Valor_Puerto);
gotoxy(1,3);            /* ubica el cursor en la posición x =1, y = 3 */
textcolor(9);           /* línea de texto en color AZUL */

cprintf("\nPulse 0 para DESACTIVAR el bit LSB pin 2 del puerto paralelo: ");
/* imprime el valor hexadecimal del puerto */
} while((getche( )! = '0'));

outportb(Valor_Puerto,0x0);  /* DESACTIVA el puerto paralelo y lo pone en
                                nivel bajo */
printf("\n");           /* retorno a nueva línea */
textcolor(14);          /* texto de color amarillo */
```

```
printf("\n");              /* retorno a nueva línea */
cprintf("GRACIAS");
gotoxy(1,6);               /* ubica el cursor en la posición x =1, y = 6 */
delay(800);                /* suspende la ejecución del programa en un intervalo de
                              tiempo de 800 milisegundos */
clreol( );                 /* borra el fin de línea en una ventana de texto */
gotoxy(1,20);              /* ubica el cursor en la posición x =1, y = 20 */
textcolor(12);             /* texto de color verde */
sound(500);                /* enciende el parlante de un PC en la frecuencia
                              especificada*/
delay(500);                /* suspende la ejecución del programa en un intervalo de
                              tiempo de 500 milisegundos */
sound(440);                /* activa el parlante de un PC en la frecuencia
                              especificada 440 Hz */
delay(500);                /* suspende la ejecución del programa en un intervalo de
                              tiempo de 500 milisegundos */
nosound( );                /* apaga el parlante del PC */

                           /* inicio del lazo de cuenta binaria 0 a 255 */
for (Contador = 0; Contador< =255 ;Contador++)
{
outportb( Valor_Puerto,Contador); /* obtener un alto en los bits según el valor
                                     binario que se requiere a la salida del puerto */
delay(400);                /* suspende la ejecución del programa en un intervalo de
                              tiempo de 400 milisegundos */
}
cprintf("Pulse cualquier tecla para continuar");
```

```
_setcursortype(_NOCURSOR);
getch( );              /* lectura de un carácter del teclado, sin visualizarlo */
return ;               /* retorno de la función */
}                      /* fin del programa */
```

4.2.5 INTERFASE GRAFICA DE CONTROL POR SOFTWARE.-

Para poder controlar las cuatro cargas monofásicas es necesario disponer de un sistema inteligente de control que permita la conmutación de las cargas una vez que se fija una determinada hora de operación a cada una de ellas. A este sistema inteligente lo hemos llamado interfase gráfica de control por software.

La interfase de control por software es el resultado de una programación en lenguaje C para un manejo adecuado del puerto. En base a lo que se ha visto en este capítulo es fácil ahora realizar la esquematización de la interfase de control. No es mi objetivo aquí de enseñar a programar sino resaltar las ventajas que se tiene al interconectar las interfases electrónica - digital con la interfase desarrollada por software.

La pantalla que describe los componentes de la interfase por software se muestra en la figura 4.6, el diseño de la presente interfase por software está conceptualizado para operar en lógica negativa. La *interfase gráfica de control por software* en combinación con la *interfase de control optoaisladora* (figura 2.2) proporciona el sistema completo de un ***controlador automático de cargas monofásicas***.

Si se deseara controlar cargas de mayor potencia entonces es factible utilizar la interfase SSR DC/AC MOC3031 optoaislador con detector de cruce por cero e interfase TRIAC (figura 2.7), realizar unas modificaciones en el software de control, recordemos que el software de control está diseñado para operar en lógica negativa.

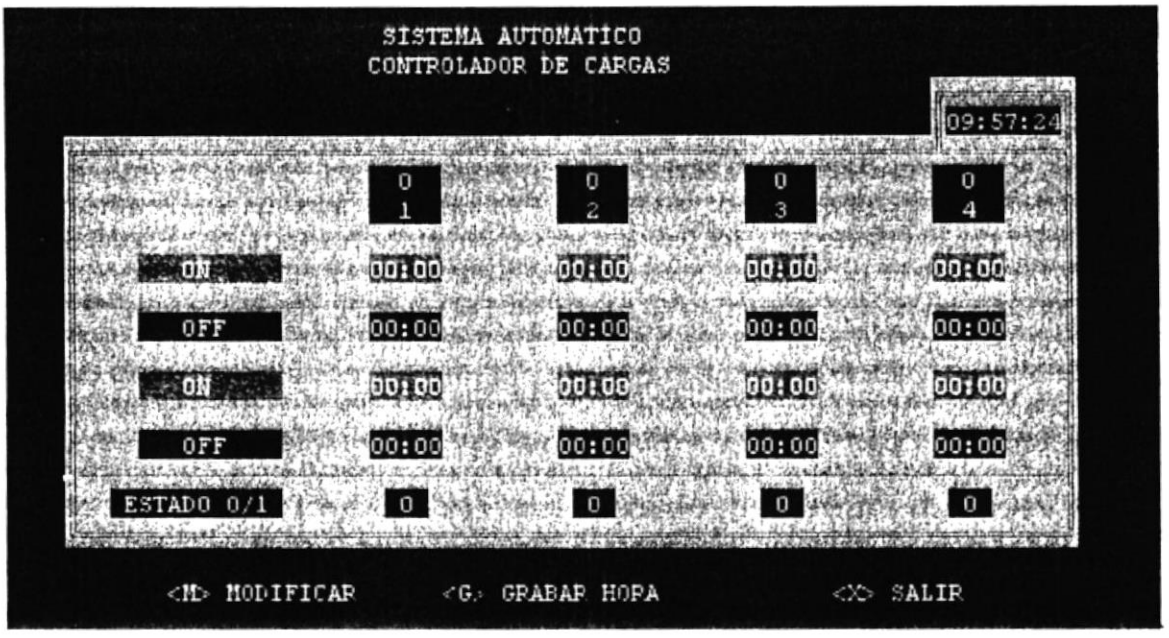


Figura 4.6 Interfase Gráfica de Control por Software Inicialización

En la pantalla de la figura 4.6 se observa los componentes de la interfase por software, en el marco superior se puede visualizar la disposición de control sobre cuatro cargas monofásicas cada una representadas como salidas por las columnas O1, O2, O3, O4.

Inicialmente al correr por primera vez el programable ejecutable (SAC.EXE) todos los tiempos que definen al control de carga se encuentran encerados es decir a cero horas, cero minutos, cero segundos.

Del lado izquierdo podemos observar una columna de energización (ON) y de desenergización (OFF) de carga, estos marcos de referencia nos indican cuando se activa una carga y cuando se desactiva de la fuente de alimentación alterna.

La pantalla por software muestra un reloj que fija el tiempo de inicialización del computador; de esta manera compara: el tiempo interno de cada controlador de carga, con el reloj del PC, de este modo se obtiene un control sobre una carga específica.

El control sobre cualquier carga monofásica se lo puede obtener en un estatus de control de tiempo de 24 horas continuas para cada carga disponible por el controlador por software.

El estado 0/1 de energización de cada una de las cargas se la verifica por su representación en lógica positiva como: 1 significa energizada la carga y 0 desenergizada. Cada estado es independiente de los demás, por lo tanto representa una ayuda gráfica para el usuario saber que carga se encuentra conmutada a la salida.

En la figura 4.7 se muestra los tiempos programados para el control periódico de las cuatro cargas. La salida O4 visualiza un estado de activación en el puerto I/O; por lo tanto este bit de salida del puerto se encuentra en un nivel lógico 0, los demás bits se encuentran en un nivel lógico 1.

La representación gráfica de que ésta carga O4 que se encuentra habilitada se la especifica en el marco de ESTADO 0/1 como un 1.

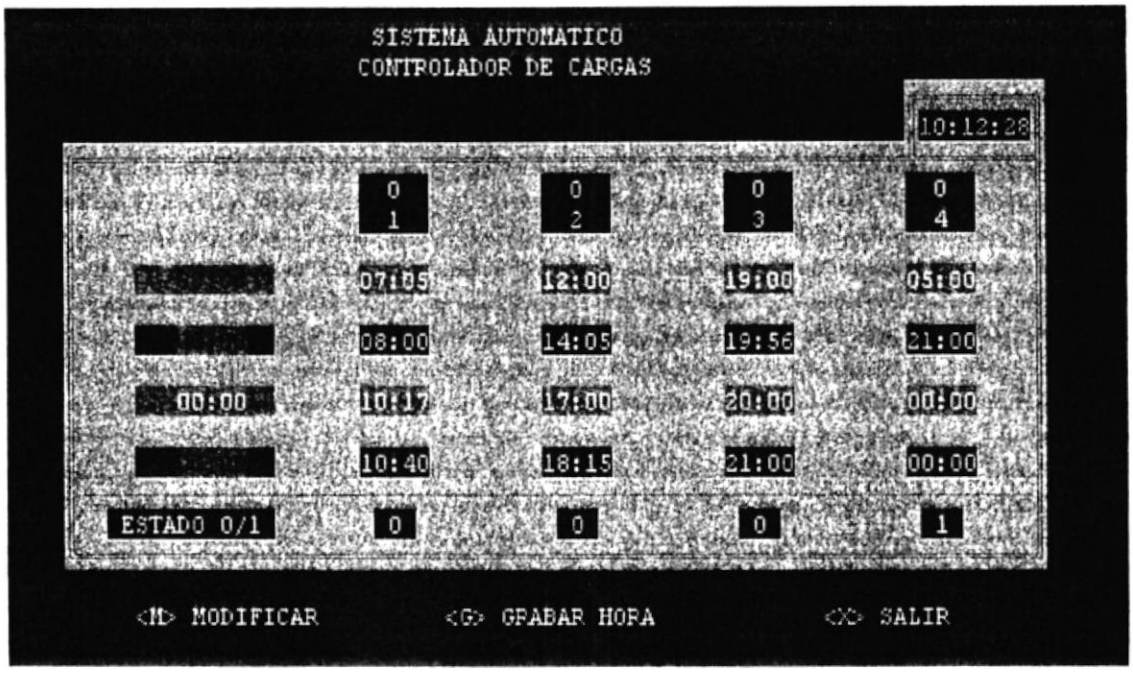


Figura 4.7 Pantalla en Ejecución Control de Carga O4

El acceso de control sobre los tiempos se lo puede disponer gracias a la barra de menú con la opción MODIFICAR que se visualiza en la pantalla de control.

Para fijar la nueva selección de las horas de habilitación deshabilitación se utiliza la opción GRABAR del menú.

Una vez fijadas las horas que gobiernan a las cargas se utiliza de la barra del menú la opción TERMINAR para retornar a la pantalla principal de control.

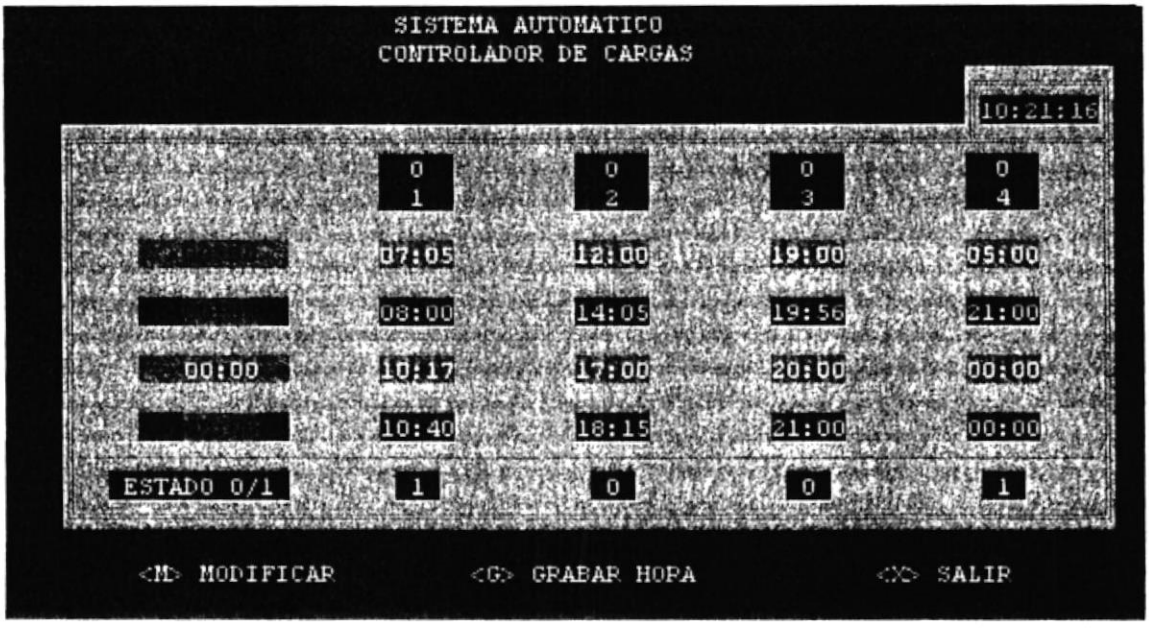


Figura 4.8 Pantalla de Ejecución y Control de Cargas O1 - O4

En la figura 4.8 se aprecia el control sobre dos cargas como son : O1 y O4, al interconectar la *Interfase Digital de Prueba* (Figura 4.5) *para el Control de Periféricos* a la salida del puerto I/O se puede observar que la salida O1 y O4 sus estados lógicos pasan a estar en un nivel bajo, sin embargo se encuentran conmutadas las dos cargas, destacamos que existe lógica negativa en los bits del puerto DB-25F para el control de los periféricos.

CAPITULO V

ADQUISICION DE DATOS

5.1 PRINCIPIOS BASICOS.-

Se entiende por adquisición de datos a la acción de medir variables, convertirlas a un formato digital, almacenarlas en la memoria del computador y por último procesarlas [SCHI91].

Hasta ahora sólo hemos hablado de un sistema de monitoreo de *lazo abierto* que lo hemos definido para un controlador automático de cargas. La expresión *lazo abierto* la denotamos como un sistema que solamente posee salidas y están presentes únicamente cuando el software de control ejecuta una acción de proceso preprogramada por instrucciones del usuario.

Los sistemas de *lazo cerrado* nos permiten la *adquisición de datos* proveniente de una variable física de cualquier medio. Una vez que ésta variable es sensada por algún dispositivo electrónico, entonces se requiere de un proceso de conversión lógica para que el PC pueda interpretar correctamente los datos a ingresar y así lograr una respuesta de salida a través del puerto I/O paralelo.

Casi el noventa porciento de los procesos industriales tienen que ver con la medición y control de la temperatura. Hoy en día un computador puede medir y controlar centenares de sensores de monitoreo. La respuesta de acción por parte del PC luego de obtener un dato puede ser dirigida al control del encendido y/o apagado de una máquina, o simplemente realizar un reajuste de datos en el software de control.

Las salidas y entradas del puerto I/O centronics DB-25F como sabemos son de tipo digital TTL, de manera que las entradas al puerto únicamente pueden ser interpretadas como niveles lógicos "1" o "0" (el puerto serial RS-232 es de tipo CMOS).

Generalmente los datos que se obtienen de las variables físicas son de tipo analógico, por lo tanto es necesario de la utilización de una interfase electrónica para la conversión analógica a digital.

Las técnicas de adquisición y conversión de datos son utilizadas en toda clase de aplicaciones en la que se requiere que una señal analógica deba convertirse a una señal digital esto es posible gracias a lo que conocemos como *modulación por codificación de pulsos* (PCM).

5.1.1 CONVERTIDOR ANALOGICO A DIGITAL.-

Uno de los métodos más sencillos de conversión A/D es el representado en la figura 5.1. Utiliza tres elementos principales: un contador, un convertidor D/A y un comparador analógico. Por simplicidad la mayoría de los circuitos lógicos de control se han omitido en el diagrama [SCH91].

El convertidor opera como sigue. En el comienzo de un ciclo el contador está puesto a cero (reset). Este produce una tensión de salida D/A $V_b = 0$ que es aplicada a una entrada del comparador.

La entrada analógica es alimentada a través de un circuito de muestreo y mantenimiento cuya salida V_a es aplicada a la otra entrada del comparador.

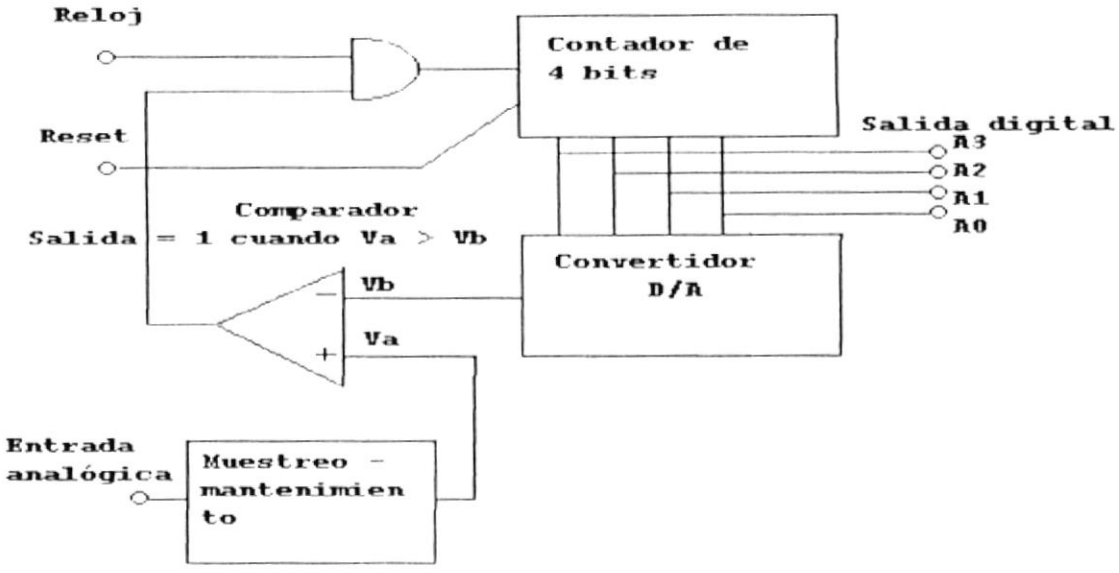


Figura 5.1 Convertidor A/D 4 Bits Controlado por Contador

En tanto que la señal analógica V_a sea mayor que V_b, la salida del comparador será 1 y la puerta AND estará habilitada, permitiendo que entren en el contador los impulsos de reloj. El contador contará entonces hacia arriba o en sentido ascendente partiendo de cero. Con cada cuenta la salida D/A V_b aumentará un paso o escalón, como muestra la figura 5.2 . Esta cuenta continuará hasta que la forma de onda en escalera exceda del valor de la señal analógica V_a. En este instante se anulará la salida del comparador, inhibiendo la puerta AND y por consiguiente parando al contador. La salida es entonces leída en los terminales de salida del contador. En la figura 5.2 la salida será 0101 correspodiente a 5 V. Este tipo de contador es relativamente lento, ya que pueden ser necesarios para la conversión muchos periodos de reloj, tantos como 2^N -1 (15 para un convertidor de 4 bits). Se puede reducir el tiempo de conversión si se utiliza un contador reversible y se modifica el convertidor para que cuente en sentido ascendente cuando

$V_b < V_a$ y cuente en sentido descendente cuando $V_b > V_a$, en este caso el comparador activa el control de modo del contador [SCHI91].

En la realización del circuito real de este convertidor se necesita una lógica adicional para controlar el tiempo de mantenimiento del circuito de muestreo y mantenimiento y proveer la sincronización de las señales de reloj, reset, y mantenimiento [SCHI91].

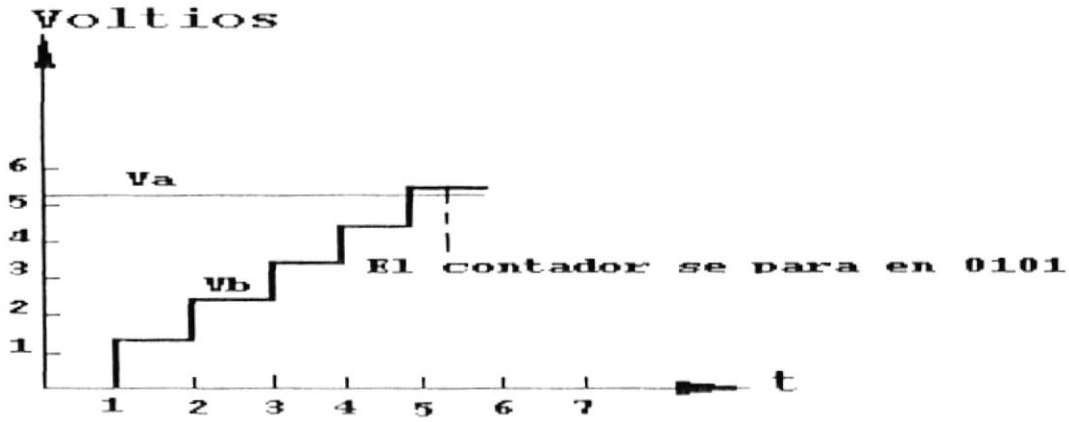


Figura 5.2 Formas de Onda en el Convertidor A/D 4 Bits

5.1.2 SISTEMA DE ADQUISICION Y CONVERSION DE DATOS.-

Básicamente un sistema de adquisición de datos se encuentra conformado: de un *multiplexor analógico* de entrada, un *amplificador de muestreo y retención*, un *convertidor analógico - digital* y un *circuito de control*. El circuito de control tiene la propiedad de suministrar las señales lógicas de sincronización y secuencia requeridas en cada instante, además proporciona una interfase con el dispositivo de procesamiento. Otros requerimientos adicionales son los filtros de acondicionamiento al multiplexaje para garantizar que las señales analógicas de entrada dispongan de un rango dinámico.

En la figura 5.3 se muestra un modelo de un sistema de adquisición y conversión de datos.

- La función del *multiplexor analógico* es de actuar como un interruptor habilitador de varias posiciones el cual periódicamente selecciona una señal analógica de entrada y la direccionada al circuito de muestreo y retención [SMIT74].

  Teorema del muestreo de Nyquist: Si una señal x(t) no contiene componentes de frecuencia para frecuencias por encima de f = W Hz, entonces queda completamente descrita por valores instantáneos de muestra, espaciados uniformemente en el tiempo y un periodo $T_s \leq W/2$ o $f_s \geq 2W$ ($f_s \geq 2 f_c N$) [SMIT74].

  $$X_{\beta}(t) = x(t) \sum_{-\infty}^{\infty} \beta (t - n T_s) \qquad f_s \geq 2 f_m$$

  T_s es el periodo de muestreo (para cada tiempo que aparece el nuevo pulso rectangular a muestrear), W es el ancho de banda, N número de señales o canales analógicos, f_c componente de más alta frecuencia presente en las señales analógicas de entrada. La señal se puede reconstruir exactamente de la forma de onda de muestra haciéndola pasar por un filtro ideal de paso bajo, con un ancho de banda B, donde $W < B < (f_s - W)$ con $f_s = T_s^{-1}$. La frecuencia 2W se conoce como la frecuencia de *Nyquist* [SMIT74].

  El convertidor A/D le asigna a cada muestra analógica un valor digital equivalente, este proceso se lo conoce como *cuantización*. El *muestreo* y la *cuantización* es la base de la existencia de cualquier sistema de adquisición y conversión de datos. Para el caso de existir varias señales analógicas o canales de datos, este parámetro depende del ancho de banda de las señales de entrada, del número de canales analógicos y del número de muestras por ciclo.

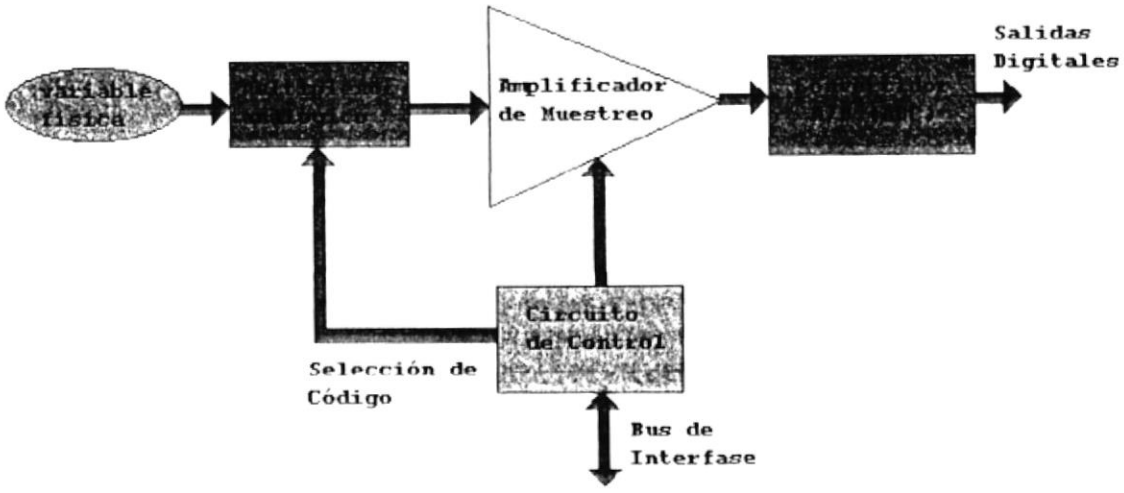


Figura 5.3 Modelo de un Sistema de Adquisición y Conversión de Datos

Una señal analógica se muestrea memorizando su amplitud instantánea a intervalos regulares e ignorando el tiempo restante. Cada muestra representa la amplitud de la señal analógica en un instante especifico de tiempo. La velocidad a la cual se realiza la actualización de las muestras se la denomina frecuencia de muestreo (f_s). En la práctica el muestreo debe realizarse a frecuencias por encima de la frecuencia de Nyquist. Al conseguir una frecuencia de muestreo mayor a la de Nyquist entonces se consigue facilitar la separación o filtrado de la señal mensaje original. El uso de una frecuencia de muestreo inferior a la establecida por el criterio de Nyquist perjudica y causa pérdidas de información [.

Cuantización: Una vez que se realiza el proceso de muestreo el siguiente paso es la digitalización de la señal analógica y esto se lo conoce como *cuantización*. La cuantización consiste en codificar todos los posibles valores de amplitud que pueden tomar las muestras analógicas presentes a la entrada del convertidor A/D en un

número limitado de amplitudes representados por un código binario BCD específico de la salida [SMIT 74].

El número de bits de salida que disponga el convertidor A/D determina el número máximo de intervalos y códigos de cuantización., además de este parámetro se fija la resolución del convertidor es decir el incremento de amplitud más pequeño que la señal puede disponer al sistema. Para un convertidor A/D que disponga de 8 códigos de salida significa que dispone de 8 bits a la salida, 3 bits de codificación binaria BCD, y 7 intervalos de cuantización. En general 2^N códigos de salida, N bits de codificación binaria, y 2^N -1 intervalos de cuantización.

La resolución es un parámetro muy importante cuando se va a elegir un convertir A/D debido a que de ello se obtendrá un valor de muestra más precisa; por la tanto una respuesta digital más exacta. La resolución de un convertidor A/D está relacionada con el número de bits de salida del convertidor . Podemos expresar la resolución de un convertidor A/D como:

$$\textbf{Resolución (V)} = \frac{V_{FSR}}{2^N - 1} \qquad \textbf{(ec. 5.1)}$$

$$\textbf{Resolución (\%)} = \frac{100}{2^N - 1} \qquad \textbf{(ec. 5.2)}$$

Donde $V_{FSR} = V_{MAX} - V_{MIN}$ el rango dinámico a escala completa de la señal analógica.

En la tabla 5.1 se hace la comparación de resoluciones para convertidores de 8 hasta 16 bits de salida, con un voltaje de amplitud de la señal analógica de 10V.

RESOLUCION		1LSB (VFSR = 10 V)	
Bits de codificación	Códigos de Salida		
N	2^N	**%**	**mV**
16	65536	0.0015	0.152
12	4096	0.0244	2.44
11	2048	0.0489	4.89
10	1024	0.0978	9.78
9	512	0.1957	19.6
8	256	0.3921	39.2

Tabla 5.1 Resolución de los Convertidores A/D

La exactitud de un proceso de conversión analógico a digital depende muchísimo del número de salidas digitales que el convertidor disponga. Dependiendo del tipo de precisión que se desee obtener entonces se elegirá el número de bits de salida que tenga un convertidor A/D.

5.1.3 CONVERTIDOR ANALOGICO A DIGITAL ADC 0808.-

Un convertidor A/D clásico estándar es el ADC 0808 el cual dispone de 8 entradas IN analógicas para convertir una palabra de 8 bits a la salida. El convertidor ADC 0808 dispone de 28 pines y sólo puede leer cada vez un sólo canal de los 8 posibles, la razón es porque dispone de 3 líneas de multiplexamiento para la selección de la entrada, las líneas de direccionamiento de entrada se las codifica como: ADD C, ADD B, ADD A, pines 23, 24, y 25. Una vez que se selecciona la señal analógica de entrada, entonces es necesario decirle al convertidor que proceda a realizar la conversión analógica a digital esto se lo realiza habilitando la señal de START pin 6.

Realizado este paso entonces el convertidor ADC 0808 se toma de un tiempo de 100 µseg para realizar la conversión digitalizada. Una vez que este proceso ha concluido entonces el convertidor habilita una señal (Output enable) OE pin 9 de reconocimiento de fin de conversión del proceso EOC (End of code) . Si se desea que el convertidor ADC0808 este continuamente habilitado para realizar conversiones entonces se debe colocar un nivel lógico 1 en la señal OE.

Los niveles de amplitud de los voltajes máximos y mínimos de una señal analógica a ser muestreada se la dispone en las señales REF + pin 12 (V_{MAX}) y REF - pin 16 (V_{MIN}).

Con los suficientes datos que hemos recopilado hasta ahora del uso del puerto centronics, y de la programación en lenguaje C, es factible poder monitorear una variable fisica externa como por ejemplo temperatura, esto proceso consiste en leer un valor analógico de voltaje proveniente de un sensor de temperatura, termocupla o un transductor, pasarlo a un convertidor A/D, luego direccionar la entrada analógica que se desea convertir por medio de la utilización del puerto centronics y del conveniente uso del lenguaje C programando el software apropiado. A continuación se diagrama cada uno de los bloques electrónicos tanto analógicos como digitales de un sistema de monitoreo lazo cerrado de variables fisicas externas para que conjuntamente sus salidas acopladas al puerto centronics se pueda capturar un dato analógico e interpretarlo en un PC y poder tomar una acción de control sobre un proceso.

El software necesario para realizar el control queda a la libre imaginación del usuario. La figura 5.4 describe cada uno de los pines del convertidor ADC 0808.

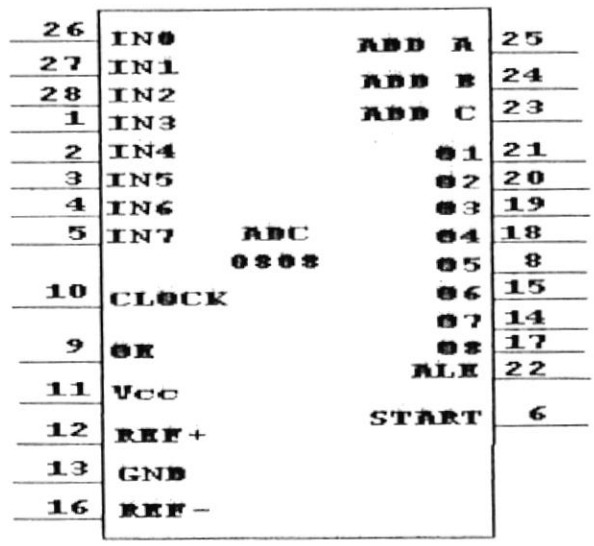


Figura 5.4 Convertidor Analógico a Digital ADC 0808

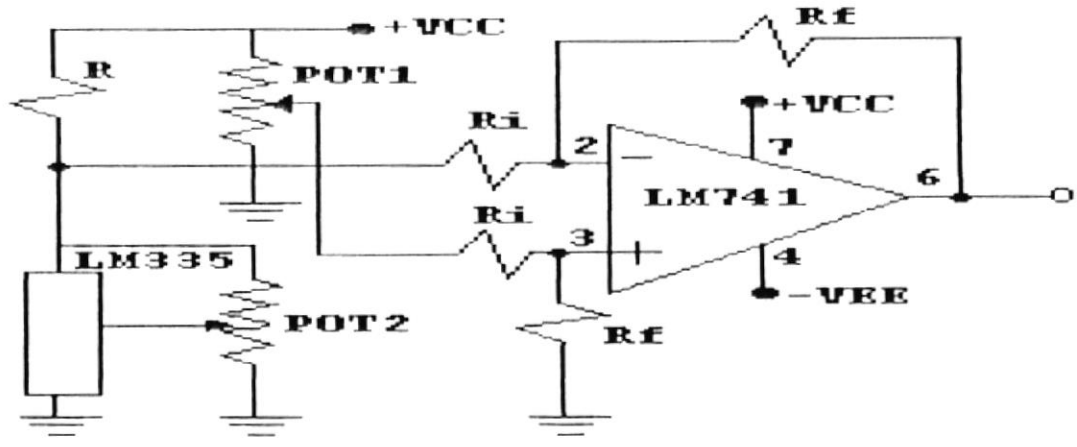


Figura 5.5 Transductor Sensor de Temperatura a Voltaje

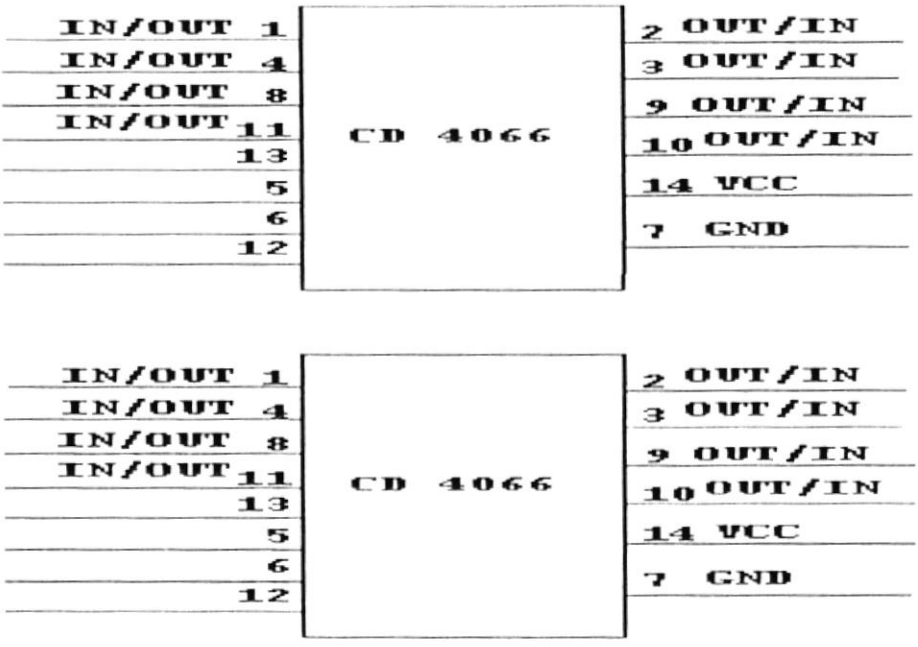


Figura 5.6 Interruptor Tri State como Interfase de Direccionamiento de las Salidas del Convertidor ADC 0808 al Puerto Centronics DB-25 F

CONCLUSIONES Y RECOMENDACIONES

En base a los objetivos de investigación, podemos realizar las siguientes conclusiones:

1. De la investigación realizada es concluyente que es posible diseñar un sistema de control sobre cargas de potencia y conseguir óptimos resultados sobre el funcionamiento de una carga AC. Las aplicaciones del sistema son variadas, en realidad se lo puede utilizar en infinidades de condiciones de control para la automatización, su uso está enmarcado a las disposiciones de un sistema de control lazo abierto o lazo cerrado. La conceptualización de este sistema está ahora implementado y utilizado en los controles automáticos de bancos y centros comerciales como dispositivos de ahorro de energía. En el caso de la industria son utilizados como sistemas de control en motores de inducción, donde la presencia del hombre necesita ser reemplazada por un sistema de control automático permanente.

2. Existe una característica importante en lo referente a seguridad para este diseño electrónico, y esta consiste en la separación física de la circuitería de control; de la de carga, esto se logra gracias al eficiente uso de los dispositivos electrónicos optoaisladores.

3. La versatilidad del sistema se presta para poder realizar cualquier modificación, corrección, e innovación en el diseño propio, tanto electrónico como por software. Hoy en día la mayoría de personas disponen de un computador, herramienta fundamental para la realización de un trabajo de investigación. Es este el punto de partida donde inicia la idea de poder diseñar y crear un sistema que sea accesible por la mayoría de los usuarios, sin necesidad de que tengan suficiente conocimiento del uso de un computador.

4. La eficiencia y seguridad de operación del sistema son satisfactorias, hoy en día lo que se busca es que se tengan resultados positivos y eso es lo que se consigue con la realización de este trabajo de investigación. No existe peligro alguno para el PC en caso de que se produzca cualquier daño en la tarjeta que gobierna a las interfases electrónicas de potencia. Las tensiones de voltaje alternas de alimentación en la carga no son vistas en las entradas digitales del computador, ambos sistemas son completamente independientes y seguros de operar.

5. La infraestructura de la interfase en su totalidad es visualizado por el diseñador como un sistema de lazo abierto, similar a un sistema *mandato ejecución* característico en los sistemas de control modernos como la robotización. Para tener una idea más clara del funcionamiento de éste sistema en particular podemos imaginarnos un PC con su respectivo CPU, y cuando le damos una orden específica a realizar por teclado; éste la ejecuta como un proceso mandato- ejecución, es 7similar el sistema de la interfase aquí diseñada, una vez realizada la configuración del modelo conceptual, el sistema se gobierna por sí sólo. Dependiendo de las necesidades que se tenga se puede implementar sistemas automáticos de control en lazo abierto o lazo cerrado. Este trabajo en particular permite obtener un sistema de lazo cerrado si ese fuere el caso, realizando unas pequeñas modificaciones a la estructura del software y al diseño electrónico, aquí radica una característica importantísima de todo diseño cual es la versatilidad.

A continuación presento algunas recomendaciones para las futuras implementaciones con el fin de aumentar el aprovechamiento y ventajas del SAC:

1. Es necesario que el cableado que va del puerto I/O centronics DB-25F hasta la interfase de control sea del tipo apantallado y blindado, esta condición es muy importante para que las señales de control no perciban ningún tipo de atenuación o pérdidas. Es conocido por todos que los efectos capacitivos en el elemento conductor perjudican al funcionamiento de las interfases digitales es por ello que la distancia máxima del cable conector con el puerto no exceda a los 2.5 metros. Esta limitante de la distancia del cable en realidad no es un problema, pensaríamos inmediatamente que el CPU y la carga de potencia ¿ sólo pueden estar separados por esta distancia?. La respuesta es no, existe una solución a este limitante y es de tipo electrónico, aquí se recomienda la utilización de una interfase convertidora de niveles lógicos TTL a CMOS conocida como MAX232, es muy fácil disponerla previamente a la salida del puerto DB-25F, y además realizar las modificaciones necesarias de acoplamiento de los niveles lógicos CMOS – TTL entre interfases.

2. Es inteligente y conveniente disponer de un software de simulación de circuitos electrónicos, éste software debe tener la característica de configurar simulaciones para dispositivos digitales y analógicos, individual o mixtos en lo referente a la circuitería, esto facilita realizar las pruebas de funcionamiento y operación del sistema. Ahora si no se lo dispone tampoco es perjudicial lo único que hace es que se seamos más juiciosos en la elaboración del diseño.

3. Se debe seguir los pasos de ejecución recomendados en la sección de APENDICES esto requiere elegir convenientemente cual es la interfase de potencia más apropiada para que se ejerza control sobre una carga. Elegir conveniente el tipo de interfase digital en lógica positiva o negativa según las necesidades.

APENDICES

MANUAL DEL USUARIO CONTROL DE LA INTERFASE POR SOFTWARE

1. MANUAL DEL USUARIO.-

Vamos a describir los pasos necesarios para poder ejecutar correctamente el sistema automático de control de cargas con todos sus periféricos.

1.1 REQUISITOS PARA INGRESAR AL SISTEMA.-

Entre los requisitos indispensables para poder ingresar al SAC debemos disponer de un PC que se encuentre correctamente configurado e instalado. El presente proyecto se lo ha realizado en un computador de las siguientes características:

- Un PC de preferencia
- Microprocesador Intel Pentium II 333 Mhz.
- Memoria de Acceso Aleatorio RAM: 64 MB.
- 8 MB memoria de video.
- Tarjeta Multipuertos, paralelo: LPT1, LPT2, LPT3, LPT4.
- Disco Duro 6 Gigabytes.
- Drive de 3 1/2".
- Tarjeta aceleradora Gráfica de 64 bits.
- Sistema Operativo: Windows 98, MS-DOS
- Monitor a color UVGA.

1.2 PASOS DE EJECUCIÓN DEL SISTEMA.-

Los pasos requeridos para realizar un control automático de carga son:

1. Crear un directorio en disco, y cargar el programa SAC.EXE dentro del directorio. Si lo que se desea es; que el control sobre la carga continúe aún después de un corte de energía, entonces cargamos el programa ejecutable dentro del archivo AUTOEXEC.BAT del sistema. El contenido del archivo se vería:

```
C:\NDW\IMAGE.EXE
@ECHO OFF
SET LMOUSE=C:\COMPAQ\MOUSE
SET BLASTER=A220 I5 D1

MODE CON CP PREP ((850)C:\WINDOWS\COMMAND\EGA.CPI)
MODE CON CP SELECT=850

KEYB LA,850,C:\WINDOWS\COMMAND\KEYBOARD.SYS

LH C:\WINDOWS\SMARTDRV.EXE 2048 1024

LH C:\WINDOWS\COMMAND\DOSKEY.COM
LH C:\COMPAQ\SUPPORT\LLHB.EXE
PATH=C:\NDW;c:\pspice\psp;C:\MSIMEV60;C:\WINDOWS;C:\WINDOWS\
COMMAND;C:\LOGIC; SET SYMANTEC=C:\SYMANTEC
SET TEMP=C:\TEMP

C:\SAC.EXE
```

2. Correr el programa PSV.EXE y obtener de él la dirección del puerto paralelo que habilita el sistema, recordar que el dato que se desea es: LPT1 dirección 0378h. Si la dirección del puerto fuere diferente a la especificada entonces modificar la configuración sistema por la dirección requerida; o simplemente cambiar la dirección designada en el código fuente del programa PSV.C por la que el sistema dispone. Luego dar la orden de salir del programa.

3. Realizar la interconexión entre el puerto centronics DB-25 F y la Interfase Digital TTL de Prueba para Control de Periféricos (figura 4.5). Correr el programa ejecutable PSV.EXE de prueba de control de puertos I/O. El código fuente del programa de prueba está escrito en la sección 4.2.4 Sentencias de Control de Puerto en Lenguaje C. El objetivo de realizar el literal 3 es comprobar que las salidas de control del puerto funcionen correctamente, sólo si ese fuere el caso entonces procedemos al siguiente literal, caso contrario verificamos el software y la IP. Luego dar la orden de salir del programa

4. Correr el programa SAC.EXE y programar los tiempos de energización y desenergización para cada una de las cargas (salidas O1, O2, O3, O4). Comprobar que la Interfase Digital TTL de Prueba para Control de Periféricos (figura 4.5) funciona correctamente con la Interfase por Software. Dar la orden de salir de la IP y ejecutar desde el editor DEBUG la sentencia -O378,0.

5. Realizar la interconexión de la Interfase Electrónica de Control Optoaisladora (figura 2.2) con el puerto I/O centronics DB-25F.

6. Antes de poner en funcionamiento a una carga (motor ac) se debe verificar los datos de placa del motor, este es : corriente de arranque, corriente de operación, tensión de bornes. Sólo si los datos nominales de placa del motor son menores a los valores de puesta a funcionamiento de la IECO se procede a conectar el motor ac de inducción (120V/220V) en los terminales de fuerza. Según los datos de placa de la carga, se debe seleccionar la interfase de potencia. Es recomendable antes de realizar un control sobre una carga resistiva inductiva (motor ac), realizar una prueba de control sobre una carga resistiva pura, como un banco de luminarias de 110/220 V - 500W.
7. Correr el programa SAC.EXE

1.3 INGRESO DE LOS DATOS EN LA INTERFASE POR SOFTWARE.-

Los pasos esenciales a seguir para programar los tiempos de control son:

1. Al correr el programa ejecutable SAC.EXE la interfase por software que se visualizará será como la de la figura AP-1.1, como podemos observar todos sus datos se encuentran inicializados y listos para tomar un valor de tiempo de control. El indicador del prompt de ESTADO 0/1 se encuentra encerado, de manera que las salidas del puerto I/O están en un nivel lógico "0".

Figura AP-1.1 Interfase Gráfica de Control por Software Inicialización

2. Vamos a ingresar los cuatro tiempos de control de la carga O1. Esto significa poder disponer un control de energización y desenergización sobre la carga dos veces. Si lo que se desea es energizar y desenergizar una sola la carga, entonces sólo se dispondrá sobre dos tiempos de control, los otros dos tiempos permanecerán encerados.

A partir de la figura AP-1.1 se digita la opción "<M>" MODIFICAR, aquí cambia la pantalla gráfica como se muestra en la figura AP-1.2.

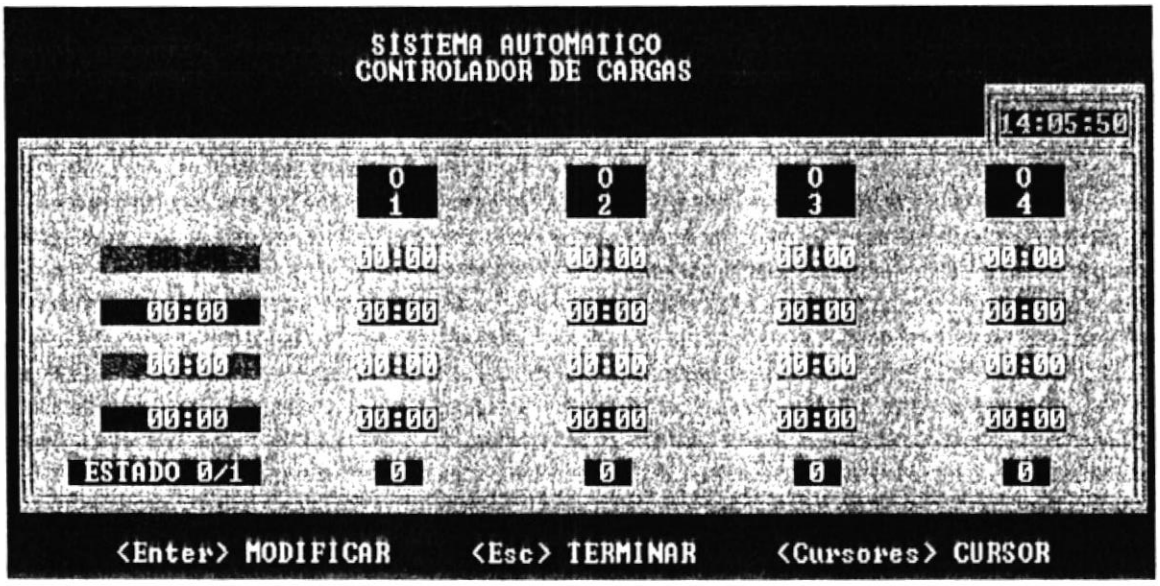


Figura AP-1.2 Pantalla Gráfica Modificación de Datos

En la esquina superior izquierda se pinta en color turquesa, la fila de la carga escogida por default es la fila uno, igualmente por default la columna será la uno.

3. Los marcadores de tiempo son las cuatro flechas de cursor, donde el indicador de horas está gobernado por los cursores : ↑ ↓ la flecha de cursor ↑ sirve para incrementar las horas, y la flecha de cursor ↓ para decrementarlos.

El indicador de los minutos está gobernado por los cursores : → ← la flecha de cursor → sirve para incrementar los minutos, y la flecha de cursor ← para

decrementar. Procedamos a dar un ENTER y aquí cambia la pantalla gráfica como se muestra en la figura AP-1.3.

Figura AP-1.3 Pantalla Gráfica de Ingreso de Datos

Vamos a asumir que los datos a ingresar son:

- Energización de la carga O1 4:05 AM
- Desenergización de la carga O1 5:00 AM

La esquina superior izquierda se pinta en color violeta, la fila de la carga escogida por default es la fila uno, igualmente por default la columna será la uno. Aquí se procede a ingresar los datos con los indicadores del cursor (flechas). Para ingresar el número de la hora (4: 00)se pulsa el indicador ↑ cuatro veces. Para ingresar los minutos se pulsa el indicador → cinco veces. Luego se procede a dar un ENTER para ingresar este dato, y se presiona el indicador ↓ de movimiento de posición, esto es necesario

para pasar a la fila dos columna uno; posición del segundo valor del tiempo de control (desenergizado).

Aquí volvemos a realizar el mismo procedimiento inicial es decir, damos un ENTER y se pulsa el indicador ↑ cinco veces. Se procede ahora a dar un escape de salida de pantalla (tecla ESC). Ahora es necesario grabar los datos ingresados caso contrario se perderán, razón por la cual se presiona la opción "<G>" GRABAR.

Los datos ingresados formarán parte de un archivo inicializador de una base de datos. El resultado de este proceso será la pantalla gráfica como se muestra en la figura AP-1.4.

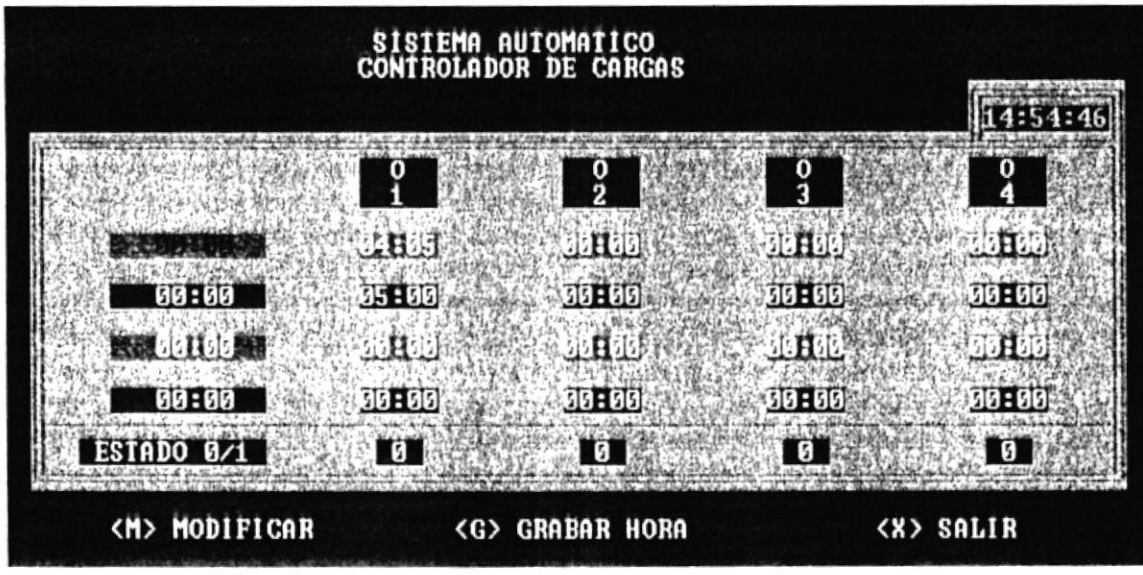


Figura AP-1.4 Pantalla en Ejecución Control de Carga O1

Ahora el control del sistema y sus periféricos depende únicamente del PC. Cualquier modificación de los tiempos de control se lo realiza de la misma manera que como se ingresara un dato por primera vez.

Una pantalla completamente programada para las cuatro cargas se vería como la figura AP-1.5:

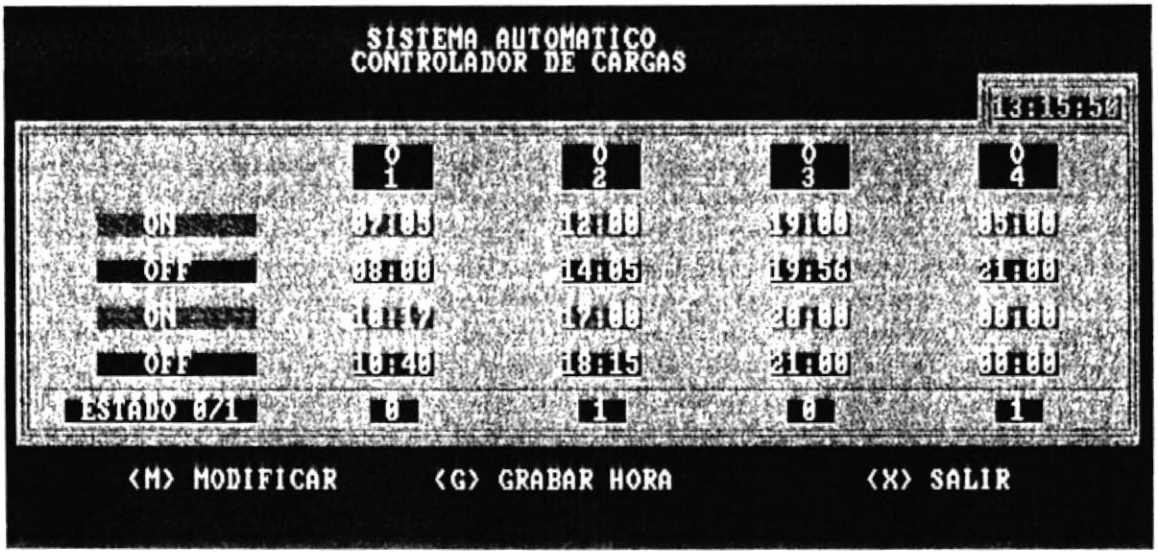


Figura AP-1.5 Pantalla en Ejecución Control de Cuatro Cargas

1.4 GRAFICAS DE SIMULACION DE LA INTERFASE ELECTRONICA SSR DC/AC MOC 3010 Q4015L5 DE MEDIA POTENCIA.-

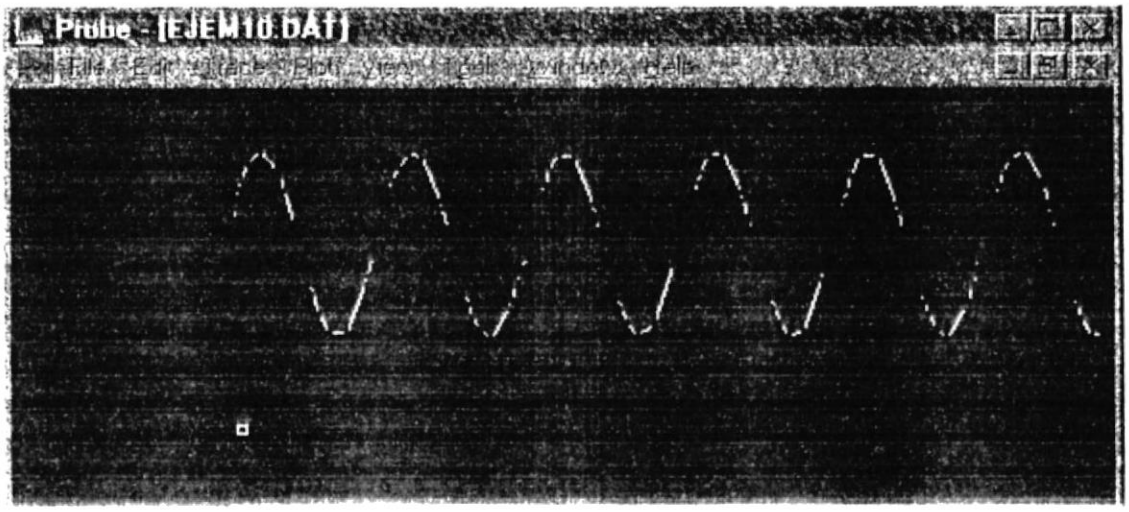


Figura AP-1.6 Voltaje en la Carga Interfase SSR DC/AC MOC3031

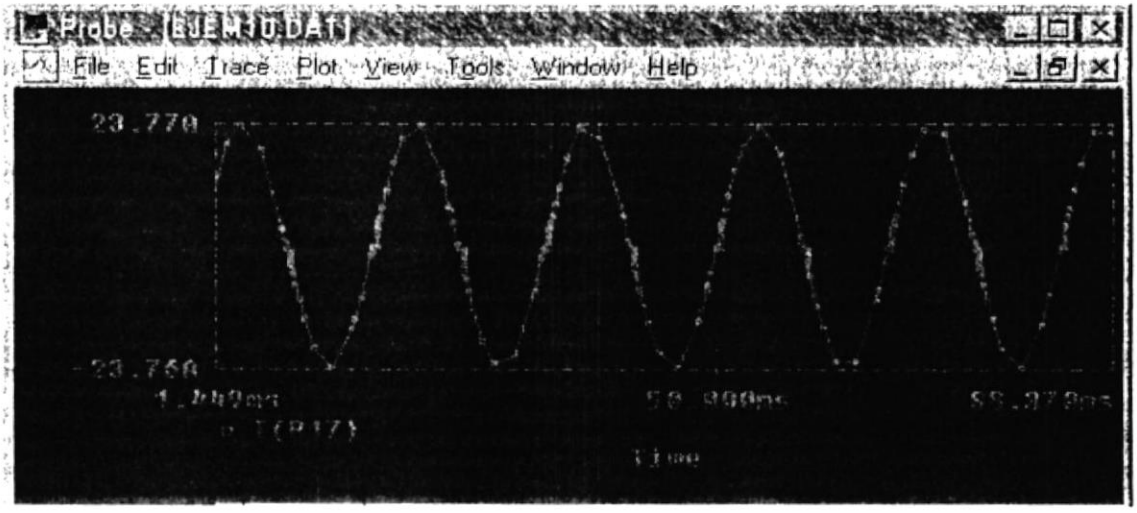


Figura AP-1.7 Corriente en la Carga Interfase SSR DC/AC MOC3031

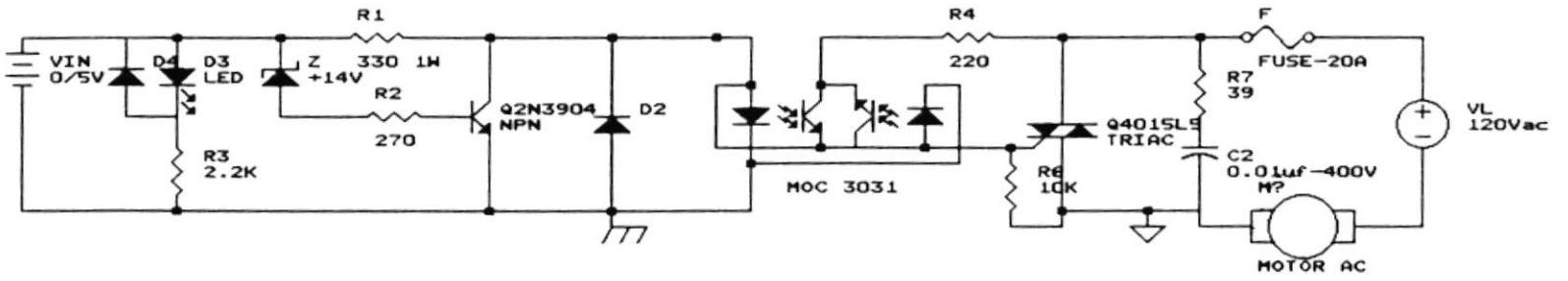


Figura AP-1.8 Interfase SSR DC/AC MOC3031 Q4015L5

INTERFASE DE POTENCIA		
Title: INTERFASE SSR DC/AC MOC3031 Q4015L5		
Size: A	Document Number: 010167	REV: PSV
Date: January 5, 1998	Sheet 1 of 1	

Análisis de resultados del circuito **figura 2.7 SSR DC/AC MOC3031** Optoaislador con Detector de Cruce por Cero e Interfase TRIAC de Media Potencia. El contenido de los resultados se encuentra en el archivo **EJEM10.OUT**

****** 01/05/98 15:03:36 **** Evaluation PSpice (Jan 1997) *********************

C:\MSIMEV80\EJEM10.SCH*CIRCUIT DESCRIPTION********************

****

```
*
* Schematics Version 8.0 - January 1997
* Mon Jan  5 15:03:32 1998
** Analysis setup **
.tran 20ms 100ms
.TEMP 27

* From [SCHEMATICS NETLIST] section of msim.ini:
.lib C:\MSIMEV80\BREAKOUT.LIB
.lib C:\MSIMEV80\NOM.LIB
.lib C:\MSIMEV80\EVAL.LIB
.stmlib C:\MSIMEV80\EVAL.LIB
.stmlib C:\MSIMEV80\NOM.LIB
.stmlib C:\MSIMEV80\BREAKOUT.LIB
.INC "EJEM10.net"
```

****** INCLUDING EJEM10.net ******

*** Schematics Netlist ***

X_U1 $N_0001 0 $N_0003 $N_0004 $N_0002 A4N25
D_D30 $N_0005 $N_0006 D1N4148 0.7V
R_R11 0 $N_0005 2.2k
R_R12 $N_0001 $N_0006 330
D_D31 $N_0007 $N_0006 D1N750 14V
R_R13 $N_0008 $N_0007 270
Q_Q3 $N_0001 $N_0008 0 Q2N3904
D_D32 0 $N_0001 D1N4148
R_R14 $N_0009 $N_0004 220
R_R15 $N_0003 $N_0010 10k
X_X1 $N_0009 $N_0003 $N_0010 2N5444
R_R16 $N_0009 $N_0011 39
C_C1 $N_0010 $N_0011 0.01u
R_R17 $N_0010 0 5
V_V8 $N_0006 0 DC 5V AC 0v 0
V_V9 $N_0009 0 DC 0V AC 0V
+SIN 0V 120V 60 0 0 0
X_U3 $N_0001 0 $N_0004 $N_0003 $N_0012 A4N25
R_R20 $N_0004 $N_0002 1k
R_R21 $N_0012 $N_0003 1k

****** RESUMING EJEM10.CIR ******

.INC "EJEM10.als"

****** INCLUDING EJEM10.als ******

*** Schematics Aliases ***

.ALIASES

X_U1 U1(1=$N_0001 2=0 4=$N_0003 5=$N_0004 6=$N_0002)

D_D30 D30(1=$N_0005 2=$N_0006)

R_R11 R11(1=0 2=$N_0005)

R_R12 R12(1=$N_0001 2=$N_0006)

D_D31 D31(1=$N_0007 2=$N_0006)

R_R13 R13(1=$N_0008 2=$N_0007)

Q_Q3 Q3(c=$N_0001 b=$N_0008 e=0)

D_D32 D32(1=0 2=$N_0001)

R_R14 R14(1=$N_0009 2=$N_0004)

R_R15 R15(1=$N_0003 2=$N_0010)

X_X1 X1(MT2=$N_0009 G=$N_0003 MT1=$N_0010)

R_R16 R16(1=$N_0009 2=$N_0011)

C_C1 C1(1=$N_0010 2=$N_0011)

R_R17 R17(1=$N_0010 2=0)

V_V8 V8(+=$N_0006 -=0)

V_V9 V9(+=$N_0009 -=0)

X_U3 U3(1=$N_0001 2=0 4=$N_0004 5=$N_0003 6=$N_0012)

R_R20 R20(1=$N_0004 2=$N_0002)

R_R21 R21(1=$N_0012 2=$N_0003)

.ENDALIASES

****** RESUMING EJEM10.CIR ******

.probe

.END

******* 01/05/98 15:03:36 ***** Evaluation PSpice (Jan 1997) ****************

*** C:\MSIMEV80\EJEM10.SCH ****   Diode MODEL PARAMETERS**

****

```
        D1N750         D1N4148        X_U1.MainLED   X_U1.PhotoLED
IS    880.500000E-18  100.000000E-15   1.100000E-12   1.100000E-12
N                                      1.9            1.9
ISR     1.859000E-09                  30.000000E-09  30.000000E-09
NR                                     3.8            3.8
IKF                                     .03            .03
BV      4.7           100              6
IBV      .020245      100.000000E-15 100.000000E-06
NBV     1.6989
IBVL    1.955600E-03
NBVL   14.976
RS       .25           16               .66            .66
TT                     12.000000E-09  500.000000E-09
CJO   175.000000E-12    2.000000E-12   40.000000E-12
VJ       .75                             .75            .75
M        .5516                           .34            .34
TBV1  -21.277000E-06

        X_U3.MainLED   X_U3.PhotoLED  X_X1.X1.Dgk    X_X1.X1.Delay
IS      1.100000E-12    1.100000E-12  100.000000E-18   1.000000E-12
N       1.9             1.9
```

ISR	30.000000E-09	30.000000E-09		
NR	3.8	3.8		
IKF	.03	.03		
BV	6			
IBV	100.000000E-06			
RS	.66	.66	5	.01
TT	500.000000E-09			
CJO	40.000000E-12		50.000000E-12	5.000000E-12
VJ	.75	.75		
M	.34	.34		

X_X1.X1.Dak

IS	40.000000E-12
CJO	5.000000E-12

****** 01/05/98 15:03:36 ***** Evaluation PSpice (Jan 1997) ****************

*** C:\MSIMEV80\EJEM10.SCH ****   BJT MODEL PARAMETERS**

****

	Q2N3904	X_U1.PhotoBJT	X_U3.PhotoBJT
	NPN	NPN	NPN
IS	6.734000E-15	10.000000E-15	10.000000E-15
BF	416.4	400	400
NF	1	1	1
VAF	74.03	60	60
IKF	.06678	.26	.26
ISE	6.734000E-15	580.000000E-12	580.000000E-12
NE	1.259	3.75	3.75
BR	.7371	.04	.04

```
NR   1            1            1
ISC               3.500000E-09   3.500000E-09
RB   10
RBM  10
RC   1
CJE  4.493000E-12   2.500000E-12   2.500000E-12
MJE  .2593          .3333          .3333
CJC  3.638000E-12   10.000000E-12  10.000000E-12
MJC  .3085          .3333          .3333
TF   301.200000E-12 1.500000E-09   1.500000E-09
XTF  2
VTF  4
ITF  .4
TR   239.500000E-09 88.000000E-06  88.000000E-06
XTB  1.5            1.5            1.5
```

****** 01/05/98 15:03:36 ****** Evaluation PSpice (Jan 1997) ****************

*** C:\MSIMEV80\EJEM10.SCH ****   Resistor MODEL PARAMETERS**

****

```
     X_U1.TempComp  X_U3.TempComp
R    1              1
TC1  -.01127        -.01127
TC2  43.460000E-06  43.460000E-06
```

******* 01/05/98 15:03:36 *****Evaluation PSpice (Jan 1994) ****************

*** C:\MSIMEV80\EJEM10.SCH **** Voltage Controlled Switch MODEL PARAMETERS**

****

X_X1.X1.Vswitch

RON .016964

ROFF 35.000000E+03

VON 5

VOFF 1.5

****** 01/05/98 15:03:36 ***** Evaluation PSpice (Jan 1997) ****************

*** C:\MSIMEV80\EJEM10.SCH **** INITIAL TRANSIENT SOLUTION TEMPERATURE = 27.000 DEG C**

****

NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE NODE VOLTAGE

(X_U1.1) 0.0000 (X_U1.2) 41.13E-12 (X_U1.4) 14.60E-12 (X_U1.5) 92.20E-12

(X_U1.6) 37.26E-09 (X_U1.7) .0174 (X_U3.1) 0.0000 (X_U3.2) 41.13E-12

(X_U3.4) 95.91E-12 (X_U3.5) 10.90E-12 (X_U3.6) 37.18E-09 (X_U3.7) .0174

($N_0001) .0872 ($N_0002) 37.25E-09

($N_0003) 14.60E-12 ($N_0004) 95.91E-12

($N_0005) 11.15E-09 ($N_0006) 5.0000

($N_0007) 1.3106 ($N_0008) .7665

($N_0009) 0.0000 ($N_0010) 2.180E-12

($N_0011) 0.0000 ($N_0012) 37.17E-09

(X_X1.X1.MT20) 118.1E-18 (X_X1.X1.MT21) 2.180E-12

(X_X1.X1.MT22) 2.180E-12 (X_X1.X1.MT23) 118.1E-18

(X_X1.X1.dlay1) 8.007E-24 (X_X1.X1.dlay2)-28.15E-24

(X_X1.X1.dvdt0) 2.180E-12 (X_X1.X1.dvdt1) 2.180E-12

(X_X1.X1.dvdt2) 2.180E-12 (X_X1.X1.gate1) 6.216E-12

(X_X1.X1.gate2) 2.180E-12 (X_X1.X1.main1) 0.0000

(X_X1.X1.main2) 0.0000 (X_X1.X1.main4) 1.0000

(X_X1.X1.cnhold) 0.0000 (X_X1.X1.cnmain) 0.0000

(X_X1.X1.cntrol) 677.3E-21 (X_X1.X1.contot) 0.0000

(X_X1.X1.dlayr1) 7.969E-24 (X_X1.X1.dlayr2)-28.01E-24

(X_X1.X1.main1r) 0.0000 (X_X1.X1.main2r) 0.0000

(X_X1.X1.cnholdr) 0.0000 (X_X1.X1.cnmainr) 0.0000

(X_X1.X1.cntrolr)-614.3E-21 (X_X1.X1.condvdt) 0.0000

(X_X1.X1.congate) 0.0000 (X_X1.X1.contotr) 0.0000

VOLTAGE SOURCE CURRENTS

NAME CURRENT

V_V8 -1.690E-02

V_V9 4.359E-13

X_U1.v_PhotoLED 4.113E-11

X_U3.v_PhotoLED 4.113E-11

X_X1.X1.Vlak -3.373E-21

X_X1.X1.Vlka 3.373E-21

X_X1.X1.VdVdt -1.073E-28

X_X1.X1.Vlgf 4.347E-13

TOTAL POWER DISSIPATION 8.45E-02 WATTS

JOB CONCLUDED

TOTAL JOB TIME 51.06

1.5 INTERFASE DIGITAL CONTROLADORA PARA 56 CARGAS.-

La presente interfase digital tiene por función disponer de ***56 salidas digitales de control*** las cuales apropiadamente acopladas a un número igual de *entradas digitales*, como es la interfase SSR DC/AC MOC3031 Optoaislador con Detector de Cruce por Cero e Interfase TRIAC de Media Potencia, entonces ***incrementamos el poder de control del sistema***, esta optimización mejora el diseño, y es como para necesidades de automatización industrial.

1.5.1 DESCRIPCION DE LA INTERFASE DIGITAL.-

Básicamente la interfase digital controladora se encuentra conformada por:

- Bus de datos proveniente del puerto I/O centronics DB-25F
- Bus de direccionamiento de salida
- Señal de aceptación de datos
- Sistema Controlador

El bus de datos de ésta interfase una vez conectada al puerto I/O paralelo DB-25F está encargada de recibir las señales de habilitación deshabilitación proveniente del CPU.

Las señales digitales de dato son las que proveen la información a las 8 salidas de los 7 Flip Flop Tri Sate 74LS374, en realidad es la señal mensaje de activación de carga. Todo esto es cierto siempre y cuando se direccione primero al Flip Flop Tri State que se desea activar. Toda la interfase está diseñada para funcionar en lógica positiva.

Por ejemplo si deseamos activar la salida SAL1H que se encuentra gobernada por el segundo F/F 74LS374 de izquierda a derecha, entonces se debe escribir en el bus de direcciones los niveles lógicos 001 (bit menos significativo LSB 1), este bus de

direccionamiento se lo puede disponer gracias a las señales de control STROBE, AUTOLF, INIT.

Una vez direccionado la palabra 001 entonces, a la salida del puerto I/O, en el bus de datos se escribe la palabra 00000001. Luego lo que resta es dar la orden de activación de la salida SALLH, esto es factible por medio de un nivel lógico "1" en la señal de aceptación de datos. Esta *señal de aceptación de datos* la podemos representar por medio de la señal de control SELECTIN.

En el software LOGIC WORK se realiza una simulación del diseño aquí presentado, si se desea realizar una verificación de su funcionamiento referenciar al archivo CIRCUI.CCT.

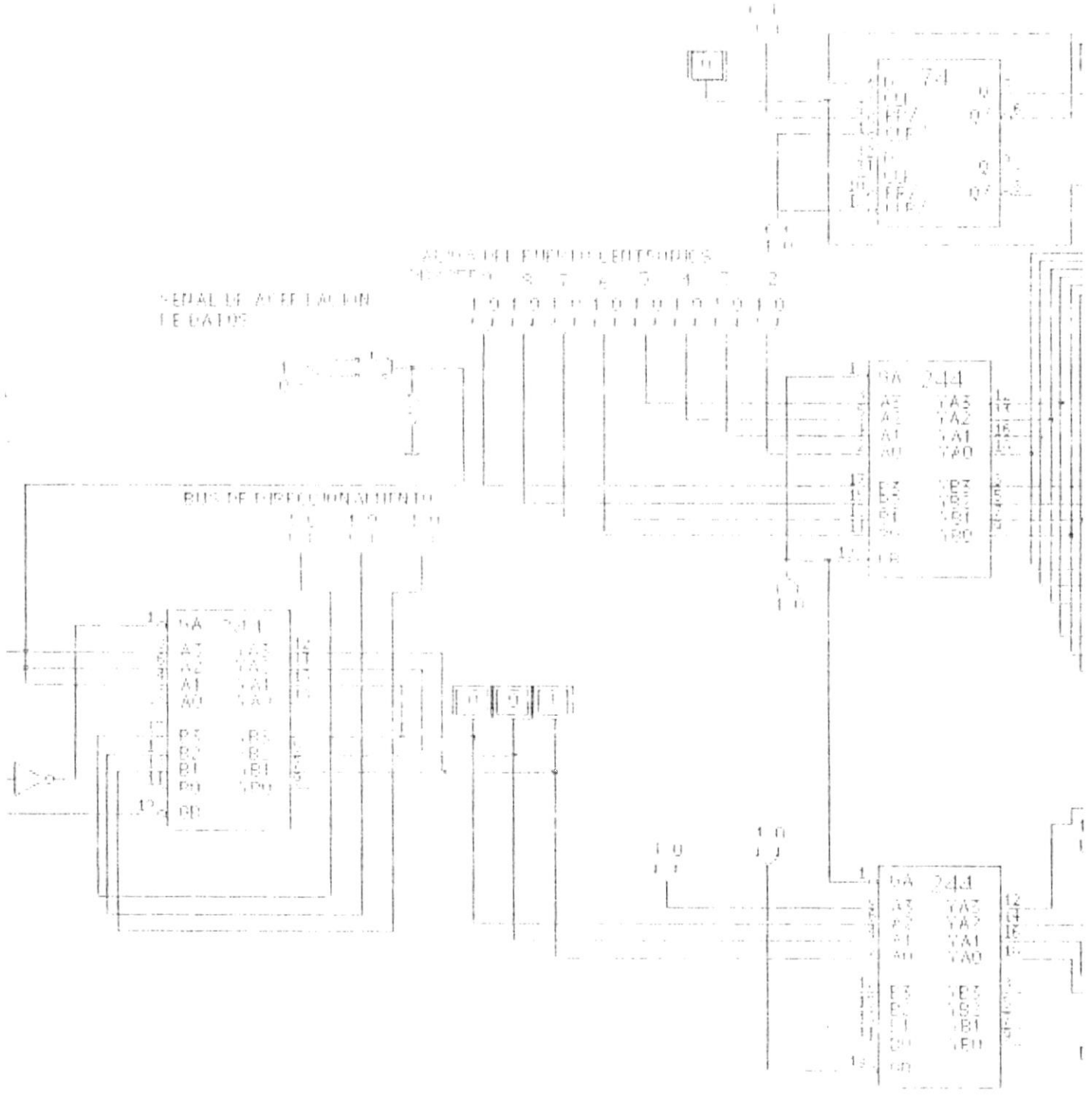


FIGURA AP-1.9 Interfase Digital TTL de Prueba para Control de Periféricos

(ARCHIVO DE SIMULACION DIGITAL CIRC1.CCT LOGIC WORK)

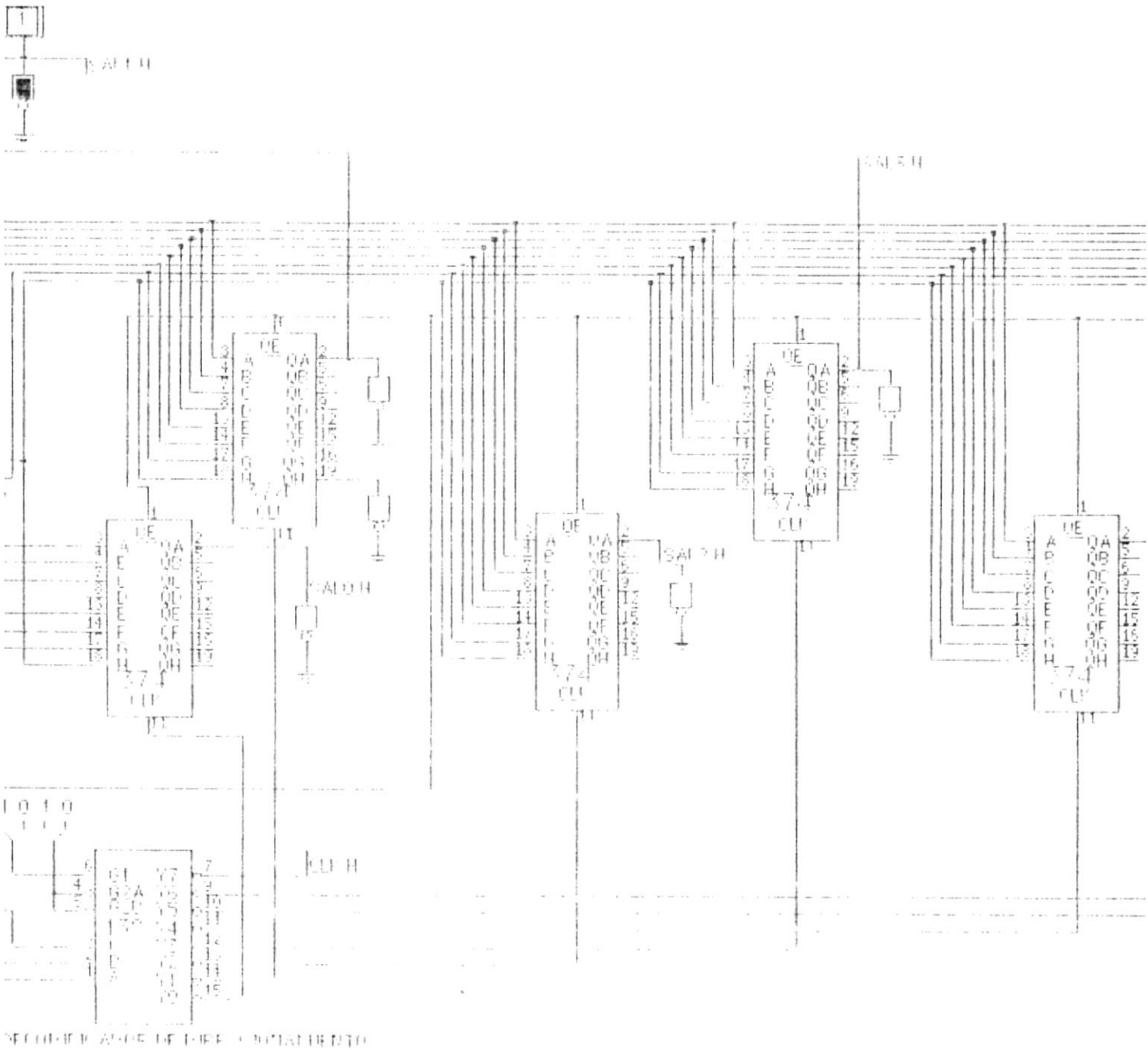


FIGURA AP-1.9 Interfase Digital TTL de Prueba para Control de Periféricos

(ARCHIVO DE SIMULACION DIGITAL CIRCUCCT LOGIC WORK)

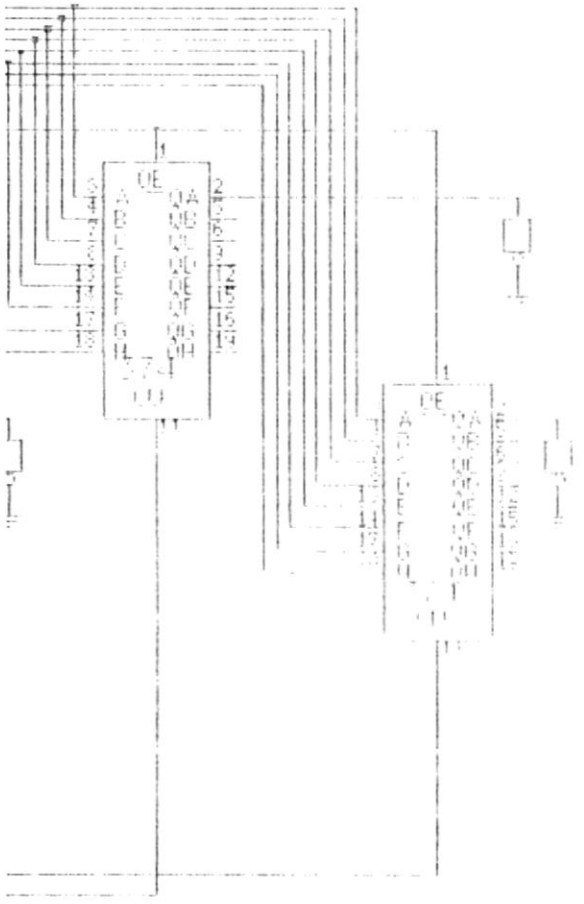


FIGURA AP-1.9 Interfase Digital TTL de Prueba para Control de Periféricos

(ARCHIVO DE SIMULACION DIGITAL CIRC1.CCT LOGIC WORK)

PRIMER EJEMPLO: APLICACIÓN DE SIMULACION DE CONTROL

Para realizar una simulación de control de carga se debe disponer primeramente de la interfase digital TTL de prueba para el control de periféricos, ésta interfase se observa en la figura AP-2.0. La característica de éste esquema digital es de proporcionar las señales de prueba para la simulación de control a los siete hexadecimales inversores Schmitt Trigger IC-74LS14N.

Las entradas del IC se encuentran conectadas a la salida del puerto I/O centronics DB-25F, y estas salidas del puerto I/O están conectadas internamente a la barra del bus de datos del PC. Vamos a realizar una simulación por *control manual*, es decir controlada por el usuario y posteriormente realizaremos un control de las señales por software.

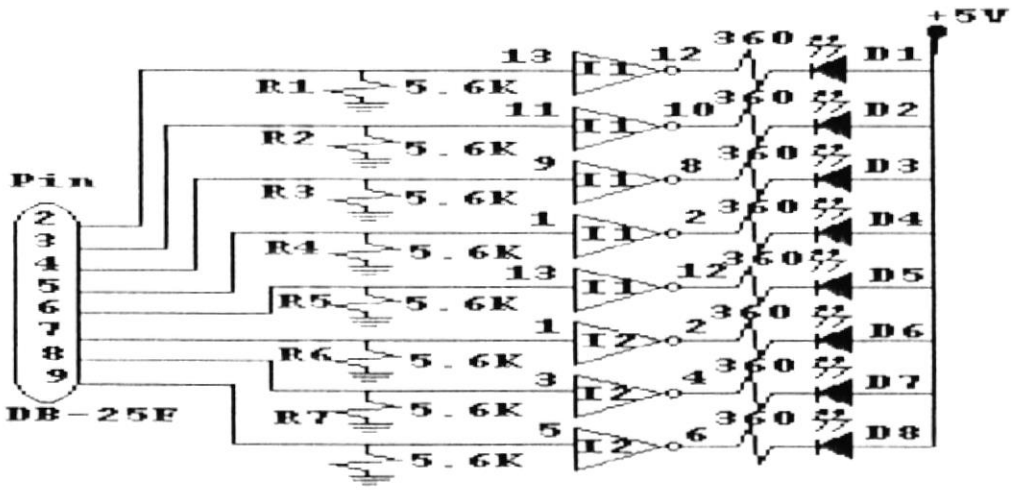


Figura AP-2.0 Interfase Digital TTL de Prueba para Control de Periféricos

Una vez realizada la conexión de la interfase digital TTL con el puerto I/O, entonces se procede a llamar, uno de los comandos del editor DEBUG del sistema operativo MS-DOS, los pasos a seguir para ejecutar el procedimiento de control será:

1.- Llamar al sistema operativo MS-DOS

2.- Utilizar el comando de línea DEBUG

3.- Direccionar la salida a controlar

Simulación por control manual:

Paso 1.-

Realicemos el primer procedimiento; esto es llamar al sistema operativo MS-DOS lo que se va a obtener es una pantalla como la de la figura AP-2.1.

Paso 2.-

C:\>debug [ENTER]

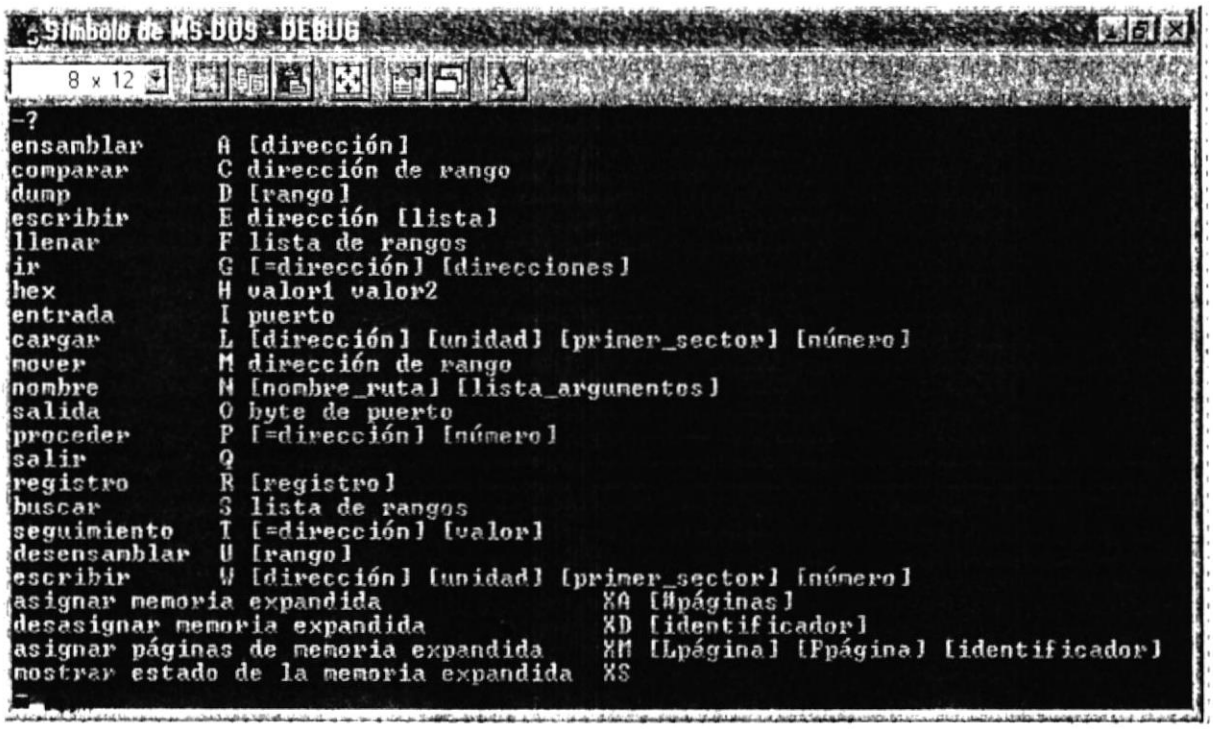


Figura AP-2.1 Comandos del Editor DEBUG Sistema Operativo MS-DOS

Paso 3.-

De todas estas características que proporciona el editor DEBUG del MS-DOS utilizamos el ***byte del puerto de salida*** representado por la letra "**O**". Además del byte del puerto de salida, se necesita la dirección del puerto y el valor del bit. La dirección del puerto LPT1 es conocida y la referimos como 378H.

El valor del bit puede ser: 1,2,4,8,10,20,40,80 (notación hexadecimal) dependiendo de cual es el LED que se desea encender, correspondiente a la interfase digital TTL de prueba para control de periféricos. La instrucción de control se la puede definir con el formato: control del byte de salida, dirección del puerto y el valor del bit

_O[Dirección del puerto, Valor del bit]

Por ejemplo si se desea colocar un nivel lógico "1" LED1 pin 2 del puerto I/O DB-25F, entonces se deberá escribir la sentencia :

_O378,1 ó _o378,1

Por ejemplo si se desea colocar un nivel lógico "1" LED2 pin 3 del puerto I/O DB-25F, entonces se deberá escribir la sentencia :

_O378,2 ó _o378,2

Para apagar o colocar un nivel lógico "0" LED2, pin 3 de la figura AP-1.8 se debe escribir :

_O378,0 ó _o378,0

Es importante destacar que cada *valor del bit de salida*, debe ser escrito en notación hexadecimal, y para definir que bit a la salida permanecerá en un nivel lógico "1" es necesario conocer su respectiva notación binaria.

La codificación hexadecimal del valor del bit de salida la podemos escribir en su equivalente notación binaria del bit del puerto I/O.

Valor del bit hexadecimal	**Bit activo en el I/O puerto**
1	► **0 0 0 0 0 0 0 1**
2	► **0 0 0 0 0 0 1 0**
4	► **0 0 0 0 0 1 0 0**
8	► **0 0 0 0 1 0 0 0**
10	► **0 0 0 1 0 0 0 0**
20	► **0 0 1 0 0 0 0 0**
40	► **0 1 0 0 0 0 0 0**
80	► **1 0 0 0 0 0 0 0**

Simulación control por software:

De igual forma se utiliza la figura AP-2.0, la interfase digital TTL de prueba para control de periféricos se la conectada al puerto I/O. Procedemos a llamar al sistema operativo MS-DOS, o bajo ambiente de Windows 95-97, los pasos a seguir para ejecutar el procedimiento de control será:

1.- Llamar al sistema operativo MS-DOS o Windows 95-97, y direccionar a la raíz C:\>

2.- Ejecutar el programa PSV.EXE Prueba de Control del Puerto Centronics DB-25F.

Paso 1.-

Realicemos el primer procedimiento; esto es llamar al sistema operativo MS-DOS lo que se va a obtener es una pantalla como la de la figura AP-2.2.

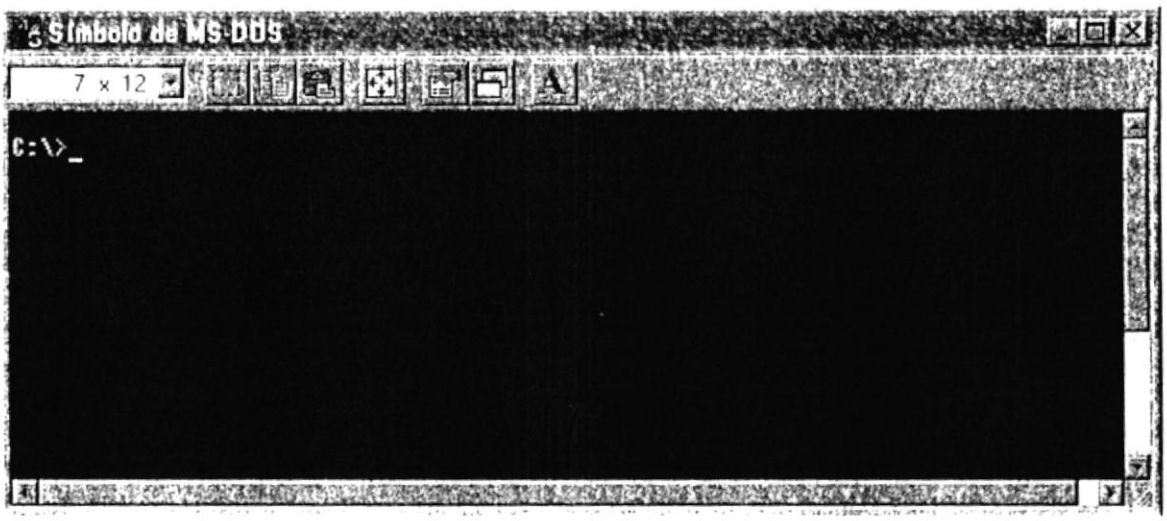


Figura AP-2.2 Sistema Operativo MS-DOS

Paso 2.-

A partir de esta pantalla llamamos al programa ejecutable de prueba para control de puerto I/O, escribimos en la línea del prompt:

C:\>PSV.EXE [ENTER]

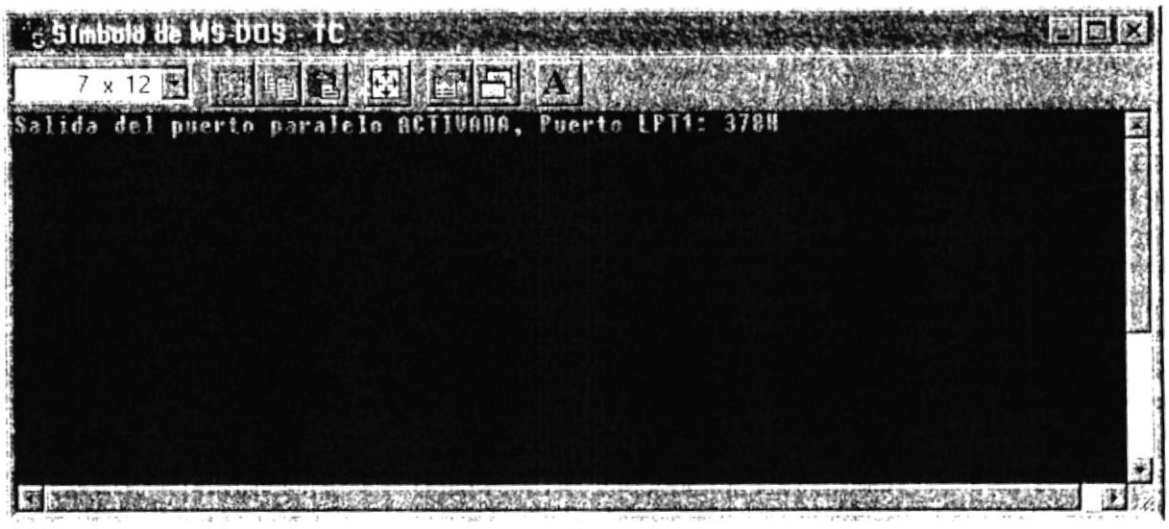


Figura AP-2.3 Programa de Prueba - Control de Puerto I/O PSV.EXE

Al ejecutar el programa de prueba - control de puerto I/O, claramente se puede apreciar que la interfase digital TTL comienza a ser administrada por el software de control que rige su funcionamiento de secuencia de encendido apagado sobre las señales digitales del puerto centronics DB-25F.

La pantalla del programa de prueba - control de puerto I/O figura AP-2.3, proporciona la información en la primera línea de texto, de que el puerto centronics DB-25F ha sido activado, es decir se encuentra plenamente habilitado por el usuario para realizar una operación de control sobre periféricos. Además se proporciona la dirección del puerto I/O activo, para este caso es el puerto LPT1 con la dirección base 378H.

La segunda línea de texto indica al usuario, que debe ingresar por teclado, el número 0 para deshabilitar el bit menos significativo del bus de datos (bit 0-LED D1) pin 2 del puerto centronics DB-25F. Al realizar esta acción el programa automáticamente realiza una simulación completa de todos los bits del bus de datos del puerto I/O figura AP-2.4.

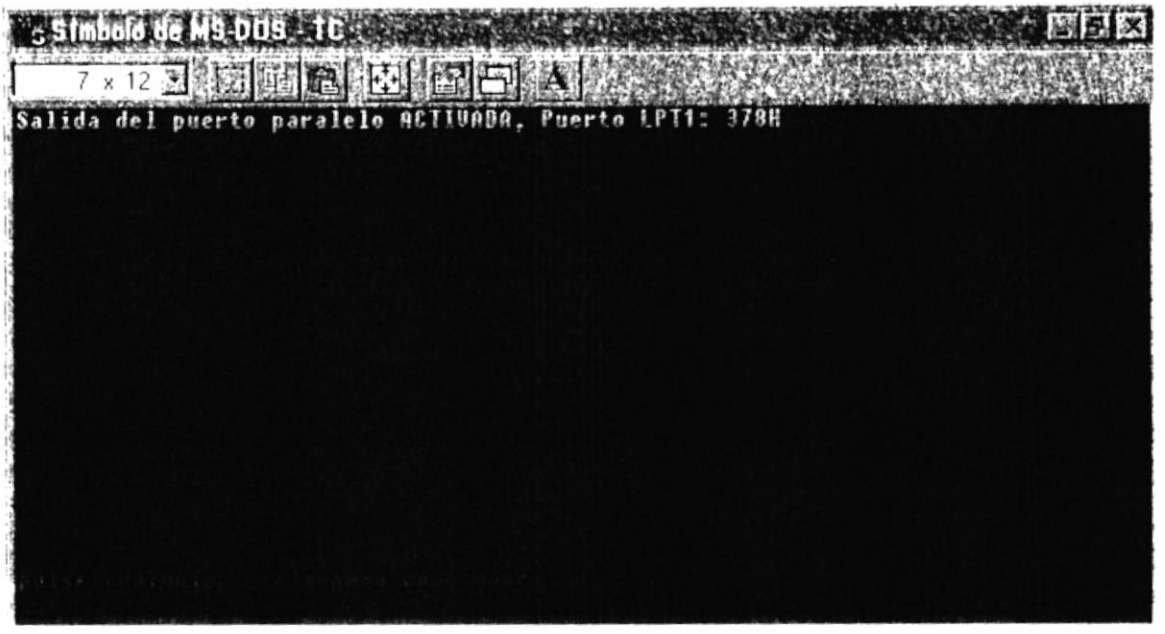


Figura AP-2.4 Programa de Prueba - Control de Puerto I/O PSV.EXE

En la figura AP-2.4 se ingresa el número 0 para apagar el LED1 pin2 de la interfase digital TTL figura AP-2.0. A partir de ésta pantalla el usuario puede observar que existe un secuenciamiento de encendido-apagado de los LED's de la interfase digital TTL.

Estos LED's simulan las señales de control que van a estar presentes en una interfase de potencia, la acción de poder controlar una secuencia de señales digitales provenientes del puerto I/O de un PC, es el principio de la idea de control sobre cualquier dispositivo.

Este ejemplo en particular está diseñado para controlar ocho señales digitales, para el caso de que el usuario desee mejorar el número de señales digitales a controlar, puede utilizar la interfase digital controladora para 56 cargas figura AP-1.9.

SEGUNDO EJEMPLO: APLICACIÓN PRACTICA DEL DISPOSITIVO

Vamos a describir un ejemplo de aplicación práctica para poder realizar un control automático de encendido y apagado sobre una carga luminaria monofásica AC 120V- 60 Hz. Primeramente definimos la *interfase electrónica de control optoaisladora* apropiada a utilizar, la esquematización de los componentes y demás dispositivos necesarios para la implementación del sistema de control se lo puede visualizar en la figura AP-2.5, cabe destacar que este esquema representa la interfase completa de control para cuatro cargas. También se pudo haber elegido otra interfase electrónica de control como por ejemplo: el esquema de la figura 2.7 *SSR DC/AC MOC3031 Optoaislador con Detector de Cruce por Cero e Interfase TRIAC de Media Potencia*, esto es en el caso de que se requiera disponer de mayor potencia en la carga.

Una vez conectada ésta interfase electrónica de control optoaisladora con el puerto centronics DB-25F y a la fase de fuerza de alimentación alterna, procedemos a definir los parámetros de tiempo de energización y desenergización de la carga. Los tiempos de energización y desenergización de la carga, pueden ser seleccionados para un periodo de control de 24 horas. Vamos a disponer de una sola carga para éste ejemplo en particular entonces nuestro control de tiempo será sobre la primera salida O1 de la *interfase gráfica de control por software*. Esta interfase gráfica de control por software se la puede visualizar en la figura AP-2.6.

En ésta figura podemos apreciar claramente que la salida de control O1 muestra cuatro tiempos de encendido y de apagado sobre la carga, inicialmente todos estos tiempos se encuentran encerados a un estado ON - OFF 00:00, y su correspondiente estado 0/1 se encuentra igualmente inicializado a un nivel lógico de 0.

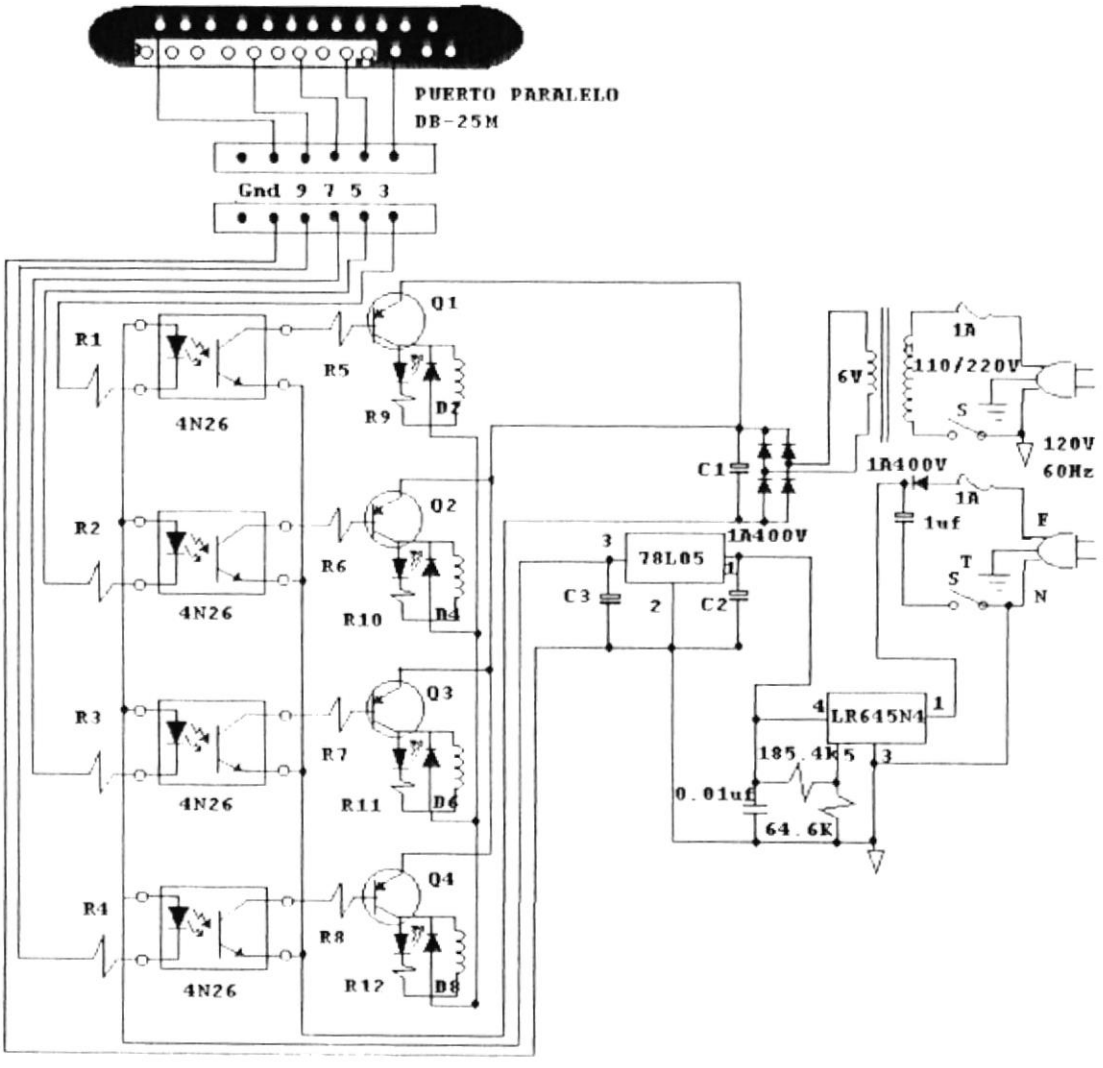


Figura AP-2.5 Interfase Electrónica de Control Optoaisladora

Una vez que se ha invocado al programa: Sistema Automático de Control (SAC) sea esto desde el sistema operativo MS-DOS o bajo ambiente de Windows 95 o 97 se mostrará en la pantalla la interfase gráfica de control por software figura AP-2.6.

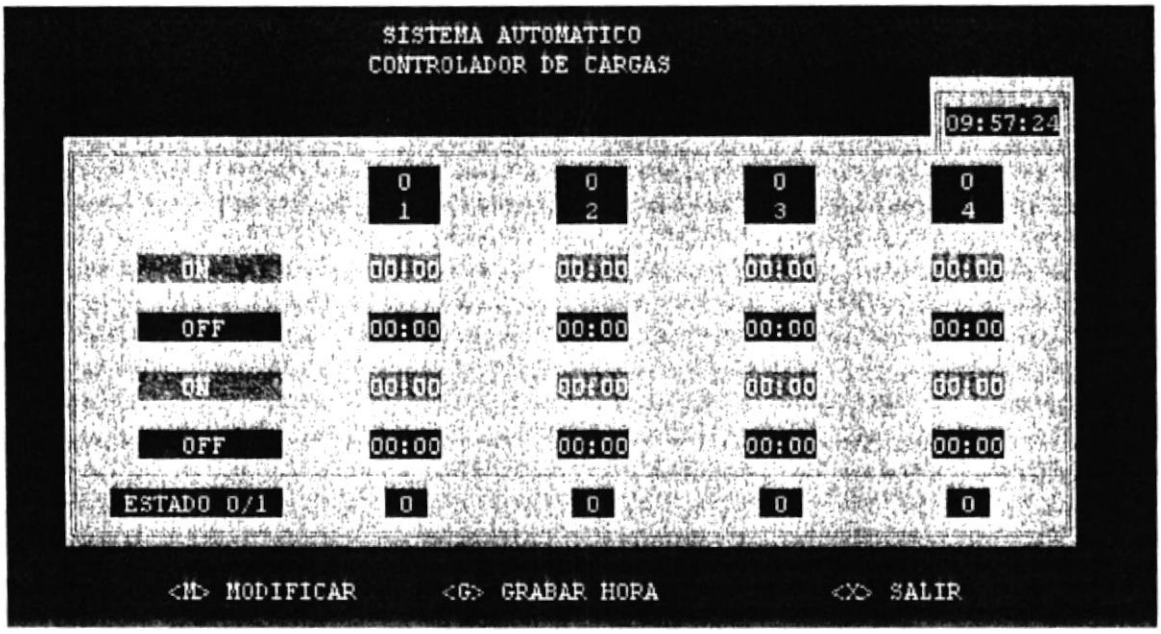


Figura AP-2.6 Interfase Gráfica de Control por Software - Inicialización

Al ejecutar el programa SAC por primera vez, podemos observar que todos los tiempos de control sobre la carga se encuentran inicializados o encerados a cero horas, cero minutos, cero segundos. En la esquina superior derecha se identifica un reloj que se fija automáticamente a la hora actual del reloj del sistema. El parámetro del tiempo está dado en un formato de 24 horas GMT.

En la pantalla por software; propiamente en la columna O1 se muestran cuatro casilleros de datos para los tiempos de control del reloj; sirven para fijar el tiempo de encendido y de apagado de la carga, de esta manera se compara: el tiempo interno de cada controlador de carga, con el reloj del PC, esta comparación continua verifica un control sobre una carga especifica.

El estado 0/1 de energización de cada una de las cargas se la verifica por su representación en lógica positiva como: 1 significa energizada la carga y 0 desenergizada. Cada estado es independiente de los demás, existe una ventaja de ayuda visual para el usuario saber distinguir por el color AZUL / ROJO (0/1) que carga se encuentra conmutada a la salida.

En la figura AP-2.7 se muestra los tiempos programados para el control periódico de una sola carga. La salida O1 visualiza un estado de desactivación en el puerto I/O DB25-F ; por lo tanto este bit de salida del puerto se encuentra en un nivel lógico 0, los demás bits se encuentran en un nivel lógico 1.

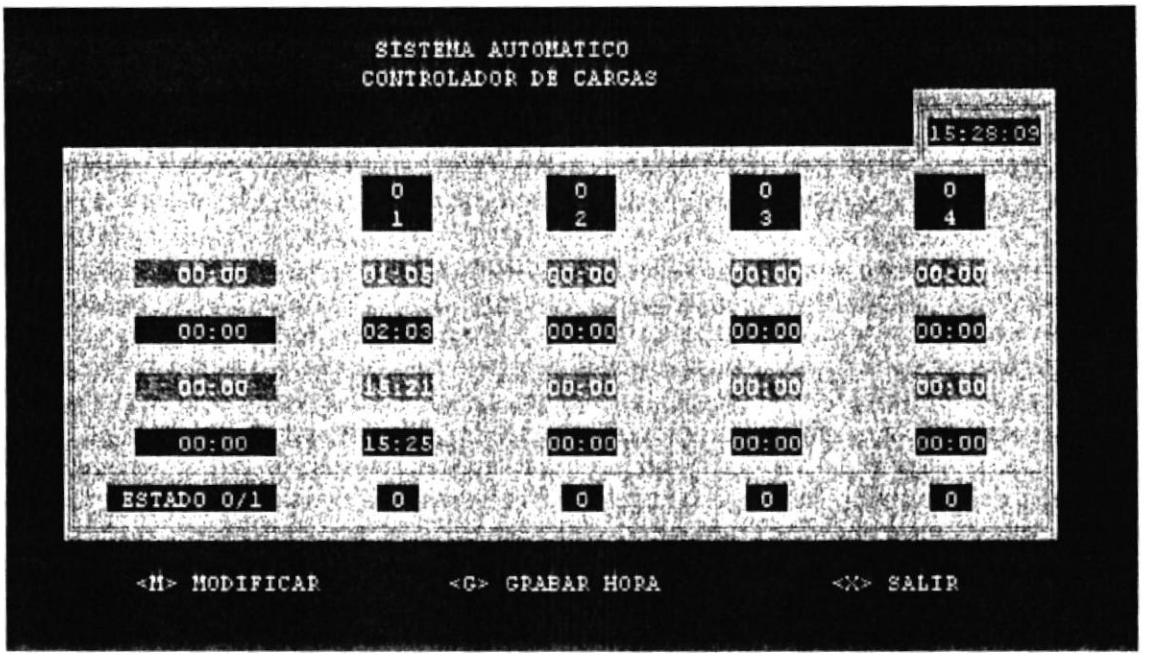


Figura AP-2.7 Pantalla de Ejecución Control de Carga O1

El acceso sobre los tiempos de control se lo puede disponer gracias a la barra de menú con la opción "M" MODIFICAR que se visualiza en la pantalla de ejecución control de carga. Por ejemplo si deseamos ingresar los siguientes datos para la carga O1:

ON 1:05
OFF 2:03
ON 15:21
OFF 15:25

Se debe seleccionar la opción "M" MODIFICAR de la barra de menú que se encuentra en la parte inferior de la pantalla, luego ingresar un "Enter" MODIFICAR por teclado para cambiar el dato seleccionado, automáticamente se fija el ingreso de los datos sobre

la primera fila ON de la carga O1 (el usuario puede visualizar que se encuentra en la selección del primer dato de tiempo; puesto que cambia el color de la barra ON a un color LILA con el código 00:00). Se ingresa el parámetro hora con las teclas del cursor ↑↓ y el parámetro minutos con la teclas de cursor →←. Se ingresa el dato ON 1:05

La barra de menú proporciona una ayuda de texto (barra de color azul) de cómo ingresar los datos, en ella comunica que los datos se ingresan con los cursores y para finalizar el ingreso de los mismos se debe dar un <Enter>.

Una vez ingresado el <Enter> se visualiza nuevamente la barra de menú indicadora de procedimiento y su estado debe ser:

<Enter> MODIFICAR <Esc> TERMINAR <Cursores> CURSOR

Para ingresar el tiempo de desenergización OFF 2:03 se selecciona la tecla de cursor ↓ esto fija el ingreso de los datos sobre la segunda fila OFF de la carga O1 (el usuario puede visualizar que se encuentra en la selección del segundo dato de tiempo; puesto que cambia el color de la barra OFF a un color TURQUESA con el código 00:00). Para ingresar a los datos nuevamente, el usuario debe dar un <Enter> por teclado, se visualiza entonces que cambia el color del código 00:00 a un color LILA. Aquí la pantalla de ejecución control de carga O1 se encuentra lista para recibir los parámetros de tiempo. El procedimiento del ingreso de los datos es exactamente el mismo que cuando se ingresó el primer tiempo, esto es :

Se ingresa el parámetro hora con las teclas del cursor ↑ ↓ y el parámetro minutos con la teclas de cursor → ←. Se ingresa el dato OFF 2:03. El mismo procedimiento se debe seguir para ingresar los otros dos tiempos de energización y desenergización de la carga ON 15:21 - OFF 15:25. Para mantener fijos estos parámetros en un archivo en

disco, se recurre a la opción GRABAR de la barra de menú. Una vez fijadas las horas que gobiernan a la carga se utiliza de la barra del menú la opción TERMINAR para retornar a la pantalla principal de control.

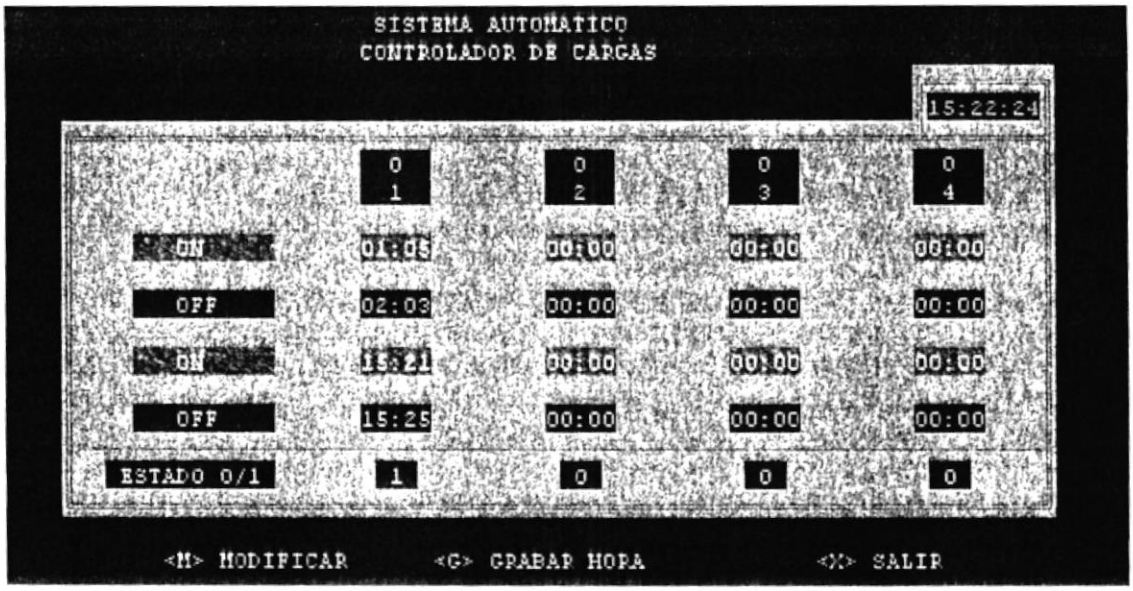


Figura AP-2.8 Pantalla de Ejecución Control de Carga O1

Se observa en la figura AP-2.8 que el software de control proporciona la información de la existencia de un estado de energización sobre una carga, esto lo podemos afirmar y comprobar visualizando los tiempos pre-programados por el usuario y comparándolos con el reloj del sistema del PC. Además el software de control visualiza que existe una carga energizada a la salida (ESTADO 1), debido a que cambia a un color ROJO la representación de la carga O1, se puede observar que también cambia a un color ROJO el indicador de ESTADO 0/1. Hemos elegido este color como indicador de precaución debido a que la carga se encuentra energizada.

Una vez que el tiempo de control de energización y desenergización finaliza entonces la carga debe retornar al estado de desenergización. Ahora la pantalla de ejecución control de carga O1 es la que se muestra en la figura AP-2.9. Esta pantalla proporciona la información del último estado de la carga, por la presencia del color AZUL en la salida O1 y en el indicador de estado concluimos entonces que la carga está desenergizada.

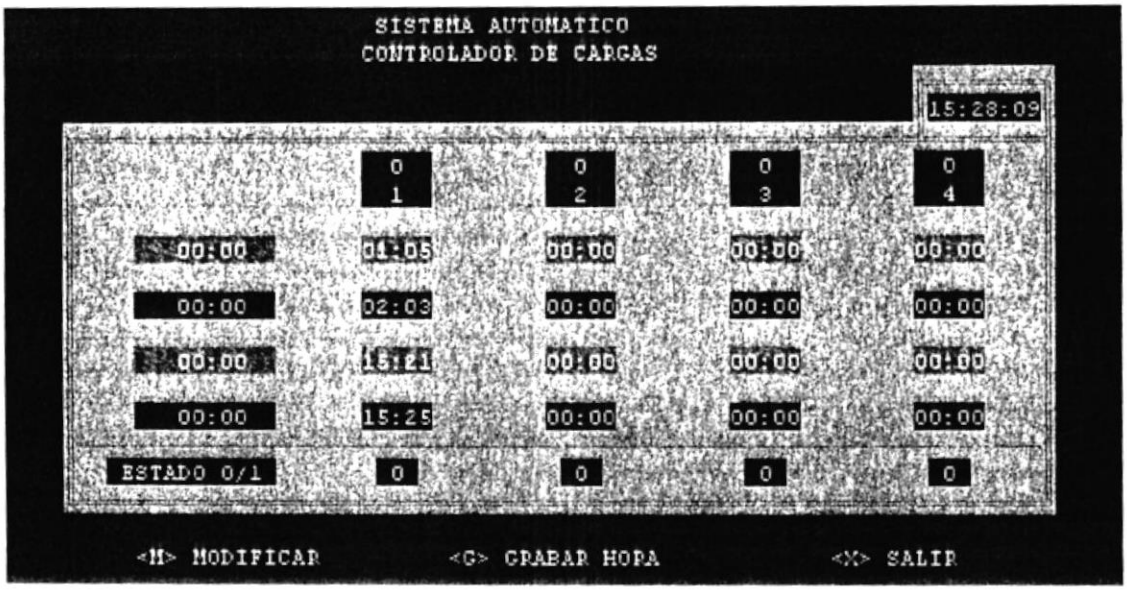


Figura AP-2.9 Pantalla de Ejecución Control de Carga O1

Para el caso en que se hubiere deseado disponer de una carga de media potencia hemos sugerido la utilización del circuito de la Figura AP-3.0. Se debe tener mucho cuidado en realizar las conexiones respectivas con el puerto centronics DB25-F y la interfase de potencia.

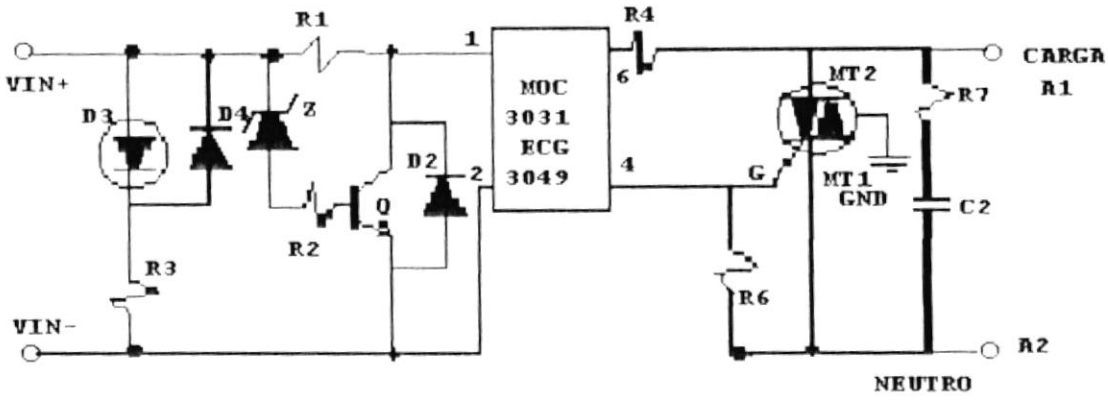


Figura AP-3.0 SSR DC/AC MOC3031 Optoaislador con Detector de Cruce por Cero e Interfase TRIAC de Media Potencia

1.8 ANALISIS COMPARATIVO DE COSTOS

Para realizar nuestro análisis comparativo, hemos elegido a uno de los mejores sistemas de automatización de procesos existente en el mercado, estamos hablando del sistema de automatización de procesos INTOUCH 7.0 de la corporación WONDERWARE.

InTouch de Wonderware, la principal interfaz hombre-máquina del mundo, proporciona una sola visión integrada de todos sus recursos de control e información. InTouch le permite a ingenieros, supervisores, administradores y operadores visualizar e interactuar con el desarrollo de toda una operación a través de representaciones gráficas de sus procesos de producción. La versión 7.0 para Windows NT 4.0 y Windows 95 incluye una serie de características nuevas y actualizadas, incluyendo la referenciación remota de Tags (variables), soporte a ActiveX, manejo de alarmas distribuidas, datos históricos distribuidos con Industrial-SQL Server, interfaz de usuario actualizada.

Adicionalmente, el ambiente de desarrollo de aplicaciones para redes permite el desarrollo de sistemas para su uso en redes a base de PCs. El poder y la facilidad de uso de InTouch disminuye de manera dramática el costo y el tiempo asociados con el desarrollo y mantenimiento de sistemas de interfaz para operador (MMI).

InTouch inclute a FactoryFocus^{MR}, un poderoso nodo de visualización que le permite a supervisores y administradores visualizar (sin alterar) datos del área de producción de la planta en tiempo real desde un PC de escritorio ubicada en cualquier lugar de la red.

InTouch 7.0 también contiene el Paquete de Productividad de Wonderware que incluye a WizGen^{MR} una útil herramienta de software que le ayuda a los usuarios a desarrollar sus propios "Wizards" (objetos pre-configurados) de acuerdo a sus necesidades. El Paquete

de Productividad tiene más de 2000 objetos que hacen que el desarrollo de aplicaciones sea más fácil.

Nuestro análisis de costo incluye todos los dispositivos reales necesarios para implementar un sistema de control de luminarias con un PLC, las características y los precios se detallan a continuación:

ITEM	DESCRIPCION	CANTIDAD	P/U	TOTAL
	CONTROL DE LUMINARIAS PLC		$	$
PC-A984	PROCESADOR: ASSY CPU 8K MEM MODULE	1	1479.60	1479.60
AS-P120	POWER SUPPLY: COMP 984 PS MOD 115/230 VAC	1	227.63	227.63
AS-BDEP	ENTRADA DISCRETA: A120 IO AC IN 16PT 115VAC	1	216.58	216.58
AS-BDAP	SALIDA DISCRETA: A120 IO AN IN 3CH	1	204.43	204.43
AS-BADU	ENTRADA ANALOGICA	1	569.08	569.08
AS-BDAU	SALIDA ANALOGICA	1	1081.80	1081.80
AS-HDTA1	RACK PRIMARIO HSG PRI-CPU 3I/O(2 I/O W/PS)	1	187.85	187.85
AS-HDTA2	RACK SECUNDARIO: HSG SEC 5 I/O	1	187.85	187.85
AS-MBKT0	MB+LINE CONNECTOR KIT	1	41.99	41.99
AS-MBKT1	MB+ TERMINATOR CONNECTOR KIT	1	87.30	87.30
490NAA	TP PVC JACKTD REEL 457M/1500	1	1221.03	1221.03
97-100D	FACTORY SUITE		15000	15000

PROGRAMA	**SOFTWARE PROGRACION PLC**	**1**	**2000**	**2000**
CURSO	**PROGRAMA MODSOFT ENSEÑANZA**		**900**	**900**
	IVA		**10%**	**2340.51**
				$ 25745.65

Ahora si realizamos un análisis comparativo de costos de un sistema profesional PLC versus nuestro sistema discreto encontramos que:

La interfase electrónica de control optoaisladora de baja potencia figura 2.2 tiene un costo de alrededor de los $150, la interfase digital TTL de prueba para control de periféricos figura 4.5 tiene un costo de $15, ahora bien; si se desea disponer de cargas para media potencia entonces se debe utilizar la interfase SSR DC/AC MOC3031 optoaislador con detector de cruce por cero e interfase TRIAC figura 2.7 su costo es de $250, este valor se ve incrementado debido a que ésta interfase utiliza dispositivos tiristores de potencia (TRIAC- DIAC).

Si las necesidades del sistema requieren salidas de control en un número mayor a 8 y menor a 56, hemos propuesto la interfase digital controladora para 56 cargas figura AP-1.9 su costo es de $200. Además de la necesidad de requerimiento de todas estas interfases electrónicas-digitales mencionadas, falta incluir el valor por el desarrollo de la interfase gráfica de control por software figura AP-11.1 su costo realmente depende del esfuerzo, trabajo, y dedicación del diseñador del software.
El sistema propuesto en ésta tesis es discreto en comparación con los sistemas de automatización de procesos profesionales existentes en el mercado. Existen ciertas características que tienen puntos en común y otros no, con el sistema InTouch de Wonderware a continuación detallo los mismos en el siguiente cuadro comparativo:

	SISTEMA INTOUCH 7.0 WONDERWARE	**SISTEMA** CONTROL TIPO PLC
CONTROL DE LUMINARIAS	**Disponible**	**Disponible**
SOFTWARE	FACTORY SUITE	SISTEMA AUTOMATICO DE CONTROL
CONTROL DE TIEMPO REAL	**Disponible**	**Disponible**
ENTRADAS/SALIDAS		
ANALOGICAS I-O	**3-4**	**NO Disponible**
DIGITALES I-O	16-8	3-8
PROCESADOR	ASSY CPU 8K	INTEL - PC
RANURAS DE EXPANSION	**Disponible**	**Disponible**
VISUALIZACION **VIA INTERNET/ INTRANET**	**SCOUT**	**NO Disponible**
VISUALIZACION	VISUAL - GRAFICA	VISUAL
PORTABILIDAD DEL SISTEMA	**Disponible**	**Disponible**

El presente sistema diseñado: cumple con la funcionalidad de desempeñar al menos, una o varias de las funciones que ejecutan los sistemas profesionales de automatización de procesos. En realidad sabemos que para obtener un sistema profesional como el INTOUCH 7.0 han estado involucrados un equipo de por lo menos de más 100 ingenieros electrónicos, e ingenieros en computación entre otros.

CODIGOS FUENTE DE SCRIPTS

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<dos.h>
#include<conio.h>
#include<ctype.h>
#include<math.h>
#include<time.h>
#include<bios.h>

 // Controlador de cargas Monofásicas - Bifásicas
 //  Uso del puerto Centronics de un Pc
 //  Pablo Salvatierra

struct tiempos
  {
        long primero, segundo, tercero, cuarto;
  };

// Declaración de funciones a utilizar en el programa
 void abrir (void);
 void colorear(void);
 void convierte(void);
 void convierte2(void);
 void cambiarHoras(void);
```

```
void grabar(void);
void modifica(void);
void colorear(void);
void horaActualizada(void);

//Disponer posiciones de memoria y definir los tipos de datos
long Timer;
struct time t;
long tempo[5][5];
long tiempos[5][5][3];
int bandas[4];
int f,c,m,n,flor,j,k;
int banda[6];
int salida,sale;
char ojo;

struct tiempos cuenta;
FILE *uno;          // puntero caracter a archivo del tipo FILE
int x,y, key;
int fila,columna, filavieja, columnavieja;

void main()
{
 // Desarrollo del programa
_setcursortype(_NOCURSOR);
abrir();
//Mostrar los tiempos en la pantalla
```

```
textbackground(0);
clrscr();
textbackground(12); //Este es el color del fondo
textcolor(15); // Este es el color de la letra
gotoxy(1,2); clreol();
gotoxy(1,2); cprintf("                    SISTEMA AUTOMATICO     ");
gotoxy(1,3); clreol();
gotoxy(1,3); cprintf("                    CONTROLADOR DE CARGAS      ");

textbackground(7);
textcolor(8);

for (f=1;f<=14;f++)
 {
        for (c=1; c<=68;c++)
         {
         gotoxy(5+c,5+f );
         cprintf(" ");
         if ((f==1) || (f==14))
                 {
                  gotoxy(5+c,5+f); cprintf("Í");
                 }
         if (f==12)
                 {
                  gotoxy( 5 + c,5 + f); cprintf("Ä");
                 }
        }
```

```
        gotoxy(5 , 5 + f); cprintf("º");
        gotoxy(74 , 5 + f); cprintf("º");
 }

gotoxy(5,6); cprintf("É");
gotoxy(74,6); cprintf("¹");
gotoxy(5,19); cprintf("È");
gotoxy(74,19); cprintf("¼");
gotoxy(5,17); cprintf("º");
gotoxy(74,17); cprintf("º");

for(n= 1; n<=8; n++)
 {
        gotoxy(65 + n,4); cprintf("Í");
 }

gotoxy(74,4); cprintf("»");
gotoxy(74,5); cprintf("º");
gotoxy(65,4); cprintf("É");
gotoxy(65,5); cprintf("º");
gotoxy(65,6); cprintf("È");

for(m=1; m<=4;m++)
 {
        flor = m;
        colorear();
        gotoxy(10, 8 + m*2 );
```

```
        if (fmod(m, 2)==0)
         cprintf(" apagado");
         else
          cprintf(" encendido");

        for (k= 1;k<=4; k++)
         convierte2();
 }
fclose(uno);
gotoxy(8,18); cprintf("Estado actual");
gotoxy(10,21); cprintf(" <M> Modificar    <G> Grabar Horario          <X> Salir  ");


do
 {
        do
        {
         banda[m] = 0; salida = 0;
         for(m= 1; m<=4;m++)
          {banda[m] = 0;
                for(j=1;j<=4;++j)
                 {gettime(&t);
                  Timer = t.ti_hour+ t.ti_min+t.ti_sec;
                  if (Timer> tempo[m][j])
                        banda[m] =(banda[m]) ^ 1;
                 }
                if (banda[m]== 0)
```

```
                {
                 textcolor(15);
                 textbackground(9);
                }
               else
                {
                 textcolor(15);
                 textbackground(12);
                }
               gotoxy( 13 + 13*m,7);
               cprintf(" O ");
               gotoxy( 13 + 13*m,8);
               cprintf(" %d ",m);
               gotoxy( 14 + 13* m, 18);
               cprintf(" %d ", banda[m]);
          }

          salida = banda[4] *128 + banda[3] *32 + banda[2]*8 + banda[1]*2;
          salida = -1 *(salida+1);
          outportb(888,salida);

          sale = 1;
          horaActualizada();
          sleep(1);
         }while (bioskey(1) == 0);
key = bioskey(0);
ojo = toupper(ojo = key);
```

```
 switch (ojo)
          {
          case 'M': cambiarHoras(); break;
          case 'G': grabar(); break;
          }
 } while (ojo != 'X');
}
 /****Fin del Programa Principal****/


void abrir(void)
 {
 uno = fopen("final.dat","rt");
 for (k = 1; k<=4; k++)
          {
          fscanf(uno,"%d                                %d                                %d
%d\n",&tempo[1][k],&tempo[2][k],&tempo[3][k],&tempo[4][k]);
          }
 fclose(uno);
 }

void horaActualizada(void)
 {
 textcolor(7);
 textbackground(1);
 x = wherex(); y = wherey();
```

```
 gettime(&t);
 gotoxy(66,5); cprintf("%2d:%02d:%02d",t.ti_hour,t.ti_min,t.ti_sec);
 gotoxy(x,y); colorear();
 }

void cambiarHoras(void)
 {
 fila = 1; columna = 1;
 columnavieja = 4; filavieja = 4;
 for(columna = 1; columna<= 4; columna++)
        for(fila = 1; fila<=4; fila++
}
```

BIBLIOGRAFIA

1. **[BOND73]** BOND, JOHH. Interfacing Peripheral Devices with Computers, Tercera Edición 1973. Pág 8-110.
2. **[BOYL94]** BOYLESTAD, LOUIS NASHELSKY, Electrónica Teoría de Circuitos. Optoiasladores. Quinta Edición 1994 Pág 861-864.
3 **[CEVA91]** CEVALLOS SIERRA. Enciclopedia del Lenguaje C. Funciones para la Consola y los Puertos de E/S. Segunda Edición 1991. Pág 374 -387
4 **[FLET89]** WILLIAN I. FLETCHER. Introduction to Multi Input System Controller Design. Segunda Edición 1989. Pág 441 -529.
5 **[INTE96]** INTERNET. INFORMATION OF MOTOROLA. 1996.
http: www.motorola.com
6 **[INTE96]** INTERNET. DIGITAL SIGNAL PROCESING. University of Holland, Electronics Engineers.1996
http: www.eeb.ele.tue.nl index.html
7 **[INTE96]** INTERNET. Control by Port. 1996
http: www.allied.avnet.com
8 **[INTE96]** INTERNET. BERKELEY DESIGN TECHNOLOGY, INC. 1996
ftp: www.dbti.com
9 **[INTE96]** INTERNET. ELECTRONICS ENGINEERS TOOLBOX. 1996
http: www.cera2.com ebox.htm
10 **[INTE96]** INTERNET. SOFTWARE IMPLEMENTATION FOR CONTROL.1996
http: www.aris.com boxtop
11 **[INTE96]** INTERNET. EXPOSICION VIRTUAL DE TECNOLOGIA TECHEXPO. 1996
http: www.intusoft.com

12 **[JACO89]** JACOB. Applications And Design With Analog Integrated Circuits. Difference Amplifier. Segunda Edición 1989. Pág 54.

13 **[KELL89]** KELLY, IRA POHL. Lenguaje C Introducción a la Programación. Segunda Edición 1989. Pág 11- 146.

14 **[MALV86]** ALBERT PAUL MALVINO, Ph,Dr. Principios de Electrónica. Optoacopladores. Tercera Edición 1986. Pág 115- 119.

17 **[SCHI90]** HERBERT SCHILDT. Programación en Lenguaje C. Primera Edición 1990. Pág 7-194.

18 **[SCHI91]** DONALD L SCHILLING, CHARLES BELOVE. Circuitos Electrónicos Discretos e Integrados. Convertidor A/D Controlado por Contador. Segunda Edición 1991. Pág 720-725.

19 **[SCHN80]** ISSAC SCHNADOWER BARAN. Circuitos Electrónicos Digitales. Señales y Circuitos Lógicos. Segunda Edición 1980. Pág 117-177.

20 **[SCRM80]** A.A. Adem, J.L.Brookmire, J.H.Galloway, F.W.Gutzwiller, E.K.Howell,D.V.Jones,H.Kaufmaun,H.R.Lowry,N.W.Mapham,J.E.Mungenast, R.M.Muth, T.A. Penkalki, G.E. Snyder, T.P.Sylvan, E.E.Von Zastrow. Manual SCR General Electric. SRC MANUAL. The Triac. Segunda Edición 1980. Pág 181-254.

21 **[SEMI87]** ECG Semiconductors Master Replacement Guide, Optoisolators, Cuarta Edición 1987. Pág 1-135.

[SEMI87] ECG Semiconductors Master Replacement Guide, Transistor, Cuarta Edición 1987. Pág 1-37 - 1-60.

[SEMI87] ECG Semiconductors Master Replacement Guide, Triacs, Cuarta Edición 1987. Pág 1-113.

22 **[SMIT74]** R.E. ZIEMMER, W.H. TRANTER. Comunication Principles. Teoría del Muestreo. Tercera Edición 1974. Pág 98-03.

23 **[TIEN 71]** TIEN, PAUL S. Techniques Computer Design. Segunda Edición 1971. Pág 49-56.