15
2ej.

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO E IMPLEMENTACION DE UN DISPOSITIVO COMERCIAL DE PROTECCION DE SOFTWARE

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A N :

MA. ISABEL CABRAL LOPEZ
JESUS HUMBERTO FIGUEROA CABALLERO
HUGO CABALLERO RAMIREZ

DIRECTOR DE TESIS:
Ing. Rodolfo Heredia Vázquez

1992

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

INTRODUCCION

I EL PROBLEMA DE LA PIRATERIA DE SOFTWARE.

La piratería de software puede definirse, simplemente, como la copia no autorizada de programas. Al igual que en la industria del disco y el cassette, con la cual tiene muchos problemas en común, la piratería se produce de diferentes formas y a distintos niveles. Del mismo modo que es ilegal realizar copias de cassettes de música propiedad de otras personas, efectuar copias de programas o en mayor número de las permitidas por la licencia de uso (inclusive aunque sean sin fines lucrativos), representa incurrir en piratería de software. Hay formas de copiar dentro de la ley: la mayoría de los proveedores de software para microcomputadoras autorizan hacer una copia para fines de respaldo. Puede darse el caso de que alguna licencia de uso autorice múltiples copias. Al nivel mas bajo, se comete piratería cada vez que el propietario de una computadora personal copia un programa que le ha prestado un amigo.

Pueden distinguirse tres tipos de piratería de software: *la piratería comercial*, que propician ciertos vendedores de computadoras como atractivo para sus clientes, por ejemplo, al ofrecer cierto procesador de palabras en la compra de x modelo de computadora, con el fin de incrementar

su valor efectivo; *la piratería empresarial*, que propician algunas compañías que adquieren un solo paquete de software y luego distribuyen copias del programa para ser usadas simultáneamente en todos sus departamentos; y *la piratería estudiantil*, la que prolifera debido al alto costo de los programas y la reducida capacidad económica de los estudiantes.

Desgraciadamente para los distribuidores, esta no sólo es difícil de evitar, sino también de detectar y procesar. Los propios fabricantes de software han hecho estimaciones (que en algunos casos son meras especulaciones) de las copias ilegales que existen, y se cree que circulan un promedio de mil copias pirateadas por cada programa original. Además, de los programas más populares pudiera ser que se utilicen hasta cinco mil copias ilegales por cada original. Aunque se pueda argüir que algunos fabricantes están en condiciones de afrontar estas pérdidas, debe recordarse que muchas personas obtienen su medio de vida gracias a las regalías de los programas.

Se ha producido una amplia controversia sobre el hecho de que los distribuidores ofrezcan software con los sistemas de préstamo "pruebe antes de comprar", puesto que éstos facilitan la copia de programas. Existen casos incluso de distribuidores que reproducen programas a gran escala y los venden a otros (con menos riesgo del que parece si éstos son de otro país). Estos productos son equivalentes, por lo tanto, a las copias de contrabando de cassettes de gran popularidad.

La piratería puede ser aún más sofisticada y más difícil de precisar. Por ejemplo, alguien toma un programa que ya existe, efectúa algunas modificaciones en él y lo pone a la venta como propio.

Como es evidente, mediante la piratería se viola el derecho de autor. Esta práctica ilegal no sólo provoca el malestar de los programadores y de las casas fabricantes de software, quienes al ver sus utilidades mermadas pueden reaccionar aumentando el precio de venta de los paquetes de software. Sin embargo, algunos distribuidores opinan que si se vendieran sus productos a un precio suficientemente bajo, la gente tendrá menos interés en copiarlos.

En fechas recientes se ha recrudecido la lucha contra los piratas de software. En México por ejemplo, la Asociación Nacional de la Industria de Programas para Computadora, A.C. (ANIPCO), ha emprendido una serie de acciones entre las que se encuentra una campaña publicitaria contra la piratería. La asociación afirma que hay varios casos penales que serán llevados a los tribunales e incluso se prevé que algunos piratas terminen en la cárcel. Se estima que en su primer año la campaña costará entre 500 y 1,000 millones de pesos. Entre los principales patrocinadores se encuentran IBM, Hewlett-Packard, Unysis, Microsoft y Lotus. En Estados Unidos existe una asociación similar llamada Business Software Association.

En programas sofisticados, un manual bien editado o una presentación atractiva aporta un cierto grado de protección. El "registro del usuario" es otro método con el que se protege software: hasta que no se envía la tarjeta incluida en el manual del usuario, no se ofrece ninguna ayuda o asistencia por vía telefónica.

La práctica malsana de la piratería ha obligado a algunos programadores a desarrollar "esquemas de protección", los cuales dificultan la tarea de copiar un disco, llegando algunos inclusive a ocasionar graves problemas en el funcionamiento de la computadora o a dañar

la información que contiene el disco -particularmente el disco fijo-, provocando así la pérdida de información importante. Esto sucede cuando se ejecuta una copia no autorizada del software original, así haya sido hecha para guardar éste en un lugar seguro.

Este tipo de daño, como se ve, es causado con la única intención de proteger al software de copias ilegales. Sin embargo en los últimos años se ha observado una proliferación de pequeños programas llamados *virus*, generados con la intención de molestar al usuario o causar grandes perjuicios a la información que tiene almacenada en sus discos. No se trata ya de algo diseñado sólo para proteger el software creado por el programador, sino más bien de un programa creado realmente para perjudicar a los usuarios que copian programas o a quienes adquieren copias ilegales del software en el mercado negro a precios muy bajos. La piratería sólo sirve para que los "terroristas de la informática" encuentren un excelente "caldo de cultivo" para diseminar sus virus.

Se ha creado así una verdadera histeria entre los usuarios, la cual indudablemente reducirá el número de copias piratas debido al temor a adquirir virus. Este temor ha servido para concientizar a los usuarios, a fin de que utilicen discos de programas originales y no se fíen de las copias que se les ofrecen.

No obstante todos los esfuerzos que se han hecho y los problemas que puede acarrear la copia ilegal de programas, la que se ha extendido mucho y es muy difícil de detectar, acabar con la práctica de la piratería no es sencillo.

II OBJETIVO DE LA TESIS.

La lucha contra la piratería es costosa; como es obvio deducir, no basta con pedirle a la gente que no copie programas. Para ayudar a reducir la piratería, la presente tesis tiene como objetivo principal diseñar un dispositivo denominado Protector de Software que impida la copia ilegal de programas. El dispositivo está destinado a los fabricantes de software para que ellos a su vez lo distribuyan con cada original del programa que vendan.

El único requisito necesario por parte del usuario es que su computadora cuente con al menos un puerto paralelo, ya que el dispositivo está diseñado para conectarse en él. Si el Protector de Software no está presente durante la ejecución del programa, éste no funcionará, generando un mensaje de error. No es necesario que se conecte una impresora, pero si así se desea, ésto puede hacerse en la parte posterior del Protector de Software utilizando un cable convencional de impresora y el funcionamiento de ambos, el dispositivo y la impresora, no se afectará en nada. Como se verá más adelante, el funcionamiento del Protector de Software se basa en una memoria EPROM del tipo 27C16.

El segundo objetivo de la tesis es desarrollar la tecnología que permita producir el Protector de Software. Para ello se diseñará también un Programador de EPROM 27C16 que permita introducir los datos necesarios en dicha memoria. Este Programador también será conectado en el puerto paralelo y se controlará desde la computadora. A diferencia del Protector, para utilizar el Programador debe desconectarse la impresora.

Todo el sistema debe cumplir con los requisitos de confiabilidad, compatibilidad, sencillez y bajo costo, a fin de poder implementar su producción y comercialización en serie con objetivos comerciales.

III CONTENIDO DE LA TESIS.

Hasta aquí se ha planteado la importancia del problema de la piratería de software y el cómo la tesis propuesta ayudará a reducir este problema.

A continuación se presentan los capítulos con objeto de aclarar, definir y respaldar la tesis propuesta.

El capítulo uno tiene por objeto comprender el funcionamiento del puerto paralelo, que actúa como interface entre la computadora y los dispositivos propuestos. Entender el puerto paralelo es la base para poder desarrollar ambos circuitos.

En el capítulo dos se trata el diseño del Programador de EPROM. Primeramente se repasan los conceptos básicos sobre este tipo de memorias para poder comprender su funcionamiento. A continuación se estudia el diseño del circuito y su funcionamiento. Finalmente se desarrolla un programa que controle la operación del Programador.

El capítulo tres contiene el diseño del Protector de Software. Inicialmente se estudia el funcionamiento del circuito desde el punto de vista de sus modos de operación. Posteriormente se diseña el programa que se encarga de la operación del circuito.

El capítulo cuatro muestra la manera en que se efectuaron las pruebas y se implementaron los diseños en

circuitos impresos.

Finalmente, se presentan las Conclusiones donde se hace una estimación de los costos, se analizan los resultados y se realiza una evaluación de los beneficios obtenidos.

En los Anexos se incluye un listado de la interrupción 17H, las hojas de especificaciones de los circuitos integrados utilizados y el manual del usuario del Programador. Por último, se encuentra la Bibliografía.

CAPITULO 1

EL PUERTO PARALELO

1.1 INTRODUCCION.

1.1.1 DEFINICION DE PUERTO.

El término puerto se ha usado con diferentes significados en computación. En algunas ocasiones se refiere al conector en la computadora en el cual se conectan los dispositivos externos o periféricos, generalmente a través de un cable. Así por ejemplo, se tienen el puerto del teclado, el puerto de video, el puerto serie y el puerto paralelo, entre otros.

En un sentido más amplio, la palabra puerto se refiere no sólo al conector, sino también a los circuitos necesarios para establecer dicha conexion y que generalmente se encuentran en una tarjeta de expansión. También se emplea la palabra interface como sinonimo de puerto porque éste funciona como enlace entre la computadora y los periféricos. A algunos de estos puertos, el Sistema Operativo DOS les asigna un nombre. Tal es el caso de COM1 para el primer puerto serie, COM2 para el segundo, etc. En la presente tesis, la palabra *puerto* se refiere al conjunto de circuitos que se requieren para comunicar a la computadora con los periféricos.

También la palabra puerto se usa para designar un número usado por el software para accesar a los dispositivos periféricos: son los llamados puertos de entrada/salida (puertos de E/S o en inglés I/O ports). La mayoría de los circuitos de soporte, es decir, los circuitos que apoyan al microprocesador (por ejemplo el Controlador Programable de Interrupciones 8259), son accesados a través de estos puertos de E/S. Lo que es más, un mismo circuito integrado puede utilizar varios puertos de E/S para diferentes propósitos. Cabe resaltar que las direcciones asociadas con algún dispositivo de E/S no son parte de la memoria principal. En otras palabras, el puerto de E/S 378H, por ejemplo, no tiene nada que ver con la dirección de memoria 00378H. Inclusive, para accesar a un puerto de E/S no se usan las instrucciones de transferencia de datos como MOV o STOS, en cambio, se usan las instrucciones especiales reservadas para acceso a puertos IN y OUT.

Así como cuando accesa a la memoria, la CPU también accesa a estos puertos por medio de su bus de direcciones pero primero envía una señal para indicar a los dispositivos de E/S que se trata de la dirección de un puerto y no de una dirección de memoria. A continuación, coloca la dirección en el bus y el dispositivo direccionado responde. Sin embargo, únicamente las líneas de dirección A0 a A15 son utilizadas para el acceso a puertos dando un rango de 00H a FFFFH (65,535). En todo caso, no debe confundirse la palabra puerto con el término puerto de E/S. Para evitar confusiones, en la presente tesis se utilizará el término *dirección de E/S* para referirse a la dirección que maneja el software para accesar a los circuitos.

1.1.2 MODOS DE TRANSMISION.

Existían dos formas para comunicar a la computadora con

los dispositivos que se diseñaron en la presente tesis: a través del puerto paralelo y a traves del puerto serie. A continuación se darán las principales características de cada uno de los modos de transmisión, analizándose después sus ventajas y desventajas.

Transmisión en Serie. La transmisión en serie consiste en el envío de los bits que componen la información en secuencia, uno tras otro, usando un solo conductor . Algunos bits adicionales se agregan antes y después de cada caracter para el control del flujo de la información.

El puerto serie en las computadoras personales está basado en el estándar RS-232 establecido por la EIA (Asociación de la Industria Electrónica), el cual fija las reglas para el intercambio de datos entre equipos que empleén comunicaciones en serie. Se trata de una comunicación bidireccional y asíncrona muy flexible, ya que está diseñada para una amplia variedad de usos. Para lograr esta flexibilidad es necesario ajustar ciertos parámetros de comunicación para adaptarse a las necesidades particulares del enlace que se desea establecer. Estos parámetros son la velocidad de transmisión, la paridad, el número de bits que componen el caracter y el número de bits de paro.

En las PC el puerto serie se usa como una alternativa para las impresoras y graficadores en paralelo, para algunos dispositivos de entrada como ratones y rastreadores, pero sobre todo, para la comunicación entre computadoras. Los conectores más comúnmente utilizados en la transmisión en serie son los DB-25 y DB-9 y en algunas ocasiones los tipo DIN. Se puede utilizar cable plano o cable convencional redondo con múltiples conductores, pero debido a los bajos voltajes utlilizados, la máxima longitud del cable recomendada es de 15 metros aproximadamente. Si se usa un

cable más largo se pueden perder datos debido a la caida de voltaje resultante del incremento de la resistencia eléctrica del cable.

Los puertos serie de la PC se basan en dispositivos UART (Transmisor Receptor Asíncrono Universal), la mayoría de los cuales siguen un estándar que permite transferencias de hasta 115,200 bits por segundo (bps). No obstante, las velocidades utilizadas en comunicaciones van de los 110 bps (en equipos obsoletos) hasta 19,200 bps. No se utilizan velocidades más altas porque se requieren líneas especiales aisladas de interferencias exteriores, así como equipo caro de modulación y transmisión.

Normalmente, cada caracter enviado se compone de 10 bits (el byto de dato más los bits adicionales). Así, una forma aproximada de conocer el número de caracteres enviados por segundo (cps), es dividir la velocidad en bps entre 10. Por ejemplo en una transmisión a 19,200 bps se estarían enviando 1,920 cps.

Transmisión en Paralelo. Por el contrario, en la trasmisión en paralelo la información se envía utilizando un conductor para cada uno de los bits que componen un caracter. Todos los bits son enviados simultáneamente por lo que al menos ocho conductores son necesarios. Adicionalmente, se utilizan algunos conductores extras para controlar la transmisión. En las computadoras personales el puerto paralelo ha sido dedicado casi exclusivamente para la conexión de la impresora, por lo cual en seguida se describe en detalle la forma en que funciona.

La computadora básicamente tiene dos cosas que enviar a la impresora: los datos a imprimir y unas cuantas señales de control para inicializarla, indicarle cuando se tiene listo

un dato para enviarle, etc. Es importante notar que también es posible controlar a la impresora a través de las líneas dedicadas al envío de los datos a imprimir. Para ésto basta con que la impresora reconozca cuando se trata de una señal de control y cuando se trata de un dato a imprimir. Existen dos formas de enviar órdenes a la impresora a través de las líneas de datos: por medio de códigos de control y de secuencias de escape. Ambas se explicarán más adelante.

Se considera que el puerto paralelo es unidireccional porque los datos son enviados en un solo sentido, aunque estrictamente no lo es porque la impresora, por su parte, utiliza algunas líneas para informarle a la computadora cuando está ocupada, cuando ha recibido un dato, cuando se ha producido un error, etc. En las computadoras PS/2 el puerto paralelo ha sido rediseñado para permitir comunicación bidireccional.

Ventajas de la Transmisión en Paralelo. Internamente las computadoras mueven los datos en paralelo, de modo que los circuitos necesarios para construir un puerto paralelo son relativamente simples. Gracias a esta simplicidad el puerto paralelo es una de las conexiones más fáciles de usar ya que se utiliza un cable estándar y no es necesario ajustar ningún parámetro de comunicación, lo cual no se cumple para la comunicación en serie.

Debido a que se cuenta con ocho conductores, los datos se pueden mover potencialmente ocho veces más rápido que utilizando un solo conductor. Considerando el tiempo mínimo requerido por las señales de control, el puerto paralelo puede transmitir a una velocidad aproximada de 100,000 bytes en un segundo.

En el caso de la comunicación con la impresora, esta velocidad disminuye notablemente porque son necesarios una serie de procedimientos, tanto de parte de la computadora como de la impresora, que retrasan el proceso. Por ejemplo, cada vez que se transmite un caracter, es necesario cargarlo en las líneas de datos y esperar la señal de recibido a través de una rutina BIOS. La velocidad final obtenida es de aproximadamente 1,000 cps, muy aproximada a la obtenida en la interface serie.

Limitaciones de la Transmisión en Paralelo. El cable utilizado para la transmisión en paralelo es más caro que el usado en la transmisión serie por poseer un mayor número de conductores, sin embargo, el mayor problema que se presenta con la transmisión en paralelo es conocido como crosstalk: el uso de múltiples conductores en un cable y la variedad de señales que transitan por él, tienden a producir interferencias entre ellas. Este problema en las líneas telefónicas es el que ocasiona que la plática en una conversación se "cruce" con otra. Mientras más largo es el cable mayor es la interferencia y de ahí es que los fabricantes recomienden un máximo de 3 metros en la longitud del cable para prevenir problemas. La sensitividad al problema de crosstalk varía entre computadoras e impresoras. Algunos sistemas trabajarán bien con cables de hasta 15 metros de largo, pero para evitar problemas, lo más recomendable es mantener la impresora lo más cerca posible de la computadora.

1.1.3 JUSTIFICACION DEL EMPLEO DEL PUERTO PARALELO.

¿Por qué utilizar el puerto paralelo en la presente tesis para la conexión del Programador de EPROM y del Protector de Software? El elemento principal del Programador y del Protector de Software es una memoria EPROM 27C16. Como

se verá más adelante, esta memoria poseé once líneas para formar las direcciones y ocho líneas que componen los datos a grabarse o a leerse. La manera más simple y lógica de proveer a la EPROM con todos estos bits es en paralelo.

Utilizar el puerto paralelo permitirá utilizar el menor número de circuitos integrados y por lo tanto, minimizar el costo de fabricación de ambos dispositivos. La limitante en la longitud del cable paralelo no es un factor que influya en estos diseños por encontrarse los dispositivos muy cerca de la computadora.

1.1.4 ASIGNACION DE DIRECCIONES BASE DE ENTRADA/SALIDA.

El sistema operativo DOS permite conectar hasta 3 puertos paralelos a una computadora a la vez. Para distinguirlos, cada puerto tiene asociada una dirección de E/S. Estas direcciones base pueden ser 3BCH (956 decimal), 378H (888 decimal) y/o 278H (632 decimal) y deben ser únicas para cada puerto. Se pueden tener tres puertos en una computadora sólo si uno de ellos está en la tarjeta de video MDA o su equivalente. De otra manera, DOS limita a la computadora a dos puertos paralelos.

La dirección 03BCH está reservada para el puerto paralelo instalado en la tarjeta MDA. Como las computadoras PS/2 no pueden usar esta tarjeta, ya que tienen su propia tarjeta de video integrada, la dirección 3BCH está asignada como estándar en esas computadoras. Las otras dos direcciones están disponibles para puertos adicionales.

Las computadoras PC, XT y AT equipadas con puertos paralelos integrados, normalmente no asignan la dirección 3BCH a su puerto por si se llega a usar una tarjeta MDA. En cambio, normalmente proveen una dirección base de 378H a su

puerto y algún medio para cambiar la dirección, típicamente jumpers o interruptores tipo DIP.

Cuando la computadora se enciende o se reinicializa, el DOS busca puertos paralelos, primero en la dirección de E/S 3BCH, luego en 378H y por último en 278H. Para cada puerto que va encontrando se almacena su dirección de E/S en el Area de Datos BIOS a partir de la dirección absoluta 00408H (1032 decimal). La dirección de E/S del primer puerto encontrado (ya sea 3BCH, 378H, 278H) queda en las localidades de memoria 00408H y 00409H. La dirección de E/S del segundo puerto (si lo hay) queda en las localidades 0040AH y 0040BH y la del tercer puerto (si lo hay) queda en las localidades 0040CH y 0040DH. Si no se encuentran uno o más puertos, las localidades de memoria correspondientes quedan en ceros. De esta forma la o las direcciones de E/S de el o los puertos quedan disponibles para cualquier programa que necesite accesarlos.

1.1.5 NOMBRES DE DISPOSITIVOS.

Como ya se mencionó, el DOS asigna nombres a los dispositivos de E/S. Para el caso de los tres puertos paralelos que soporta el sistema, estos nombres de dispositivos son LPT1, LPT2 Y LPT3 (abreviación de Line PrinTer). También existe el nombre de dispositivo PRN que es sinónimo de LPT1. Estos nombres lógicos no necesariamente tienen que corresponder con un conjunto dado de direcciones de E/S.

Si sólo se tiene un puerto paralelo en la computadora, DOS siempre lo nombra LPT1 sin importar la dirección de E/S que use. Si se tienen dos puertos paralelos, DOS le asigna el nombre de LPT1 al puerto que tiene la dirección más alta. Por ejemplo, si los puertos usan 378H y 278H, el puerto en

378H se vuelve LPT1 y el que está en 278H se vuelve LPT2. Si se tienen tres puertos paralelos instalados, el puerto 3BCH se vuelve LPT1, el 378H se vuelve LPT2 y el 278H se vuelve LPT3. Así, si se tiene una PC con tarjeta MDA o una PS/2 con puerto paralelo incluido, ese puerto será LPT1 o PRN.

Como consecuencia de este esquema de asignación, se garantiza que haya un dispositivo LPT1 (o PRN) en una computadora, independientemente de la dirección de E/S asignada a él (suponiendo que hay al menos un puerto paralelo conectado, por supuesto). Si se tienen dos puertos con la misma dirección de E/S, el DOS les asignará el mismo nombre de dispositivo y es probable que ninguno trabajará. Tampoco debe excederse el máximo de tres puertos paralelos, que es el límite impuesto por el DOS (si se tienen más de tres puertos deberán deshabilitarse los sobrantes).

1.1.6 CONECTORES Y SEÑALES.

Conectores. La razón de éxito del puerto paralelo es su simplicidad. La compañía que hizo popular la interface en paralelo es Centronics. Aunque ninguna organización de estándares fijó las normas para la interface en paralelo, ésta se ha vuelto la forma estándar de conectar una PC a una impresora. Centronics publicó los detalles técnicos de su puerto paralelo al público con el objetivo de vender más impresoras. En efecto, Centronics vendió más impresoras hasta que Epson introdujo su impresora compatible con Centronics. Epson cambió el conector usado en la impresora. El conector Centronics usado para conectar el cable a la impresora era un conector en eje. Este tipo de conector es escencialmente parte de una tarjeta de circuito impreso que se prolonga hacia afuera del chasís de la impresora. El cable que se conectaba no tenía forma de asegurarse y se zafaba con facilidad llegando a causar corto circuito.

Conector DB-25 (Computadora) Pin Número	Direc.	Conector Centronics (Impresora) Pin Número	Función
1	►	1	-STROBE
2	►	2	DATA0
3	►	3	DATA1
4	►	4	DATA2
5	►	5	DATA3
6	►	6	DATA4
7	►	7	DATA5
8	►	8	DATA6
9	►	9	DATA7
10	◄	10	-ACK
11	◄	11	BUSY
12	◄	12	PE
13	◄	13	SLCT
14	►	14	-AUTO FDXT
15	◄	32	-ERROR
16	►	31	-INIT
17	►	36	-SLCT IN
18-25		16,19-30,33	GND

FIG. 1.1
CONEXIONES DEL CABLE PARALELO

Epson popularizó un conector diferente que permite asegurar el cable impidiendo que se zafe. Este es un conector hembra con treinta y seis cables. En 1981 IBM compró impresoras Epson y las vendio con algunos cambios en la ROM. Este hecho fue significativo porque Epson (no Centronics) se volvió el estándar de IBM y como se sabe, por lo general, los estándares de IBM se vuelven estándares mundiales. Posteriormente la versión del puerto paralelo de Epson se adoptó por la industria y actualmente todas las impresoras son compatibles con Epson. Así, el conector utilizado actualmente fue introducido por Epson pero es conocido por todos como conector Centronics, quien dictó el estándar para la interface en paralelo.

Respecto al conector utilizado en la computadora, previo a la introducción de las PC, las computadoras tenían conectores de treinta y seis terminales (u otros no estandarizados) para conectarse con otro del mismo tamaño en la impresora. Para ahorrar espacio en la parte trasera de la PC, IBM usó un conector DB-25S que, como su nombre lo indica, tiene forma de D y veinticinco terminales. La S significa hembra (del inglés socket) a diferencia del usado en la transmisión serie, que usa un conector macho DB-25P (P del término inglés plug). El cable usado para conectar la computadora con la impresora debe hacer la conversión entre el conector DB-25S (veinticinco terminales) y el Centronics (treinta y seis terminales).

Señales. Se puede dividir a las señales usadas en la conexión en paralelo en tres grupos: las señales de datos, las señales de control, que van de la computadora hacia la impresora, y las señales de estado, que van de la impresora hacia la computadora. A continuación se describe cada una de estas señales y entre paréntesis se da el nombre abreviado, si lo hay, utilizado en los diagramas.

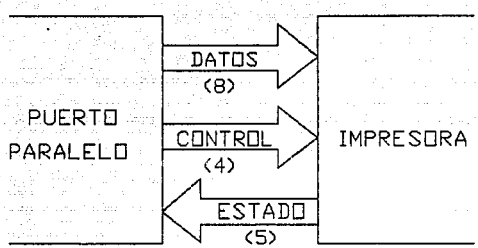


FIG. 1.2
GRUPOS DE SEÑALES

- Líneas de Datos (DATA0 a DATA7).

La información que se va a imprimir viaja a través de ocho líneas de datos, una para cada bit del byte de código ASCII. Estas señales se encuentran a niveles de voltaje TTL estándar: un "alto" o cinco volts indicando un uno lógico; un "bajo" o cero volts indicando un cero lógico.

Algunas de las instrucciones más necesarias son tan comunes que están incorporadas en el conjunto de caracteres ASCII, donde tienen asignados valores específicos. Por ejemplo, para retroceder un caracter, la computadora manda un byte con el ASCII 08. Al recibir este caracter la impresora moverá la cabeza de impresión un lugar a la izquierda en lugar de imprimir. Al grupo de estos valores especiales ASCII se les llama caracteres o códigos de control y representan una orden para la impresora y no un caracter a imprimir.

El número de códigos de control es limitado y no es suficiente para llevar a cabo el gran número de funciones que la impresora puede realizar. Para poder ejercer control sobre todas estas funciones a través de las líneas de datos, se utilizan además cadenas de caracteres conocidas como secuencias de escape. Una secuencia de escape es simplemente una serie de caracteres ASCII que comienzan con el código ASCII especial 27 decimal (1BH). Este código es llamado escape y se abrevia ESC. Lo que hace este código es llamar la atención de la impresora y avisarle que el o los caracteres que le siguen deben interpretarse como comandos en lugar de imprimirse. El Instituto Nacional Americano de Estándares (ANSI por sus siglas en inglés) ha definido un conjunto de secuencias de escape para controlar impresoras pero, como a menudo sucede en computación, este conjunto no es tan estándar. Casi todos los fabricantes de impresoras han ampliado, modificado o completamente ignorado el estándar para ajustarse a las necesidades particulares de su propia impresora.

-Strobe.

Después de colocar los ocho bits en las líneas de datos es necesario indicarle a la impresora que el caracter está listo para imprimirse. De otra manera, como los bits están cambiando contínuamente, la impresora no sabría cuando se tienen los ocho bits simultáneamente con el valor correcto. Tampoco habría forma de saber cuando se quiere imprimir un caracter dos o más veces. La señal -Strobe le indica a la impresora que el dato está listo y lo puede imprimir.

-Strobe es una señal que funciona con lógica negativa, esto es, normalmente es alta y cuando un byte está listo para transmitirse, baja. La sincronización entre las líneas de datos y -Strobe es crítica. Todas las líneas de datos

deben estar en su valor apropiado antes de que -Strobe se baje, de modo que los circuitos en la impresora tengan tiempo de asumir sus valores. Es necesario medio microsegundo. La señal -Strobe permanece baja otro medio microsegundo para que la impresora se de cuenta que está ahí y por último, los datos deben esperar otro medio microsegundo después de que -Strobe termine. Este traslape de señales ayuda a evitar errores. Así, el proceso datos-Strobe-datos requiere un mínimo de uno y medio microsegundos por caracter.

- Busy.

La velocidad de envío de datos es muy grande comparada con la velocidad a la que se imprimen los caracteres y por tanto, la impresora necesita una forma de indicarle a la computadora que está ocupada imprimiendo un caracter. La señal Busy (ocupado) realiza esta tarea. Tan pronto como la impresora recibe la señal -Strobe y comienza el proceso de impresión, pone la señal Busy en un estado lógico alto. La señal permanece en este estado hasta que la impresora pueda recibir el siguente byte de datos. Esta condición de ocupada puede prolongarse mientras la impresora no pueda aceptar otro caracter para imprimir por cualquier causa. Por ejemplo, el buffer puede estar lleno o la cinta pudo haberse atorado.

- -Acknowledge (-Ack).

La línea -Acknowledge (reconocimiento) lleva una señal de la impresora a la computadora indicándole que el caracter previamente enviado ha sido recibido correctamente, ha sido impreso y que está lista para el siguiente caracter. Al igual que -Strobe, -Acknowledge está normalmente en estado lógico alto y baja para indicar que el caracter se recibió.

Típicamente este pulso dura alrededor de doce microsegundos.

Resumiendo, los bits que componen el dato son colocados en las líneas correspondientes, -Strobe le avisa a la impresora que el dato está listo y se le envía a la impresora, ésta le indica si puede recibirlo con la señal Busy y, una vez que se ha impreso el dato, le comunica a la computadora que se recibió usando -Acknowledge. La figura 1.3 ilustra este proceso.

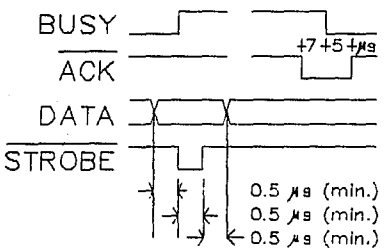


FIG. 1.3
DIAGRAMA DE TIEMPOS

Hasta aquí se han descrito las señales suficientes para establecer una comunicación en paralelo. Bastaría con estas señales para lograr la transferencia de información. No obstante, la interface en paralelo utiliza algunas señales adicionales para controlar a la impresora y, a su vez, la impresora utiliza otras para indicar a la computadora en que estado se encuentra.

Señales de Control.

- -Init.

La impresora siempre comienza en cierta condición al encenderse, con su tipo de letra, tamaño y modos predeterminados por el fabricante. Por medio de comandos, la computadora puede cambiar estos valores para ajustarlos a las necesidades particulares del usuario. Hay dos formas de regresar a los valores de la condición inicial. Una es apagando y volviendo a encender la impresora. La otra es a través de la señal -Init (inicializar). Esta señal se encuentra normalmente en un estado lógico alto y al bajarla provoca que la impresora se inicialice pues funciona con lógica negativa.

- -Select In (-Slct In).

-Select In usa también lógica negativa: cuando es baja la impresora aceptará datos, cuando es alta, no. Es el equivalente en señales al botón "en línea" que se encuentra en el panel de control de la impresora, sólo que es activada por la computadora y no por el usuario.

- -Auto Feed XT (-Auto FDXT).

Hay dos códigos de control que a menudo causan confusión: CR o Retorno de Carro y LF o Alimentación de Línea. Estrictamente hablando, CR sólo debe regresar la cabeza de impresión al margen izquierdo de la línea que se está imprimiendo y LF debe avanzar a la siguiente línea. Juntos producen el efecto normalmente deseado de bajar al extremo izquierdo de la siguiente línea cuando se ha terminado de imprimir un renglón. No obstante, algunas impresoras, al recibir el código CR, realizarán también la

acción de LF. La mayoría de las impresoras proveen un interruptor tipo DIP para determinar cómo deben reaccionar al código CR. La señal -Auto FDXT permite que la computadora controle también esta característica trabajando con lógica negativa. Si es baja, la impresora automaticamente ejecuta una alimentación de línea cuando detecta un código CR. Si es alta, se requerirá un código LF para avanzar a la siguiente línea.

Señales de Estado de la Impresora.

- Select (Slct).

Con esta línea se le indica a la computadora que la impresora está lista para recibir datos o, en otras palabras, que la impresora está "en línea". Esta señal actúa como la luz de "en línea" en el panel de la impresora, sólo que la detecta la computadora en lugar de ser visible al usuario. Si Select es alta la impresora está en línea; si es baja la computadora no enviará datos.

- Paper Empty (PE).

Está señal se encuentra normalmente en un estado lógico bajo. Cuando el papel se agota, Paper Empty pasa a un estado alto para informárselo a la computadora. De la misma manera, una luz en el panel frontal de la impresora y un tono audible le informan al usuario.

- -Error.

Esta señal reporta a la computadora todos los demás posibles errores (aparte de que se termine el papel) sin indicar la causa exacta del fallo. De otra manera se necesitaría una línea para cada error que se pudiera

producir. -Error también funciona con lógica negativa. es decir, normalmente es alta y al bajar indica una anomalía.

Señales Opcionales. Cuando se termina el papel, muchas impresoras, además de poner Paper Empty en alto, activan también Busy. Cuando se produce algún otro error, muchas impresoras ponen no sólo a -Error sino también a Busy y a Select en sus estados de alerta. Algunas de las señales no son totalmente indispensables ya que la interface en paralelo podría operar sin ellas. Cuando se tienen problemas en la comunicación a veces es mejor desconectarlas. Las señales que van de la impresora a la computadora que se pueden desconectar son Paper Empty, -Error y Select. Las dos primeras porque, como ya se dijo, activan a Busy y Select generalmente es ignorada por los programas. En cuanto a las señales que envía la computadora, -Select In tampoco es usada comúnmente por los programas y -Auto FDXT puede quitarse porque el mismo efecto se logra con los interruptores DIP ya mencionados.

1.1.7 EL PUERTO BIDIRECCIONAL.

Las computadoras PS/2 mejoraron las funciones del puerto sobre las que proveen los sistemas PC, XT y AT. Usando la capacidad de Selección de Opciones Programables (POS) el programador puede definir la dirección base del puerto paralelo y seleccionar el modo compatible con las PC, XT y AT o el modo extendido. En modo extendido el puerto puede definirse como puerto de entrada, puerto de salida o puerto bidireccional.

1.2 HARDWARE.

1.2.1 FUNCIONES DEL PUERTO PARALELO.

Desde un punto de vista funcional el puerto paralelo puede realizar cinco operaciones con los tres grupos de señales descritos anteriormente:

1.- Enviar datos al exterior.
2.- Leer los bits de las líneas de datos.
3.- Leer las líneas de estado que provienen del exterior.
4.- Enviar las señales de control al exterior.
5.- Leer las señales de control que provienen del exterior.

Obsérvese que las funciones 1 y 4 son de salida desde el punto de vista del puerto (operación conocida como escritura) y que las funciones 2, 3 y 5 son de entrada (también llamada lectura). Por lo general, el exterior se refiere a la impresora, pero el puerto puede conectarse a otros dispositivos como en la presente tesis. Nótese también que las funciones de lectura 3 y 5 permiten leer desde el exterior, lo que no se puede hacer con la función 2 por tratarse de datos. El dato que se puede leer es únicamente el que se ha mandado escribir en las líneas de datos con la función 1 y no uno que provenga del exterior. Esto es así porque el puerto no es bidireccional. Esta función se utiliza para comprobar que el dato que se escribió con la función 1 ha sido correctamente escrito. ¿Cómo se controlan todas estás funciones? De un análisis del diagrama eléctrico del puerto se observa que a un decodificador 74LS155 llegan cinco líneas denominadas 378F, -XIOW, -XIOR, A0 y A1. Estas señales y el decodificador actúan como lógica de control del puerto.

1.2.2 HABILITACION DEL PUERTO.

Como ya se dijo, cada puerto paralelo puede tener una de tres posibles direcciones base: 3BCH, 378H, o 278H. Es importante notar que se está hablando de direcciones *base* para evitar confusiones porque las funciones de cada uno de los puertos se controlan a través de tres direcciones de E/S. La dirección base es la primera de tres direcciones consecutivas con que trabaja un puerto en particular. En otras palabras, el puerto que tenga asignada la dirección base 3BCH será controlado a través de las direcciones de E/S 3BCH, 3BDH y 3BEH; el puerto que tenga la dirección base 378H usará las direcciones de E/S 378H, 379H y 37AH para controlar sus funciones y el puerto que tenga la dirección base 278H se controlará con las direcciones de E/S 278H, 279H y 27AH.

Ya se mencionó también que al momento de cargarse, el DOS determina cuantos puertos paralelos están disponibles en el sistema buscando en 3BCH, en 378H y en 278H y que las direcciones base de los puertos que encuentre disponibles las colocará en el Area de Datos BIOS en las direcciones absolutas 00408H, 0040AH y 0040CH, donde pueden ser consultadas por cualquier programa.

Las direcciones 3BCH, 378H y 278H se forman así:

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
3BCH	0	0	0	0	0	0	1	1	1	0	1	1	1	1	0	0
378H	0	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0
278H	0	0	0	0	0	0	1	0	0	1	1	1	1	0	0	0

Las líneas A2 a A15 están conectadas de manera que al hacer referencia a una de estas direcciones (con las

instrucciones OUT o IN), se pone en baja una señal que habilita al puerto paralelo correspondiente. Se puede decir que esta señal es la encargada de avisarle al puerto que la orden que manda el sistema es para él en particular y no para los otros dos puertos (si los hay). Por ejemplo, si se va a utilizar el puerto 378H, al producirse la combinación 0000 0011 0111 10 en las líneas de direcciones A2 a A15, se pone en baja la señal que habilita a este puerto. Para el caso del puerto 378H esta señal se denomina 378F. Al bajar 378F el puerto 378H "sabe" que el procesador se está refiriendo a él y las acciones que ejecute dependerán de las otras líneas que llegan al decodificador 74LS155.

1.2.3 CONTROL DEL PUERTO PARALELO.

Primeramente se analizará el cicuito integrado (CI) 74LS155 con más detalle ya que es la base del funcionamiento de la interface paralelo. Para facilitar la explicación se utilizará la dirección base 378H como ejemplo, considerando que funciona de igual manera para las otras dos direcciones base; posteriormente se volverá a generalizar. Asimismo se darán los nombres de las señales en inglés, por usarse así en los diagramas.

- Decodificador 74LS155.

El CI 74LS155 contiene dos decodificadores y sus tablas de función se pueden consultar en el apéndice A. En un decodificador sólo una de las salidas es baja a la vez. La salida que sea baja depende de las líneas de selección, A y B en este caso. Por lo tanto existen 2^2=4 líneas de salidas. -Strobe debe ser baja para que funcione el decodificador. Si -Strobe es alta todas las salidas del decodificador son altas. Las entradas C1 y C2 sirven para escoger cual de los

dos decodificadores del circuito integrado trabaja. El decodificador que tiene la entrada C1 funciona cuando esta entrada es alta y el otro decodificador funciona cuando su entrada C2 es baja.

378F C2	$\overline{378F}$ C1	$\overline{XIOW}$ G2	$\overline{XIOR}$ G1	A1 B	A0 A	2Y0	2Y1	2Y2	2Y3	1Y0	1Y1	1Y2	1Y3	
L	H	L	L	L	L	L	H	H	H	L	H	H	H	*
				L	H	H	L	H	H	H	L	H	H	*
				H	L	H	H	L	H	H	H	L	H	*
				H	H	H	H	H	L	H	H	H	L	*
L	H	L	H	L	L	L	H	H	H	H	H	H	H	
				L	H	H	L	H	H	H	H	H	H	
				H	L	H	H	L	H	H	H	H	H	
				H	H	H	H	H	L	H	H	H	H	
L	H	H	L	L	L	H	H	H	H	L	H	H	H	
				L	H	H	H	H	H	H	L	H	H	
				H	L	H	H	H	H	H	H	L	H	
				H	H	H	H	H	H	H	H	H	L	
L	H	H	H	L	L	H	H	H	H	H	H	H	H	*
				L	H	H	H	H	H	H	H	H	H	*
				H	L	H	H	H	H	H	H	H	H	*
				H	H	H	H	H	H	H	H	H	H	*
H	L	L	L	L	L	H	H	H	H	H	H	H	H	
				L	H	H	H	H	H	H	H	H	H	
				H	L	H	H	H	H	H	H	H	H	
				H	H	H	H	H	H	H	H	H	H	
H	L	L	H	L	L	H	H	H	H	H	H	H	H	
				L	H	H	H	H	H	H	H	H	H	
				H	L	H	H	H	H	H	H	H	H	
				H	H	H	H	H	H	H	H	H	H	
H	L	H	L	L	L	H	H	H	H	H	H	H	H	
				L	H	H	H	H	H	H	H	H	H	
				H	L	H	H	H	H	H	H	H	H	
				H	H	H	H	H	H	H	H	H	H	
H	L	H	H	L	L	H	H	H	H	H	H	H	H	
				L	H	H	H	H	H	H	H	H	H	
				H	L	H	H	H	H	H	H	H	H	
				H	H	H	H	H	H	H	H	H	H	

Como se ve en el diagrama eléctrico, la señal 378F va a la entrada C2 y su negado a la entrada C1. Cuando 378F es baja, las dos entradas se cumplen porque son complementarias y entonces ambos decodificadores trabajan. Cuando 378F es alta ninguno de los dos decodificadores internos trabajará.

En la tabla anterior se omitieron los casos en que las entradas C1 y C2 son iguales (ya se vio que son complementarias) y se analizan todas las demás posibles combinaciones. No obstante, se puede simplificar aún más la tabla. Ya se dijo que 378F es la señal que habilita al puerto paralelo cuando es baja; si es alta, toda la mitad inferior de la tabla puede descartarse porque, como se observa, su negada -378F será baja en C1. Estas son las condiciones para que ninguno de los dos decodificadores internos del 74LS155 trabaje y se comprueba que todas las salidas son altas.

Algunas otras combinaciones no pueden producirse en el diseño del puerto paralelo. Véase ahora como se comportan -XIOR y -XIOW.

La señal -XIOR (lectura de E/S) es una señal de salida del Controlador de Bus 8288. Se usa para indicar a los puertos de E/S que el microprocesador ha iniciado un ciclo de lectura de puerto al haber ejecutado una instrucción IN y que los bits A0 a A15 en el bus de direcciones forman una dirección de E/S. El puerto direccionado debe responder poniendo su dato en el bus de datos.

La señal -XIOW (escritura de E/S) también proviene del Controlador de Bus 8288 e indica a los puertos de E/S que el procesador ha encontrado una instrucción OUT y al ejecutarla se ha iniciado un ciclo de escritura de puerto de E/S. Por tanto, los bits A0 a A15 del bus de direcciones forman una

dirección válida de E/S y los bits del bus de datos deben ser escritos en el puerto direccionado.

En otras palabras, la señal -XIOR se activa con la instrucción IN para leer un dato de una dirección de E/S y la señal -XIOW se activa con la instrucción OUT para escribir un dato en una dirección de E/S. Aquí se ve que sólo una de estas señales puede ser baja a la vez y se pueden eliminar de la tabla todos los renglones donde estas dos señales tienen el mismo valor (marcados con un asterisco en la tabla).

Podemos entonces deducir la tabla de función del decodificador 74LS155 como parte del circuito del puerto paralelo. Los dos decodificadores internos trabajan como si se tratara de un solo decodificador con tres entradas (A, B y las dos G que forman una sola) y ocho salidas. La señal 378F funciona como habilitador.

378F C2	$\overline{378F}$ C1	$\overline{XIOW}$ G2	$\overline{XIOR}$ G1	A1 B	A0 A	$\overline{PDW}$ 2Y0		$\overline{PCW}$ 2Y1	2Y2	2Y3	$\overline{PDR}$ 1Y0	$\overline{PST}$ 1Y1	$\overline{PCR}$ 1Y2	1Y3
L	H	L	H	L	L	L	H	H	H		H	H	H	H
				L	H	H	L	H	H		H	H	H	H
				H	L	H	H	L	H		H	H	H	H
				H	H	H	H	H	L		H	H	H	H
L	H	H	L	L	L	H	H	H	H		L	H	H	H
				L	H	H	H	H	H		H	L	H	H
				H	L	H	H	H	H		H	H	L	H
				H	H	H	H	H	H		H	H	H	L

Del diagrama eléctrico y de la tabla anterior se observa que sólo cinco de las ocho posibles salidas están conectadas y son 2Y0, 2Y2, 1Y0, 1Y1 y 1Y2. Cada salida recibe un nombre en inglés y en la tabla se han abreviado como sigue:

-PDW = -PRINT DATA WRITE
-PCW = -PRINT CONTROL WRITE
-PDR = -PRINT DATA READ
-PST = -PRINT STATUS
-PCR = -PRINT CONTROL READ

Cada una de estas señales corresponde a una de las cinco funciones de la interface paralelo. Todas llevan el nombre PRINT porque la función básica del puerto paralelo es enviar datos a imprimir. Las señales que llevan la letra W (WRITE) son de escritura y las que llevan la letra R (READ) son de lectura. Las señales con la palabra DATA controlan las líneas de datos; las que tienen la palabra CONTROL manejan las líneas de control; y las que llevan la palabra STATUS controlan las líneas de estado.

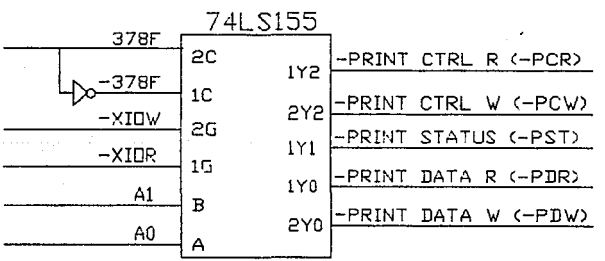


FIG. 1.4
DECODIFICADOR 74LS155 COMO CONTROL DEL PUERTO

Cuando se realiza una escritura, -XIOW (G2) es baja y -XIOR es alta, de manera que las salidas 2Y0, 2Y1, 2Y2 y 2Y3 pueden tomar un valor bajo, una a la vez, dependiendo del valor de las entradas B y A (dado por A1 y A0). De estas salidas sólo 2Y0 y 2Y2 son usadas por el puerto. Por su parte, las salidas 1Y0 a 1Y3 son altas en todos los casos. Es decir, cuando se realiza una escritura con la instrucción OUT, las señales -PRINT DATA W y -PRINT CONTROL W pueden ser activadas, como era de esperarse puesto que son las líneas que controlan las funciones de escritura.

Cuando se efectúa una lectura, -XIOR (G1) es baja y -XIOW es alta y esta vez las salidas 1Y0, 1Y1, 1Y2 Y 1Y3 son las que trabajan, mientras que las salidas 2Y0 a 2Y3 están en estado lógico alto. De las cuatro salidas que se pueden activar sólo 1Y0, 1Y1 y 1Y2 están conectadas y cual de ellas se active dependerá de los valores en las líneas de dirección A0 y A1. En otras palabras, cuando se realiza una lectura con la instrucción IN, las señales -PRINT DATA R, -PRINT STATUS y -PRINT CONTROL R pueden ser activadas ya que son las líneas que controlan las funciones de lectura.

Las líneas A0 y A1 se utilizan para indicar cual de las funciones de escritura o de lectura se llevará a cabo. Recordando que los bits A2 a A15 del bus de direcciones no pueden ser alterados (porque la combinación de ellos es la que habilita el puerto), sólo las entradas A1 y A0, que son las líneas de selección, pueden modificarse. Esto permite cuatro combinaciones para la escritura y cuatro para la lectura que se analizan a continuación:

a) A1-A0-0. Como se observa, la dirección base del puerto 378H cumple esta condición ya que tiene valor cero en los dos bits menos significativos. Si se efectúa un OUT con una de esas direcciones, se está poniendo en bajo

la salida -PRINT DATA W (2Y0). Por el contrario, si se hace un IN con una de esas direcciones se pone en baja la salida -PRINT DATA R (1Y0).

b) A1=0, A0=1. Esta condición se cumple si se incrementa en uno la dirección base (con la dirección 379H). Con un OUT activamos la línea 2Y1, que no está conectada. Con un IN ponemos en baja 1Y1, es decir, -PRINT STATUS.

c) A1=1, A0=0. Incrementando en dos la dirección base se cumplen estos valores (dirección 37AH). Con un OUT se pone en baja la salida -PRINT CONTROL W (2Y2) y con un IN activamos la salida 1Y2, -PRINT CONTROL R.

d) A1=1, A0=1. Las direcciones 3BFH, 37BH ó 27BH no nos sirven pues 2Y3 y 1Y3 no están conectadas.

Volviendo a generalizar para las tres posibles direcciones base del puerto, se tiene la tabla siguiente que muestra las funciones del puerto y como se controlan:

Dirección			Operación	Sali-da	Señal que se activa en bajo
3BCH	378H	278H	OUT	2Y0	-PRINT DATA W
3BEH	37AH	27AH	OUT	2Y2	-PRINT CONTROL W
3BCH	378H	278H	IN	1Y0	-PRINT DATA R
3BDH	379H	279H	IN	1Y1	-PRINT STATUS
3BEH	37AH	27AH	IN	1Y2	-PRINT CONTROL R

Como se ve, de acuerdo a la dirección y a la operación utilizada, sólo una de las señales de salida es baja a la vez. De esta forma se logra realizar una de las cinco funciones del puerto paralelo.

Resumiendo, las señales 378F o 3BCF o 278F habilitan al puerto correspondiente, si hay más de uno. La dirección de E/S utilizada determinará, a través de las líneas A0 y A1, sobre cual de los grupos de señales se llevará a cabo la operación:

- Dirección Base. Si se utiliza esta dirección se trabajará con las señales de datos.

- Dirección Base + 1. Con esta direccion se manejan las señales de estado de la impresora.

- Dirección Base + 2. Esta dirección permite manipular las señales de control.

Las instrucciones OUT e IN, por medio de las señales -XIOW y -XIOR, determinan si se escribe en el puerto o se lee de él.

1.2.2 SEÑALES INTERNAS.

Lo explicado hasta aquí es la forma en que se controla el puerto paralelo. El decodificador 74LS155 y las señales que le llegan son las encargadas de ordenar al puerto lo que debe hacer. Las señales que salen del decodificador llevarán a cabo esa tarea. A continuación se estudiará cómo trabajan cada una de éstas. La figura 1.5 muestra el diagrama de bloques del puerto paralelo y la figura 1.7 el diagrama eléctrico. Ambas figuras ayudan a aclarar la siguiente explicación.

-PRINT DATA W.

La señal -PRINT DATA W sirve para enviar los bits que componen el caracter al exterior, a través de un CI 74LS273.

Este circuito integrado consta de 8 flip-flops D que siempre están habilitados, ya que la entrada CLR está conectada a 5 volts permanentemente. La señal -PRINT DATA W funciona como reloj de los flip-flops y es invertida de antemano porque las salidas se activan durante el flanco positivo (de subida). Cuando -PRINT DATA W baja, en las salidas de este circuito se tienen los mismos valores que hay en las entradas. De este modo, un bit 1 enviado al puerto resultará en un nivel lógico TTL en el pin correspondiente del conector DB-25.

Los capacitores que están conectados a cada uno de los bits de salida de dicho integrado sirven para eliminar ruido.

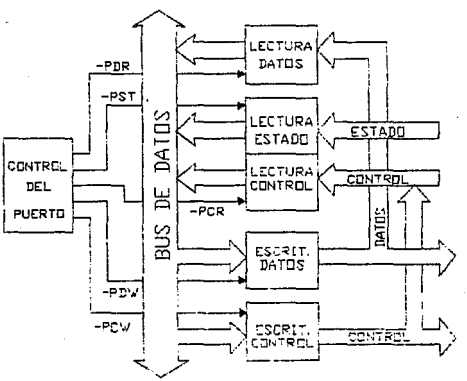


FIG. 1.5
DIAGRAMA DE BLOQUES DEL PUERTO

-PRINT DATA R.

La señal -PRINT DATA R permite leer los bits que se encuentran en las líneas de datos con el propósito de verificar que el dato se escribió correctamente. -PRINT DATA R funciona como habilitador de ocho buffers tres estados, contenidos en el CI 74LS244. Cuando -PRINT DATA R es baja, se tiene en las salidas el mismo valor que en las entradas.

En el diagrama se observa que las salidas del 74LS244 están conectadas a otras señales pero éstas no afectan al dato leído. Como las señales -PRINT STATUS y -PRINT CONTROL R son altas (recordar que de las salidas del 74LS155 sólo una es baja a la vez), las salidas del 74LS240 están en alta impedancia y es como si no estuvieran conectadas a las líneas de salida del 74LS244. También la señal de -Error se encuentra en alta impedancia como puede deducirse del circuito, tomando en cuenta que el estado de -PRINT STATUS es alto.

-PRINT CONTROL W.

La línea -PRINT CONTROL W permite escribir las señales de control a través del CI 74LS174, el cual contiene seis flip-flops tipo D. El valor que toman estas cuatro señales está dado por las líneas de datos XD0 a XD3, donde -Strobe es el negado de XD0, -Auto FDXT es el negado de XD1, -Init es igual a XD2 y -Slct In es el negado de XD3. -PRINT CONTROL W es invertida para actuar como reloj de los flip-flops, los cuales se activan con flanco positivo.

La línea XD4 pasa también a través del 74LS174 y es la que permite que la señal -Acknowledge genere o no una interrupción de nivel 7. Si se escribe un uno en XD4, al bajar -PRINT CONTROL W, la línea de -Acknowledge queda en

posibilidad de producir interrupciones.

-PRINT CONTROL R.

El CI 74LS240 contiene ocho buffers inversores con salida tres estados. Dos entradas de habilitación controlan los buffers manejándose como dos conjuntos independientes de cuatro buffers cada uno.

-PRINT CONTROL R habilita los buffers que permiten leer las señales de estado -Strobe, -Auto FDXT, -Init y -Slct In. Esta señal tiene dos usos: leer los valores de las líneas de control que se han mandado escribir con -PRINT CONTROL W o leer señales que provengan del exterior. Todas las señales se leén por el bus de datos (XD0 a XD7).

En el primer caso los valores de -Strobe, -Auto FDXT, -Init y -Slct In que se escriban, serán los mismos que se lean en las líneas XD0, XD1, XD2 y XD3 respectivamente. Esto es así porque -Strobe, -Auto FDXT y -Slct In son negadas al salir y al leerse por el 74LS240 se vuelven a negar. -Init es negada dos veces al salir pero es negada otras dos al leerse (por un inversor y por el 74LS240).

Para el segundo caso, como las cuatro señales son salidas colector abierto, estas mismas líneas pueden usarse como entradas. Si el registro de salida 74LS174 produce un nivel lógico TTL alto en -Strobe, -Auto FDXT, -Init y -Slct In (después de los inversores), este nivel puede bajarse por otras señales que se unan a estas líneas. Así, un circuito externo puede controlar el nivel de estas señales y, usando el CI 74LS240, éstas pueden ser sensadas. Si se programan estas señales de salida a un estado lógico alto, pueden usarse como entradas. En este caso los valores de -Strobe, -Auto FDXT y -Slct In se leerán invertidos en las líneas XD0,

XD1 y XD3 (porque el 74LS240 los invierte) y el mismo valor de -Init se leerá en XD2 (porque es invertido dos veces).

En ambos casos la línea XD4 contiene el valor del bit que habilita las interrupciones de nivel 7. Si se lee un uno, las interrupciones están habilitadas; si se lee un 0 no lo están.

Hay otras cinco señales conectadas al bus de datos pero están deshabilitadas. -Error está deshabilitada porque -PRINT STATUS es alta, por tanto -XIOR está en alta impedancia y ésto ocasiona que -Error también esté en alta impedancia. El hecho de que -XIOR esté en alta impedancia provoca que la otra mitad de los buffers del 74LS240 se encuentren deshabilitados ya que 1G1 es alta.

Señal -PRINT CONTROL R baja
(usada para leer los valores escritos)
Puertos: 37AH, 27AH o 3BEH
Operación: Lectura (IN)

XD0 = STROBE
XD1 = AUTO FDXT
XD2 = INIT
XD3 = SLCT IN (ERROR está en alta impedancia)
XD4 = 1 si las interrup. están habilitadas.
0 si no lo están.
(es el valor de XD4 anteriormente escrito)
XD5 = XD6 = XD7 = ALTA IMPEDANCIA

Señal -PRINT CONTROL R baja
(usada para leer del exterior)
Puertos: 37AH, 27AH o 3BEH
Operación: Lectura (IN)

XD0 = Inverso de STROBE
XD1 = Inverso de AUTO FDXT
XD2 = INIT
XD3 = Inverso de SLCT IN
XD4 = 1 si las interrup. están habilitadas.
0 si no lo están.
(es el valor de XD4 anteriormente escrito)
XD5 = XD6 = XD7 = ALTA IMPEDANCIA

-PRINT STATUS.

-PRINT STATUS permite leer las líneas de estado -Error, Slct, PE, -Ack y Busy. Todas se leén por el bus de datos (XD0 a XD7). En XD3 se puede leer el valor de -Error porque -XIOR es baja y habilita al buffer independiente que controla a -Error. -PRINT STATUS está conectada a un buffer tres estados habilitando, cuando es baja, a la señal -XIOR. Como -XIOR también es baja por ser una lectura, habilita cuatro buffers del CI 74LS240 que permiten leer las otras cuatro señales de estado. En XD4 se presenta el valor de Slct; en XD5 se presenta el valor de PE, en XD6 el de -Ack y en XD7 el negado de Busy. Por otra parte, XD0 a XD2 están deshabilitadas porque -PRINT CONTROL R es alta. -Slct In, que también llega a XD3, está en alta impedancia. El bit que controla las interrupciones y que llega a XD4 está deshabilitado también porque -PRINT CONTROL R es alta. Resumiendo:

Señal -PRINT STATUS baja
Direc. E/S: 379H, 279H o 3BDH
Operación: Lectura (IN)

XD0 = XD1 = XD2 = ALTA IMPEDANCIA
XD3 = ERROR (SLCT IN está en alta impedancia)
XD4 = SLCT
XD5 = PE
XD6 = ACK
XD7 = Inverso de BUSY

En la figura 1.6 se muestra cómo operan estas señales.

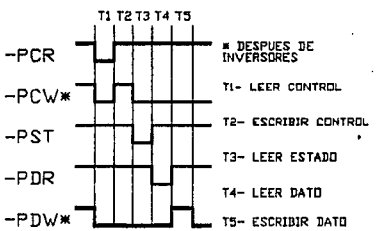


FIG. 1.6
DIAGRAMA DE TIEMPOS DEL CONTROL DEL PUERTO

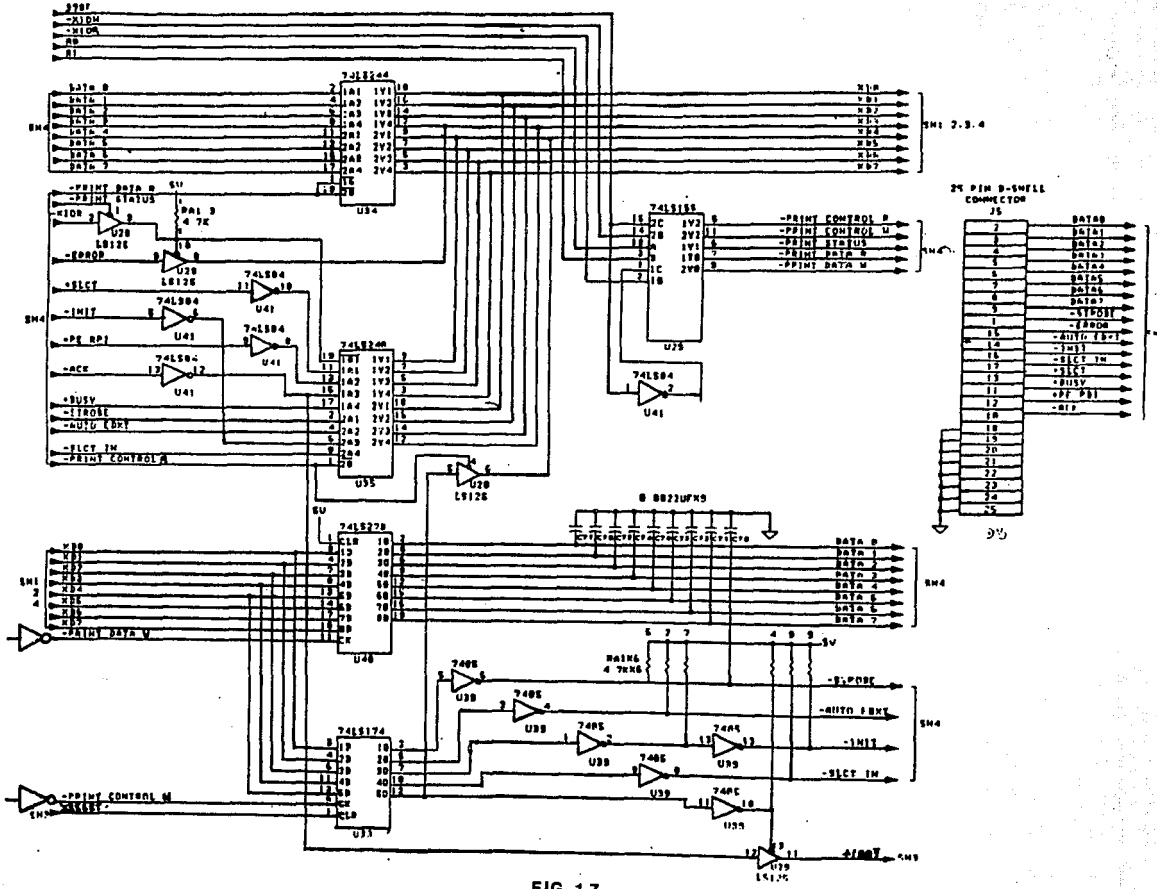


FIG. 1.7

1.3 SOFTWARE.

1.3.1 EL BIOS.

BIOS significa Servicios Básicos de Entrada/Salida (Basic Input/Output Services). El BIOS es parte de la memoria ROM que está en uso activo siempre que la computadora está trabajando. Se puede dividir a la ROM de las computadoras PC, en cuatro partes principales: las rutinas de encendido, el BIOS, ROM BASIC y las extensiones de ROM.

La primera parte es usada sólo cuando la computadora es encendida, prueba e inicializa los programas que hacen seguro el buen funcionamiento de la computadora.

La segunda parte, conocida como BIOS (algunas personas llaman BIOS a toda la memoria ROM y otras se refieren con este término exclusivamente a esta parte), tiene como función proveer los servicios fundamentales que se necesitan para la operación de la computadora, como por ejemplo, la pantalla, el teclado, los drives, etc. En otras palabras, el BIOS es el programa controlador de dispositivos, esto es, el programa que traduce un simple comando, como leer algo de disco, en todos los pasos necesarios para ejecutar dicho comando, incluyendo detección y corrección de errores. Además incluye las rutinas que contienen información o ejecutan tareas que son fundamentales para otros aspectos de la operación de la computadora, como mantener la hora del día. El programa BIOS se apoya entre los programas que son ejecutados en la RAM y el hardware actuando como interface entre ambas partes. Debido a ello trabaja hacia ambas partes recibiendo, por un lado, los requerimientos de los programas y ejecutando los servicios estándar de entrada/salida de BIOS. Un programa llama a este servicio con un par de

números, uno correspondiente a la interrupción y el otro al servicio requerido. Por el otro lado, el BIOS se comunica con los dispositivos hardware de la computadora, usando el código de comandos que cada dispositivo necesita. También maneja cualquier interrupción de hardware que el dispositivo genera.

La tercera parte se aplica sólo a la familia de las PC hechas por IBM y contiene el núcleo del lenguaje de programación BASIC, el cual puede ser usado por sí mismo o servir como parte del BASIC que viene con DOS.

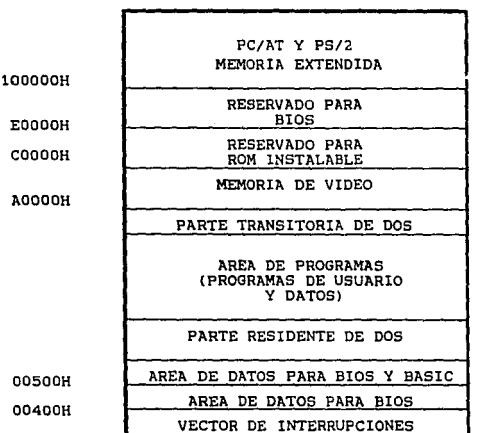


FIG. 1.8
MAPA DE MEMORIA

La cuarta y última parte es conocida como extensiones de la ROM, es decir, son programas agregados a la ROM principal cuando cierto equipo opcional se añade a la computadora.

Los programas ROM ocupan las direcciones F000:0000H a F000:FFFFH de la memoria en lo que se refiere a la familia de PC, XT y AT, y en las PS/2 se encuentran en las direcciones E000:0000H a F000:FFFFH.

Area de Datos BIOS. La parte más baja de la memoria de la computadora se considera aparte para usos importantes en el funcionamiento de la misma. Se tienen tres divisiones para estos usos especiales. La primera de ellas es la tabla del vector de interrupciones que se encuentra en los primeros 1024 bytes de memoria (direcciones absolutas de memoria 00000H a 003FFH). La segunda de las divisiones es el Area de Datos BIOS ocupando los siguientes 256 bytes de memoria (direcciones absolutas 00400H a 004FFH). La última de esas divisiones es el área usada por DOS y BASIC, la cual ocupa los siguientes 256 bytes (direcciones absolutas de memoria 00500H a 005FFH).

En las localidades del Area de Datos BIOS se encuentran datos importantes para la operación de las rutinas de servicio de DOS y BIOS. En cualquier momento que un programa lo solicite puede obtener la información almacenada en esas localidades. Un ejemplo de información importante almacenada en esta área de datos es la dirección base donde se encuentra el puerto paralelo, la cual se localiza en la dirección 0040:0008H para el primer puerto, en la 0040:000AH para el segundo y en la 0040:000CH para el tercero, si tiene los tres puertos. En el caso de que no los tenga todos, las direcciones restantes contendrán ceros. Además en las direcciones 0040:0078H, 0040:0079H y 0040:007AH se encuentra

el valor de time out para cada uno de los puertos respectivamente.

1.3.2 VECTOR DE INTERRUPCIONES.

Todas las PC basadas en la familia de microprocesadores Intel 80X86, son controladas en mayor parte por el uso de interrupciones que pueden ser generadas por hardware o software. Las rutinas de servicio BIOS trabajan de igual forma. Una interrupción es una llamada para el procesador indicándole que se requiere su atención. Estas interrupciones tienen asignado un número con el que pueden ser llamadas. Como regla general las interrupciones pueden dividirse en seis categorías:

1.- Interrupciones del Microprocesador, a menudo llamadas interrupciones lógicas y corresponden a las primeras cinco interrupciones, de la 00H a la 04H.

2.- Interrupciones Hardware, construidas dentro del hardware de la PC. En éstas y en las XT corresponden a los números 08H al 0FH y en las PS/2 del 08H al 0FH y del 70H al 77H.

3.- Interrupciones Software, incorporadas dentro del diseño de la PC son parte de los programas BIOS y son generadas por la instrucción INT. Las interrupciones correspondientes son de la 10H a la 1FH y de la 40H a la 5FH.

4.- Interrupciones DOS, siempre disponibles cuando DOS está en uso por los requerimientos de los lenguajes y programas. Los números son del 20H al 3FH.

5.- Interrupciones BASIC, son asignadas por el lenguaje mismo y son de la 80H a la F0H.

6.- Interrupciones de uso general, son disponibles para el uso temporal en los programas. Corresponden de la 60H a la 66H.

Cuando una interrupción ocurre, el control de la computadora se pasa a una subrutina llamada manejador de interrupciones, que a menudo es almacenada en la ROM del sistema. Dicho manejador de interrupciones es una rutina de servicio BIOS y se lo solicita poniendo la dirección de segmento y desplazamiento en el registro que controla el programa actual, es decir, en los registros CS (Segmento de Código) e IP (Apuntador de Instrucción), que juntos, CS:IP, son conocidos como registro par.

A la dirección que localiza el manejador de interrupciones se le conoce como Vector de Interrupción. La dirección de cada uno de los manejadores de interrupciones se encuentra en una tabla denominada Tabla de Vector de Interrupciones que comienza en 0000:0000H, esto es, al comienzo de la memoria. Cada vector de interrupción tiene cuatro bytes de tamaño y en la tabla están ordenados numéricamente. Durante el proceso de encendido, el BIOS coloca los vectores de interrupción para que apunten a los manejadores de interrupción en la ROM. Así, cada interrupción introducida en la tabla es almacenada como un par de palabras (cada cuatro bytes hay una interrupción), colocando en la primera de ellas el desplazamiento (IP) y en la segunda palabra el segmento (CS). En otras palabras, cuando se desea saber en que dirección se encuentra una interrupción, se debe invertir el orden de las palabras y de los bytes de éstas. Esto se debe hacer para cualquier localidad de memoria que se lee.

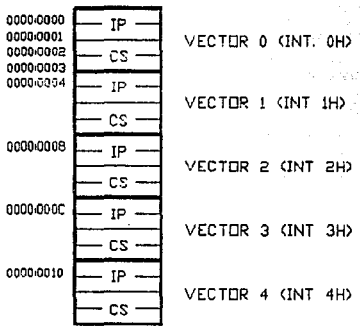


FIG. 1.9
TABLA DE VECTORES DE INTERRUPCION

Para encontrar en qué localidad de la tabla en memoria se encuentra el vector de una interrupción en particular, basta con multiplicar el número de la interrupción por cuatro (cuatro bytes por interrupción). Por ejemplo para obtener en qué dirección de memoria se encuentra el vector de la interrupción 5H, que convoca el servicio de imprimir el contenido de la pantalla, se multiplica 5H por 4, obteniéndose 20 (16H). Se observa entonces que en las localidades 14H 15H 16H 17H (20 a 23 decimal) se tiene 54 FF 00 F0 respectivamente (estos valores pueden variar entre computadoras). Por lo tanto, cambiando las palabras y los bytes, la dirección resultante es F000:FF54H, lo que quiere decir que la interrupción 5H se encuentra en el segmento de código F000H y en el desplazamiento FF54H.

1.3.3 INTERRUPCION 17H.

La interrupción 17H se encarga de solicitar los servicios de la impresora. Su vector de interrupción se encuentra en la localidad 92 (5CH) de la tabla del vector de interrupciones porque 23(17H) x 4 = 92. Esta interrupción consta de tres servicios para el uso de la impresora: el servicio 00H se utiliza para enviarle un byte ya sea para imprimirlo o para alguna orden especial como código de control o secuencia de escape. El servicio 01H se encarga de inicializar la impresora. Por último el servicio 02H se solicita cuando se requiere saber el estado de la impresora.

Para hacer uso de cada uno de estos servicios, se debe poner en el registro AH el número de servicio que se solicita y en el registro DX el número de impresora, de 0 a 2, en la cual se realizará el servicio. Este número de impresora depende del número de puertos que se tienen, es decir, si sólo se tiene uno, será LPT1 con número de impresora 0. Para el caso que se tengan los tres puertos, al puerto llamado LPT1 le corresponde el número de impresora 0, al puerto LPT2 el número de impresora 1 y al puerto LPT3 el número de impresora 2. Además para utilizar el servicio 00H, enviar un byte a la impresora, se debe poner en el registro AL el byte que se desea enviar. El estado de la impresora se puede ver en el registro AH para cada uno de los servicios, tomando en cuenta que se deben invertir algunos bits.

La interrupción 17H trabaja de la siguiente forma: después de colocar en los registros los valores correspondientes y llamar a la interrupción, el BIOS localiza en las posiciones 5CH 5DH 5EH 5FH (92 a 95 decimal) los números D2 EF 00 F0 con los cuales, siguiendo la regla antes descrita, se tiene que la interrupción se localiza en la dirección de memoria F000:EFD2H, esta dirección depende

de la computadora con que se trabaje.

Lo primero que hace la interrupción es guardar en el stack los valores que contienen los registros que va a utilizar, ésto lo hace para que cuando termine la interrupción los registros recuperen sus valores que tenían antes de entrar a ella y el programa que la llamó continúe sin problema. A continuación coloca en el registro DS la dirección de segmento de comienzo del Area de Datos BIOS para localizar si hay puerto y cuál es. Esto lo hace sumando primero a la dirección de inicio un ocho, puesto que como ya se dijo, la dirección base del puerto se encuentra en la dirección 0040:0008H. Después a esta dirección le suma 0, 2 o 4 dependiendo de la impresora con la que se desea trabajar (si es que hay varias), LPT1, LPT2 o LPT3 respectivamente, según el valor escrito en el registro DX. También coloca en otro registro el valor de time out, que se encuentra en la dirección de comienzo del Area de Datos BIOS más setenta y ocho (más 0, 1 o 2 dependiendo de la impresora).

En el caso de que no haya puerto, la interrupción termina recuperando los valores de los registros y regresando a donde fue llamada. Una vez que ya tiene definido con qué puerto o impresora trabajará, determina qué servicio va a relizar y lo ejecuta.

Para el servicio 00H pone en la dirección base del puerto el byte a ser enviado. Por dirección base + 1 lee el estado de la impresora (usando el servicio 02H que se explicará más adelante) para darse cuenta si puede enviar el dato o no, ya que la impresora puede estar ocupada, por ejemplo. Si este es el caso, comienza a correr el time out para ver si en ese tiempo el dato puede ser impreso. Si el tiempo se agotó y no se imprime, leerá nuevamente el estado de la impresora por dirección base + 1, el cual indicará que

el time out ya terminó y el dato no fue impreso y con ésto la interrupción termina.

Si no hay nada que impida la impresión del dato, desde que se manda o antes de que el time out termine, entonces lo que hace es mandar por las líneas de control (por dirección base + 2) un 0DH, bajando las señales de -Strobe y -Slct In. Con ésto se indica que el dato está listo y que la impresora está seleccionada. Después, manda por la misma dirección un 0CH para subir -Strobe y dejar -Slct In en bajo, indicando que seguirá seleccionada la impresora. Por último lee en dirección base + 1 el estado de la impresora para saber si el dato fue recibido e impreso y la interrupción termina.

En el caso del servicio 01H, que se utiliza para inicializar la impresora, lo que hace es enviar dos valores por dirección base del puerto + 2. Primero manda un 08H para bajar -Init y -Slct In (-Strobe y -Auto FDXT se ponen en alto), y después envía un 0CH para subir -Init, dejando únicamente -Slct In en bajo. Es decir, se ha seleccionado e inicializado la impresora. Ya hecho ésto, lee por dirección base + 1 el estado de la impresora y la interrupción termina.

Para obtener el estado de la impresora, servicio 02H, lee el contenido del puerto usando la dirección base + 1 y poniendo el resultado en el registro AH pero invirtiendo los bits correspondientes a -Error y -Acknowledge. Es decir, cuando se produce un problema, la señal -Error es baja pero la interrupción lo reporta como un 1 y cuando se recibe un caracter -Acknowledge es baja pero la interrupción también lo reporta como un 1. Cuando la impresora esta ocupada la señal Busy es alta pero para la interrupción es baja, ésta no es invertida por ella porque el hardware se encarga de hacerlo.

CAPITULO 2

PROGRAMADOR DE EPROM

2.1 INTRODUCCION.

2.1.1 ANTECEDENTES.

Como se mencionó anteriormente, el elemento principal del Dispositivo Protector de Software es una memoria del tipo EPROM. Para la producción comercial de este Protector de Software es necesario contar con un programador de EPROM que permita grabar la información en la memoria. Debido a que uno de los objetivos de este trabajo de tesis es diseñar la tecnología necesaria para producir el Protector de Software, el presente capítulo está enteramente dedicado al diseño e implementación de un programador de EPROM. Este programador de EPROM puede ser comercializado inclusive como un producto independiente.

Es importante recordar que el objetivo principal no es el de crear un programador universal de memorias EPROM, es decir, uno que permita grabar cualquier tipo de memoria EPROM, sino diseñar sólo el programador adecuado al tipo de memoria que utiliza el Protector de Software. Por este motivo, el programador que se ha diseñado es aplicable sólo al tipo de memoria EPROM 2716 o 27C16.

2.1.2 MEMORIAS EPROM.

La memoria EPROM (Erasable Programmable Read Only Memory o Memoria Programable Borrable de Sólo Lectura) es un tipo de memoria que se puede considerar intermedia entre una memoria de sólo lectura .(ROM) y una memoria de lectura-escritura de acceso aleatorio (RAM). La memoria EPROM proporciona las ventajas de la no-volatilidad, ya que es utilizada como memoria de sólo lectura la mayor parte del tiempo sin perder su información cuando el suministro de energía es suspendido, junto con la ventaja de poder borrarla y escribir en ella ocasionalmente, si fuera necesario.

INTEL y otros fabricantes, utilizan como elemento básico de la memoria un transistor con una compuerta flotante se silicio llamada FAMOS (Semiconductor de Oxido Metálico de Inyección por Avalancha y Puerta Flotante o Floating Gate Avalanche - Inyection MOS). Cuando se aplica un voltaje determinado a los elementos direccionados, se pueden escribir en la EPROM las configuraciones de bits deseadas. Esta carga puede durar años, ya que está rodeada de un material aislante llamado óxido de silicio (SiO_2). El proceso de borrado se realiza con la exposición del sustrato del chip a la acción de luz ultravioleta, ya que el chip está provisto de una ventana transparente de cuarzo. La exposición a este tipo de luz puede drenar la carga por efecto fotoeléctrico ocasionando que se borre la información almacenada.

Como se sabe, existen muchos tipos de memorias EPROM en el mercado. Por ejemplo, las hay desde algunas que necesitan varios pulsos de programación para ser grabadas (la 2708 requiere 100 pulsos para cada localidad de memoria), hasta las que necesitan sólo uno. El valor del voltaje aplicado en

el pulso de programación también es variable: desde niveles TTL hasta 25V o más. El tamaño de la memoria (el número de localidades que la componen y el número de bits que componen cada localidad) es otro factor a considerar en la elección de la misma.

2.1.3 MEMORIA EPROM 27C16.

De esta diversidad de memorias, se ha optado por la EPROM 27C16 por su bajo costo y su capacidad de almacenamiento, que sin ser muy grande, es adecuada a nuestras necesidades (2K X 8). Por su tiempo de acceso y su desempeño es ideal para aplicarse con los microprocesadores 8085 y la familia 80X86 de INTEL. Además, su consumo de energía es muy reducido al utilizar tecnología CMOS, lo cual se indica con la letra C en su nomenclatura. A continuación se mencionan las características principales de la memoria EPROM 27C16:

- Organización: 2K X 8.
- Tiempo de acceso rápido: 300 ns.
- Bajo consumo de potencia CMOS:
  Potencia activa: 26.25 mW máxima.
  Potencia "a la espera" (standby): 0.53 mW máxima.
- El pulso de programación es nivel TTL.
- Salidas y entradas compatibles con niveles TTL.
- Salidas Tres-Estados.
- Compatible en terminales con la memoria 2716.

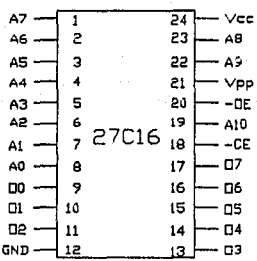


FIG. 2.1
TERMINALES DE LA MEMORIA 27C16

En el apéndice A se pueden consultar las hojas de especificaciones técnicas completas de la 27C16 proporcionadas por el fabricante INTEL.

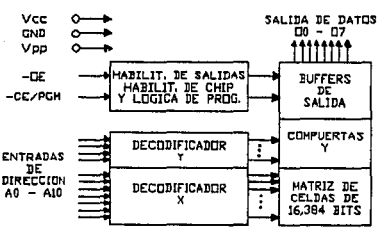


FIG. 2.2
DIAGRAMA DE BLOQUES DE LA MEMORIA 27C16

La EPROM 27C16 tiene los siguientes grupos de señales:

A0 a A10	Direcciones.
O0 a O7	Datos.
-CE/PGM	Habilitación de Chip/Programa (Chip Enable/ Program).
-OE	Habilitación de salida (Output Enable).
Vpp	Voltaje de programación.
Vcc	Voltaje de alimentación.
GND	Tierra.

2.2 HARDWARE.

2.2.1 PROPUESTA.

Se ha decidido que la programación de la memoria se efectúe a través de una computadora PC compatible, debido a que ésta proporciona facilidad de manejo para el usuario, ya que se creó un programa que actúa como una interface que permite introducir los datos a grabar y visualizar el contenido de la memoria.

El proceso de diseño del programador se ha divido en cuatro partes:

a) Diseño del Control de la Memoria. Se le ha dado este nombre a la parte que realiza la programación propiamente dicha de la memoria para distinguirla del Programador de EPROM completo.

b) Diseño de la Interface Computadora-Programador.

c) Diseño de la Fuente de Alimentación.

d) Diseño del software para la programación.

El diagrama de la figura 2.3 muestra esquemáticamente el diseño del programador.

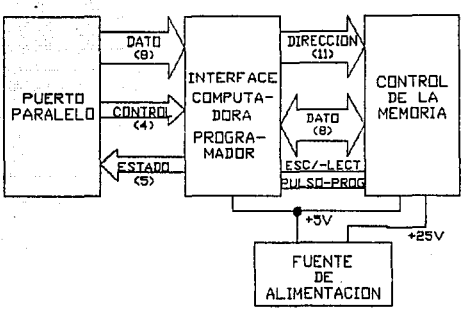


FIG. 2.3
DIAGRAMA DE BLOQUES DEL PROGRAMADOR

A continuación se analizan por separado y en detalle cada una de estas partes.

2.2.2 DISEÑO DEL CONTROL DE LA MEMORIA.

En la figura 2.3 se observan las señales que llegan al Control de la Memoria provenientes de la Interface Computadora-Programador. Estas señales son: las 11 direcciones que se han denominado DIR0 a DIR10; 8 líneas de datos, llamadas DAT0 a DAT7 (no confundir con las líneas que salen del puerto y que son DATA0 a DATA7); la señal ESC/-LECT; y la señal PULSO_PROG.

Sin importar por el momento la procedencia de estas señales (ya que ésto corresponde a la Interface Computadora-

Programador), se analizará el Control de la Memoria, el cual consiste en el circuito encargado de generar el pulso de programación y establecer los valores adecuados en las entradas de alimentación de la memoria para que ésta pueda ser grabada o leída.

Programación. La programación de cualquier EPROM se efectúa en tres pasos:

1.- Direccionar la localidad donde se va a almacenar el dato.

2.- Colocar el dato en los pines que normalmente son salidas.

3.- Aplicar el pulso de programación con lo cual el dato quedará permanentemente grabado en la memoria.

Inicialmente todas las localidades de la EPROM 27C16 contienen UNOS. Cuando se introducen los datos, las posiciones de los bits con ceros cambian dichos unos por ceros y los bits con valor uno no se modifican. La única manera de cambiar un cero a uno es utilizando luz ultravioleta para borrar la EPROM. Las localidades de memoria se pueden programar en forma individual, secuencial o aleatoriamente.

La entrada Vcc siempre es +5V. Para programar la 27C16 la entrada Vpp debe estar forzosamente en +25V y la señal -OE debe ser de nivel lógico alto (+5V). En seguida, se coloca en los pines A0 a A10 la dirección que será grabada; esto se hace a niveles TTL, es decir, un 0 estará representado por un voltaje entre 0 y 1.5 volts y un 1 por un voltaje entre 3 y 5 volts. A continuación se aplica el

dato a grabar en los pines de salida de datos 00 a 07, también usando niveles TTL. Tanto los bits de las direcciones como de los datos se aplican en paralelo a sus respectivos pines. Una vez que las direcciones y los datos se estabilizan (2 microsegundos después de poner los datos) se aplica un pulso de programación de 50 milisegundos de duración (55 milisegundos máximo) en la entrada -CE/PGM para cada dirección a programar.

Lectura. La lectura de la memoria se realiza con la entrada Vpp a +5V y -CE/PGM en nivel de voltaje bajo. Vcc debe ser +5V, al igual que en la programación. Se coloca a continuación la dirección para la cual se quiere obtener el dato en las líneas A0 a A10. Posteriormente, -OE cambia a un voltaje bajo y un tiempo T_{OE} después (160 ns máximo), el dato estará disponible en las salidas 00 a 07.

La tabla siguiente resume las condiciones necesarias para la programación y la lectura de la 27C16:

\ PINES MODO \	-CE/PGM (18)	-OE (20)	Vpp (21)	Vcc (24)	SALIDAS (9-11,13-17)
LECTURA	V_{IL}	V_{IL}	+5	+5	DATOS DE SALIDA
PROGRAMACION	PULSO DE V_{IL} A V_{IH}	V_{IH}	+25	+5	DATOS DE ENTRADA

Pulso de Programación. Para obtener el pulso de programación se utilizó el circuito integrado temporizador 555 en modo monoestable o de "un disparo" (one-shot). Como su nombre lo indica, en este modo se obtiene un único pulso positivo a la salida con una duración preestablecida, cuando a la entrada se le aplica un pulso negativo.

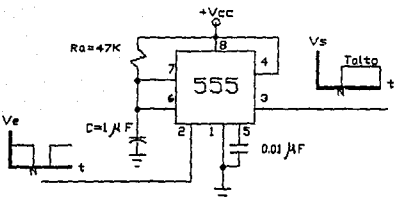


FIG. 2.4
TEMPORIZADOR 555

La duración del pulso está dada por la ecuación:

$$T_{alto} = 1.1RaC$$

donde Ra y C se refieren a los valores de la resistencia y del capacitor externos mostrados en la figura 2.4.

El pulso negativo debe tener una duración menor que la deseada en la salida.

En nuestro caso Ra = 47 KΩ; C = 1 μF; por lo tanto:

$$T_{alto} = 1.1(47x10^{3}\Omega)(1x10^{-6}F) = 0.0517s = 51.7ms$$

A este pulso de salida del CI 555 se le ha denominado PULSO_PROG y es aplicado directamente a la entrada -CE/PGM de la 27C16.

Control de Escritura y Lectura. A la señal que llega a la entrada -OE se le ha denominado ESC/-LECT para indicar que, cuando es alta, la EPROM está en modo de programación (o escritura) y cuando es baja está en modo de lectura.

Como se ve en la tabla anterior, Vpp varía de acuerdo a -OE: cuando -OE es baja (LECTURA), Vpp es +5 volts y cuando -OE es alta (ESCRITURA), Vpp es +25 volts. Aprovechando esta condición, se ha utilizado la señal ESC/-LECT para controlar un transistor Q1 que actúa como conmutador. Cuando ESC/-LECT es alta, el 7406 (U3) la invierte; la base recibe un voltaje bajo y el transistor está saturado, es decir, conduce. El diodo D1 queda conectado en inversa y se comporta como circuito abierto, protegiendo a la fuente de +5 volts. Por lo tanto, la entrada Vpp recibe los +25 volts necesarios para programar la EPROM. Ver la figura 2.5 para el diagrama del Control de la Memoria.

Cuando ESC/-LECT es baja, la base de Q1 recibe un voltaje alto y el transistor queda en la región de corte, es decir, se comporta como circuito abierto. Por tanto, la entrada Vpp recibe +5 volts para poder leer la EPROM.

La resistencia R1 y el capacitor C18 sirven para dar estabilidad al transistor. El LED 1 solamente sirve para indicar que la fuente de +5V está funcionando, mientras que el LED 2 indicará el estado de la señal ESC/-LECT (cuando es alta el led se apaga y viceversa).

Las señales restantes son DIR0 a DIR10 para recibir las direcciones y DAT0 a DAT7 para recibir los datos durante la programación y enviarlos a la computadora durante la lectura de la 27C16.

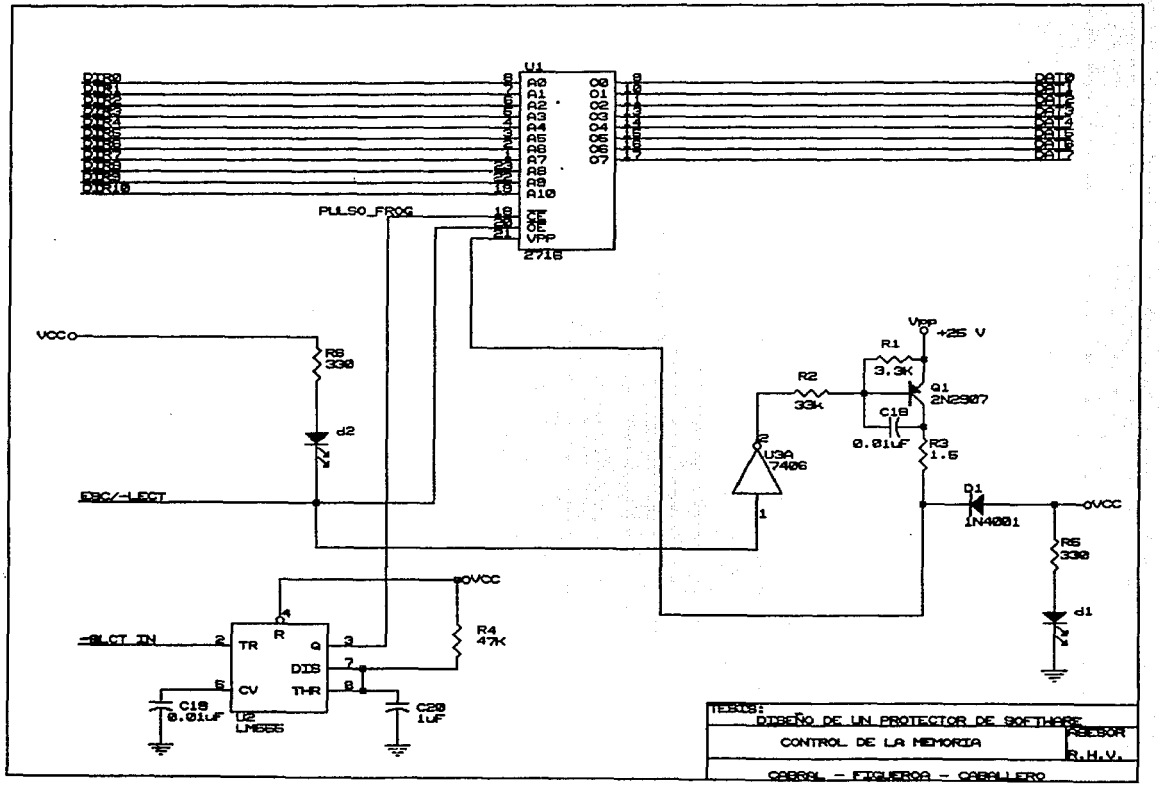

U1
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
O0
O1
O2
O3
O4
O5
O6
O7
VCC
OE
VPP
2718
PULSO_FROG
VCCO
R5
330
d2
EDO/-LECT
VPP
+25 V
R1
3.3K
R2
33K
Q1
2N2907
C18
0.01uF
R3
1.5
U3A
7406
D1
1N4001
oVCC
R5
330
d1
-SLCT IN
R
TR
Q
DIS
CV
THR
U2
LM555
R4
47K
C19
0.01uF
C20
1uF
TEMA:
DISEÑO DE UN PROTECTOR DE SOFTWARE
CONTROL DE LA MEMORIA
ASESOR
R.H.V.
CABRAL - FIGUEROA - CABALLERO

FIG. 2.5

2.2.3 INTERFACE COMPUTADORA-PROGRAMADOR.

Como ya se explicó en el capítulo dedicado al puerto paralelo, éste puede enviar al exterior (hacia la impresora) 8 bits de datos y 4 líneas de control. Además puede recibir 5 señales para determinar el estado de una impresora. A continuación se trata el diseño de la interface entre estas señales del puerto paralelo y las señales del Control de la Memoria, es decir, las que realmente se encargan de la grabación y lectura de la misma. En la figura 2.6 se puede observar el diagrama de bloques de la Interface.

Funciones de la Interface. La Interface Computadora-Programador debe llevar a cabo las siguientes funciones (la numeración no indica necesariamente el orden en que se realizan):

1.- Controlar la señal ESC/-LECT para indicar con un nivel lógico alto que se desea grabar. Por el contrario, un nivel bajo indicará una operación de lectura.

2.- Permitir el paso de la parte baja de la dirección, de las ocho líneas de datos hacia las líneas de direcciones DIR0 a DIR7.

3.- Permitir el paso de la parte alta de la dirección, de las líneas de datos hacia las líneas de direcciones DIR8 a DIR10.

4.- Permitir el paso del dato (durante la grabación) a través de las ocho líneas de datos.

5.- Permitir la lectura de la parte baja del dato (en la lectura) a través de las Líneas de Estado.

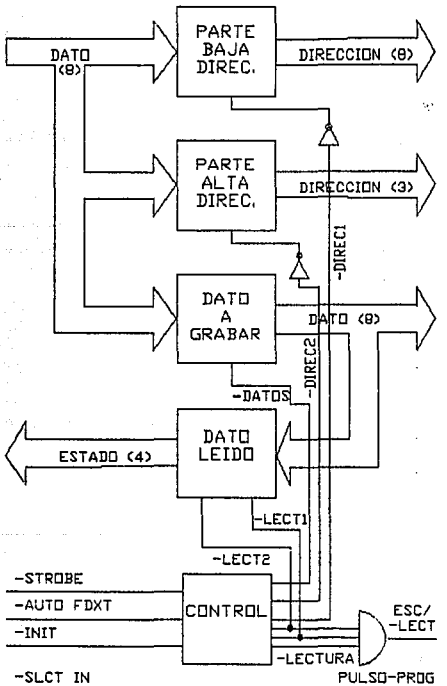


FIG. 2.6
DIAGRAMA DE BLOQUES DE LA INTERFACE

6.- Permitir la lectura de la parte alta del dato (durante la lectura) a través de las Líneas de Estado.

7.- Activar el pulso de programación durante la escritura.

Para poder proporcionar las once líneas de direcciones que la memoria 27C16 necesita, se utilizaron las líneas de datos. Sin embargo, debido a que el número de líneas de datos es menor que el número de líneas de direcciones, el proceso se efectúa en dos pasos: primeramente se envían los ocho bits menos significativos de la dirección y posteriormente, a través de las mismas líneas de datos, se envían los tres bits restantes para completarla. También, a través de las líneas de datos, se envía el dato a grabar durante el proceso de programación. Por otra parte, cuatro de las cinco señales de Estado se utilizaron como entradas cuando se efectúa la operación de lectura. Así, el dato a recuperar de la memoria se leerá en dos partes: primero los cuatro bits menos significativos, y a continuación los cuatro bits de la mitad superior. Por último, el grupo de las cuatro Señales de Control se utilizó para controlar al Programador.

Control del Programador. La base del circuito de control es un decodificador 74LS155. Este es operado a través de las señales -Strobe, -Auto FDXT e -Init. En la figura 2.7 se muestra el diagrama completo del Programador de EPROM. Debido a que el Programador es el único dispositivo conectado al puerto paralelo al trabajar con la memoria, estas señales se pueden dedicar exclusivamente a las funciones de control del mismo. Utilizar tres líneas como entradas permite tener $2^3=8$ salidas, de las cuales solamente una es baja a la vez. Para ésto, -Strobe está

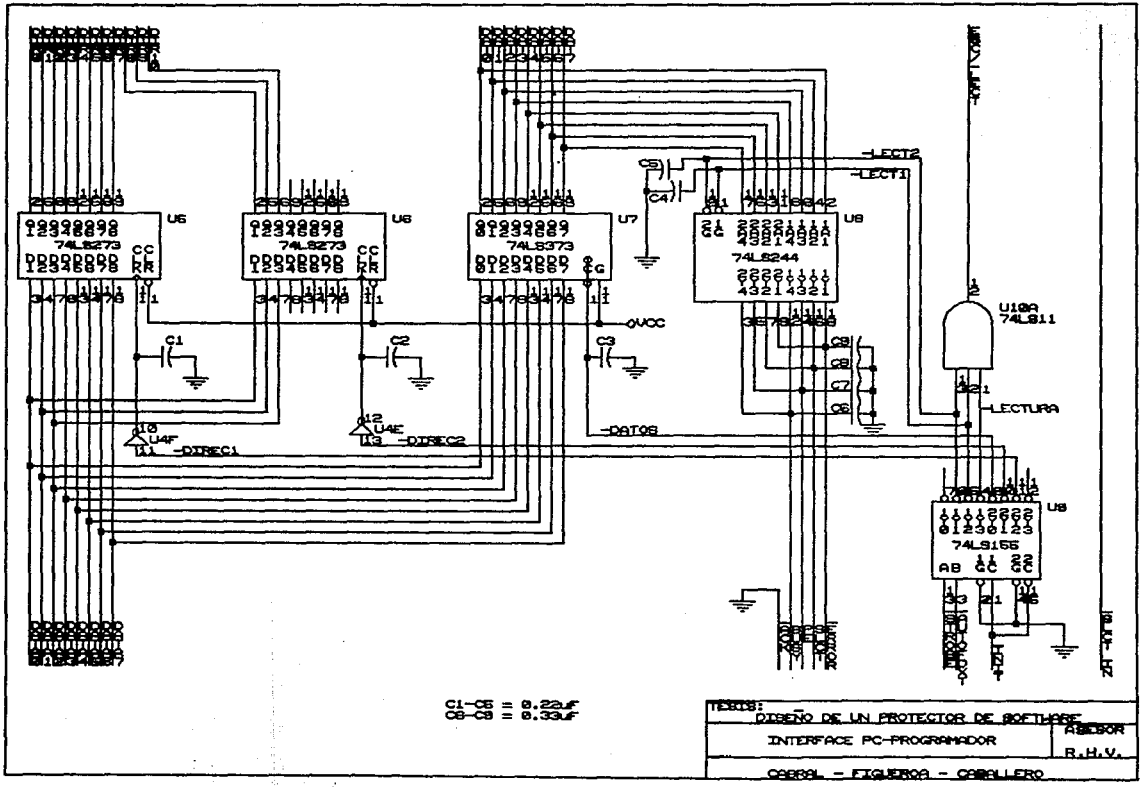

U5
U6
U6
U7
U8
74LS273
74LS273
74LS373
74LS244
C1
C2
C3
C4
C5
C6
C7
C8
C9
+VCC
-LECT2
-LECT1
U10A
74LS11
-LECTURA
U4F
-DIREC1
U4E
-DIREC2
-DATOS
U8
74LS155
C1-C5 = 0.22uF
C6-C9 = 0.33uF
TESIS:
DISEÑO DE UN PROTECTOR DE SOFTWARE
INTERFACE PC-PROGRAMADOR
ASESOR
R.H.V.
CABRAL - FIGUEROA - CABALLERO

FIG.2.7

conectada a la entrada A del decodificador, -Auto FDXT a la entrada B e -Init a las entradas C1 y C2. Con esto último se logra que los dos decodificadores internos del 74LS155 trabajen en forma complementaria: cuando -Init sea baja, el decodificador interno 2 estará habilitado y el decodificador 1 no y viceversa. En conjunto, las entradas A, B y C funcionan como líneas de selección. Por otra parte, las entradas -G1 y -G2 están conectadas a tierra para que el decodificador esté siempre habilitado. La tabla de función del decodificador 74LS155 como parte del Programador de EPROM se muestra a continuación:

Selección C	B	A	Hab. G	(0) 2Y0	(1) 2Y1	(2) 2Y2	(3) 2Y3	(4) 1Y0	(5) 1Y1	(6) 1Y2	(7) 1Y3
X	X	X	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H
L	H	L	L	H	H	L	H	H	H	H	H
L	H	H	L	H	H	H	L	H	H	H	H
H	L	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	L	H	H
H	H	L	L	H	H	H	H	H	H	L	H
H	H	H	L	H	H	H	H	H	H	H	L

Las salidas fueron asignadas como se indica:

No. SALIDA	NOMBRE EN DECO	NOMBRE EN DISEÑO	No. DE PIN
0	2Y0	-DATOS	9
1	2Y1	-DIREC2	10
2	2Y2	-DIREC1	11
3	2Y3	N.C.	12
4	1Y0	N.C.	7
5	1Y1	-LECT1	6
6	1Y2	-LECT2	5
7	1Y3	-LECTURA	4

Como se observa, las salidas 3 y 4 no están conectadas pero desempeñan una función importante: al referirse a cualquiera de ellas se ponen todas las demás señales (que si

están conectadas) en estado lógico alto. Posteriormente se verá la utilidad de este hecho. Las seis señales restantes tienen por objeto controlar las diferentes funciones del Programador. -DATOS sirve para permitir el paso de los datos durante la grabación (es la función que se numeró como 4 anteriormente). -DIREC1 y -DIREC2 habilitan a los circuitos que permiten alimentar las direcciones, en dos pasos, al leer o grabar (funciones 2 y 3). -LECT1 y -LECT2 habilitan al circuito que lee los datos en dos mitades (funciones 5 y 6). Por último, -LECTURA es la señal con la cual se controla a la señal ESC/-LECT, que a su vez le indica a la memoria si se graba o se lee de ella.

Se habrá observado que se han descrito seis señales mientras que se tienen siete funciones posibles. Esto es así porque la función número 7, producir el pulso de programación, no es controlada a través del decodificador 74LS155, sino que se activa directamente con la señal -Select In.

Al igual que se hizo en el capítulo referente al puerto paralelo, se analizarán por separado y con mayor detalle cada una de las señales de salida y los circuitos que éstas controlan.

- -DIREC1

Esta señal funciona como reloj del CI 74LS273 (IC5 en los diagramas), el cual contiene ocho flip-flops D. La información en las entradas D se transfiere a las salidas Q en el flanco positivo del pulso de reloj, esto es, durante la transición bajo-alto. Cuando en las entradas del decodificador 74LS155 se presenta la combinación que forma el número 2 en binario, de todas sus salidas solamente la señal -DIREC1 será baja. Es necesario entonces incluir un

inversor para lograr que -DIREC1 sea alta y pueda producir los valores deseados en las salidas de los flip-flops del CI 74LS273.

Las entradas D están conectadas a las líneas de datos DATA0 a DATA7 provenientes del puerto paralelo. Inicialmente estas líneas contienen los primeros ocho bits de la dirección que se proporciona a la memoria 27C16. Al bajar la señal -DIREC1 en la salida del 74LS155, es invertida y llega en estado lógico alto a la entrada CLK de los flip-flops. Por lo tanto, en las salidas de éstos se tienen los ocho bits denominados DIR0 a DIR7 que forman parte de la dirección. La entrada Clear (CLR), cuando es baja, pone todas las salidas en estado lógico bajo, esto es, las limpia. En el circuito está conectada permanentemente a +5V, lo cual la inactiva.

- -DIREC2

Al igual que la señal anterior, -DIREC2 activa las salidas de un CI 74LS273 (IC6), al estar conectada a la entrada de reloj CLK. Cuando en el 74LS155 se presenta un 1 binario en las entradas, la única salida en estado bajo es -DIREC2. Para lograr que un flanco positivo active las salidas de los flip-flops, es necesario invertir esta señal usando un 74LS04. La entrada Clear está conectada siempre a +5V por la misma razón que en el caso anterior. En las salidas 1Q a 3Q se tienen los mismos valores que en las entradas 1D a 3D. A su vez, estas entradas representan los tres bits más altos de la dirección, denominados DIR8, DIR9 y DIR10 en las salidas. Las líneas de datos restantes DATA3 a DATA7, no son utilizadas para este paso de proporcionar la parte alta de la dirección y por lo tanto no importa el valor que tomen. Las entradas que provienen de las líneas de datos en este momento han cambiado, pero ello no afecta a

las líneas DIR0 a DIR7, que conservan su valor, mientras -DIREC1 no cambie a estado lógico bajo en la salida del 74LS155.

- -DATOS

Esta señal controla las salidas de un CI 74LS373 cuya tabla de función es la siguiente:

CONTROL DE SALIDA	HABILIT. G	D	SALIDA
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

Como se observa, cuando la entrada Control de Salida es alta, no importando ni la entrada de habilitación G ni las entradas D, las salidas siempre serán alta impedancia. Si Control de Salida es baja y la habilitación G es alta, en las salidas se tiene el mismo valor que en las entradas. Si Control de Salida es baja y la habilitación G es baja, en las salidas se tiene el último valor almacenado cuando G subió por última vez, sin importar lo que haya en las entradas D.

En el circuito la entrada G está conectada en forma permanente a +5V para que permita el paso de las entradas D, en cualquier momento, hacia las salidas Q. El valor de estas salidas permanece hasta que otro dato sea colocado en las entradas. Estas entradas están conectadas a las líneas de datos del puerto DATA0 a DATA7 al igual que en los casos anteriores. Las salidas forman el dato a grabar en la memoria y han sido denominadas DAT0 a DAT7.

De la misma forma que en el caso anterior, los bits de direcciones DIR0 a DIR10 no son afectados y mantendrán su valor mientras -DIREC1 y -DIREC2 no cambien a un estado lógico bajo. La señal -DATOS llega a la entrada Control de Salida con un nivel bajo cuando a la entrada del decodificador 74LS155 se produzca un 0 binario en las líneas de selección. Este estado bajo de la entrada Control de Salida, junto con la habilitación G en nivel alto permanente, permiten que los datos pasen a través del 74LS373 hacia la memoria. Por el contrario, cuando la señal -DATOS tenga un valor alto, en las salidas de este circuito se tendrá alta impedancia, lo cual es necesario para poder utilizar las mismas líneas DAT0 a DAT7 como entradas, cuando se lee la memoria.

- -LECT1

El CI 74LS244 contiene ocho buffers con salida Tres-Estados, agrupados en dos conjuntos de cuatro buffers cada uno. La señal -LECT1 está conectada a la entrada -1G y sirve para habilitar al primero de estos conjuntos ya que, cuando en las líneas de selección del 74LS155 se tiene un 6, esta señal será baja permitiendo el paso de los cuatro bits menos significativos del dato a leer por las Señales de Estado: -Error, Slct, PE y Busy. Las otras cuatro salidas del segundo conjunto de buffers, que están conectadas a estas mismas Señales de Estado, se encuentran en alta impedancia y no interfieren con estas líneas.

- -LECT2

La señal -LECT2, conectada a la entrada -2G del 74LS244, habilita al segundo conjunto de buffers. Cuando en las líneas de selección del 74LS155 se tiene un 5, -LECT2 es baja y permite el paso de los cuatro bits más significativos

del dato a leer de la memoria. Con esta señal se completa la lectura del dato el cual es tomado de las líneas DAT0 a DAT7. Esta lectura también se efectúa mediante las Señales de Estado antes mencionadas y las cuatro salidas del primer conjunto de buffers no interfieren porque se encuentran en alta impedancia. Sin embargo, a cada una de las cuatro Señales de Estado utilizadas, fue necesario agregarles un capacitor de tantalio de 0.33 µF para evitar que el ruido, producido por el cambio rápido de habilitación entre los dos conjuntos internos de buffers del 74LS244, genere interferencia provocando una lectura equivocada del dato.

Como ya se mencionó, de las cinco Señales de Estado sólo se utilizan cuatro. La línea restante, -Ack, está conectada a tierra permanentemente para fijar su valor en un estado lógico bajo y no afecte la lectura del dato.

A las señales DIREC1, DIREC2, -DATOS, -LECT1 y -LECT2 se les conectó un capacitor de cerámica de 0.22 µF porque el ruido provocaba que los valores de los bits de dirección, DIR0 a DIR10, se alteraran cuando las salidas del decodificador cambiaban.

- -LECTURA

Ya se dijo que -LECTURA es la señal que controla a ESC/-LECT para indicar en la entrada -OE de la memoria, en que momento se escribe o se lee. Es necesario que ESC/-LECT sea baja mientras se leén las dos mitades del dato, esto es, mientras -LECT1 y -LECT2 son bajas. Además, ESC/-LECT debe bajar un tiempo T_{OE} antes de que el dato esté disponible en las salidas. Esto no se puede controlar directamente con otra salida del decodificador porque sólo una de las salidas es baja a la vez. La señal -LECTURA ayuda a agregar la demora T_{OE} necesaria antes de bajar -LECT1 y -LECT2. -

LECTURA es una salida del decodificador y se activa al colocar un 7 en las líneas de selección del mismo. Utilizando una compuerta AND (del CI 74LS11) con las señales -LECTURA, -LECT1 y -LECT2 como entradas, se obtiene a la salida la señal ESC/-LECT que será baja durante el tiempo en que cualquiera de las entradas sea baja. Recordar que en la función AND basta con que una de las entradas sea baja para que la salida también sea baja. A continuación se muestra el diagrama de tiempos de las señales utilizadas durante el tiempo de lectura de un dato.

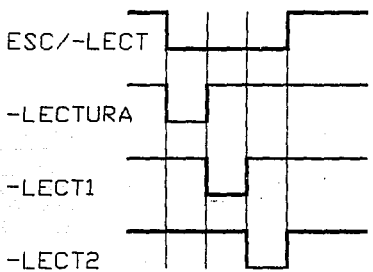


FIG. 2.8
DIAGRAMA DE TIEMPOS EN LA LECTURA

- -SLCT IN

Por último, la señal -Slct In se utilizó para activar directamente al temporizador 555. Al generarse una transición de alto a bajo en esta señal, este circuito produce el pulso de programación necesario para grabar una localidad de memoria.

2.2.4 FUENTE DE ALIMENTACION.

El Programador de EPROM depende completamente de una fuente de alimentación. Por tal motivo se diseñó una fuente que permita suministrar los voltajes necesarios para el funcionamiento de los circuitos integrados TTL y para la programación de la memoria 27C16. En realidad se trata de dos fuentes independientes que proporcionan +5V y +25V respectivamente. Para la fuente de +25V se utilizó un trimpot o resistencia variable de ajuste fino para calibrar con precisión el voltaje de programación en la entrada Vpp de la memoria. La figura 2.9 muestra la fuente.

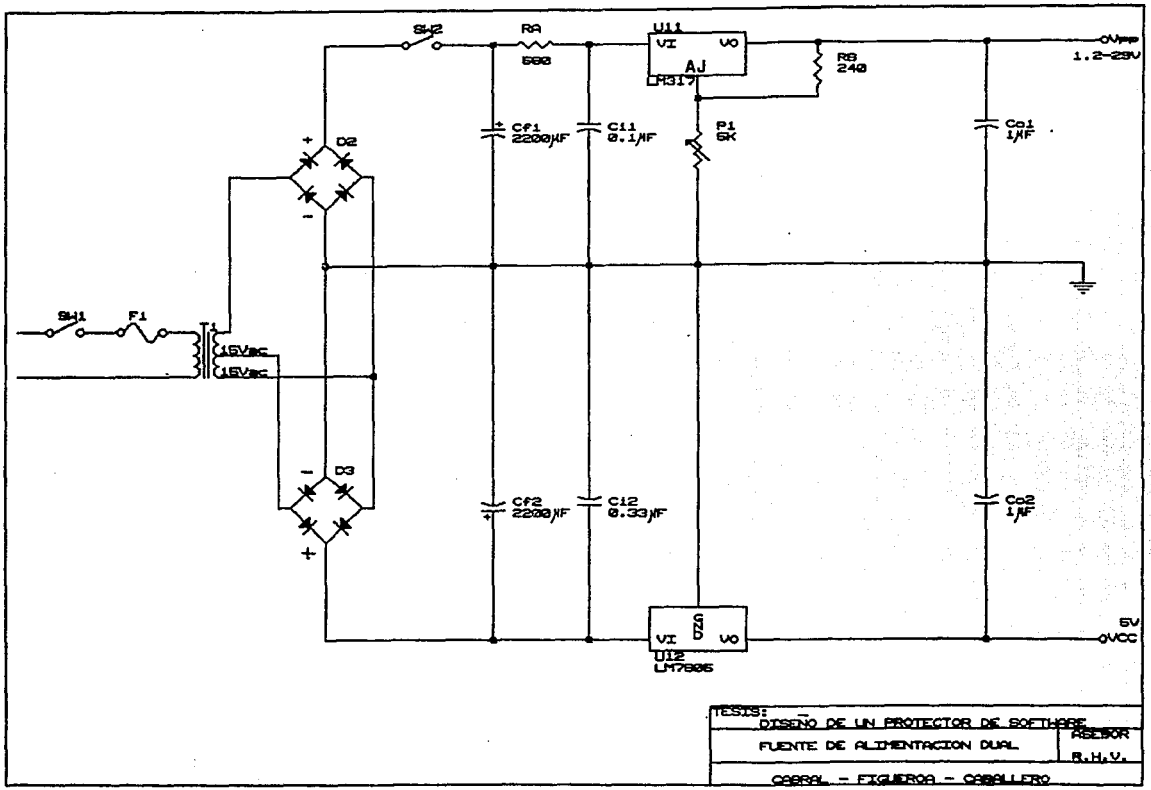

SW2
RA
560
U11
VI
VO
AJ
LM317
RB
240
Vpp
1.2-29V
Cf1
2200µF
C11
0.1µF
P1
5K
Co1
1µF
D2
SW1
F1
15Vac
15Vac
D3
Cf2
2200µF
C12
0.33µF
Co2
1µF
GND
VI
VO
U12
LM7806
5V
VCC
TESIS:
DISEÑO DE UN PROTECTOR DE SOFTWARE
FUENTE DE ALIMENTACION DUAL
ASESOR
R.H.V.
CABRAL - FIGUEROA - CABALLERO

FIG.2.9

2.3 SOFTWARE.

2.3.1 REQUISITOS.

Los objetivos que debe cumplir el programa que se diseñó para el Programador de EPROM son los siguientes:

- Visualizar los datos almacenados en la memoria en el rango de direcciones proporcionado por el usuario.

- Permitir que el usuario introduzca los datos que desea grabar proporcionando el rango de direcciones.

- Mostrar el dato que se grabó, para verificar que éste se grabó correctamente.

- Enviar al puerto paralelo los valores adecuados para que el Programador realice sus funciones en la secuencia requerida.

El diseño del programa se dividió en tres etapas: arquitectura, diagrama de estructura y descripción de procesos del programa.

2.3.2 ARQUITECTURA.

La arquitectura del sistema es la primera etapa en la fase de diseño. Aquí se muestran los diferentes niveles de abstracción del programa que permiten tener una visión global del mismo e iniciar un análisis más detallado subdividiéndolo en módulos.

La arquitectura contiene la descripción de los conjuntos de información y un diagrama de flujo de información.

Conjuntos de Información. El análisis de los conjuntos de información se divide en dos partes: las entradas y las salidas del sistema.

- ENTRADAS.

Las entradas que requiere el programa son tomadas del teclado, mediante el cual el usuario seleccionará opciones de menúes, oprimirá teclas específicas para tal objeto y responderá a preguntas sencillas para proporcionar los datos requeridos.

- SALIDAS

Las salidas generadas por el sistema serán enviadas a la pantalla, donde se desplegarán también los diversos mensajes que guían la operación del programa, como son: texto de instrucciones, menúes, mensajes de aviso y de error.

Diagrama de Flujo de Información. El diagrama de flujo de información muestra de manera global el recorrido y la transformación de la información a través del sistema. Este diagrama permite visualizar rápidamente los procesos del sistema así como los efectos que tienen sobre la información.

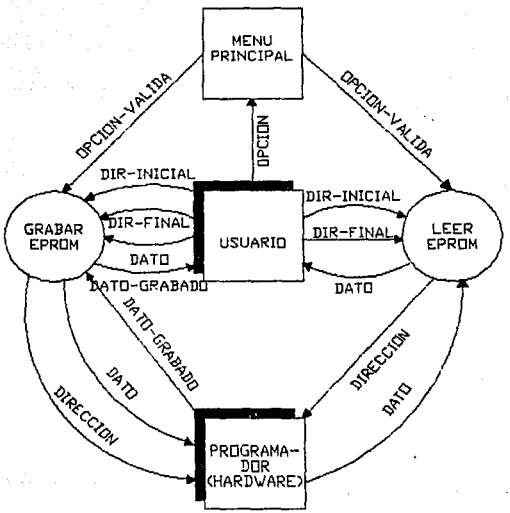


FIG. 2.10
DIAGRAMA DE FLUJO DE INFORMACION

2.3.3 DIAGRAMA DE ESTRUCTURA.

Este diagrama permite observar la jerarquía de cada uno de los módulos para comprender rápidamente el sistema. En la presente tesis se utilizó la técnica de dividir el programa

de aplicación en módulos, mediante un diagrama de estructura.

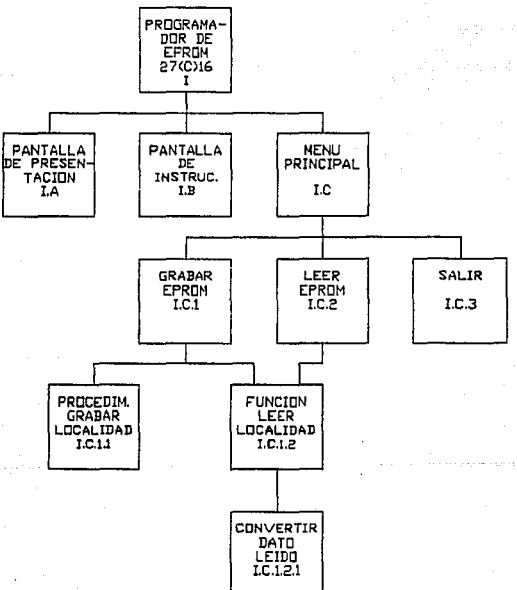


FIG. 2.11
DIAGRAMA DE ESTRUCTURA

2.3.4 DESCRIPCION DE PROCESOS DEL PROGRAMA.

Esta sección muestra el pseudocódigo de las rutinas principales del sistema con el propósito de que sea claro su funcionamiento y los desarrollos posteriores, basados en el programa, sean fáciles de incorporar.

- *I Programa principal.*

Reseña del proceso.

El programa principal del Programador de EPROM muestra la pantalla de presentación, la pantalla de instrucciones y el menú principal. Además inicializa las variables del sistema.

Detalle del proceso.

```
EPROM_27C16
INICIO
     S. Inicializar variables
     S. Desplegar pantalla de presentación
     S. Desplegar pantalla de instrucciones
     MIENTRAS (opción no es salir) HACER
          S. Desplegar menú principal (rutina MENUS)
     FIN DE MIENTRAS
FIN DE EPROM_27C16
```

- *I.A Pantalla de presentación.*

Reseña del proceso.

Mostrar una pantalla con la presentación del programa. Al oprimir cualquier tecla se sale de la misma.

- *I.B Pantalla de instrucciones.*

Reseña del proceso.

Mostrar una pantalla con las instrucciones para las conexiones previas a la operación del programa.

- *I.C Menú principal.*

Reseña del proceso.

Presentar las opciones del menú, capturar la opción seleccionada por el usuario y llamar a la rutina correspondiente.

Detalle del proceso.

```
MENUS
INICIO
     S. Desplegar menú de opciones
     S. Desplegar menú de teclas para elegir opción
     S. Leer opción de 1 a 3
     SI (opción es 1-Grabar) ENTONCES
          S. Llamar rutina GRABAR_EPROM
     SI (opción es 2-Leer) ENTONCES
          S. Llamar rutina LEER_EPROM
     SI (opción es 3-Salir) ENTONCES
          S. Llamar rutina SALIR
FIN DE MENUS
```

- *I.C.1 Grabar EPROM.*

Reseña del proceso.

Esta opción del menú se encarga de pedir y validar las

direcciones y los datos a grabar.

Detalle del proceso.

```
GRABAR_EPROM
INICIO
     S. Limpiar pantalla
     S. Pedir formato a trabajar (hexa o decimal)
     SI (tecla es A-Decimal) ENTONCES
          S. Trabajar en sistema decimal
     DE OTRA FORMA
          S. Trabajar en sistema hexa
     FIN DE SI
     MIENTRAS (ambas direcciones no sean correctas) HACER
          S. Pedir dirección inicial
          S. Pedir dirección final
     FIN DE MIENTRAS
     S. Inicializa variable DIREC con el valor de la
     dirección inicial
     MIENTRAS (dirección inicial ≤ dirección final y bandera
     CONTINUA prendida) HACER
          MIENTRAS (no se llene pantalla y direc. inicial ≤
          direc. final) HACER
               MIENTRAS (dato no sea válido) HACER
                    S. Pedir dato
               FIN DE MIENTRAS
               S. Mostrar dato en decimal, hexa, binario y
               ASCII
               S. Llamar procedimiento GRABA
               S. Hacer demora
               S. Llamar función LEE para comprobar
               grabación
               S. Mostrar dato grabado en decimal, hexa,
               binario y ASCII
               S. Incrementa variable DIREC en 1
```

```
        FIN DE MIENTRAS
        SI (DIREC ≤ direc. final) ENTONCES
            S. Prende bandera CONTINUA
        DE OTRA FORMA
            S. Apaga bandera CONTINUA
        FIN DE SI
    FIN DE MIENTRAS
FIN DE GRABAR EPROM
```

- *I.C.2 Leer EPROM.*

Reseña del proceso.

Esta opción del menú se encarga de pedir y validar las direcciones de los datos a leer.

Detalle del proceso.

```
LEER_EPROM
INICIO
    S. Limpiar pantalla
    S. Pedir formato a trabajar (hexa o decimal)
    SI (tecla es A-Decimal) ENTONCES
        S. Trabajar en sistema decimal
    DE OTRA FORMA
        S. Trabajar en sistema hexa
    FIN DE SI
    MIENTRAS (ambas direcciones no sean correctas) HACER
        S. Pedir dirección inicial
        S. Pedir dirección final
    FIN DE MIENTRAS
    S. Inicializa variable DIREC con el valor de la
    dirección inicial
    MIENTRAS (dirección inicial ≤ dirección final y bandera
    CONTINUA prendida) HACER
```

MIENTRAS (no se llene pantalla y direc. inicial ≤ direc. final) HACER
S. Llamar a la función LEE
S. Mostrar dato leído en decimal, hexa. binario y ASCII
S. Incrementa variable DIREC en 1
FIN DE MIENTRAS
SI (DIREC ≤ direc. final) ENTONCES
S. Prende bandera CONTINUA
DE OTRA FORMA
S. Apaga bandera CONTINUA
FIN DE SI
FIN DE MIENTRAS
FIN DE LEER EPROM

- *I.C.1.1 Procedimiento grabar localidad.*

Reseña del proceso.

Este procedimiento se encarga de grabar el dato en la dirección proporcionada. Recibe como entradas la dirección y el dato y no genera salidas.

Detalle del proceso.

GRABA
INICIO
S. Poner en estado alto las salidas del decodificador
S. Dividir la dirección en parte alta y parte baja
S. Colocar la parte baja en las líneas de datos
S. Bajar la señal ~DIREC1
S. Colocar parte alta de la dirección en las líneas de datos
S. Bajar la señal ~DIREC2
S. Colocar el dato en las líneas de datos

S. Bajar la señal -DATOS
S. Bajar la señal -SLCT IN generando el pulso de programación

FIN_DE_GRABA

- *I.C.1.2 Función leer localidad.*

Reseña del proceso.

Esta función lee un dato de la localidad proporcionada. Recibe como entrada la dirección y como salida entrega el dato.

Detalle del proceso.

LEE
INICIO
S. Poner en estado alto las salidas del decodificador
S. Dividir la dirección en parte alta y parte baja
S. Colocar la parte baja en las líneas de datos
S. Bajar la senal -DIREC1
S. Colocar parte alta de la dirección en las líneas de datos
S. Bajar la señal -DIREC2
S. Bajar la señal -LECTURA
S. Bajar la señal -LECT1
S. Leer la parte baja del dato
S. Llamar rutina DATO_REAL
S. Bajar la señal -LECT2
S. Leer la parte alta del dato
S. Llamar rutina DATO_REAL
S. Formar el dato a partir de las partes alta y baja
S. Asignar el dato a la función LEE
FIN_DE_LEE

- *I.C.1.2.1 Convertir el dato leído.*

Reseña del proceso.

Transforma el dato leído a su valor real. Cada dato es leído en dos mitades de cuatro bits cada una, sin embargo, el puerto paralelo lo recibe en los ocho bits del bus de datos. Esta rutina elimina los bits que no son utilizados e invierte el valor de BUSY.

Detalle del proceso.

DATO_REAL
INICIO
    S. Inicializa la variable SUMA a 0
    SI (el bit XD3-ERROR es diferente de 0) ENTONCES
        S. Agregar un 1 a la variable SUMA
    SI (el bit XD4-SLCT es diferente de 0) ENTONCES
        S. Agregar un 2 a la variable SUMA
    SI (el bit XD5-PE es diferente de 0) ENTONCES
        S. Agregar un 4 a la variable SUMA
    SI (el bit XD7-BUSY es igual a 0) ENTONCES
        S. Agregar un 8 a la variable SUMA
    S. El dato leído es igual a SUMA
FIN DE DATO_REAL

2.3.5 RUTINAS DE GRABACION Y LECTURA.

El programa se implementó en Turbo Pascal versión 5.0. Debido a que los módulos de grabar y leer una localidad de memoria son la parte esencial del programa, ya que son los que interactúan con los circuitos del Programador, se tratan con mayor detalle a continuación. Primeramente se verá como se determina la dirección base del puerto paralelo. La instrucción de Pascal

```
DIRBASE := MEMW[$0040:$0008];
```

tiene por objeto asignar a la variable DIRBASE el contenido de las localidades absolutas 00408H y 00409H, donde se encuentra dicha dirección.

Para facilitar el manejo de las direcciones de E/S del puerto se crearon las variables DIREDO (dirección de Estado) y DIRCTRL (dirección de Control), que se refieren a las direcciones de E/S para manejar las señales de Estado y de Control respectivamente. Estas variables se inicializan como se muestra:

```
DIREDO  := DIRBASE + 1;
DIRCTRL := DIRBASE + 2;
```

La rutina de grabación se implementó como procedimiento por ser una subrutina a la cual se le pasan dos parámetros, la dirección y el dato a grabar, y no produce salidas. Este procedimiento debe ser usado de la siguiente forma:

```
GRABA(DIREC,DATO);
```

Por su parte la rutina de lectura se implementó como función porque se le pasa un parámetro, la dirección, y entrega el valor del dato leído en esa dirección, el cual debe asignarse a una variable. Dicha función debe usarse como se indica:

```
DATO := LEE(DIREG);
```

Inicialmente, ambas rutinas envían un 0 por DIRCTRL para obligar a que las salidas del decodificador tomen un valor lógico alto. Se debe recordar que de las salidac de control del puerto, sólo -INIT conserva su valor y las demás

lo invierten. Por lo tanto, no es un 0 lo que recibe el decodificador, sino un 3. En este caso la salida 3 está en estado bajo, pero como no está conectada, se logra el efecto deseado de subir todas las salidas que sí son utilizadas por el circuito. Se puede consultar en la tabla de la figura 2.12 los valores enviados por el programa y los que realmente recibe el decodificador. Téngase en cuenta que -SLCT IN no esta conectada al decodificador sino que se usa para producir el pulso de programación.

VALOR ENVIADO POR EL PROG A TRAVES DE DIR. BASE+2				VALOR PRESENTE EN LAS SALIDAS DEL PUERTO				SALIDAS DEL DECODIFICADOR	
					ENTRADAS DECO.				
D	C	B	A	$\overline{\text{SLCT IN}}$	$\overline{\text{INIT}}$	$\overline{\text{AUTO}}$	$\overline{\text{STROBE}}$	No. SALIDA	SEÑAL
0	0	0	0	1	0	1	1	3	-
0	0	0	1	1	0	1	0	2	-DIREC1
0	0	1	0	1	0	0	1	1	-DIREC2
0	0	1	1	1	0	0	0	0	-DATOS
0	1	0	0	1	1	1	1	7	-LECTURA
0	1	0	1	1	1	1	0	6	-LECT1
0	1	1	0	1	1	0	1	5	-LECT2
1	0	1	1	0	0	0	0	0	-DATOS

FIG. 2.12
VALORES ENVIADOS POR EL PROGRAMA
PARA CONTROLAR EL DECODIFICADOR

Para poder efectuar la grabación y lectura de datos, como ya se mencionó, se tiene que dividir la dirección en dos partes. Para que la parte baja sea enviada (DIR0 a DIR7), se tiene que enviar un 1 usando DIRCTRL, con lo cual

baja la señal -DIREC1. A continuación, se envía un 2 también usando DIRCTRL para bajar la señal -DIREC2, que deja pasar la parte alta de la dirección (DIR8 a DIR10).

A partir de aquí las rutinas son diferentes y se explicará primero la de grabación. Después de enviar la dirección, el dato a grabar se coloca en las líneas de datos usando DIRBASE. Posteriormente se baja la señal -DATOS colocando un 3 en DIRCTRL para que el dato pase. Por último la señal -SLCT IN es puesta en nivel lógico bajo para producir el pulso de progamación y que el dato sea grabado. Para ello se envía un 11 por DIRCTRL con lo cual todas las salidas de Control son bajas, logrando que se produzca el pulso de programación y que al mismo tiempo -DATOS siga siendo baja.

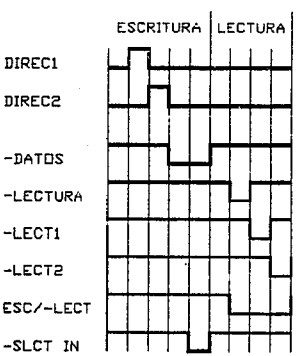


FIG. 2.13
DIAGRAMA DE TIEMPOS ESCRITURA-LECTURA

Para el caso de la función de lectura, después de enviar la dirección, se baja la señal -LECTURA enviando un 4 por DIRCTRL para tener el dato listo en las salidas de la EPROM. En seguida, la señal -LECT1 baja colocando un 5 en DIRCTRL, para después leer la parte baja del dato por DIREDO. La mitad leída del dato es transformada a su valor real. Para leer la mitad alta del dato, se coloca un 6 en DIRCTRL bajando -LECT2 y a continuación se lee el dato por DIREDO. Esta mitad también es transformada para obtener su valor real. Por útimo las dos mitades se unen para formar el dato leído.

El diagrama de tiempos de un ciclo completo escritura-lectura se muestra en la figura 2.13.

CAPITULO 3

PROTECTOR DE SOFTWARE

3.1 HARDWARE.

3.1.1 MODOS DE OPERACION DEL PROTECTOR.

El Protector de Software tiene dos modos de operación:

PROTECTOR.- Impide el uso no autorizado de un programa.

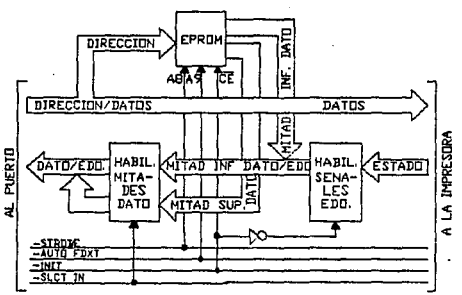


FIG. 3.1
DIAGRAMA DE BLOQUES DEL PROTECTOR

TRANSPARENTE.- Cuando no está actuando en modo Protector, el circuito debe funcionar de manera transparente para la impresora. En otras palabras, si se tiene una impresora conectada al puerto paralelo, el circuito no debe afectar el funcionamiento de la impresora.

Ambos modos de operación son mútuamente excluyentes, es decir, cuando el circuito funciona como protector de software, la impresora no puede ser utilizada y viceversa. Sin embargo, la impresora puede estar conectada, e incluso prendida, mientras el circuito está en modo Protector, sin que ésto altere su funcionamiento.

En la figura 3.1 se muestra el diagrama de bloques del Protector. Como se puede observar, las líneas de datos provenientes del puerto paralelo junto con dos de las señales de control proporcionan la dirección del dato a leer de la memoria. Las señales -Init y -Slct In controlan el circuito. Cada dato es leído en dos mitades a través de las señales de estado que vienen de la impresora.

El diagrama eléctrico del Protector se encuentra en la figura 3.2. El Protector está dotado de dos conectores DB-25; uno macho para conectarse al puerto paralelo y, en el otro extremo, uno hembra para conectar el cable de la impresora. En este caso se utilizaron circuitos C^2MOS por su baja disipación de potencia. Se explicará su funcionamiento analizando cada uno de los dos modos de operación por separado.

3.1.2 MODO PROTECTOR.

Durante la operación en modo Protector se debe accesar a los datos contenidos en la memoria 27C16, por tanto, ésta

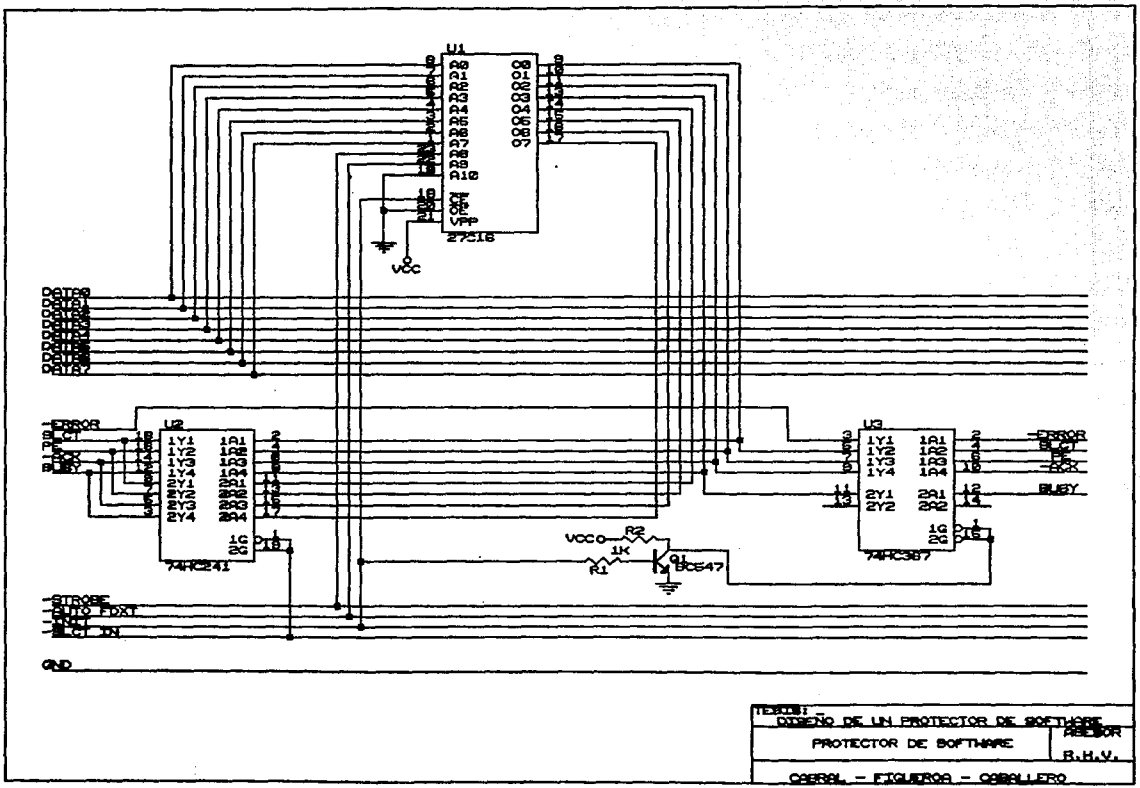

U1
27C16
VCC
U2
74HC241
U3
74HC367
R1
1K
R2
Q1
BC547
VCC
DATA0
DATA1
DATA2
ERROR
BUSY
ACK
STROBE
AUTO FDXT
GND
DISEÑO DE UN PROTECTOR DE SOFTWARE
PROTECTOR DE SOFTWARE
ASESOR
R.H.V.
CABRAL - FIGUEROA - CABALLERO

FIG. 3.2

se debe encontrar en modo lectura cumpliendo las siguientes condiciones:

\ PINES MODO \	-CE/PGM (18)	-OE (20)	Vpp (21)	Vcc (24)	SALIDAS (9-11,13-17)
LECTURA	V_{IL}	V_{IL}	+5	+5	DATOS DE SALIDA

Las entradas Vpp y Vcc de la memoria están conectadas a la fuente de alimentación. Los ocho bits menos significativos de la dirección a leer (A0 a A7) se proporcionan a través de las líneas de datos DATA0 a DATA7 y las señales -Strobe y -Auto FDXT forman los bits A8 y A9 respectivamente. La entrada A10 de la dirección no se utilizó y se conectó a tierra para fijar su valor a cero. De este modo, se utilizan diez de las once líneas de dirección de la memoria, dando un total de 2^{10} = 1024 bytes disponibles.

Para poder leer la memoria, la señal -Init, que está conectada a la entrada -CE/PGM de la memoria, debe ser baja. Asimismo, la entrada -OE debe ser baja para lo cual está conectada permanentemente a tierra. De este modo se tienen las condiciones necesarias para obtener el dato en las salidas O0 a O7 de la memoria.

Obsérvese que la señal -Init está conectada también a un transistor, el cual está configurado como inversor. La salida del mismo está conectada a las entradas -G1 y -G2 de un CI 74HC367 que contiene seis buffers con salidas Tres Estados. Debido a que -Init es baja para habilitar a la memoria, se tiene un valor alto en las entradas de habilitación -G1 y -G2 y por tanto, las señales de estado provenientes de la impresora están en alta impedancia. Así, los ocho bits del dato proveniente de la memoria llegan a un CI 74HC241 que contiene ocho buffers con salidas Tres

Estados separados en dos conjuntos de cuatro buffers cada uno. Las entradas de habilitación de cada conjunto son complementarias y están conectadas a la señal -Slct In.

Cuando -Slct In es baja, el primer conjunto interno de buffers estará habilitado y en las señales de estado Slct, PE, -Ack y Busy se tendrá el valor de los bits 00 a 03, que forman la mitad inferior del dato. Cuando -Slct In es alta, el segundo conjunto interno de buffers será habilitado y las señales Slct, PE, -Ack y Busy tendrán el valor de los bits 04 a 07, es decir, la otra mitad del dato. Por su parte, -Error proviene directamente de una salida del 74HC367 y su valor será alta impedancia, lo que se leerá como un 1 en el puerto. Por otra parte, las señales de datos y las de control están conectadas también a las salidas que van hacia la impresora.

3.1.3 MODO TRANSPARENTE.

Cuando el circuito opera en este modo, la computadora debe controlar a la impresora sin que la memoria interfiera en el proceso. Para ésto, la memoria se debe encontrar en el modo standby o "a la espera" durante el cual las salidas de la misma están en alta impedancia. Las condiciones a cumplirse son las siguientes:

\ PINES MODO \	-CE/PGM (18)	-OE (20)	Vpp (21)	Vcc (24)	SALIDAS (9-11,13-17)
STANDBY	V_{IH}	*	Vcc	+5	Z

* = no importa.

Ya se dijo que tanto Vpp como Vcc están conectadas a la alimentación permanentemente y A10 y -OE están conectadas a tierra. En este caso la señal -Init, que está conectada a la

entrada -CE/PGM, queda en estado lógico alto después de inicializar la impresora. Con esto se cumplen todas las condiciones para que la memoria opere en el modo standby. Ahora, no importan los valores que se tengan en las entradas de la dirección A0 a A10, las salidas O0 a O7 serán alta impedancia y no interferirán con las señales de estado que salen del CI 74HC367.

Debido a que -Init es alta, a las entradas -G1 y -G2 del 74HC367 les llega un nivel bajo porque -Init pasa primero por el transistor, el cual le invierte el valor. Por lo tanto los buffers internos están habilitados. Así los valores que provengan de la impresora por las líneas de estado -Error, Slct, PE, -Ack y Busy pasarán a través del 74HC367. Excepto -Error, las demás señales están conectadas después a uno de los conjuntos internos de cuatro buffers del CI 74HC241. Como la señal -Slct In queda en estado bajo después de inicializar la impresora, dicho conjunto de buffers permanece habilitado y permite también que las señales de estado pasen hacia la computadora. Las salidas del segundo conjunto interno de buffers quedará en alta impedancia sin afectar a las salidas del primero y permanecerán en ese estado mientras se trabaje con la impresora ya que -Slct In no cambiará de valor.

Así, en modo Transparente, el circuito permitirá el paso de las señales de estado desde la impresora hasta el puerto paralelo mientras -Init sea alta y -Slct In sea baja. Las señales de datos y de control, al estar conectadas directamente entre ambos conectores, trabajan normalmente como si no estuviera presente el Protector.

La figura 3.3 muestra los valores que deben tomar las señales -Init y -Slct In para controlar al Protector de Software en los dos modos de operación. Durante el modo

Protector -Init y -Slct In están en estado lógico bajo, esto es como el proceso de inicialización de la impresora pero con una duración mayor. Por lo tanto, aunque la impresora esté conectada y encendida, no se altera su funcionamiento. Durante el modo Transparente -Init está en estado alto y -Slct In permanece en estado bajo. Para pasar del modo Protector al modo Transparente, basta con inicializar la impresora nuevamente.

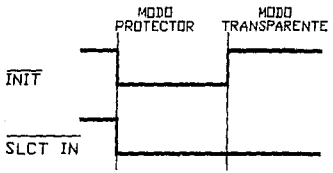


FIG. 3.3
CONTROL DEL PROTECTOR DE SOFTWARE

3.1.4 FUENTE DE ALIMENTACION.

Se plantearon cuatro alternativas para alimentar el circuito del Protector de Software:

a) Utilizando alguna de las señales de datos DATA0 a DATA7. Esto no fue posible porque al operar en modo Transparente, por las líneas de datos se transmiten códigos ASCII, secuencias de escape y códigos de control hacia la impresora y por lo tanto los valores contenidos en estas líneas cambian continuamente. Al usar alguna de estas líneas como alimentación puede perderse su valor en cualquier momento y cortar el suministro de energía.

b) Utilizando una de las señales de control. También se descartó esta opción porque -Init y -Slct In se ocuparon para el control del Protector. De haberse utilizado -Strobe o -Auto FDXT se hubiera reducido el número de bytes disponibles de la memoria, de la mitad a una cuarta parte, porque se utilizarían solamente nueve de las diez entradas restantes de dirección, esto es, se tendría un máximo de 2^9 = 512 localidades de memoria disponibles.

c) Utilizando alguna señal de estado, en particular se seleccionó -Error por ser la señal que menos variaciones tiene. Esto no dió resultados satisfactorios debido a que, en primer lugar, la impresora tenía que estar conectada y encendida en todo momento. En segundo lugar, al conectar la señal a todo el circuito, la caída de voltaje producida provocaba que el voltaje no fuera suficiente para alimentar al circuito.

d) Usar una fuente de alimentación independiente.

Se determinó que esta última era la mejor forma porque es independiente de la operación del puerto paralelo y de la impresora. Como los circuitos CMOS tienen un bajo consumo de potencia, se decidió alimentar el circuito con una pila de litio de 3 volts, voltaje suficiente para el funcionamiento de los circuitos que componen el Protector. Se eligió la pila de las llamadas tecnología de botón.

Las prestaciones de la pila de litio son mucho mejores que las de las pilas salinas y alcalinas clásicas. Su autodescarga, en condiciones de almacenamiento es muy débil. Por término medio, la perdida de capacidad de una pila de litio almacenada no excede del 3% anual, mientras que es

superior al 5% anual en las pilas alcalinas clásicas y está comprendida entre el 10% y el 15% en una pila salina. Todos los elementos integrantes de estas pilas se encuentran en Asia y los fabricantes suministran importantes cantidades de pilas tipo botón a precios extremadamente bajos, por lo que estas pilas equipan ya a un cierto número de calculadoras, de relojes y de cámaras fotográficas y de video.

3.2 SOFTWARE.

3.2.1 REQUISITOS.

El programa que se diseñó para el Protector de Software tiene por objeto impedir el uso no autorizado de otro programa denominado en la presente tesis programa a proteger. Para ello, debe cumplir los siguientes requisitos:

- Durante el modo Protector debe accesar a la memoria para leer la información contenida en ella.

- Durante el modo Transparente debe poner la memoria en modo standby para no interferir con la operación del puerto paralelo y la impresora.

Para operar en el modo Transparente, como ya se explicó en la parte dedicada al hardware, basta con inicializar la impresora y las señales de control quedarán en los valores adecuados para que la memoria quede en modo standby. El modo Protector, por el contrario, es la parte medular del Protector de Software y se explica a continuación.

3.2.2 MODULOS DEL PROGRAMA.

Módulo Verificar Clave. La idea básica para impedir que se haga uso de un programa para el cual no se tiene autorización es la siguiente: al comienzo del programa a proteger se inserta un módulo encargado de leer una clave que se encuentra en la memoria EPROM y verificar que coincida con la que se encuentra en el módulo. A este módulo se le denominó Verificar Clave y debe leer ciertas localidades de la memoria, de donde extraerá los números que forman la clave. En otras palabras, se puede pensar en la clave como un número de serie único para cada copia

distribuida del programa a proteger y para su respectivo Protector de Software. Tanto el programa a proteger como el Protector, contienen la clave o número de serie y deben coincidir desde el inicio para poder continuar con la utilización del programa. Si no se encuentra conectado el Protector o en el momento que un número de la clave leída no coincida, se generará un mensaje de error y se abortará la ejecución del programa.

Módulo Leer Programa. Para mayor protección, se implementó un segundo módulo al cual se le denominó Leer Programa y consiste en lo siguiente. Del programa a proteger se quita alguna parte que sea importante, sin la cual el programa no funcione o no pueda ejecutar alguna de las funciones para lo que fue creado. Esa parte se deja "en blanco" llenando todos los bytes que la componen con el valor 90H, que representa el código de la instrucción NOP o No Operación. En realidad NOP no realiza operaciones y es una instrucción de un byte que ocupa espacio pero no afecta el funcionamiento del programa excepto al registro (E)IP, que es el apuntador de instrucciones y se incrementa en uno.

El código verdadero que debería ocupar esta parte del programa se encuentra en la memoria 27C16 del Protector de Software y al momento de la ejecución del programa, el módulo Leer Programa se encarga de traer uno a uno los bytes desde la memoria EPROM y los coloca en el lugar que les corresponde, sobre los códigos NOP del programa. En otras palabras, es como si el programa a proteger tuviera un "hueco" que debe ser llenado desde la memoria EPROM en tiempo de ejecución. Si el Protector de Software no se encuentra conectado, la parte faltante será llenada con "basura" y al llegar el IP a la misma, el programa se detendrá por no entender el procesador el significado de esos códigos. De esta forma, si una persona trata de hacer

uso del programa sin tener conectado el Protector de Software correspondiente, el programa no funcionará porque simplemente no está completo.

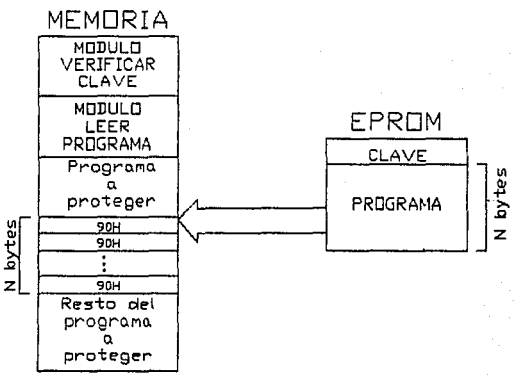


FIG. 3.4
MODULOS DEL PROGRAMA

3.2.3 DISEÑO DEL PROGRAMA.

Al igual que con el Programador de EPROM, el diseño del programa se dividió en tres etapas: arquitectura, diagrama de estructura y descripción de procesos del programa.

Arquitectura. Se divide a su vez en descripción de los conjuntos de información y el diagrama de flujo de información, los cuales permiten tener una visión global del sistema.

Conjuntos de Información.

- Entradas. En este caso no hay entradas.

- Salidas. La única salida se produce cuando no se ha autorizado la utilización del programa a proteger y consiste en un mensaje enviado a la pantalla.

Diagrama de Flujo de Información. Muestra de manera global la transformación de la información a través del sistema.

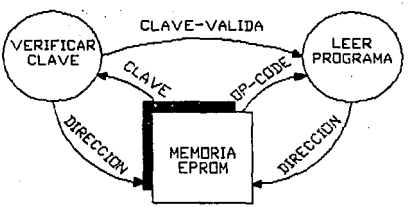


FIG. 3.5
DIAGRAMA DE FLUJO DE INFORMACION

Diagrama de Estructura. Permite observar la jeraquía de cada módulo.

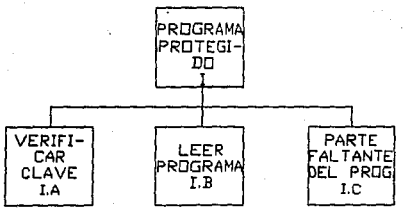


FIG. 3.6
DIAGRAMA DE ESTRUCTURA

Descripción de Procesos del Programa. Aquí se muestra el pseudocódigo de los módulos que forman el programa.

- *I Programa protegido.*

Reseña del proceso.

Al programa original se le añadieron los módulos Verificar Clave y Leer Programa que hacen que el circuito trabaje en modo Protector. Se inicializa después la impresora para trabajar en modo Transparente y ejecutar lo que resta del programa a proteger.

Detalle del proceso.

PROGRAMA_PROTEGIDO
INICIO
S. Checar que exista puerto paralelo y obtener su dirección base

S. Ejecutar VERIFICAR_CLAVE
S. Ejecutar LEER_PROGRAMA
S. Inicializar impresora
S. Ejecutar parte restante del programa
FIN_DE_PROGAMA_PROTEGIDO

- *I.A Verificar clave.*

Reseña del proceso.

Verificar que la clave contenida en el programa sea igual a la del Protector de Software, contenida en la memoria EPROM. De lo contrario se aborta la ejecución del programa.

Detalle del proceso.

VERIFICAR_CLAVE
INICIO
MIENTRAS (no se termine de leer la clave y ésta coincida con la del programa) HACER
S. Enviar por las líneas de datos la parte baja de la dirección (A0 a A7)
S. Enviar ceros por las líneas de control: -Strobe y -Auto FDXT para completar la parte alta de la dirección (A8 y A9), -Init para habilitar la EPROM y -Slct In para que pase la parte baja del dato a leer por el 74HC241
S. Leer por las líneas de estado la parte baja del dato (O0 a O3)
S. Invertir el valor de Busy para corregir el dato leído
S. Enviar por las líneas de control -Strobe, -Auto FDXT e -Init ceros y por -Slct In un 1 para que pase la parte alta del dato leído por el 74HC241

S. Leer por las líneas de estado la parte alta del dato (O4 a O7)
S. Invertir el valor de Busy para corregir el dato leído
S. Juntar las dos mitades del dato, para formar un dígito de la clave
SI (dato leído no es igual a la clave del programa) ENTONCES
S. Prender bandera ERROR
FIN DE MIENTRAS
SI (bandera ERROR está prendida) ENTONCES
S. Enviar mensaje de acceso negado
S. Inicializar impresora
S. Abortar la ejecución del programa
FIN DE SI
FIN DE VERIFICAR_CLAVE

- *I.B Leer programa.*

Reseña del proceso.

Leer la parte faltante del programa desde la memoria EPROM 27C16.

Detalle del proceso.

LEER_PROGRAMA
INICIO
MIENTRAS (no se terminen las localidades a leer) HACER
S. Enviar por las líneas de datos la parte baja de la dirección (A0 a A7)
S. Enviar por las líneas de control -Strobe y -Auto FDXT la parte alta de la dirección (A8 y A9). Enviar un 0 por -Init para habilitar la EPROM así como por -Slct In para que pase la parte baja del

dato a leer por el CI 74HC241
S. Leer por las líneas de estado la parte baja del dato (00 a 03)
S. Invertir el valor de Busy para corregir el dato leído
S. Enviar por la línea de control -Slct In un 1 para que pase la parte alta del dato leído por el 74HC241, sin modificar los valores de -Strobe, -Auto FDXT e -Init colocados anteriormente
S. Leer por las líneas de estado la parte alta del dato (04 a 07)
S. Invertir el valor de Busy para corregir el dato leído
S. Juntar las dos mitades del dato, para formar un op-code o código de operación
S. Mover el dato a su posición en el programa

FIN DE MIENTRAS
FIN DE LEER PROGRAMA

A diferencia del módulo Verificar Clave, el módulo Leer Programa puede accesar a todas las localidades de la memoria para el caso en que todas éstas sean ocupadas. Por ello los valores enviados por las líneas de datos y de control cambian de acuerdo a la dirección que se desea leer y si se trata de la parte baja o alta del dato, como se muestra en la siguiente tabla:

DIRECCION (DIR)	MITAD A LEER	VALOR A ENVIAR POR: DIRCTRL	DIRBASE
0 - 255	BAJA	0BH	DIR
	ALTA	03H	DIR
256 - 511	BAJA	0AH	DIR-256 (100H)
	ALTA	02H	DIR-256
512 - 767	BAJA	09H	DIR-512 (200H)
	ALTA	01H	DIR-512
768 - 1023	BAJA	08H	DIR-768 (300H)
	ALTA	00H	DIR-768

CAPITULO 4

PRUEBAS E IMPLEMENTACION

4.1 PRUEBAS DE PROTOTIPOS.

Para comprender mejor el funcionamiento del puerto paralelo, se implementaron los circuitos mostrados en la figura 4.1 compuestos por los circuitos integrados TIL 111. Estos fotoacopladores protegen al puerto paralelo de algún corto circuito que se llegara a cometer. Se sugiere utilizarlos si se desea experimentar con el puerto.

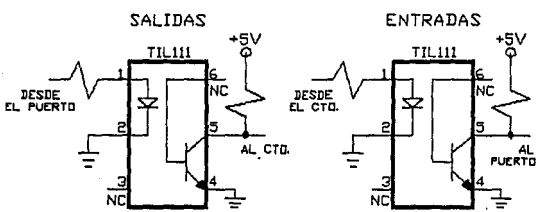


FIG. 4.1
CIRCUITOS DE PROTECCION

Después de la fase del diseño de los dispositivos, tanto el Protector de Software como el Programador de EPROM se montaron en tabletas de prueba de prototipos conocidas como Protoboard. Estas tabletas permiten montar los componentes fácilmente sin utilizar soldadura ya que consisten de una serie de orificios sobre una base en los cuales se enchufan con firmeza los componentes. Los orificios actúan como conductores ya que están dotados de unas pinzas metálicas de manera que cada grupo de cinco agujeros está conectado eléctricamente. Con esta matriz es fácil trasladar los diseños de circuitos a la tableta, utilizando trozos cortos de cable para conectar grupos separados de agujeros.

De la experiencia obtenida durante el desarrollo de esta tesis se recomienda ampliamente utilizar LEDs para monitorear el estado de las principales señales y así poder detectar fallas más fácilmente. De este modo, por ejemplo, se descubrió que ruido en las salidas del CI 74LS244 del Programador de EPROM (que sirve para leer las dos mitades del dato), estaba provocando que el dato se perdiera antes de poder leer su valor. En los LEDs de las salidas se observó que por un instante el dato estaba presente pero se perdía. El problema se solucionó conectando capacitores de tantalio en las salidas de dicho CI.

Otro problema causado por ruido en el Programador de EPROM involucraba las salidas del decodificador 74LS155. En este caso el problema era que el dato a grabar y la dirección correspondiente se mezclaban perdiéndose tanto uno como el otro. También los LEDs ayudaron a percibir que momentáneamente los valores de estas líneas eran correctos pero después se modificaban. Con ayuda de un osciloscopio se determinó que el problema era de ruido y se colocaron capacitores en cada una de las salidas del decodificador.

4.2 IMPLEMENTACION EN CIRCUITOS IMPRESOS.

Se decidió dividir el circuito del Programador de EPROM en tres placas independientes: una que contiene la fuente de alimentación, otra en la cual se encuentran los circuitos integrados necesarios para la programación de la memoria, y por último la placa que contiene una base para la memoria EPROM 27C16. Esto permite trabajar con el concepto de modularidad ya que si, por ejemplo, la fuente se descompone, es posible cambiarla sin reemplazar todo el circuito. Para el caso de la base de la EPROM, permite que el montaje se efectúe en la parte superior del chasís mientras que las otras dos placas se encuentran en el fondo del mismo.

Por su parte el Protector de Software consiste de una sola placa montada en una caja pequeña para ser conectada atrás de la computadora, si el espacio lo permite. De lo contrario puede utilizarse el cable provisto para tal efecto. En ambos casos, en la parte posterior del Protector de Software se conecta el cable convencional que va a la impresora.

Para el diseño de cada placa se siguieron los pasos que se muestran en el diagrama de flujo de la figura 4.2.

El proceso de trazado de las placas se realizó utilizando el paquete SMARTWORK. Las operaciones que pueden realizarse utilizando este sistema se resaltan en la figura 4.2 con líneas dobles. Como se ve en la figura, es frecuente que el proceso de diseño tenga que ser repetido a veces incluso desde varios pasos atrás. El programa permite la rápida y precisa revisión y modificación del trabajo previo para obtener circuitos impresos de calidad profesional. Sin embargo, para el caso de los conectores DB-25 y Centronics no fue posible obtener la separación necesaria para las

marcas de las perforaciones de los pines, por lo que éstas tuvieron que dibujarse a mano.

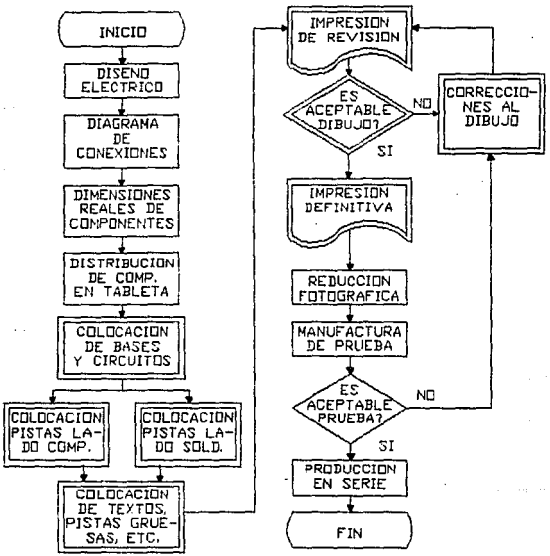
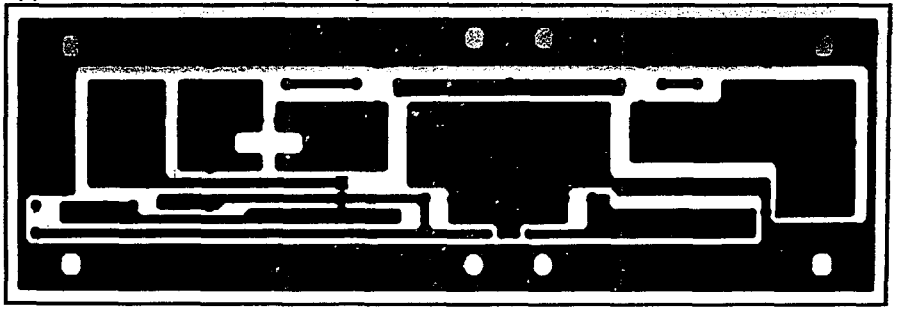


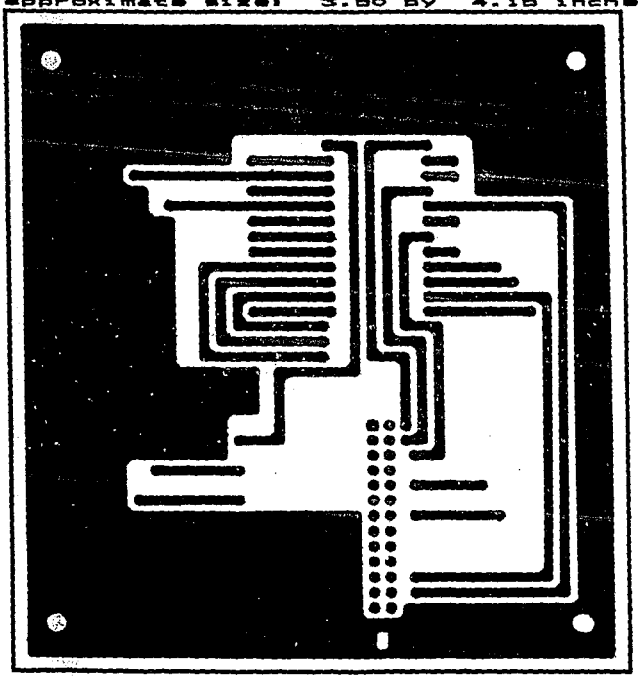
FIG. 4.2
DIAGRAMA DE FLUJO DEL DISEÑO Y
FABRICACION DE CIRCUITOS IMPRESOS

En las páginas siguientes se incluyen los diagramas de pistas de cada una de las placas de circuitos impresos.

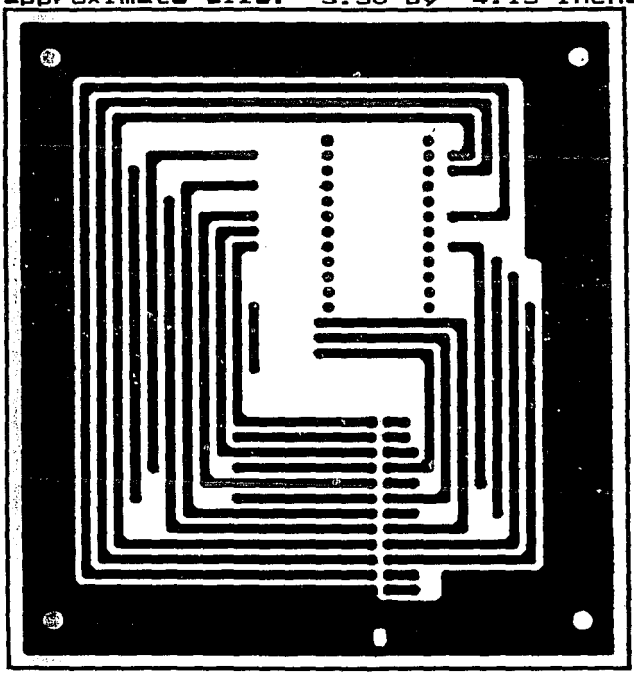
2X artwork 6 Mar 91 14:43:30
B-25.pcb
v1.3 F2 holes: 43 solder side
approximate size: 6.15 by 1.80 inches

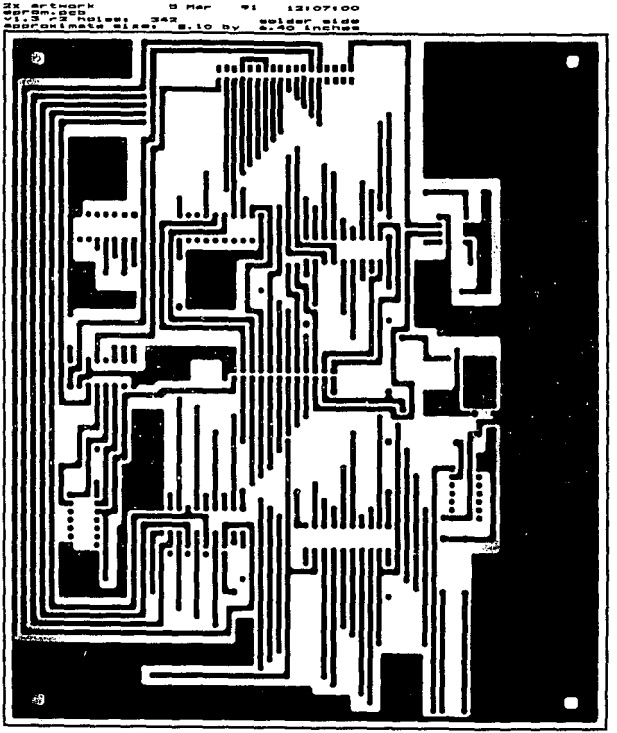


ZX artwork 5 Mar 91 09:19:34
case.pcb
v1.3 r2 holes: 84 solder side
approximate size: 3.80 by 4.15 inches

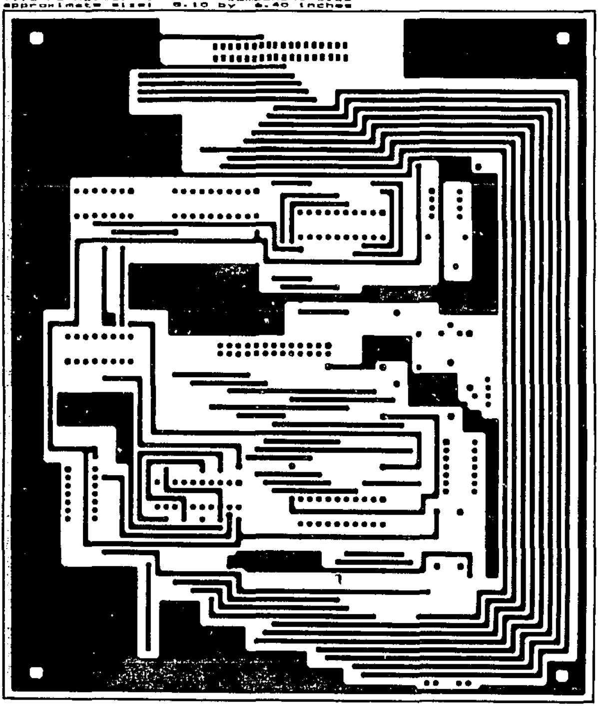


2X artwork 5 Mar 91 09:14:34
base.pcb
v1.3 r2 holes: 84 component side
approximate size: 3.50 by 4.15 inches

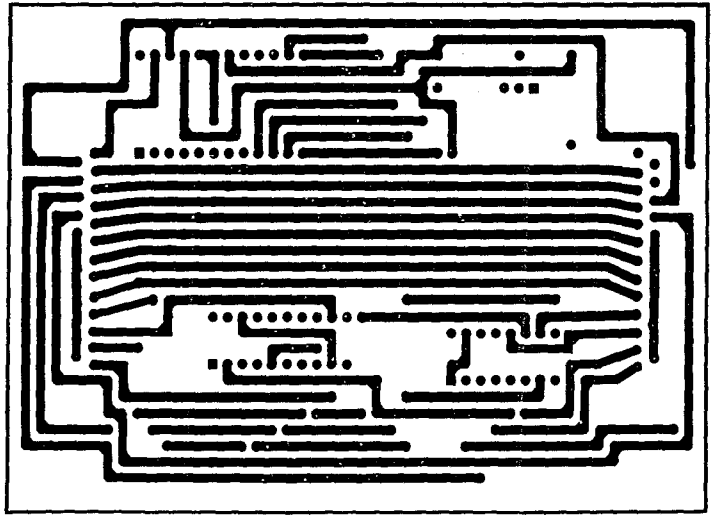



2X artwork 8 Mar 91 13:07:00
eprom.pcb
v1.3 42 holes 342
approximate size: 8.10 by 6.40 inches
solder side

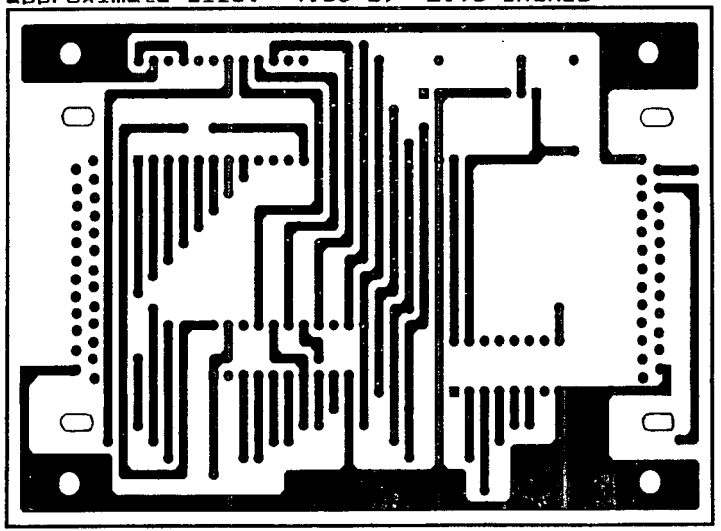
2X artwork 5 Mar 91 15:24:42
eprom.pcb
v1.3 r2 holes: 342 component side
approximate size: [illegible] by 6.40 inches



2X artwork 13 Dec 91 13:45:18
vigila.pcb
v1.3 r2 holes: 120 solder side
approximate size: 4.50 by 2.95 inches



2X artwork 13 Dec 91 13:50:08
vigila.pcb
v1.3 r2 holes: 120 component side
approximate size: 4.50 by 2.95 inches



CONCLUSIONES

UNA EXPERIENCIA REAL DE DISEÑO EN MEXICO.

Se considera que el objetivo principal del trabajo se ha logrado, dado que se ha podido diseñar y construir un dispositivo protector de software, utilizando tecnología totalmente nacional, abarcando desde el planteamiento de la idea básica, su desarrollo técnico, fabricación y pruebas de funcionamiento.

Todo el material con el que se fabricó el dispositivo, incluyendo los circuitos integrados seleccionados, las tabletas para las pruebas preliminares de circuitos, los componentes electrónicos tales como transistores, capacitores, etc., se encuentran disponibles en el mercado nacional. La elaboración de las placas de circuitos impresos y del chasís para montar los dispositivos fueron hechas por empresas mexicanas que cuentan con la tecnología propia adecuada.

El trabajo de diseño realizado, tanto del Programador de EPROM como del Protector de Software, lo hemos podido lograr gracias a las enseñanzas básicas que recibimos durante nuestros estudios en la Facultad de Ingeniería de la Universidad Nacional Autónoma de México, lo que nos permitió

analizar y comprender toda la información especializada que tuvimos que consultar en los libros que se mencionan en la Bibliografía y que también se encuentran disponibles en el país y que complementaron los conocimientos que aplicamos para llevar a buen término nuestro objetivo, sin necesidad de recurrir a ninguna tecnología exterior.

El Protector de Software, independientemente del Programador de EPROM, se calcula que una vez fabricado en serie en forma industrializada, tendrá un precio de venta de $103,000.00 por unidad, precio que consideramos bastante asequible para proteger software cuya importancia así lo requiera.

Por su parte, el Programador de EPROM, con las mismas condiciones de fabricación anteriores, tendría un precio de $474,000.00 por unidad, con la aclaración de que de este dispositivo, el fabricante de software únicamente tendría que utilizar una unidad, misma que le serviría para codificar todos los Protectores de Software que vendiera.

En la actualidad, el avance tecnológico de nuestro país se ampara mucho en la tecnología extranjera, prefiriendo pagar derechos de uso y de patente a desarrollar nuestra propia tecnología. La gran necesidad que tiene nuestro país de soluciones tecnológicas propias, aún para los problemas más sencillos, se ve ampliada con la próxima celebración del Acuerdo de Libre Comercio que firmará nuestro país con Canadá y Estados Unidos de Norteamérica, lo que nos obliga a ser altamente competitivos para tener posibilidades de entrar en el mercado de esos países y ésto sólo lo podemos lograr desarrollando nuestras propias capacidades.

Haciendo un análisis de todo lo anterior junto con los resultados obtenidos, se concluye que si tenemos el firme

propósito de lograr un objetivo, podemos hacerlo utilizando nuestros propios conocimientos y recursos disponibles sin necesidad de recurrir a tecnologías extranjeras, consiguiendo productos de fabricación nacional que permitirán satisfacer nuestras necesidades y en muchos casos competir con el extranjero.

APENDICE **A**

HOJAS DE ESPECIFICACIONES

Se anexan las hojas de especificaciones de los principales circuitos integrados utilizados.

National Semiconductor

NMC27C16
16,384-Bit (2048 x 8) UV Erasable CMOS PROM

General Description

The NMC27C16 is a high speed 16k UV erasable and electrically reprogrammable CMOS EPROM, ideally suited for applications where fast turnaround, pattern experimentation and low power consumption are important requirements.

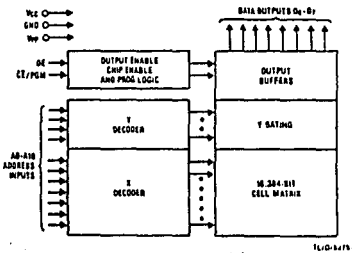
The NMC27C16 is packaged in a 24-pin dual-in-line package with transparent lid. The transparent lid allows the user to expose the chip to ultraviolet light to erase the bit pattern. A new pattern can then be written into the device by following the programming procedure.

This EPROM is fabricated with the reliable, high volume, time proven, P²CMOS™ silicon gate technology.

Features

- Access time down to 300 ns
- Low CMOS power consumption
  - Active Power: 28.25 mW max
  - Standby Power: 0.53 mW max (98% savings)
- Performance compatible to NSC800™ CMOS microprocessor
- Single 5V power supply
- Extended temperature range available (NMC27C16E-45). −40°C to +85°C, 450 ns ±5% power supply
- Pin compatible to MM2716 and higher density EPROMs
- Static—no clocks required
- TTL compatible inputs/outputs
- TRI-STATE® output

Block Diagram



TL/D/5475-1

Pin Names

A0–A10	Addresses
CE	Chip Enable
OE	Output Enable
O0–O7	Outputs
PGM	Program
NC	No Connect

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Temperature Under Bias	−10°C to +80°C
Storage Temperature	−65°C to +125°C
All Input Voltages with Respect to Ground	+6.5V to −0.3V
All Output Voltages with Respect to Ground (Note 11)	V_{CC} +0.3V to GND−0.3V
V_{PP} Supply Voltage with Respect to Ground During Programming	+26.5V to −0.3V
Power Dissipation	1.0W
Lead Temperature (Soldering, 10 seconds)	300°C

Operating Conditions (Note 9)

Temperature Range	
NMC27C16-30, -35, -45, -55	0°C to +70°C
NMC27C16E-45	−40°C to +85°C
V_{CC} Power Supply (Notes 2 and 3)	5V ±5%
V_{PP} Power Supply (Note 3)	V_{CC}

READ OPERATION

DC Electrical Characteristics

Symbol	Parameter	Conditions	Min	Typ (Note 4)	Max	Units
I_{LI}	Input Load Current	$V_{IN} = V_{CC}$ or GND			10	μA
I_{LO}	Output Leakage Current	$V_{OUT} = V_{CC}$ or GND, $\overline{CE} = V_{IH}$			10	μA
I_{CC1} (Note 3)	V_{CC} Current (Active) TTL Inputs	$\overline{OE} = \overline{CE} = V_{IL}$, f = 1 MHz Inputs = V_{IH} or V_{IL}, I/O = 0 mA		2	10	mA
I_{CC2} (Note 3)	V_{CC} Current (Active) CMOS Inputs	$\overline{OE} = \overline{CE} = V_{IL}$, f = 1 MHz Inputs = V_{CC} or GND, I/O = 0 mA		1	5	mA
I_{CCSB1}	V_{CC} Current (Standby) TTL Inputs	$\overline{CE} = V_{IH}$		0.1	1	mA
I_{CCSB2}	V_{CC} Current (Standby) CMOS Inputs	$\overline{CE} = V_{CC}$		0.01	0.1	mA
V_{IL}	Input Low Voltage		−0.1		0.8	V
V_{IH}	Input High Voltage		2.0		V_{CC} + 1	V
V_{OL1}	Output Low Voltage	I_{OL} = 2.1 mA			0.45	V
V_{OH1}	Output High Voltage	I_{OH} = −400 μA	2.4			V
V_{OL2}	Output Low Voltage	I_{OL} = 0 μA			0.1	V
V_{OH2}	Output High Voltage	I_{OH} = 0 μA	V_{CC} − 0.1			V

AC Electrical Characteristics

Symbol	Parameter	Conditions	NMC27C16								Units
			-30		-35		E-45, -45		-55		
			Min	Max	Min	Max	Min	Max	Min	Max	
t_{ACC}	Address to Output Delay	$\overline{CE} = \overline{OE} = V_{IL}$		300		350		450		550	ns
t_{CE}	$\overline{CE}$ to Output Delay	$\overline{OE} = V_{IL}$		300		350		450		550	ns
t_{OE}	$\overline{OE}$ to Output Delay	$\overline{CE} = V_{IL}$		120		120		120		150	ns
t_{DF}	$\overline{OE}$ High to Output Float	$\overline{CE} = V_{IL}$	0	100	0	100	0	100	0	100	ns
t_{OH} (Note 5)	Output Hold from Addresses, $\overline{CE}$ or $\overline{OE}$, Whichever Occurred First	$\overline{CE} = \overline{OE} = V_{IL}$	0		0		0		0		ns

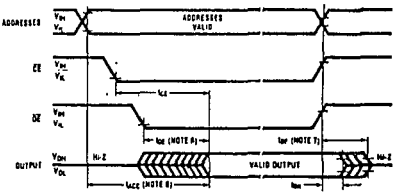
Capacitance T_A = +25°C, f = 1 MHz (Note 5)

Symbol	Parameter	Conditions	Typ	Max	Units
C_{IN}	Input Capacitance	V_{IN} = 0V	4	8	pF
C_{OUT}	Output Capacitance	V_{OUT} = 0V	8	12	pF

AC Test Conditions

Output Load	1 TTL Gate and C_L = 100 pF	Timing Measurement Reference Level	
		Inputs	1V and 2V
Input Rise and Fall Times	≤ 20 ns	Outputs	0.8V and 2V
Input Pulse Levels	0.8V to 2.2V		

AC Waveforms (Notes 2, 8, 9, 10)



TL/D/8478–3

Note 1: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Note 2: V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}.

Note 3: V_{PP} may be connected to V_{CC} except during programming. I_{CC1} ≤ the sum of the I_{CC} active and I_{PP} read currents.

Note 4: Typical values are for T_A = +25°C and nominal supply voltages.

Note 5: This parameter is only sampled and is not 100% tested.

Note 6: $\overline{OE}$ may be delayed up to $t_{ACC} - t_{OE}$ after the falling edge of $\overline{CE}$ without impact on t_{ACC}.

Note 7: The I_{DF} compare level is determined as follows:
High to TRI-STATE, the measured V_{OH1} (DC) = 0.10V
Low to TRI-STATE, the measured V_{OL1} (DC) = 0.10V

Note 8: TRI-STATE may be attained using $\overline{OE}$ or $\overline{CE}$.

Note 9: The power switching characteristics of EPROMs require careful device decoupling. It is recommended that a 0.1 μF ceramic capacitor be used on every device between V_{CC} and GND.

Note 10: The NMC27C16 requires one address transition after initial power-up to reset the outputs.

Note 11: The outputs must be restricted to V_{CC} + 0.3V to avoid latch-up and device damage.

PROGRAMMING CHARACTERISTICS (Note 1)

DC Programming Characteristics (Notes 2 & 3)

(T_A = +25°C ±5°C, V_{CC} = 5V ±5%, V_{PP} = 25V ±1V)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{LI}	Input Current (for Any Input)	V_{IN} = V_{CC} or GND			10	µA
I_{PP}	Vpp Supply Current During Programming Pulse	$\overline{CE}$/PGM = V_{IH}			30	mA
I_{CC}	V_{CC} Supply Current				10	mA
V_{IL}	Input Low Level		−0.1		0.8	V
V_{IH}	Input High Level		2.0		V_{CC} + 1	V

AC Programming Characteristics (Notes 2 & 3)

(T_A = +25°C ±5°C, V_{CC} = 5V ±5%, V_{PP} = 25V ±1V)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
t_{AS}	Address Setup Time		2			µs
t_{OES}	$\overline{OE}$ Setup Time		2			µs
t_{DS}	Data Setup Time		2			µs
t_{AH}	Address Hold Time		2			µs
t_{OEH}	$\overline{OE}$ Hold Time		2			µs
t_{DH}	Data Hold Time		2			µs
t_{DF}	Output Enable to Output Float Delay	$\overline{CE}$/PGM = V_{IL}	0		150	ns
t_{OE}	Output Enable to Output Delay	$\overline{CE}$/PGM = V_{IL}			150	ns
t_{PW}	Program Pulse Width		45	50	55	ms
t_{PRT}	Program Pulse Rise Time		5			ns
t_{PFT}	Program Pulse Fall Time		5			ns

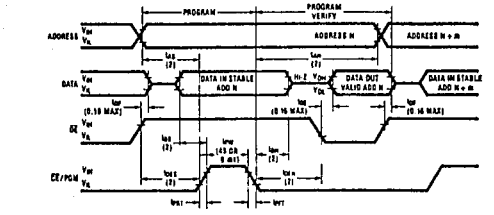
AC Test Conditions

V_{CC}	5V ±5%
V_{PP}	25V ±1V
Input Rise and Fall Times	≤20 ns
Input Pulse Levels	0.8V to 2.2V

Timing Measurement Reference Level

Inputs	1V and 2V
Outputs	0.8V and 2V

Programming Waveforms (Note 3) V_{PP} = 25V ±1V, V_{CC} = 5V ±5%



Note: All times shown in parentheses are minimum and in µs unless otherwise specified.

Note 1: National's standard product warranty applies only to devices programmed to specifications described herein.

Note 2: V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}. The NMC27C16 must not be inserted into or removed from a board with V_{PP} at 25V ±1V to prevent damage to the device.

Note 3: The maximum allowable voltage which may be applied to the V_{PP} pin during programming is 26V. Care must be taken when switching the V_{PP} supply to prevent overshoot exceeding the 26V maximum specification. A 0.1 µF capacitor is required across V_{PP}, V_{CC} to GND to suppress spurious voltage transients which may damage the device.

Functional Description

DEVICE OPERATION

The six modes of operation of the NMC27C16 are listed in Table I. It should be noted that all inputs for the six modes are at TTL levels. The power supplies required are a 5V V_{CC} and a V_{PP}. The V_{PP} power supply must be at 25V during the three programming modes, and must be at 5V in the other three modes.

Read Mode

The NMC27C16 has two control functions, both of which must be logically active in order to obtain data at the outputs. Chip Enable ($\overline{CE}$) is the power control and should be used for device selection. Output Enable ($\overline{OE}$) is the output control and should be used to gate data to the output pins, independent of device selection. Assuming that addresses are stable, address access time (t_{ACC}) is equal to the delay from $\overline{CE}$ to output (t_{CE}). Data is available at the outputs t_{OE} after the falling edge of $\overline{OE}$, assuming that $\overline{CE}$ has been low and addresses have been stable for at least $t_{ACC} - t_{OE}$. The NMC27C16 requires one address transition after initial power-up to reset the outputs.

Standby Mode

The NMC27C16 has a standby mode which reduces the active power dissipation by 98%, from 26.25 mW to 0.53 mW. The NMC27C16 is placed in the standby mode by applying a TTL high signal to the $\overline{CE}$ input. When in standby mode, the outputs are in a high impedance state, independent of the $\overline{OE}$ input.

Output OR-Tying

Because NMC27C16s are usually used in larger memory arrays, National has provided a 2-line control function that accommodates this use of multiple memory connections. The 2-line control function allows for:

a) the lowest possible memory power dissipation, and

b) complete assurance that output bus contention will not occur.

To most efficiently use these two control lines, it is recommended that $\overline{CE}$ (pin 18) be decoded and used as the primary device selecting function, while $\overline{OE}$ (pin 20) be made a common connection to all devices in the array and connected to the READ line from the system control bus. This assures that all deselected memory devices are in their low power standby modes and that the output pins are active only when data is desired from a particular memory device.

Programming

CAUTION: Exceeding 26.5V on pin 21 (V_{PP}) will damage the NMC27C16.

Initially, and after each erasure, all bits of the NMC27C16 are in the "1" state. Data is introduced by selectively programming "0s" into the desired bit locations. Although only "0s" will be programmed, both "1s" and "0s" can be presented in the data word. The only way to change a "0" to a "1" is by ultraviolet light erasure.

The NMC27C16 is in the programming mode when the V_{PP} power supply is at 25V and $\overline{OE}$ is at V_{IH}. It is required that a 0.1 µF capacitor be placed across V_{PP}, V_{CC} to ground to suppress spurious voltage transients which may damage the device. The data to be programmed is applied 8 bits in parallel to the data output pins. The levels required for the address and data inputs are TTL.

When the address and data are stable, a 50 ms, active high, TTL program pulse is applied to the $\overline{CE}$/PGM input. A program pulse must be applied at each address location to be programmed. You can program any location at any time—either individually, sequentially, or at random. The program pulse has a maximum width of 55 ms. The NMC27C16 must not be programmed with a DC signal applied to the $\overline{CE}$/PGM input.

Functional Description (Continued)

Programming multiple NMC27C16s in parallel with the same data can be easily accomplished due to the simplicity of the programming requirements. Like inputs of the paralleled NMC27C16s may be connected together when they are programmed with the same data. A high level TTL pulse applied to the $\overline{CE}$/PGM input programs the paralleled NMC27C16s.

Program Inhibit

Programming multiple NMC27C16s in parallel with different data is also easily accomplished. Except for $\overline{CE}$/PGM, all like inputs (including $\overline{OE}$) of the parallel NMC27C16s may be common. A TTL level program pulse applied to an NMC27C16's $\overline{CE}$/PGM input with V_{PP} at 25V will program that NMC27C16. A low level $\overline{CE}$/PGM input inhibits the other NMC27C16 from being programmed.

Program Verify

A verify should be performed on the programmed bits to determine whether they were correctly programmed. The verify may be performed with V_{PP} at 25V. V_{PP} must be at V_{CC}, except during programming and program verify.

ERASURE CHARACTERISTICS

The erasure characteristics of the NMC27C16 are such that erasure begins to occur when exposed to light with wavelengths shorter than approximately 4000 Angstroms (Å). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000Å–4000Å range. Opaque labels should be placed over the NMC27C16 window to prevent unintentional erasure. Covering the window will also prevent temporary functional failure due to the generation of photo currents.

The recommended erasure procedure for the NMC27C16 is exposure to short wave ultraviolet light which has a wavelength of 2537 Angstroms (Å). The integrated dose (i.e., UV intensity × exposure time) for erasure should be a minimum of 15W-sec/cm². The erasure time with this dosage is approximately 21 minutes using an ultraviolet lamp with a 12,000 μW/cm² power rating. The NMC27C16 should be placed within 1 inch of the lamp tubes during erasure. Some lamps have a filter on their tubes which should be removed before erasure.

Note: The NMC27C16-55 may take up to 80 minutes for complete erasure to occur.

An erasure system should be calibrated periodically. The distance from lamp to unit should be maintained at one inch. The erasure time increases as the square of the distance. (If distance is doubled the erasure time increases by a factor of 4.) Lamps lose intensity as they age. When a lamp is changed, the distance has changed, or the lamp has aged, the system should be checked to make certain full erasure is occurring. Incomplete erasure will cause symptoms that can be misleading. Programmers, components, and even system designs have been erroneously suspected when incomplete erasure was the problem.

SYSTEM CONSIDERATION

The power switching characteristics of EPROMs require careful decoupling of the devices. The supply current, I_{CC}, has three segments that are of interest to the system designer—the standby current level, the active current level, and the transient current peaks that are produced on the falling and rising edges of chip enable. The magnitude of these transient current peaks is dependent on the output capacitance loading of the device. The associated transient voltage peaks can be suppressed by properly selected decoupling capacitors. It is recommended that a 0.1 μF ceramic capacitor be used on every device between V_{CC} and GND. This should be a high frequency capacitor of low inherent inductance. In addition, a 4.7 μF bulk electrolytic capacitor should be used between V_{CC} and GND for each eight devices. The bulk capacitor should be located near where the power supply is connected to the array. The purpose of the bulk capacitor is to overcome the voltage drop caused by the inductive effects of the PC board traces.

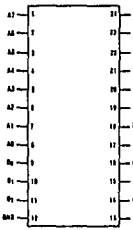
TABLE I. Mode Selection

Pins / Mode	$\overline{CE}$/PGM (18)	$\overline{OE}$ (20)	V_P (21)	V_{CC} (24)	Outputs (9–11, 13–17)
Read	V_{IL}	V_{IL}	V_{CC}	5	D_{OUT}
Standby	V_{IH}	Don't Care	V_{CC}	5	Hi-Z
Program	Pulsed V_{IL} to V_{IH}	V_{IH}	25	5	D_{IN}
Program Verify	V_{IL}	V_{IL}	25	5	D_{OUT}
Program Inhibit	V_{IL}	V_{IH}	25	5	Hi-Z
Output Disable	X	V_{IH}	V_{CC}	5	Hi-Z

Connection Diagram

27C256	27C128	27C64	27C32
27256	27128	2764	2732
V_{PP}	V_{PP}	V_{PP}	
A12	A12	A12	
A7	A7	A7	A7
A6	A6	A6	A6
A5	A5	A5	A5
A4	A4	A4	A4
A3	A3	A3	A3
A2	A2	A2	A2
A1	A1	A1	A1
A0	A0	A0	A0
O_0	O_0	O_0	O_0
O_1	O_1	O_1	O_1
O_2	O_2	O_2	O_2
GND	GND	GND	GND

Dual-In-Line Package
NMC27C16



TL/D/5278-2

Top View

Note: Socket compatible EPROM pin configurations are shown in the blocks adjacent to the NMC27C16 pins

Order Number NMC27C16
See NS Package Number J24AQ

27C32	27C64	27C128	27C256
2732	2764	27128	27256
	V_{CC}	V_{CC}	V_{CC}
	PGM	PGM	A14
V_{CC}	NC	A13	A13
A8	A8	A8	A8
A9	A9	A9	A9
A11	A11	A11	A11
$\overline{OE}/V_{PP}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$
A10	A10	A10	A10
$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
O_7	O_7	O_7	O_7
O_6	O_6	O_6	O_6
O_5	O_5	O_5	O_5
O_4	O_4	O_4	O_4
O_3	O_3	O_3	O_3

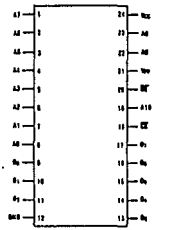
Commercial Temp Range (0°C to +70°C) V_{CC} = 5V ± 5%

Parameter/Order Number	Access Time (ns)
NMC27C16-30	300
NMC27C16-35	350
NMC27C16-45	450
NMC27C16-55	550

Connection Diagram

27C256 27256	27C128 27128	27C64 2764	27C32 2732
V_{PP}	V_{PP}	V_{PP}	
A12	A12	A12	
A7	A7	A7	A7
A6	A6	A6	A6
A5	A5	A5	A5
A4	A4	A4	A4
A3	A3	A3	A3
A2	A2	A2	A2
A1	A1	A1	A1
A0	A0	A0	A0
O_0	O_0	O_0	O_0
O_1	O_1	O_1	O_1
O_2	O_2	O_2	O_2
GND	GND	GND	GND

Dual-In-Line Package
NMC27C16



TL/D/5275-2

27C32 2732	27C64 2764	27C128 27128	27C256 27256
	V_{CC}	V_{CC}	V_{CC}
	$\overline{PGM}$	$\overline{PGM}$	A14
V_{CC}	NC	A13	A13
A8	A8	A8	A8
A9	A9	A9	A9
A11	A11	A11	A11
$\overline{OE}/V_{PP}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$
A10	A10	A10	A10
$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
O_7	O_7	O_7	O_7
O_6	O_6	O_6	O_6
O_5	O_5	O_5	O_5
O_4	O_4	O_4	O_4
O_3	O_3	O_3	O_3

Top View

Note: Socket compatible EPROM pin configurations are shown in the blocks adjacent to the NMC27C16 pins

Order Number NMC27C16
See NS Package Number J24AQ

Commercial Temp Range (0°C to +70°C) V_{CC} = 5V ±5%

Parameter/Order Number	Access Time (ns)
NMC27C16-30	300
NMC27C16-35	350
NMC27C16-45	450
NMC27C16-55	550

TOSHIBA INTEGRATED CIRCUIT TECHNICAL DATA

TC74HC240P/F • TC74HC241P TC74HC244P/F

C²MOS DIGITAL INTEGRATED CIRCUIT

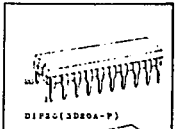
PRELIMINARY

TC74HC240P/F OCTAL BUS BUFFER WITH INVERTED 3-STATE OUTPUTS
TC74HC241P OCTAL BUS BUFFER WITH NONINVERTED 3-STATE OUTPUTS
TC74HC244P/F OCTAL BUS BUFFER WITH NONINVERTED 3-STATE OUTPUTS

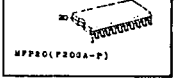
The TC74HC240, TC74HC241 and TC74HC244 are high speed CMOS OCTAL BUS BUFFER's fabricated with silicon C²MOS technology.
They achieve the high speed operation similar to equivalent LSTTL while maintaining the CMOS low power dissipation.
The designer has a choice of selected combinations of inverting and noninverting outputs, symmetrical $\overline{G}$ (active-low output control) inputs, and complementary G and $\overline{G}$ inputs. Each control input govern four BUS BUFFERs.
These divices are designated to be used with 3-state memory address drivers, etc.
All inputs are equipped with protection circuits against static discharge or transient excess voltage.

FEATURES:

- High Speed.....t_{pd}=11ns(Typ.) at V_{CC}=5V
- Low Power Dissipation.....I_{CC}=4uA(Max.) at Ta=25°C
- High Noise Immunity.....V_{NIH}=V_{NIL}=28% V_{CC}(Min.)
- Output Drive Capability.....15 LSTTL Loads
- Symmetrical Output Impedance..$|I_{OH}|$=I_{OL}=6mA(Min.)
- Balanced Propagation Delays...t_{pLH} ≈ t_{pHL}
- Wide Operating Voltage Range..$V_{CC(opr)}$=2V ~ 6V
- Pin and Function Compatible with 74LS 240/241/244



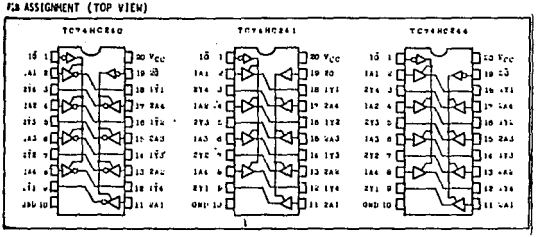
DIP20(3D20A-P)



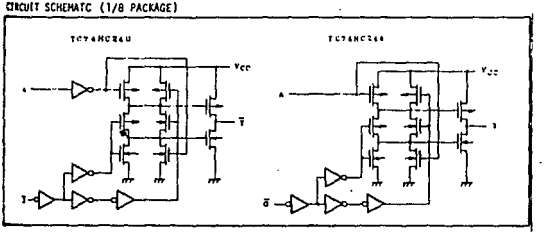
MFP20(F20GA-P)

TC74HC240P/F•TC74HC241P
TC74HC244P/F

PIN ASSIGNMENT (TOP VIEW)



CIRCUIT SCHEMATC (1/8 PACKAGE)



TRUTH TABLE

INPUTS			OUTPUTS	
$\overline{G}$	$G^{(*)}$	A_n	Y_n	$\overline{Y}_n$
L	H	L	L	H
L	H	H	H	L
H	L	X	Z	Z

(*) : Applied only for TC74HC241

(**) : Applied only for TC74HC240

X : Don't Care

Z : High Impedance

TC74HC367P/F
TC74HC368P/F

CMOS DIGITAL INTEGRATED CIRCUIT

PRELIMINARY

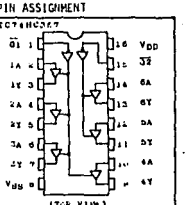
HEX BUS BUFFER
TC74HC367P/F NON-INVERTING
TC74HC368P/F INVERTING

The TC74HC367 and TC74HC368 are high speed CMOS 3-STATE BUS BUFFERs fabricated with silicon gate C^2MOS technology.
These devices achieve the high speed operation similar to equivalent LSTTL, while maintaining the CMOS low power dissipation. These devices contain six buffers, and four buffers are controlled by a enable input ($\overline{G1}$) and the other two buffers are controlled by the other enable input ($\overline{G2}$); these outputs of each buffer group are enabled when $\overline{G1}$ and/or $\overline{G2}$ inputs are held low, and when held high these outputs are disabled to be high-impedance.
These outputs are capable of driving up to 15 LSTTL.
The designer has a choice of non-inverting outputs (HC367) and inverting outputs (HC368).
All outputs are equipped with protection circuits against static discharge or transient excess voltage.

FEATURES:

- High Speed ..... t_{pd}=13ns(Typ.) at V_{CC}=5V
- Low Power Dissipation ..... I_{CC}=4μA(Max.) at Ta=25°C
- High Noise Immunity ..... V_{NIH}=V_{NIL}=28% V_{CC}(Min.)
- Output Drive Capability ..... 15 LSTTL Loads
- Symmetrical Output Impedance ..... $|I_{OH}|$=I_{OL}= 6mA
- Balanced Propagation Delays ..... t_{pLH}≒t_{pHL}
- Wide Operating Voltage Range ..... $V_{CC(opr.)}$=2V~6V
- Pin and Function Compatible with 74LS367/368

PIN ASSIGNMENT



TRUTH TABLE

INPUTS		OUTPUTS	
$\overline{G}$	An	Yn(367)	$\overline{Yn}$(368)
L	L	L	H
L	H	H	L
H	X	Z	Z

X : DON'T CARE
Z : HIGH IMPEDANCE

National Semiconductor

DM54LS155/DM74LS155, DM54LS156/DM74LS156 Dual 2-Line to 4-Line Decoders/Demultiplexers

General Description

These TTL circuits feature dual 1-line-to-4-line demultiplexers with individual strobes and common binary-address inputs in a single 16-pin package. When both sections are enabled by the strobes, the common address inputs sequentially select and route associated input data to the appropriate output of each section. The individual strobes permit activating or inhibiting each of the 4-bit sections as desired. Data applied to input C1 is inverted at its outputs and data applied at C2 is true through its outputs. The inverter following the C1 data input permits use as a 3-to-8-line decoder, or 1-to-8-line demultiplexer, without external gating. Input clamping diodes are provided on these circuits to minimize transmission-line effects and simplify system design.

Features

- Applications:
  - Dual 2-to-4-line decoder
  - Dual 1-to-4-line demultiplexer
  - 3-to-8-line decoder
  - 1-to-8-line demultiplexer
- Individual strobes simplify cascading for decoding or demultiplexing larger words
- Input clamping diodes simplify system design
- Choice of outputs:
  - Totem pole (LS155)
  - Open-collector (LS156)

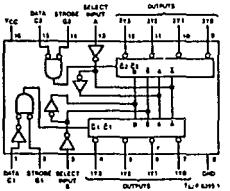
Absolute Maximum Ratings (Note 1)

Supply Voltage	7V
Input Voltage	7V
Storage Temperature Range	−65°C to 150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the "Electrical Characteristics" table are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

Connection Diagram

Dual-In-Line Package



S4LS155 (J)
S4LS156 (J)
74LS155 (N)
74LS156 (N)

Function Tables

3-Line-to-4-Line Decoder or 1-Line-to-4-Line Demultiplexer

Inputs				Outputs			
Select	Strobe	Data					
B	A	G1	C1	1Y0	1Y1	1Y2	1Y3
X	X	H	X	H	H	H	H
L	L	L	H	L	H	H	H
L	H	L	H	H	L	H	H
H	L	L	H	H	H	L	H
H	H	L	H	H	H	H	L
X	X	H	X	H	H	H	H

Inputs				Outputs			
Select	Strobe	Data					
B	A	G2	C2	2Y0	2Y1	2Y2	2Y3
X	X	H	X	H	H	H	H
L	L	L	L	L	H	H	H
L	H	L	L	H	L	H	H
H	L	L	L	H	H	L	H
H	H	L	L	H	H	H	L
X	X	H	X	H	H	H	H

3-Line-to-8-Line Decoder or 1-Line-to-8-Line Demultiplexer

Inputs				Outputs							
Select		Strobe On Data		(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
C1*	B	A	G1	1Y0	1Y1	1Y2	1Y3	1Y0	1Y1	1Y2	1Y3
X	X	X	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H
L	H	L	L	H	H	L	H	H	H	H	H
L	H	H	L	H	H	H	L	H	H	H	H
H	L	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	L	H	H
H	H	L	L	H	H	H	H	H	H	L	H
H	H	H	L	H	H	H	H	H	H	H	L

*C = inputs C1 and C2 connected together
†G = inputs G1 and G2 connected together
H = high level, L = low level, X = don't care

APENDICE **B**

LISTADO DE LA INTERRUPCION 17H

Para entender el funcionamiento del puerto paralelo fue necesario desensamblar la interrupcion 17H del BIOS. A continuación presentamos el listado de la interrupción 17H desensamblada y con comentarios. Cabe aclarar que el codigo varía entre los diferentes modelos de computadoras.

- ENTRADAS

  AH - Numero de servicio:

  00H - Enviar un byte.

  01H - Inicializar impresora.

  02H - Leer Estado de la impresora.

  AL - Caracter a enviar para el servicio 00H.

  DX - Número de impresora (0 a 2).

- SALIDA

  AH - Estado de la impresora para todos los servicios de acuerdo a la siguiente tabla:

Bit 7	6	5	4	3	2	1	0	Significado (cuando vale 1)
1	.	.	.	.	.	.	.	Impresora *no* ocupada (0 = ocupada)
.	1	.	.	.	.	.	.	Recibido por la impresora (Ack)
.	.	1	.	.	.	.	.	Sin papel
.	.	.	1	.	.	.	.	Impresora seleccionada
.	.	.	.	1	.	.	.	Error
.	.	.	.	.	1	.	.	No usado
.	.	.	.	.	.	1	.	No usado
.	.	.	.	.	.	.	1	Tiempo agotado (time-out)

- INTERRUPCION 17H

```
F000:EFD2 FB          STI             ;Responder a int.
                                      ;mascarables
F000:EFD3 1E          PUSH DS         ;Salvar registros
F000:EFD4 52          PUSH DX
F000:EFD5 56          PUSH SI
F000:EFD6 51          PUSH CX
F000:EFD7 53          PUSH BX
F000:EFD8 E87E0A      CALL FA59       ;Apuntar al área de datos
                                      ;BIOS
F000:EFDB 8BF2        MOV  SI,DX      ;No. de puerto en SI
F000:EFDD 8A5C78      MOV  BL,[SI+78] ;Traer valor de Time-Out
F000:EFE0 D1E6        SHL  SI,1       ;Desplazar a la izq. el
                                      ;No. de puerto dando 0, 2
                                      ;ó 4
F000:EFE2 8B5408      MOV  DX,[SI+08] ;Traer direc. base del
                                      ;puerto
F000:EFE5 0BD2        OR   DX,DX      ;Prender bandera Z según
                                      ;resultado
F000:EFE7 740C        JZ   EFF5       ;Si direc. = 0 terminar
                                      ;porque no hay puerto
F000:EFE9 0AE4        OR   AH,AH
F000:EFEB 740E        JZ   EFFB       ;SERVICIO 00H
F000:EFED FECC        DEC  AH
F000:EFEF 743F        JZ   F030       ;SERVICIO 01H
F000:EFF1 FECC        DEC  AH
F000:EFF3 7428        JZ   F01D       ;SERVICIO 02H
F000:EFF5 5B          POP  BX         ;Recuperar registros
F000:EFF6 59          POP  CX
F000:EFF7 5E          POP  SI
F000:EFF8 5A          POP  DX
F000:EFF9 1F          POP  DS
F000:EFFA CF          IRET

F000:FA59 50          PUSH AX         ;Rutina que hace que DS
                                      ;apunte
F000:FA5A B84000      MOV  AX,0040    ;al área de datos BIOS
F000:FA5D 8ED8        MOV  DS,AX
F000:FA5F 58          POP  AX
F000:FA60 C3          RET
```

- SERVICIO 00H

```
F000:EFFB 50          PUSH AX         ;Salvar AX
F000:EFFC EE          OUT  DX,AL      ;Enviar dato por líneas
                                      ;Datos
F000:EFFD 42          INC  DX         ;Incrementar a dir.
                                      ;base + 1
F000:EFFE 2BC9        SUB  CX,CX      ;Limpiar CX
F000:F000 EC          IN   AL,DX      ;Leer Estado en AL
F000:F001 8AE0        MOV  AH,AL      ;Respaldar Estado en AH
```

```
F000:F003 A880      TEST AL,80      ;Si el bit 8 es 1
                                    ;(desocupada)
F000:F005 750E      JNZ  F015       ;salta a continuar
F000:F007 E2F7      LOOP F000       ;Demora
F000:F009 FECB      DEC  BL         ;Decrementa Time-Out
F000:F00B 75F1      JNZ  EFFE       ;Ciclo que espera a que
                                    ;la impresora se desocupe
F000:F00D 80CC01    OR   AH,01      ;Prender bit 0 (se acabó
                                    ;tiempo)
F000:F010 80E4F9    AND  AH,F9      ;Apagar bits 1 y 2 que no
                                    ;se usan
F000:F013 EB13      JMP  F028       ;Terminar porque se acabo
                                    ;el tiempo y no respondió
                                    ;la imp.
F000:F015 B00D      MOV  AL,0D
F000:F017 42        INC  DX         ;Incrementar a dir.
                                    ;base + 2
F000:F018 EE        OUT  DX,AL      ;Enviar 0D por Control
                                    ;(Slct In = Strobe = L)
F000:F019 B00C      MOV  AL,0C
F000:F01B EE        OUT  DX,AL      ;Enviar 0C por Control
                                    ;(Slct In = L)
F000:F01C 58        POP  AX         ;Recuperar en AL el
                                    ;caracter
F000:F01D 50        PUSH AX         ;A partir de esta
                                    ;instruc. es
F000:F01E 8B5408    MOV  DX,[SI+08] ;el servicio 02H
F000:F021 42        INC  DX
F000:F022 EC        IN   AL,DX
F000:F023 8AE0      MOV  AH,AL
F000:F025 80E4F8    AND  AH,F8
F000:F028 5A        POP  DX
F000:F029 8AC2      MOV  AL,DL
F000:F02B 80F448    XOR  AH,48
F000:F02E EBC5      JMP  EFF5
```

* SERVICIO 01H

```
F000:F030 50        PUSH AX
F000:F031 42        INC  DX         ;Incrementar a dir.
                                    ;base + 1
F000:F032 42        INC  DX         ;Incrementar a dir.
                                    ;base + 2
F000:F033 B008      MOV  AL,08
F000:F035 EE        OUT  DX,AL      ;Enviar un 8 por Control
F000:F036 B8E803    MOV  AX,03E8    ;Repetir un
F000:F039 48        DEC  AX         ;ciclo 1000 veces
F000:F03A 75FD      JNZ  F039       ;como demora
F000:F03C B00C      MOV  AL,0C
F000:F03E EE        OUT  DX,AL      ;Enviar un 12 por Control
F000:F03F EBDD      JMP  F01E       ;Saltar al servicio 02H
```

```
F000:F01E 8B5408      MOV   DX,[SI+08];Servicio 02H
F000:F021 42          INC   DX
F000:F022 EC          IN    AL,DX
F000:F023 8AE0        MOV   AH,AL
F000:F025 80E4F8      AND   AH,F8
F000:F028 5A          POP   DX
F000:F029 8AC2        MOV   AL,DL
F000:F02B 80F448      XOR   AH,48
F000:F02E EBC5        JMP   EFF5
```

- SERVICIO 02H

```
F000:F01D 50          PUSH  AX
F000:F01E 8B5408      MOV   DX,[SI+08];Releer dir. base
F000:F021 42          INC   DX        ;Incrementar a dir.
                                      ;base + 1
F000:F022 EC          IN    AL,DX     ;Leer Estado en AL
F000:F023 8AE0        MOV   AH,AL     ;Respaldar Estado en AH
F000:F025 80E4F8      AND   AH,F8     ;Apagar bit 0, 1 y 2
F000:F028 5A          POP   DX
F000:F029 8AC2        MOV   AL,DL     ;AL recupera el valor que
                                      ;tenía antes del servicio
                                      ;y AH queda con el Edo.
                                      ;de la impresora
F000:F02B 80F448      XOR   AH,48     ;Invierte valor de bits 3
                                      ;y 6 (Error y Ack)
F000:F02E EBC5        JMP   EFF5      ;Salta a terminar int.
```

APENDICE **C**

MANUAL DEL USUARIO DEL PROGRAMADOR DE EPROM

INTRODUCCION.

Este programador le permitirá grabar una memoria EPROM 27C16 o 2716 con gran facilidad utilizando su computadora PC compatible como interface. Para ello se cuenta con el software adecuado que le permitirá leer el contenido de la memoria así como grabarla una sola vez. Recuerde que para borrar la EPROM es necesaria una lámpara de luz ultravioleta que se debe adquirir por separado.

Este manual explica brevemente como utilizar el software y el hardware necesarios.

CONEXIONES PREVIAS.

El equipo necesario es el siguiente:

- Memoria EPROM 27(C)16.
- Programador de EPROM.
- Cable de impresora (DB-25P a CENTRONICS).
- Computadora PC o compatible.

Proceda como se indica a continuación. IMPORTANTE: DE NO SEGUIR ESTA SECUENCIA SE QUEMARA LA MEMORIA.

1.- Coloque el CI 27(C)16 en la base provista para ello en el grabador, teniendo cuidado de NO tocar los pines.

2.- Conecte la salida del puerto paralelo de su computadora a la entrada correspondiente en el grabador.

3.- Conecte el programador a la toma de corriente.

4.- Encienda la computadora si aún no está encendida.

5.- Presione el switch de encendido (+5V). Asegúrese de encender primero la computadora.

6.- Si sólo va a leer una EPROM, salte al siguiente paso. Si va a grabar, presione el switch de grabación (+25V).

7.- Ejecute el programa EPROM.EXE.

UTILIZACION DEL PROGRAMA EPROM.EXE.

En general, el programa es muy sencillo de ejecutar ya que indica en cada paso las posibles alternativas a seguir y la manera de llevarlas a cabo.

La primera pantalla que se presenta al correr el programa es una presentación. Salga de ella presionando cualquier tecla.

En seguida se encontrará en la pantalla de instrucciones. Estas instrucciones son un recordatorio de las conexiones previas que ya se mencionaron. En este menú tiene las siguientes opciones:

- F1 para leer las instrucciones nuevamente. Cada vez que se llena una ventana puede presionar la tecla PgDn para

continuar o, si desea finalizar por cualquier motivo, presione ESC.

- S para salir al sistema. Use esta opción si se olvidó de efectuar alguna conexión o para terminar la ejecución del programa.

- Cualquier otra tecla para omitir las instrucciones y poder continuar.

La siguiente pantalla contiene el menú principal. Este menú le permite tres opciones que se seleccionan moviendo las flechas del cursor hacia arriba o hacia abajo y presionando ENTER cuando se encuentre sobre la opción deseada. También le permite utilizar ESC para regresar al menú de instrucciones. Las tres opciones mencionadas son las siguientes:

Grabar la EPROM

Inicialmente se le pregunta en que formato desea trabajar, esto es, el sistema de numeración en que se van a introducir las direcciones y los datos a grabar. Presione la tecla A para elegir el sistema decimal o la B para elegir el sistema hexadecimal.

A continuación el programa le pide que introduzca la dirección inicial y luego la dirección final, es decir el rango de localidades de memoria que desea grabar. Si desea grabar sólo una localidad, repita esa dirección cuando se le solicite la dirección final. Tenga cuidado de proporcionar la dirección y, en su momento los datos, en el formato adecuado así como dentro de los límites válidos. El programa le recordará el formato escogido y los valores permitidos. Si se comete algún error al introducir los valores, éstos

serán rechazados y se le solicitarán nuevamente.

Una vez proporcionadas las direcciones, se mostrará una ventana con tres columnas. La de la izquierda muestra la dirección inicial en decimal, hexadecimal y binario, dada por el usuario. En la columna siguiente se espera el dato que deberá ser grabado. Este dato se introduce en la línea que parpadea abajo. Tan pronto como se proporciona el dato, éste es mostrado en los tres sistemas de numeración mencionados y en código ASCII en la segunda columna. En fracciones de segundo se procede a grabar la EPROM y en la tercera columna se muestra el contenido leído para que el usuario verifique que sea correcto.

Al terminar, puede presionar ESC para repetir la acción de grabar o ENTER para regresar al menú principal, desde donde podrá leer o salir.

Leer la EPROM

Inicialmente se le pregunta en que formato desea trabajar, esto es, el sistema de numeración en que se van a introducir las direcciones a leer. Presione la tecla A para elegir el sistema decimal o la B para elegir el sistema hexadecimal.

A continuación el programa le pide que introduzca la dirección inicial y luego la dirección final, es decir el rango de localidades de memoria que desea leer. Si desea leer sólo una localidad, repita esa dirección cuando se le solicite la dirección final. Tenga cuidado de proporcionar las direcciones en el formato adecuado, así como dentro de los límites válidos. El programa le recordará el formato escogido y los valores permitidos. Si se comete algún error al introducir los valores, éstos serán rechazados y se le

solicitarán nuevamente.

Una vez proporcionadas las direcciones, se mostrará una ventana con dos columnas. La de la izquierda muestra las direcciones, desde la inicial hasta la final (si caben en la ventana), en decimal, hexadecimal y binario. El contenido de cada localidad es mostrado en los tres sistemas de numeración mencionados y en código ASCII en la segunda columna. Si las direcciones no caben en una sola ventana presione ENTER para continuar cada vez que se llene ésta.

Al terminar, puede presionar ESC para repetir la acción de leer o ENTER para regresar al menú principal, desde donde podrá grabar o salir.

Salir al sistema.

Use esta opción para regresar al DOS y terminar la sesión.

NOTA: Durante la introducción de las direcciones y/o datos en los menús de grabación y lectura puede utilizar las siguientes teclas:

- ESC para borrar el valor completo (antes de dar ENTER)
- BACKSPACE para borrar el último caracter
- CTRL-D para tomar un caracter de la última línea que se introdujo
- CTRL-F para tomar toda la última línea que se introdujo

FIN DE LA SESION.

IMPORTANTE: SIGA ESTA SECUENCIA PARA NO QUEMAR LA EPROM.

1.- Salga del programa como ya se indicó.
2.- Si grabó, apague el switch de grabación (+25V), si no, salte al paso 3.
3.- Apague el switch de encendido (+5V).
4.- Retire la EPROM de su base (recordando NO tocar los pines).
5.- Apague su computadora, si lo desea.

CALIBRACION DEL VOLTAJE DE PROGRAMACION (Vpp).

Para ajustar Vpp, en la parte posterior del Programador se encuentra un tornillo de calibración y dos puntas de prueba para tierra y Vpp. Introducir en éstas últimas las puntas de un multímetro mientras se gira el tornillo hasta lograr el valor de +25 V.

BIBLIOGRAFIA

- Boylestad, Robert [y] Nashelsky, Louis. "Electrónica Teoría de Circuitos". D.F. México: Prentice-Hall Hispanoamericana S.A., 1986.

- Castruita Avila, César. "¡Al Abordaje Contra los Piratas!". En: Personal Computing. Año 3, No. 29. D.F. México: Editorial Sayrols S.A., 1990. pp. 22-24.

- Ciarcia, Steve. "Construya una Microcomputadora basado en el Z80. Guía de Diseño y Funcionamiento". México: McGraw-Hill de México S.A. de C.V., 1981.

- Eggebrecht, Lewis C. "Interfacing to the IBM Personal Computer". 2nd. ed. Carmel, Indiana, E.U.A.: SAMS, 1991.

- EPSON America, Inc. FX-850/1050 User's Manual. Torrance, California, E.U.A., 1989.

- Ferreyra Cortés, Gonzalo. "Virus en las Computadoras". D.F. México: Macrobit Editores S.A. de C.V., 1991.

- Foerster, Scott. "The Printer Bible". Carmel, Indiana, E.U.A.: Que Corporation, 1990.

- National Semiconductor Corporation. Logic Databook. Vol. II. Santa Clara, California, E.U.A., 1984.

- National Semiconductor Corporation. Memory Databook. Vol. I. Santa Clara, California, E.U.A., 1984.

- Norton, Peter. "Inside the IBM PC and PS/2". 3rd. ed. Nueva York, N.Y., E.U.A.: Brady, 1990.

- Norton, Peter [y] Wilton, Richard. "The New Peter Norton Programmer's Guide to the IBM PC & PS/2". Redmond, Washington, E.U.A.: Microsoft Press, 1988.

- Rosch, Winn L. "The Winn Rosch Hardware Bible". Nueva York, N.Y., E.U.A.: Brady, 1989.

- Sarrazin, Christian. "Las Pilas de Litio". En: Mundo Científico. No. 112. Vol. 11. Barcelona, España: Fontalba S.A. pp. 378-384.

- Scanlon, Leo J. "IBM PC & XT Assembly Language. A Guide for Programmers". Nueva York, N.Y., E.U.A.: Brady, 1985.

- Seyer, Martin D. "RS-232 Made Easy". 2nd. ed. Englewood Cliffs, Nueva Jersey, E.U.A.: Prentice Hall, 1991.

- Tocci, Ronald J. "Circuitos y Dispositivos Electrónicos". D.F. México: Interamericana S.A. de C.V., 1985.

- Toshiba Corporation. Toshiba Integrated Circuit Technical Data HS-C^2MOS TC74HC Series. 2nd. ed. Abril, 1985.