ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

" PROGRAMADOR DE MEMORIAS Y MICROCONTROLADORES INTEL"

MARCELO IVAN CUENCA CISNEROS

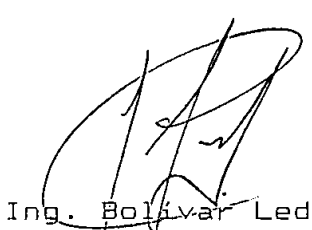
TESIS PREVIA A LA OBTENCION DEL TITULO DE

INGENIERO ELECTRONICO EN CONTROL

DICIEMBRE - 1994

Certificación:

Certifico que bajo mi dirección la presente tesis fue desarrollada en su totalidad por el señor Marcelo Iván Cuenca Cisneros.



Ing. Bolívar Ledesma

DIRECTOR DE TESIS

DEDICATORIA

A Dios por permitirme culminar mi carrera, A la memoria de mi madre, A la abnegada labor de mi padre y Al apoyo y estímulo de mis hermanos.

AGRADECIMIENTO

Al Ing. Bolívar Ledesma por su acertada dirección y al Sr. Raúl Benítez por el financiamiento.

RESUMEN

La presente tesis consiste en el diseño y construcción de un prototipo de un programador para memorias EPROM y microcontroladores INTEL de las familias 8051 y 8048.

El prototipo desarrollado permite:

- Grabar memorias del tipo EPROM comerciales: 2716, 2732, 2764, 27128, 27256, 27512.
- Grabar memoria de programa interna en microcontroladores de la familia MCS-51: 8751H, 87C51, 8751BH, 8752BH
- Grabar memoria de programa interna en microcontroladores de la familia MCS-48: 8748, 8749, 8748H, 8749H.
- Seleccionar el voltaje de programación 12, 21, ó 25 voltios.
- Ser enlazado a un computador personal mediante puerto serial, para lo cual se desarrolló todo el software que se requiere a nivel de PC utilizando QBASIC para realizar las funciones típicas de un programador de memorias y que a continuación se indican:
  - Seleccionar tipo de memoria
  - Seleccionar el voltaje de programación
  - Leer memoria
  - Chequear si la memoria se encuentra en blanco
  - Programar la memoria
  - Almacenar el programa en diskette o disco duro
  - Verificar contenido de una memoria y comparar con un

archivo

- Dispondrá de un algoritmo de programación rápido-lento
- El software es capaz de aceptar formato INTEL.

INDICE

CAPITULO I

CONSIDERACIONES PARA PROGRAMACION DE MEMORIAS Y MICROCONTROLADORES

CAPITULO II

DISEÑO DEL HARDWARE

CAPITULO III

DESARROLLO DEL SOFTWARE

CAPITULO I

CONSIDERACIONES PARA PROGRAMACION DE MEMORIAS Y MICROCONTROLADORES (1)

Un procesador digital requiere generalmente un medio para almacenar información. La información aquí almacenada puede consistir en varios números que deben ser utilizados en un cálculo, resultados de cálculos intermedios, instrucciones que dirigirán un cálculo, o todo lo mencionado. Cuando no hay implicado cálculo alguno, un medio de almacenamiento puede consistir simplemente en un fichero de datos. Por ejemplo, una máquina diseñada para poner direcciones postales en los sobres necesitará medios de almacenamiento para nombres y direcciones. La parte de un procesador digital que provee este medio de almacenamiento se llama memoria.

Una de las mayores ventajas que tienen los sistemas digitales sobre los sistemas analógicos es la de guardar información por periodos de tiempo cortos o largos. Esta capacidad de almacenar información es por lo tanto lo que hace a los sistemas digitales muy versátiles.

Para poder grabar un byte en una localidad de memoria el proceso a seguir es el siguiente: primero en el bus de direcciones se coloca la dirección de la localidad en la que se desea grabar el dato, seguidamente el dato a ser grabado debe ser colocado en el bus de datos de la memoria

y por último se da un pulso de programación que tiene una cierta duración de tiempo. Finalmente, el dato ha sido grabado en la localidad deseada. Durante este proceso se utiliza un voltaje (V_{pp}) que es el voltaje de programación, cuyo valor depende del elemento a ser programado.

1.1 PROGRAMACION DE MEMORIAS

1.1.1 GENERALIDADES (2)

Un dispositivo de memoria es aquel que provee un medio de almacenamiento de la información, que luego se desea recuperar nuevamente.

Los dispositivos de memoria son los elementos básicos de todo circuito secuencial, su utilización principal es en los computadores por ser sistemas complejos que requieren que se guarde información.

En la FIG. 1.1 puede verse la distribución de una memoria:

00	0	0	1	1	0	← Localidad de memoria
01	1				0	← Contenido (0 ó 1)
10	2	1	0	0	1	
11	3	0	0	1	0	

↑ Direcciones — ↑ Número de localidad — Datos

FIGURA 1.1

Puede entenderse a una memoria como un medio físico que puede adoptar un estado definido, el mismo que puede ser determinado de alguna manera como por ejemplo una cinta perforada, un biestable, etc. Se la considera como un conjunto de celdas que son espacios donde se va a almacenar

la información, en una celda se guarda 1 bit; la información puede ser recuperada celda a celda y en los sistemas digitales normalmente se trabaja con grupos de 8 bits (a lo que se conoce como 1 byte); al espacio físico de almacenamiento que ocupa un conjunto de bits se lo conoce como una localidad de memoria y al conjunto de bits en si, como información, se le denomina palabra de información. La longitud de palabra varía desde 4 a 64 bits, en dependencia del sistema con el que se esté trabajando.

Una localidad es diferente de una palabra de información, ya que una localidad puede contener parte de una palabra.

Para independizar una localidad se le asocia con un cierto número que la identifica y al cual se lo conoce como dirección.

Otras definiciones importantes son las siguientes:

Capacidad, la que puede ser expresada como bits o en palabras de información que es lo más usual, como se puede observar en la FIGURA 1.2; por lo general los dispositivos más lentos tienen una capacidad mayor y un costo más bajo.

Otra característica es la velocidad de operación de una memoria, la misma que se define en base al tiempo de acceso (t_Acc).

Para el usuario de una memoria las dos operaciones principales que se deben considerar son:

Escritura.- Mediante la cual se guarda o almacena información en una determinada localidad y en este caso se modifica la información existente de dicha localidad.

Lectura.- Proceso mediante el cual se recupera información de una localidad. Esta operación no debe ser destructiva, en tal forma que pueda realizarse repetidamente.

La forma en que se realizan estas dos operaciones depende del tiempo de acceso.

Tiempo de Acceso (t_{Acc}).- Es el tiempo que requiere el dispositivo de memoria desde que aplicamos una dirección, habilitamos el chip de memoria y es posible realizar una operación ya sea de lectura o escritura.

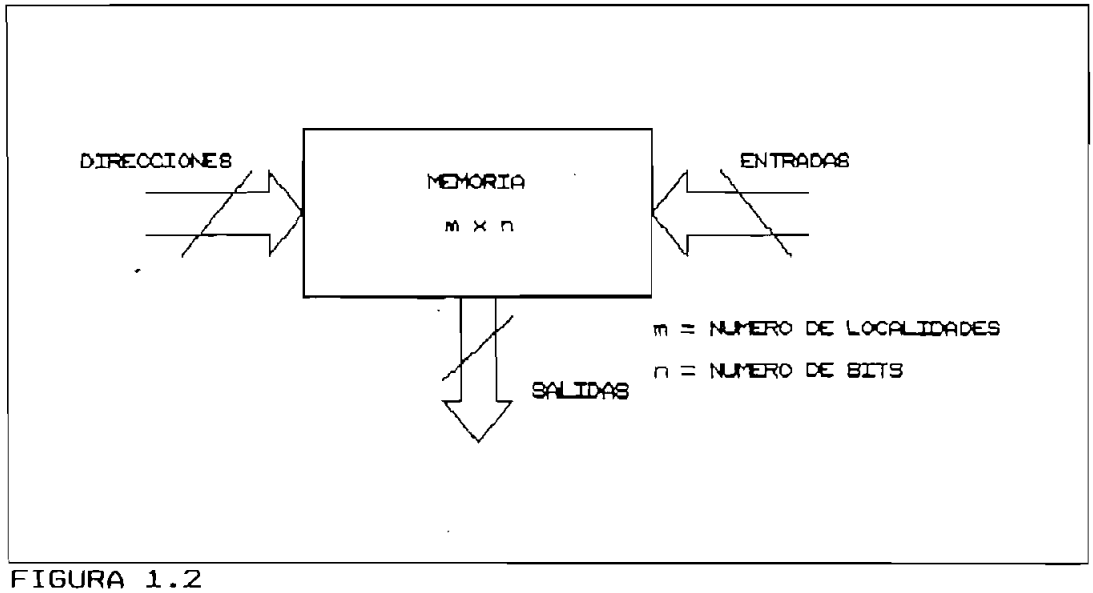


FIGURA 1.2

1.1.2 CLASIFICACIÓN (2)

Se las clasifica en base a dos criterios fundamentales, los cuales son:

1. De acuerdo a la forma de acceso a la información almacenada.
2. De acuerdo a la tecnología utilizada en su fabricación.

En base a la forma de acceso se las puede clasificar en:

a) Memorias solamente de lectura.

b) Memorias de acceso aleatorio.

c) Memorias de acceso secuencial.

De acuerdo a la tecnología de su fabricación pueden ser: magnéticas, ópticas, de semiconductores, etc.

1.1.2.1 Memorias solamente de lectura.

Son dispositivos en los que una vez realizada la programación son inalterables, por lo que permiten únicamente la operación de lectura.

De acuerdo a la definición dada este tipo de dispositivos son destinados únicamente a almacenar un tipo de información, la misma que se mantiene fija, es decir como se ha planteado hay una operación inicial en la cual se guarda la información (programación) y después la memoria sólo puede ser leída y su contenido es inalterable. En las memorias solamente de lectura, ROM, la información es grabada por el fabricante.

Una cualidad importante de estos dispositivos es que la información que guardan no se pierde si se interrumpe la fuente de alimentación.

Este tipo de memorias tienen su aplicación en: generadores de funciones, constantes que deben aparecer siempre, conversión de códigos, para almacenar los programas monitores, microprogramación, etc.

La organización de este tipo de dispositivos se da en la FIG. 1.3, en la misma que se presenta como ejemplo una memoria de 32 palabras de 8 bits. Las palabras son

seleccionadas por las líneas de Ao - A4, denominadas "dirección" las mismas que normalmente tienen buffers; estas líneas son las entradas al circuito decodificador que es el encargado de seleccionar una de las 32 palabras en base al código aplicado. La palabra seleccionada aparece en las salidas de datos de Oo - 07. La línea CS es la encargada de poner la salida en estado de alta impedancia o habilitar la salida, lo cual permite que las líneas de salida puedan integrarse a un bus común de datos.

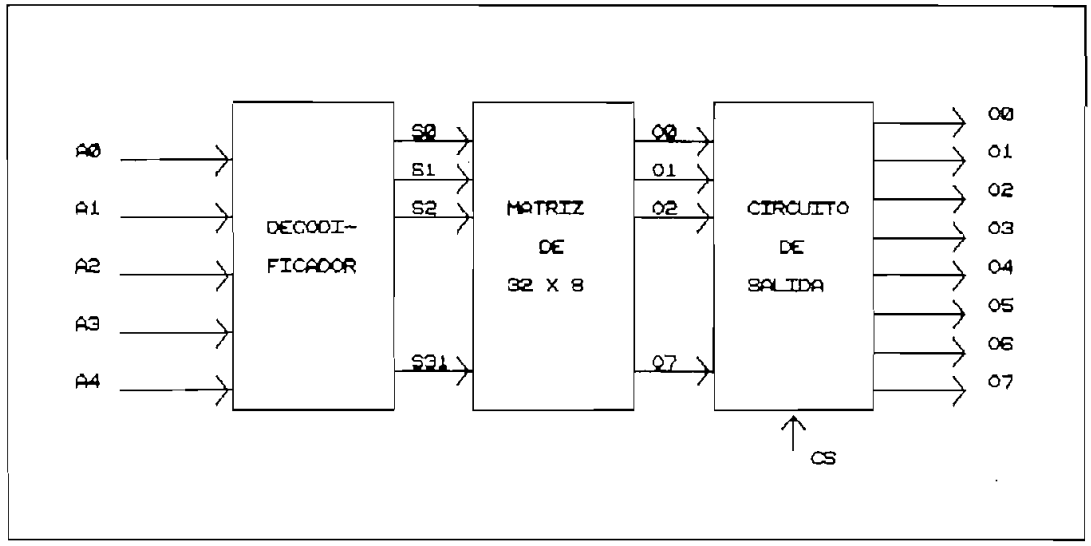


FIGURA 1.3

Siendo este circuito una ROM no necesita las líneas de escritura-lectura o de datos de entrada.

La estructura de la matriz indicada en la FIG. 1.3 se presenta en la FIG. 1.4 donde, como ejemplo se ilustra un arreglo de cuatro palabras de cuatro bits, la cual puede considerarse como una representación del caso real. Los dos bits Ao y A1 son los encargados de seleccionar una de las cuatro filas de palabras de información. Este

decodificador produciría entonces un uno lógico en la línea de salida correspondiente dejando un cero en las líneas de salida restantes.

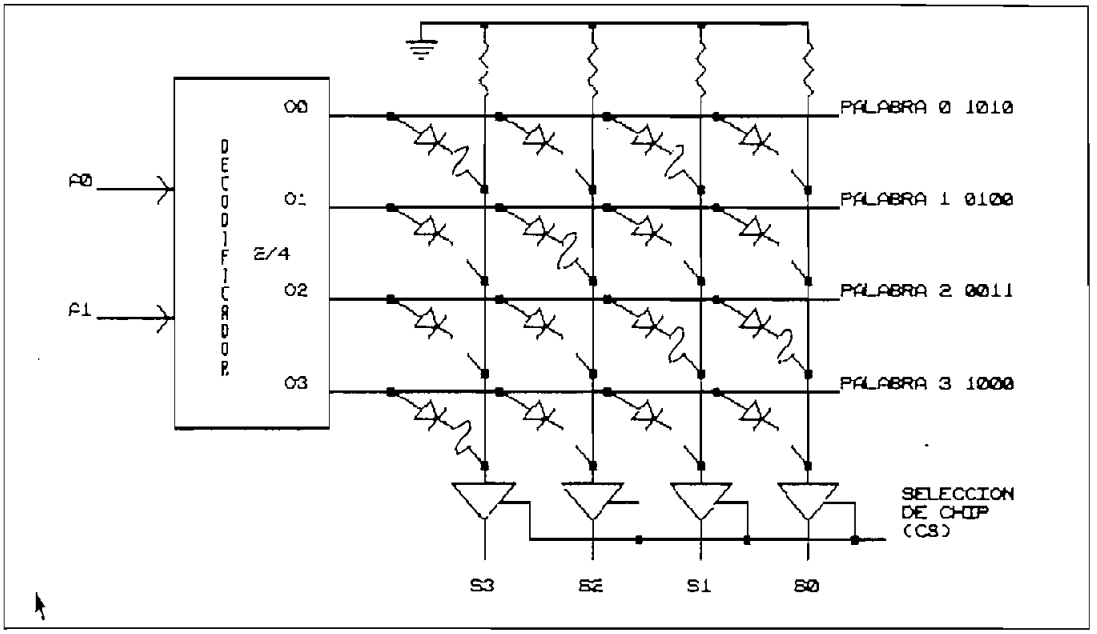


FIGURA 1.4

Así por ejemplo si Ao = 0 y A1 = 1 se pondría un 1L en la salida 02 del decodificador, lo que determina que en las entradas de las compuertas se tenga 0011 y se habilitan las compuertas con la señal de selección de chip (CS). Las salidas de las compuertas son normalmente tres estados para posibilitar la interconexión en paralelo de varias ROM a la vez, determinando el trabajo de cada una con la línea de selección de chip.

Un inconveniente que presenta este circuito es que las salidas del decodificador son las encargadas de suministrar corriente a la carga que está puesta en las mismas.

La dificultad planteada se soluciona utilizando en lugar de los diodos transistores y de esta manera la corriente ya no

tiene que suministrar la salida del decodificador sino la fuente de polarización. También existen ROM disponibles con transistores MOS, tomando en cuenta que en los dos casos depende de las necesidades y por tanto las características que presentan los dos dispositivos; las bipolares si se requiere velocidades de operación mayores y las MOS si se requiere mayor densidad en el empaquetado o menor disipación, a pesar de que el uso de esta tecnología significa una velocidad de operación menor.

La tecnología MOS es especialemente ventajosa en la integración a gran escala (LSI) a causa de que generalmente el transistor MOS ocupa mucha menos superficie en la pastilla de silicio que un transistor bipolar similar.

En la FIG. 1.5 se dan los ejemplos para el caso en que se utilicen transistores en lugar de diodos.

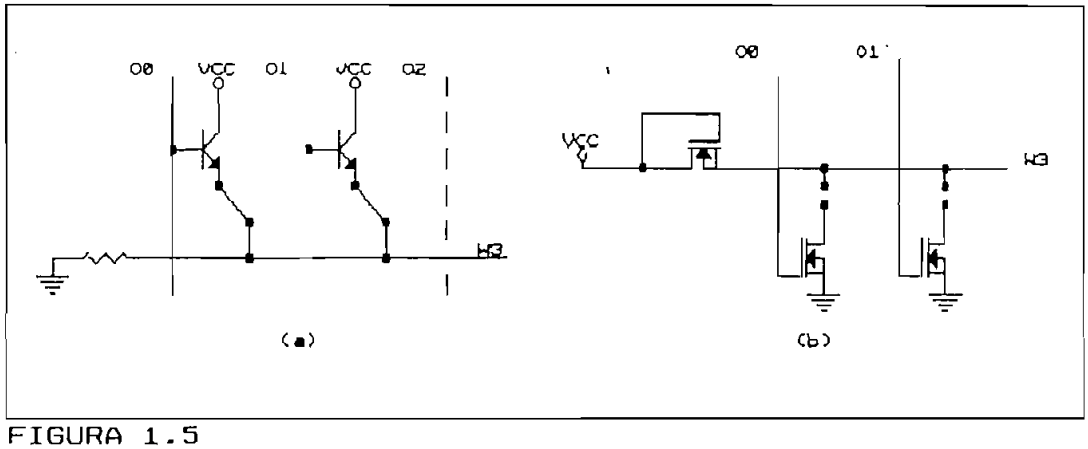


FIGURA 1.5

Como una modificación de las memorias ROM se dispone también las siguientes:

1.1.2.2 Memorias PROM

Estos dispositivos son del tipo ROM programables por el

usuario, generalmente corresponden a dispositivos bipolares que usan uniones fusibles, las cuales son derretidas por una corriente relativamente alta (20-30 mA) para romper las conexiones circuitales donde se desea y de esta manera programar la memoria. Así pues, el usuario de la PROM puede destruir los fusibles cuando sea necesario, dejando conectados los diodos o transitores en las posiciones en que se desea un 1L.

1.1.2.3 Memorias EPROM

Una vez que se ha grabado una ROM o PROM el proceso es irreversible y un cambio requiere de un nuevo dispositivo; por esa razón aparece un nuevo tipo de memoria que es llamada PROM borrable o EPROM, la misma que puede ser recuperada a su condición inicial colocándola bajo la incidencia de luz ultravioleta, la radiación de onda corta retira las cargas almacenadas en su compuerta y regresa a su estado inicial.

EPROM es un tipo de memoria PROM borrable y alterable su contenido, y utiliza transitores FAMOS que son transistores que tienen una compuerta llamada flotante que esta separada de cualquier conexión eléctrica en el circuito, como se ilustra en la FIG. 1.6(a). En estos transistores es posible establecer una carga en la compuerta aplicando una tensión elevada entre el drenaje y la fuente. La carga almacenada hace que se genere el canal del transistor, como se muestra en la FIG. 1.6(b); en la FIG. 1.6(c) se indica su símbolo.

La forma de operación de las memorias FAMOS (Floating-gate Avalancha - injection MOS) depende de la carga almacenada por inyección de avalancha de electrones.

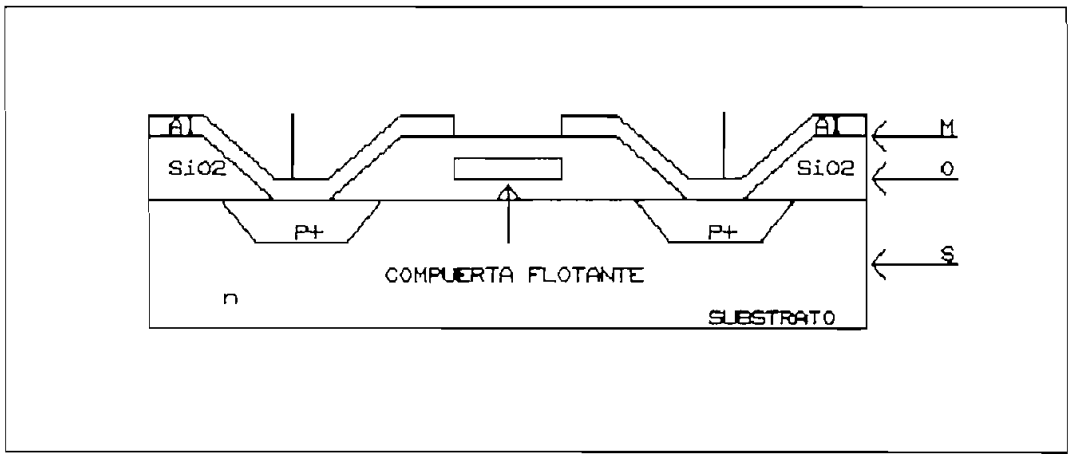


FIGURA 1.6(a)

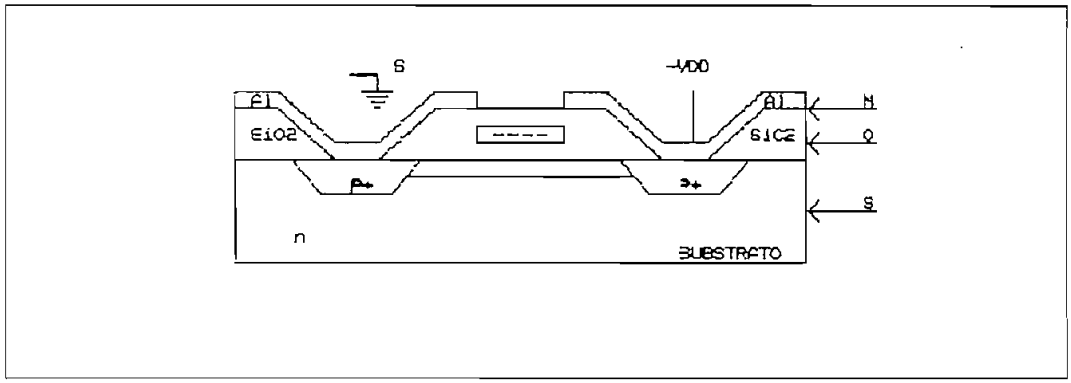


FIGURA 1.6(b)

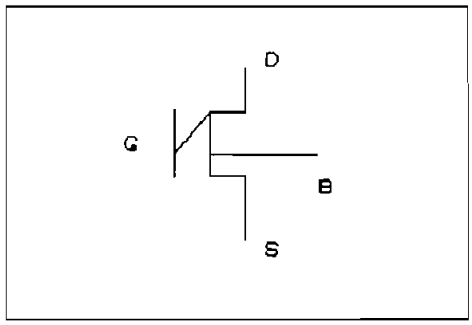


FIGURA 1.6(c)

Al realizar el borrado o regreso a su estado inicial, todos los bits de las localidades se ponen en 1L; para el proceso

de borrado se utiliza la iluminación del empaquetado con luz ultravioleta, la cual provoca un retiro de las cargas guardadas en la compuerta flotante (UVPROM); una memoria similar a la EPROM es la EAPROM (Electrically Alterable PROM).

La información generada en estas memorias se mantiene por períodos muy largos por lo que tienen la cualidad de no ser volátiles, igual que en una memoria ROM cualquiera; sin embargo, tienen una pequeña descarga del orden de 20 a 30% en 10 años.

Un diagrama de bloques de la configuración interna de la memoria 2716 se da en la FIG. 1.7. (2)

La designación de las señales que se pueden observar en la FIG. 1.7 es:

Ao - A10	líneas de dirección
CE/PGM	habilitación del circuito/programación
OE	Habilitación de salida
Oo - O7	líneas de salida.

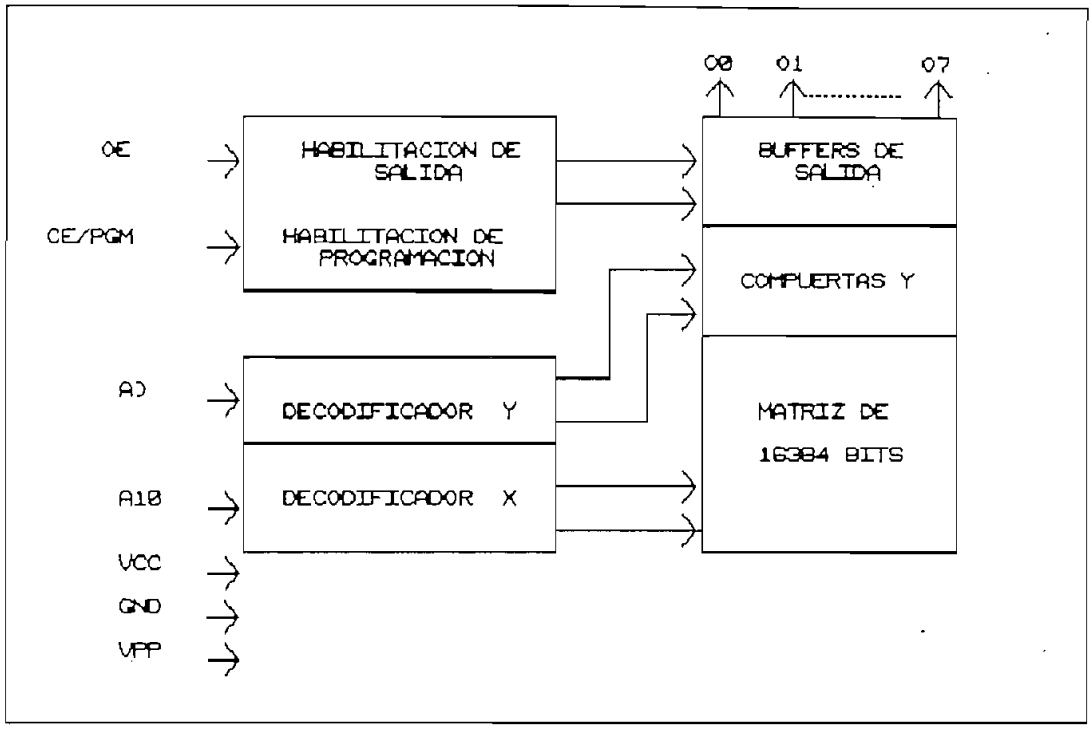


FIG. 1.7

Las características dinámicas de este chip (UVEPROM 2716) se dan con la ayuda del diagrama de tiempos de la FIG. 1.8. Los datos de los parámetros indicados en la FIG. 1.8, son los siguientes:

TAcc: tiempo desde una dirección válida hasta una salida válida (450 ns máximo), tiempo de acceso.

TCE: tiempo desde la habilitación del chip hasta una salida válida (450 ns máximo).

TOE: tiempo desde la habilitación de las salidas hasta una salida válida (120 ns máximo).

TDF: tiempo desde la deshabilitación de las salidas hasta que la salida se pone en alta impedancia (100 ns máximo).

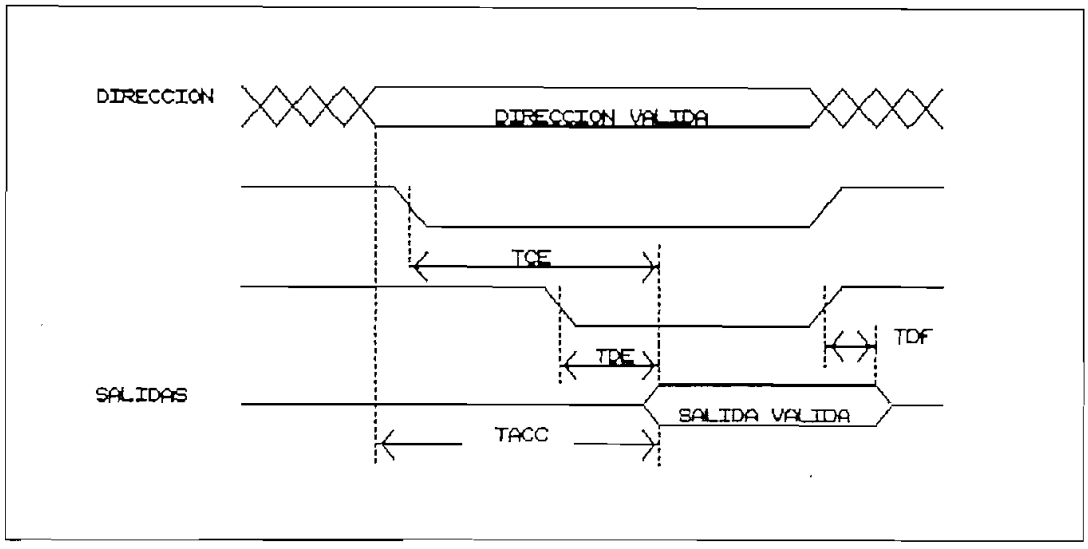


FIG. 1.8

1.1.3 Modos de Operación (2)

Son cinco los modos de operación de las memorias los cuales son:

a) Lectura

b) Stanby

c) Programación

d) Verificación

e) Inhibición de Programación

Se requiere una fuente de alimentación de 5 [V]. Todas las entradas son niveles TTL excepto para el pulso de entrada en V_{PP} durante la programación del chip.

a) Lectura

Existen dos entradas de control para que la memoria pueda ser usada en modo de lectura, de tal manera que se puedan obtener los datos en las salidas. Chip Enable (CE) es el control para selección del chip. Output Enable (OE) permite obtener datos en los pines designados para la salida.

Asumiendo que las entradas a las direcciones son fijas, el tiempo de acceso es igual al retardo desde habilitación del chip hasta tener las salidas (tCE). Los datos que se encuentran en la dirección especificada son disponibles a las salidas despues del flanco negativo de OE, asumiendo que CE estaba en bajo y la dirección especificada fue estable por lo menos tACC - tOE.

b) Standby

Las EPROMS pueden ser puestas en modo Standby el cual reduce al máximo el consumo de corriente del diseño aplicando una señal TTL en alto en la entrada CE. Cuando se encuentra en modo standby, las salidas están en estado de alta impedancia, independiente del estado de la entrada de OE.

c) Programación

Una de las precauciones especiales que debe considerarse es que voltajes más elevados de los que se necesitan para la programación producirán daños permanentes en la memoria.

Inicialmente y después de cada borrado, todos los bits de la EPROM están en "1". Los datos son introducidos selectivamente programando "0" en los bits respectivos. Aunque solamente "0" serán programados, ambos "1" y "0" son presentados en la palabra.

La única manera de cambiar un "0" a "1" es por medio del borrado ultravioleta.

La memoria se encuentra en el modo de programación cuando la entrada V_{pp} está en el nivel especificado de acuerdo al

voltaje de programación que especifique el fabricante. Se requiere que un capacitor de 0.1µF se coloque entre V_{PP} y tierra para suprimir esporádicos transitorios de voltaje los cuales pueden dañar el elemento. Los datos a ser programados son aplicados en forma paralela de 8 bits a los pines de salida de datos. Los niveles requeridos para direccionamiento y datos son TTL.

Cuando la dirección y datos son estables, un pulso cuya duración depende del algoritmo, es aplicado a la entrada de CE. Un pulso debe ser aplicado para cada dirección a ser programada. Cualquier localidad puede ser programada a cualquier tiempo, secuencialmente o en forma aleatoria.

Resulta fácil la programación con los mismos datos de memorias en paralelo por la simplicidad de los requerimientos de programación.

d) Verificación

Una vez realizada la programación de un byte se procede a la lectura del mismo con el objeto de ver si el dato grabado es correcto. Esta operación se la realiza para comprobar que el dato que se grabó es el mismo que se deseó; en caso de que la verificación no sea correcta será un indicativo para detener el proceso de programación. Gracias a ésto se previene daños del equipo y además evita el seguir con el proceso en caso de que la memoria no se encuentre en blanco. Para la verificación tanto CE como OE deben tener un nivel lógico en bajo.

e) Inhibición de programación

Esta opción se puede usar cuando se desea programar en paralelo algunas memorias EPROM pero con diferentes datos. Un nivel alto en CE, inhibe las otras EPROMS.

1.1.4 PROGRAMACION DEPENDIENDO DE LA DURACION DEL PULSO (3)

1.1.4.1 Programación Standard

Para la programación, CE deberá permanecer en bajo (habilitado), mientras V_{pp} (voltaje de programación) se mantiene en el nivel deseado. Cuando la dirección y los datos son estables será necesario un pulso de 50 ms, activo en bajo, mientras el voltaje de programación V_{pp} es aplicado. Un pulso debe ser aplicado para cada dirección a ser programada. Cualquier localidad puede ser programada a cualquier tiempo, secuencialmente o en forma aleatoria. El pulso tiene como ancho máximo 55 ms. y mínimo 45 ms. Para este tipo de programación Vcc debe estar comprendido entre 4.75 y 5.25 voltios.

1.1.4.2 Algoritmo Inteligente de Programación

El algoritmo inteligente de programación es el método preferido de programación puesto que permite realizar el proceso de programación mucho más rápido que el standard con los 50 ms por cada byte. En este caso el pulso para la programación tiene una duración de 1 milisegundo, con un límite mínimo de 0.95 milisegundos y un máximo de 1.05 milisegundos. Para este modo de programación Vcc debe estar comprendido entre 5.75 y 6.25 voltios. Los diagramas de tiempo para la programación y verificación se indican en los anexos.

Las memorias, para las que se desarrolló el prototipo de programador que es el objeto de este trabajo de tesis, son del tipo EPROM comerciales: 2716, 2732, 2764, 27128, 27256 y 27512.

La capacidad de las memorias indicadas es:

Memoria	Capacidad
2716	2Kx8
2732	4Kx8
2764	8Kx8
27128	16Kx8
27256	32Kx8
27512	64Kx8

NOTA: 1K = 1024 bytes

La FIG. 1.9 indica la designación numérica de los pines de las memorias a ser programadas, la memorias 2716 y 2732 son de 24 pines y el resto de 28 pines. La disposición permite apreciar que GND de las memorias de 24 pines que corresponde al pin 12, queda en superposición con GND de las memorias de 28 pines y que corresponde al pin 14.

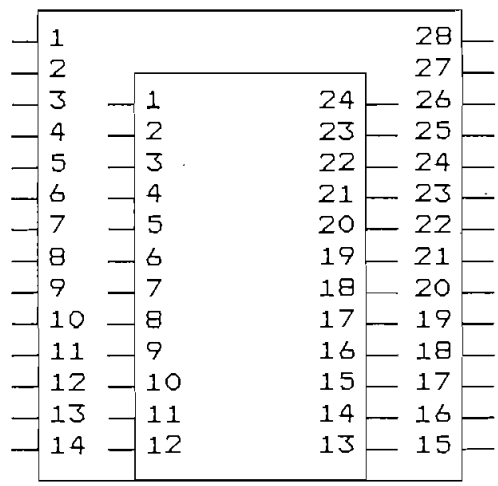


FIGURA 1.9

Gracias a la disposición anterior, y considerando la numeración de pines de la memoria de 24 pines como que fuera de 28, (es decir el pin 12 de la memoria de 24 pines se considerará como el catorceavo pin), se desarrolló el siguiente cuadro el mismo que permitirá conocer cuales son las condiciones en las que se deben encontrar los distintos pines de las memorias para poder realizar operaciones de lectura y de programación, que son las dos operaciones fundamentales en el equipo. Esto se ha hecho con el objeto de utilizar un solo zócalo para alojar a cualquiera de las memorias a programar.

	CS	OE/V_{PP}	A11/V_{PP}	V_{PP}	PGM
PIN	20	22	23	1	27
2716					
Read	LOW	LOW	HI	X	X
Prog	PULSO	HI	V_{PP}	X	X
2732					
Read	LOW	LOW	A11	X	X
Prog	PULSO	V_{PP}	A11	X	X
2764					
Read	LOW	LOW	A11	HI	HI
Prog PULSO	LOW	HI	A11	V_{PP}	
27128					
Read	LOW	LOW	A11	HI	HI
Prog PULSO	LOW	HI	A11	V_{PP}	
27256					
Read	LOW	LOW	A11	HI	A14
Prog	LOW	HI	A11	V_{PP}	A14
27512					
Read	LOW	LOW	A11	HI	A14
Prog	LOW	HI	A11	V_{PP}	A14

La condición X es una condición no importa; pudiendo ser 1_L ó 0_L.

El voltaje V_{pp} indica que en ese pin debe aplicarse el voltaje de programación, el cual depende del tipo de memoria que se disponga. V_{pp} puede tomar los valores de 12.5[V], 21[V] ó 25[V].

El duración del pulso depende del tipo de algoritmo que se esté utilizando en la programación.

1.2 PROGRAMACION DE MICROCONTROLADORES DE LA FAMILIA MCS51[4]

Entre las características de los microcontroladores de la familia MCS-51 que pueden ser programados tenemos:

EPROM Versión	ROM Bytes	RAM Bytes	16-bit Timers	Ckt Type	Vpp
8751H, 8751BH	4K	128	2	HMOS	21V/12.75V
8752BH	8K	256	3	HMOS	12.75V
87C51	4K	128	2	CHMOS	12.75V

TABLA 1

1.2.1 PROGRAMACION DE LA EPROM DEL 8751H

Para programar la EPROM es necesario utilizar un oscilador de 4-6 MHz. (La razón de necesitar este oscilador es que el bus interno del microcontrolador está siendo usado para transferir direcciones y datos a registros internos apropiados. Al realizar esta operación de transferencia el microcontrolador entra en funcionamiento, siendo necesario un cristal; el mismo que determina la velocidad de operación del microcontrolador). La dirección de la EPROM a ser programada es aplicada al puerto 1 y pines P2.0-P2.3

del puerto 2, mientras que la palabra a ser programada se coloca en el puerto 0. Los otros pines del puerto 2, RST, PSEN y EA deben permanecer en los niveles que se indica en la tabla 2. ALE es colocado en bajo por 50 ms. para programar el byte en la dirección especificada.

Mode	RST	PSEN	ALE	EA	P2.7	P2.6	P2.5	P2.4
Program	1	0	0*	V_{pp}	1	0	X	X
Inhibit	1	0	1	X	1	0	X	X
Verify	1	0	1	1	0	0	X	X
Secu. Set	1	0	0*	V_{pp}	1	1	X	X

TABLA 2

"1" = Nivel lógico alto para cada pin

"0" = Nivel lógico bajo para cada pin

"X" = Condición no importa

"Vpp" = +21 +/- 0.5 [V]

*ALE es pulsado en bajo por 50 ms.

Si el bit de seguridad no ha sido programado, es posible realizar la lectura para propósitos de verificación, esto lo puede realizar durante o después del proceso de programación. Para poderlo hacer la dirección de la localidad de memoria a ser leída debe ser aplicada al puerto 1 y a los pines P2.0 - P2.3. Los otros pines deberán ser retenidos en niveles de verificación indicados en la tabla 2. El contenido de la localidad de memoria aparecerá en el puerto 0. Para realizar la operación de lectura es necesario utilizar resistencias de pullup externas en el puerto 0, de 10K por recomendación del fabricante.

Existe un bit de seguridad, el cual una vez que ha sido programado impide el acceso eléctrico al programa almacenado en la memoria. El procedimiento a seguir para programar este bit de seguridad es el mismo que para programar la EPROM, excepto que P2.6 debe ser retenido en un nivel alto. Puerto 0, Puerto 1 y los pines P2.0-P2.3 pueden permanecer en cualquier estado. Los otros pines deben permanecer en los niveles indicados en la tabla 2. Cuando el bit de seguridad ha sido programado, este puede ser borrado únicamente borrando toda la memoria del programa. Cuando el bit de seguridad es programado, la memoria interna de programa no puede ser leída, además el chip no puede ser programado, y no se puede ejecutar programa de memoria externa. Una vez que se borre la EPROM, se quita el bit de seguridad, y el chip se hace totalmente funcional, pudiendo este ser reprogramado.

1.2.2 PROGRAMACION DE LA EPROM DEL 8751BH

Para programar la EPROM es necesario utilizar un oscilador de 4-6 MHz. (La razón de necesitar este oscilador es que el bus interno del microcontrolador está siendo usado para transferir direcciones y datos a registros internos apropiados). La dirección de la EPROM a ser programada es aplicada al puerto 1 y pines P2.0-P2.3 del puerto 2, mientras que la palabra a ser programada se coloca en el puerto 0. Los otros pines del puerto 2 y 3, y RST, PSEN y EA/V_{pp} deben permanecer en los niveles que se indica en la tabla 3. ALE/PROG es colocado en bajo para programar el

byte en la dirección especificada.

Normalmente EA/V_{pp} es retenida en un nivel lógico alto hasta justo antes de que ALE/PROG va a ser pulsado. Luego EA/V_{pp} es llevado a V_{pp}, ALE/PROG es pulsado en bajo, y posteriormente EA/V_{pp} es llevado a un nivel válido alto. Formas de onda pueden apreciarse en los anexos.

EA/V_{pp} no puede ser llevado más allá del nivel especificado por mucho tiempo. Picos de voltaje pueden causar daños al chip. Es por eso que la fuente para V_{pp} debe ser regulada y libre de perturbaciones.

El 87518H puede ser programado usando el algoritmo con pulso rápido de programación para microcontroladores.

Las dos características más relevantes son el uso de un más bajo V_{pp} (12.75 voltios comparado con los 21 voltios del 8751H) y un pulso de programación más corto.

Para la programación las condiciones son las siguientes: V_{pp} debe ser 12.75 +/- 0.25 volts. ALE/PROG es pulsado en bajo por 100 µsegundos, 25 veces. Posteriormente el byte debe ser verificado. Una vez programado totalmente el arreglo puede ser verificado.

Mode	RST	PSEN	ALE	EA	P2.7	P2.6	P3.6	P3.7
Program	1	0	0*	V_{pp}	1	0	1	1
Secu. Set	1 1	0 0	0* 0*	V_{pp} V_{pp}	1 1	1 1	1 0	1 0
Verify	1	0	1	1	0	0	1	1

TABLA 3

"1" = Nivel lógico alto para cada pin

"0" = Nivel lógico bajo para cada pin

"X" = Condición no importa

"Vpp" = +12.75 +/- 0.25 [V]

*ALE es pulsado en bajo por 100 µs para programación (mediante el algoritmo rápido de programación).

Para programar los bits de seguridad es necesario que se pongan los niveles de voltaje en los pines respectivos indicados en la tabla 3.

Si los bits de seguridad no han sido programados es posible realizar la lectura para propósitos de verificación, esto lo puede realizar durante o después del proceso de programación. Para poderlo hacer la dirección de la localidad de memoria a ser leída debe ser aplicada al puerto 1 y a los pines P2.0 - P2.3. Los otros pines deberán ser retenidos en niveles de verificación indicados en la tabla 3. El contenido de la localidad de memoria aparecerá en el puerto 0. Para realizar la operación de lectura es necesario utilizar resistencias de pullup externas en el puerto 0, de 10k por recomendación del fabricante.

Este chip viene en un empaque plástico sin una ventana y, por lo tanto no puede ser borrado.

Al grabar los "Lock bits" tomar en cuenta las características de estos. La tabla 4 indica las características de los bits de seguridad:

BITS SEGURIDAD		HABILITACION LOGICA
LB1	LB2	
U	U	Sin bits de seguridad es posible realizar empaquetado del programa.
P	U	MOVC instrucciones ejecutadas desde memoria externa son deshabilitados, pudiendo solamente ser realizada esta operación desde la memoria interna, EA es retenido y mantenido durante el reset, y además la programación de la EPROM es deshabilitada
P	P	Igual que la anterior pero además verificación es también deshabilitada
U	P	Reservada para futura definición

TABLA 4

P = Programado

U = No programado

1.2.3 PROGRAMACION DE LA EPROM DEL 87C51

El 87C51 es programado por un algoritmo con pulso rápido de programación. Este difiere de otros viejos métodos en el valor usado para V_{pp} (Fuente de voltaje para programación) y en el ancho y número de pulsos que se dan en ALE/PROG.

El 87C51 contiene 3 bytes que pueden ser leidos y usados para la identificación del chip.

La tabla 5 muestra los niveles de voltaje en los diferentes pines para programación, verificación y bits de seguridad.

Mode	RST	PSEN	ALE	EA	P2.7	P2.6	P3.6	P3.7
Program	1	0	0*	V_{pp}	1	0	1	1
Secu. Set	1 1 1	0 0 0	0* 0* 0*	V_{pp} V_{pp} V_{pp}	1 1 0	1 1 1	1 0 1	1 0 0
Verify	1	0	1	1	0	0	1	1

TABLA 5

"1" = Nivel lógico alto para cada pin

"0" = Nivel lógico bajo para cada pin

"Vpp" = +12.75 +/- 0.25 [V]

*ALE es pulsado en bajo por 100 µs para programación (mediante el algoritmo rápido de programación).

De igual manera a los dos anteriores existe requerimiento de oscilador de 4 a 6 MHz. y para las distintas posibilidades que se presentan es necesario tomar en cuenta los niveles de voltaje los mismos que se presentan en la tabla 5.

Para programar el 87C51 se debe seguir los siguientes pasos:

1. Poner la dirección válida en las líneas de direcciones.
2. Ingresar el byte de dato en las líneas de datos.
3. Activar la correcta combinación de señales de control.
4. LLevar EA/V_{pp} desde V_{cc} a 12.75V +/- 0.25V
5. Pulse ALE/PROG 5 veces* para el arreglo de la EPROM, y 25 veces para la tabla de empaquetado y los bits de seguridad.

* Para compatibilidad 25 pulsos pueden ser usados.

Repetir los pasos desde el 1 hasta el 5 cambiando la dirección y datos, hasta el final del archivo objeto.

Este chip dispone de 3 bits de seguridad, la tabla 6 indica las características de los Lock bits.

PROGRAM LOCK BITS				TIPO DE PROTECCION
	LB1	LB2	LB3	
1	U	U	U	Sin protección
2	P	U	U	Instrucciones MOVC de memoria externa son deshabilitadas, quedando la posibilidad únicamente a nivel de memoria interna. EA es mantenido en reset y la programación de la memoria es deshabilitada.
3	P	P	U	Igual que la anterior pero además la verificación es también deshabilitada.
4	P	P	P	Igual que la 3 pero además ejecución externa es deshabilitada.

TABLA 6

P = Programado

U = No programado

Cualquier otra combinación de los lock bits no está definida.

Nota: Antes de decidir programar los lock bits, primeramente conocer las limitaciones a las que se expone. Cuando lock bit 1 es programado, el nivel lógico de EA es muestreado y retenido durante el reset. Si el diseño es encendido sin un reset, el latch inicializa en un valor randómico, y se mantiene ese valor hasta que reset sea activado. Es necesario que el valor retenido de EA esté de acuerdo con el nivel lógico que se ponga en el pin del

chip, de tal manera que el chip funcione apropiadamente.

1.2.4 PROGRAMACION DE LA EPROM DEL 8752BH

Al igual que los anteriores es necesario utilizar un oscilador de 4-6 MHz. La dirección de la EPROM a ser programada es aplicada al puerto 1 y pines P2.0-P2.4 del puerto 2, mientras que la palabra a ser programada se coloca en el puerto 0. Los otros pines del puerto 2 y 3, RST, PSEN y EA deben permanecer en los niveles que se indica en la tabla 7. ALE/PROG es colocado en bajo para programar el byte en la dirección especificada.

Normalmente EA/V_{pp} es retenida en un nivel lógico alto hasta justo antes de que ALE/PROG va a ser pulsado. Luego EA/V_{pp} es llevado a V_{pp}, ALE/PROG es pulsado en bajo, y posteriormente EA/V_{pp} es llevado a un nivel lógico alto. Las formas de onda pueden apreciarse en los anexos.

EA/V_{pp} no puede ser llevado más allá del nivel especificado por mucho tiempo. Picos de voltaje pueden causar daños al chip. Es por eso que la fuente para V_{pp} debe ser regulada y libre de perturbaciones.

Mode	RST	PSEN	ALE	EA	P2.7	P2.6	P3.6	P3.7
Program	1	0	0*	V_{pp}	1	0	1	1
Secu. Set	1 1	0 0	0* 0*	V_{pp} V_{pp}	1 1	1 1	1 0	1 0
Verify	1	0	1	1	0	0	1	1

TABLA 7

El 8752BH puede ser programado usando el algoritmo con pulso rápido de programación para microcontroladores. Las

características del nuevo método de programación son el uso de un más bajo V_{pp} (12.75 voltios comparado con los 21 voltios) y un pulso de programación más corto.

Para la programación las condiciones son las siguientes: V_{pp} debe ser 12.75 +/- 0.25 volts. ALE/PROG es pulsado en bajo por 100 µsegundos, 25 veces.

Si los bits de seguridad no han sido programados. Es posible realizar la lectura para propósitos de verificación, esto lo puede realizar durante o después del proceso de programación. Para poderlo hacer la dirección de la localidad de memoria a ser leída debe ser aplicada al puerto 1 y a los pines P2.0 - P2.4. Los otros pines deberán ser retenidos en niveles de verificación indicados en la tabla 7. El contenido de la localidad de memoria aparecerá en el puerto 0. Para realizar la operación de lectura es necesario utilizar resistencias de pullup externas en el puerto 0.

La tabla 8 indica las características de los bits de seguridad:

BITS SEGURIDAD		HABILITACION LOGICA
LB1	LB2	
U	U	Sin bits de seguridad es posible realizar empaquetado del programa.
P	U	MOVC instrucciones ejecutadas desde memoria externa son deshabilitados, pudiendo solamente ser realizada esta operación desde la memoria interna, EA es retenido y mantenido durante el reset, y además la programación de la EPROM es deshabilitada
P	P	Igual que la anterior pero además verificación es también deshabilitada
U	P	Reservada para futura definición

TABLA 8

P = Programado

U = No programado

Al realizar el proceso de borrado el elemento se hace totalmente funcional.

1.3 PROGRAMACION DE MICROCONTROLADORES DE LA FAMILIA MCS48 (4)

Entre las características de los microcontroladores de la familia MCS-48 que pueden ser programados tenemos:

CHIP	MEMORIA INTERNA	MEMORIA RAM
8748/8748H	1K x 8 EPROM	64 x 8
8749/8749H	2K X 8 EPROM	128 x 8

TABLA 9

1.3.1 PROGRAMACION DE LA EPROM DEL 8748 Y DEL 8749 (5)

En breve, el proceso de programación consiste de: activando el modo de programa, aplicando una dirección, reteniendo la dirección, aplicar los datos y luego aplicar un pulso de programación. Cada palabra de información puede ser grabada y verificada.

La tabla 10 indica una lista de los pines usados para la programación y una descripción de sus funciones:

PINES	FUNCION
XTAL 1	Clock input (1 to 6 Mhz)
XTAL 2	
RESET	Inicialización y retención de direcciones
TEST 0	Selección de modo de programación/verific.
EA	Activación de los modos de program./verific.
BUS	Entrada de direcciones y datos. Salida de datos durante verificación
P20-P22	Entrada de direcciones
V_{DD}	Fuente de alimentación para programación
PROG	Entrada de pulso de programación

TABLA 10

El chequeo del ALE como paso previo para realizar la programación previene al equipo el continuar con el proceso de programación o lectura en el caso de que el chip se encuentre dañado; gracias a esto pueden evitarse daños en el programador. Así también, el no tener la señal del ALE puede ser indicativo de que el socket en el que ha sido colocado el microcontrolador no es el correcto, de esta manera se previene daños del chip y del equipo. Por lo tanto, el no tener esta señal de reloj puede ser utilizado para deshabilitar al programador.

La secuencia para la programación/verificación es:

1. V_{DD} = 5 V, señal de reloj aplicada ó oscilador

interno operando, RESET = 0 [V], TEST 0 = 5 [V], EA = 5 [V] BUS and PROG flotantes.

2. Insertar el 8748 (8749) en el socket de programación
3. TEST 0 = 0 voltios (seleccionado modo de programación)
4. EA = 23 voltios (activado modo de programación)
5. Aplicar la dirección al BUS y P20-P22
6. RESET = 5 voltios (retener la dirección)
7. Aplicar los datos al BUS
8. V_{DD} = 25 voltios (fuente de alimentación de programación)
9. PROG = 0 voltios seguido por un pulso de 50 ms a 23 voltios
10. V_{DD} = 5 voltios
11. TEST 0 = 5 voltios (modo de verificación)
12. Leer y verificar datos en el bus
13. TEST 0 = 0 voltios
14. RESET = 0 voltios y repetir desde el paso 5
15. El momento que se saque el 8749 (8748) del socket, el programador debe encontrarse en las condiciones del paso 1.

1.3.2 PROGRAMACION DE LA EPROM DEL 8748H Y DEL 8749H (4)

En breve, el proceso de programación consiste de: activando el modo de programa, aplicando una dirección, reteniendo la dirección, aplicar los datos y luego aplicar un pulso de programación. Cada palabra de información puede ser grabada

y verificada.

La tabla 11 indica una lista de los pines usados para la programación y una descripción de sus funciones:

PINES	FUNCION
XTAL 1 XTAL 2	Clock input (3 to 4 Mhz)
RESET	Inicialización y retención de direcciones
TEST 0	Selección de modo de programación/verific.
EA	Activación de los modos de program./verific.
BUS	Entrada de direcciones y datos. Salida de datos durante verificación
P20-P22	Entrada de direcciones
V_{DD}	Fuente de alimentación para programación
PROG	Entrada de pulso de programación

TABLA 11

Un intento de programar en un socket equivocado producirá un daño en el chip. Una indicación de tener un socket apropiado es la aparición de la señal de salida ALE. El no tener esta señal de reloj puede ser utilizado para deshabilitar al programador.

La secuencia para la programación/verificación es:

1. V_{DD} = 5 V, señal de reloj aplicada ó oscilador interno operando, RESET = 0 [V], TEST 0 = 5 [V], EA = 5 [V] BUS and PROG flotantes. P10 y P11 deben ser llevados a tierra.
2. Insertar el 8749H (8748H) en el socket de programación
3. TEST 0 = 0 voltios (seleccionado modo de programación)
4. EA = 18 voltios (activado modo de programación)
5. Aplicar la dirección al BUS y P20-P22

6. RESET = 5 voltios (retener la dirección)

7. Aplicar los datos al BUS

8. V_{DD} = 21 voltios (fuente de alimentación de programación)

9. PROG = Vcc voltios o flotante seguido por un pulso de 18 voltios con duración de 55 ms.

10. V_{DD} = 5 voltios

11. TEST 0 = 5 voltios (modo de verificación)

12. Leer y verificar datos en el bus

13. TEST 0 = 0 voltios

14. RESET = 0 voltios y repetir desde el paso 5

15. El momento que se saque el 8749H (8748H) del socket, el programador debe encontrarse en las condiciones del paso 1.

FORMATO INTEL (6)

El formato usado en los archivos objeto (.HEX) para la programación de memorias y microcontroladores es el formato INTEL que se detalla a continuación:

Designación del Formato Descripción	Nº de Caracteres ASCII	
Cabecera	1	Siempre ":"
Bytes a Grabar	2	2 dígitos HEX especifican el # de datos a ser grabados o indican fin de grabado.
Starting	4	Indicado mediante 4 dígitos HEX representa la dirección de la localidad de memoria

		donde se va a localizar el primer dato.
Separador	1	00
CHECK SUM	2	2 dígitos HEX que representa el complemento a 1 de la suma de los bytes precedentes.

A continuación se indica con un ejemplo el formato INTEL:

:10000000F43638F737D73430333E393742FE4A256F

La interpretación es la siguiente:

:	indica la cabecera
10	indica que son 16 los datos de esa línea
0000	estos cuatro dígitos hexadecimales indican que la dirección de inicio de los 16 datos es la 0000H
00	estos dos dígitos hexadecimales representan el separador
F43638F737D73430333E393742FE4A25	estos 32 dígitos son los 16 datos de la línea. (Cada dato se compone de 2 dígitos hexadecimales)
6F	indica el valor del check-sum.

CARACTERISTICAS DE BORRADO (3.4)

Las características de borrado son tales que el mismo empieza a ocurrir bajo exposición de luz con longitudes de onda corta de aproximadamente 4000 Angstroms (A). Cabe anotar que los rayos solares y ciertos tipos de lámparas fluorescentes tienen longitudes de onda en el orden de 3000 a 4000 Angstroms. Experiencias demuestran que exposiciones

permanentes en un cuarto con luz fluorescente podría borrar la EPROM en aproximadamente 3 años, mientras que esto puede tomar aproximadamente una semana en el caso de exposición directa a la luz solar. En el caso que el chip vaya a ser usado en este tipo de condiciones por periodos de tiempo largos, se recomienda utilizar etiquetas opacas que deben ser colocadas sobre la ventana para prevenir borrados no intencionales.

El proceso de borrado recomendado se logra exponiendo la memoria a luz ultravioleta la misma que tiene una longitud de onda de 2537 Angstroms (A). La dosis a la que la se expone la memoria viene dada por la intensidad x tiempo de exposición a la luz. Para lograr un buen borrado la energía mínima requerida debe ser de 15 Wsec/cm². Así por ejemplo para una lámpara con luz ultravioleta de 12000μW/cm² de razón de potencia el tiempo de exposición es de 15 a 20 minutos. La EPROM debe ser colocada a una distancia de 1 pulgada [2,54cm] durante el proceso de borrado. La máxima exposición que puede soportar una memoria sin dañarse es de 7258 Wsec/cm² (1 semana con 12000μW/cm²). Exposiciones por periodos mayores provocan daños permanentes en la memoria. Una vez realizado el borrado, en el caso de las memorias EPROM y de los microcontroladores de la familia del 51 el arreglo queda con todas las localidades de memoria en "1␣"; mientra que en los microcontroladores de la familia del 48 las localidades de memoria quedan en "0␣".

Tanto las memorias como los microcontroladores se colocan

en zócalos energizados; por esta razón, en el caso de que las memorias o los microcontroladores, a ser programados, se encuentren quemados pueden producir daños en el equipo. Cuando se procede a programar una memoria o un microcontrolador de la familia MCS51 quemado, voltajes de programación aparecen y esto puede dar problemas al equipo. En este sentido los microcontroladores de la familia MCS48, gracias a la señal del ALE, permiten paralizar el proceso de programación en el caso de que esta señal de reloj no sea verificada, esto resulta una gran ayuda de protección para el equipo. Cualquier operación de lectura o programación debe ser interrumpida en el caso de que no se tenga la señal del ALE.

CAPITULO II

DISEÑO DEL HARDWARE

2.1 REQUERIMIENTOS GENERALES

Para cumplir el objetivo propuesto en este trabajo es necesario tomar en cuenta los siguientes requerimientos para el equipo de programación:

1.- Interface para comunicación serial entre el computador personal y el programador, de norma EIA-RS232.

2.- Fuentes de alimentación reguladas para voltajes de programación y alimentación a los chips.

3.- Señales de control, gracias a las cuales es posible obtener los voltajes adecuados para la programación; así como también generación de señales que sean necesarias durante el proceso de programación.

4.- Adicionalmente es necesario direccionar localidades de memoria y un bus para datos (sean estos leídos o datos a ser programados).

1.- La comunicación entre el computador y el programador se realiza vía serial, para lo cual será necesario adecuar las señales CMOS del puerto serial del computador a niveles TTL utilizados en el microcontrolador, y viceversa.

2.- Las fuentes de alimentación, dependiendo de los chips a programar, son las siguientes:

Para memorias se requiere de los siguientes voltajes de programación (=):

- 12.75 +/- 0.25 [V]
- 21 +/- 0.5 [V]

- 25 [V] +/- 1 [V]

Uno de estos voltajes es seleccionado para ser utilizado en la programación, esto depende de la memoria que se disponga. Por defecto el voltaje a ser utilizado será de 21 +/- 0.25 [V]. Este voltaje puede ser seleccionado por el usuario, la forma de selección se indicará en el siguiente capítulo.

Además, para la polarización de las memorias, debe disponerse de una fuente de alimentación que permita variar el voltaje de salida; lo cual depende del algoritmo usado para la programación, teniendo valores de: 5+/-0.25 [V] y 6+/-0.25[V]

Para los microcontroladores de la familia MCS51 los voltajes requeridos son los siguientes:

- 21 +/- 0.5 [V]
- 12.75 +/- 0.25 [V]

Estos voltajes se utilizan dependiendo del elemento de la familia MCS51, que se está programando; estos voltajes no son seleccionados por el usuario, pues van determinados de acuerdo al elemento de la familia, tal como se especificó en el capítulo I.

Para los microcontroladores de la familia MCS48 los voltajes son los siguientes:

- Para EA/PROG:

  23 +/- 1.5 [V]

  18 +/- 0.5 [V]

- Para VDD:

25 +/- 1 [V]

21 +/- 0.5 [V]

Estos voltajes se utilizan dependiendo del elemento de la familia MCS48 que se está programando; estos voltajes no son seleccionados por el usuario.

Adicionalmente se requiere una fuente de 5[V] para alimentación de los chips a usarse en el equipo.

3.- Las señales de control que se requieren dependen del elemento que se encuentre programando; la secuencia de las mismas, durante la programación, se encuentra en los anexos.

Para el caso de memorias, considerando la superposición de pines, que se presentó en el capítulo 1, entre las memorias de 24 y 28 pines; los pines y sus niveles de voltaje que necesitan son los siguientes:

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
20	CS	LOW/PULSO (HI/LOW)
22	OE/V_{pp}	LOW/HI/V_{pp}
23	A11/V_{pp}	DIRECCION A11/HI/V_{pp}
1	V_{pp}	HI/V_{pp}
27	PGM	DIRECCION A14/HI/LOW
26	A13/VCC	DIRECCION A13/VCC

Para el caso de los microcontroladores de la familia del 51 los pines sobre los que es necesario actuar para el proceso de programación son los siguientes:

RST PSEN ALE EA P2.6 P2.7 P3.6 y P3.7

Mediante el control sobre estos pines es posible realizar

las distintas tareas para las que se encuentra diseñado el programador.

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
4	RST	Permanece en Vcc
29	PSEN	Permanece en GND
30	ALE	Sobre este pin se aplica el pulso de programación, dependiendo su duración del elemento de la familia MCS51 (HI/LOW)
31	EA	En este pin se aplica el voltaje de programación cuyo valor depende del elemento a programar. (V_{pp})
27-28	P2.6 P2.7	
17-18	P3.6 P3.7	En estos pines se aplican señales lógicas dependiendo de lo que se requiera realizar. Para nuestros propósitos mantendremos las señales en P3.6 y P3.7 en 1 lógico, mientras que los valores de P2.6 y P2.7 tendrán la posibilidad de ser seleccionados. (HI/LOW)

Adicionalmente es necesario disponer de un cristal de 4-6Mhz.

Para el caso de los microcontroladores de la familia MCS48 los pines sobre los que es necesario actuar para el proceso de programación son los siguientes:

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
4	RESET	HI/LOW (Iniciación y retención de las direcciones)
1	T0	HI/LOW (Selección de modo de programación/verificación)
7	EA	18.5/23/5[V] (Activación de los modos de program.\verificac)
26	V_{DD}	21/25/5[V] (Fuente de alimentación para programación)
25	PROG	18.5/23 (Entrada de pulso de programación)
11	ALE	HI/LOW (Para verificación del socket)

Puesto que los microcontroladores 8748 y 8749 requieren de un cristal de 1-6 Mhz, y los microcontroladores 8748H y 8749H requieren de un cristal de 3-4 MHz, el cristal a ser usado debe ser de 3-4 Mhz.

4.- Dependiendo de la capacidad del elemento a ser programado debe tenerse en cuenta el número de líneas de dirección necesarias. Además para todos los elementos será necesario disponer de 8 líneas de datos.

En el caso de memorias el máximo direccionamiento es de 64k

por lo tanto se necesita 16 líneas destinadas para el bus de direcciones. Independientemente se dispondrá de 8 líneas de datos.

En el caso del MCS-51 el máximo direccionamiento es de 8K. Se necesita 13 líneas de direcciones; además 8 líneas adicionales para datos.

En el caso del MCS-48 el máximo direccionamiento es de 2K; por lo tanto, se necesita 11 líneas de direcciones; cabe anotar que los ocho bits menos significativos del bus de direcciones es compartido con el bus de datos; por esta razón, no será necesario 8 líneas adicionales para datos.

2.2 DIAGRAMA DE BLOQUES DEL PROGRAMADOR

La figura 2.1 presenta un esquema general del programador.

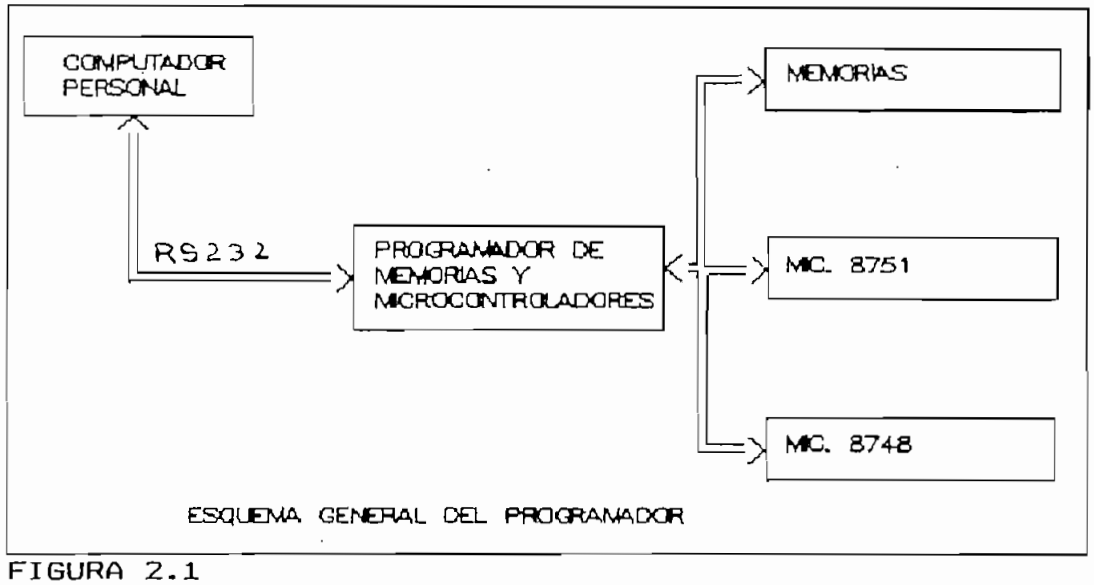


FIGURA 2.1

De acuerdo al análisis anterior el programador en sí debe constar de:

- Interfaz para comunicación con el computador personal.

- Reguladores de voltaje.
- Señales de control que serán proporcionadas por un microcontrolador. Estas señales incluyen direcciones y datos.

La figura 2.2 muestra en diagrama de bloques un esquema más detallado del programador con sus bloques constitutivos y que se tomarán en consideración para el diseño.

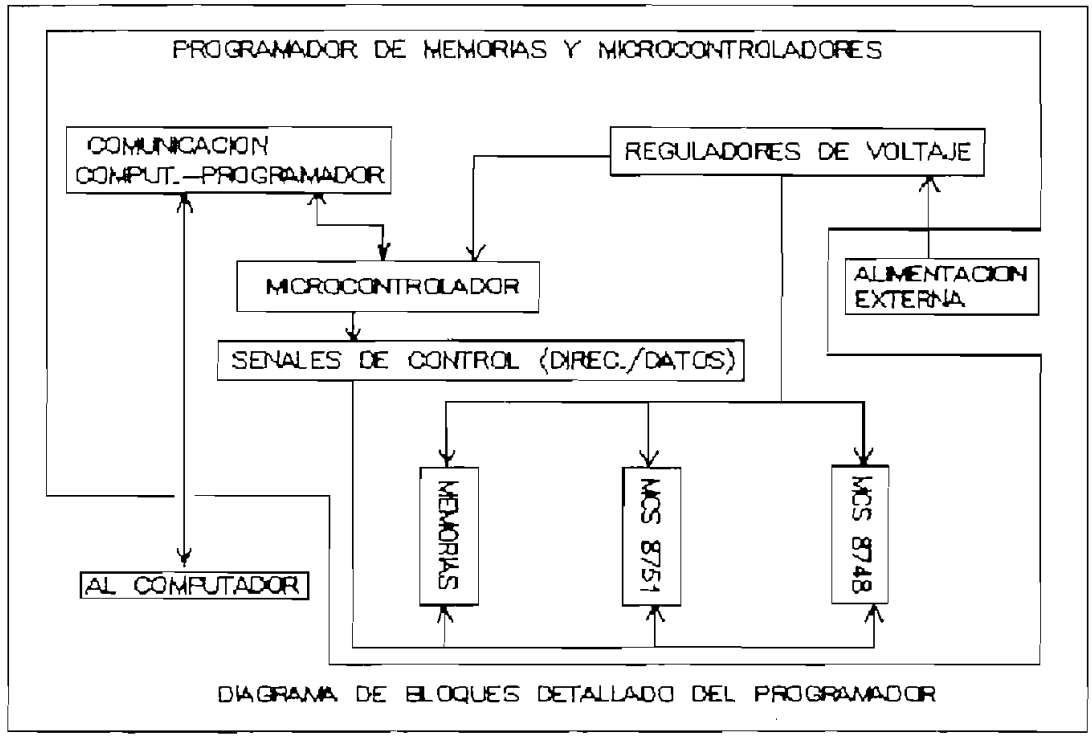


FIGURA 2.2

2.3 DISEÑO CIRCUITAL

Siendo el programador un equipo que debe realizar operaciones destinadas a leer o grabar información en memorias mediante direccionamiento, envío de datos y voltajes adecuados en los distintos pines de los elementos, se utilizará un microcontrolador que permita, mediante un

programa monitor, realizar en secuencia las distintas operaciones necesarias.

El microcontrolador, el mismo que permitirá hacer la secuencia correcta de los pasos requeridos tanto para programación como para lectura de los distintos chips que se ha propuesto, es el INTEL8031. Se ha seleccionado este microcontrolador por el conocimiento amplio que se tiene en su funcionamiento y aplicaciones, además por su relativo menor costo en comparación con el 8751, el mismo que incorpora memoria de programa interna.

El microcontrolador dispone de 4 puertos, los mismos que pueden ser utilizados de acuerdo a nuestras necesidades.

Al utilizar el microcontrolador 8031, es necesario disponer de memoria externa de programa; para el efecto se utilizará una memoria EPROM 2732 de 4Kx8.

Para el enlace de la memoria y el microcontrolador es necesario disponer de: un 74LS373, y resistencias de pull-up de 10k.

El microcontrolador opera con un cristal externo de 7.3728Mhz.

Al utilizar memoria externa de programa, los puertos 0 y 2 serán utilizados para direccionamiento de esta memoria y su respectivo envío de datos. Quedando inhabilitados estos puertos, para envío de las señales de programación. Se tiene disponible, por lo tanto, solamente dos puertos: uno y tres. Debe considerarse que del puerto tres las líneas: PSEN, ALE RD, WR, RX y TX estarán siendo ocupadas para

propósitos específicos: ya sea de comunicación, o para señales de lectura en caso de la memoria de programa, o para señales de lectura/escritura en el caso de que se trabaje, externamente, con un dispositivo de este tipo. Con estas consideraciones se dispone de un total de 12 líneas disponibles; 8 líneas del puerto 1 y 4 líneas adicionales (INT0, INT1, T1 y T0) del puerto 3.

Para el caso de memorias, de acuerdo a los requerimientos, se necesita: 16 líneas para el bus de direcciones; además de esto, se necesitan 8 líneas para el bus de datos; otras tres líneas, que permitan controlar los leds de señalización; tres señales adicionales, las mismas que permitirán programar los reguladores de voltaje; por último, 6 líneas para señales de control de la programación. Con esto el total de líneas requeridas es de 36. Dado que solamente tenemos 12 líneas disponibles nos hace falta 24 líneas más.

Para el caso del MCS51, de acuerdo a los requerimientos, se necesita: 13 líneas para el bus de direcciones; además, 8 líneas para el bus de datos; 3 líneas para los leds de señalización; tres líneas adicionales, las mismas que permitan programar los reguladores de voltaje; por último, se requiere de 4 líneas para señales de programación. Con esto se tiene un total de 31 líneas requeridas. Necesitándose 19 líneas adicionales.

Para el caso del MCS48, de acuerdo a los requerimientos, se necesita: 11 líneas, de las cuales 8 son compartidas entre

las líneas inferiores del bus de direcciones y el bus de datos: tres líneas para los leds de indicación; adicionalmente. tres líneas para programar los reguladores de voltaje; por último, 6 líneas para control de la programación. Existiendo un total de 31 líneas requeridas. Por lo tanto, se necesitan 19 líneas adicionales.

Del análisis anterior se requieren 24 líneas adicionales, en el caso de memorias. Al suplir este requerimiento los otros dos:

MCS51 y MCS48 ya no tendrán problema pues el número de líneas requeridas es menor. Siendo tres dispositivos que se programan uno a la vez es posible tener líneas compartidas las mismas que darán su estado de acuerdo a las necesidades de programación.

Una solución es el uso de un ampliador de puertos el mismo que permite tener 24 líneas, IN/OUT, adicionales. Con esto superamos el problema. Un ampliador de puertos, muy conocido, es el INTEL8255; al mismo que se lo maneja como que fuera memoria de datos externa. Este dispositivo requiere ser programado, ésta programación se indica en los anexos.

El uso dado a los diferentes pines es el siguiente:

- Para bus de datos se utiliza el puerto 1 del 8031 (8 líneas).
- Bus de direcciones: el puerto A del 8255, para direcciones A0 - A7 y el puerto B del 8255, para direcciones A8 - A15 (16 líneas).

- Para los leds indicadores tenemos tres líneas del puerto C del 8255. PC3 para MCS48, PC4 para MCS51, PC5 para memorias (3 líneas).
- Para los reguladores de voltaje las líneas a utilizarse son: PC0 para regulador de alimentación a memorias (5[V] ó 6[V]); PC1 y PC2 para regulador de voltajes de programación (12.5[V], 21[V], 25[V]); PC2 para regulador de voltajes de programación (18[V] y 23[V]). (3 líneas)
- Las líneas PC6, PC7, INT0, INT1, T0 y T1, (6 líneas), serán utilizadas para control de señales de programación así:

Para memorias:

INT0 va conectado al pin 20 (CE).

INT1 tiene influencia sobre el pin 26 permitiendo el paso del voltaje de alimentación, o la dirección AD13.

PC6 tiene influencia sobre el pin 1 permitiendo el paso del voltaje de programación, o la dirección AD15.

T1 tiene influencia sobre el pin 23 permitiendo el paso del voltaje de programación, o la dirección AD11.

T0 tiene influencia sobre el pin 22 permitiendo el paso del voltaje de programación, o señales HI/LOW, dadas por **PC7**.

Para MCS51:

Las conexiones son las siguientes:

T1 va conectado al pin 30 (ALE).

INT0 va conectado al pin 27 (P2.6).

INT1 va conectado al pin 28 (P2.7).

T0 tiene influencia sobre el pin 31 (EA/Vpp) permitiendo el paso del voltaje de programación, o de Vcc.

Para MCS48:

Las conexiones son las siguientes:

INT1 va conectado al pin 4 (Reset).

PC7 va conectado al pin 1 (T0).

INT0 va conectado al pin 11 (ALE).

PC6 tiene influencia sobre el pin 7 (EA) permitiendo el paso del voltaje necesario para programación o VCC.

T1 tiene influencia sobre el pin 25 (PROG) permitiendo el paso del voltaje necesario para programación o niveles de voltaje dados por **PC7**.

T0 tiene influencia sobre el pin 26 (VDD) permitiendo el paso del voltaje necesario para programación o VCC.

Para el microcontrolador así como para el 8255 se incorporó un circuito de reset basado en una red R-C. Para su diseño se tomó en cuenta el tiempo de carga del capacitor de esta red:

El tiempo de carga del capacitor es de:

$$t \approx RC$$

$$t \approx 8.2K \times 10\mu F \approx 82ms$$

La figura 2.3 presenta las conexiones del microcontrolador (8031), el ampliador de puertos (8255), la memoria externa de programa (EPROM 2732), el 74LS373, resistencias de PULL-UP y el circuito de reset.

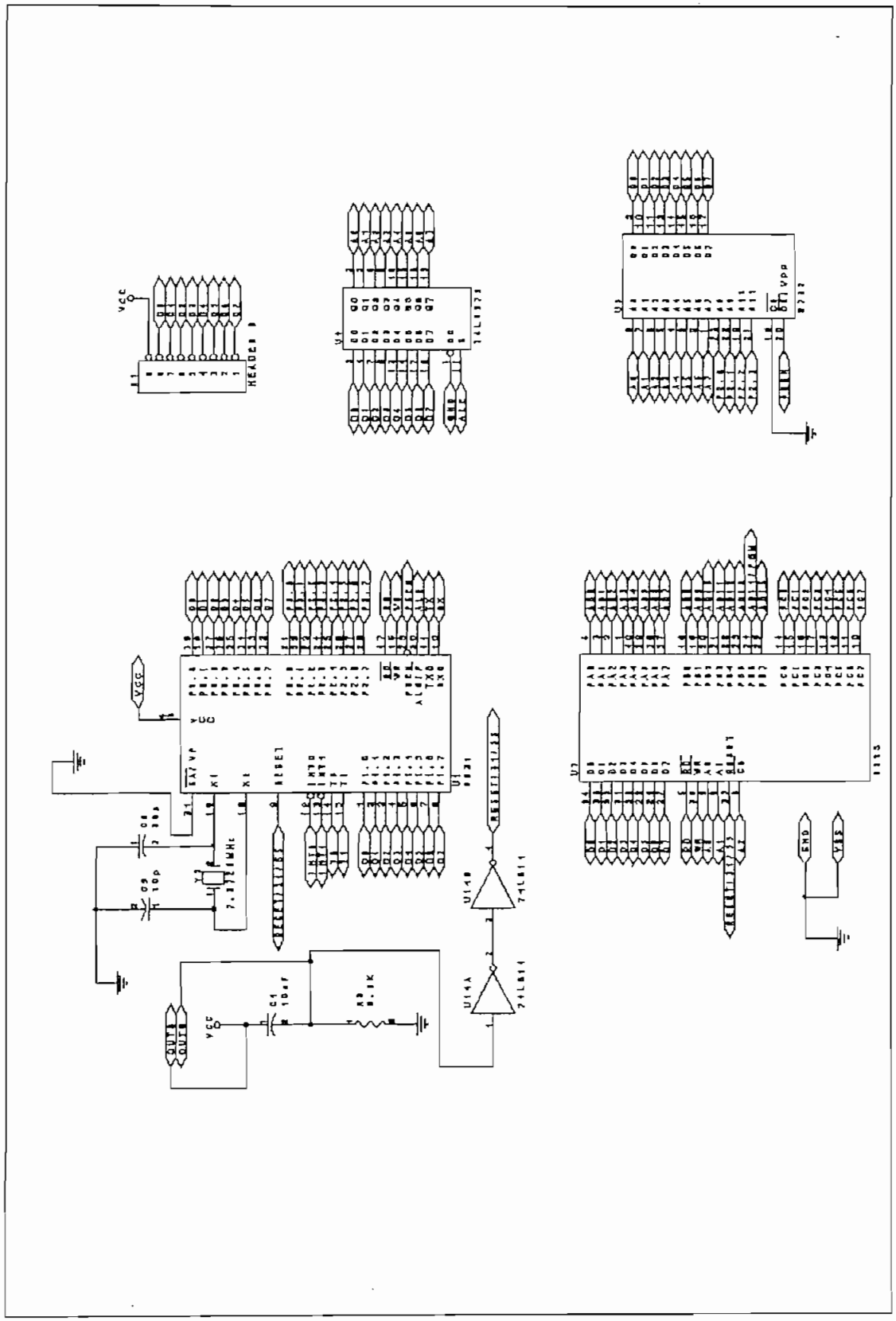


FIGURA 2.3

Con una idea más clara de como se distribuyen las señales se procederá al diseño de los módulos especificados anteriormente.

2.3.1 INTERFACE PARA COMUNICACION ENTRE EL COMPUTADOR PERSONAL Y EL PROGRAMADOR.

En primer lugar uno de los bloques de consideración para el diseño circuital es el que corresponde a la comunicación computador-programador.

Una solución económica constituye realizar un arreglo con transistores, diodos, resistencias y compuertas para el receptor serial del equipo, tal como se encuentra en la figura 2.4.

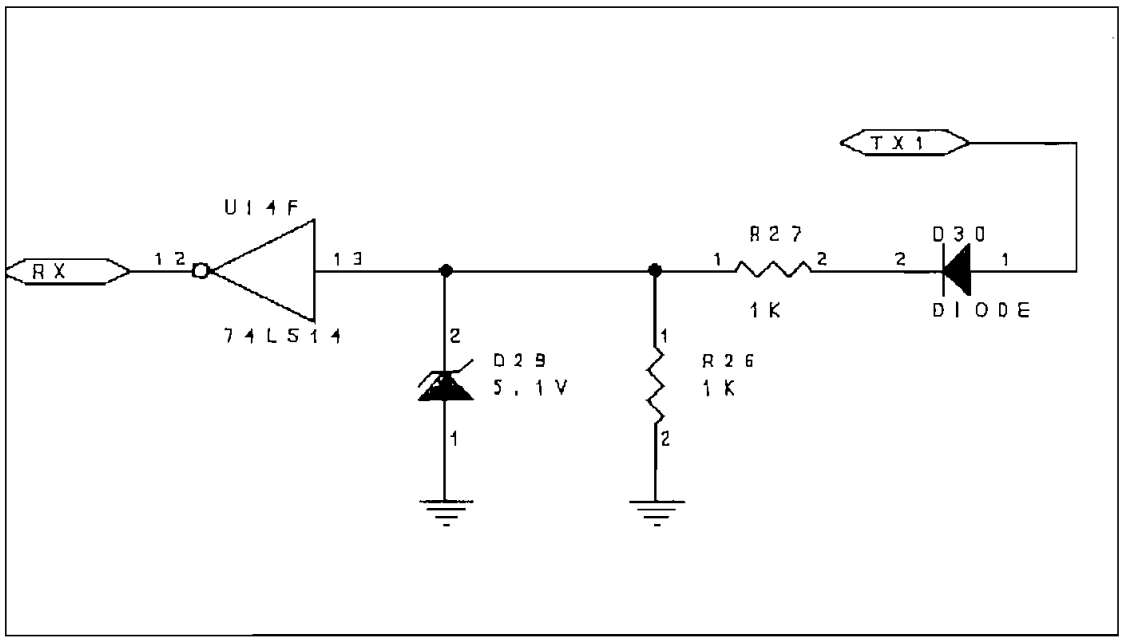


FIGURA 2.4

Puesto que las señales provenientes del computador son +12[V] y -12 [V] representando estas 0_L y 1_L respectivamente, estas deben ser acopladas a señales TTL de

0V y +5V.

Con el arreglo indicado cuando la señal proveniente del computador sea de +12[V] el diodo zener se encontrará regulando teniendo a la entrada de la compuerta 74LS14 +5[V], y a la salida 0[V], por lo tanto 0_L; en cambio cuando la señal proveniente del computador es de -12[V] el diodo zener no podrá regular por lo que a la entrada de la compuerta 74LS14 tenemos un 0_L y a su salida se tiene un 1_L.

Las resistencias R26 y R27 son de 1K, se utiliza un diodo zener de 5.1 V, un diodo ECG519 y una compuerta del chip 74LS14.

Para la señal de recepción en el computador se necesita realizar un acoplamiento de señales TTL +5[V] y 0[V] del circuito, a señales que se necesita en el computador de -12[V] y +12[V] respectivamente. Para poder satisfacer este requerimiento se desarrolló el circuito de la figura 2.5. El pin4 que se indica en la figura corresponde al pin4 del conector DB25 del puerto serial del computador (señal RTS), el mismo que tiene un voltaje de -12 V. El pin 20 también correspondiente al conector DB25 del puerto serial del computador presenta un voltaje de +12[V] y corresponde a la señal (DTR).

Cuando TX es 0_L la salida de la compuerta 7406 es de 1_L con lo cual el transistor Q8 estará en corte y así mismo el transistor Q9, la señal que llegue a RX1 en este caso es de +12[V]. En cambio, cuando TX es 1_L la salida del 7406 es

de O_L con lo cual Q8 y Q9 estarán saturados y RX1 tendrá un voltaje de -12[V].

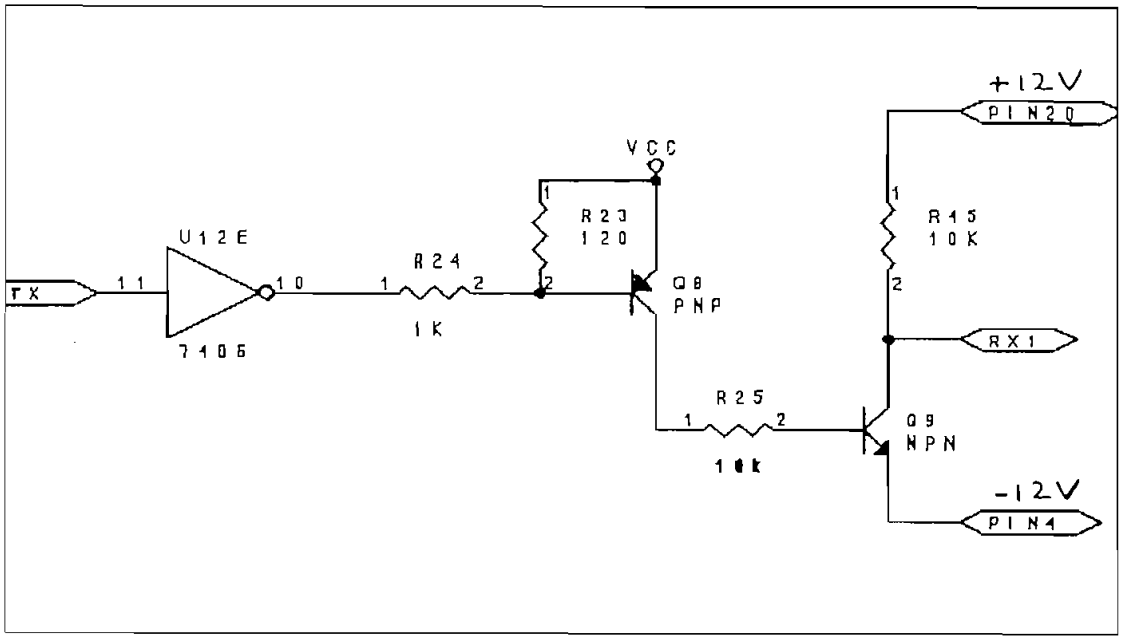


FIGURA 2.5

Para Q9 tenemos V[R251] = 5[V], y V[R252] = -12[V]; teniendo una diferencia de voltaje de 17V en R25. Con una resistencia de 10K aseguramos saturación de Q9 con 1.7mA de corriente de base.

2.3.2 REGULADORES DE VOLTAJE

Los reguladores de voltaje se requieren para dos propósitos:

a) Para la polarización de los chips

b) Para obtener los voltajes necesarios para la programación.

En el numeral 2.1 se pueden apreciar los requerimientos de señales de voltaje, los mismos que dependen de la memoria que se desea programar. Puesto que existen distintos

voltajes para programación dependiendo de la memoria que se disponga, se ha optado por realizar el diseño mediante reguladores programables de voltaje, la programación de estos se la realiza mediante señales enviadas del PC al microcontrolador, el mismo que será el encargado de ejecutarlas y seleccionar los voltajes necesarios.

Las memorias requieren de los tres voltajes indicados anteriormente.

Como la programación de los microcontroladores 51 y 48 se realiza independientemente, entonces los dos pueden compartir el mismo regulador de 12.5, 21 y 25 voltios.

En cuanto a la corriente de programación, la máxima corriente de programación en las memorias es de 50mA(3), los microcontroladores de la familia MCS48 requieren una corriente de 30mA (la máxima) (4,5), y los microcontroladores MCS51 requieren una corriente de 75mA (4) (como máximo). A parte de la corriente de programación el regulador debe ser capaz de alimentar a 3 resistencias de 1kΩ siendo el caso crítico de requerimiento de corriente cuando las tres son alimentadas por el voltaje de 25[V] teniendo por lo tanto un consumo adicional de 75mA.

Con esto el regulador debe ser capaz de entregar 150mA.

Por lo tanto se necesita un regulador programable: 12.5[V], 21 [V] y 25[V], el mismo que sea capaz de entregar una corriente máxima de 150mA. Para esto se utilizará un regulador LM317 (ECG 956).

En la figura 2.6 puede apreciarse el arreglo circuital

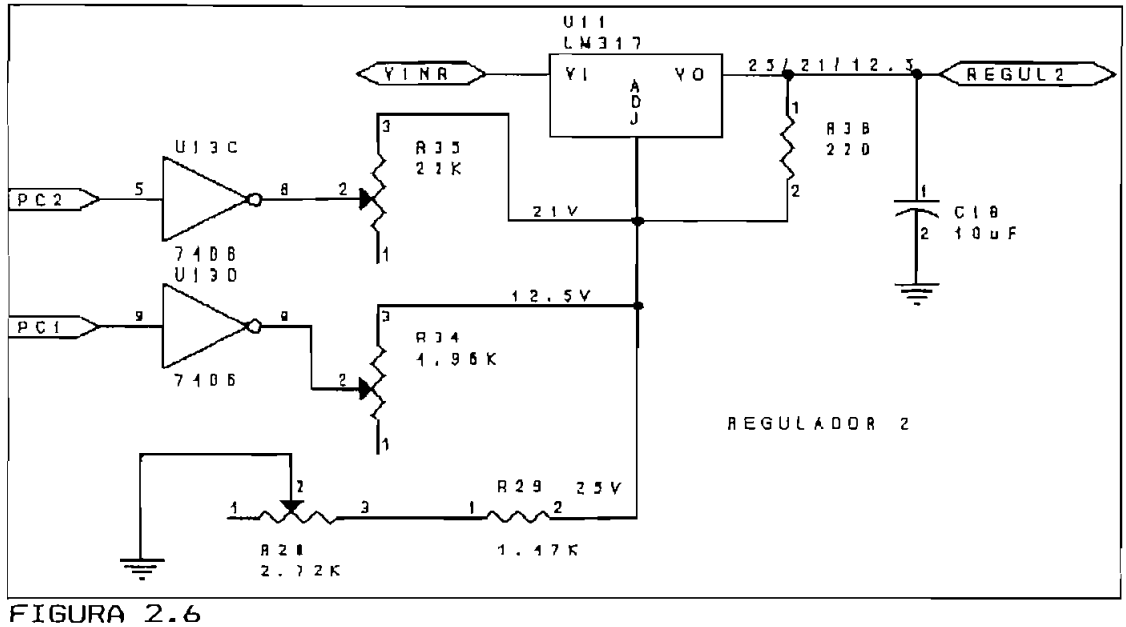


FIGURA 2.6

Se puede observar que las compuertas utilizadas son buffers open collector; cuando se pone un alto en la entrada de las mismas se obtiene un bajo en la salida, y en cambio cuando la entrada está en bajo la salida se acoplará al voltaje que se tenga en ADJ del regulador. Puesto que el mayor voltaje que se requiere obtener es de 25 voltios se ha colocado directamente el potenciómetro R28 más la resistencia R29 a tierra, para éste caso las entradas a las dos compuertas deben ser 0_L.

Cuando se requiera obtener 21 voltios la entrada 13 del 7406 debe estar en alto para así lograr el paralelo de R35 con (R28 + R29), en este caso la entrada 5 de la compuerta del 7406 debe encontrarse en bajo.

En el último caso cuando se requiera obtener 12.5 voltios las entradas a las dos compuertas deben estar en alto con lo que $R_{equivalente}$ = al paralelo de las tres resistencias: R34, R35 y (R28 + R29).

Lo explicado puede verse en la tabla de estados siguiente:

ENTRADAS		REGULADOR 2 [VOLTIOS]
5(PC1)	13(PC2)	
0	0	25 [V]
0	1	21 [V]
1	0	CONDICION NO REQUERIDA
1	1	12.5 [V]

En este tipo de reguladores, el voltaje de salida está relacionado con las resistencias con la siguiente fórmula:

$V_{out} = 1.25V * (1 + R2/R1)$

Si R2 = 240Ω

Para $V_{out} = 25V$

R1 = (25/1.25 - 1)*240

R1 = 4560Ω

R1 = R28 + R29

Siendo R29 = 1.47K, R28 = 3090K

Para $V_{out} = 21V$

R1' = R1 || R2

R1' = (21/1.25 - 1)*240

R1' = 3792Ω

Por lo tanto R2 = 22,471K. Se utilizará un potenciómetro de precisión R35 = 25K

Para $V_{out} = 12.5V$

R1" = R1 || R2 || R3

R1" = (12.5/1.25 - 1)*240

R1" = 2160Ω

Por lo tanto R3 = 5022Ω. Se utilizará un potenciómetro de

precisión R34 = 10K

Los MCS48 requieren de dos fuentes reguladas, una que debe entregar 18 voltios y 23 voltios, y la otra que debe proporcionar 21 y 25 voltios; asociada a la de 18 voltios tenemos la de 21 voltios, el momento que nos encontremos programando; y a la de 23 voltios se encuentra asociada la de 25 voltios. Uno de los reguladores, el de 21/25, puede ser el que estamos tomando en cuenta para memorias y que se encuentra diseñado anteriormente. Mientras que será necesario incorporar un regulador adicional para los restantes 2 voltajes que hacen falta. (1 regulador programable: 18[V] y 23[V]).

Este regulador será únicamente para las señales de EA48 y PROG48 siendo el consumo de corriente crítico de 17mA en el caso de 8748 y 8749 y de 2mA para el caso de 8748H y 8749H (4,5); adicionalmente este regulador alimentará a una resistencia de 1KΩ siendo el caso crítico de consumo de corriente cuando el regulador trabaje a 23 voltios teniendo un consumo de 23mA. En total el consumo máximo de corriente es de 40mA; por lo tanto, se utilizará un regulador ECG1900.

El modo de funcionamiento es muy similar al anterior con la diferencia de que aquí solamente se seleccionan dos voltajes.

Se utilizará la misma entrada PC2 para dar compatibilidad a los pares de voltajes necesitados para este tipo de microcontroladores.

La figura 2.7 presenta el diagrama circuital para este regulador.

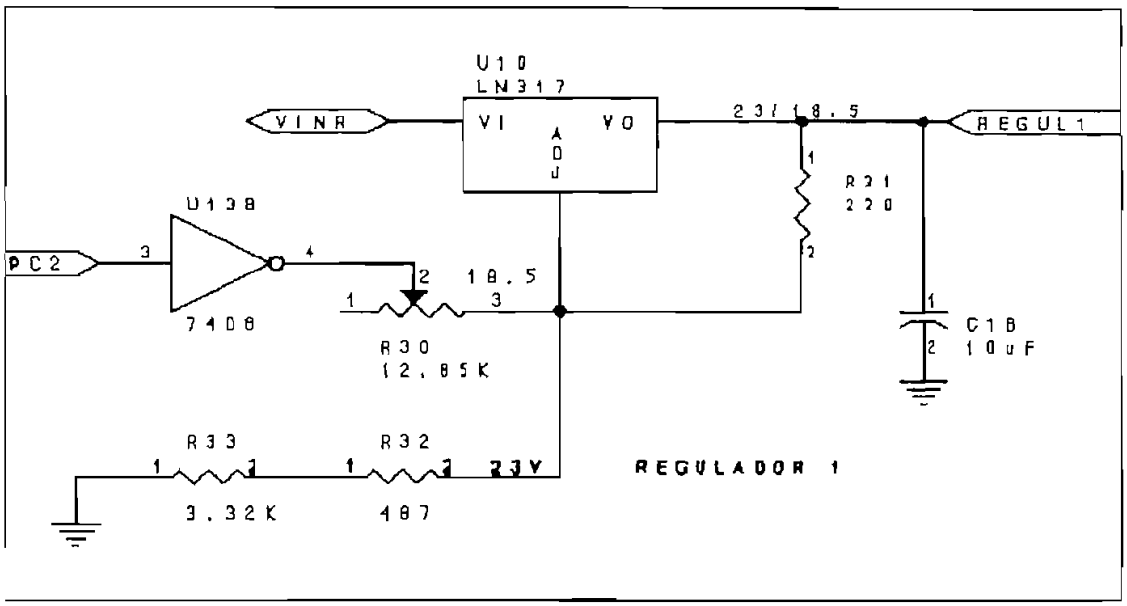


FIGURA 2.7

La tabla de estados siguiente me indica el estado del regulador de acuerdo a la entrada en la compuerta:

ENTRADAS	REGULADOR 1 [VOLTIOS]
3(PC2)	
0	23 [V]
1	18 [V]

El cálculo de las resistencias, para este regulador, es el siguiente:

V_{out} = 1.25V * (1 + R2/R1)

Si R2 = 240Ω

Para V_{out} = 23V

R1 = (23/1.25 - 1)*240

R1 = 4176Ω

El voltaje de 23[V] se logró con el uso de dos resistencias de precisión:

R33 = 3.32K y R32 = 487Ω

Para V_{out} = 18V

R1' = R1 || R2

R1' = (18/1.25 - 1)*240

R1' = 3216Ω

Por lo tanto R2 = 13,98 K

Se utilizó una potenciómetro de precisión R30 = 25K.

Adicionalmente se requiere un regulador para la alimentación de memorias que sea capaz de proporcionar +5[V] ó 6[V]. (1 regulador programable: 5[V], 6[V]). Este regulador entregará corriente para alimentación de las memorias; en este caso el máximo consumo de corriente durante la programación lo tiene la memoria 27512 con 150mA (∞) . Adicionalmente este regulador debe ser capaz de alimentar a una resistencia de 1KΩ siendo el máximo consumo de la misma de 6mA; por lo tanto, el consumo total es de 156mA. Se utilizará un regulador LM317 (ECG956). El modo de funcionamiento es similar a los dos anteriores; la figura 2.8 presenta el diagrama circuital.

El cálculo de las resistencias utilizadas es:

V_{out} = 1.25V * (1 + R2/R1)

Si R2 = 240Ω

Para V_{out} = 6.25V

R1 = (6.25/1.25 - 1)*240

R1 = 960Ω

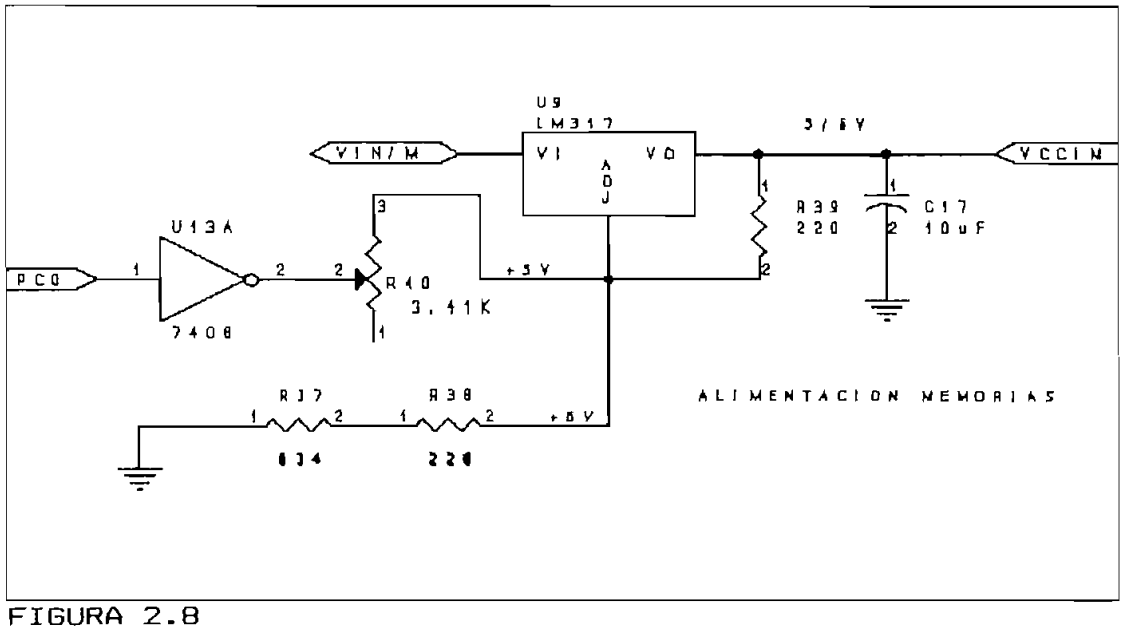


FIGURA 2.8

El voltaje requerido se logró con el uso de dos resistencias de

precisión: R37 = 634Ω y R38 = 220Ω

Para V_{out} = 6V

R1' = R1 || R2

R1' = (5/1.25 − 1)*240

R1' = 720Ω

Por lo tanto R2 = 2,88K

Se utilizó un potenciómetro de precisión R40 = 5K

Adicionalmente se utilizan capacitores de 10µF a la salida de cada regulador.

La fuente de alimentación para todos los chips es de 5[V]; la corriente de consumo se calcula a continuación (x,x,x):

Elemento	Corriente (mA)
CUATRO LEDS	89
8031	100
8255	120
DOS 7406	70
74LS373	40
74LS14	40
2732	100
Reset	0.6
Elemento a programar	80
Total	639.6mA

Para este propósito se utilizará un LM7805 que permite regular a 5[V]. El circuito de esta fuente se presenta en la figura 2.9. A la entrada se utiliza un capacitor de 4700µF/40V, 3 capacitores a la salida de 47µF, 10µF de tantalio y .1µF.

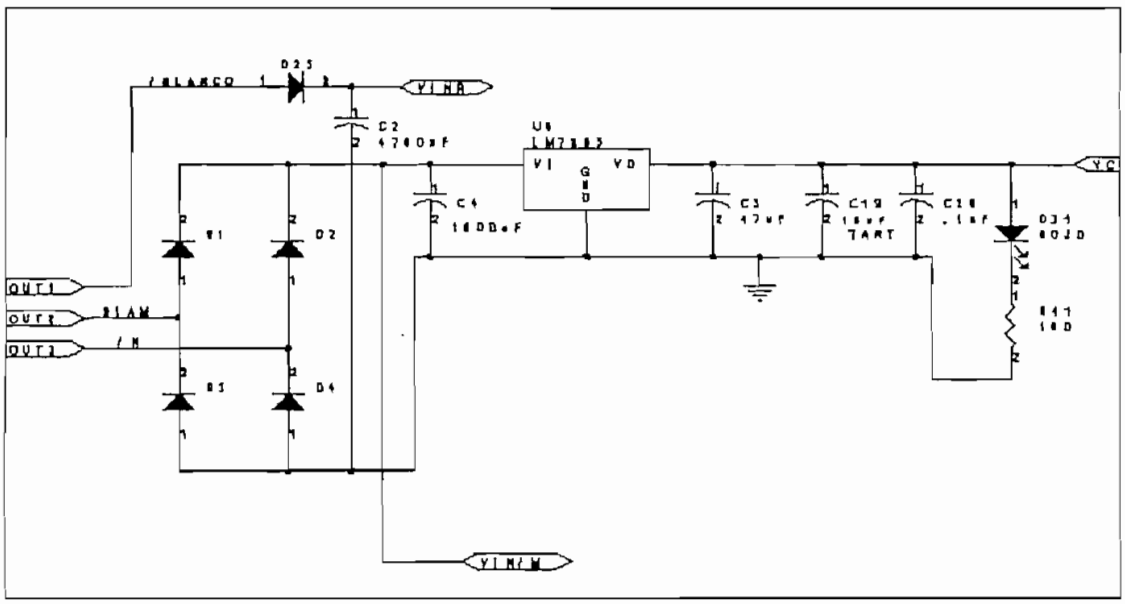


FIGURA 2.9

Para la señal de entrada se utiliza un transformador con tap central de 110[V] a 9/24[V] y 2 amperios. Para las dos fuentes de 5 voltios y la otra de 5/6 voltios usamos un rectificador tipo puente y el tap de 9 voltios; mientras que para las otras dos se utiliza un rectificador de media onda y el tap de 24 voltios. Para fijar la señal de entrada a los reguladores 1 y 2 se utilizó un capacitor de 4700μF/50V.

2.3.3 SEÑALES DE CONTROL, DIRECCIONAMIENTO Y DATOS MEMORIAS

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
20	CS	LOW/PULSO (HI/LOW)
22	OE/V_{pp}	$LOW/HI/V_{pp}$
23	$A11/V_{pp}$	$DIRECCION\ A11/HI/V_{pp}$
1	V_{pp}	HI/V_{pp}/DIRECCION A15
27	PGM	DIRECCION A14/HI/LOW
26	A13/VCC	DIRECCION A13/VCC

Para el caso de memorias se requiere señales de control en los pines indicados anteriormente; por lo tanto se irá especificando uno a uno que tipo de arreglo se requiere.

Para el pin 20 que corresponde al CS las únicas señales que se requieren son niveles lógicos, por lo tanto utilizaremos uno de los pines del microcontrolador (INT0).

Para el pin 22 que corresponde al OE/V_{pp} además de señales lógicas se requiere tener acceso al regulador 2. La figura 2.10 presenta el esquema circuital. PC7 se encargará de las señales lógicas en el caso de que T0 sea 0_L. En cambio

cuando T0 sea 1_L en el pin 22 tendremos el valor de voltaje que esté proporcionando el regulador 2.

Para los arreglos de este tipo se utilizará una resistencia de 1K entre la base del transistor y la salida de la compuerta correspondiente, y una resistencia de 120Ω entre la base y el emisor del transistor. El arreglo de diodos y capacitor que se aprecia permite eliminar picos de voltaje. La resistencia a la salida permite la presencia de 0_L a la entrada de pin22. Esta resistencia, R19, es de 1k. Los valores tomados permiten asegurar saturación del transistor.

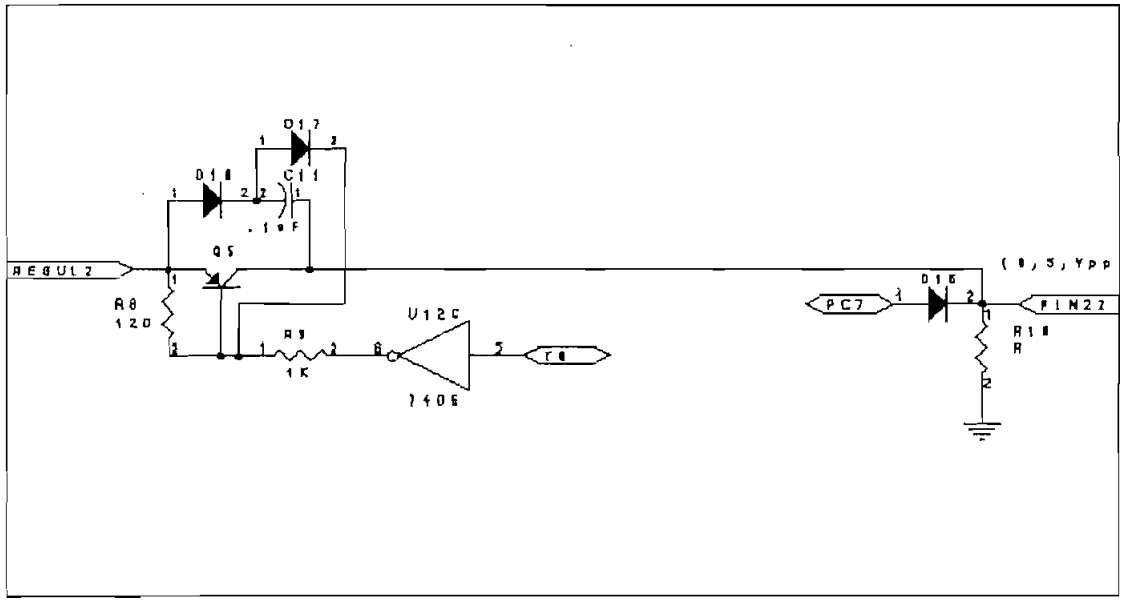


FIGURA 2.10

El pin 23 requiere de señales lógicas así como de V_{pp}. Se utilizará un arreglo circuital muy parecido al utilizado para el pin 22, en este caso T1 servirá de switch para el transistor Q7. AD11 dará los niveles lógicos de 0_L ó 1_L. Los otros elementos cumplen la misma función que en el caso

analizado para el pin 22.

La figura 2.11 presenta el arreglo circuital para el pin 23.

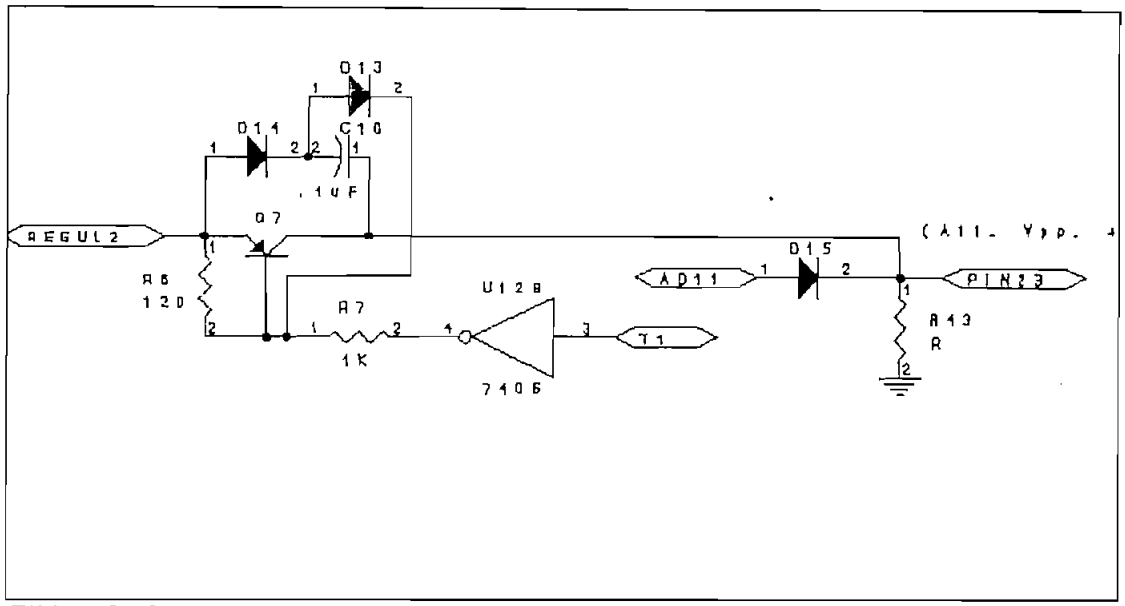


FIGURA 2.11

El pin 1 requiere de señales lógicas en alto y bajo así como de V_{PP}; siendo un caso muy similar a los anteriores solamente presentaremos el arreglo circuital en la figura 2.12.

El pin 27 solamente requiere señales lógicas por lo tanto se conectará directamente la salida AD14/PGM del ampliador de puertos a este pin.

El pin 26 de las memorias permitirá alimentación a memorias ó el direccionamiento de AD13, el arreglo circuital es similar a los tres presentados anteriormente con la diferencia de la resistencia de base-salida de compuerta que en este caso es de 330Ω. Con este valor de resistencia la corriente de base en saturación es de 11mA. El arreglo

circuital se muestra en la figura 2.13

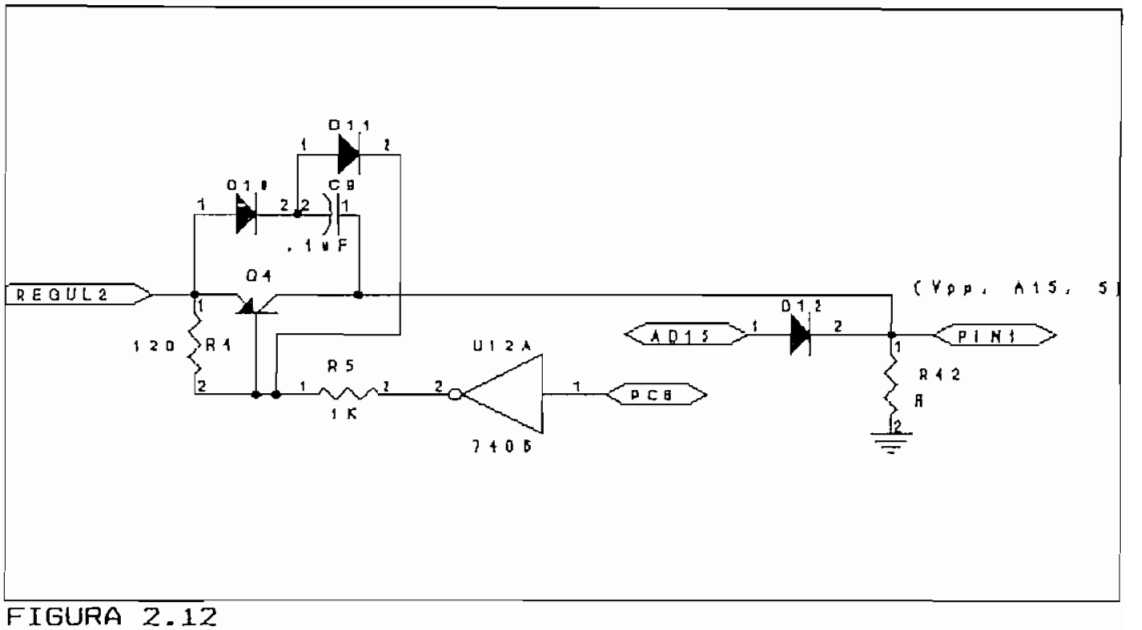


FIGURA 2.12

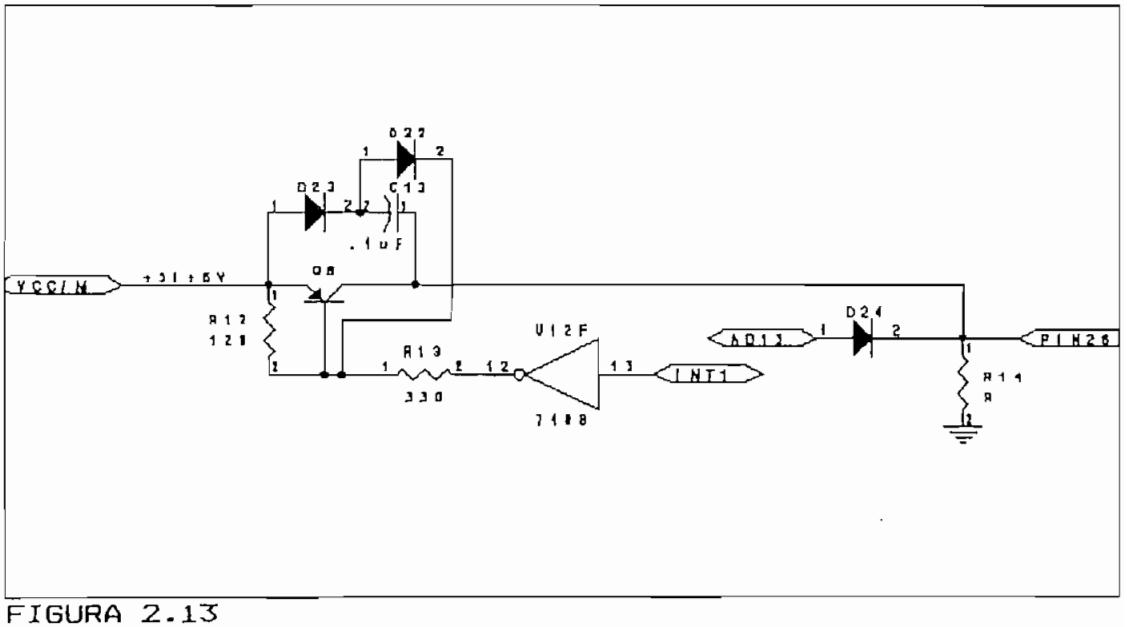


FIGURA 2.13

El direccionamiento se lo realiza mediante los puerto A y B del ampliador de puertos; la salida de datos va

conectada al puerto 1 del 8031. La figura 2.14 presenta las conexiones hacia la memoria.

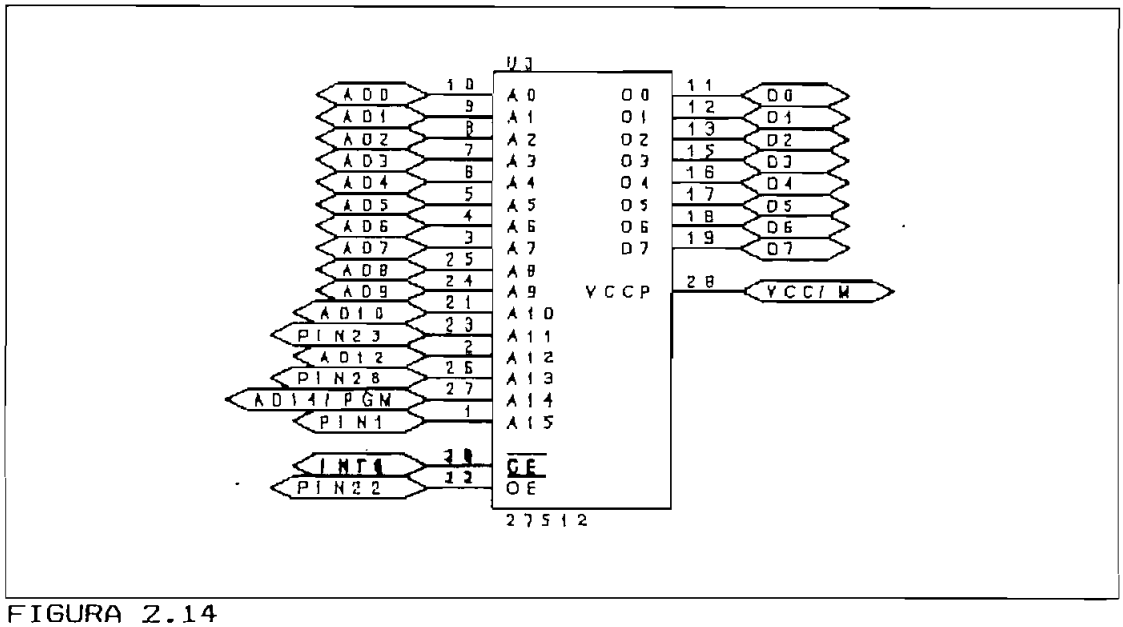


FIGURA 2.14

MCS51

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
4	RST	Permanece en Vcc
29	PSEN	Permanece en GND
30	ALE	HI/LOW
31	EA	V_{pp}
27-28	P2.6 P2.7	
17-18	P3.6 P3.7	P3.6 y P3.7 en 1 lógico, P2.6 y P2.7 HI/LOW

El pin 4 se fija a VCC, el pin 29 se fijará a GND y los pines 17 y 18 se conectarán a VCC.

El pin 30 al tener la posibilidad de variar entre niveles lógicos de 0_L y 1_L, será conectado directamente a T1 del 8031.

El pin 31 al ser el encargado de proporcionar el pulso de

programación tendrá un arreglo y funcionamiento similar al indicado para las memorias. El diagrama circuital se presenta en la figura 2.15

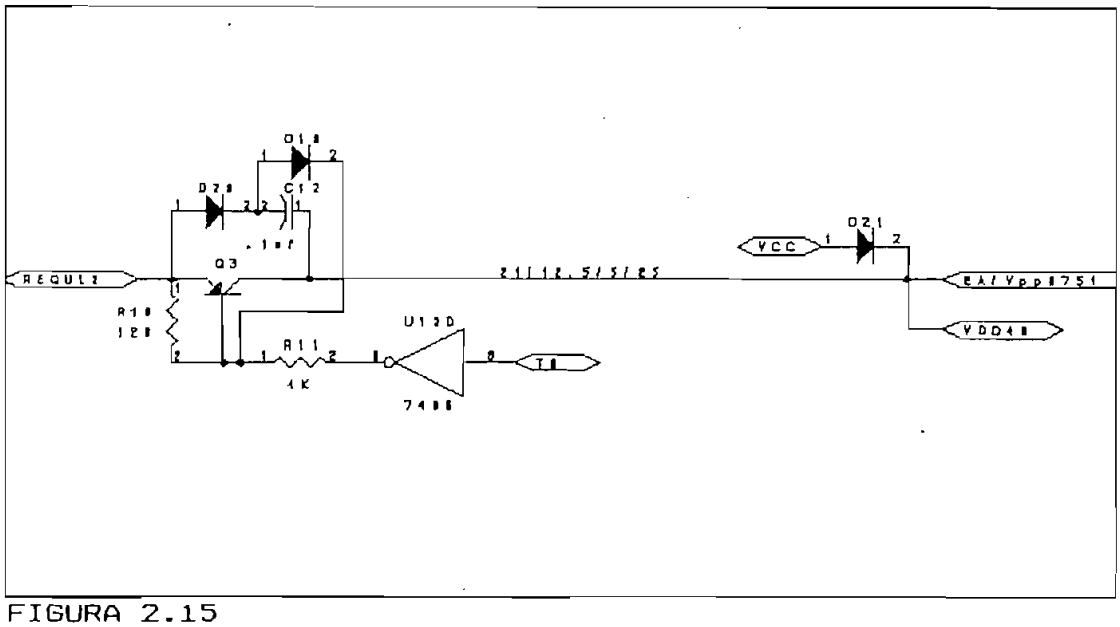


FIGURA 2.15

Por último los pines 27 y 28 al tener que variar entre niveles lógicos de 0_L y 1_L serán conectados, respectivamente, a las salidas INT0 e INT1 del microcontrolador 8031.

La figura 2.16 presenta las conexiones hacia el microcontrolador 8751. Se disponen además de resistencias de pull-up en el puerto 0. El direccionamiento de localidades de memoria se lo realiza gracias a la ayuda del ampliador de puertos, y el bus de datos viene del puerto 1 del 8031.

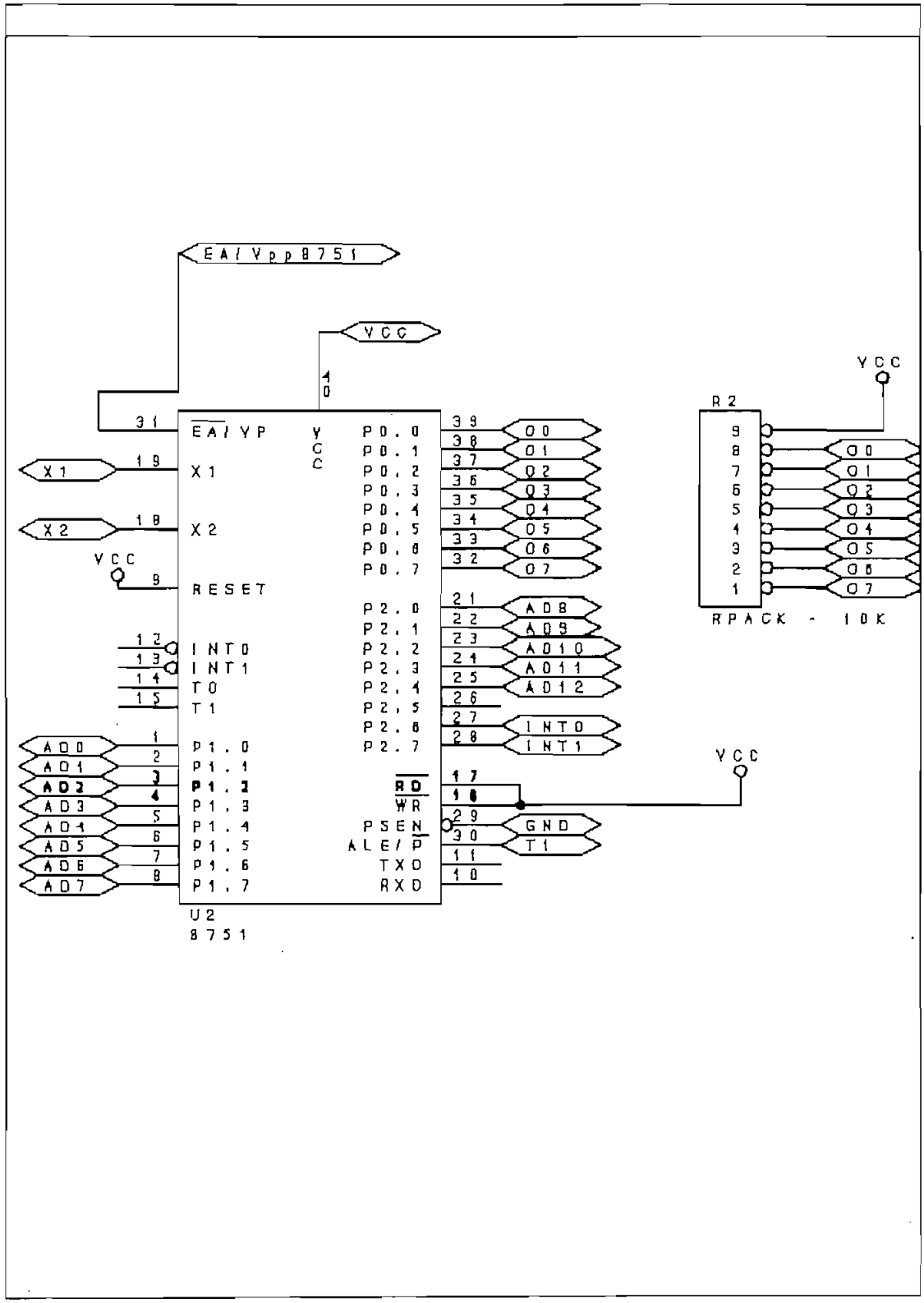

EA/Vpp 8751
VCC
31
EA/VP
VCC
40
X1
19
X1
X2
18
X2
VCC
9
RESET
12
13
14
15
INT0
INT1
T0
T1
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7
1
2
3
4
5
6
7
8
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
U2
8751
P0.0
P0.1
P0.2
P0.3
P0.4
P0.5
P0.6
P0.7
39
38
37
36
35
34
33
32
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7
P2.0
P2.1
P2.2
P2.3
P2.4
P2.5
P2.6
P2.7
21
22
23
24
25
26
27
28
AD8
AD9
AD10
AD11
AD12
INT0
INT1
RD
WR
PSEN
ALE/P
TXD
RXD
17
16
29
30
11
10
GND
T1
VCC
R2
9
8
7
6
5
4
3
2
1
RPACK - 10K

FIGURA 2.16

Se dispone de un cristal, compartido con el 8748, cuyo valor es de 4Mhz.

MCS48

# PIN	NOMBRE	REQUERIMIENTOS (VOLTAJE)
4	RESET	HI/LOW
1	TO	HI/LOW
7	EA	18.5/23/5[V]
26	V_{DD}	21/25/5[V]
25	PROG	18.5/23

El pin 4 necesita solamente señales lógicas; por lo tanto, se lo conecta a INT1.

El pin 1 igualmente necesita solo señales lógicas y se lo conecta a PC7.

Los pines 7 y 25 requieren de 18.5/23/5 voltios; los diagramas circuitales para la alimentación a estos dos pines se presenta en las figuras 2.17A y 2.17B.

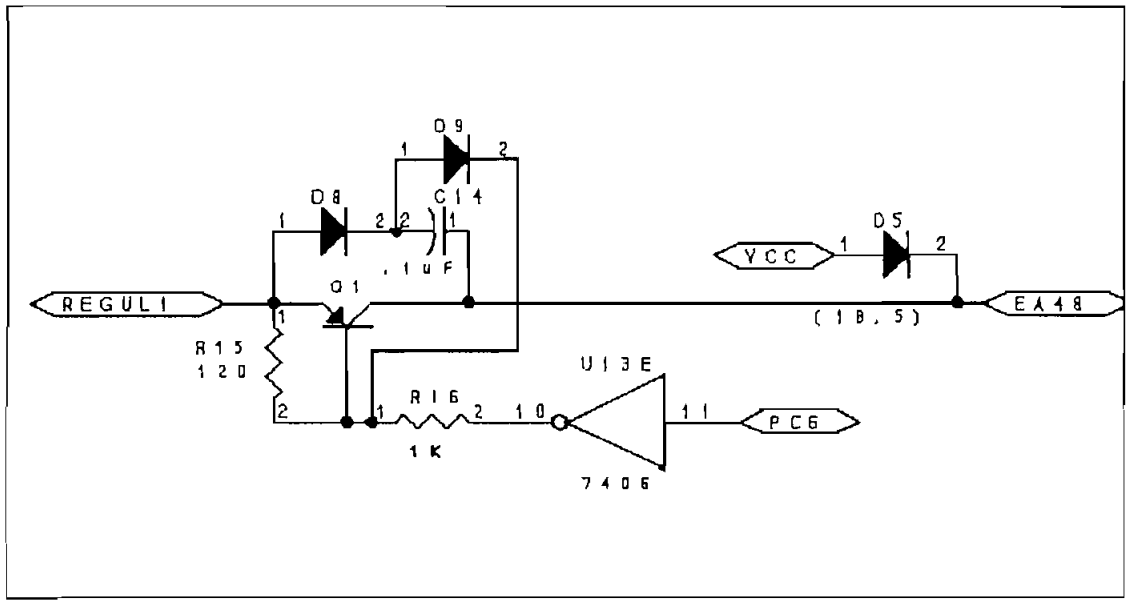


FIGURA 2.17A

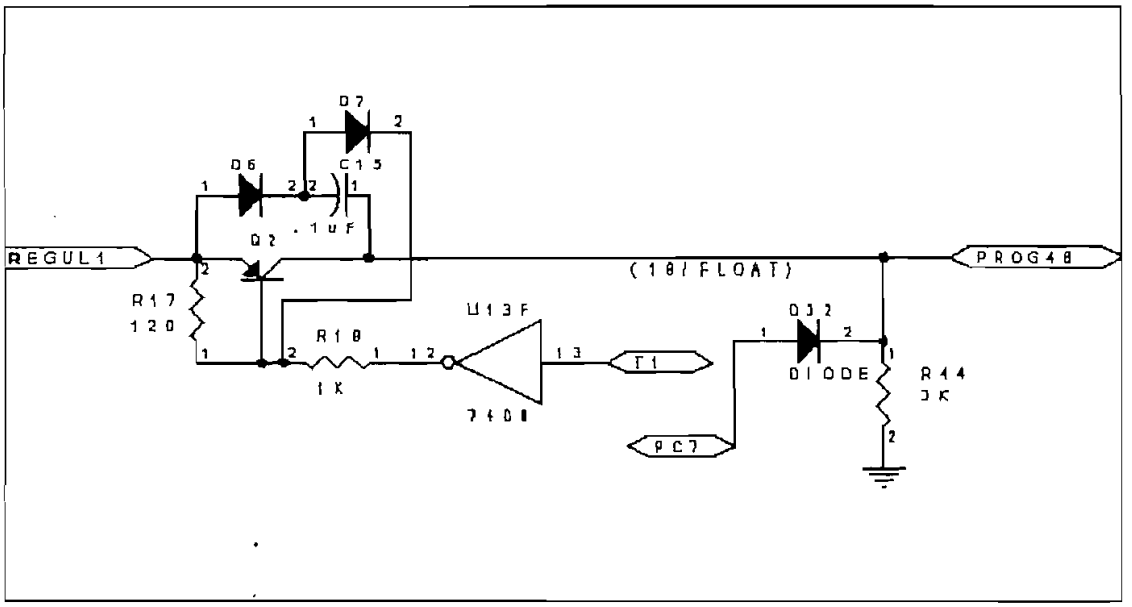


FIGURA 2.17B

El pin 26 requiere de 21/25 voltios teniendo un diagrama circuital como el mostrado en la figura 2.18. Su diseño es igual al realizado para las memorias.

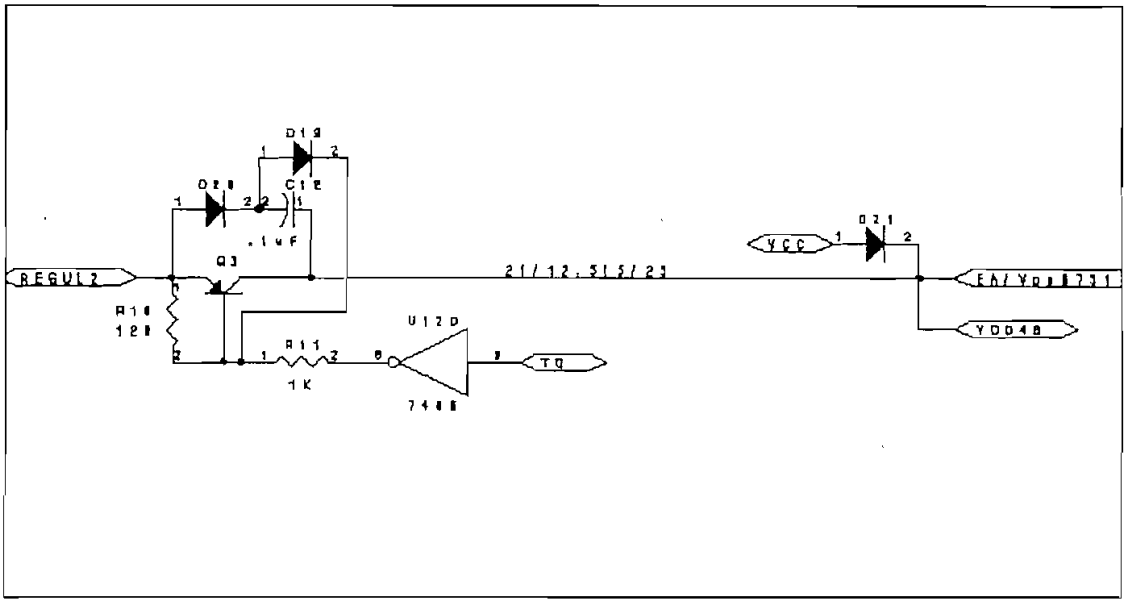


FIGURA 2.18

La figura 2.19 presenta las conexiones hacia el microcontrolador 8748. En este caso el puerto 0 será encargado de recibir tanto bus de direcciones menos significativas como el dato a ser grabado ó leído.

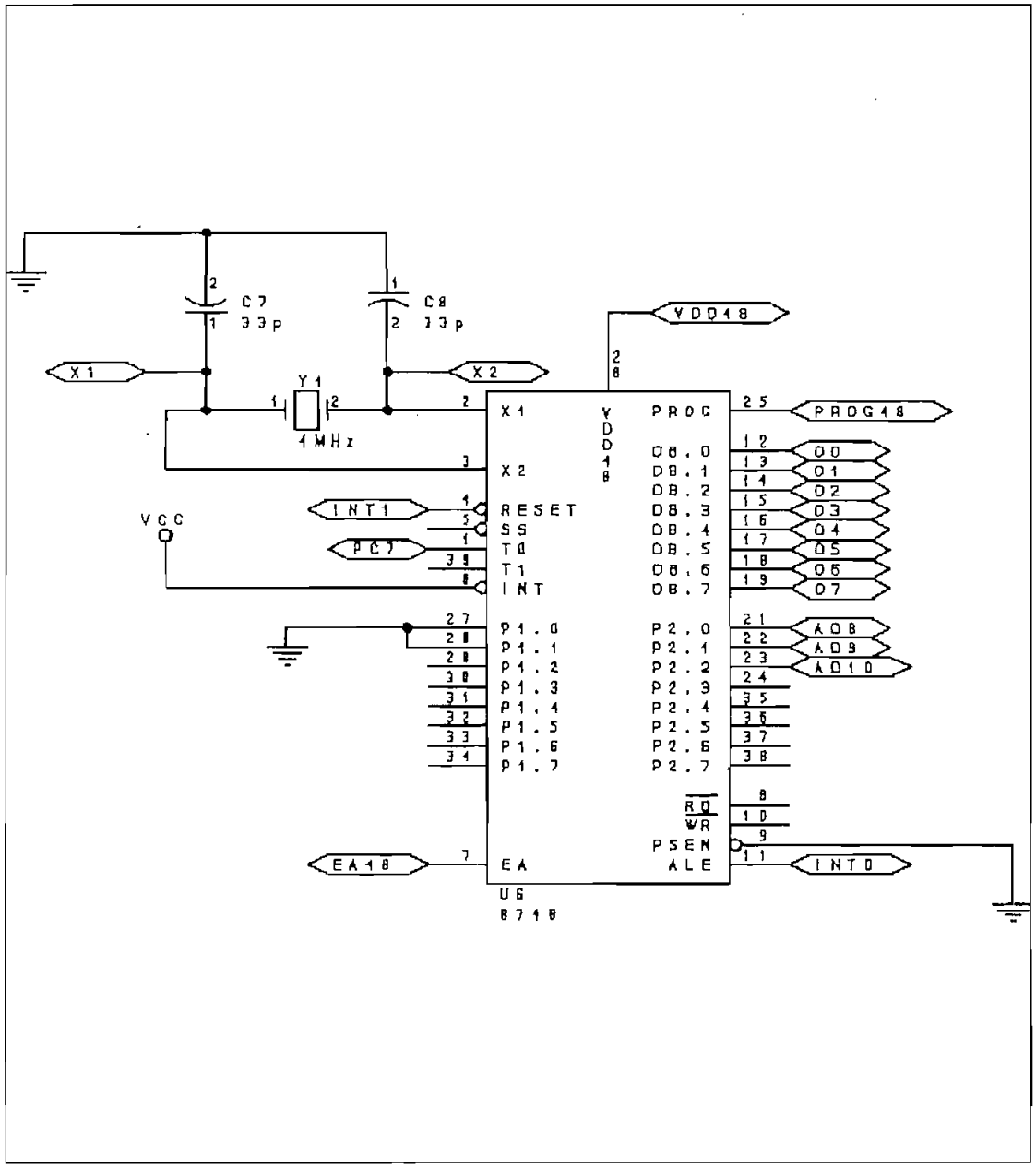


FIGURA 2.19

2.3.4 LEDS INDICADORES

Se utilizan cuatro leds indicadores: el primero para la indicación de alimentación a todos los elementos y los otros tres que especificarán, mediante encendido, cual de los elementos (memorias, MCS51 ó MCS48) es el que se encuentra habilitado para alguna de las funciones (Programación o Lectura).

Para tener control sobre los tres leds indicadores se ha utilizado buffers inversores (open collector) tal como muestra la figura 2.20

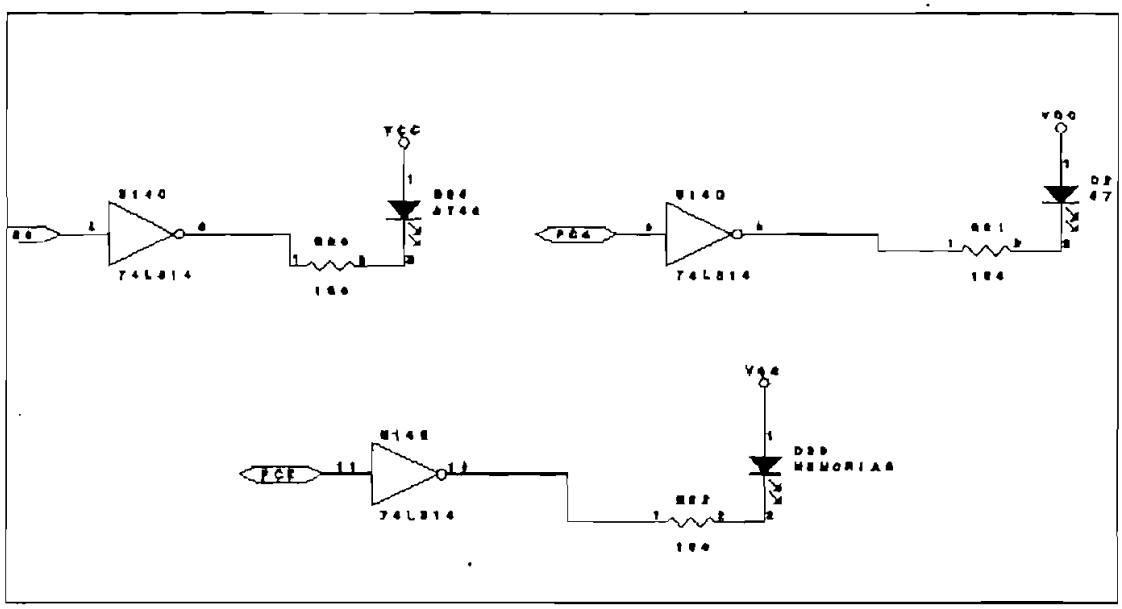


FIGURA 2.20

Adicionalmente se ubicarán capacitores entre Vcc y GND por cada integrado que se esté utilizando con el fin de desacoplar cualquier señal de ruido de la fuente.

NOTA: Los transistores utilizados son 2N2907 y los diodos son ECG519.

Un esquema total del diseño circuital se lo presenta en la figura 2.21

CAPITULO III

DESARROLLO DEL SOFTWARE

3.1 DESCRIPCION GENERAL

Es necesario desarrollar software tanto para el computador como para el microcontrolador.

El software para el computador es el medio de comunicación entre el usuario y la máquina. Este se encuentra desarrollado en QBASIC y presenta pantallas de selección de fácil uso.

El programa en QBASIC permite seleccionar las distintas funciones para las que fue diseñado el equipo, las mismas que a continuación se detallan.

3.1.1 FUNCIONES DEL PROGRAMADOR

- ESPECIFICAR EL TIPO DE MEMORIA

Permite al usuario especificar el tipo de memoria que se va a programar. Primeramente existe una selección del elemento a programar, el mismo que puede ser: un elemento de la familia MCS51, un elemento de la familia MCS48, ó una memoria EPROM comercial de las que se especifica en el capítulo I. Una vez seleccionado uno de estos tres grupos es posible escoger un elemento en particular para ser programado, leído, etc. Por ejemplo: de la familia MCS51 se puede seleccionar el elemento 87C51, o el 8751, etc.

- LECTURA DE UNA EPROM

Este comando permite al usuario leer la memoria y su contenido es almacenado en el buffer de memoria del PC. Esta operación puede ser interrumpida presionando la tecla

ESC.

- MOSTRAR CONTENIDO

Gracias a esta opción se puede ver el contenido del buffer de memoria.

El usuario podrá desplazarse a través de paginamientos de 256 bytes, el número de páginas depende de la cantidad de localidades de memoria. Los datos son mostrados en el monitor en hexadecimal y también como caracter ASCII. A cada 256 bytes se les denomina por facilidad página. Adicionalmente se tiene opción para desplazamiento sobre cada página con posibilidad de PGUP para avanzar y PGDN para regresar. Al presionar cualquier otra tecla esta operación es interrumpida.

- GRABAR EN UN ARCHIVO

Este comando permite al usuario escribir los datos desde el buffer de memoria a un archivo en cualquier unidad de disco.

Esta opción pedirá el nombre del archivo en el que se desea almacenar el contenido del buffer. El formato en el que se graban los datos es INTEL.

- ARCHIVO.HEX

Esta opción permite cargar un archivo.HEX en el buffer de memoria. Es necesario especificar el "path" de ubicación así como el nombre en forma correcta. El formato del archivo se indica en el capítulo I.

- PROGRAMAR UNA MEMORIA

Después de cargar el archivo.HEX se puede realizar la

programación de una memoria en blanco. El archivo.HEX es abierto como archivo de lectura, el mismo que es decodificado adecuadamente para enviar comandos y datos de acuerdo al formato INTEL.

Cada dato programado debe ser verificado; de esta manera aseguramos que la operación se encuentra realizando adecuadamente. En caso de que la memoria a ser programada no se encuentre en blanco la operación de programación es interrumpida.

- ESPECIFICAR VOLTAJE DE PROGRAMACION

Esta opción permite indicar el voltaje a ser usado para la programación, es necesario tomar en cuenta cuál es el voltaje que la memoria requiere para ser programada puesto que existe memorias de la misma capacidad con distinto voltaje de programación. En caso de no indicarse el voltaje de programación, la opción por default será de 21[V] con un algoritmo lento de programación.

El usuario podrá tener acceso al cambio de voltaje de programación únicamente en el caso de las memorias EPROM; puesto que, en el caso de microcontroladores, los voltajes de programación vienen especificados de acuerdo al tipo.

Es posible habilitar esta opción después de seleccionar una memoria EPROM.

- CHEQUEAR MEMORIA

Mediante esta opción se puede chequear si una memoria se encuentra en blanco, o no. Para esto se realiza una lectura de la memoria y se compara cada dato de la misma con: el

valor FF para el caso de las memorias EPROM y los microcontroladores de la familia MCS51, o con 00 en el caso de los microcontroladores de la familia MCS48. Si todas las localidades tienen el valor dado, el software deberá enviar el mensaje: "LA MEMORIA SE ENCUENTRA EN BLANCO"; caso contrario el mensaje dado será: "LA MEMORIA NO SE ENCUENTRA EN BLANCO".

- VERIFICACION DE LA MEMORIA

La verificación consiste en; teniendo un archivo.hex en el buffer de memoria del computador, realizar una operación de lectura de la EPROM y comparar para ver si los contenidos son iguales. En caso de que los contenidos no sean iguales se especifica cual es la localidad en la cual los datos no coinciden. En caso de que estos valores no coincidan se considera como dato correcto el que se encuentra en el buffer de memoria y el incorrecto será el que se haya leído de la memoria.

En el caso de que la verificación de la MEMORIA sea satisfactoria, el software imprime "VERIFICACION O.K."; caso contrario, se indicará la localidad en la que no coinciden los datos; además, el dato correcto, el incorrecto y una etiqueta de "VERIFICACION NO OK".

- EDITAR BYTES

Esta opción permite editar una localidad de memoria. Se ingresa la dirección de la localidad de memoria a ser editada visualizándose con esto, el dato de la localidad; luego el sotfware esperará por un nuevo dato a ser

ingresado. Si se presiona solamente enter el dato no cambia y el software permitirá avanzar a la próxima localidad de memoria.

Al presionar la tecla ESC puede abandonarse esta opción.

- BIT DE SEGURIDAD

Esta opción solamente la tienen los microcontroladores de la familia MCS51 y consiste en grabar un bit que impide posteriores lecturas de la memoria del microcontrolador. Esta opción puede ser habilitada únicamente después de seleccionar "programar", siempre y cuando, el ingreso a menú de opciones se lo haya hecho seleccionando previamente un elemento de la familia MCS51.

Se definen dos operaciones básicas que realiza el software del microcontrolador: operación de lectura y operación de programación con verificación.

El software para el microcontrolador se encargará de recibir comandos desde el computador que le permiten saber:

1. Cuál es el elemento que se va a programar
2. Cuál es el voltaje de programación (Palabra de control).
3. El tamaño de la memoria
4. En caso de memorias, el tipo de algoritmo de programación.
5. Para microcontroladores de la familia MCS51 indica si se quiere o no bit de seguridad.
6. Si la operación que se va a realizar es una operación

de lectura, o de escritura.

Tanto el computador como el programador deben tener una forma de comunicación. La comunicación como se especificó en el diseño del hardware es vía serial. El tipo de comunicación es HALF DUPLEX. En este caso existe comunicación en ambos sentidos pero el que da la iniciativa es el computador, el mismo que, al enviar un comando, debe esperar una respuesta de aceptación del microcontrolador. Existe una validación de respuestas del microcontrolador, en el PC.

Al existir un fallo en la comunicación se debe prever que el microcontrolador se resetee por medio de software; mientras que, el programa en QBASIC debe indicarle al usuario que existe algún error en la comunicación. De esta manera se logra que no exista dependencia total de operación del uno y del otro.

Al iniciar la comunicación se considera el reseteo por software del microcontrolador a la localidad 0050H en caso de perderse la comunicación por más de un segundo.

3.2 RUTINAS PARA PROGRAMACION

Al tener comunicación serial, HALFDUPLEX, se necesita especificar la estructura del programa que se va a utilizar para realizar la atención a la misma.

El diagrama 1 presenta un tipo de estructura de programa para comunicación HALF DUPLEX. Esta estructura permite que el programa principal pueda ser interrumpido cada vez que el computador requiera enviar un comando o un dato.

Seguidamente el microcontrolador ejecutará alguna acción en respuesta.

El diagrama 2 es otra opción para comunicación serial HALF DUPLEX; en este caso el microcontrolador tiene asignado un lugar especial en el programa principal para averiguar si existe algún dato en el SBUF, siendo así procederá a la lectura del dato que se encuentra ahí y dará atención a lo que le pide el computador, caso contrario continúa con el programa principal.

El diagrama 3 constituye una mezcla de los dos anteriores, el computador puede enviar un dato cuando desee, el microcontrolador atiende a la interrupción serial, almacena el dato en algún lugar especial y setea una bandera. Pero la atención a la petición del computador se realizará cuando el microcontrolador lo decida.

De los tres el que más nos conviene es el primero puesto que el computador es el que dirige en este caso la comunicación. El PC envía un dato en cualquier momento y el microcontrolador dará una respuesta de aceptación. El microcontrolador solamente se encuentra a la espectativa de lo que le envíe el computador. Si +tuviera que realizar otras funciones; tal vez, convenga utilizar el esquema dos o tres. Además la atención al dato proveniente del PC, para el propósito del programador, debe ser inmediata.

El diagrama 4 presenta un diagrama de bloques del programa del microcontrolador.

Existe un inicio donde se realiza asignación de etiquetas a

memoria RAM externa, inicialización del SP, habilitación de interrupción serial, habilitación de interrupción del TIMER0, programación de los modos de los timers, se especifica el tipo de comunicación (inicializacion de registros de control del microcontrolador), se carga TH1 Y TL1 para la velocidad de comunicación, se arranca el TIMER 1 que genera el baud rate, y además se programa el ampliador de puertos 8255.

El microcontrolador, después de la inicialización, espera el envío de comandos por parte del computador. Existen 2 comandos o códigos que envía el PC los mismos que son RX y RY. Estos comandos le dan toda la información del elemento que se quiere programar, esto es: tamaño, voltaje de programación, tipo y además en el caso de memorias se indica si se usará algoritmo rápido o lento de programación.

Con la información de estos 2 registros el microcontrolador ejecutará subrutinas para inicialización de los "sockets", de esta manera se asegura que existan las condiciones adecuadas de voltaje en los sockets al momento de colocar el chip.

Si el computador envía un "55H" quiere decir que lo que se va a programar es un 8748 y se procede a la revisión de la señal del ALE del mismo.

Si el comando enviado es 40H lo que se pide al programador es la grabación del bit de seguridad en microcontroladores de la familia MCS51.

Con un 00H la operación es de lectura y con un valor comprendido entre 01H y 0FH (número de datos de una línea) la operación a realizarse es la programación de la memoria. Después de realizar cualquier operación o al producirse una falla de comunicación, el socket siempre regresa a condiciones iniciales.

En el diagrama 5 se presenta el diagrama de bloques para operación de programación del microcontrolador. Los datos recibidos por el PC son el # de bytes de una línea así como la dirección de inicio, el separador y el contenido de la línea (datos a ser programados). Se inicia la operación de programación de byte a byte de una línea. Después de cada programación se realiza una verificación. Al programar una línea el socket regresa a condiciones iniciales.

La operación de lectura se esquematiza en el diagrama 6. Esta operación es más sencilla que la operación de programación pues lo único que se requiere es manejar adecuadamente señales de CS y OE. Al terminar esta operación el socket regresa a condiciones iniciales.

ESTRUCTURA DEL PROGRAMA PARA COMUNICACION HALF DUPLEX

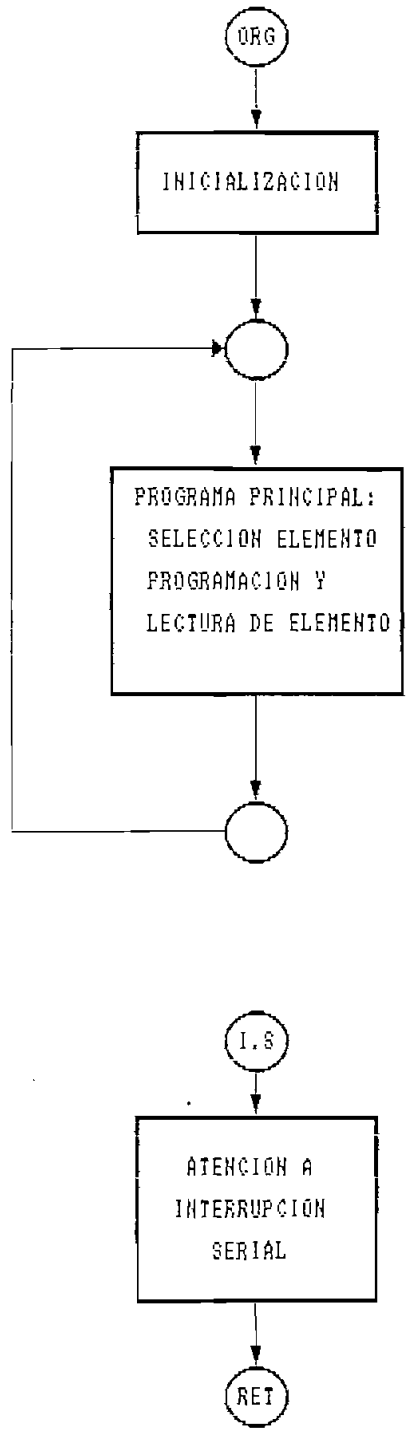
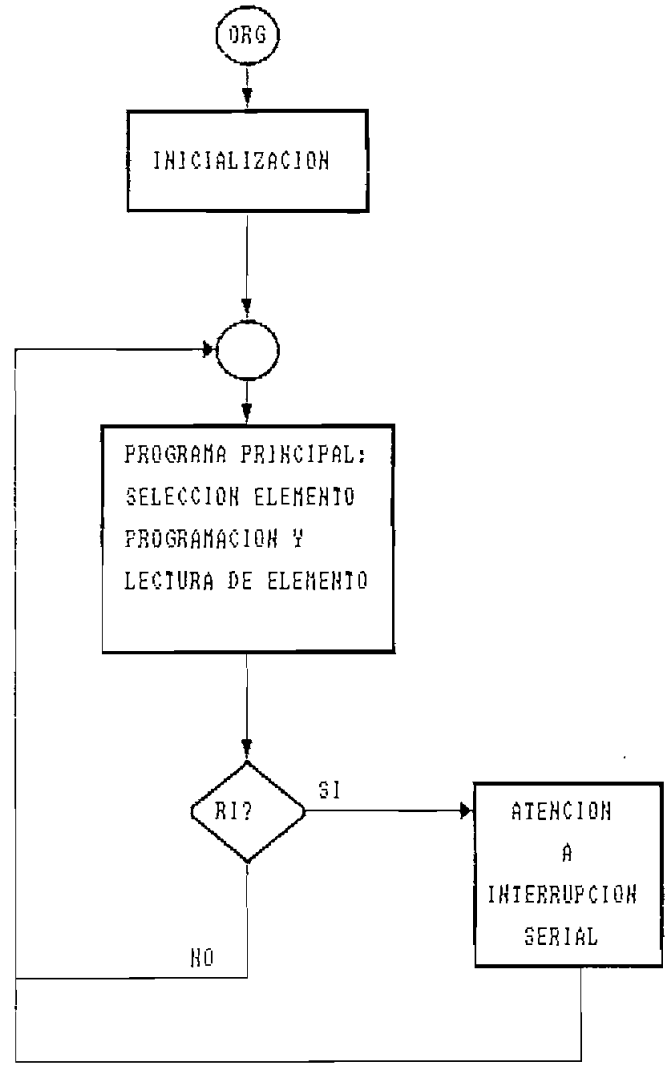


DIAGRAMA 1

CASO A

PARA MICROCONTROLADORES

ESTRUCTURA DE PROGRAMA PARA COMUNICACION HALF DUPLEX

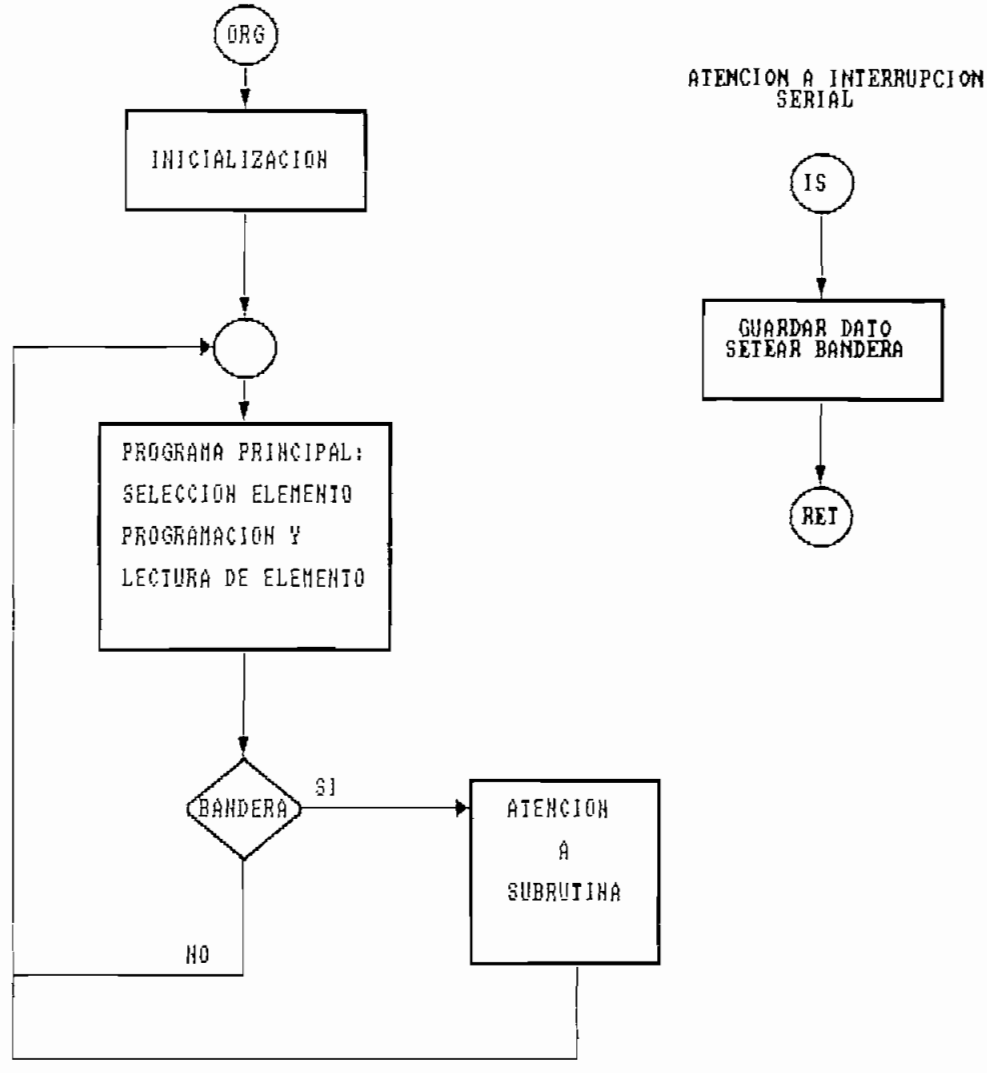


CASE B

PARA MICROCONTROLADOR

DIAGRAMA 2

ESTRUCTURA DE PROGRAMA PARA COMUNICACION HALF DUPLEX



CASE C

PARA MICROCONTROLADOR

DIAGRAMA 3

DIAGRAMA DE BLOQUES DEL PROGRAMA PARA EL MICROCONTROLADOR

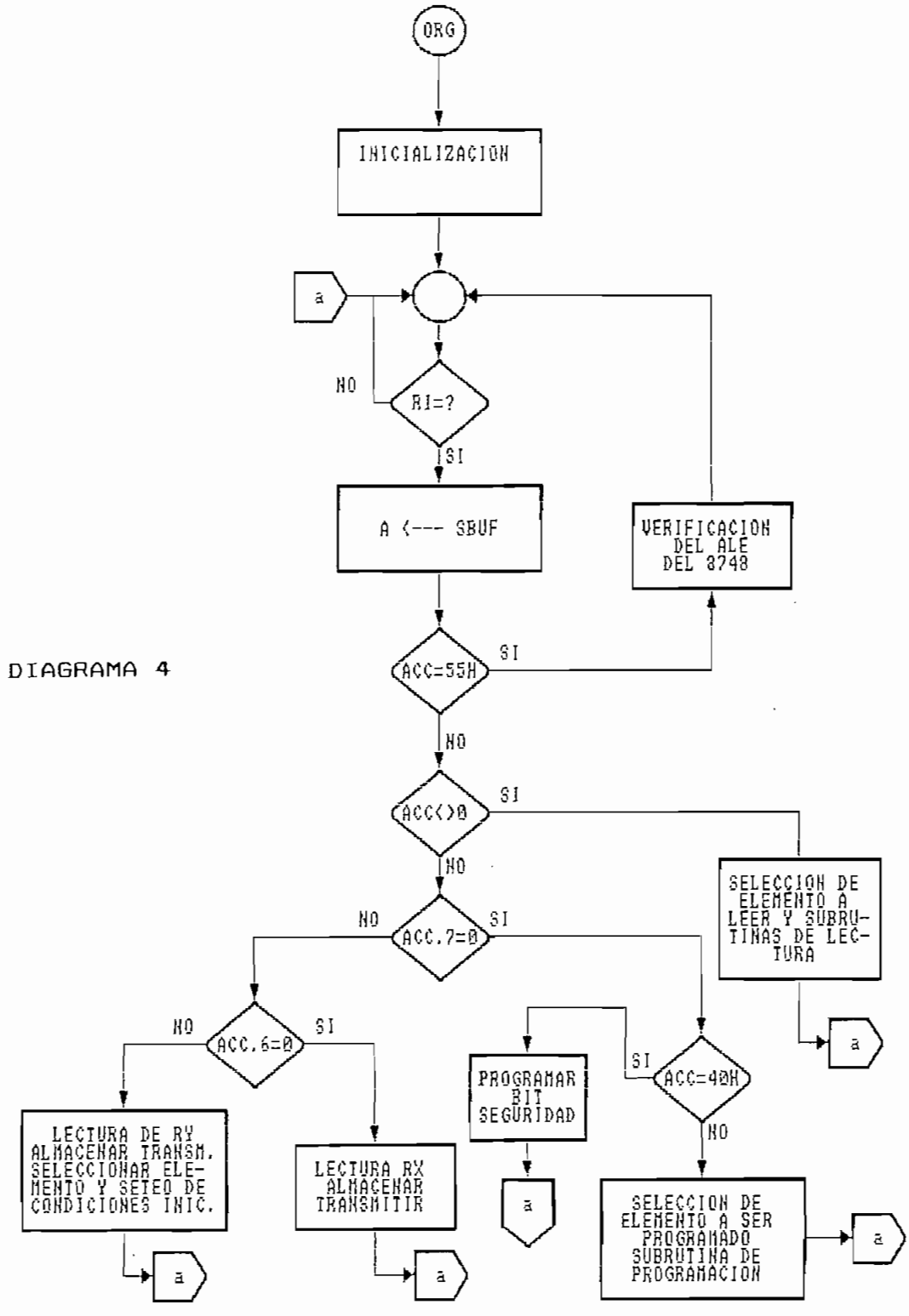


DIAGRAMA 4

DIAGRAMA DE FLUJO PARA PROGRAMACION (MICROCONTROLADOR)

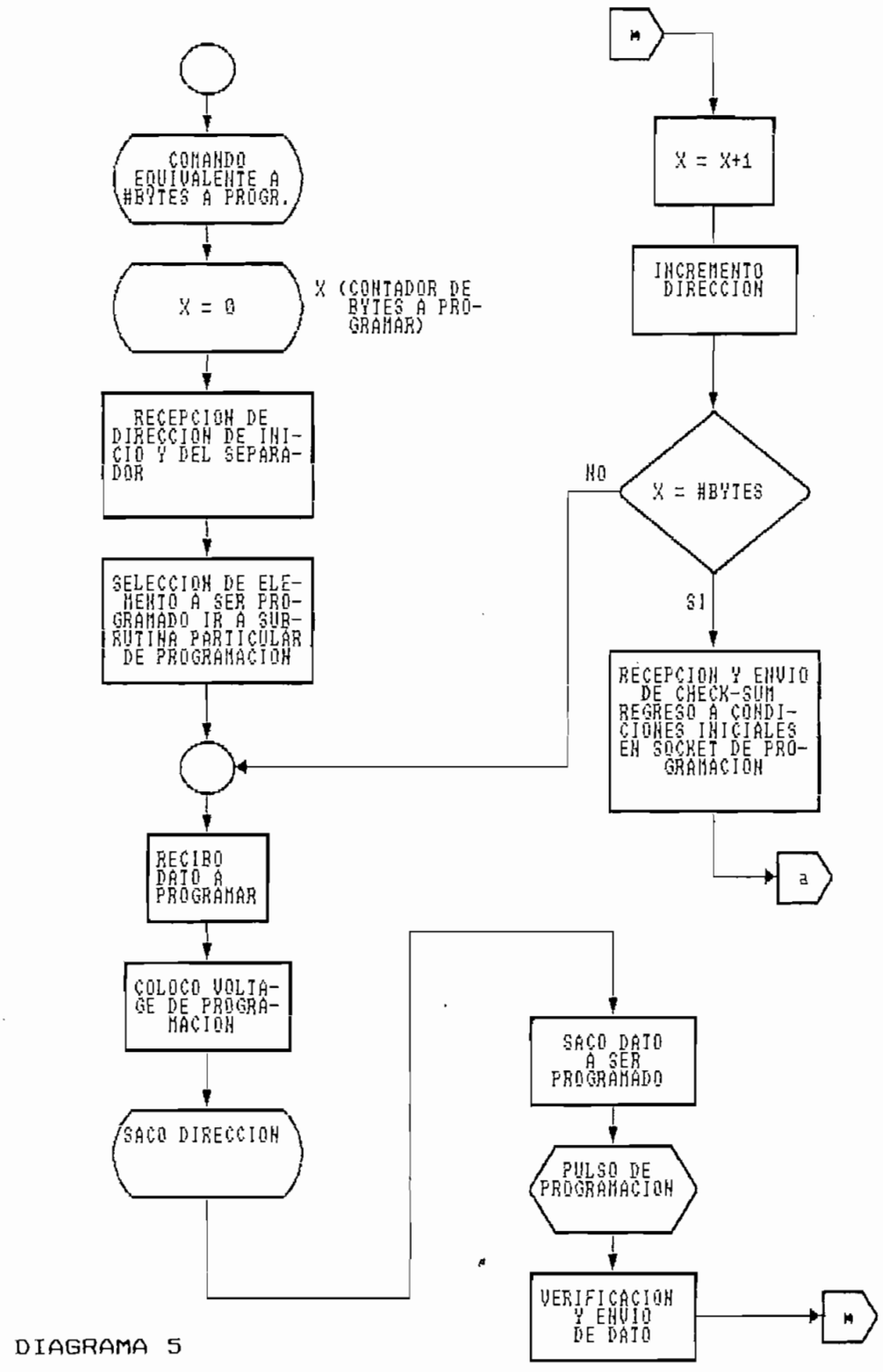


DIAGRAMA 5

DIAGRAMA DE FLUJO PARA LECTURA (MICROCONTROLADOR)

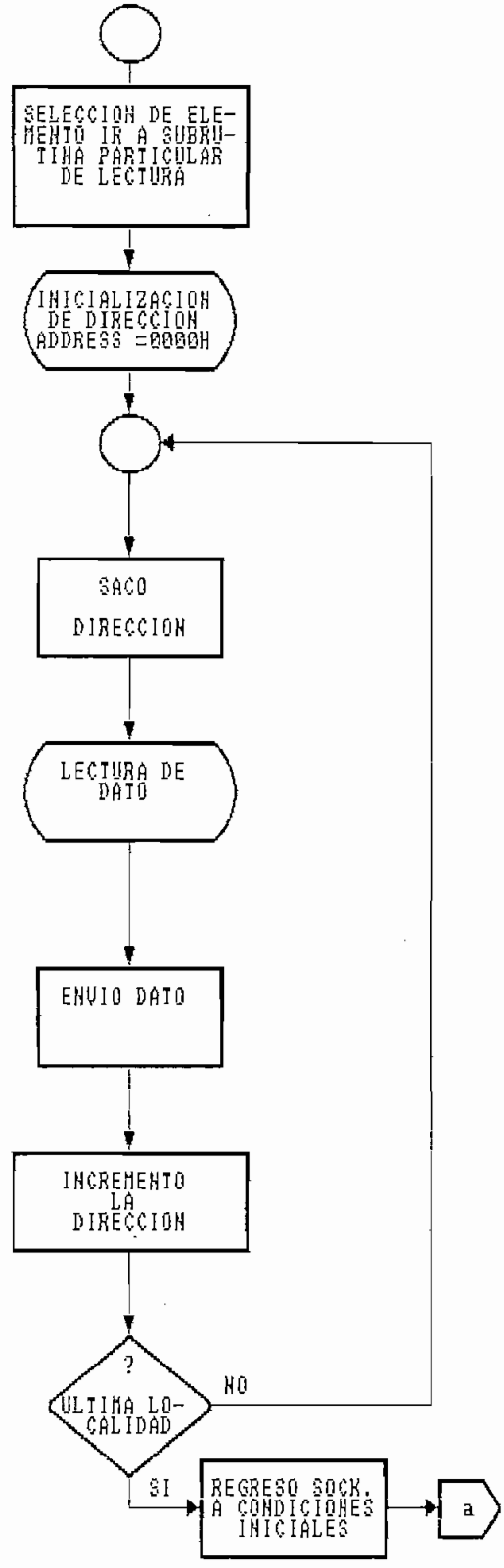


DIAGRAMA 6

3.3 RUTINAS AUXILIARES

Dentro de estas rutinas auxiliares tenemos: la rutina para atención a interrupción serial; rutinas de seleccion de elemento e inicialización de socket; rutinas para selección de memoria; subrutina de atención a TIMERO en caso de que exista falla el momento de la comunicación.

La subrutina de atención al TIMERO permite dar un segundo de tiempo antes de que se realice un reset por software del microcontrolador. El reset ubicará condiciones iniciales en el socket de programación y ubicará el contador del programa en 0050H.

La rutina de atención a interrupción serial permite almacenar en un registro especial el dato enviado por el computador, así como también realizar seteo de banderas cuando se realice transmisión o recepción.

Las rutinas de selección de elemento son del tipo de comparación de contenido de algún bit.

El protocolo de comunicación se encuentra en los Anexos, en el mismo que se puede apreciar el significado de cada bit de los comandos enviados desde el computador.

Los diagramas de flujo de atención a estas subrutinas auxiliares se presentan en los diagramas 7, 8, y 9.

3.4 PROGRAMAS EN QBASIC

Puesto que estos programas son los que se encuentran en directa comunicación con el usuario se ha desarrollado presentación de pantallas de fácil uso. En estas pantallas existe selección del elemento que se desea programar; así

como, selección de la función que se desee que el programador realice.

El diagrama de flujo del programa principal en QB45 se indica en el diagrama 10. Aquí el inicio corresponde a pantalla de presentación, así como inicialización de variables, etc. Luego aparece un menú en el que se selecciona uno de los tres elementos a programar, presentando adicionalmente una opción para salir del programa.

Una vez seleccionada la familia en la que se encuentra el elemento a ser programado, se nos presenta un menú en el que se puede seleccionar el elemento específico a programar. Como puede verse en este punto las tres familias tienen selección independiente de elemento.

Una vez seleccionado el elemento se envia desde el computador el contenido de los registros RX y RY, los que serán decodificados por el microcontrolador. Si este envío fue satisfactorio aparece un menú de selección en el que se encuentran básicamente las funciones del programador y que se indicaron anteriormente en este capítulo.

El diagrama 11 presenta el flujo del programa para lectura-chequeo de la memoria. El comando para que inicie la lectura es un 0. En el programa se prevé el caso en que exista una falla de comunicación durante la lectura, con un prudente tiempo de espera en la respuesta del microcontrolador.

El dato leído es almacenado en el buffer de memoria. En

caso de lectura existirá una aceptación de la recepción del dato; mientras que, en el caso de chequeo, el valor recibido se lo compara con el valor para el cual se considera que la memoria se encuentra en blanco. La operación se repite en caso de lectura hasta el momento que se acabe de direccionar todas las localidades de memoria, y en el caso de chequeo hasta que alguna de las localidades no se encuentre en blanco, o hasta que se termine de leer toda la memoria ocurriendo efectivamente que la memoria se encuentra con todas las localidades en blanco.

El diagrama 12 corresponde al flujo de programa para verificación de la memoria. Para poder realizar la verificación de memoria es necesario que de antelación exista en el buffer de memoria un archivo.hex. Se realiza una comparación entre el contenido del buffer de memoria del PC y el contenido de la EPROM. Al igual que el anterior caso existe un lazo interno que en caso de falla en la comunicación indicará al usuario esta situación. Al tener iguales contenidos la verificación será considerada OK, caso contrario se indicará la dirección de la localidad en la que los datos no coinciden; así como el dato correcto y el incorrecto. Se considera correcto el dato que se encuentra en el buffer de memoria del computador.

El diagrama 13 representa en bloques el programa utilizado para la operación de programación.

El programa realiza un desempaquetado del archivo.hex de acuerdo al formato INTEL y envía: # de datos, dirección de

inicio, separador, los datos de cada línea y el checksum. Esto se repite hasta que se haya llegado al final del archivo indicado por el EOF. Cada vez que se envía un dato a ser programado, existe luego una comprobación con el dato que el microcontrolador envíe después de la verificación. En caso de que no exista compatibilidad entre los dos se paraliza la programación dando una indicación de error de programación.

El diagrama 14 presenta en bloques el flujo del programa para mostrar el contenido. Esta opción no necesita comunicación con el programador. Se imprime el contenido del buffer de memoria en páginas de 256 bytes con opción de avanzar (PgUp) o retroceder (PgDn) a lo largo del contenido del buffer de memoria.

El diagrama 15 presenta el flujo del programa para grabar en archivo. Esta opción no necesita comunicación con el programador. Se almacena el contenido del buffer de memoria en un archivo que se abre como de salida.

El diagrama 16 presenta el flujo de programa para cargar un archivo.hex. En este caso se ingresa el nombre del archivo.hex el mismo que se va a almacenar en el buffer de memoria. Es necesario realizar un desempaquetado del mismo para tener en el buffer de memoria solamente los datos del programa. Esta opción tampoco presenta comunicación con el programador.

El diagrama 17 presenta en bloques el flujo de programa para la edición de bytes. Esta opción no requiere de

comunicación con el programador. En este caso se ingresa la localidad que se desea editar y automáticamente aparece el dato de la localidad especificada, posteriormente este dato puede o no ser cambiado con un nuevo dato.

Puesto que las velocidades entre distintos computadores varía y para optimizar la comunicación entre el computador y el equipo existe un programa adicional de instalación cuyo diagrama de flujo se indica en el diagrama 18.

DIAGRAMA DE BLOQUES PARA SELECCION DE ELEMENTO E INICIALIZACION DE SOCKET DE PROGRAMACION

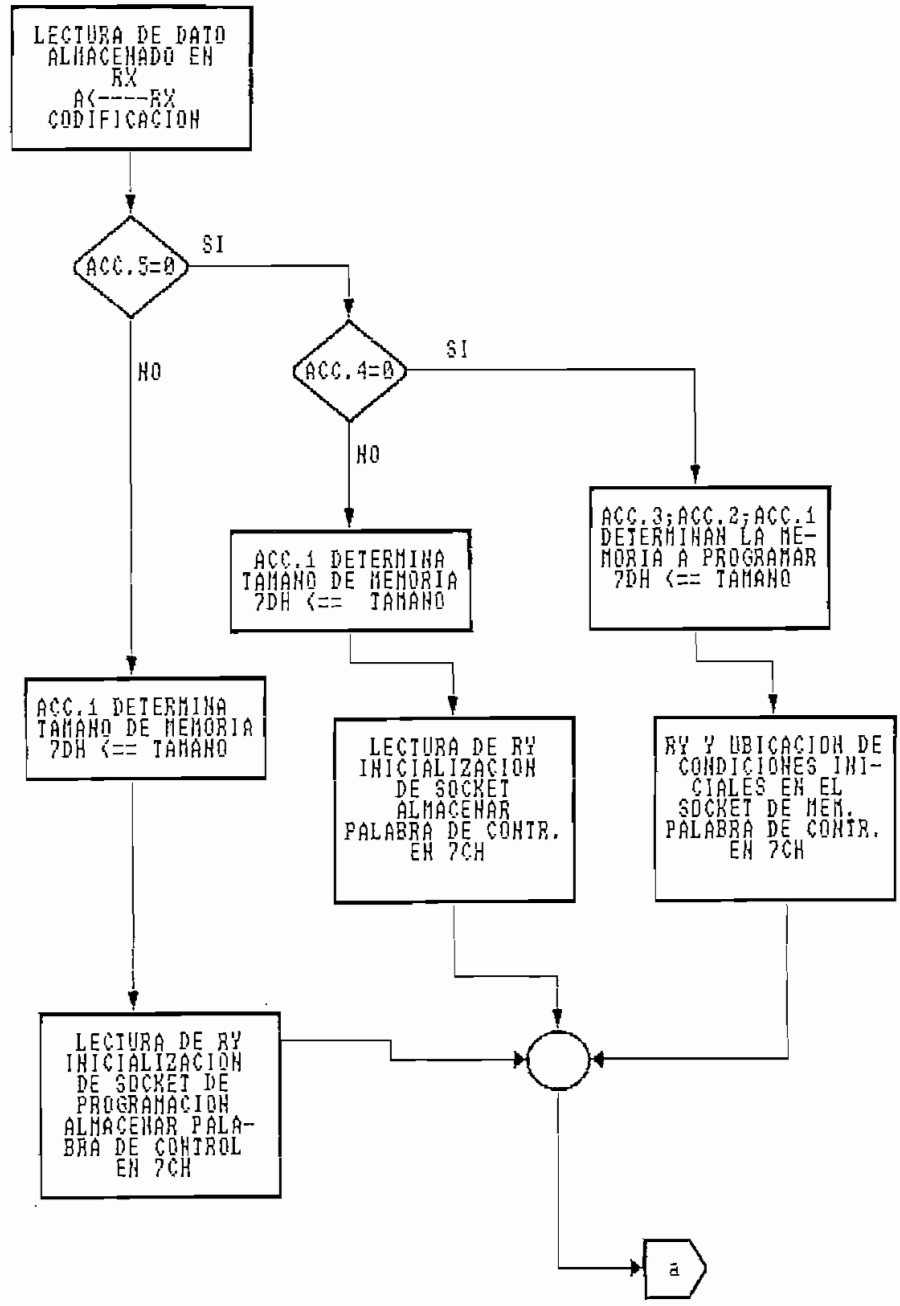


DIAGRAMA 7

RUTINA DE SELECCION DE MEMORIA (MICROCONTROLADOR)

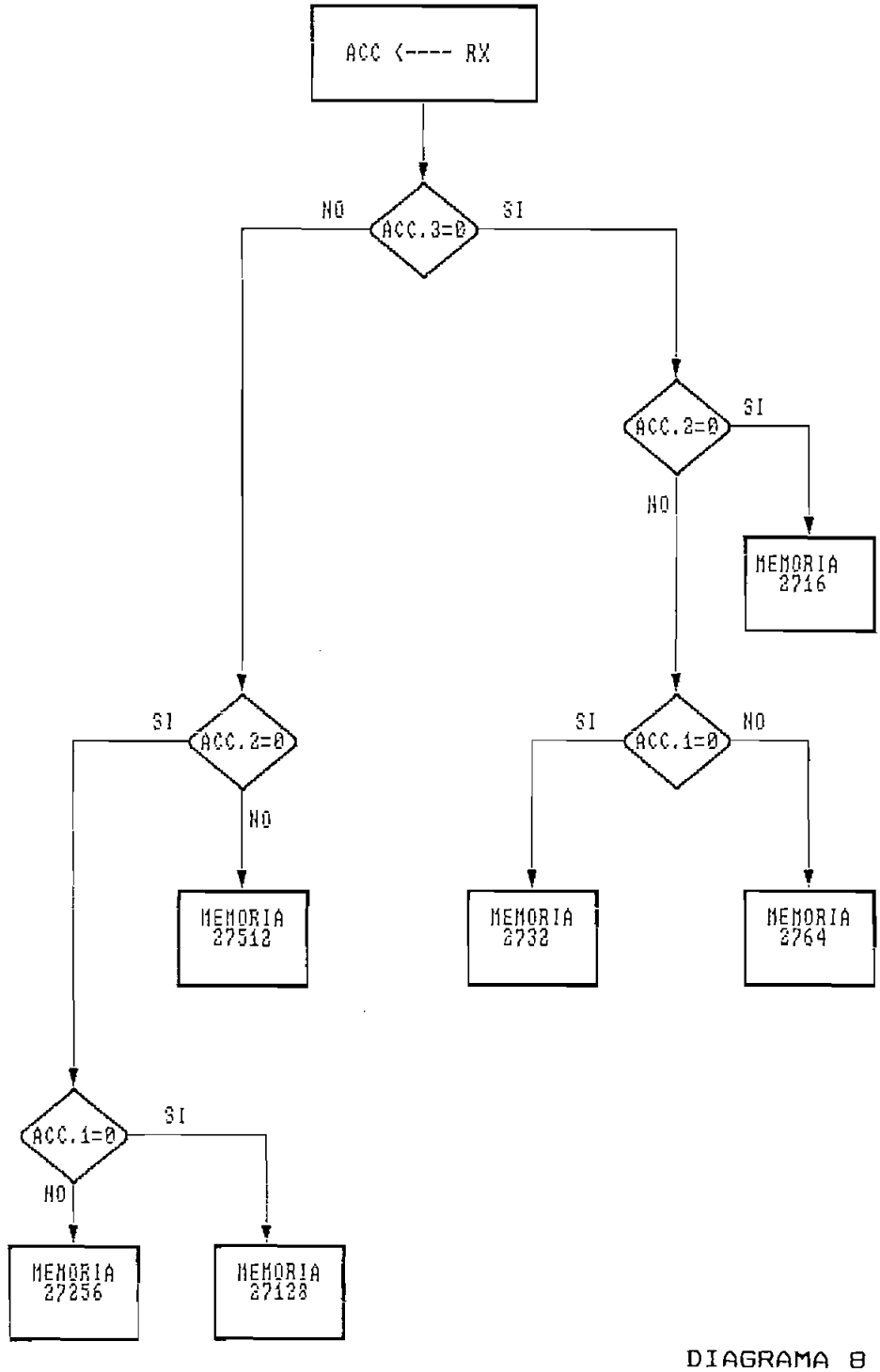
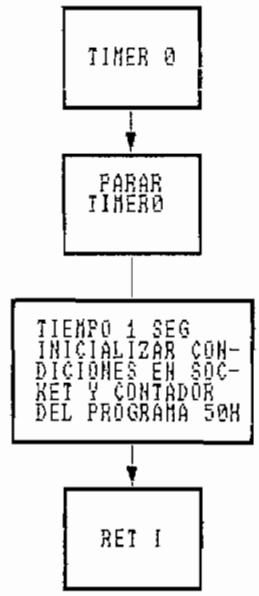


DIAGRAMA 8

SUBRUTINA PARA TIMER 0



SUBRUTINA PARA COMUNICACION SERIAL

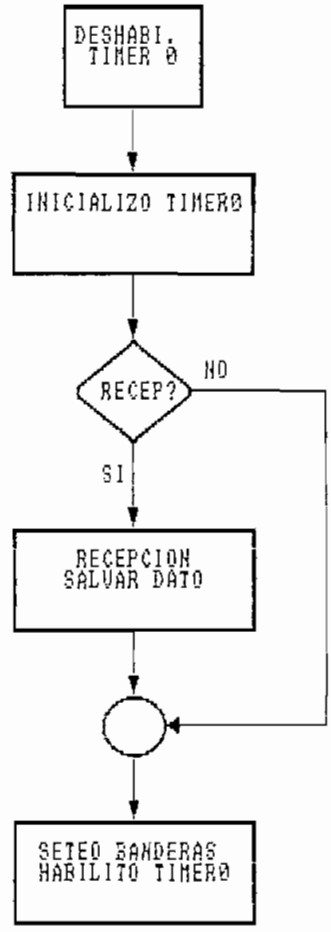


DIAGRAMA 9

DIAGRAMA DE FLUJO DEL PROGRAMA EN QB45

DIAGRAMA 10

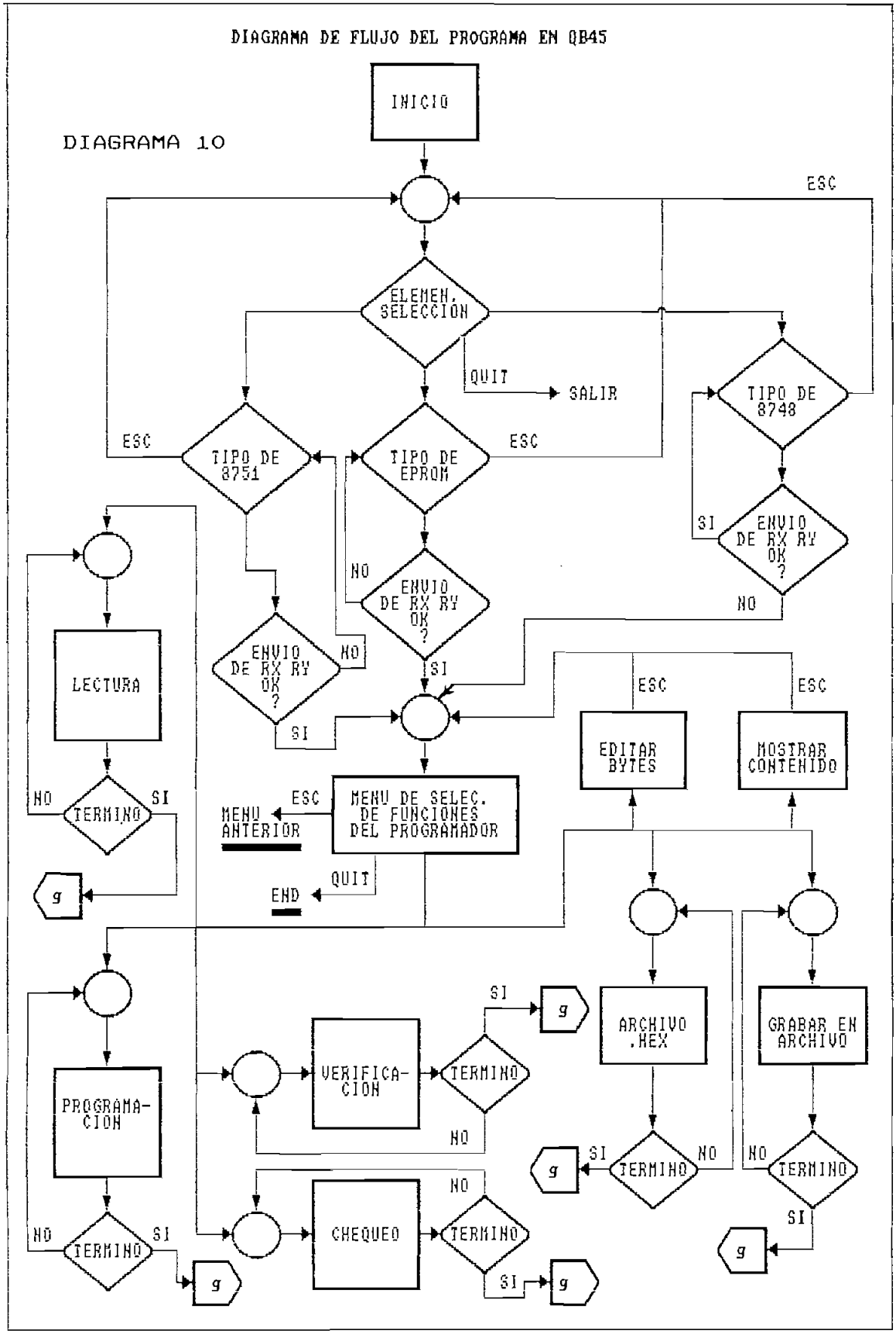


DIAGRAMA DE FLUJO PARA LECTURA-CHEQUEO (QB45)

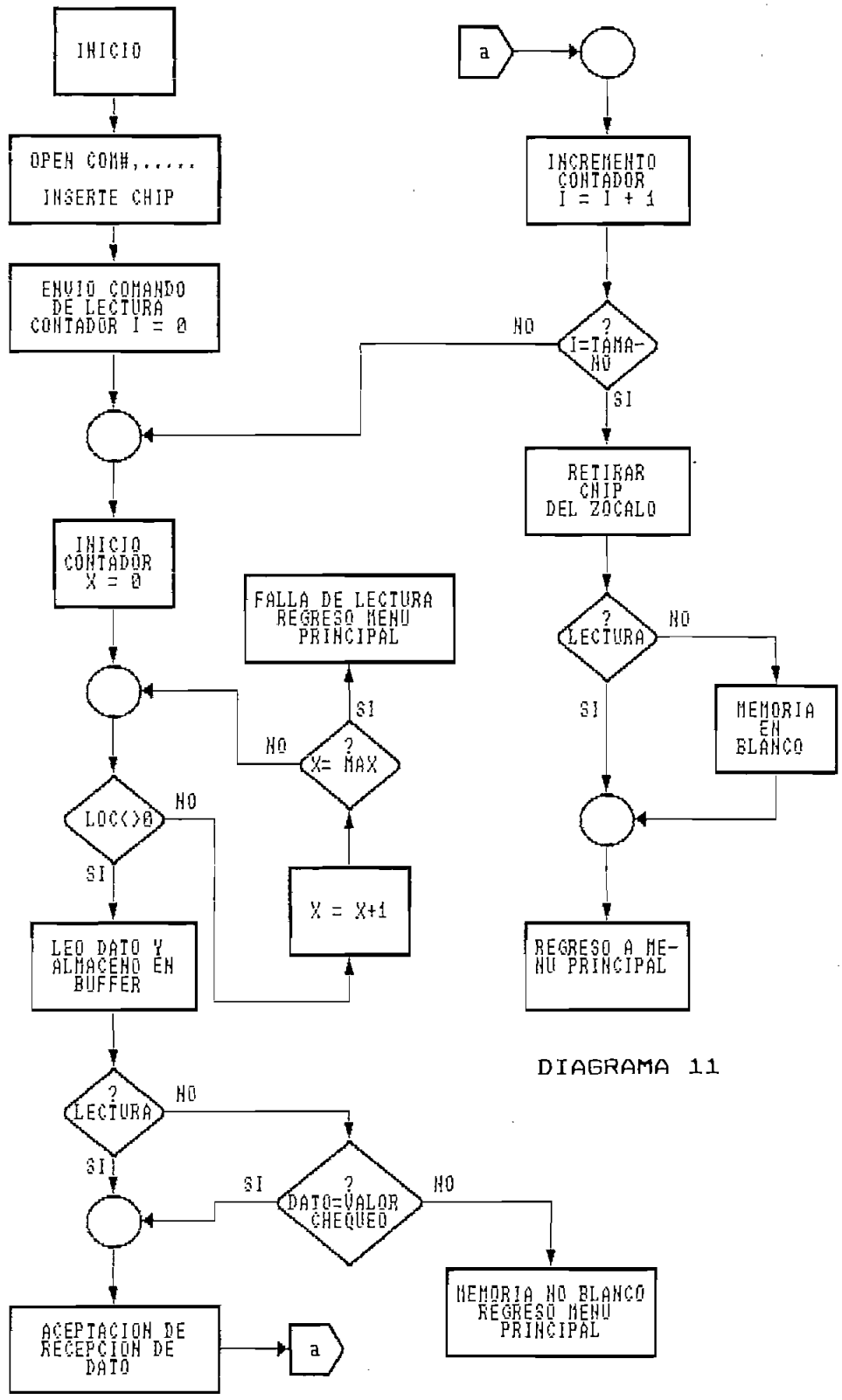


DIAGRAMA 11

DIAGRAMA DE FLUJO PARA VERIFICACION (QB45)

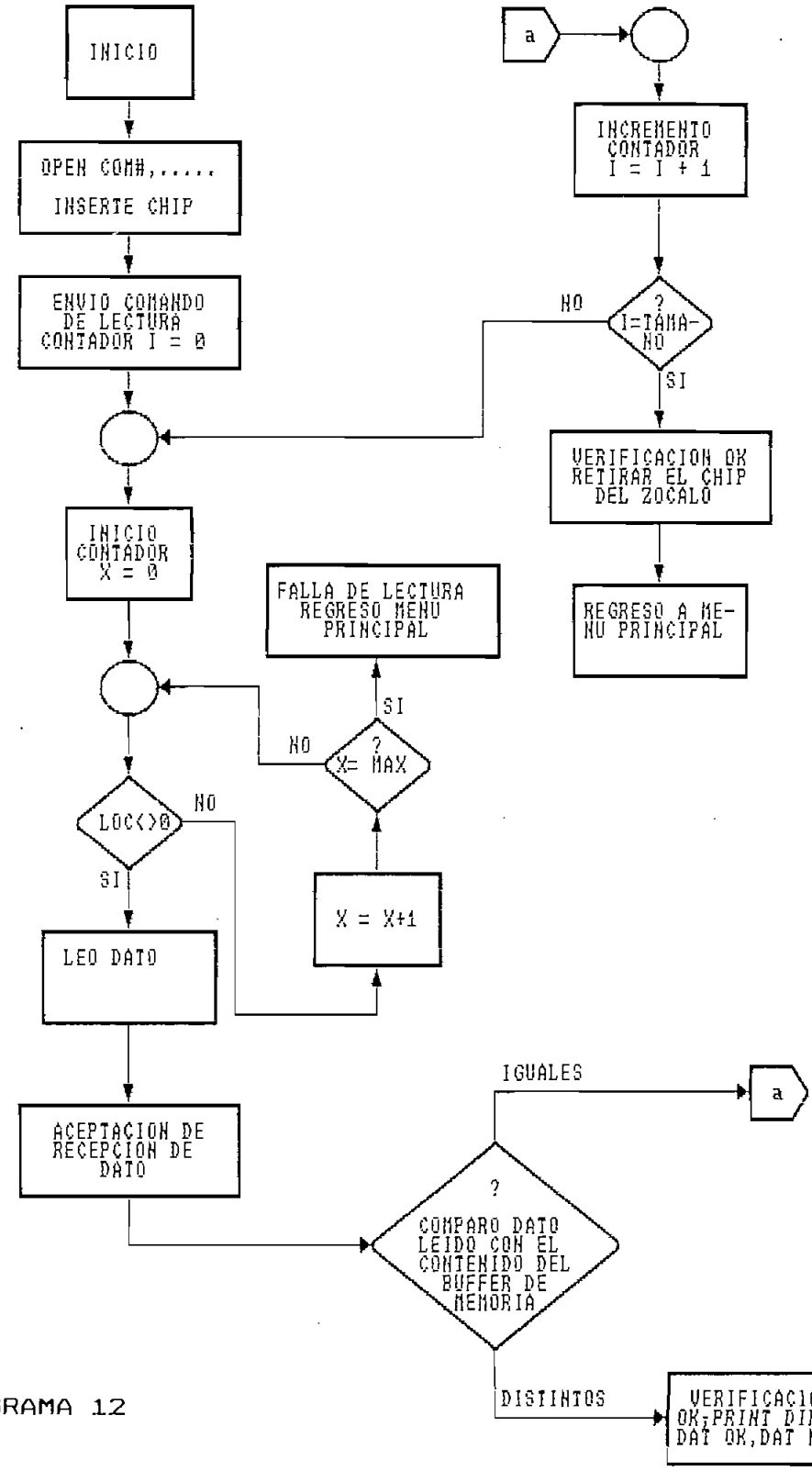


DIAGRAMA 12

DIAGRAMA DE FLUJO PARA PROGRAMACION (QB45)

DIAGRAMA 13

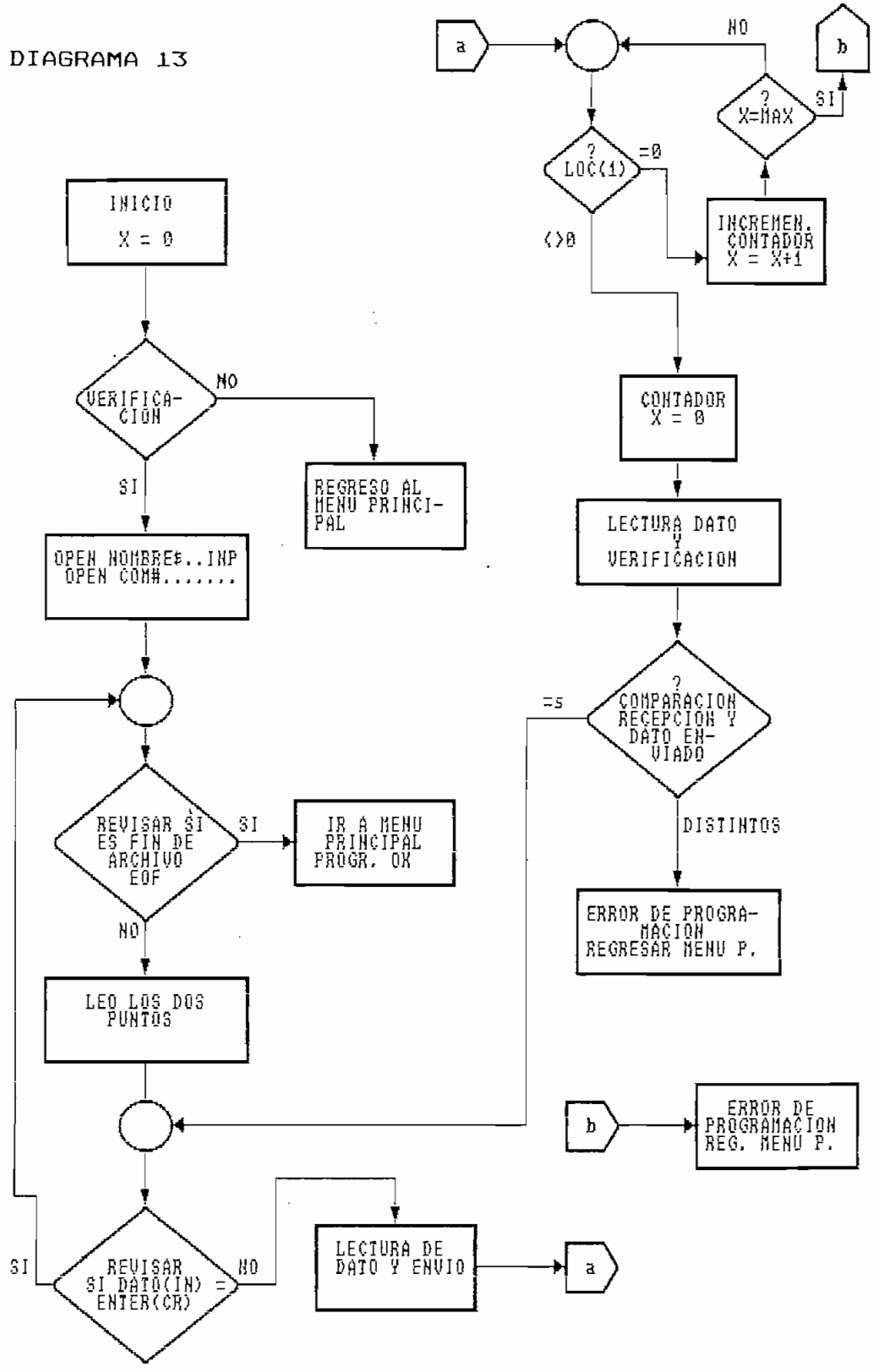


DIAGRAMA DE FLUJO PARA MOSTRAR CONTENIDO (QB45)

DIAGRAMA 14

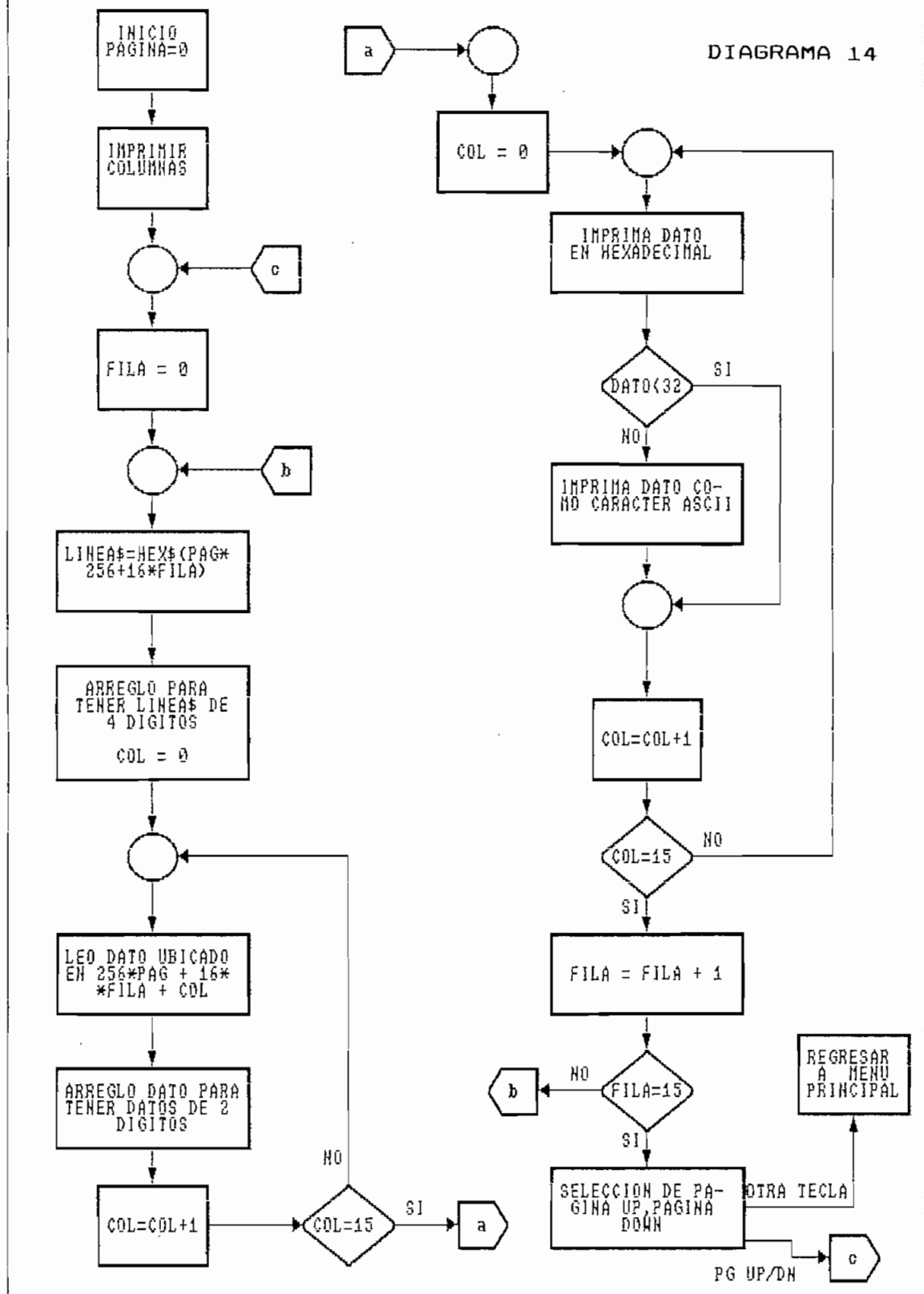


DIAGRAMA DE FLUJO PARA GRABAR EN ARCHIVO (QB45)

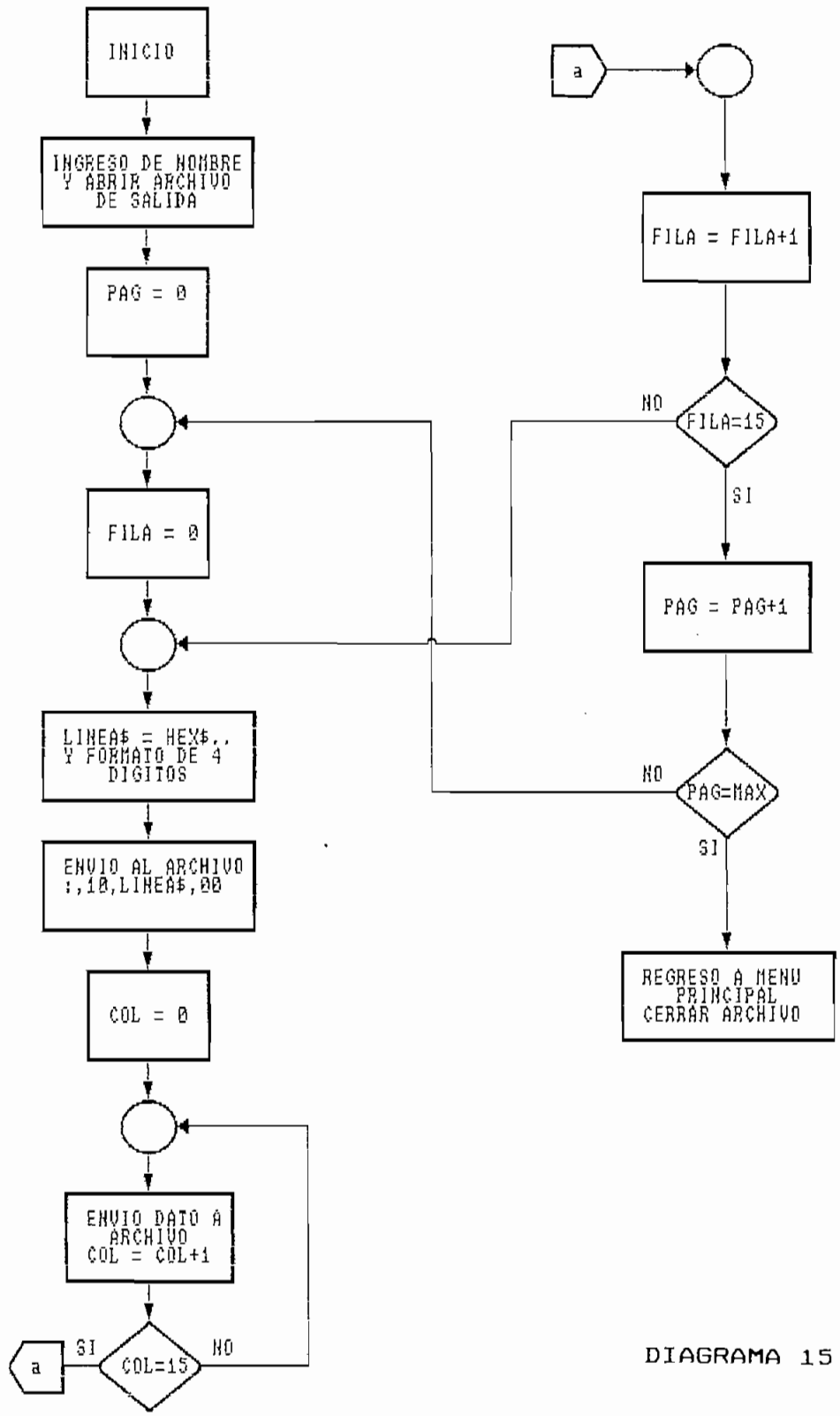


DIAGRAMA 15

DIAGRAMA DE FLUJO PARA ARCHIVO.HEX (QB45)

DIAGRAMA 16

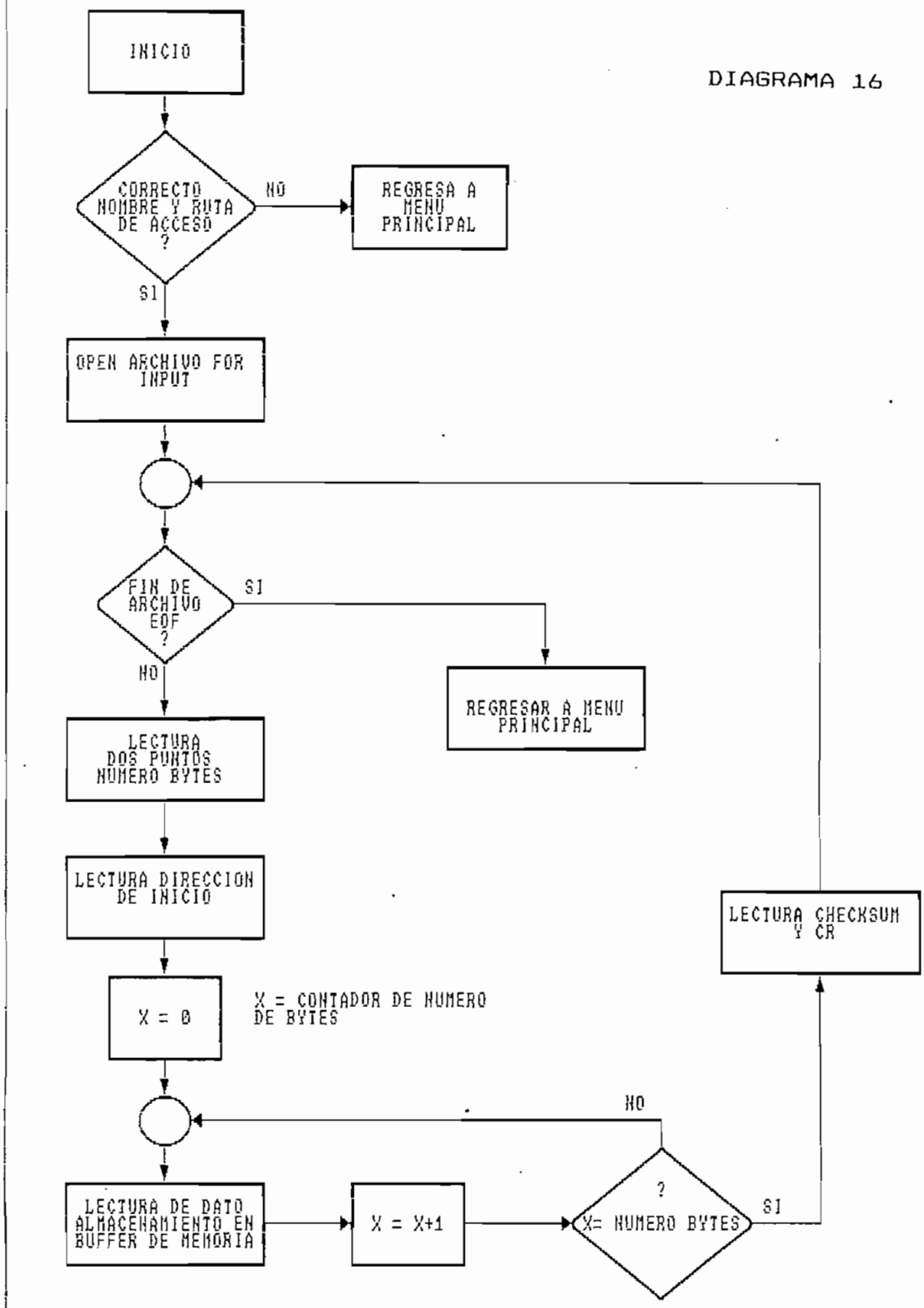


DIAGRAMA DE FLUJO PARA EDITAR BYTES (QB45)

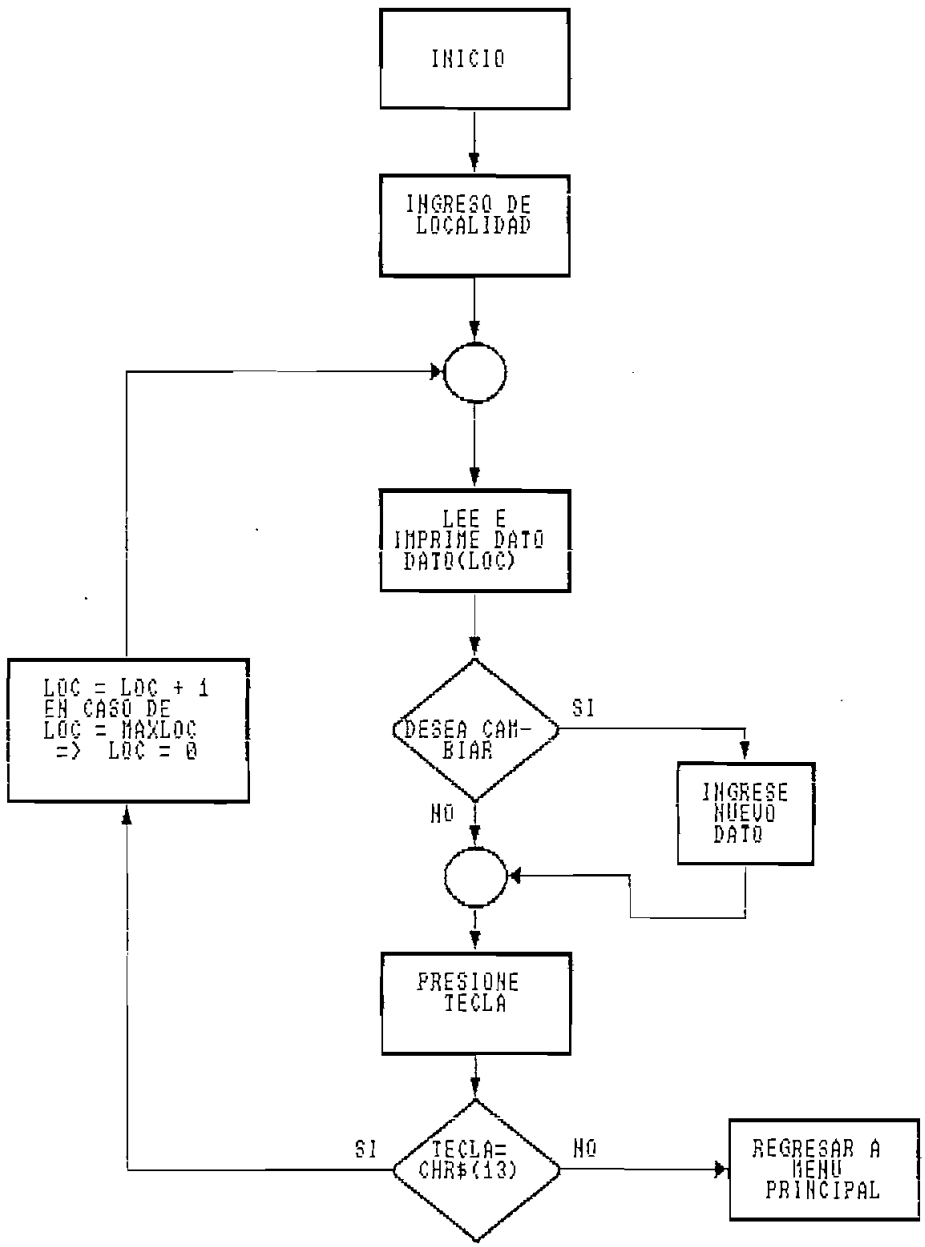


DIAGRAMA 17

RUTINA DE INICIALIZACION DEL PROGRAMADOR

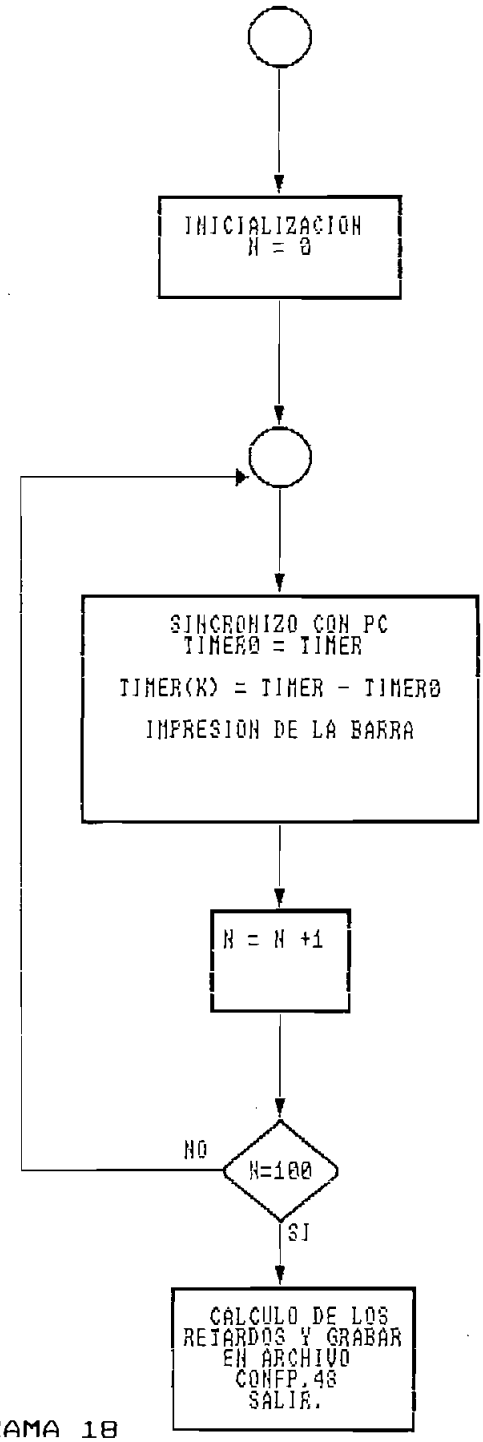


DIAGRAMA 18

CAPITULO IV

PRUEBAS Y RESULTADOS

Puesto que las pruebas efectuadas para las memorias fueron las mismas y al ser los resultados similares se indicará el set de pruebas y resultados únicamente para una de las memorias que fue programada. Se presenta el set de pruebas asi como los resultados para la memoria 2716 la misma que requirió un voltaje de programación de 25 voltios y algoritmo lento de programación.

Primeramente se realizó un chequeo de la memorias el mismo que permitió ver el estado de la misma.

Con la comprobación de que la memoria se encontraba en blanco se procedió a la programación, lectura y verificación de la misma.

El archivo con el que se trabajó para realizar la programación es el siguiente:

AD.HEX

```
:03000000020030CB
:0300230002008652
:10003000758130758907587007589207598507501
:100040008DF07588F0D28EC2003000FDC200E57FCE
:10005000B400F6E4FFFE7910906000E0C32EFEE4E9
:100060003FFFD9F4EF540FC4FFEE54F0C44FFFF537
:10007000999A000E4F0A3F0A3EF54F0C4F0A3EF34
:10008000540FF0020049209906E599F57FD200C28D
:0400900099C2983247
```

Se utilizó la opción de cargar archivo.hex, previo para la

programación del chip.

Aprovechando que el archivo se encontraba en el buffer de memoria se utilizó la opción de mostrar contenido. En la figura 4.1 se presenta la presentación en pantalla del contenido del buffer en la página 1 de 256 bytes. La presentación es tanto en hex como en ASCII.

Seguidamente se procedió a la programación de la memoria, mediante la utilización de la opción de programar. Previamente el momento de seleccionar la memoria se realizó la selección del voltaje de programación.

Una vez realizada la programación se utilizó la opción de verificación de programación. Al tener ya un archivo en el buffer de memoria el programa realizó la verificación de la programación, obteniéndose un resultado de verificación ok.

Se procedió a cargar en el buffer de memoria un archivo distinto al anterior y que se indica a continuación:

PRO.HEX

```
:0300000020030CB
:03000B0002092FB8
:030023000209418E
:1000300075813075A89275870075892175985075FE
:100040008DF8758BF8D28E758C00758A00790AC28E
:10005000BCC200C2B2C2B3C2B5C2B478037480F21B
:1000600074007802F27800F27801F27590FF3000A7
:10007000FDC200E57FF57EF5993000FDC200E57F09
:10008000F57CF5993000FDC200E57FF57DF59930EE
:1000900000FDC200E57FF57BF599E57E6005B401C2
```

:1000A0000580060202EF0201CF7802E57CF2C2B2BF

:1000B000D2B3C2B4D2B5757A00E578B40103020085

:1000C000C402018A3000FDC200E57FB40030E4FDC7

:1000D000FE74FFF5907800EDF27801EEF231CAC28D

:1000E000B331CAE590F5993000FDC200D2B331CAF0

:1000F000C374012DFDE43EFEB57DDA02006EE57F9E

Al realizar nuevamente una operación de verificación se obtuvo como resultado una verificación no OK teniendo como dato correcto el 02H y dato incorrecto el FFH y la dirección en la que se produce la falla en la verificación fue indicada como la 000BH

Para que el buffer de memoria quede nuevamente con los valores fijados de memoria en blanco, se regresó al menú anterior de selección de memoria y después de haber seleccionado nuevamente el mismo elemento se procedió a la lectura con lo cual el resultado presentado fue exactamente el mismo mostrado en la figura 4.1.

Se ejecutó adicionalmente la opción de grabar en archivo. El contenido del buffer de memoria fue almacenado en el archivo prueba.hex. Al salir del programa y editar el archivo el resultado obtenido fue el siguiente:

PRUEBA.HEX

:030000000020030C8

:0300080002092FB8

:030023000209418E

:10003000758130758A9275870075892175985075FE

:10004000BDF8758BF8D28E758C00758A00790AC28E

:100050008CC200C2B2C2B3C2B5C2B478037480F21B

:10006000740078027F2780F27801F27590FF3000A7

:10007000FDC200E57FF57EF5993000FDC200E57F09

:10008000F57CF5993000FDC200E57FF57DF59930EE

:1000900000FDC200E57FF57BF599E57E6005B401C2

:1000A000058006020EF0201CF7802E57CF2C2B2BF

:1000B000D2B3C2B4D2B5757A00E57BB401030200B5

:1000C000C402018A3000FDC200E57FB40030E4FDC7

:1000D000FE74FFF5907800EDF27801EEF231CAC2BD

:1000E000B331CAE590F5993000FDC200D2B331CAF0

:1000F000C374012DFDE43EFEB57DDA02006EE57F9E

Para utilizar la opción de editar bytes con el archivo AD.HEX en el buffer de memoria se procedió a cambiar las localidades desde la 00H hasta la 08H con el valor 55H obteniéndose como resultado el presentado en la figura 4.2. Como puede apreciarse los datos han sido cambiados de su valor original al valor 55H.

Para utilizar los distintos voltajes de programación y los algoritmos de programación se procedió a la programación de la memoria 2732 que utilizó 21 voltios y algoritmo rápido y a la programación de la memoria 27256 de 12.5 voltios (voltaje de programación) y algoritmo rápido de programación. La programacion fue exitosa.

La grabación del chip de seguridad se la realizó en el chip 8751H, la comprobación de que la operación se realizo en forma satisfactoria fue el intentar leer el contenido de la memoria del mismo, leyéndose únicamente FFH

Adicionalmente se trabajó con los microcontroladores de la familia MCS48, y con el microcontrolador 8752H en los cuales se realizó pruebas tanto de programación como de lectura.

CAPITULO V

ANALISIS TECNICO ECONOMICO

A continuación se obtiene el precio total del equipo tomando referencia los precios de los elementos dados por Jameco en el catálogo 394 de agosto-octubre 94. Los precios están en dólares.

ELEMENTO	CANTIDAD	PREC. UNITARIO	PRECIO TOTAL
DIODOS 1N4004	5	.09	.45
DIODOS 1N4148	22	.05	1.1
LEDS ROJOS	4	.11	.44
DIODO ZENER 1N751	1	.19	.19
CAP. 4000µF/50V	1	3.18	3.18
CAP. 1000µF/40V	1	1.09	1.09
CAP. 47µF/50V	1	.14	.14
CAP. 10µF/35V TAN.	1	.65	.65
CAP. .1µF/35V TAN.	10	.19	1.9
CAP. 10µF/50V	4	.09	.36
CAP. 1nF	7	.79	5.35
CAP. 33pF	4	.25	1
TRANSIST. 2N2907A	8	.25	2
TRANSIST. 2N2222	1	.25	.25
CONECTOR DB9 M	1	.45	.45
CONECTOR DB9 F	1	.49	.49
TAPA PLASTICA DB9	1	.39	.39
CONECTOR DB25 F	1	.75	.75
TAPA PLASTICA DB25	1	.39	.39
ALAMBRE 24AWG/4hi.	5 ft.	.29/10ft	.145
SIPS 10/9 10k	2	.27	.54
RESIST. 1K	14	.0179	.2506
RESIST. 120Ω	8	.0179	.1432
RESIST. 180Ω	4	.0179	.0716
RESIST. 8.2k	1	.0179	.0179
RESIST. 330	1	.0179	.0179

ELEMENTO	CANTIDAD	PREC. UNITARIO	PRECIO TOTAL
RESIST. 10k	2	.0179	.0358
RESIST. P. 220Ω	4	.0179	.0716
RESIST. P. 634Ω	1	.0179	.0179
RESIST. P. 3.32k	1	.0179	.0179
RESIST. P. 487Ω	1	.0179	.0179
RESIST. P. 1.47k	1	.0179	.0179
POTENCI. P. 5k	2	.85	1.7
POTENCI. P. 10k	1	.85	.85
POTENCI. P. 25k	2	.85	1.7
CI 7406	2	.45	.9
CI 74LS14	1	.39	.39
CI 74LS373	1	.65	.65
CI 8255	1	2.75	2.75
CI 8031	1	3.59	3.59
CI 2732A	1	3.95	3.95
LM317T	2	.75	1.5
LM317LZ	1	.69	.69
7805T	1	.49	.49
CRISTAL 7.15909	1	1.09	1.09
CRISTAL 4	1	1.09	1.09
PULSADOR	1	0.39	.39
TRANSFORMADOR	1	9	9
SOCKET 40 PINES	6	1.19	7.14
SOCKET 14 PINES	3	.59	1.77
SOCKET 20 PINES	1	.79	.79
SOCKET 24 PINES	1	.85	.85
SOCKET 28 PINES	2	.99	1.98
DISIPADOR	1	.49	.49
AISLADOR	1	.225	.225
ZIF 40 PINES	2	21.95	43.9

ELEMENTO	CANTIDAD	PREC. UNITARIO	PRECIO TOTAL
ZIF 28 PINES	1	16.95	16.95
CABL(6ft)CONECTOR	1	.79	.79
TORNILLOS-SILICON	1	1	1
HEADER DOBLE FILA 18 CONTACTOS	1	.45	.45
CAJA	1	10	10
TARJETA	1	25	25
FUSIBLE/PORTA FU.	1	1.5	1.5
TOTAL			165.73

Precio total es de ciento sesenta y cinco dólares con setenta y tres centavos.

Entre las bondades del equipo se tiene:

No necesita abrir el computador para poder instalar el equipo.

Utiliza un programa de fácil uso de tal manera que el manejo no resulta ser un problema.

El soporte técnico se encuentra a la mano lo cual garantiza un funcionamiento permanente del equipo.

Total documentación del equipo.

Permite grabar memorias EPROM comerciales 2716-27512 con algoritmos normal e inteligente de programación. Además los voltajes de programación son seleccionables por software.

Bajo costo del producto.

Puede ser operado desde cualquier computador equipado con interface RS232.

Permite programación de microcontroladores de la familia MCS48 y de la familia MCS51. A diferencia de algunos

programadores.

Fácil de instalar.

El tamaño no requiere de mucho espacio para su instalación.

Programa de uso para ser instalado en cualquier PC compatible.

Al disponer de tener potenciómetros internos el operador puede hacer de su programador un equipo versátil pudiendo acomodar voltajes de acuerdo a los requerimientos.

Conexión a 110V ac/60hz

Voltajes de programación seleccionables por software.

Opción de grabar el bit de seguridad en caso de microcontroladores de la familia MCS51.

Realiza todas las operaciones de un programador. Estas son: leer, programar, chequear, verificar, editar bytes, grabar en archivo, mostrar contenido y cargar archivo en buffer de memoria.

Acepta formato INTEL del programa a ser grabado.

Peso 2 kg

Dimensiones 23cm de largo x 18cm de ancho x 9cm de altura

Potencia 9.3W

Diferentes programadores que ofrece el mercado:

1 socket 16k hasta 512k **101400** ofrecido por JAMECO en el catálogo 394 de agosto-octubre 94.

E(E)PROM Programmer

Este programador permite programar EPROM's y EEPROM's presentado tres algoritmos de programación

Tipo de formatos aceptados INTEL HEX, MOTOROLA S HEX, TEKTRONIX HEX Y BINARIO.
Incluye software. manual y tarjeta para ser colocada en el computador. Peso 1.5 libras 1 año de garantía, tamaño 7"largo, 5.5" ancho y 1.75" de alto.
Precio $129.95

Con respecto a este programador si bien es cierto el precio resulta ser menor; sin embargo, el nuestro permite adicionalmente grabar microcontroladores, lo cual le da una ventaja mayor a nuestro equipo. Las memorias que programa son tanto EPROM como EEPROM en este sentido tiene una capacidad mayor en el rango de memorias a programar. El nuestro graba solamente las EPROM. Si bien es cierto que presenta algunos formatos .HEX sin embargo el usuario al tener el archivo.asm puede utilizar programas existentes para que el formato dado sea el INTEL que requiere nuestro equipo. En este sentido no se tendría mayores problemas.

1 socket 16k hasta 8MB **78457** ofrecido por JAMECO en el catálogo 394 de agosto-octubre 94.
E(E)PROM Programmer
Este programador permite programar EPROM's, EEPROM's y memorias rápidas presentado tres algoritmos de programación
Tipo de formatos aceptados INTEL HEX, MOTOROLA S HEX, TEKTRONIX HEX Y BINARIO.
Incluye software y manual y tarjeta para ser colocada en el

computador. Peso 1.5 libras 1 año de garantía, tamaño 7"largo, 5.5" ancho y 1.75" de alto.
Programa todas las memorias desde 16k hasta 512k y la intel 27C011 de 1 MB, voltaje programable de 5 hasta 25 voltios.
Precio $209.95

La gran ventaja de este equipo es que permite grabar memorias de mayor capacidad hasta 8MB. Una desventaja es el tener que abrir el computador para instalar la tarjeta que viene con el equipo, no permite grabar microcontroladores y el precio resulta ser mayor. En cuanto a nuestro equipo el formato INTEL no resulta ser problema como se indicó anteriormente.

4 Socket 16k-8MB **78465** ofrecido por JAMECO en el catálogo 394 de agosto-octubre 94.
Graba memorias EPROM's, E(E)PROM's y memorias rápidas.
3 algoritmos normal inteligente y quick
Tipo de formatos aceptados INTEL HEX, MOTOROLA S HEX, TEKTRONIX HEX Y BINARIO.
Incluye software y manual y tarjeta para ser colocada en el computador. Peso 1.5 libras 1 año de garantía, tamaño 7"largo, 5.5" ancho y 1.75" de alto.
Precio $279.95
Muy similar a la anterior la diferencia es que presenta 4 sockets para programación paralela de memorias. El precio es mucho mayor y no permite grabar microcontroladores.

Shooter este programador es ofrecido por SPECIALIZED PRODUCTS COMPANY en el catálogo de 1993

Este es un programador de EPROM's posee una memoria RAM interna de 512k. Permite la programación de todas las EPROM's desde la 2716 hasta la 27512 incluyendo la NMOS, CMOS, y la HMOS EPROMs. Puede ser conectada al computador o a un terminal tonto para un control total incluyendo edición del contenido de RAM chequeo de blanco, etc. En el modo de funcionamiento solo puede directamente copiar y verificar EPROMs incluyendo CMOS y "A" versiones con la ayuda de 3 switches de control y LEDS indicadores del estado.

Trabaja con MS-DOS compatible, terminal tonto o puerto serial

Copia y verifica en estado de trabajar sola.

110Vac/60Hz

socket 28 ZIF

Un buffer de RAM DE 512k

Para programar microcontroladores es posible añadir un módulo que le permite grabar 8751 y 87C51.

3 libras

7"x4.5"x3"

$395.00 stand alone

módulo $125.00

Un total de $520

Este programador es muy similar en especificaciones al nuestro.

CAPITULO VI

CONCLUSIONES

El arreglo de transistores para la comunicación serial en lugar de un MAX232 permitió abaratar costos del equipo. En el diseño se evita la implementación de fuentes de +/-12 voltios puesto que estos voltajes son tomados del computador. La utilización de este arreglo para la comunicación no presentó ningún problema siendo recomendable su uso para aplicaciones de comunicación entre el computador y el microcontrolador.

Al necesitar mas lineas del microcontrolador no se lo expandió mediante el uso de retenedores y buffers que al ser manejados como memoria RAM externa permitan obtener las líneas adicionales que se necesitan; en lugar de eso se utilizó un 8255. Al utilizar este ampliador programable de puertos el sistema es dinámico puesto que el ampliador de puertos es programable y además es bidireccional, posibilidad que no se presentaría en el caso de utilizar retenedores.

El compartir las fuentes reguladas de voltaje de programación redujo los costos. Esto se logró gracias a que dependiendo del software se pueden ubicar los voltajes deseados para los distintos elementos en los pines correspondientes. Si bien es cierto, esto reduce costos sin embargo se debe tener mucho cuidado en colocar los elementos en el socket adecuado asi como el manejo de software. Esta opción resulta mucho mejor que el utilizar

reguladores de voltaje para cada elemento solución que encarecería costos en el equipo. Se necesita mínimo dos reguladores puesto que para programar el 8748 éste requiere de dos fuentes para la programación, para los otros elementes se puede utilizar el regulador que disponga de los voltajes de programación necesarios. La opción de utilizar el mismo regulador para voltaje de programación tanto para el MCS51 y para memorias resulta ser buena en el sentido de que la programación es una función excluyente: no se puede programar 2 elementos al mismo tiempo.

El utilizar la comunicación tipo serial entre el PC y el equipo resulta una opción más económica que el uso del puerto paralelo; sin embargo, lo que se pierde es velocidad. De acuerdo a los resultados obtenidos ha sido posible ver que el proceso de programación utilizando el puerto serial resulta una opción bastante buena y que la velocidad de comunicación no representa un problema puesto que el proceso en sí de programación requiere de retardos. Por lo tanto no se justifica el uso de comunicación paralela.

Para la comunicación vía serial no se utilizaron líneas de hand-shake, pues la aplicación es muy sencilla, en ese sentido la implementación de subrutinas de error en falla de comunicación permiten una solución mucho más fácil que el uso de líneas de hand-shake.

Siendo el programador un equipo que por las funciones que realiza tiene que trabajar con elementos externos cuyo

estado no se lo puede predecir, se convierte en un equipo que, al colocar un elemento quemado, es expuesto a daños. De esta manera un programador no puede ser concebido como cualquier otro equipo en el que los parámetros se encuentran dados y cuya confiabilidad de funcionamiento es prácticamente del 100%. El funcionamiento del programador depende de la condición en la que se encuentran los elementos a ser programados.

Como concepción global, el tener un equipo autónomo permite una fácil desconexión y transportación; en este sentido el equipo diseñado representa una opción más conveniente que el utilizar aquellos programadores cuyo funcionamiento se ve ligado a tarjetas las mismas que es necesario colocarlas al interior del computador. Para realizar la conexión del programador diseñado, no es necesario proceder a la apertura del mismo. Esta operación en muchos casos limita al operador a tener instalado el programador en un solo computador y cuyo funcionamiento se vea totalmente ligado al funcionamiento del PC.

Una opción que pudo haberse implementado en el desarrollo del equipo constituye el uso de un sólo zócalo de programación. En ese sentido se complicaría el diseño circuital teniendo que disponer de más líneas de control y además el programa de manejo del mismo resulta ser más complicado.

Al tener 8 tipos distintos de memoria a programar puede de entrada pensarse en el uso de 8 sockets distintos; sin

embargo, gracias a la opción de superposición para memorias, la utilización de 3 sockets resulta ser mejor alternativa.

La utilización de tres zocalos independientes resulta ser mejor en el sentido de menor complicación para desarrollo del equipo, asi como para mantenimiento, en el caso de que exista una falla.

El desarrollo local de un programador si bien es cierto a tomado su tiempo, sin embargo se tiene actualmente la garantia de mantenimiento y soporte técnico, ventajas que al traer un equipo del exterior, generalmente no pueden hacerse efectivas en nuestro pais.

La existencia en el mercado de potenciales compradores permite tener una compensación en el tiempo invertido para el desarrollo del equipo.

Por el costo, mantenimiento soporte técnico y garantia es recomendable la construcción del mismo.

BIBLIOGRAFIA

1 TAUB H. SCHILLING D., Electrónica digital integrada, Marcombo, España, 1980

2 BUITRON D. Folleto de analógico digitales, EPN, 1990

3 INTEL MEMORY COMPONENTS HANDBOOK, Intel Literature Sales, Santa Clara CA, 1986

4 INTEL MICROCONTROLERS HANDBOOK, Intel Literature Sales, Santa Clara CA, 1991

5 INTEL COMPONENT DATA CATALOG, Intel Literature Sales, Santa Clara CA, 1979

6 USER MANUAL, ADVANCED MICROCOMPUTER SYSTEMS PROM 2015 UNIVERSAL PROM PROGRAMMER 1987.